L'algorithmique au lycée entre développement de savoirs spécifiques et usage dans différents domaines mathématiques

Dominique Laval

Université Sorbonne Paris Cité

Thèse de doctorat

de l'Université Sorbonne Paris Cité

Préparée à l'Université Paris Diderot

Ecole doctorale

« Savoirs scientifiques : épistémologie, histoire des sciences, didactique des disciplines » (ED 400)

Laboratoire de didactique André Revuz

L'algorithmique au lycée entre développement de savoirs spécifiques et usage dans différents domaines mathématiques

Par Dominique LAVAL

Thèse de doctorat de Didactique des Mathématiques

Dirigée par Monsieur Alain KUZNIAK

Présentée et soutenue publiquement à Paris le 8 mars 2018

Présidente du jury : OUVRIER-BUFFET Cécile, Professeure des Universités, Université de Reims Champagne-Ardenne

Rapporteurs : BRONNER Alain, Professeur des Universités, Université de Montpellier

RICHARD Philippe R., Professeur des Universités, Université de Montréal (Canada)

Examinatrice : ARTIGUE Michèle, Professeure des Universités, Université Paris Diderot

Directeur de thèse : KUZNIAK Alain, Professeur des Universités, Université Paris Diderot

Titre :
L'algorithmique au lycée entre développement de savoirs spécifiques et usage dans différents domaines mathématiques

Résumé :
Les nouveaux programmes des lycées français, mis en place depuis la rentrée 2010, ont fixé des objectifs précis en matière d'algorithmique. En effet, l'algorithmique est un champ transversal du lycée qui doit permettre aux élèves d'apprendre à construire une démarche scientifique. Autour de cet enseignement, trois objectifs fondamentaux doivent émerger :

- **approfondir les bases de la logique et du raisonnement ;**
- **illustrer les concepts enseignés par l'utilisation d'outils informatiques ;**
- **développer chez les élèves l'esprit de créativité et d'initiative au travers de l'expérimentation.**

A la lecture des programmes du lycée (Seconde (Grade 10) et cycle terminal scientifique (Grades 11 et 12)), l'enseignement de l'algorithmique apparaît comme *outil* (au sens de Douady, 1986) pour donner sens à un certain nombre de notions étudiées. Comment dépasser ce stade pour que l'algorithmique devienne *objet* d'apprentissage (au sens de Douady, 1986) ?
Le travail de recherche se situe dans le cadre d'apprentissages de connaissances sur les algorithmes en mathématiques dans l'enseignement au niveau des classes de Seconde et du cycle terminal scientifique du lycée. L'étude et la construction d'algorithmes par les élèves sont situées dans un cadre plus général de raisonnement et de preuve, mais aussi de démarches de modélisation en mathématiques.
Il s'agit d'étudier l'effectivité de tels enseignements dans le cadre institutionnel français du point de vue des apprentissages effectivement réalisés par les élèves et des pratiques des enseignants, et d'en inférer des résultats plus généraux sur le raisonnement mathématique en *théorie élémentaire des nombres* et en analyse, ainsi que dans le cadre d'un enseignement des probabilités et des simulations d'une situation aléatoire, pour les classes du lycée.
Le travail entrepris privilégie la place occupée par les algorithmes dans l'enseignement des mathématiques et propose un cadre théorique tenant compte des cadres généraux de la didactique des mathématiques, en particulier les *Espaces de Travail Mathématique* (ETM) (Kuzniak, Richard, 2014) associés à des domaines mathématiques spécifiques.
Plus particulièrement, poursuivant la spécification d'un modèle *Espaces de Travail Algorithmique* (ETA) (Laval, 2014, 2016), nous précisons ce que peuvent être les plans épistémologique et cognitif dans ces espaces en mettant l'accent sur leurs interactions liées aux genèses sémiotique, instrumentale et discursive auxquelles ces plans donnent lieu. Nous étudions aussi quels espaces personnels peuvent se construire chez les élèves des différents niveaux scolaires du lycée, et comment ils articulent des connaissances sur les algorithmes et les domaines mathématiques scolaires.
Les modèles des ETM/ETA sont consacrés à l'analyse du travail mathématique dans des domaines mathématiques spécifiques avec, en particulier, des paradigmes guidant et orientant le travail des élèves.
De plus, partant du fait que peu d'études sur des tâches de modélisation ont été basées sur les modèles ETM/ETA, nous affinons certaines de nos analyses dans le cadre des ETM/ETA sur la base du cycle de modélisation proposé par Blum et Leiss (2005) en relation avec certains domaines spécifiques des mathématiques.
Pour cela, nous construisons plusieurs ingénieries didactiques mettant en place des expérimentations dans divers domaines mathématiques. Nous avons ainsi le domaine de la *théorie élémentaire des nombres*, où les élèves doivent énoncer une conjecture sur l'*algorithme de Kaprekar* pour des nombres entiers compris entre 000 et 999, puis construire deux types de preuves (non explicative et explicative) de cette conjecture. De même dans le domaine de l'analyse, nous avons deux expérimentations menées par les élèves sur l'*algorithme de dichotomie*. Une première dite « dichotomie discrète », où les élèves s'approprient un problème d'identification d'un nombre

entier « secret » dans un intervalle d'entiers donnés, par des essais successifs, afin d'élaborer une stratégie « gagnante » et « rapide » qui serait programmable dans un environnement numérique. Une deuxième dite « dichotomie continue », où les élèves doivent produire une preuve basée sur la dichotomie du *théorème des valeurs intermédiaires*. D'autre part, dans le domaine des probabilités et des simulations aléatoires, les élèves étudient les conséquences possibles sur la démographie d'un pays si celui-ci met en place une *politique des naissances* précise, en lien avec la loi géométrique tronquée.

Ces ingénieries sont expérimentées et analysées dans les trois niveaux du lycée français : Seconde et cycle terminal scientifique. Le travail comporte des outils d'analyse des tâches et des activités dans ces différents domaines. La méthodologie employée permet d'obtenir des données globales et d'observer finement les activités des élèves en classe et les pratiques des enseignants.

Mots clefs :
Mathématiques ; algorithme ; algorithmique ; ingénierie didactique ; Espaces de travail Algorithmique (ETA) ; Espaces de travail Mathématique Spécifique (ETMs) ; paradigmes algorithmiques ; théorie élémentaire des nombres ; analyse ; statistiques-probabilités ; modélisation ; preuves ; algorithme de « dichotomie discret » ; algorithme de « dichotomie continue » ; algorithme de Kaprekar ; politique des naissances

Title:
Algorithmics in High school between development of specific knowledge and use in various mathematical domains

Abstract:
The new programs of French High schools, since 2010, precise objectives in terms of algorithmics. According to High schools curricula, algorithmics teaching appears as a tool (in the sense of Douady, 1986) to give meaning to some studied notions. How to go beyond this level so that algorithmic becomes an object of learning (in the sense of Douady, 1986)?
This research work is in the framework of learning of mathematical knowledge in algorithmics at the level of Grade 10 and Scientific Terminal Cycle (Grades 11 and 12) of the French high school. The study and construction of algorithms by students are located in a more general framework of reasoning and proof, but also mathematical modelling.
We build three didactic engineerings in High school to study the work of student and to watch teacher's practices. Our aim is to infer more general results on mathematical reasoning in some specific mathematical domains.
The research work favours algorithms' place in mathematics teaching. We propose a theoretical framework taking into account the general frameworks of mathematics didactics, in particular the Mathematical Working Spaces (MWS) (Kuzniak, Richard, 2014) associated with specific mathematical domains.
Following the specification of an Algorithmics Working Spaces (AWS) (Laval, 2014, 2016) we specify the possibilities of the epistemological and cognitive plans inside of these spaces increasing their interactions with their semiotician, instrumental and discursive geneses. We also study which personal spaces can be built for students at different levels of High school system, and how they articulate knowledge about algorithms and school mathematical domains.
The models of MWS/AWS aim at analysing of mathematical work in specific mathematical domains, with in particular, paradigms guiding and directing the work of the student.
Moreover, since few studies of modelling tasks have been built on MWS/AWS models, we refined some our analyses in the framework MWS/AWS basing on the modelling cycle proposed by Blum & Leiss (2005) in relation to some specific mathematical domains.

We build several didactic engineerings that we experimented in various mathematical domains: (1) elementary number theory; (2) mathematical analysis; (3) probabilities and random simulations.
These didactic engineerings are experimented and analysed in various French High school's grade: Grade 10 and Scientific Terminal Cycle (Grades 11 and 12). Our research work includes tools for analysing tasks and activities in different mathematical domains. The methodology obtains aggregated global data and finely observes students' activities in classroom and teacher's practices.

Keywords:
Mathematics; algorithm; algorithmics; didactic engineering; Algorithmics Working Space (AWS); Specific Mathematical Working Space (MWS_s); algorithmics paradigms; elementary theory of numbers; analysis; statistics – probabilities; modelling; proof; dichotomy method; Kaprekar's routine; truncated geometric distribution

A ma femme, mes enfants, mes petits-enfants, mes parents et ma grand-mère

Remerciements

Au cours de ces six années qui viennent de s'écouler, j'ai pu vivre une expérience riche et intense accompagnée de moments de doute, de joie, de questionnements mais surtout de développement intellectuel qui m'a permis de sortir de mes propres pratiques enseignantes dans un lycée de la région parisienne. C'est ainsi, que je remercie avec une profonde émotion tous ceux qui, de près ou de loin, m'ont accompagné tout au long de cette aventure.

Je tiens tout d'abord à remercier Jean-baptiste Lagrange qui aura été mon directeur de thèse pendant les cinq premières années. Grace à ta patience, ton soutien, tes précieux conseils, j'ai pu progresser dans mes réflexions et dans l'organisation de mon travail de recherche. Cela aura été un honneur et une grande satisfaction de t'avoir comme conseiller tout au long de ces années, même si certaines fois les échanges pouvaient être un peu délicats. Merci aussi d'avoir accepté d'être invité lors de ma soutenance.

Je remercie Alain Kuzniak, qui a accepté de remplacer Jean-baptiste, comme directeur de thèse, pendant la sixième et dernière année de mon travail de thèse consacrée à la mise en forme de l'écriture finale du manuscrit de ma thèse. Un immense merci Alain pour tes conseils, tes encouragements, tes remarques, cela m'aura été d'un précieux secours. Tu as su me guider pour finir le travail que j'avais entrepris avec Jean-baptiste et me donner confiance en particulier sur mes écrits. Ta disponibilité et ton écoute m'aura été d'une très grande aide et permis de retrouver confiance en moi-même.

Je remercie Philippe R. Richard et Alain Bronner pour avoir accepté la charge de rapporteur de cette thèse. Je les remercie pour l'intérêt et les conseils qu'ils m'ont donné suite à la lecture de mon manuscrit. Les observations et remarques qu'ils m'ont indiquées dans leurs rapports respectifs m'encouragent de donner une suite à ce travail de recherche dans les domaines de la didactique des mathématiques en lien avec la mise en place récente d'un enseignement de l'algorithmique dans les classes du secondaire, voire du primaire. Leurs encouragements et leurs remarques m'aideront à clarifier mes idées sur la pensée algorithmique, mais aussi la pensée mathématique.

Je remercie Michèle Artigue et Cécile Ouvrier-Buffet qui m'ont fait l'honneur d'être membres de mon jury de thèse.

Mes remerciements vont aussi à Aline Robert, qui avec Michèle Artigue, lors de mon Master de Didactique des Mathématiques, m'ont permis d'avoir une première approche de la recherche dans le domaine de la didactique des mathématiques. Votre confiance et votre soutien auront été le déclencheur de ce goût à la recherche.

Je remercie aussi Brigitte Grugeon-Allys, Maha Abboud, Laurent Vivier et Fabrice Vandebrouck pour leurs encouragements et leurs conseils à certains moments clés de mon travail de doctorant.

Je remercie aussi l'ensemble des membres du LDAR que j'ai eu l'occasion de côtoyer durant ces années de Master et de Thèse. Je remercie plus particulièrement les différents membres du groupe ETM.

Merci aussi à Sandrine, Evelyne, Laetitia, Jérôme pour leur gentillesse et leur travail.

Merci à l'ensemble des « jeunes » chercheurs du laboratoire pour les échanges, les écoutes, les partages de ressources, les encouragements,…

Merci à Marc, Kalalin, Inès, Ratha, Zakaria, Romain,… collègues doctorants avec qui j'ai partagé pendant toutes ces années de thèse le bureau 5002.

Merci aux « jeunes » chercheurs de l'ARDM que je croisais lors de séminaires en particulier celui des WEJCH.

Merci à mon colocataire de Faro, Simon Modeste, avec qui j'ai pu échanger tout au début de ma thèse sur l'algorithmique. Merci à Julia Pilet et Nicolas Pelay avec qui j'ai pris plaisir d'échanger sur des thèmes divers autour de la didactique et des mathématiques.

Merci à Assia Nechache et Edith Petitfour avec qui j'ai préparé le WEJCH 2014.

Merci à Madame Natta, Proviseure, et à mes anciens collègues de mathématiques du lycée Galois à Sartrouville et plus particulièrement Daniel T. avec qui nous avons eu de longs échanges sur mes travaux de recherches. Merci aussi à Natalie et Pierre L. professeurs de mathématiques au lycée Thibault de Champagne à Provins. Merci à Sophie R. professeure de mathématiques au lycée International de Saint-Germain-en-Laye. Merci à Paul-Emile C. professeur de mathématiques au lycée Richelieu à Rueil-Malmaison. Vous m'avez très

gentiment ouvert les portes de vos classes afin que je puisse procéder aux différentes ingénieries didactiques que j'avais élaborées dans le cadre de ma thèse.

Merci à Marie-Noëlle Guy avec laquelle j'ai mené une des expérimentations qui m'a servi de base pour l'élaboration de l'ingénierie *Kaprekar*.

Merci à l'ensemble de mes collègues de l'ESPE-UCP, en particulier Hélène, Marie-Noëlle et Denis, collègues de mathématiques et à Madame Laroque responsable pédagogique du site de Saint-Germain-en-Laye pour leur soutien moral.

Ich danke Prof. Dr. Gabriele Kaiser. In den Arbeitsgruppen der internationalen Symposien (PME und CERME), die Sie geleitet haben, haben Sie mir während meiner wissenschaftlichen Präsentationen auf Englisch sehr geholfen.

Je remercie ma famille, en particulier ma femme Marion qui a été d'un indéfectible soutien moral lors de ces années de thèse. Je remercie aussi chacun de mes enfants et leurs conjoints qui m'ont encouragé à mener à bien mon travail de recherche. Je remercie mes parents pour m'avoir donné le goût d'apprendre, à chercher et à transmettre. Danke an meine Schwiegereltern für ihre Geduld, wenn habe ich in der Stube meine Bücher und Papiere über den ganzen Tisch gestreut habe.

Table des matières

REMERCIEMENTS 6

INTRODUCTION 21

1. LES ORIGINES DE NOTRE TRAVAIL DE RECHERCHE 21
2. QUELLES APPROCHES POSSIBLES D'UN ENSEIGNEMENT DES ALGORITHMES EN COURS DE MATHEMATIQUES ? 26
3. DEROULEMENT ET PLAN DE LA THESE 31

PREMIERE PARTIE : CONSTITUTION D'UN TERRAIN DE RECHERCHE AUTOUR D'UN ENSEIGNEMENT DE L'ALGORITHMIQUE EN CLASSE DE MATHEMATIQUES AU LYCEE 36

CHAPITRE 1 : DES DEFINITIONS – LES PROGRAMMES SCOLAIRES – LES QUESTIONS INITIALES. 38

1. DES DEFINITIONS CLES 38
1.1 UN ALGORITHME 38
1.2 UN PROGRAMME INFORMATIQUE 40
1.3 LE CONCEPT DE LANGAGE DE PROGRAMMATION 41
1.4 L'ALGORITHMIQUE 42
2. LES OBJECTIFS DES PROGRAMMES DE MATHEMATIQUES : SECONDE GENERALE ET CLASSES DU CYCLE TERMINAL DE LA SERIE SCIENTIFIQUE 43
3.1 LA CLASSE DE SECONDE. 43
3.2 LES CLASSES DU CYCLE TERMINAL DE LA SERIE SCIENTIFIQUE. 48
3. ETAT DE L'ART SUR LES RECHERCHES EN DIDACTIQUE DES MATHEMATIQUES DANS LES DOMAINES DE LA PROGRAMMATION ET DE L'ALGORITHMIQUE 52
3.1 UN RETOUR SUR LA RECHERCHE EN DIDACTIQUE AUTOUR DE LA PROGRAMMATION A LA FIN DES ANNEES 80 52
3.2 UNE PREMIERE APPROCHE DU FONCTIONNEMENT DES MEMOIRES ET DES VARIABLES DANS LE CADRE DE L'ALGORITHMIQUE ET DE LA PROGRAMMATION 53
4. UNE VISION DE LA RECHERCHE EN DIDACTIQUE SUR UNE INTRODUCTION DE L'ALGORITHMIQUE DANS L'ENSEIGNEMENT DES MATHEMATIQUES AU LYCEE 53
5. LES ASPECTS « OUTIL » ET « OBJET » DES ALGORITHMES 56
6. PLACE ET ROLE DES ALGORITHMES DANS LES PROGRAMMES ET LES MANUELS DEPUIS 2010 57
7. L'ALGORITHMIQUE ET SON MODE DE PENSEE SPECIFIQUE 58
8. UN POINT SUR LES TRAVAUX EN DIDACTIQUE DES MATHEMATIQUES AUTOUR DE L'INFORMATIQUE ET DE L'ALGORITHMIQUE 59
8.1 UN CADRE THEORIQUE BASE SUR UNE MODELISATION THEORIQUE PAR LES CONCEPTIONS ET LA DIALECTIQUE OUTIL-OBJET POUR CARACTERISER L'OBJET « ALGORITHME » 59
8.2 LE CADRE DE LA TRANSPOSITION DIDACTIQUE POUR ETUDIER LES APPORTS DE L'ALGORITHMIQUE A L'ENSEIGNEMENT DES MATHEMATIQUES AU LYCEE 61
8.3 PRESENTATION DES ETUDES DE RECHERCHE FAITES EN PSYCHOLOGIE DE LA PROGRAMMATION 63
9. CONCLUSION DU CHAPITRE 1 : CONSEQUENCES DES OBSERVATIONS FAITES SUR LES DIFFICULTES OBSERVEES CHEZ L'ELEVE DEBUTANT EN INFORMATIQUE 67
9.1 L'ALGORITHME COMME « OBJET » D'APPRENTISSAGE EN MATHEMATIQUES 67
9.2 VERS L'ELABORATION D'UN NOUVEAU CADRE THEORIQUE 68

CHAPITRE 2 : CONSTRUCTION D'UN CADRE THEORIQUE 70

1. DEVELOPPEMENT THEORIQUE ET QUESTION 70
2. VERS UN NOUVEAU CADRE THEORIQUE. POURQUOI ? 72
3. LES ESPACES DE TRAVAIL MATHEMATIQUE 74
3.1 LES ESPACES DE TRAVAIL GEOMETRIQUE ET LES PARADIGMES GEOMETRIQUES 74
3.2 LES ESPACES DE TRAVAIL MATHEMATIQUE 77
3.3 LES ESPACES DE TRAVAIL MATHEMATIQUE SPECIFIQUES 85
4. LES ESPACES DE TRAVAIL ALGORITHMIQUE (ETA) 86
4.1 LE PLAN EPISTEMOLOGIQUE ET SES COMPOSANTES 86
4.2 LE PLAN COGNITIF 89
4.3 LES PARADIGMES ALGORITHMIQUES 91
5. ESPACES DE TRAVAIL MATHEMATIQUE DE REFERENCE 92
6. UNE APPROCHE « ALGORITHMIQUE » DU CONCEPT DE MODELISATION SELON BLUM ET LEISS 93
7. CONCLUSION DU CHAPITRE 2 : UN RETOUR SUR NOS CHOIX DE CADRES THEORIQUES 94

CHAPITRE 3 : PROBLEMATIQUE ET METHODOLOGIE DE RECHERCHE. 97

1. SPECIFICITES DES ETA ET ETM PERMETTANT D'AFFINER L'ETUDE DU TRAVAIL DE L'ELEVE DANS CERTAINS DOMAINES MATHEMATIQUES 98
1.1 QUELLES INTERACTIONS ENTRE ETA ET ETM DANS DIVERS DOMAINES DES MATHEMATIQUES SCOLAIRES ? 98
1.2 HYPOTHESES DE TRAVAIL SUR LES ETA-ETM IDOINES 101
2. METHODOLOGIE POUR LA CONCEPTION ET LA MISE EN ŒUVRE D'INGENIERIES DIDACTIQUES 103
2.1 NOTRE METHODOLOGIE 103
2.2 NOS INGENIERIES 104
3. CONCLUSION DU CHAPITRE 3 : DES ETA-ETM IDOINES ET PERSONNELS 111

CONCLUSION DE LA PARTIE 1. 113

PARTIE 2 : LES INGENIERIES DIDACTIQUES 117

CHAPITRE 1 : DES INGENIERIES DIDACTIQUES 118

1. LES DIFFERENTES PHASES DE LA METHODOLOGIE 118
2. NOS CHOIX GENERAUX 119
3. LES INGENIERIES DIDACTIQUES : LES DOMAINES, LES TACHES ET LES PHASES 119
3.1 LES DOMAINES MATHEMATIQUES SPECIFIQUES 119
3.2 LES TACHES 122
3.3 LES PHASES 123
4. LES DEUX PRINCIPAUX ENVIRONNEMENTS NUMERIQUES UTILISES DANS LE CADRE DE NOS INGENIERIES 125
4.1 PRESENTATION DU LOGICIEL ALGOBOX 125
4.2 PRESENTATION DU LOGICIEL LARP 126
5. CONCLUSION DU CHAPITRE 1 127

CHAPITRE 2 : ARTICULATION *ESPACE DE TRAVAIL ALGORITHMIQUE* ET *ESPACE DE TRAVAIL MATHEMATIQUE SPECIFIQUE* A LA *THEORIE ELEMENTAIRE DES NOMBRES* 128

1. POURQUOI PARLER DE *THEORIE ELEMENTAIRE DES NOMBRES* ET NON D'*ARITHMETIQUE* ? 128
1.1 LA *THEORIE DES NOMBRES* 128
1.2 L'ARITHMETIQUE 129
1.3 LA *THEORIE ELEMENTAIRE DES NOMBRES* 129

2. CHOIX PRINCIPAUX POUR UNE INGENIERIE DANS LE DOMAINE DE LA *THEORIE ELEMENTAIRE DES NOMBRES* 129
3. PRESENTATION DE L'ALGORITHME DE KAPREKAR 130
3.1 DESCRIPTION DU PROCESSUS DE CALCUL 130
3.2 ÉTUDE SELON LE NOMBRE ENTIER INITIAL CHOISI - CHOIX DU NOMBRE DE CHIFFRES DU NOMBRE ENTIER INITIAL POUR NOTRE INGENIERIE 131
3.3 UN ALGORITHME QUI TROUVE SA PLACE A TOUS LES NIVEAUX SCOLAIRES DE L'ELEMENTAIRE AU SUPERIEUR – LE CHOIX DU NIVEAU 132
4. ANALYSE DES PROGRAMMES ET DE DEUX MANUELS EN LIEN AVEC L'*ALGORITHME DE KAPREKAR* 135
4.1 LE PROGRAMME SUR « NOMBRES ET CALCULS » EN CLASSE TROISIEME..... 135
4.2 LE PROGRAMME SUR L'ARITHMETIQUE EN TERMINALE SCIENTIFIQUE ENSEIGNEMENT DE SPECIALITE 139
4.3 ETUDE DE DEUX MANUELS SCOLAIRES DE TERMINALE PROPOSANT DES TACHES EN LIEN AVEC L'ALGORITHME DE KAPREKAR 142
4.4 QUE RETENIR DES ANALYSES DES PROGRAMMES EN LIEN AVEC LE DOMAINE DE L'ARITHMETIQUE ET DES DEUX MANUELS PROPOSANT DES TACHES SUR L'*ALGORITHME DE KAPREKAR*, VOIRE DE L'*ALGORITHME D'EUCLIDE*, DANS LA PERSPECTIVE DES ANALYSES CONCERNANT NOTRE INGENIERIE, EN FONCTION DE NOTRE CADRE THEORIQUE ? 156
5. L'INGENIERIE.....156
5.1 L'« INGENIERIE-GUY »..... 157
5.2 NOTRE INGENIERIE 159
6. CONCLUSION DU CHAPITRE 2 210

CHAPITRE 3 : VERS UNE INGENIERIE DIDACTIQUE AUTOUR DE L'*ALGORITHME DE DICHOTOMIE*..212

1. INTRODUCTION 212
2. ANALYSE DES PROGRAMMES DE LA SECONDE ET DES CLASSES DU CYCLE TERMINAL SCIENTIFIQUE EN LIEN AVEC NOTRE RECHERCHE.....214
2.1 LES FONCTIONS..... 214
2.2 LES SUITES 216
2.3 L'ALGORITHMIQUE DANS LE DOMAINE DE L'ANALYSE 216
2.4 VERS UNE PREUVE ALGORITHMIQUE DU *THEOREME DES VALEURS INTERMEDIAIRES* EN TERMINALE SCIENTIFIQUE 219
3. ETUDE DE MANUELS SCOLAIRES ET D'AUTRES RESSOURCES.....219
3.1 AU NIVEAU DE LA SECONDE 220
3.2 AU NIVEAU DE LA PREMIERE SCIENTIFIQUE..... 245
3.3 AU NIVEAU DE LA TERMINALE SCIENTIFIQUE..... 254
4. SYNTHESE SUR L'ANALYSE DES PROGRAMMES ET L'ETUDE DE MANUELS ET DOCUMENTS RESSOURCES DE SECONDE ET DU CYCLE SCIENTIFIQUE TERMINAL.....269
4.1 AU NIVEAU DE LA SECONDE 269
4.2 AU NIVEAU DE LA PREMIERE SCIENTIFIQUE..... 270
4.3 AU NIVEAU DE LA TERMINALE SCIENTIFIQUE..... 272
4.4 SYNTHESE GLOBALE 274
5. NOS MOTIVATIONS.....275
6. CONCLUSION DU CHAPITRE 3 : LA STRUCTURE DE L'INGENIERIE 276

CHAPITRE 4 : UNE INGENIERIE ARTICULANT ETA ET ETM SPECIFIQUE (PARTIE 1) : *ALGORITHME DE DICHOTOMIE* « DISCRETE » – UNE STRATEGIE « RAPIDE » ET « GAGNANTE » 278

1. NOS OBJECTIFS ET NOS CHOIX.....278
1.1 LES OBJECTIFS..... 278
1.2 LES CHOIX..... 279
2. LES PHASES.....290

2.1 PHASE 1 (ALGORITHME DU JOUEUR A – CONCEPTION GENERALE)..... 290
2.2 PHASE 2 (ALGORITHME DE JOUEUR B – CONCEPTION GENERALE ET ECRITURE D'UN ORGANIGRAMME PAPIER-CRAYON) 291
2.3 PHASE 3 (ALGORITHME DE JOUEUR B – FORMALISATION SOUS FORMES « TEXTUELLE » ET « SPATIALE » – IMPLEMENTATION DANS UN ENVIRONNEMENT NUMERIQUE DE L'ALGORITHME) 292
3. ANALYSE *A PRIORI* 292
3.1 PHASE 1 (ALGORITHME DU JOUEUR A – CONCEPTION GENERALE)..... 292
3.2 PHASE 2 (ALGORITHME DE JOUEUR B – CONCEPTION GENERALE ET ECRITURE D'UN ORGANIGRAMME « PAPIER-CRAYON ») 293
3.3 PHASE 3 (ALGORITHME DE JOUEUR B – FORMALISATION SOUS FORMES « TEXTUELLE » ET « SPATIALE » – IMPLEMENTATION DE L'ALGORITHME DANS L'ENVIRONNEMENT NUMERIQUE)..... 294
4. MISE EN PLACE DE L'EXPERIMENTATION, DEROULEMENT ET ANALYSE *A POSTERIORI* 295
4.1 LES CLASSES, LES SUPPORTS D'OBSERVATIONS..... 295
4.2 PHASE 1 (ALGORITHME DU JOUEUR A – CONCEPTION GENERALE. DUREE : 20 MINUTES EN SECONDE – 15 MINUTES EN PREMIERE ET TERMINALE SCIENTIFIQUE)..... 296
4.3 PHASE 2 (ALGORITHME DE JOUEUR B – CONCEPTION GENERALE ET ECRITURE D'UN ORGANIGRAMME PAPIER-CRAYON. DUREE : 70 MINUTES EN SECONDE – 40 MINUTES EN PREMIERE ET TERMINALE SCIENTIFIQUE) 307
4.4 PHASE 3 (ALGORITHME DE JOUEUR B – FORMALISATION SOUS FORMES « TEXTUELLE » ET « SPATIALE » – IMPLEMENTATION DANS UN ENVIRONNEMENT NUMERIQUE DE L'ALGORITHME. DUREE : 30 MINUTES EN CLASSE QUI SE TERMINE A LA MAISON POUR LA CLASSE DE SECONDE – 40 MINUTES EN PREMIERE ET TERMINALE SCIENTIFIQUE) 325
5. SYNTHESE DE LA SOUS-INGENIERIE « ALGORITHME DE DICHOTOMIE DISCRETE »..... 345
5.1 LES STRUCTURES 345
5.2 LES VARIABLES..... 346
5.3 LES MODES D'EXPRESSION 347
6. LES PERSPECTIVES POUR LA SOUS-INGENIERIE « ALGORITHME DE DICHOTOMIE CONTINUE » : ARTICULATION ETA – ETM ; NIVEAUX DE PARADIGMES 349
6.1 PRESENTATION 349
6.2 LES TABLEAUX 349
7. CONCLUSION DU CHAPITRE 4 356

CHAPITRE 5 : UNE INGENIERIE ARTICULANT ETA ET ETM SPECIFIQUES (PARTIE2) : *ALGORITHME DE DICHOTOMIE* « CONTINU » – VERS UNE PREUVE DU TVI.....357

1. UNE SOUS-INGENIERIE.....357
1.1 PRESENTATION 357
1.2 LES PREMIERES PROBLEMATIQUES DE LA SOUS-INGENIERIE..... 357
1.3 ARRIERE-PLAN MATHEMATIQUE ET INSTITUTIONNEL 359
2. PREMIERE PARTIE DE LA SOUS-INGENIERIE « DICHOTOMIE CONTINUE » 361
2.1 LES OBJECTIFS ET LES ATTENTES 361
2.2 LES CHOIX POUR CHAQUE PHASE..... 362
2.3 LES TACHES ET LE MODE DE TRAVAIL..... 368
2.4 LES ANALYSES *A PRIORI* 372
2.5 MISE EN PLACE DE L'EXPERIMENTATION, DEROULEMENTS ET ANALYSES *A POSTERIORI*..... 377
2.6 SYNTHESE DE LA PREMIERE PARTIE DE LA SOUS-INGENIERIE « DICHOTOMIE CONTINUE » 437
3. DEUXIEME PARTIE DE LA SOUS-INGENIERIE « DICHOTOMIE CONTINUE ».....438
3.1 LES OBJECTIFS ET LES ATTENTES 439
3.2 LES CHOIX POUR CHACUNE DES DEUX PHASES..... 440
3.3 LES TACHES ET LE MODE DE TRAVAIL..... 442
3.4 LES ANALYSES *A PRIORI* 444
3.5 MISE EN PLACE DE L'EXPERIMENTATION, DEROULEMENT ET ANALYSES *A POSTERIORI* 449

3.6 SYNTHESE EN TERMES D'ANALYSE EN LIEN AVEC LES ESPACES DE TRAVAIL MATHEMATIQUE ET ALGORITHMIQUE DES PHASES 5 ET 6 DE LA SOUS-INGENIERIE « DICHOTOMIE CONTINUE » 462
4. CONCLUSION DU CHAPITRE 5 – SYNTHESE DE L'INGENIERIE COMPLETE « *ALGORITHME DE DICHOTOMIE* »..... 464

CHAPITRE 6 : UNE INGENIERIE ARTICULANT *ETA* ET *ETM* SPECIFIQUE : *UNE POLITIQUE DES NAISSANCES*..... 468

1. INTRODUCTION 469
2. APPROCHE INSTITUTIONNELLE 471
2.1 EN SECONDE 471
2.2 EN PREMIERE SCIENTIFIQUE 481
2.3 CONCLUSION 488
3. UNE INGENIERIE DIDACTIQUE DANS LE CHAMP DES PROBABILITES 488
3.1 NOS MOTIVATIONS 488
3.2 NOTRE INGENIERIE DIDACTIQUE 492
4. CONCLUSION DU CHAPITRE 6 541
4.1 L'ASPECT MODELISATION 542
4.2 DONNER DU SENS AUX OBJETS ALGORITHMIQUES ET INFORMATIQUES MIS EN JEU LORS DE CETTE EXPERIMENTATION 544
4.3 LES DIFFERENTES APPROCHES DE LA POLITIQUE DES NAISSANCES 545
4.4 UNE SUITE POSSIBLE A CETTE INGENIERIE 545

CONCLUSION DE LA PARTIE 2..... 546

1. DEUX INGENIERIES SUR ALGORITHMES ET APPRENTISSAGE DE LA PREUVE 546
1.1 L'ALGORITHME DE KAPREKAR 546
1.2 L'ALGORITHME DE DICHOTOMIE 548
2. UNE INGENIERIE SUR LA MODELISATION ET UNE DOUBLE APPROCHE : « FREQUENTISTE » ET « PROBABILISTE », DANS LE CADRE D'UNE *POLITIQUE DES NAISSANCES*..... 551
3. L'INTRODUCTION DE L'*ETA* : UN REEL BENEFICE POUR FAVORISER L'ACCES A DE NOUVELLES COMPETENCES MATHEMATIQUES. 552

CONCLUSION : QUELS ENSEIGNEMENTS TIRONS-NOUS DE CE TRAVAIL DE RECHERCHE ?..... 554

1. *POURQUOI UN NOUVEAU CADRE THEORIQUE ?*..... 555
2. UN TRAVAIL SPECIFIQUE SUR LA *THEORIE ELEMENTAIRE DES NOMBRES* 560
3. DES TRAVAUX SPECIFIQUES AU DOMAINE D'ANALYSE 561
4. UN TRAVAIL SPECIFIQUE A UNE SIMULATION ALEATOIRE : *UNE POLITIQUE DES NAISSANCES*..... 564
5. LES INGENIERIES SUR LA DICHOTOMIE ET LA SIMULATION SONT PRATIQUEES DANS LES MEMES CLASSES. QUELLES SONT LES CONSEQUENCES DE CE CHOIX ? QUELS ENSEIGNEMENTS EN TIRONS-NOUS DE CE CHOIX ? POUR LES ELEVES ? POUR LES PRATIQUES ENSEIGNANTES ? 566
6. PENSEE MATHEMATIQUE / PENSEE ALGORITHMIQUE 568
7. LES LIMITES OBSERVEES AU COURS DE NOTRE RECHERCHE – QUELLES PERSPECTIVES POUVONS-NOUS ENVISAGER A CE TRAVAIL DE RECHERCHE ? 571
7.1 LES LIMITES OBSERVEES AU COURS DE NOTRE RECHERCHE 571
7.2 QUELLES PERSPECTIVES POUVONS-NOUS ENVISAGER A CE TRAVAIL DE RECHERCHE ? 574
8. CONCLUSION 576

BIBLIOGRAPHIE 577

PROGRAMMES SCOLAIRES.....585

DOCUMENTS RESSOURCES 586

MANUELS SCOLAIRES 587

ANNEXES 588

ANNEXE 1 – ECRIRE UN ALGORITHME AU FORMAT SPATIAL : LES ORGANIGRAMMES (DOCUMENT DISTRIBUE AUX ELEVES DES CLASSES AYANT PARTICIPE AU MOINS A UNE DES INGENIERIES) 589

LES FIGURES USUELLES UTILISEES DANS LA CONSTRUCTION D'ORGANIGRAMMES 589
TYPES D'ORGANIGRAMMES UTILISABLES POUR REPRESENTER LES STRUCTURES ETUDIEES DANS LES CLASSES DE LYCEE 591

ANNEXE 2 – ARTICLE SUR LE RUBIK'S CUBE (EXTRAIT DE LA REVUE « POUR LA SCIENCE N° 400 – FEVRIER 2011 ») 594

ANNEXE 3 – ARTICLE « ELEMENTS POUR LA FORMATION INITIALE DES PROFESSEURS D'ECOLE A PARTIR DE L'ALGORITHME DE KAPREKAR » DE J.-C. RAUSCHER (EXTRAIT DES ACTES DU XXXIVEME COLLOQUE : *EXPERIMENTATION ET MODELISATION DANS L'ENSEIGNEMENT SCIENTIFIQUE : QUELLES MATHEMATIQUES A L'ECOLE ?* - TROYES 2007).....600

Table des illustrations

FIGURE 1 (EXTRAIT DU PROGRAMME DE 2NDE : BOUCLE ET ITERATEUR, INSTRUCTION CONDITIONNELLE) 44
FIGURE 2 (EXTRAIT DU PROGRAMME DE 2NDE : RESOLUTION GRAPHIQUE ET ALGEBRIQUE D'EQUATIONS) 46
FIGURE 3 (EXTRAIT DU PROGRAMME DE 2NDE : PROPRIETES DES FIGURES PLANES) 46
FIGURE 4 (EXTRAIT DU PROGRAMME DE 2NDE : ECHANTILLONNAGE) 47
FIGURE 5 (SCHEMA DES TRANSPOSITIONS DIDACTIQUE ET INFORMATIQUE – CHEVALLARD (1982) & BALACHEFF (1994)) 61
FIGURE 6 (RESOLUTIONS MATHEMATIQUE, ALGORITHMIQUE ET INFORMATIQUE – BRIANT & BRONNER (2015)) 62
FIGURE 7 (LES PLANS EPISTEMOLOGIQUE ET COGNITIF DES ETG) 76
FIGURE 8 (PLAN EPISTEMOLOGIQUE DES ETM) 80
FIGURE 9 (PLAN COGNITIF DES ETM) 82
FIGURE 10 (LES 3 PLANS D'INTERACTIONS IDENTIFIES PAR LES GENESES MISES EN ŒUVRE DANS LES ETM) 83
FIGURE 11 (PLANS EPISTEMOLOGIQUE ET COGNITIF ET LES GENESES DANS LE CAS DES ETM) 83
FIGURE 12 (PLAN EPISTEMOLOGIQUE DES ETA) 86
FIGURE 13 (PLAN COGNITIF DES ETA) 86
FIGURE 14 (PLAN EPISTEMOLOGIQUE DES ETA ASSOCIE A L'ALGORITHME DE KAPRKAR) 88
FIGURE 15 (PLAN EPISTEMOLOGIQUE DES ETA ASSOCIE A LA « DICHOTOMIE DISCRETE ») 89
FIGURE 16 (« THE MODELING CYCLE ACCORDING TO BLUM AND LEIß (2006) ») 94
FIGURE 17 (ETM/ETA ASSOCIES AU CYLCE DE MODELISATION DE BLUM ET LEISS (2005)) 95
FIGURE 18 (UN EXEMPLE DE CYCLE DE MODELISATION : « DICHOTOMIE DISCETE » (LAVAL, 2016)) 96
FIGURE 19 (REPRESENTATION GRAPHIQUE D'UNE FONCTION CONTINUE F SUR [A ; B] AVEC F(A)×F(B)<0) 108
FIGURE 20 (ECRAN ALGOBOX) 125
FIGURE 21 (ECRAN LARP) 127
FIGURE 22 (KAPREKAR – EXTRAIT DU 34E COLLOQUE COPIRELEM) 133
FIGURE 23 (EXTRAIT DU PROGRAMME OBLIGATOIRE ET SPECIALITE DE LA TERMINALE SCIENTIFIQUE) 140
FIGURE 24 (EXTRAIT D'UN EXERCICE DU MANUEL HYPERBOLE (ED. 2011) – KAPREKAR) 143
FIGURE 25 (EXTRAIT D'UN EXERCICE DU MANUEL HYPERBOLE – ALGORITHME D'EUCLIDE) 148
FIGURE 26 (EXTRAIT D'UN TP DU MANUEL MATH'X – LA RECETTE DE KAPREKAR) 151
FIGURE 27 (EXTRAIT D'UN TP DU MANUEL MATH'X – ALGORITHME D'EXTRACTION DES CHIFFRES D'UN ENTIER) 152
FIGURE 28 (EXTRAIT D'UN TP DU MANUEL MATH'X – LES CHIFFRES DES UNITES) 153
FIGURE 29 (COPIE D'ECRAN D'UNE FEUILLE DE CALCUL – TRIS DECROISSANT ET CROISSANT DE TROIS CHIFFRES) 154
FIGURE 30 (NOMBRE DE NŒUDS DU GRAPHE DU RUBIK'S CUBE AYANT UNE DISTANCE DONNEE A LA CONFIGURATION INITIALE) 169
FIGURES 31 (EXEMPLES D'ALGORITHMES OBTENUS A LA FIN DE L' « INGENIERIE-GUY » SUR KAPREKAR) 176
FIGURE 32 (CAS D'UN PREMIER ALGORITHME SOUS ALGOBOX PERMETTANT DE DETERMINER UNE PREUVE PAR EXHAUSTION DES CAS DE LA CONJECTURE KAPREKAR) 185
FIGURE 33 (EXTRAIT DES RESULTATS OBTENUS LORS DE L'EXECUTION SOUS ALGOBOX DE L'ALGORITHME DE LA FIGURE 32) 185
FIGURE 34 (SOUS ALGOBOX, UNE PREMIERE MODIFICATION DE L'ALGORITHME DE LA FIGURE 32) 186
FIGURE 35 (EXTRAIT DES RESULTATS OBTENUS LORS DE L'EXECUTION SOUS ALGOBOX DE L'ALGORITHME DE LA FIGURE 34) 186
FIGURE 36 (SOUS ALGOBOX, UNE MODIFICATION DE L'ALGORITHME DE LA FIGURE 34) 187
FIGURE 37 (EXTRAIT D'UN ALGORITHME KAPREKAR SOUS ALGOBOX) 188
FIGURE 38 (EXTRAIT D'UN ALGORITHME KAPREKAR SOUS ALGOBOX) 188
FIGURE 39 (SOUS ALGOBOX, UN ALGORITHME AUTOMATISE PERMETTANT UNE PREUVE PAR EXHAUSTION DES CAS DE LA CONJECTURE KAPREKAR, POUR TOUS LES ENTIERS NATURELS INFERIEURS A 999) 190
FIGURE 40 (SOUS ALGOBOX, UN ALGORITHME AUTOMATISE AVEC UTILISATION D'UNE LISTE, PERMETTANT UNE PREUVE PAR EXHAUSTION DES CAS DE LA CONJECTURE KAPREKAR, POUR TOUS LES ENTIERS NATURELS INFERIEURS A 999) 191

FIGURE 41 (SOUS ALGOBOX, UN ALGORITHME AUTOMATISE AVEC UTILISATION DE DEUX LISTES, PERMETTANT UNE PREUVE PAR EXHAUSTION DES CAS DE LA CONJECTURE KAPREKAR, POUR TOUS LES ENTIERS NATURELS INFERIEURS A 999) 192
FIGURE 42 (EXTRAIT D'UNE COPIE SUR KAPREKAR D'UN ELEVE DE TERMINALE SCIENTIFIQUE) 194
FIGURE 43 (EXTRAIT D'UNE COPIE SUR KAPREKAR D'UN ELEVE DE TERMINALE SCIENTIFIQUE) 195
FIGURE 44 (EXTRAIT D'UNE COPIE SUR KAPREKAR D'UN ELEVE DE TERMINALE SCIENTIFIQUE) 196
FIGURE 45 (EXTRAIT D'UNE COPIE SUR KAPREKAR D'UN ELEVE DE TERMINALE SCIENTIFIQUE) 197
FIGURE 46 (DIVERS ALGORITHMES D'ELEVES SUR KAPREKAR IMPLEMENTES DANS ALGOBOX) 202
FIGURE 47 (CAS D'UN ALGORITHME KAPREKAR « AUTOMATISE » SOUS ALGOBOX AVEC UTILISATION D'UNE LISTE DE SIX ENTIERS DE LA FORME 99 K, OU K EST UN ENTIER ALLANT DE 0 A 5) 203
FIGURE 48 (CAS D'UN ALGORITHME KAPREKAR « AUTOMATISE » SOUS ALGOBOX AVEC UNE UTILISATION DE DEUX LISTE POUR OBTENIR UNE PREUVE « EXPLICATIVE » DE LA CONJECTURE) 204
FIGURE 49 (CAS D'UN ALGORITHME KAPREKAR « AUTOMATISE » SOUS ALGOBOX AVEC UTILISATION D'UNE LISTE COMPLETEE AVANT L'ITERATION SANS REFERENCE AUX MULTIPLES DE 99) 205
FIGURE 50 (EXTRAIT D'UN EXERCICE DU MANUEL HYPERBOLE DE 2NDE : UN ALGORITHME DE DICHOTOMIE) 222
FIGURE 51 (EXTRAIT D'UN EXERCICE DU MANUEL HYPERBOLE DE 2NDE : UN ALGORITHME POUR ENCADRER) 222
FIGURE 52 (EXTRAIT D'UN EXERCICE DU MANUEL HYPERBOLE DE 2NDE : RESOLUTION D'UNE EQUATION) 223
FIGURE 53 (EXTRAIT D'UN EXERCICE DU MANUEL DE 2NDE MATHS-REPERES : METHODE DE DICHOTOMIE) 229
FIGURE 54 (EXTRAIT D'UN EXERCICE DU MANUEL DE 2NDE MATHS-REPERES : PAR DICHOTOMIE ET ALGOBOX) 230
FIGURE 55 (EXTRAIT D'UNE « ACTIVITE DE RECHERCHE » DU MANUEL DE 2NDE MATHS-REPERES : ECRITURE DECIMALE DU NOMBRE REEL 2) 232
FIGURE 56 (UN ALGORITHME ECRIT EN LANGAGE NATUREL) 240
FIGURE 57 (UN ALGORITHME DE « DICHOTOMIE CONTINUE ») 240
FIGURE 58 (UN ALGORITHME DE TRACE D'UN NOMBRE FINI DE POINTS DE LA REPRESENTATION GRAPHIQUE D'UNE FONCTION GENERIQUE) 242
FIGURE 59 (VALEUR APPROCHEE D'UNE SOLUTION D'UNE EQUATION DE LA FORME F(X) = 0 PAR LA METHODE DE DICHOTOMIE) 243
FIGURE 60 (L'ALGORITHME DE DICHOTOMIE (ROLAND, 2010) D'APRES UN DOCUMENT IREM) 244
FIGURE 61 (EXTRAIT D'UN EXERCICE DU MANUEL ODYSSEE DE 1^{ERE} SCIENTIFIQUE : RESOLUTION D'UNE EQUATION PAR DICHOTOMIE) 247
FIGURE 62 (EXTRAIT D'UN EXERCICE DU MANUEL ODYSSEE DE 1^{ERE} SCIENTIFIQUE : METHODE DE NEWTON) 249
FIGURE 63 (EXTRAIT D'UN EXERCICE DU MANUEL SYMBOLE DE 1^{ERE} SCIENTIFIQUE : PRINCIPE DE DICHOTOMIE) 251
FIGURE 64 (EXTRAIT D'UN EXERCICE DU MANUEL MATH'X DE TERMINALE SCIENTIFIQUE : COMPARAISON DE DEUX ALGORITHMES) 256
FIGURE 65 (EXTRAIT D'UNE SERIE D'EXERCICES DU MANUEL MATH'X DE TERMINALE SCIENTIFIQUE : EQUATIONS DE LA FORME F(X) = K) 257
FIGURE 66 (EXTRAIT D'UN TP DU MANUEL MATH'X DE TERMINALE SCIENTIFIQUE : L'ALGORITHME DE DICHOTOMIE AFIN D'APPROCHER L'IMAGE D'UN NOMBRE REEL X PAR LA FONCTION $x \mapsto ex$) 259
FIGURE 67 (EXTRAIT D'UN EXERCICE DU MANUEL MATH'X DE TERMINALE SCIENTIFIQUE : JUSTIFIER L'ALGORITHME DE DICHOTOMIE) 260
FIGURE 68 (EXTRAIT D'UN EXERCICE DU MANUEL DECLIC DE TERMINALE SCIENTIFIQUE : DETERMINATION DU POINT MOBILE M D'UNE PARABOLE P D'EQUATION $y = x^2$ TELLE QUE LA DISTANCE AM, OU A(1 ; 0), SOIT MINIMALE) 263
FIGURE 69 (EXTRAIT D'UN EXERCICE DU MANUEL DECLIC DE TERMINALE SCIENTIFIQUE : RESOLUTION D'UNE EQUATION A L'AIDE D'UN ALGORITHME) 266
FIGURE 70 (EXTRAIT D'UN EXERCICE DU MANUEL RADIAL DE TERMINALE SCIENTIFIQUE : UNE DEMONSTRATION DU THEOREME DES VALEURS INTERMEDIAIRES) 267
FIGURE 71 (EXTRAIT D'UN EXERCICE DU MANUEL RADIAL DE TERMINALE SCIENTIFIQUE : METHODE DE DICHOTOMIE) 268
FIGURE 72 (UN ORGANIGRAMME (FORMAT « SPATIAL » DE L'ALGORITHME DU JOUEUR A, CELUI QUI PENSE LE NOMBRE ENTIER « SECRET »)) 281
FIGURE 73 (UN ALGORITHME AU FORMAT « TEXTUEL » DU JOUEUR A) 283
FIGURE 74 (UN ALGORITHME AU FORMAT « SPATIAL » DU JOUEUR A) 283
FIGURE 75 (UN ALGORITHME AU FORMAT « TEXTUEL » DU JOUEUR A) 284
FIGURE 76 (UN ALGORITHME AU FORMAT « SPATIAL » DU JOUEUR A) 285

FIGURE 77 (UN ALGORITHME AU FORMAT « TEXTUEL » DU JOUEUR B, CELUI QUI PROPOSE DES REPONSES) 287
FIGURE 78 (DEUX ALGORITHMES POSSIBLES DU JOUEUR B AUX FORMATS « TEXTUEL » ET « SPATIAL ») 288
FIGURE 79 (DEUX ALGORITHMES POSSIBLES DU JOUEUR B AUX FORMATS « TEXTUEL » ET « SPATIAL ») 289
FIGURE 80 (UNE COPIE D'ELEVE OU CELUI-CI AFFECTE « NOMBRE ENTIER SECRET » A LA VARIABLE « NBRSEC ») 298
FIGURE 81 (EXTRAIT D'UN ORGANIGRAMME OBTENU PAR UN ELEVE N'AYANT PAS UTILISE LES MARQUEURS D'ITERATION « TANT QUE » OU « REPETER » POUR SIMULER LE JOUEUR B) 299
FIGURE 82 ((EXTRAIT D'UN ALGORITHME AU FORMAT « TEXTUEL » OBTENU PAR UN ELEVE N'AYANT PAS UTILISE LES MARQUEURS D'ITERATION « TANT QUE » OU « REPETER » POUR SIMULER LE JOUEUR B) 300
FIGURE 83 (ALGORITHME PROPOSE PAR UN ELEVE REPRESENTANT LES REPONSES DU JOUEUR A (MACHINE) EN FONCTION DES REPONSES DU JOUEUR B (UTILISATEUR)) 301
FIGURE 84 (ALGORITHME PROPOSE PAR UN ELEVE REPRESENTANT LES REPONSES DU JOUEUR A (MACHINE) EN FONCTION DES REPONSES DU JOUEUR B (UTILISATEUR)) 302
FIGURE 85 (ORGANIGRAMME PROPOSE PAR UN ELEVE REPRESENTANT LES REPONSES DU JOUEUR A (MACHINE) EN FONCTION DES REPONSES DU JOUEUR B (UTILISATEUR)) 302
FIGURE 86 (ALGORITHME DU JOUEUR A PROPOSE PAR UN ELEVE) 302
FIGURE 87 (DANS LE CADRE D'UN ALGORITHME DU JOUEUR A : CONSTRUCTION PROPOSEE PAR DEUX ELEVES D'UNE SERIE D'ALTERNATIVES EN NOMBRE INDETERMINE) 305
FIGURE 88 (UN ALGORITHME DU JOUEUR B PROPOSE PAR UN ELEVE DE 2NDE) 308
FIGURE 89 (UNE COPIE D'ELEVE AVEC REPRESENTATION GRAPHIQUE DU DECOUPAGE D'UN INTERVALLE ET ALGORITHME) 309
FIGURE 90 (UN ALGORITHME « TEXTUEL » DU JOUEUR B NE TENANT PAS COMPTE DU FAIT QUE LA MOYENNE DES BORNES D'UN INTERVALLE N'EST PAS SYSTEMATIQUEMENT UN NOMBRE ENTIER) 311
FIGURE 91 (STRUCTURE « REPETER… JUSQU'A », ALGORITHME DU JOUEUR B AU FORMAT « TEXTUEL ») 313
FIGURE 92 (ALGORITHME DU JOUEUR B AU FORMAT « SPATIAL ») 313
FIGURE 93 (UN ALGORITHME DU JOUEUR B NE METTANT PAS EN PLACE UNE BOUCLE DE TYPE « TANTQUE » OU « REPETER … JUSQU' ») 314
FIGURE 94 (UN ALGORITHME DU JOUEUR B DU TYPE « ARBRE DE PROBABILITE ») 314
FIGURE 95 (UN ALGORITHME AVEC ERREUR D'AFFECTATION DANS LE CORPS DE BOUCLE) 317
FIGURE 96 (ALGORITHME « SPATIAL » AVEC COMPTEUR ET STRATEGIE « DICHOTOMIE ») 318
FIGURE 97 (ALGORITHME « TEXTUEL » DU JOUER B AVEC COMPTEUR ET STRATEGIE DICHOTOMIE) 318
FIGURE 98 ALGORITHME « TEXTUEL » SANS STRATEGIE « DICHOTOMIE » ET SANS COMPTEUR 319
FIGURE 99 ALGORITHME « TEXTUEL » SANS STRATEGIE « DICHOTOMIE » AVEC COMPTEUR 319
FIGURE 100 (ALGORITHME SOUS ALGOBOX AVEC STRUCTURE CORRECTE ET IDENTIQUE, MAIS QUI NE FONCTIONNE PAS LORS DE SON EXECUTION) 329
FIGURE 101 (ALGORITHME SOUS ALGOBOX AVEC STRUCTURE CORRECTE ET IDENTIQUE, MAIS QUI NE FONCTIONNE PAS LORS DE SON EXECUTION) 329
FIGURE 102 (ALGORITHME SOUS ALGOBOX QUI NE FONCTIONNE PAS LORS DE SON EXECUTION : ERREUR SUR LA CHAINE DE CARACTERE) 329
FIGURE 103 (ALGORITHME SOUS ALGOBOX QUI NE FONCTIONNE PAS LORS DE SON EXECUTION) 329
FIGURE 104 (ALGORITHME SOUS ALGOBOX QUI NE FONCTIONNE PAS LORS DE SON EXECUTION : ERREUR SUR LE TYPE D'UNE VARIABLE) 330
FIGURE 105 (ALGORITHME SOUS ALGOBOX QUI NE FONCTIONNE PAS LORS DE SON EXECUTION : ERREUR SUR LE TYPE D'UNE VARIABLE) 330
FIGURE 106 (CAS D'UN ORGANIGRAMME COMPORTANT DES ERREURS SUR LA CONCEPTION ET LA LECTURE DE VARIABLES DE TYPE « CHAINE DE CARACTERE ») 331
FIGURE 107 (ORGANIGRAMME OU LES REPONSES PROPOSEES PAR LE JOUEUR B SONT « – 1 » POUR « TROP PETIT », « 1 » POUR « TROP GRAND » ET 0 POUR « EGAL ») 332
FIGURE 108 (ALGORITHMES AVEC COMMENTAIRE POUR DECRIRE LES « CHAINES DE CARACTERES » POUVANT SERVIR DE REPONSES A L'ORDINATEUR PAR L'UTILISATEUR JOUANT LE ROLE DU JOUEUR B) 333
FIGURE 109 (INSTRUCTION « LIRE » ET « AFFICHER » D'ALGOBOX) 334
FIGURE 110 (ORGANIGRAMME OU L'ELEVE OUBLIE QU'UNE MOYENNE DE DEUX NOMBRES ENTIERS PEUT NE PAS ETRE UN NOMBRE ENTIER) 334
FIGURE 111 (UTILISATION DU TABLEUR POUR LE CALCUL DE LE MOYENNE DE DEUX NOMBRES ENTIERS) 336
FIGURE 112 (UN ALGORITHME DU JOUEUR A) 338
FIGURE 113 (ORGANIGRAMME DE L'ALGORITHME DU JOUEUR A ET SA TRANSCRIPTION EN PSEUDO-CODE) 338
FIGURE 114 (BRIQUE DE CONSTRUCTION D'UNE BOUCLE « REPETER-JUSQU'A » AVEC LARP) 339

FIGURE 115 BRIQUE DE CONSTRUCTION D'UNE BOUCLE « TANTQUE » AVEC LARP) 339
FIGURE 116 (REPRESENTATION GRAPHIQUE D'UNE FONCTION CONTINUE F SUR [A ; B] AVEC F(A)×F(B)<0 358
FIGURE 117 (ILLUSTRATION DU THEOREME DES VALEURS INTERMEDIAIRES) 359
FIGURE 118 (ILLUSTRATION DU THEOREME DE LA BIJECTION) 359
FIGURE 119 (ENONCE AUTOUR D'UNE FONCTION POLYNOMIALE DE DEGRE 3) 364
FIGURE 120 (CORRIGE DE L'EXERCICE PROPOSE A LA FIGURE 119) 364
FIGURE 121 (CAS D'UNE FONCTION « CACHEE » ADMETTANT PLUSIEURS ZEROS SUR L'INTERVALLE ETUDIE) 366
FIGURE 122 (CAS D'UNE FONCTION « CACHEE » QUI N'EST PAS DEFINIE EN UNE VALEUR DE L'INTERVALLE ETUDIE) 366
FIGURE 123 (REPRESENTATION GRAPHIQUE DE LA FONCTION « CACHEE » ADMETTANT PLUSIEURS ZEROS SUR L'INTERVALLE ETUDIE) 368
FIGURE 124 (REPRESENTATION GRAPHIQUE DE LA FONCTION « CACHEE » QUI N'EST PAS DEFINIE EN UNE VALEUR DE L'INTERVALLE ETUDIE) 368
FIGURE 125 (ORGANIGRAMME POUR UNE FONCTION F GENERIQUE) 370
FIGURE 126 (ALGORITHME SOUS ALGOBOX AVEC L'UTILISATION D'UN IDENTICATEUR F1 POUR LA FONCTION) 381
FIGURE 127 (ALGORITHME SOUS ALGOBOX AVEC RECOURS A L'EXPRESSION DE LA FONCTION CHAQUE FOIS QU'ELLE EST APPELEE PAT LE PROGRAMME) 381
FIGURE 128 (ALGORITHME OU LA FONCTION EST VUE COMME UNE VARIABLE DE TYPE « CHAINE DE CARACTERE ») 382
FIGURE 129 (UN ALGORITHME SOUS ALGOBOX AVEC UNE STRUCTURE « TANTQUE ») 387
FIGURE 130 (ALGORITHMES PROPOSANT OU NON D'UTILISER LA FONCTIONNALITE D'ALGOBOX : « AFFICHERCALCUL ») 387
FIGURE 131 (ORGANIGRAMME SOUS LARP AVEC UTILISATION D'UNE STRUCTURE « TANTQUE ») 388
FIGURE 132 (TRANSCRIPTION EN « PSEUDO-CODE » DE L'ORGANIGRAMME OBTENU A LA FIGURE 131 SOUS LARP) 388
FIGURE 133 (CAS OU LE TEST DE SORTIE DE BOUCLE EST $F(X + 10^{-3}) < 0$) 389
FIGURE 134 (CAS OU LE TEST DE SORTIE DE BOUCLE EST $F(X + 10^{-3}) – F(X) > 0$) 389
FIGURE 135 (CAS D' UN AJOUT D'UNE LIGNE D'AFFECTATION « F(X) PREND LA VALEUR $F(X) \times F(X + 10^{-3})$ » DANS LA BOUCLE « REPETER… JUSQU'A », SUIVIE DE LA CONDITION DE SORTIE « F(X) > 0 ») 389
FIGURE 136 (TYPE D'ERREUR LORS DE L'IMPLETEMTATION D'UN ALGORITHME DANS L'ENVIRONNEMENT LARP) 389
FIGURE 137 (TYPE D'ERREUR LORS DE L'IMPLETEMTATION D'UN ALGORITHME DANS L'ENVIRONNEMENT LARP) 389
FIGURE 138 (TYPE D'ERREUR LORS DE L'IMPLETEMTATION D'UN ALGORITHME DANS L'ENVIRONNEMENT LARP) 390
FIGURE 139 (TYPE D'ERREUR LORS DE L'IMPLETEMTATION D'UN ALGORITHME DANS L'ENVIRONNEMENT LARP) 390
FIGURE 140 (TYPE D'ERREUR LORS DE L'IMPLETEMTATION D'UN ALGORITHME DANS L'ENVIRONNEMENT LARP) 390
FIGURE 141 (TYPE D'ERREUR LORS DE L'IMPLETEMTATION D'UN ALGORITHME DANS L'ENVIRONNEMENT LARP) 390
FIGURE 142 (UN ALGORITHME « PAPIE-CRAYON ») 392
FIGURE 143 (TYPE D'ERREUR LORS DE L'IMPLETEMTATION D'UN ALGORITHME DANS L'ENVIRONNEMENT LARP) 394
FIGURE 144 (TYPE D'ERREUR LORS DE L'IMPLETEMTATION D'UN ALGORITHME DANS L'ENVIRONNEMENT LARP) 396
FIGURE 145 (TYPE D'ERREUR LORS DE L'IMPLETEMTATION D'UN ALGORITHME DANS L'ENVIRONNEMENT LARP) 396
FIGURE 146 (ALGORITHME PERMETTANT D'AFFICHER LORS DE SON EXECUTION LES CALCULS DES IMAGES DES BORNES DE L'INTERVALLE PAR LA FONCTION F1 DONT LES ELEVES NE CONNAISSENT PAS L'EXPRESSION) 405
FIGURE 147 (UNE COPIE D'ELEVE DE 2NDE SUR LA DICHOTOMIE AVEC LA FONCTION « CACHEE » ADMETTANT PLUSIEURS ZEROS SUR L'INTERVALLE D'ETUDE) 406
FIGURE 148 (EXTRAIT DE COPIE OU TOUT EST EN JEU SUR LES REGISTRES ET LA SEMIOTIQUE) 407
FIGURE 149 (DIVERSES REPRESENTATIONS GRAPHIQUES OBTENUES AVEC ALGOBOX DE LA FONCTION « CACHEE » ADMETTANT PLSUIEURS ZEROS SUR L'INTERVALLE ETUDIE) 408

FIGURE 150 (COPIE D'ELEVES DE 2NDE AVEC PROPOSITION D'UNE CONCLUSION FAUSSE DANS LE CAS OU LA FONCTION « CACHEE » ADMET PLUSIEURS ZEROS SUR L'INTERVALLE ETUDIE) 409
FIGURE 151 (COPIE D'ELEVES DE 2NDE AVEC TRAÇAGE DES POINTS DE D'ABSCISSES LES BORNES SUP ET INF DE LA COURBE REPRESENTATIVE DE LA FONCTION F1) 410
FIGURE 152 (COPIE D'ELEVES DANS LE CAS DE LA FONCTION « CACHEE » ADMETTANT PLUSIEURS ZEROS SUR L'INTERVALLE D'ETUDE) 410
FIGURE 153 (EXEMPLE D'IMPLEMENTATION DANS ALGOBOX DE L'ALGORITHME DE LA FONCTION « CACHEE » NON DEFINIE EN UN POINT DE L'INTERVALLE D'ETUDE) 421
FIGURE 154 (REPRESENTATION GRAPHIQUE DE LA FONCTION « CACHEE » AVEC UN PAS DE 0,001) 425
FIGURE 155 (CAS OU LE PAS EST DE 1) 425
FIGURE 156 (CAS OU LE PAS EST DE 0,1) 425
FIGURE 157 (CAS OU LE PAS EST DE 0,01) 425
FIGURE 158 (ZOOM DE LA REPRESENTATION GRAPHIQUE DE LA FONCTION « CACHEE » SUR [1 ;2]) 428
FIGURE 159 (SERIE D'EXERCICES SUR LES SUITES ADJACENTES) 439
FIGURE 160 (ALGORITHME SOUS ALGOBOX SUR LA RESOLUTION DE L'EQUATION $X^3 + X – 1 = 0$ PAR LA METHODE DE DICHOTOMIE) 447
FIGURE 161 (DEUX ENONCES SUR LES SUITES ADJACENTES) 450
FIGURE 162 (EXTRAIT D'UN ENREGISTREMENT AUDIO CONCERNANT LES REPONSES D'UN ELEVE SUR LES SUITES ADJACENTES) 451
FIGURE 163 (EXTRAIT D'UNE COPIE D'UN BINOME A SUR L'ETUDE DE LA MONOTONIE D'UNE SUITE) 455
FIGURE 164 (EXTRAIT D'UNE COPIE D'UN BINOME B SUR L'ETUDE DE LA MONOTONIE D'UNE SUITE) 456
FIGURE 165 (EXTRAIT D'UNE COPIE DU BINOME B SUR L'ETUDE DE DEUX SUITES ADJACENTES) 457
FIGURE 166 (COPIE DE DEUX ELEVES SUR LA DEMONSTRATION DU TVI PAR LA DICHOTOMIE) 459
FIGURE 167 (COPIE D'UN BINOME MONTRANT UNE UTILISATION D'ALGOBOX POUR L'IMPLEMENTATION D'UN ALGORITHME DE DICHOTOMIE APPLIQUE A LA FONCTION F DEFINIE SUR [0 ; 1], PAR $F(X) = X^3 + X – 1$) 460
FIGURE 168 (COPIE D'UN BINOME SUR LES SUITES ADJANCES ET UNE DEMONSTRATION DU TVI) 461
FIGURE 169 (ENONCE SUR L'ETUDE DE LA CONVERGENCE DE DEUX SUITES DONT ON CONNAIT LEURS EXPRESSIONS EN FONCTION DE N) 463
FIGURE 170 (UNE MARCHE ALEATOIRE SUR UN AXE GRADUE) 474
FIGURE 171 (UN ALGORITHME MODELISANT UNE MARCHE ALEATOIRE SUR UN AXE GRADUE) 475
FIGURE 172 (EXTRAIT D'UN EXERCICE DU MANUEL MATHS-REPERES DE SECONDE : ALGORITHME ET PARAMETRE STATISTIQUE) 476
FIGURE 173 (EXTRAIT D'UN EXERCICE DU MANUEL MATHS-REPERES DE 2NDE : LE LIEVRE ET LA TORTUE) 478
FIGURE 174 (ALGORITHME PERMETTANT DE SIMULER UNE PARTIE DU « LIEVRE ET LA TORTUE ») 479
FIGURE 175 (EXTRAIT D'UN EXERCICE DU MANUEL SYMBOLE EN PREMIERE SCIENTIFIQUE : EXEMPLE D'UN LANCER D'UNE PIECE) 486
FIGURE 176 (REPRESENTATION GRAPHIQUE DE LA DEMARCHE A METTRE EN PLACE LORS D'UN TRAVAIL SUR LE SIMULATION D'UN PROCESSUS ALEATOIRE) 490
FIGURE 177 (LES OBJECTIFS DES UNIVERS DES EXPERIENCES, DE L'ALGORITHMIQUE ET DES MATHEMATIQUES EN DIRECTION DE L'UNIVERS DE LA SIMULATION) 491
FIGURE 178 (UNE POLITIQUE DES NAISSANCES) 492
FIGURE 179 (ARBRE PONDERE REPRESENTANT LES DIFFERENTS TYPES POSSIBLES DE FAMILLES A 4 ENFANTS) 496
FIGURE 180 (PRINCIPE D'ELABORATION D'UNE LISTE D'INSTRUCTIONS) 501
FIGURE 181 (TEXTE DISTRIBUE AUX ELEVES SUR LA POLITIQUE DES NAISSANCES 1ERE PARTIE) 504
FIGURE 182 (TEXTE DISTRIBUE AUX ELEVES SUR LA POLITIQUE DES NAISSANCES 2EME PARTIE) 504
FIGURE 183 (SONDAGE D'OPINION DISTRIBUE AUX ELEVES SUR LA POLITIQUE DES NAISSANCES 3EME PARTIE) 505
FIGURE 184 (DOCUMENT FOURNI AUX ELEVES EN LIEN AVEC UNE UTILISATION DU TABLEUR POUR UN NOMBRE DE FAMILLES FIXE) 505
FIGURE 185 (TACHES ALGORITHMIQUES QUE LES ELEVES DOIVENT FAIRE AUTOUR DE LA POLITIQUE DES NAISSANCES) 506
FIGURE 186 (RAPPEL DE LA POLITIQUE DES NAISSANCES) 507
FIGURE 187 (UN ARBRE DE PROBABILITE) 517
FIGURE 188 (UN ARBRE DE PROBABILITE) 517
FIGURE 189 (LE TABLEUR POUR REPRESENTER LES NAISSANCES DES ENFANTS, LE NOMBRE D'ENFANTS PAR FAMILLE ET LE NOMBRE DE GARÇON PAR FAMILLE) 519

FIGURE 190 (LE TABLEUR POUR REPRESENTER LES NAISSANCES DES ENFANTS, LE NOMBRE D'ENFANTS PAR FAMILLE ET LE NOMBRE DE GARÇON PAR FAMILLE) 519
FIGURE 191 (EXTRAIT D'UNE COPIE : UN ORGANIGRAMME DE LA POLITIQUE DES NAISSANCES OU EST MIS EN PLACE UNE SERIE DE « SI … ALORS … SINON ») 522
FIGURE 192 (EXTRAIT D'UNE COPIE : UN ORGANIGRAMME DE LA POLITIQUE DES NAISSANCES UTILISANT UNE BOUCLE « TANTQUE ») 523
FIGURE 193 (EXTRAITS DE COPIES PRESENTANT DES ALGORITHMES DE LA POLITIQUE DES NAISSANCES) 524
FIGURE 194 (EXTRAIT D'UNE COPIE SUR LA POLITIQUE DES NAISSANCES) 525
FIGURE 195 (DEUX EXTRAITS DE COPIES SUR LA POLITIQUE DES NAISSANCES) 527
FIGURE 196 (UN MODELE REPRESENTATIF DE LA SITUATION DE LA POLITIQUE DES NAISSANCES A PARTIR D'UN JEU DE LANCERS SUCCESSIFS D'UNE PIECE DE MONNAIE EQUILIBREE OU PAS) 529
FIGURE 197 (EXTRAIT D'UNE COPIE OU EST PROPOSE UN ALGORITHME SUR LE JEU DE LANCERS SUCCESSIFS D'UNE PIECE DE MONNAIE EQUILIBREE AVEC L'UTILISATION D'UNE BOUCLE « TANTQUE ») 529
FIGURE 198 (EXTRAITS DE COPIES REPRESENTANT DES ALGORITHMES AUX FORMATS « TEXTUEL » OU « SPATIAL » DU JEU DU LANCERS SUCCESSIFS D'UNE PIECE ET DE LA POLITIQUE DES NAISSANCES) 530
FIGURE 199 (EXTRAITS DE COPIES SUR LA POLITIQUE DES NAISSANCES) 531
FIGURE 200 (EXTRAIT D'UNE COPIE OU L'ALGORITHME « COMPLET » PERMET D'ENVISAGER LA PRISE EN COMPTE DU NOMBRE MAXIMUM D'ENFANTS PAR FAMILLE, DU NOMBRE DE FAMILLES ETUDIEES ET DE LA PROBABILITE DE NAISSANCE D'UN GARÇON) 532
FIGURE 201 (EXTRAITS DE COPIES DONNANT LES PROBABILITES DES DIFFERENTES TYPES DE FAMILLE) 534
FIGURE 202 (EXTRAIT D'UNE COPIE DONNANT LES PROBABILITES DES DIFFERENTES TYPES DE FAMILLE) 535
FIGURE 203 (ETM/ETA ASSOCIES AU CYCLE DE MODELISATION DE BLUM ET LEISS (2005)) 542
FIGURE 204 (REFERENCES AUX ETM ET ETA (PHASES 2, 3 ET 4)) 542
FIGURE 205 (ADAPTATION DES ETM ET ETA (PHASES 2, 3 ET 4) 543
FIGURE 206 (LA DYNAMIQUE D'EVOLUTION D'UN TRAVAIL MATHEMATIQUE ET ALGORITHMIQUE DURANT LES DIFFERENTES SESSIONS) 543
FIGURE 207 (ETM/ETA ASSOCIES AU CYCLE DE MODELISATION DE BLUM ET LEISS (2005)) 559
FIGURE 208 (ORGANIGRAMME : REPRESENTATION D'UNE SEQUENCE) 589
FIGURE 209 (ORGANIGRAMME : REPRESENTATION D'UN CHOIX BINAIRE) 590
FIGURE 210 (ORGANIGRAMME : REPRESENTATION D'UNE ACTION D'ECHANGE ENTRE LA MACHINE ET L'UTILISATEUR (ENTREE OU SORTIE)) 590
FIGURE 211 (ORGANIGRAMME : REPRESENTATION POUR LA SELECTION) 590
FIGURE 212 (ORGANIGRAMME : DEBUT DE L'ALGORITHME) 590
FIGURE 213 *(ORGANIGRAMME : FIN DE L'ALGORITHME)* 590
FIGURE 214 (ORGANIGRAMME : LA SEQUENCE) 591
FIGURE 215 (ORGANIGRAMME : L'INSTRUCTION CONDITIONNELLE) 591
FIGURE 216 (ORGANIGRAMME : L'ALTERNATIVE) 591
FIGURE 217 (ORGANIGRAMME : BOUCLE « TANTQUE … FAIRE ») 592
FIGURE 218 (ORGANIGRAMME : BOUCLE « JUSQU'A CE QUE … FAIRE ») 592
FIGURE 219 (ORGANIGRAMME : BOUCLE « « POUR… ALLANT DE … JUSQU'A ») 593

Introduction

1. Les origines de notre travail de recherche

Depuis 2010, une introduction progressive d'un enseignement de l'algorithmique dans les programmes de mathématiques s'est mise en place dans les trois années de lycée (Seconde en 2010, Première en 2011 et Terminale en 2012). Cette introduction s'est élargie à partir de 2015 au cycle 1 (classes de maternelles), puis en 2016 aux cycles 2, 3 et 4, dans le cadre de l'enseignement des mathématiques à l'école primaire et au collège.

Afin de mieux comprendre le cheminement qui a amené l'institution à introduire un enseignement de l'algorithmique dans le système scolaire depuis le début de cette décennie, nous proposons de faire un retour historique sur les travaux mis en place par la commission Kahane dont on peut penser qu'elle est à l'origine de cette introduction de l'algorithmique dans le système scolaire.

1.1 La Commission Kahane

A la demande du ministère de l'Education Nationale, le Professeur Kahane[1] réunit en 1999 un groupe d'enseignants et de chercheurs pour conduire une Commission[2] de réflexion sur l'enseignement des mathématiques, en amont du Conseil national des programmes et du groupe d'experts chargés d'élaborer les programmes de mathématiques de l'enseignement secondaire, ainsi qu'une réflexion sur l'enseignement des mathématiques de l'Ecole élémentaire à l'Université.

Lors d'une première phase de concertations, la Commission produit des rapports assortis d'annexes sur quatre thèmes estimés prioritaires : **(1)** *la géométrie et son enseignement* ; **(2)** *l'informatique et l'enseignement des mathématiques* ; **(3)** *le calcul* ; **(4)** *les probabilités et les statistiques*.

[1] Mathématicien français, membre de l'Académie des Sciences

[2] https://www.pedagogie.ac-aix-marseille.fr/jcms/c_75043/fr/commission-kahane (Consulté le 04/10/17)

1.2 Une (ré)introduction d'éléments d'informatique proposée par la commission Kahane : quelles conséquences en termes de besoins dans le domaine de l'algorithmique ?

1.2.1 Une (ré)introduction d'éléments d'informatique

Depuis 2000, suite aux travaux de la Commission Kahane, une réintroduction d'éléments en lien avec le domaine de l'informatique dans les programmes scolaires est mise en place. Elle fait suite à une évolution perceptible qui s'est mise en place petit à petit depuis la deuxième moitié des années 90, après une première existence à travers un enseignement optionnel au cours de la décennie des années 80 (cf. Thèse de Lagrange, 1991).

En parallèle, des domaines mathématiques comme l'analyse, les statistiques et les probabilités font l'objet d'une introduction d'éléments en lien avec l'informatique. Le développement des calculatrices à calcul formel et graphique, de logiciels de géométrie dynamique, et l'utilisation de tableurs favorisent cette introduction à travers les TICE[3]. L'institution, à travers des documents d'accompagnement et les manuels scolaires proposent de nombreuses activités sollicitant l'utilisation de tels instruments.

1.2.2 Une première introduction « timide » d'éléments issus de l'informatique, de 2000 à 2009 : le cas du domaine de l'arithmétique

A partir de 2000, l'arithmétique, domaine spécifique des mathématiques, est introduite dans l'enseignement, en particulier dans cadre des programmes de l'enseignement de spécialité de mathématiques de la Terminale Scientifique. Cet enseignement de l'arithmétique permet entre autres une introduction de diverses tâches en lien avec l'utilisation d'éléments informatiques.

Cependant, comme le soulignent Thanh & A. Bessot (2006, p. 14), *l'enseignement* [des mathématiques] *veut attacher des notions de base de l'informatique (comme boucle, test, structure de données) à des domaines des mathématiques, comme […] l'arithmétique, sans que la notion d'algorithme soit objet d'enseignement*.

Les manuels proposent de nombreux exercices d'exécution de programmes informatiques

[3] Technologies de l'information et de la communication pour l'enseignement

ne permettant qu'*une très faible familiarisation des élèves avec des algorithmes mathématiques* (Birebent et Chi Thanh, 2005, p. 4). Les élèves utilisent des algorithmes sans que pour autant un enseignement explicite et institutionnalisé de l'algorithmique trouve sa place dans le système éducatif tout au long de cette décennie. Il est ainsi proposé aux élèves et aux enseignants, des programmes scolaires sans que pour autant une spécificité de l'algorithmique et de la programmation ne soit enseignée. De plus, comme le soulignent Birebent et Chi Thanh (2005), l'institution ne propose pas de langages particuliers de programmation.

Au cours de cette période, la difficulté de faire vivre des apprentissages autour des notions de base en informatique *se traduit dans les manuels par le refus de construire des types de tâches relatives à la programmation des algorithmes et par le recours à des logiciels de calcul pour instrumenter les techniques associées à ces tâches* (Nguyen, 2006, p.40).

1.2.3 Une introduction progressive d'un enseignement de l'algorithmique et ses conséquences d'un point de vue formation des enseignants

A partir de 2009, l'idée d'une introduction officielle d'un enseignement de l'algorithmique dans le secondaire se met en place. Un tel enseignement se fait de façon progressive à tous les niveaux de l'enseignement secondaire français, de 2009 à 2012. Une approche de l'algorithmique associée à l'enseignement des mathématiques est aussi introduite au cours des années 2015 et 2016 dans les cycles de l'enseignement primaire et des collèges. L'institution insiste ainsi sur la nécessité d'amener les élèves à s'initier à l'algorithmique.

De même, selon les directives institutionnelles, cette introduction de l'algorithmique se fait de façon transversale, quel que soit le niveau d'enseignement, de la maternelle au lycée. Cela a pour conséquence de mettre en place de nouveaux besoins nécessitant un complément de formation des enseignants dans le domaine de l'algorithmique.

Dans le cadre de cette thèse, nous nous interrogeons alors sur les apports que peuvent avoir cette introduction de l'algorithmique auprès des élèves, dans certains domaines mathématiques enseignés. Pour cela, dans le cadre d'*ingénieries didactiques* (au sens d'Artigue, 1992), nous proposons des expérimentations en classe, permettant d'observer chez des élèves de lycée, comment l'algorithmique peut donner du sens à l'introduction de certains

concepts mathématiques enseignés, dans les domaines de l'arithmétique, de l'analyse, et des probabilités – statistiques.

1.3 De la commission Kahane aux programmes

L'évolution technologique autour des ordinateurs et de l'informatique existante depuis le début des années 40, conduit la commission Kahane (1999, 2000), dans la continuité de ce qui avait été entrepris au cours des décennies précédentes, à s'intéresser à l'évolution de l'enseignement des mathématiques au lycée (et à l'Université).

La commission propose quelques éléments de réponses autour de trois grandes questions :

(1) *Pourquoi introduire une part d'informatique dans l'enseignement des sciences mathématiques et dans la formation des maitres ?*
(2) *Comment faire évoluer les programmes pour accompagner cette évolution* ?
(3) *Quels professeurs* ?
(Extrait du rapport (2002) : *Informatique et enseignement des mathématiques*)

avec l'objectif de faire évoluer progressivement les contenus « *pour intégrer de nouveaux objets et notions d'algorithmique et programmation* ». (Ibid.)

Comme le rapportent Chi Thanh et Bessot (2010, p. 10), ce rapport souligne une distinction fondamentale :

- *D'une part, l'utilisation de logiciels à travers des ordinateurs et des calculatrices ;*
- *D'autre part, l'apprentissage et l'enseignement des concepts de base de l'algorithmique et de la programmation.*

Pour la commission, les deux principaux concepts de base de la programmation sont : **(1)** les structures de contrôle, c'est-à-dire les boucles et les branchements ; **(2)** la récursivité. De même, pour la notion d'algorithme, la commission propose de faire travailler les élèves sur les structures de données et la notion de complexité sans pour autant trop théoriser ces thématiques.

Chi Thanh et Bessot constatent que le travail de la commission Kahane influence les programmes qui sont mis en place en 2009 au niveau de la classe de Seconde, en particulier autour de l'enseignement de l'algorithmique. En effet, le programme va fondamentalement distinguer l'algorithmique de l'usage des logiciels. Ainsi, l'algorithmique peut exister

indépendamment de la programmation. La commission Kahane identifie la présence de certains concepts de base de l'algorithmique et de la programmation, qui sont déclinés sous forme de compétences. *Les machines sont mentionnées explicitement, mais comme un environnement de programmation* (Thanh & Bessot, 2010, p. 10). L'écriture d'un programme informatique est aussi l'occasion d'appliquer des règles issues du domaine de la logique. Ainsi, écrire un « programme qui marche » peut récompenser le concepteur de ses efforts de réflexion, d'analyse et de synthèse, mais *cela ne le dispense pas de s'assurer que l'algorithme termine dans tous les cas envisagés et qu'il le fait en temps raisonnable*. […] *Cette étape est une démonstration mathématique* (Extrait du rapport « Informatique et enseignement »). Toutefois, les programmes ne mentionnent pas d'enseignement explicite d'un tel type de démonstration.

1.4 Un enseignement des algorithmes au lycée

Indépendamment de cela, la commission signale que les programmes informatiques et les algorithmes sont partout. Certains peuvent être simples, d'autres plus complexes. *Ces algorithmes, même les plus simples, mettent en jeu des structures de données comme les arbres ou les graphes dont les sommets représentent les différents états du système en fonctionnement*. Par ailleurs, se référant au système éducatif, la commission introduit le concept « d'*esprit algorithmique* » (Knuth parle de *pensée algorithmique*) qui se manifesterait assez tôt comme objet d'enseignement.

> *Dans l'enseignement, l'esprit algorithmique se manifeste assez tôt comme objet d'enseignement, par exemple lors de la résolution des premiers problèmes « d'arithmétique élémentaire » comportant plusieurs « calculs » ; par la suite, il accompagne les résolutions et les démonstrations mathématiques à tous les niveaux. Les solutions sont traditionnellement présentées sous forme de « raisonnements » à l'école primaire, mais qui n'ont pas de statut solide par la suite, de sorte que leur formulation tend à disparaitre, même aux niveaux supérieurs. Ces solutions pourraient être plus commodément cherchées, étudiées, établies, explicitées, apprises et enseignées grâce à un usage concerté (au moins reconnu tout au long de la scolarité) des instruments de l'algorithmique*.
> (Kahane, 2002)

Ainsi, la commission formule la nécessité d'un enseignement explicite pour lequel nos travaux de recherche proposent un cadre de travail spécifique.

1.5 La pensée algorithmique

Pour Knuth (1985), la *pensée algorithmique* est celle de l'informatique. Il *s'appuie sur des ouvrages de mathématiques et sur des preuves qui y sont présentées pour analyser la pensée mathématique et la comparer à la pensée algorithmique* (Modeste, 2013, p. 474). Selon Knuth, l'algorithmique fait appel à des raisonnements communs aux mathématiques mais aussi à un mode de pensée spécifique. *Il divise la « pensée mathématique » en neuf « modes de pensée »* (Ibid., p. 474), dont six d'entre elles seraient commune à la *pensée algorithmique* : **(1)** manipulation de formules ; **(2)** représentation d'une réalité ; **(3)** réduction à des problèmes plus simples ; **(4)** raisonnement abstrait ; **(5)** structures d'informations ; *(6)* Algorithmes. Il ajoute deux catégories à la pensée algorithmique qui lui semblent ne pas être nécessairement présentes dans la pensée mathématique : **(7)** notion de complexité ; **(8)** notion d'affectation symbolisée par := ou ←.

De même, Modeste propose *une vision de la pensée algorithmique* qui se rapproche de celle de Hart (1998). *L'activité mathématique étant centrée sur la résolution de problèmes, la pensée algorithmique, en tant que pensée mathématique parmi d'autres, serait une approche particulière des problèmes mathématiques (de certains en tout cas)*. […] *En mathématiques, l'activité algorithmique peut se définir comme une étape dans la résolution d'une certaine catégorie de problèmes dans laquelle on recherche un procédé systématique et effectif de résolution. Il est possible d'aborder de manière utile l'objet algorithme et la pensée algorithmique en classe dans le cadre d'une réelle activité mathématique (résolution de problèmes, preuve de solutions, etc.)* (Ibid., 2012, pp. 472 & 473).

2. Quelles approches possibles d'un enseignement des algorithmes en cours de mathématiques ?

Comme nous l'avons signalé précédemment, l'enseignement de l'algorithmique au lycée fait maintenant partie intégrante de l'enseignement des mathématiques. Quel que soit le niveau scolaire de l'élève, l'enseignement de l'algorithmique se fait de façon transversale. Il n'est cependant pas prévu un enseignement théorique de l'algorithmique au lycée.

2.1 Dichotomie entre deux modes de pensées : mathématique et algorithmique

L'institution part du constat que l'élève sait que des langages informatiques sont régulièrement utilisés afin de programmer des outils numériques et réaliser des traitements automatiques de données. En revanche, l'élève ne connaît pas nécessairement les principes de base de l'algorithmique et de la conception des programmes informatiques.

Au cours des années du lycée, nous faisons l'hypothèse que la mise en œuvre d'algorithmes dans des domaines mathématiques spécifiques et leur implémentation dans des environnements numériques afin de les tester, vont permettre à l'élève d'acquérir une *pensée algorithmique* en complément à une *pensée mathématique*.

2.2 Quelle place et quel rôle donner à l'informatique selon les programmes de mathématiques des lycées ?

Nous pouvons lire dans le document ressource « Algorithmique et programmation » proposé par Eduscol[4] que la lettre de saisine du Conseil supérieur des programmes datée du 19 décembre 2014 précise les objectifs et démarches d'apprentissages suivants :

> « *L'enseignement de l'informatique et de l'algorithmique […] n'a pas pour objectif de former des élèves experts, mais de leur apporter des clés de décryptage d'un monde numérique en évolution constante. Il permet d'acquérir des méthodes qui construisent la pensée algorithmique et développe des compétences dans la représentation de l'information et de son traitement, la résolution de problèmes, le contrôle des résultats. [...] La maîtrise des langages informatique n'est pas la finalité de l'enseignement, mais leur pratique est le moyen d'acquérir […] d'autres modes de résolution de problèmes, de simulation ou de modélisation.* »

Ainsi, la conception d'algorithmes et leur programmation dans des environnements numériques deviennent un thème d'étude à part entière. L'objectif est de guider les élèves à écrire et mettre au point des algorithmes en lien avec différents domaines mathématiques enseignés afin qu'ils puissent les implémenter dans des environnements numériques pour les tester. L'institution semble partir de l'hypothèse que les élèves peuvent s'initier en autonomie (au moins partielle) à la construction d'algorithmes répondant à une problématique mathématique donnée. Cependant, l'institution n'attend pas que l'élève acquière *une*

[4] http://cache.media.eduscol.education.fr/file/Algorithmique_et_programmation/67/9/RA16_C4_MATH_algorithmique_et_programmation_N.D_551679.pdf (Consulté le 1er octobre 2017)

connaissance experte et exhaustive d'un langage ou d'un logiciel particulier. En créant un algorithme implémentable dans un environnement numérique, l'élève doit développer *des méthodes de programmation,* revisiter *les notions de variables et de fonctions sous une forme différente, et* s'entraîner *au raisonnement.*[5]

2.3 Contexte et questions générales de notre recherche

2.3.1 Le contexte de notre recherche

Au cours de ce travail de recherche, nous proposons la construction et l'expérimentation en classe d'un certain nombre d'ingénieries didactiques dans divers domaines mathématiques spécifiques enseignés dans les classes de Seconde et du Cycle Terminal Scientifique. En effet, l'introduction d'un enseignement de l'algorithmique au niveau des classes du lycée depuis le début des années 2010 permet d'étudier quelle peut être la contribution de l'algorithmique, quand cela est possible, chez l'élève débutant en informatique aux apprentissages dans différents domaines mathématiques enseignés et quels sont ainsi les apports de l'algorithmique au développement de savoirs spécifiques.

De même, nous situons notre travail de recherche dans un contexte particulier. En effet, nous souhaitons rester dans la continuité des propositions avancées par les membres de la commission Kahane sur l'introduction d'un enseignement de l'informatique dans le secondaire. Pour cela, nous partons de l'hypothèse qu'une introduction d'un enseignement de l'algorithmique dans les classes de lycées demande une adaptation de la part de nombreux enseignants qui ont, pour une grande majorité, en particulier pour les plus anciens, reçu une formation initiale où l'utilisation de l'informatique est pour l'essentiel en relation avec une discipline mathématique universitaire, l'« *Analyse numérique* » et qui leur semble bien loin des attentes institutionnelles contemporaines. En effet, cet enseignement universitaire a pour objectif de développer chez l'étudiant des compétences sur la résolution analytique ou numérique de systèmes d'équations différentielles, aux dérivées partielles ou intégrales. Des problèmes d'existence et d'unicité de solutions sont aussi également présents. Un tel enseignement universitaire semble bien éloigné pour ces enseignants ayant reçu une telle formation, des attentes de l'institution avec l'introduction récente de l'algorithmique dans le

[5] http://cache.media.eduscol.education.fr/file/Initiation_a_la_programmation/92/6/RA16_C2_C3_MATH_initiation_programmation_doc_maitre_624926.pdf (Consulté le 01 octobre 2017)

secondaire. Cela nous amène alors à nous positionner sur une approche des tâches qui peuvent conduire l'enseignant à donner du sens, auprès de ses élèves, à un enseignement de l'algorithmique ne relevant pas nécessairement de l'enseignement universitaire décrit ci-dessus, et permettant à l'élève d'étudier l'algorithmique comme objet d'apprentissage en classe de mathématique. Il s'agit donc bien de transformer un savoir savant en un savoir à enseigner.

Les nouveaux programmes de lycée fixent des objectifs précis en matière d'algorithmique. Celui-ci constitue un champ transversal du lycée, permettant aux élèves d'apprendre à construire une démarche scientifique. Ainsi, d'après les auteurs des programmes, dans le cadre d'un enseignement de l'algorithmique, trois objectifs fondamentaux émergent :

- L'approfondissement des bases de la logique et du raisonnement ;
- L'illustration des concepts enseignés par l'utilisation d'outils informatiques ;
- Le développement chez les élèves d'un esprit de créativité et d'initiative au travers de l'expérimentation.

Nous partons du constat qu'à travers la lecture de ces programmes des trois années du lycée, l'enseignement de l'algorithmique apparaît comme un *outil* permettant de donner du sens à un certain nombre de notions étudiées. Cependant, dans la continuité des travaux de Modeste (2012) et de Briant (2013), nous nous demandons comment dépasser ce stade pour que l'algorithmique, dans le cadre d'une dialectique « *outil-objet* » au sens de Douady, devienne un objet d'apprentissage dans des domaines mathématiques spécifiques.

Pour cela, nous proposons dans une première partie un premier chapitre permettant de présenter les objectifs des programmes mathématiques dans la classe de Seconde et des classes du cycle Terminal Scientifique ainsi qu'un aperçu sur un certain nombre de travaux sur la recherche en didactique autour de l'introduction de l'algorithmique au lycée.

2.3.2 Les questions générales qui sont les éléments moteurs de notre recherche

Dans le cadre de notre problématique et de notre méthodologie, nous choisissons d'élaborer un cadre théorique se situant dans la continuité des travaux mis en place par les chercheurs Houdement, Kuzniak et Richard sur le concept d'*Espaces de Travail*. En effet, nous partons de l'hypothèse que les activités mathématiques proposées aux élèves des lycées français articulent plusieurs champs, certains proprement mathématiques et d'autres

relevant d'un autre domaine scientifique ou professionnel, ou encore de l'expérience quotidienne. L'hypothèse sous-jacente est que les « interactions » ou « articulations » faites par les élèves entre ces différents champs peuvent être porteuses de significations et par conséquent contribuer aux apprentissages de nouveaux concepts mathématiques et informatiques. Se pose alors, une première question de recherche pour mener à bien ce travail de thèse :

Q1 : Comment le cadre des *Espaces de Travail* et les niveaux de paradigmes peuvent-il permettre la conception et l'analyse d'ingénieries didactiques dans différents domaines mathématiques ?

Afin de montrer comment la rencontre avec les *Espaces de Travail Mathématique* nous a conduit à l'élaboration d'un nouveau cadre théorique basé sur les articulations et interactions entre *Espaces de Travail Mathématique* et *Espaces de Travail Algorithmique*, mais aussi sur l'étude des niveaux de paradigmes mathématiques et algorithmiques permettant de mener des analyses fines du travail des élèves suivant le domaine mathématique et leurs niveaux scolaires, nous souhaitons répondre aux deux questions clés suivantes dans le cadre de ce travail de recherche :

Q2 : Comment ce cadre théorique permet-il l'articulation entre mathématique et informatique dans la construction d'ingénieries ?

Q3 : Comment ce cadre théorique peut nous permettre dans des situations de type algorithmique, et à partir d'un présupposé épistémologique d'interactions fortes entre les mathématiques et l'informatique, de construire des ingénieries articulant ces deux domaines ?

Pour cela, nous proposons trois scénarios représentant chacun une ingénierie associée à un domaine particulier des mathématiques :

la ***théorie élémentaire des nombres***, l'**analyse** et les **statistiques-probabilités**.

Chacune de ces ingénieries va être associée à une étude précise en termes d'articulations et d'interactions entre *Espaces de Travail Mathématique* et *Algorithmique* autour de l'apport de l'algorithmique à l'apprentissage de nouveaux concepts mathématiques.

Ainsi, lors des ingénieries dans les domaines de la *théorie élémentaire des nombres* et de l'*analyse*, nous souhaitons étudier les questions suivantes :

Q4 : Quel peut être l'apport de l'algorithmique au concept d'une *preuve heuristique* (c'est-à-dire *non explicative*) et d'une *preuve explicative* dans le cas

d'un travail sur une conjecture autour d'un processus de calcul dans le domaine de la *théorie élémentaire des nombres* ?

Q5 : Quelle peut être la contribution d'une approche algorithmique de la *preuve* dans le domaine de l'analyse ?

De plus, dans le cadre d'une tâche associée au domaine des statistiques-probabilités, où nous allons analyser les apports de l'algorithmique aux apprentissages de phénomènes aléatoires, nous souhaitons traiter les questions suivantes :

Q6 : Quel est le caractère « algorithmique » d'un *Espace de Travail* intervenant dans le *cycle de modélisation* (au sens de Blum & Leiss, 2005) d'une simulation aléatoire et la spécificité de cet *Espace de Travail* dans les modèles proposés par les élèves ?

Q7 : Au cours du travail sur la simulation d'un phénomène aléatoire, quelles relations entretient l'*Espace de Travail Algorithmique* avec d'autres *Espaces de Travail Mathématique*, en particulier lors des différentes phases du *cycle de modélisation* ?

3. Déroulement et plan de la thèse

Le travail de recherche effectué pendant ces années de thèse se situe autour d'apprentissages de compétences sur la construction d'algorithmes et de leurs utilisations dans l'enseignement des mathématiques des classes de Seconde et du cycle Terminal Scientifique. Comme nous le signalions précédemment, nous avons élaboré des ingénieries didactiques dans des domaines mathématiques spécifiques qui ont été, suivant le domaine mathématique, expérimentées dans des classes de Seconde et des classes du cycle Terminal Scientifique.

Nous souhaitons ainsi, dans le cadre des interactions et articulations entre *Espaces de Travail*, et en fonction des paradigmes algorithmiques et mathématiques, analyser les différentes phases des activités des élèves pour chacune de nos ingénieries. Nous partons de l'hypothèse que l'algorithmique et la programmation font parties de l'activité humaine (Lagrange, 2016). En effet, quand nous mettons en place des activités nécessitant le recours à la programmation, il nous semble pertinent que l'élève ait une réflexion de nature algorithmique. Les questions centrales lors de la construction d'algorithmes sur la

terminaison, l'*effectivité* et la *complexité,* permet de justifier que le passage à la programmation peut constituer un champ expérimental utile, justifiant ainsi le fait que les élèves sont amenés à implémenter des algorithmes écrits au « papier-crayon » dans des environnements numériques permettant un travail d'analyse des algorithmes plus approfondi.

3.1 Le déroulement

Cette thèse est découpée en deux grandes parties, d'inégales longueurs, qui sont elles-mêmes découpées en plusieurs chapitres. Pour chaque chapitre, nous proposons une conclusion permettant de faire la transition avec le chapitre suivant. De plus, les deux parties se terminent par une conclusion globale synthétisant le travail présenté dans les différents chapitres constituant la partie. Nous terminons le manuscrit par une conclusion générale donnant une vision synthétique de la recherche entreprise pendant les années de thèse. Dans le cadre d'une prise de recul sur le travail entrepris pendant ces années, la conclusion générale va permettre aussi de revenir sur certaines hypothèses et questions faites au début de la thèse, et ainsi de poser de nouveaux questionnements ouvrant la possibilité d'une suite au travail de recherche entrepris, ainsi que l'élaboration d'un corpus de formation des enseignants du secondaire autour des articulations envisageables entre un enseignement de la *pensée algorithmique* et de la *pensée mathématique*, plus traditionnelle au niveau du secondaire. Cela s'inscrit ainsi dans les attentes institutionnelles récentes et dans la continuité des travaux déjà entrepris par Modeste (2012) et Briant (2013).

3.2 Le plan

3.2.1 Présentation de la première partie de la thèse

La première partie est découpée en trois chapitres. Dans le premier chapitre, nous présentons quelques définitions clés que nous retenons pour parler d'algorithmes et d'algorithmique. De plus, nous exposons une première analyse des programmes des niveaux scolaires retenus pour nos expérimentations en lien avec un enseignement transversal de l'algorithmique.

Pour des questions d'organisation et de temps, mais aussi aux vues des différents domaines mathématiques spécifiques choisis pour nos expérimentations, nous faisons le choix de ne

considérer que les classes de Seconde générale et du cycle Terminal Scientifique.

Cette analyse nous permet aussi de présenter des objectifs institutionnels en lien avec notre problématique basée sur l'étude des articulations entre *Espaces de Travail Algorithmique* et *Espaces de travail Mathématiques spécifiques* (cf. Partie I, chapitre 2). Nous complétons celle-ci par une présentation de certains travaux, que nous considérons représentatifs de la recherche en didactique autour d'un enseignement de l'informatique et de l'algorithmique en cours de mathématique dans les classes des lycées français.

Dans le deuxième chapitre de cette première partie, pour traiter la question Q1 donnée dans la section 2.3.2 :

> **Q1 : Comment le cadre des *Espaces de Travail* et les niveaux de paradigmes peuvent-il permettre la conception et l'analyse d'ingénieries didactiques dans différents domaines mathématiques ?**

nous exposons notre cadre théorique qui doit nous permettre de mener à bien nos analyses des différentes expérimentations qui seront présentées dans la deuxième partie. En effet, nous rappelons que nous partons de l'hypothèse principale que les interactions entre *Espaces de Travail Algorithmique* (ETA) et *Espaces de Travail Mathématiques spécifiques* (ETM_s) peuvent aider des élèves débutants en algorithmique à mieux cerner les apports de l'informatique, tant sur le plan algorithmique que sur le plan programmation, pour l'apprentissage de nouvelles compétences mathématiques autour de la *preuve* dans des domaines mathématiques comme la *théorie élémentaire des nombres* ou l'analyse, mais aussi autour de la modélisation d'une simulation d'un phénomène aléatoire dans le domaine des statistiques-probabilités.

Dans le troisième et dernier chapitre de cette première partie, nous présentons notre problématique et notre méthodologie de recherche qui vont se situer autour des *ETA* et *ETM* idoines des élèves, et des genèses associées. Ainsi, dans ce chapitre, nous souhaitons traiter les premiers éléments de réponse aux questions 2 et 3 énoncées dans la section 2.3.2 :

> **Q2 : Comment ce cadre théorique permet-il l'articulation entre mathématique et informatique dans la construction d'ingénieries ?**
>
> **Q3 : Comment ce cadre théorique peut nous permettre dans des situations de type algorithmique, et à partir d'un présupposé épistémologique d'interactions**

fortes entre les mathématiques et l'informatique, de construire des ingénieries articulant ces deux domaines ?

3.2.2 Présentation de la deuxième partie de la thèse

La deuxième partie, découpée en six chapitres, est le cœur de notre travail de recherche. En effet, dans cette partie, nous présentons nos trois ingénieries associées à des domaines mathématiques distincts. Cette présentation inclue des analyses des travaux des élèves. Ces derniers travaillant soient de façon individuelle, soient en binôme, suivant les directives données par le chercheur aux enseignants encadrant les classes, où ont lieu les expérimentations. Les ingénieries étant découpées en phases, les analyses sont faites en fonction des différentes phases constituant l'ingénierie.

Suivant le domaine mathématique où se situe notre ingénierie, nous traitons dans cette deuxième partie les différentes questions Q4 à Q7 que nous avons présentées dans la section 2.3.2 :

Q4 : Quel peut être l'apport de l'algorithmique au concept d'une *preuve heuristique* (c'est-à-dire *non explicative*) et d'une *preuve explicative* dans le cas d'un travail sur une conjecture autour d'un processus de calcul dans le domaine de la *théorie élémentaire des nombres* ?

Q5 : Quelle peut être la contribution d'une approche algorithmique de la *preuve* dans le domaine de l'analyse ?

Q6 : Quel est le caractère « algorithmique » d'un *Espace de Travail* intervenant dans le *cycle de modélisation* (au sens de Blum & Leiss, 2005) d'une simulation aléatoire et la spécificité de cet *Espace de Travail* dans les modèles proposés par les élèves ?

Q7 : Au cours du travail sur la simulation d'un phénomène aléatoire, quelles relations entretient l'*Espace de Travail Algorithmique* avec d'autres *Espaces de Travail Mathématique*, en particulier lors des différentes phases du *cycle de modélisation* ?

Ainsi, dans le premier chapitre de cette deuxième partie, nous avons pour objectifs de présenter nos choix concernant les différentes phases de notre méthodologie et nos choix généraux, ainsi que les modes de travail retenus pour nos trois ingénieries didactiques qui

sont expérimentées dans un certain nombre de classes de Seconde et du cycle Terminal Scientifique, afin d'obtenir un nombre significatif de données sur le travail entrepris par les élèves en fonction de leurs savoirs théoriques et pratiques « anciens » en mathématiques et de leurs compétences acquises au fur et à mesure de l'évolution du travail sur la construction et la maitrise des algorithmes ainsi que de leurs utilisations dans les environnements numériques retenus par le chercheur, en accord avec les enseignants participant aux expérimentations.

Dans les cinq autres chapitres de cette deuxième partie, nous présentons les différentes ingénieries mises en place, ainsi que des analyses fines en fonction de notre cadre théorique, des différentes phases des ingénieries. Nous faisons aussi le choix de présenter des ingénieries dans trois domaines spécifiques des mathématiques enseignés, afin de donner des possibilités d'une exploitation de celles-ci dans le cadre d'une formation initiale ou continue des enseignants du secondaire second cycle autour de l'utilisation de l'algorithmique en cours de mathématique.

Les chapitres 2 et 6 de la partie deuxième partie sont associés respectivement à l'étude possible des articulations entre des *ETA* et des *ETMs* que sont la *théorie élémentaire des nombres* (Chapitre 2 : *Algorithme de Kaprekar*) et les *statistiques-probabilités* (Chapitre 6 : *Une politique des naissances*).

Le travail demandé lors de l'ingénierie sur l'*algorithme de dichotomie* en lien avec des savoirs mathématiques issus, entre autres, du domaine de l'*analyse* est découpé en deux sous-ingénieries permettant un travail progressif et en presque totale autonomie de la part des élèves dans le domaine de l'algorithmique, sachant que les élèves participants initialement à cette ingénierie sont pour la majorité des débutants en informatique, en particulier chez les élèves de Seconde, voire de Première Scientifique. C'est pour cela que nous faisons le choix de découper en trois chapitres (3, 4 et 5) la présentation de cette ingénierie.

Première partie : Constitution d'un terrain de recherche autour d'un enseignement de l'algorithmique en classe de mathématiques au lycée

- **Chapitre 1 :** Des définitions. Les programmes scolaires. Les questions initiales.
- **Chapitre 2 :** Construction d'un cadre théorique.
- **Chapitre 3 :** Problématique et méthodologie de recherche.
- **Conclusion de la partie 1.**

A partir de nos réflexions sur le bien-fondé d'un enseignement explicite de l'algorithmique comme objet d'apprentissage de nouvelles compétences mathématiques, nous développons cette première partie qui définit le cadre de notre recherche. Elle est découpée en trois chapitres.

Dans le premier chapitre, nous présentons quelques définitions clés. En effet, nous définissons ce que nous attendrons tout au long de cette thèse, sur les concepts d'algorithmes et de programmes informatiques, mais aussi sur le fait de parler d'algorithmique comme étant une science en soi. De plus, nous rappelons que nous présentons une première analyse des programmes scolaires concernant les niveaux de classe où nous expérimentons nos ingénieries didactiques bâties autour de trois domaines mathématiques spécifiques. Cette analyse présente entre autres les objectifs des programmes en lien avec notre problématique sur les interactions et articulations entre *Espaces de Travail Algorithmique (ETA)* et *Espaces de travail Mathématique spécifique (ETM$_S$)*. Nous terminons ce premier chapitre, par une présentation de certains travaux de recherche représentatifs en didactique dans le cadre d'un enseignement de l'informatique et de l'algorithmique en cours de mathématique pour les classes de lycée.

Dans le deuxième chapitre, nous exposons notre cadre théorique auquel nous nous référons pour mener nos analyses des expérimentations décrites dans la partie 2. En effet, nous rappelons que nous partons de l'hypothèse principale que les interactions et les articulations entre *ETA* et *ETM$_S$* peuvent être une aide à la compréhension du travail mené par des élèves débutants en algorithmique et à mieux cerner les apports de l'informatique, tant sur le plan algorithmique que sur le plan programmation, pour l'apprentissage de nouvelles compétences mathématiques autour de la *preuve* dans des domaines mathématiques comme la *théorie élémentaire des nombres* ou l'analyse, mais aussi autour de la modélisation d'une simulation d'un phénomène aléatoire dans le domaine des statistiques-probabilités.

Dans le dernier chapitre de cette partie, sont présentées notre problématique, notre méthodologie et notre recherche qui se situent dans un travail de recherche analysé autour des *ETA* et *ETM* idoines des élèves, et des genèses associées.

Chapitre 1 : Des définitions – Les programmes scolaires – Les questions initiales.

Dans ce premier chapitre, nous souhaitons présenter quelques définitions clés, mais aussi les principaux objectifs des programmes de mathématiques des lycées français depuis 2009. Nous nous intéressons plus particulièrement à l'enseignement des algorithmes, en particulier aux aspects de transversalité de cet enseignement dans tous les domaines mathématiques enseignés au niveau des classes de lycée.

Nous concluons ce chapitre par une première approche des questions initiales : **(1)** Un tel enseignement est-il viable en classe ? **(2)** Que peut-il générer du point de vue des apprentissages ?

1. Des définitions clés

Nous souhaitons présenter dans cette section, quelques définitions générales sur les concepts d'algorithme, d'algorithmique et de programmation informatique. Pour cela, nous allons partir de définitions issues de la littérature scientifique et celles que nous retenons.

1.1 Un algorithme

Les expressions « algorithme » et « programme informatique » sont polysémiques. Il nous semble ainsi nécessaire de clarifier le sens de ces expressions, et de préciser la définition que nous retiendrons pour le concept d'algorithme.

De nombreux chercheurs, tant en informatique qu'en mathématique, ont donné différentes définitions du concept d'algorithme. Nous proposons ici de ne présenter que celles qui nous ont semblé les plus en accord avec notre approche de l'algorithmique comme objet d'apprentissage de nouvelles notions mathématiques. Cependant, ces définitions ne sont pas exhaustives.

Selon Knuth (1973), « *La signification moderne de l'algorithme est assez similaire à celle de la recette, du procédé, de la méthode, de la technique, de la procédure, de la routine, sauf que le mot « algorithme » connote quelque chose de légèrement différent* ». En effet, un algorithme ne se réduit pas à un simple ensemble fini de règles donnant une séquence d'opérations pour résoudre un type spécifique de problème. Un algorithme doit répondre à

cinq caractéristiques importantes que sont : **(1)** la finitude (**Finiteness**). L'algorithme doit se terminer après un nombre fini d'étapes ; **(2)** la définition de chaque étape (**Definiteness**). Chaque étape de l'algorithme est définie avec précision, et les actions qui les caractérisent sont effectuées sans aucune ambiguïté ; **(3)** les entrées (**Input**). Un algorithme peut avoir zéro entrée, une entrée ou plusieurs entrées. On entend par « entrée », les quantités, les valeurs qui lui sont données initialement avant que ne débute l'algorithme. Ces entrées vont être prises dans des ensembles d'objets spécifiés ; **(4)** les sorties (**Output**). Un algorithme est constitué d'une ou plusieurs sorties. Ces sorties sont constituées des quantités ayant une relation spécifiée avec les entrées ; **(5)** l'efficacité (**Effectiveness**). Pour qu'un algorithme puisse être utilisé pour répondre à une tâche donnée, celui-ci doit également être *efficace*. On entend par *efficacité* le fait que toutes les opérations à effectuer dans l'algorithme doivent être suffisamment fondamentales pour pouvoir en principe être exécutées exactement et dans un laps de temps fini par un homme utilisant du papier et un crayon.

La définition de Bouvier, George, et Le Lionnais (2005) met aussi ce côté effectif en avant. En effet, ils définissent un algorithme comme étant « *une suite finie de règles à appliquer dans un ordre déterminé à un nombre fini de données, pour arriver avec certitude (i.e. sans indétermination ni ambiguïté), en un nombre fini d'étapes, à un certain résultat, et cela indépendamment des données* ». Ainsi, selon Bouvier & al (2005), un algorithme n'est pas un objet qui permet de ne résoudre qu'un problème unique, mais au contraire sa particularité est de permettre la résolution de *toute une classe de problèmes* qui ne différent que par les données, et qui sont *gouvernés par les mêmes prescriptions*.

De même, selon Dowek[6], le concept d'algorithme n'est pas propre à l'informatique, puisqu'il est également utilisé en mathématiques, où existent de nombreux algorithmes pour résoudre divers problèmes dans différents domaines mathématiques. Si on fait démarrer l'informatique au cours des années 1930, ce concept d'algorithme est par conséquent bien antérieur à l'informatique. Mais si on fait démarrer les mathématiques au Ve siècle avant notre ère, il est également antérieur aux mathématiques.

Toujours selon Dowek[7], *un algorithme est une recette qui permet de résoudre un certain problème de manière systématique. Un exemple paradigmatique est la recette de la tarte aux pommes, qui permet de résoudre un problème : faire une tarte aux pommes. Toutefois, dans*

[6] https://www.epi.asso.fr/revue/articles/a1204g.htm (Consulté le 12 octobre 2017)
[7] Ibid.

cet exemple, il est important de distinguer la recette en tant que texte de la recette en tant que pratique. Une recette peut être exécutée et même être transmise de génération en génération, sans être écrite, ni même verbalisée. C'est cette seconde notion qui correspond à la notion d'algorithme. Dès que la recette est écrite ou verbalisée, elle doit être comparée, non à la notion d'algorithme, mais à celle de programme.

Dans le cadre du travail de recherche mis en place au cours de cette thèse, nous proposons de retenir la définition proposée par Modeste (2012) : « *Un algorithme est une procédure de résolution de problème,* [abstraction faite des caractéristiques spécifique que celui-ci peut revêtir] *s'appliquant à une famille d'instances du problème et produisant, en un nombre fini d'étapes constructives, effectives, non-ambigües et organisées, la réponse au problème pour toute instance de cette famille.* » Par exemple, *un algorithme de tri ne résout pas le problème du tri d'un jeu particulier de données mais a pour objectif de trier n'importe quel jeu de données : le problème du tri s'applique à différentes instances, c'est-à-dire à différents jeux de données*[8]. Un algorithme n'est pas un programme. Il est une description abstraite des étapes qui conduisent à la solution d'un problème.

De plus, Modeste (2012) indique quelques précisions supplémentaires sur certains des termes qu'il emploie dans sa définition. Ainsi, dans le domaine de l'informatique, *un « problème » est constitué d'un ensemble d'instances (pouvant être décrites par plusieurs paramètres) et d'une question portant sur ces instances. Une « instance » d'un problème est obtenue en spécifiant des valeurs précises pour tous les paramètres. Les valeurs des paramètres qui constituent une instance sont aussi appelées « données d'entrée » ou simplement « entrées ». La réponse obtenue à l'issue de l'exécution est aussi appelée « sortie de l'algorithme ». Toute exécution de l'algorithme doit se terminer (« un nombre fini d'étapes ») en donnant un résultat correct (« la réponse à la question »).*

1.2 Un programme informatique

Le concept d'algorithme est moins primitif que celui d'un problème. En effet, un même problème, comme trier une liste, peut-être résolu par plusieurs algorithmes. Cependant, ce concept est plus primitif que la notion de programme informatique. En effet, plusieurs

8 http://cache.media.eduscol.education.fr/file/Mathematiques/73/3/Algorithmique_et_programmation_787733.pdf (consulté le 12 octobre 2017)

programmes informatiques peuvent correspondre à un même algorithme. Se pose alors la question de définir le concept de programme informatique.

Selon Chí Thành et Bessot (2010), qui s'appuient sur une définition proposée par Arsac (1995), « *un programme informatique est une liste des instructions auxquelles la machine devra obéir, dans l'ordre de leur exécution […]. On le chargera dans la mémoire de la machine, où elle puisera les instructions au fur et à mesure de leur exécution, à sa propre vitesse*. »

1.3 Le concept de langage de programmation

L'algorithme utilisé par nos cellules pour synthétiser une protéine à partir d'un brin d'ARN[9] messager n'a pas été écrit, ni verbalisé, avant d'être exécuté dans les premières cellules, puis transmis de génération en génération (d'après les connaissances scientifiques actuelles). Cependant, comme le souligne Dowek[10], écrire cet algorithme est une étape obligatoire afin de communiquer le processus à une machine paramétrable. Pour cela, il est nécessaire d'avoir un **langage de communication**. *Un pâtissier n'a pas besoin de connaître les verbes « éplucher », « mélanger », « étaler » ... pour exécuter la recette de la tarte aux pommes, mais il a besoin de créer, ou d'apprendre, ces mots, s'il veut la verbaliser et l'écrire* (Dowek[11]).

Pour écrire des algorithmes, nous pouvons utiliser un **langage naturel** comme le français. Cependant, écrire un algorithme dans un **langage de programmation** est actuellement indispensable pour que celui-ci puisse être implémenté dans un environnement numérique afin qu'il soit exécuté. En effet, nous ne savons pas encore fabriquer des machines qui pourraient tester des algorithmes exprimés dans un langage naturel.

*Les **langages de programmation** permettent de définir les ensembles d'instructions effectuées par l'ordinateur lors de l'exécution d'un programme. Un programme est donc l'expression d'un algorithme dans un langage donné pour une machine donnée.* (Chí Thành et Bessot, 2010, p. 138).

Les **langages de programmation** sont l'intermédiaire entre l'humain et l'environnement numérique. Ils permettent d'écrire dans un langage proche de la machine mais intelligible par l'humain, les opérations que l'ordinateur va être amené à effectuer après l'implémentation de l'algorithme

[9] Acide ribonucléique
[10] https://www.epi.asso.fr/revue/articles/a1204g.htm (Consulté le 10 avril 2018)
[11] Ibid.

On retrouve dans la plupart des **langages de programmation** des instructions permettant de contrôler l'enchaînement des opérations décrite par l'algorithme. Le plus souvent, elles comportent : - des **instructions « conditionnelles »** de types « *Si C alors I* » et « *Si C alors I1 sinon I2* » qui permettent de réaliser la séquence d'instructions I si la condition C est vraie dans le premier cas et de réaliser la séquence d'instructions I1 si la condition C est vraie et la séquence d'instructions I2 sinon, dans le second cas ; - ou encore, des **boucles « conditionnelles »**, de type « *Tant que… Faire…* » qui permet de répéter une séquence d'instructions tant qu'une certaine condition reste vraie, ou de type « *Répéter… Jusqu'à* » qui permet de répéter une séquence d'instructions jusqu'à la condition devienne fausse ; - ou encore, la **boucle de « répétition »** de type « *Pour … Faire …* » permettant de répéter une séquence d'instructions un certain nombre de fois.

On parle aussi de **langage** « **pseudo-code »** lorsque le langage choisi reste proche des langages de programmation. Dans le cas d'un tel langage, nous avons un mélange de « langage mathématique » et d'éléments de langage de programmation permettant d'exprimer des algorithmes sans entrer dans les spécificités d'un langage de programmation.

De plus, selon Chí Thành et Bessot (2010), l'articulation entre programmation et algorithmique se construit en même temps que l'architecture des environnements numériques évoluent.

1.4 L'algorithmique

L'objet de l'algorithmique est la conception, l'évaluation et l'optimisation des méthodes de calcul en mathématiques et en informatique[12]. C'est aussi l'ensemble des règles et des techniques qui sont impliquées dans la définition et la conception de processus systématiques de résolution d'un problème mathématique, permettant de décrire précisément les étapes nécessaires pour résoudre ce problème par une approche algorithmique.

L'algorithmique est par conséquent la **science des algorithmes**. Elle s'intéresse à l'art de construire des algorithmes ainsi qu'à caractériser **leurs validités**, **leurs robustesses**, **leurs réutilisabilités**, **leurs complexités** et **leurs efficacités**.

[12] https://www.universalis.fr/encyclopedie/algorithmique/ (Consulté le 10 avril 2018)

2. Les objectifs des programmes de mathématiques : Seconde générale et classes du cycle terminal de la série scientifique

2.1 La classe de Seconde[13].

2.1.1 Un point sur les principaux objectifs attendus par l'institution

L'objectif principal attendu par les auteurs du programme de 2009 est d'amener les élèves à adopter une démarche scientifique afin de les rendre capables de (extraits du programme) :

- *modéliser et s'engager dans une activité de recherche ;*
- *conduire un raisonnement, une démonstration ;*
- *pratiquer une activité expérimentale ou algorithmique ;*
- *faire une analyse critique d'un résultat, d'une démarche ;*
- *pratiquer une lecture active de l'information (critique, traitement), en privilégiant les changements de registre (graphique, numérique, algébrique, géométrique) ;*
- *utiliser les outils logiciels (ordinateur ou calculatrice) adaptés à la résolution d'un problème ;*
- *communiquer à l'écrit et à l'oral.*

Pour cela, il est attendu que les problèmes posés puissent s'inspirer de situations en lien avec la vie courante ou soient associés à d'autres domaines disciplinaires. Les auteurs souhaitent que les élèves acquièrent de l'autonomie lors de la résolution de problèmes. Par conséquent, les tâches proposées aux élèves doivent leur laisser une prise d'initiative dans leur réflexion et leur démarche.

Dans la continuité du travail fait en classe de mathématiques durant les deux dernières années du collège, les élèves approfondissent au cours de l'année de Seconde leurs connaissances sur les *fonctions*, la *géométrie*, les *statistiques* et les *probabilités*. On attend aussi des élèves qu'ils acquièrent des compétences sur les n*otations et raisonnement mathématiques* ainsi que sur l'*algorithmique*.

Une utilisation d'environnements numériques – calculatrice ou ordinateur, dans le cadre de la simulation et de la programmation – doit faire l'objet de travaux transversaux de la part des élèves. En effet, le programme indique que l'exploitation d'environnements numériques peut aider les élèves à développer *la possibilité d'expérimenter*, et *ouvrir largement la dialectique entre l'observation et la démonstration et change profondément la nature de*

[13] B.O. n° 30 du juillet 2009

l'enseignement. Cette utilisation peut se faire *sous forme de travaux pratiques de mathématiques*.

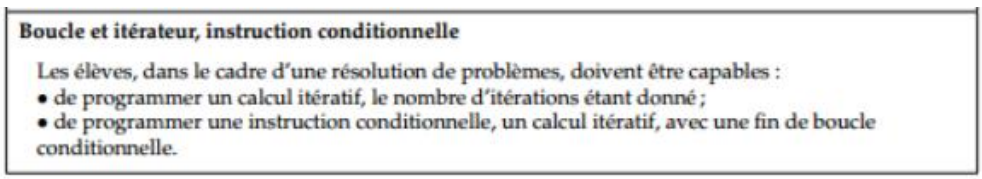
Les capacités attendues en algorithmique sont transversales. Elles sont signalées dans les différentes parties du programme par le symbole ◊ (losange). La démarche algorithmique est considérée comme une composante de l'activité mathématique. Pendant les cours de mathématiques des classes antérieures à la Seconde, les élèves ont déjà eu connaissance de divers algorithmes comme des algorithmes opératoires, l'algorithme d'Euclide, des algorithmes de construction en géométrie, etc.

En Seconde, on attend que les élèves maîtrisent une formalisation en *langage naturel* d'un algorithme afin de pouvoir l'implémenter dans un environnement numérique et de le tester. Il s'agit ainsi *de familiariser les élèves avec les grands principes d'organisation d'un algorithme : gestion des entrées-sorties, affectation d'une valeur et mise en forme d'un calcul*. Dans le cadre de tâches algorithmiques, les élèves doivent (extraits du programme) :

- *décrire certains algorithmes en langage naturel ou dans un langage symbolique ;*
- *en réaliser quelques-uns à l'aide d'un tableur ou d'un petit programme réalisé sur une calculatrice ou avec un logiciel adapté ;*
- *interpréter des algorithmes plus complexes.*

Cependant, il n'est pas attendu que les élèves maîtrisent un langage algorithmique particulier.

On attend des débutants en informatique qu'ils acquièrent des compétences dans le domaine de la programmation sur les instructions élémentaires comme les affectations, les calculs, les entrées et les sorties nécessaires au traitement des problèmes posés et ceci quel que soit le domaine mathématique. De plus, pour compléter ces compétences, les élèves doivent être aussi capables de gérer des tâches nécessitant la mise en place de boucle et d'itérateur, ainsi que des instructions conditionnelles (fig. 1).



Boucle et itérateur, instruction conditionnelle

Les élèves, dans le cadre d'une résolution de problèmes, doivent être capables :
- de programmer un calcul itératif, le nombre d'itérations étant donné ;
- de programmer une instruction conditionnelle, un calcul itératif, avec une fin de boucle conditionnelle.

Figure 1 (Extrait du programme de 2nde : Boucle et itérateur, instruction conditionnelle)

2.1.2 Les objectifs du programme de Seconde sur la transversalité d'un enseignement de l'algorithmique dans les différents domaines mathématiques

Selon les directives du programme, un enseignement spécifique de *l'algorithmique a une place naturelle dans tous les domaines des mathématiques et les problèmes posés doivent être en relation* avec les fonctions, la géométrie, les statistiques, les probabilités et la logique. Cet enseignement de l'algorithmique doit aussi trouver sa place dans l'étude de situations de la vie courante ou en lien avec d'autres disciplines scolaires.

2.1.2.1. Cas des fonctions numériques

Dans le domaine de l'étude de fonctions numériques, les élèves doivent savoir étudier des situations se ramenant à une équation de la forme $f(x)=k$ ou à une inéquation de la forme $f(x)>k$ (en particulier lors de la résolution de problèmes d'optimisation). Pour ce type de problèmes, les élèves doivent déterminer ou justifier l'existence et l'expression de la fonction f correspondant au modèle mathématique schématisant la situation décrite, ou exploiter une représentation graphique ou un tableau de données correspondant à l'évolution du modèle.

Par ailleurs, la résolution de problèmes sur les fonctions *vise aussi à progresser dans la maîtrise du calcul algébrique et à approfondir la connaissance des différents types de nombres, en particulier pour la distinction d'un nombre de ses valeurs approchées*[14].

Il s'agit également d'apprendre aux élèves à distinguer la courbe représentative d'une fonction des dessins obtenus avec un traceur de courbe ou comme représentation de quelques données. Autrement dit, il s'agit de faire comprendre que des dessins peuvent suffire pour répondre de façon satisfaisante à un problème concret mais qu'ils ne suffisent pas à démontrer des propriétés de la fonction[15].

En ce qui concerne la résolution approchée d'une équation, il est demandé aux enseignants de proposer aux élèves des activités sur l'encadrement d'une racine d'une équation grâce à un *algorithme de dichotomie* (fig. 2).

[14] Extrait du programme
[15] Ibid.

CONTENUS	CAPACITÉS ATTENDUES	COMMENTAIRES
[...]	[...]	[...]
Équations Résolution graphique et algébrique d'équations.	• Mettre un problème en équation. • Résoudre une équation se ramenant au premier degré. ◦ Encadrer une racine d'une équation grâce à un algorithme de dichotomie.	Pour un même problème, combiner résolution graphique et contrôle algébrique. Utiliser, en particulier, les représentations graphiques données sur écran par une calculatrice, un logiciel.

Figure 2 (Extrait du programme de 2nde : Résolution graphique et algébrique d'équations)

Nous supposons que cet *algorithme de dichotomie* peut permettre aux élèves de travailler sur des encadrements ou des approximations sans pour autant faire appel à des connaissances mathématiques non accessibles à ce niveau scolaire, contrairement à certaines méthodes d'approximation comme la *méthode de Newton* qui nécessite des compétences sur la dérivée d'une fonction. Ainsi, nous pouvons penser que l'*algorithme de dichotomie* va permettre aux élèves d'afficher un encadrement à ε près d'une solution d'une équation de la forme $f(x) = 0$ dans un intervalle $[a ; b]$, a, b et ε étant saisis par l'utilisateur et la fonction f étant stockée dans une variable informatique.

2.1.2.2. Cas de la géométrie plane

Divers objectifs en lien avec la géométrie plane sont formulés. Ainsi, dans la continuité des apprentissages faits autour des configurations planes, on attend des élèves qu'ils puissent construire des algorithmes simples traduisant numériquement des propriétés géométriques et par conséquent résoudre certains problèmes autour des propriétés des figures planes (fig. 3).

CONTENUS	CAPACITÉS ATTENDUES	COMMENTAIRES
[...]	[...]	[...]
Configurations du plan Triangles, quadrilatères, cercles.	Pour résoudre des problèmes : • Utiliser les propriétés des triangles, des quadrilatères, des cercles. • Utiliser les propriétés des symétries axiale ou centrale.	Les activités des élèves prennent appui sur les propriétés étudiées au collège et peuvent s'enrichir des apports de la géométrie repérée. ◦ Le cadre de la géométrie repérée offre la possibilité de traduire numériquement des propriétés géométriques et permet de résoudre certains problèmes par la mise en œuvre d'algorithmes simples.

Figure 3 (Extrait du programme de 2nde : Propriétés des figures planes)

2.1.2.3. Cas des statistiques et des probabilités

Les statistiques représentent un domaine des mathématiques où on attend des élèves qu'ils soient capables de mener des activités dans le cadre de l'analyse de données ainsi que dans le cadre de l'échantillonnage afin de compléter des compétences acquises au cours des années du collège et d'avoir une première approche des statistiques inférentielles qui feront

l'objet d'un approfondissement pendant les deux années du cycle terminal scientifique (cf. section 1.2).

Nous pouvons observer que les tâches en lien avec l'algorithmique sont plus précisément inscrites dans l'item sur l'échantillonnage d'un point de vue mathématique et une utilisation d'instructions conditionnelles d'un point de vue algorithmique (fig. 4).

CONTENUS	CAPACITÉS ATTENDUES	COMMENTAIRES
[…]	[…]	[…]
Échantillonnage Notion d'échantillon. Intervalle de fluctuation d'une fréquence au seuil de 95%*. Réalisation d'une simulation.	• Concevoir, mettre en œuvre et exploiter des simulations de situations concrètes à l'aide du tableur ou d'une calculatrice. • Exploiter et faire une analyse critique d'un résultat d'échantillonnage.	Un échantillon de taille n est constitué des résultats de n répétitions indépendantes de la même expérience. À l'occasion de la mise en place d'une simulation, on peut : • utiliser les fonctions logiques d'un tableur ou d'une calculatrice, ◇ mettre en place des instructions conditionnelles dans un algorithme. L'objectif est d'amener les élèves à un questionnement lors des activités suivantes : • l'estimation d'une proportion inconnue à partir d'un échantillon ; • la prise de décision à partir d'un échantillon.

Figure 4 (Extrait du programme de 2nde : Echantillonnage)

Certaines activités qui peuvent être mises en place sur les caractéristiques de position et de dispersion, peuvent permettre aux élèves de pratiquer une « étude fréquentiste » d'une série statistique, en particulier lors de simulations de situations concrètes qu'elles soient aléatoires ou non, comme nous le verrons dans le cas des probabilités.

Selon les directives du programme autour de la résolution de problèmes probabilistes, les élèves doivent être capables (extrait du programme) :

- *d'étudier et modéliser des expériences relevant de l'équiprobabilité (par exemple, lancers de pièces,…) ;*
- *de proposer un modèle probabiliste à partir de l'observation de fréquences dans des situations simples ;*
- *d'interpréter des événements de manière ensembliste ;*
- *de mener à bien des calculs de probabilité.*

Les situations mises en place en classe peuvent concerner des expériences à une ou plusieurs épreuves, ainsi que la répétition d'expériences aléatoires pouvant donner lieu à l'écriture d'un ou plusieurs algorithmes simples pouvant faire intervenir des structures de boucles, des instructions conditionnelles, …

Nous observons aussi que les élèves ont à leur disposition pour le calcul de probabilités les arbres de probabilités. En revanche, à ce niveau scolaire, il n'est pas attendu que les élèves aient des compétences sur les probabilités conditionnelles et le calcul d'*espérance mathématique*. Pour les élèves de Seconde, seule une approche fréquentiste d'une simulation aléatoire peut leur permettre d'approcher la notion d'espérance ou de moyenne théorique sans pour autant en comprendre le réel sens de ce que peut représenter cette espérance.

2.2 Les classes du cycle terminal de la série scientifique.

2.2.1 Un point sur les objectifs principaux des programmes du cycle terminal de la série scientifique

Nous faisons le choix de présenter dans une même section les programmes des deux classes présentes dans ce cycle afin de donner une vision plus globale des attentes des auteurs des programmes. Cependant, suivant le schéma retenu pour la classe de Seconde, nous présentons nos analyses en fonction des domaines mathématiques étudiés au cours de ce cycle.

Le cycle scientifique terminal a pour vocation de procurer auprès des élèves *un bagage mathématique solide* afin de les aider à *s'engager dans des études supérieures scientifiques, en les formant à la pratique d'une démarche scientifique et en renforçant leur goût pour des activités de recherche.*

Les programmes visent ainsi le développement de compétences comme la mise en œuvre d'activités de recherche conduites de façon autonome, de mener des raisonnements mais aussi de conduire les élèves à avoir une attitude critique vis-à-vis des résultats obtenus.

Comme pour la classe de Seconde, les *objectifs à atteindre* sont donnés *en termes de capacités*. Ils sont conçus pour favoriser une acquisition progressive des notions mathématiques et leur pérennisation.

2.2.2 Continuité d'un enseignement de l'algorithmique dans les classes du cycle terminal de la série scientifique

Les compétences dans le domaine de l'algorithmique sont rappelées à la fin des programmes et sont les mêmes que celles décrites en Seconde. Celles-ci sont aussi complétées par un item portant sur le raisonnement logique qui peut avoir aussi sa place dans la

construction d'algorithmes quel que soit le domaine mathématique étudié. En effet, les élèves doivent être familiarisés, sur des exemples simples, à :

- *utiliser correctement les connecteurs logiques « et », « ou » et à distinguer leur sens des sens courants de « et », « ou » dans le langage usuel ;*
- *distinguer, dans le cas d'une proposition conditionnelle, la proposition directe, sa réciproque, sa contraposée et sa négation ;*
- *utiliser un contre-exemple pour infirmer une proposition universelle ;*
- *reconnaître et utiliser des types de raisonnement spécifiques : raisonnement par disjonction des cas, recours à la contraposée*, …[16].

2.2.2.1. Cas de l'analyse

Un des objectifs du cycle terminal des classes scientifiques est de doter les élèves d'outils mathématiques utilisant des notions relatives aux suites et aux fonctions permettant de traiter des problèmes relevant de la modélisation de phénomènes discrets ou continus.

Au niveau de la classe de Première, des activités algorithmiques peuvent être introduites autour de travaux sur le second degré, les fonctions numériques, les suites numériques… Le programme insiste aussi sur une utilisation du tableur et de l'algorithmique dans le cadre de l'enseignement des suites. En effet, il est demandé aux enseignants de mettre en œuvre auprès de leurs élèves des algorithmes permettant *d'obtenir une liste de termes de suite*, *de calculer un terme de rang donné*, mais aussi d'étudier *des suites générées par une relation de récurrence* et d'utiliser des algorithmes afin de *traiter des problèmes de comparaison d'évolutions et de seuils*.

Au niveau de la classe de Terminale, la section du programme concernant l'analyse est découpée en deux sous sections : les suites et les fonctions. Dans le cadre des suites, les auteurs attendent que les élèves mettent en place des algorithmes afin d'étudier le comportement asymptotique d'une suite quand cela s'avère être pertinent. Dans le cadre des fonctions, les élèves ont des nouvelles compétences comme la *continuité sur un intervalle*, le *théorème des valeurs intermédiaires* (TVI) et son corollaire : le *théorème de la bijection* (TDB), permettant un travail sur des fonctions strictement monotones sur un intervalle. Ainsi, des

[16] Extrait du *Bulletin officiel* spécial n° 9 du 30 septembre 2010 pour la classe de Premières scientifique.

activités algorithmiques peuvent être réalisées par les *élèves dans le cadre de la recherche de solutions de l'équation* $f(x) = k$. De plus, les auteurs proposent de mettre en œuvre des algorithmes pour déterminer un encadrement d'une intégrale dans le cas où le problème porterait sur une fonction monotone positive.

2.2.2.2. Cas de la géométrie

Tant au niveau de la Première que de la Terminale, les programmes ne donnent aucune piste sur une utilisation possible de l'algorithmique répondant à des questions en lien avec les différentes compétences nouvelles attendues en géométrie qu'elle soit plane (en Première) ou spatiale (en Terminale), ainsi qu'avec l'introduction des nombres complexes au niveau de la classe de Terminale. En effet, les nombres complexes sont vus pour l'essentiel *comme constituant un nouvel ensemble de nombres avec ses opérations propres*. Cette introduction présente ainsi les bases algébriques et les premières applications de certaines propriétés des nombres complexes à la géométrie plane. Elle s'inscrit ainsi *dans la perspective d'un approfondissement lors d'une poursuite d'études*.

2.2.2.3. Cas des statistiques et des probabilités

Au niveau de la classe de Première, la notion de loi de probabilité associée à une variable aléatoire est introduite. Cette introduction permet de *modéliser des situations aléatoires*, afin d'en *proposer un traitement probabiliste et de justifier certains faits observés expérimentalement en classe de Seconde*. Une utilisation des arbres pondérés est développée au cours de cet enseignement afin de *modéliser la répétition d'expériences identiques et indépendantes*. Le programme introduit la *loi binomiale* mais propose aussi aux enseignants de traiter avec leurs élèves des situations aléatoires autour de la *loi géométrique tronquée*. Dans le cadre de l'enseignement de ces deux lois, il est indiqué qu'elles peuvent faire l'objet d'activités en lien avec l'algorithmique. De plus, à ce niveau scolaire, les élèves ont comme nouvelles compétences le calcul de l'espérance, de la variance et de l'écart type de la loi binomiale. De plus, dans le cadre de l'échantillonnage, les élèves peuvent *exploiter l'intervalle de fluctuation à un seuil donné, déterminé à l'aide de la loi binomiale pour rejeter ou non une hypothèse sur une proportion*. Cet intervalle peut être déterminé à l'aide d'un algorithme.

Au niveau de la Terminale, le programme introduit les lois de probabilité à densité, en particulier la *loi uniforme* et la *loi normale*. De même, les élèves ont comme nouvelles

compétences les notions de *conditionnement* et d'*indépendance*. Dans le cadre de ces enseignements, il est proposé de mener des activités algorithmiques afin de *simuler une marche aléatoire*.

2.2.2.4. L'enseignement de spécialité en mathématiques dans les classes de Terminale scientifique

Les thèmes abordés en « enseignement de spécialité mathématique » peuvent permettre la mise en œuvre d'algorithmes en lien avec l'*arithmétique* ou les *matrices et suites* dans divers environnements numériques. En effet, dans le domaine de l'arithmétique, les problèmes peuvent être issus de la cryptographie, mais aussi *relever directement de questions mathématiques,* comme les *nombres premiers* et de propriétés sur la *divisibilité dans Z*. Dans le domaine du calcul matriciel, l'élève doit étudier *des exemples de processus discrets, déterministes ou stochastiques, à l'aide de suites ou de matrices*, en particulier des *marches aléatoires* simples sur des graphes à deux ou trois sommets.

2.2.2.5. L'enseignement de spécialité en informatique et sciences du numérique[17] (ISN) dans les classes de Terminale scientifique

Dans le cadre de l'enseignement optionnel ISN, les élèves sont amenés à travailler autour de quatre grandes parties : *représentation de l'information, algorithmique, langages et programmation, architectures matérielles.* Ainsi, les séquences mises en place en classe sont *construites en combinant des savoirs et capacités extraits des quatre parties du programme*.

Selon le programme, un algorithme est *défini comme une méthode opérationnelle permettant de résoudre, en un nombre fini d'étapes clairement spécifiées, toutes les instances d'un problème donné*. Celui-ci va être implémenté dans un environnement numérique afin de le tester. En effet, les algorithmes construits par les élèves sont *exprimés dans un langage de programmation et exécutés sur une machine ou bien définis de manière informelle*.

De plus, selon le programme, l'apprentissage de la programmation doit aider l'élève *à savoir programmer un algorithme décrit en langue naturelle et d'autre part à comprendre un programme et exprimer en langue naturelle l'algorithme sous-jacent*.

[17] http://www.education.gouv.fr/pid25535/bulletin_officiel.html?cid_bo=57572, consulté le 23/05/17

3. Etat de l'art sur les recherches en didactique des mathématiques dans les domaines de la programmation et de l'algorithmique

À la vue de la présentation des objectifs des programmes décrites dans la première partie de ce chapitre, nous souhaitons revenir sur les apports et les questionnements en lien avec la recherche en didactique dans les domaines de la programmation et de l'algorithmique au regard des apprentissages des élèves débutants en informatique. En effet, l'introduction d'un enseignement de l'informatique autour de la programmation depuis le début des années 80 et, plus récemment, de l'algorithmique dans les classes du secondaire, nous amène à nous questionner sur la possibilité ou non de mettre en place un réel enseignement de l'algorithmique et de la programmation en classe. Et, à analyser ce que cela peut impliquer du point de vue des apprentissages en mathématiques.

3.1 Un retour sur la recherche en didactique autour de la programmation à la fin des années 80

Dans sa thèse, Lagrange (1990) s'est questionné sur les représentations mentales et les processus d'acquisition en informatique chez l'élève débutant en informatique, en particulier dans les classes de lycée. Il résume ainsi son travail de recherche, comme étant une étude des *apprentissages de la programmation par des débutants dans le cadre d'un enseignement informatique ouvert à tout public.* Ainsi, Lagrange cherche *à clarifier les fondements de ce type d'apprentissage,* à *préciser et* à *analyser certains aspects des difficultés que peuvent rencontrer* les élèves*.*

Lagrange observe que de nombreux chercheurs pensaient que la *programmation* était une activité scientifique, et pouvait être enseignée en tant que quelle. Il lui semblait alors urgent *d'aller voir comment ça se passait du côté des apprenants*, plutôt que de se poser des questions de type *métaphysique* sur la nature des apprentissages. Il constate ainsi qu'à l'époque le débat portait sur des *paradigmes de programmation* (Lagrange, 1990). En effet, dans l'enseignement secondaire cela relevait d'un *paradigme impératif* (Ibid.) tandis que dans l'enseignement supérieur, ce qui était mis en avant était un *paradigme fonctionnel* (Ibid.), et que celui-ci se complétait d'autres objets comme la logique. Ainsi, les acteurs de ces enseignements ne se souciaient pas d'opposer algorithmique et programmation, les « règles » reposant sur des *structures algorithmiques* spécifiques.

3.2 Une première approche du fonctionnement des mémoires et des variables dans le cadre de l'algorithmique et de la programmation

En 2005, dans le cadre de sa thèse, Nguyen s'intéresse à une « *étude didactique de l'introduction d'éléments d'algorithmique et de programmation dans l'enseignement mathématique secondaire à l'aide de la calculatrice* ». Sa recherche est menée dans le contexte des programmes de mathématiques des lycées des années 2000. Il analyse le concept d'itération en informatique dans le cadre d'un enseignement de l'algorithmique et de la programmation en classe de Première. Il ancre son travail en didactique des mathématiques en s'appuyant d'abord sur « l'algorithmique », puis il parle d'éléments « d'algorithmique » et « de programmation » dans l'enseignement secondaire sans les considérer comme deux domaines distincts.

A partir des observations de Nguyen sur des tâches autour de la *tabulation de fonctions* avec des élèves débutants en informatique, Lagrange et Rogalski (2017) constatent qu'il serait avantageux de faire que les élèves utilisent dans un premier temps des mémoires, puis organisent l'activation des calculs et des mémoires dans une structure se répétant et finalement imaginent une instruction de répétition. Ils mettent ainsi en valeur le fonctionnement des variables dans une itération. Ils observent que selon Samurçay (1985), les relations entre les éléments d'une itération *sont particulièrement délicates, et même après une première initiation avec un enseignant analysant un programme itératif simple, cette complexité entraîne des difficultés persistantes chez les débutants*. Selon Nguyen, *un des points forts, lié à la construction d'une structure en réponse à un problème sur des objets mathématiques bien identifiés avec un contrôle direct de l'implémentation sur le dispositif invoqué, est que les élèves distinguent bien les opérations d'initialisation de celles constituant le corps de boucle* (Lagrange et Rogalski, 2017, p. 17). Lors de nos ingénieries, nous aurons la possibilité de revenir sur ces problèmes de gestion de mémoires et des variables informatiques (cf. Partie 2).

4. Une vision de la recherche en didactique sur une introduction de l'algorithmique dans l'enseignement des mathématiques au lycée

En 2014, Lagrange écrit dans l'« *Encyclopedia of Mathematics Education* » que l'« Algorithmique » peut être définie comme la conception et l'analyse d'algorithmes (Knuth,

2000). Ainsi, en tant que domaine mathématique, l'algorithmique n'est pas précisément concernée par une exécution humaine des algorithmes, comme par exemple, pour le calcul arithmétique, mais plutôt par une réflexion sur la façon dont les algorithmes sont construits et leur performance. Lagrange observe que la recherche autour des domaines des mathématiques enseignées et de l'exploitation des ordinateurs en classe se concentre le plus souvent sur une utilisation des environnements technologiques comme aide pédagogique. Ainsi, des auteurs comme Papert et Harel (1991), Dubinsky (1999), ou Wilensky et Resnick (1999) proposent une programmation informatique comme un domaine d'activités important à aborder par des élèves débutants les notions et la compréhension de certains concepts issus de divers domaines mathématiques. Cependant, ce domaine de recherche ne considère pas la conception et l'analyse des algorithmes comme un objectif en soi. L'hypothèse est que construire des algorithmes fonctionnant sur des objets mathématiques et les mettre en œuvre dans un langage de programmation dédié serait capable de promouvoir une approche « constructiviste » des concepts scientifiques. Les fonctionnalités du langage, comme la récursivité, les fonctions, … sont choisies afin de soutenir cette approche. L'accès des élèves débutants en informatique à un langage algorithmique formel n'est généralement pas un réel problème car les tâches qui leur sont proposées impliquent généralement des programmes courts avec une structure simple. Les concepts d'algorithmique sont enseignés en amenant les élèves à résoudre des problèmes de calcul à l'aide d'une « *machine à enregistrer* » concrète.

L'introduction de l'algorithmique en 2009 dans les programmes de mathématiques des lycées français fait suite à des tentatives antérieures, qui ne furent pas suivies d'effet dans les classes, d'instituer des tâches en lien avec le domaine de l'algorithmique dans l'enseignement des mathématiques (Lagrange, 2002). Ainsi, tout comme les récentes propositions d'introduire une initiation au « codage informatique » à tous les niveaux de l'enseignement scolaire, cette introduction d'un enseignement de l'algorithmique dans le secondaire réactualise des problématiques qui étaient apparues dans les années 80 avec la création d'une option informatique des lycées où la construction et l'exécution d'algorithmes étaient proposées comme une activité scientifique (Baron, 1989).

Cette (ré)introduction a donné lieu à la publication de nombreuses ressources pour les enseignants, qui sont souvent produites par les enseignants eux-mêmes. Celles-ci constituent des propositions de situations qui sont motivées par des objectifs d'enseignement en classe.

En revanche, à notre connaissance il n'existe que très peu d'études relatives aux élèves, aux difficultés et aux succès qu'ils peuvent éprouver lorsqu'ils sont confrontés à de semblables tâches. Comme le soulignent Lagrange et Guy (2015), ceci attire *notre attention car l'option informatique des lycées avait, quant à elle, donnée lieu à quelques études didactiques portant sur les élèves* (Lagrange et Guy, 2015, p. 46). Celles-ci montraient entre autres *que beaucoup parmi eux rencontraient de grandes difficultés à construire seuls des algorithmes, même simples* (Ibid.).

Ces difficultés ont concerné deux aspects. Tout d'abord, la compréhension des contraintes dues au langage algorithmique utilisé, puis quand cela est nécessaire le découpage en différentes étapes et leurs organisations lors de la construction de l'algorithme. Ainsi, selon Lagrange et Guy (2015), ces difficultés, à quelques exceptions près, sont celles que rencontrent des débutants en informatique quand il passe à la programmation. Des observations informelles confirment qu'elles existent bien dans le cas d'un enseignement de l'algorithmique au lycée.

Les difficultés relatives au premier aspect ont fait l'objet de travaux de recherche, en particulier de la part de Samurçay (1985) et de Lagrange (1992a, 1992b). Des pistes de remédiation ont été alors proposées. Les analyses faites au cours de ces travaux de recherche insistent sur la distance entre les contraintes d'écriture dans le passage d'un langage « naturel » à un langage de programmation. Les pistes de remédiation visent à faire prendre conscience aux élèves de cette distance. Lagrange et Guy (2015) notent que des études sur des activités de programmation par des élèves plus jeunes que les lycéens français ont été réalisées notamment aux Etats-Unis. Ces études ont été pour l'essentielle orientées vers une évaluation de la contribution de ces activités à des compétences générales, notamment autour de la « *pensée procédurale* » (Lagrange & Guy, 2015, p. 46). Crahay (1987), dans une revue de travaux autour de LOGO, qui était est à la fois une conception de l'éducation et une famille de langages de programmation, à la rencontre entre un courant cognitiviste en intelligence artificielle et des théories sur l'apprentissage issues de travaux du psychologue Piaget et de ses conceptions en matière d'éducation, rapporte que les études empiriques faites ont montré des résultats décevants. Selon Lagrange et Guy (2015), Crahay situe notamment les difficultés dans une approche naïve d'un « *apprentissage sans instruction, [...] les enseignants (s'étant) imagines que leur rôle dans l'environnement LOGO était minime.* » (Crahay, 1987, p. 46).

Comme Lagrange et Guy, nous pouvons nous étonner de ce que les ressources qui ont été produites à l'occasion de la (ré)introduction de l'algorithmique ne s'intéressent pas davantage aux élèves, aux difficultés qu'ils rencontrent, et aux stratégies et remédiations que doivent mettre en place les enseignants afin de faire face aux problèmes rencontrés par leurs élèves.

De même, sur le second aspect, Lagrange et Guy (2015) constatent que pour *des tâches simples, le découpage en étapes et leur organisation dans un algorithme peuvent ne pas être triviaux et imposer une prise de distance avec les traitements « manuels » familiers aux élèves.* Ils constatent aussi que celle-ci est *analogue à celle qui est nécessaire pour la compréhension du langage* (Samurçay & Rouchier, 1985).

Les programmes de mathématiques au niveau du lycée fournissent un contexte pour étudier ces aspects, mais aussi pour les aborder dans des perspectives différentes. En effet, selon les objectifs institutionnels, il est précisé dans les programmes scolaires du lycée que : « *l'algorithmique a une place naturelle dans tous les champs des mathématiques et les problèmes poses doivent être en relation avec les autres parties du curriculum (analyse, géométrie, statistiques et probabilités, logique)* », contrairement à l'option informatique des années 80 qui ne liait pas les tâches de type algorithmique à une discipline scolaire. En effet, les objectifs de cette option étaient « *de développer des aptitudes intellectuelles, la créativité, pour l'invention de nouveaux algorithmes, la rigueur de pensée, pour que ces algorithmes résolvent le problème posé* » (Arsac, 1988, p.79). Ainsi, à la lecture des programmes des lycées, nous observons que pour les mathématiques enseignées au cours des trois années de lycée, *l'activité algorithmique des élèves est orientée, non plus vers des aptitudes générales, mais vers des compétences ou connaissances mathématiques dans les champs classiques des mathématiques scolaires* (Lagrange et Guy, 2015, p. 47).

5. Les aspects « outil » et « objet » des algorithmes

Dans cette section, nous souhaitons aborder les aspects « outil » et « objet » des algorithmes, au sens de la dialectique *outil-objet* de Douady (1986). Modeste, Gravier et Ouvrier-Buffet (2010, p. 52) montre qu'un *algorithme n'est pas uniquement un outil pour la résolution de problèmes mais c'est aussi un objet mathématique à part entière*. Pour cela, ils considèrent que regarder un algorithme *en tant qu'objet, c'est s'intéresser aux questions de bon fonctionnement, de domaine de validité, de complexité et de description* de l'algorithme. En revanche, selon ces chercheurs, *regarder l'algorithme en tant qu'outil, c'est s'intéresser à*

l'utilisation que l'on en fait pour résoudre des problèmes. Les aspects résolution de problème, effectivité et formule réfèrent à l'outil algorithme. L'étude des algorithmes en tant qu'*objets* d'apprentissage permet la confrontation entre mode de *pensée algorithmique* et de *pensée mathématique*. *Une préoccupation centrale des mathématiques est de fournir des démonstrations* […], *il est donc inévitable, si l'on regarde l'algorithme comme un objet mathématiques, d'interroger le lien algorithme-preuve* (Modeste, Gravier et Ouvrier-Buffet, 2010, p. 55).

6. Place et rôle des algorithmes dans les programmes et les manuels depuis 2010

En étudiant les algorithmes clés proposés par les programmes et permettant une approche sur son aspect *objet* à travers des questions de preuve, de complexité, d'optimalité, nous pouvons observer que ceux-ci sont en fait peu nombreux. Au niveau de l'enseignement de spécialité de la Terminale scientifique, nous avons l'*algorithme d'Euclide*, où la majorité des manuels proposent une démonstration dans la partie cours, utilisant des suites strictement décroissantes et à termes positifs. De même, au niveau de l'enseignement obligatoire de la Terminale scientifique, les élèves peuvent construire une démonstration algorithmique du « Théorème des valeurs intermédiaires » utilisant la dichotomie et des notions sur les suites adjacentes sachant que celles-ci ne peuvent plus faire l'objet de savoirs attendus de la part des élèves depuis la rentrée 2012.

De même, des problèmes comme la résolution d'équations algébriques du second degré avec l'introduction de l'étude du signe du discriminant, peuvent être aussi vues d'un point de vue algorithmique. Cependant, dans ce cas, l'algorithme mis en place ne nécessite pas un questionnement sur sa preuve et sa complexité, ainsi que son efficacité. Par conséquent, ici l'algorithme reste dans une approche *outil* au sens de Douady. Comme le soulignent Modeste, Gravier et Ouvrier-Buffet (2010), se sont des formules, que l'on peut appliquer de manière systématique et implémenter dans un environnement numérique afin de les tester sans que soit interrogé l'*objet mathématique algorithme*.

Au regard des programmes des lycées depuis 2010, nous constatons que l'algorithmique est un domaine transversal permettant *d'aborder des concepts mathématiques de manière pratique et concrète et de distinguer résolutions théorique et effective de problèmes* (Modeste,

Gravier et Ouvrier-Buffet, 2010, p. 58). Les élèves sont préparés *à des techniques relatives à l'algorithmique : décrire, interpréter et mettre en œuvre des algorithmes simples* (Ibid.).

Dans les manuels, comme nous le verrons dans la deuxième partie, pour chacun des domaines mathématiques où nous expérimentons nos ingénieries, le concept d'algorithme est institutionnellement présent dans le savoir à enseigner, mais son rôle est restreint. En particulier, son lien avec la preuve n'est quasiment pas exploité. Sa place se situe pour l'essentiel dans une approche de type « application numérique », en particulier, dans les domaines d'analyse (cf. la deuxième partie sur l'ingénierie autour de l'*algorithme de dichotomie*) et de probabilités-statistiques lors d'approches « fréquentistes » d'une situation aléatoire (cf. la deuxième partie sur l'ingénierie : *une politique des naissances*).

Comme le soulignent Modeste, Gravier et Ouvrier-Buffet (2010), au regard des tâches algorithmiques proposées par les manuels, l'algorithme *ne vit que dans la niche technique de résolution*, bien que dans le savoir mathématique savant : la *niche preuve* peut être un lieu « écologique » justifiant l'utilisation de l'algorithmique.

7. L'algorithmique et son mode de pensée spécifique

Selon Knuth, nous avons neuf modes de *pensée en mathématique*, dont six sont communs avec ce qu'il nomme la *pensée algorithmique* : la *manipulation de formules*, la *représentation d'une réalité*, la *réduction à des problèmes plus simples*, le *raisonnement abstrait*, les *structures d'informations* et les *algorithmes* (Modeste, 2012, p. 49). En revanche, il lui semble que deux autres modes de pensée sont propres à l'algorithmique : la *notion de complexité* et la *notion d'affectation*. Cependant, Modeste (2010) souligne que selon Knuth *certains textes mathématiques sont extrêmement proches de ce qu'il appelle la pensée algorithmique, en particulier un livre d'analyse constructive : Foundations of constructive Analysis* de Bishop (2012). Knuth remarque que le plus souvent en mathématique, on ne tient pas compte du « coût » de construction.

Ainsi, bien que l'algorithmique fasse appel à des raisonnements communs à ceux mis en place en mathématique, il a aussi son mode de pensée spécifique. On parle alors de *pensée algorithmique*. Dans un domaine en lien avec l'informatique, nous avons l'activité qui est liée à l'instrument. C'est le cas où l'élève doit résoudre une tâche avec l'instrument et son système spécifique de signes. En revanche, dans un domaine mathématique, l'activité consiste à

comprendre la tâche et la solution au niveau mathématique et à travailler avec des signes mathématiques, justifiant mathématiquement la solution obtenue.

De ce fait, comme le souligne Modeste (2012) dans sa thèse, nous pouvons faire l'hypothèse que la *pensée algorithmique* pourrait être vue comme faisant partie de la *pensée mathématique*. Nous avons précédemment évoqué cet aspect. Cependant, toujours selon Modeste (2012), nous pouvons faire l'hypothèse que voir *la pensée algorithmiqu*e comme pensée majeure de l'informatique permet un réel enrichissement de son analyse. La *pensée algorithmique* peut influer sur la *pensée mathématique*, comme nous verrons dans la deuxième partie lors des différentes analyses de nos ingénieries didactiques mises en place dans des domaines mathématiques spécifiques et ceci en fonction du niveau scolaire des élèves. Il est ainsi indispensable d'aborder en parallèle les points de vue, *intra-mathématique* et *extra-mathématique* (au sens de Modeste (2102, p. 58)), pour comprendre le rôle et la place de la *pensée algorithmique* en lien avec la *pensée mathématique*.

Pour rappel, depuis 2010, dans les différents niveaux scolaires français, des algorithmes liés à la programmation sont proposés comme des tâches pour les élèves du secondaire. Cependant, le temps consacré à ces tâches reste court, et par conséquent l'incompréhension des élèves pour les structures algorithmiques et les langages associés semblent être un véritable défi. L'algorithme au sens de Knuth (2010) semble inaccessible aux élèves débutants en informatique sans cette condition préalable. Ainsi, dans la continuité des travaux de Modeste, nous devons poursuivre les diverses études dans le domaine de la recherche didactique en se concentrant sur cette compréhension.

8. Un point sur les travaux en didactique des mathématiques autour de l'informatique et de l'algorithmique

8.1 Un cadre théorique basé sur une modélisation théorique par les conceptions et la dialectique outil-objet pour caractériser l'objet « algorithme »

En 2012, Modeste soutient une thèse en didactique des mathématiques intitulée « *Enseigner l'algorithme pourquoi ? Quelles nouvelles questions pour les mathématiques ? Quels apports pour l'apprentissage de la Preuve ?* ». Il part du constat que le concept d'algorithme est lié à l'informatique mais aussi aux mathématiques et à la preuve. Il soulève ainsi de nombreuses questions didactiques. Dans cette perspective, Modeste propose alors

une analyse épistémologique, ainsi qu'une modélisation, du concept d'algorithme dans le but d'étudier ce qui donne du sens à ce concept. Il pose la question « *A quel besoin répond-il et quelles sont ses spécificités ?* » (Modeste, 2012). De même, il s'interroge sur ce que pourrait être la transposition de ce concept. Pour cela, il construit un certain nombre de situations didactiques permettant de mieux comprendre ce que peut être l'activité algorithmique en classe et quel rôle serait-elle amenée à jouer dans l'activité mathématique.

Afin de répondre à cette problématique, Modeste débute par une première approche épistémologique détaillée du concept en mettant en avant les aspects fondamentaux des algorithmes et de leurs enseignements dans les classes de lycée. Cette démarche permet à Modeste de proposer un modèle de conception pour l'algorithme du point de vue du savoir savant tant en mathématiques qu'en informatique dans le cadre d'une épistémologie contemporaine en tenant compte des différentes formes d'écritures et de représentations que peut prendre un algorithme. Cette approche est complétée par une approche épistémologique historique avec une présentation de la genèse du concept d'algorithme et de son évolution au cours du temps, en particulier depuis la fin du 20ème siècle.

Les résultats de ces recherches sont validés expérimentalement à travers des analyses d'entretiens avec de nombreux chercheurs. Ceci lui permet de mener une étude de la transposition en jeu dans l'enseignement des mathématiques au cours des trois années de lycée. Ainsi, il propose une étude des instructions officielles, d'un certain nombre de manuels scolaires et de ressources disponibles en ligne. Il souhaite mettre en évidence les attendus institutionnels sur un enseignement de l'algorithmique au lycée en décrivant à travers ses observations et ses analyses des documents institutionnels qu'une transposition du concept d'algorithme serait pour l'essentielle orientée vers la programmation et un usage de l'algorithme comme un *outil* (au sens de Douady) d'apprentissage aux vues des choix des auteurs des programmes scolaires et des auteurs de manuels scolaires. Modeste propose dans la dernière partie de sa thèse une caractérisation des problèmes fondamentaux autour de l'algorithme et des perspectives de construction et d'étude de situations didactiques en algorithmique permettant d'envisager une approche de l'enseignement de l'algorithme pas uniquement comme un *outil* pour la résolution de problèmes mais aussi comme un *objet* mathématique à part entière, pour lequel il propose un cadre d'étude précis.

Tenant compte du travail fait par Modeste autour de différents aspects de l'algorithme suivant une *dichotomie outil-objet* en particulier celui qui privilégie le lien que l'algorithme entretient avec la preuve, nous proposerons deux ingénieries (cf. Partie 2), où le travail des élèves consiste à construire des preuves d'une conjecture mais aussi d'un théorème dans deux domaines spécifiques des mathématiques.

8.2 Le cadre de la transposition didactique pour étudier les apports de l'algorithmique à l'enseignement des mathématiques au lycée

Briant (2013) positionne ses recherches sur la place de la *pensée algorithmique*, au sens de Knuth, relativement à la *pensée mathématique*, en considérant l'émergence d'une *pensée algorithmique* lors de la résolution d'un problème issu d'un domaine spécifique des mathématiques : l'algèbre élémentaire, en mettant en place des stratégies de raisonnement et d'approche autour de la construction d'algorithmes implémentables dans un environnement numérique. Ainsi, elle définit une *double transposition* : *de la résolution mathématique au programme informatique* (Briant, Bronner, 2015). Elle part du concept de transposition didactique défini par Chevallard (1985) que Balacheff en 1994 a retravaillée en introduisant le concept de *transposition informatique* qui serait une conséquence des contraintes liées à l'apprentissage de savoirs dans des environnements numériques. Ainsi, *le savoir enseigné dans une situation classique d'enseignement est différent du savoir enseigné avec un ordinateur* (Briant & Bronner, 2015). Nous avons alors le schéma des *Transpositions didactique et informatique* proposé par Chevallard (1982) et Balacheff (1994) permettant d'illustrer cette double transposition (fig. 5).

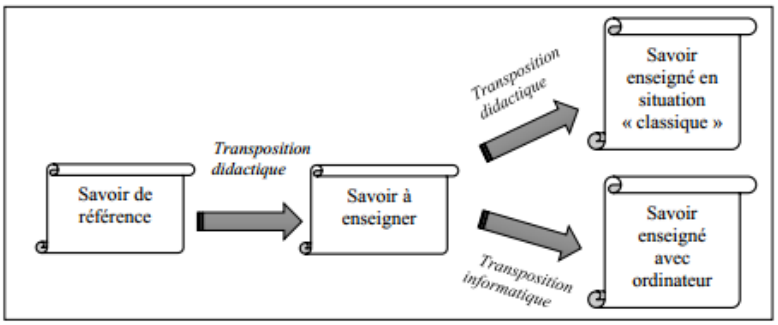


Figure 5 (Schéma des Transpositions didactique et informatique – Chevallard (1982) & Balacheff (1994))

Selon Briant et Bronner (2015), Balacheff (1994) explique qu'*aux contraintes de la transposition didactique s'ajoutent, ou plutôt se combinent, celles de modélisation et d'implémentation informatiques*.

Ainsi, nous avons *deux types de contraintes liées à la transposition informatique : les contraintes de la modélisation computable et les contraintes logicielles et matérielles des supports informatiques. Les premières portent sur la représentation et le traitement interne des savoirs dans la machine et les secondes sur la représentation et le traitement au niveau de l'interface, autrement dit ce qui est « visible » pour le sujet* (Briant & Bronner, 2015).

Partant de ce constat, les recherches de Briant l'ont conduite à étudier l'intégration récente de l'algorithmique dans l'enseignement des mathématiques au lycée, en particulier dans le domaine de l'algèbre élémentaire et plus précisément la résolution des équations algébriques. Elle reprend ainsi *le concept de transposition informatique de Balacheff mais avec une adaptation, tenant compte de la singularité de l'algorithmique* (Ibid.). En effet, lorsque des élèves débutants en informatique ont comme consigne de répondre à une tâche de type « concevoir un algorithme implémentable dans un environnement numérique pour résoudre un problème de mathématique », il émerge alors deux transpositions associées *à des techniques différentes, justifiées par des technologies relevant du domaine mathématique* spécifique*, du domaine informatique,* (Ibid.) mais aussi […] *des deux conjointement* (Ibid.). Briant et Bronner schématisent cette double transposition de la résolution d'un problème mathématique dans un domaine spécifique en vue de la construction d'algorithmes pour une résolution informatique (fig. 6).

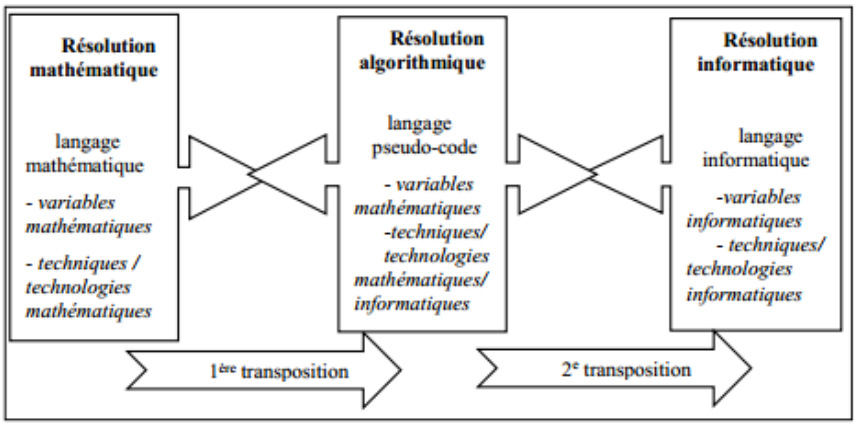


Figure 6 (Résolutions mathématique, algorithmique et informatique – Briant & Bronner (2015))

Ceci permet de caractériser la *démarche algorithmique* chez l'élève selon Briant. Ainsi, Briant montre comment le détour par une *pensée algorithmique* peut permettre *de développer une pensée algébrique et d'asseoir des concepts algébriques relatifs à la notion*

d'équation (Ibid.). Pour illustrer cette caractérisation, Briant (2013) propose d'élaborer une ingénierie didactique expérimentée en classe avec des élèves de Seconde.

Tenant compte des travaux menés par Briant et Bronner, nous souhaitons poursuivre l'étude de la *démarche algorithmique* dans d'autres domaines des mathématiques chez des élèves débutants en informatique, afin d'observer et d'analyser à travers notre cadre théorique les articulations nécessaires entre domaines mathématique et algorithmique sur une tâche relative à un domaine mathématique spécifique.

8.3 Présentation des études de recherche faites en psychologie de la programmation

Dans cette section, nous souhaitons présenter certains travaux issus de la recherche sur les difficultés qui ont pu être observées chez les élèves dans le cadre d'activités de types algorithmique et de programmation, en lien avec la conception d'algorithmes et de leurs implémentations dans des environnements numériques.

8.3.1 Difficultés cognitives en lien avec le concept de variables en itération chez les élèves débutants en informatique

En 1985, Samurçay s'intéresse aux problèmes cognitifs des élèves de Seconde relativement aux variables en itération. La méthode consistait à demander aux élèves de compléter des programmes itératifs dans lesquels des instructions manquaient. Les instructions manquantes étaient de trois types : *l'initialisation de la variable itérative*, *une affectation de la variable itérative dans le corps de la boucle* et *l'état de sortie de la boucle*. Des incompréhensions importantes sur la sémantique des variables sont identifiées. Par exemple, en ce qui concerne l'initialisation, certains élèves pensent que la valeur initiale doit nécessairement être entrée par une instruction de lecture. Tandis que d'autres, systématiquement, initialisent les variables à zéro. Ils sont clairement influencés par la préconception de la façon dont un ordinateur fonctionnerait et par des exemples antérieurs d'algorithmes qui n'auraient pas permis de contester ces idées préconçues. Samurçay conclut qu'il faudrait mettre plus d'études de recherche pour comprendre comment les élèves conceptualisent les notions associées à l'itération et la conception de situations didactiques adéquates.

8.3.2 Difficultés de compréhension autour des procédures récursives chez les élèves débutants en informatique

En 1990, Samurçay et Rouchier étudient chez les élèves débutants l'incompréhension en informatique sur les procédures récursives en distinguant deux aspects : l'*aspect relationnel* et l'*aspect procédural*. Ils conçoivent ainsi des séances d'enseignement ayant pour but d'aider les élèves à construire un modèle relationnel de récursivité, mettant au défi les modèles de procédure existant déjà chez les élèves. Après plusieurs séances, où les élèves prennent connaissance d'un langage graphique comme celui du LOGO, où il n'est pas fait appel au concept de récursivité, Samurçay et Rouchier proposent une première introduction sur les procédures graphiques permettant aux élèves de distinguer les récursivités de type initiale, centrale et finale, puis de les aider à généraliser les structures récursives en transférant les procédures récursives aux objets numériques pour des tâches générant les séquences. Après observation des élèves, Samurçay et Rouchier concluent qu'une introduction du concept de récursivité est un « *détour* » non évident du modèle procédural de l'itération déjà existant et un domaine prometteur pour la recherche.

8.3.3 Difficultés de compréhension des différentes représentations d'objets de base dans un langage de programmation chez les élèves débutants en informatique

En 1995, Lagrange considère la façon dont les élèves des classes de Seconde et de Première comprennent les représentations d'objets de base comme les chaines de caractères, les booléens, … dans un langage de programmation. En analysant les erreurs des élèves dans des tâches impliquant des traitements algorithmiques simples sur ces objets, il constate que les malentendus résultent de l'assimilation des objets « ordinaires » et des traitements. Par exemple, lors de la programmation de l'extraction d'une sous-chaîne à l'intérieur d'une chaîne, les élèves oublient souvent d'attribuer le résultat à une variable. La cause de ce fait en est qu'ils ne sont pas nécessairement conscients de la nature fonctionnelle de l'instruction de la sous-chaîne, étant influencé par l'action « ordinaire » orientée du langage. Un autre exemple est que les élèves ne considèrent généralement pas l'affectation à une valeur booléenne, sans comprendre que, dans un langage algorithmique, les « conditions » sont des entités calculables. Des difficultés similaires observées au cours de cette étude sont analysées en relation avec des obstacles analogues à l'accès au symbolisme algébrique au niveau du

collège. Ainsi, la programmation d'algorithmes simples impliquant ces objets non numériques semble prometteuse pour surmonter ces obstacles.

8.3.4 Une première tentative de remédiation afin d'aider l'élève débutant en informatique autour d'une introduction d'éléments d'algorithmique et de programmation

Comme nous l'avons vu dans la section 3.2, Nguyen (2005) s'interroge sur une introduction d'éléments d'algorithmique et de programmation dans l'enseignement mathématique du secondaire. En effet, il montre que, d'une part, il existe un lien fondamental entre les mathématiques et l'informatique basée sur l'histoire et la pratique récente de ces deux disciplines et que d'autre part, l'*écologie de l'algorithmique et de la programmation* dans les classes de l'enseignement secondaire n'est pas évidente. En mettant l'accent sur l'enseignement et l'apprentissage de la structure de boucle et des notions variables informatiques dans les lycées français[18], Nguyen propose une unité d'enseignement expérimental afin que les élèves de Seconde puissent s'approprier le concept de structure itérative. Pour cela, il choisit de faire en sorte que les élèves créent des représentations appropriées de cette structure en résolvant des tâches de tabulation de valeurs de polynômes à l'aide d'un calculateur dédié, émulé sur ordinateur et basé sur le modèle de calculateur existant dans l'enseignement secondaire avec la capacité supplémentaire de pouvoir enregistrer l'historique des touches utilisées.

Cet enseignement expérimental est conçu comme une genèse de l'*architecture de Von Neumann*. Nous rappelons que l'*architecture de Von Neumann* est un modèle pour un ordinateur utilisant une structure de stockage unique permettant de conserver à la fois les instructions et les données demandées ou produites par le processus de calcul. Elle décompose l'ordinateur en quatre parties distinctes :

- une *unité arithmétique et logique* (UAL ou ALU en anglais) ou encore *unité de traitement* ayant pour but d'effectuer les opérations de base ;
- une *unité de contrôle* ayant pour tâche le « séquençage » des opérations ;
- la mémoire contenant à la fois les données et le programme exécuté par l'unité de contrôle. Celle-ci se divise entre la *mémoire volatile* (dite RAM[19]) contenant les

[18] Et aussi Vietnamien.

[19] *Random Access Memory*

programmes et les données en cours de traitement, et la *mémoire permanente* (dite ROM[20]) permettant de stocker les programmes et les données de base de la machine ;

- des dispositifs d'*entrée-sortie* permettant de communiquer avec l'utilisateur.

Ces différents composants sont reliés entre eux par des *bus*[21].

Ainsi, les élèves conçoivent de nouvelles capacités pour le calculateur, en particulier le principe de mémoire effaçable et la répétition contrôlée afin d'effectuer des calculs et des programmes itératifs à travers l'écriture de messages successifs (au sens de programmes) de machines dotées de caractéristiques différentes. Cela permet alors l'émergence de la notion de variables itératives et de traitements chez des élèves débutants en informatique. Dans le cadre de la théorie des situations didactiques, un milieu et une situation fondamentale peuvent alors être proposés pour la construction de la structure itérative.

8.3.5 Du point de vue de la psychologie de la programmation

Parallèlement à la recherche en didactique des mathématiques enseignées dans le secondaire, des études sont aussi réalisées en *psychologie de la programmation*. Cependant, la plupart des études sur le terrain traitent de la programmation professionnelle et discutent des opportunités et contraintes des langages de programmation et des stratégies de conception pour les experts (Cf. les travaux de Petre et Blackwell (1997)).

Certaines études portent sur la résolution de problèmes de programmation par des débutants en informatique avec des tâches très proches de l'activité des élèves lors des premiers cours d'algorithmique. Ainsi, Rogalski et Samurçay (1990) mettent l'accent sur l'acquisition de connaissances en programmation *comme en témoigne la capacité des élèves à résoudre un problème de programmation*, c'est-à-dire de passer des objets et des situations du monde réel à une implémentation efficace du programme. Rogalski et Samurçay (1990) insistent sur *la variété des activités cognitives et des représentations mentales liées à la conception du programme, à sa compréhension, à sa modification, et au débogage* (ainsi qu'à *la documentation*) *du programme*. Ils soulignent la nécessité pour les débutants d'avoir des modèles mentaux adéquats de représentation et de traitement des données. Ces modèles

[20] *Read Only Memory*

[21] En informatique, un *bus* permet le transport des informations entre différents composants de l'ordinateur. Il sert à relier le microprocesseur à la mémoire centrale, aux dispositifs de stockage ou aux périphériques.

incluent des *schémas statiques* et des *plans*. Les *schémas* sont définis comme des ensembles de connaissances organisées utilisées dans le traitement des données permettant d'atteindre des objectifs à petite échelle. Les *plans* sont des ensembles organisés de procédures dynamiques liées aux *schémas*.

Ainsi, lors de la programmation de la somme des nombres dans une liste de longueur arbitraire, les schémas sont liés à différentes sous-tâches comme la saisie de la liste et le calcul des sommes itérativement partielles, et les plans permettent de définir une stratégie, en séparant les deux sous-tâches ou en fusionnant celles-ci en une seule itération.

Plus généralement, selon Lagrange (Communication personnelle), la recherche dans le domaine de la psychologie de la programmation sur des débutants en informatique complète de manière pertinente la recherche dans le domaine de mathématiques enseignées. En effet, elle permet d'introduire des modèles théoriques de la pensée humaine afin de rendre compte des compétences requises pour élaborer ou comprendre des programmes ou des algorithmes.

9. Conclusion du chapitre 1 : Conséquences des observations faites sur les difficultés observées chez l'élève débutant en informatique

9.1 L'algorithme comme « objet » d'apprentissage en mathématiques

Dans le cadre des programmes de mathématiques, nous avons présenté dans ce chapitre ce qui nous semblait être les principaux objectifs attendus par l'institution avec l'introduction d'un enseignement de l'algorithmique dans les classes du secondaire second cycle et plus précisément en Seconde et dans les classes du cycle Terminal Scientifique.

Comme nous l'avons vu lors de la présentation des programmes autour de l'introduction de l'algorithmique en classe de mathématiques, les pratiques de cet enseignement ont, entre autres, pour objectif de permettre à l'élève débutant en informatique d'acquérir des compétences informatiques sur les notions de variables et de leurs organisations, de mémoires, de traitements des données. Ces points sont aussi complétés d'un travail sur les structures informatiques afin de permettre à l'élève débutant de comprendre qu'un emplacement mémoire correspondant à une donnée invariante lors d'un calcul répétitif peut prendre successivement des valeurs différentes et de bien distinguer, dans le cadre de la

gestion des variables informatiques, les opérations d'initialisation de celles constituant le corps de boucle.

Tenant compte des travaux déjà mis en place dans le domaine de la recherche en didactique des mathématiques et de l'informatique, nous pouvons penser que trouver des façons d'aider des élèves débutants en informatique à accéder à un langage algorithmique ainsi qu'à des modèles mentaux adéquats de représentation et de traitement des données, semble être une condition sine qua non afin de pouvoir aborder des questions centrales comme la *preuve* ou la *modélisation* à l'aide d'algorithmes.

Depuis 2009, le discours institutionnel met en avant la « notion d'algorithme » en l'opposant à la programmation. Ce discours se positionne par rapport aux programmes de mathématiques du secondaire français qui, à partir de 2009, ont introduit une composante « algorithmique » sans proposer autre chose que des exemples qui peuvent être vus comme sous-tendant une activité tournée vers la programmation.

De plus, selon Ouvrier-Buffet, Modeste et Gravier (2010), l'algorithme est un objet méconnu qui est souvent vu comme un objet de l'informatique. Il est en général associé à la programmation. *Cependant, l'algorithme est avant tout un objet des mathématiques. « Avant tout », car la notion d'algorithme précède de beaucoup l'informatique* (Ouvrier-Buffet, Modeste et Gravier, 2010, p. 51).

Ainsi, nous constatons que les attentes institutionnelles dans le cadre d'un enseignement de l'algorithmique en classe de mathématiques semblent se situer pour l'essentiel dans une approche de type « application numérique » à un problème mathématique donné. Tenant compte des travaux menés ces dernières années par des chercheurs comme Nguyen, Modeste et Briant, nous souhaitons approfondir la dialectique « application numérique/analyse numérique » (cf. partie 2) que pourrait favoriser la mise en place d'un enseignement de l'algorithmique dans l'introduction de nouvelles compétences mathématiques.

9.2 Vers l'élaboration d'un nouveau cadre théorique

Nous partons de l'hypothèse de la nécessité de construire un nouveau cadre théorique permettant une étude fine du travail mis en place par l'élève débutant en informatique lors de l'apprentissage de certains nouveaux concepts mathématiques à partir d'une approche

algorithmique afin de diversifier les registres de représentations de l'objet mathématique, mais aussi de donner du sens à l'algorithme comme objet d'apprentissage.

Dans les approches actuelles de l'enseignement des mathématiques au niveau des classes des lycées français, les activités proposées aux élèves impliquent plusieurs domaines d'interaction. Bien que des recherches aient été menées sur la modélisation et de nombreux domaines mathématiques, comme les fonctions, les probabilités, l'arithmétique, …, nous nous interrogeons sur le développement d'activités impliquant des algorithmes et de la programmation informatique dans de nombreux domaines mathématiques des programmes scolaires. Pour nous, afin que ces activités mathématico-algorithmique dans les domaines mentionnées ci-dessus amènent les élèves à comprendre et à conceptualiser, il faut que chaque domaine soit considéré comme un espace particulier permettant un travail scientifique spécifique et des conceptualisations spécifiques. La motivation du chapitre qui suit, est qu'un nouveau cadre théorique approprié, différent de ceux proposés par Modeste ou Briant, est nécessaire afin d'affiner les analyses des différentes expérimentations dans le cadre de trois ingénieries didactiques que nous présentons dans la deuxième partie, chacune étant rapportée à un domaine spécifique des mathématiques.

Chapitre 2 : Construction d'un cadre théorique

Nous présentons dans ce chapitre notre cadre théorique qui va nous permettre d'organiser les analyses de nos ingénieries didactiques mises en place dans des classes de Seconde et du cycle terminal scientifique et qui vont porter sur trois domaines des mathématiques (cf. partie 2).

1. Développement théorique et question

Classiquement, les activités impliquant une tâche mathématique dans plusieurs domaines sont analysées en considérant que les entités impliquées dans la tâche apparaissent sous différentes représentations sémiotiques, chacune appartenant à un champ. C'est le principe dit « multi-représentation ». Parmi les nombreuses approches théoriques des « multi-représentations », nous partons de la prise en compte par Duval de la pluralité des représentations pour un objet donné.

Pour Duval (1999, p. 4), « *there is no other ways of gaining access to the **mathematical objects** but to produce **some semiotic representations** (…) On the other hand, the understanding of mathematics requires not confusing the mathematical objects with the used representations* ». Il souligne aussi que les représentations sont organisées dans des systèmes sémiotiques. Dans un système sémiotique, certaines représentations, appelées « registres », offrent des processus de travail spécifiques (traitements) ainsi que des moyens de passer d'une représentation à une autre (conversion). Duval insiste sur la nécessité de se concentrer spécifiquement sur ces processus de travail à l'intérieur et entre les registres, et des recherches récentes démontrent que les enseignants peuvent ignorer les opportunités offertes par les « multi-représentations » pour comprendre le travail des élèves (Iori, 2018). Dans cette approche « multi-représentationnelle », les activités pour les élèves dans différents domaines sont considérées comme utiles en raison des possibilités offertes pour travailler sur différentes représentations sémiotiques et de les coordonner. Malgré l'importance d'un cadre théorique comme celui proposé par Duval, le point de vue « multi-représentation » nous semble trop réduit à l'aspect sémiotique et ne peut à lui seul donner réellement sens à des activités impliquant plusieurs domaines en interactions et à leurs potentialités. Certains programmes scolaires ont mis l'accent sur les travaux en lien avec les représentations. Les élèves maîtrisent parfaitement les processus de *conversion* et de

traitement, mais cela n'implique pas nécessairement une compréhension profonde des notions en jeu. Par exemple, même lorsque les élèves maîtrisent les quatre représentations classiques de fonctions (expression, symbolique, graphique et tabulaire), les aspects fondamentaux de ces fonctions (correspondance, variation, courbe, …) et leur coordination restent problématiques (Ayalon, Watson, Lerman 2014).

Douady (1996) construit un autre cadre théorique permettant de donner du sens aux activités de coordination de différents domaines (en particulier des domaines mathématiques). Pour Douady, un *cadre* est constitué d'objets issus d'une branche des mathématiques, de la relation entre ces objets, de leurs diverses expressions et des images mentales associées à ces objets. Lorsque l'élève résout un problème, il peut l'examiner dans différents contextes. Passer d'un *cadre* à un autre est important pour que l'élève progresse et que ses conceptions des objets mathématiques évoluent. Cependant, il est parfois difficile de distinguer les approches de représentation et de cadre, en particulier lorsqu'une phase de travail peut être considérée à la fois comme un changement de cadre et de conversion de représentations. En réalité, plutôt que de se contredire, les deux approches se complètent : au-delà de son contenu mathématique, chaque paramètre offre des systèmes sémiotiques spécifiques, et la coordination des paramètres implique également la coordination des systèmes sémiotiques. Pour nous, ce cadre théorique est potentiellement productif dans le sens où, au-delà des représentations, il met l'accent sur les contenus et les raisonnements mathématiques et sur leur coordination entre les différentes branches des mathématiques.

Cependant, beaucoup de domaines impliqués dans les activités des élèves ne sont pas des domaines mathématiques. Par exemple, la notion de fonction est présente dans plusieurs branches des mathématiques avec différentes définitions et propriétés, mais ces définitions et propriétés n'ont de sens que lorsqu'elles sont liées à des pratiques judicieuses dans des domaines non mathématiques : expérience physique quotidienne, mécanismes physiques, etc.

Une autre préoccupation est la prise en compte des instruments dans l'activité mathématique des élèves. Il y a vingt ans, des calculatrices sophistiquées devenaient disponibles pour le travail des élèves et un cadre théorique a été mis au point : l'approche instrumentale de l'utilisation des technologies numériques pour enseigner et apprendre les

mathématiques. Cette approche s'inspirait des travaux de recherche en ergonomie cognitive (Rabardel, Vérillon (1995)). Mais certains chercheurs comme Lagrange, Artigue, ont insisté sur le développement indissociable des connaissances liées à l'instrument et des connaissances en mathématiques dans une genèse instrumentale. Ce constat est important, sinon une approche instrumentale ne serait qu'un cadre psychologique avec peu de connaissances pour l'enseignement des mathématiques.

D'autres chercheurs, comme Mariotti[22], ont également noté que l'utilisation d'instruments et la réflexion qui y est associée, impliquent beaucoup de signes qui, pour un élève, peuvent ne pas avoir de sens mathématique dans l'immédiat. Ils proposent alors l'idée de « *médiation sémiotique* » pour se référer à l'activité en classe nécessaire afin d'assurer la productivité du travail avec des instruments à un niveau sémiotique.

Pour nous, le croisement des connaissances liées à l'instrument et des connaissances liées aux mathématiques, ainsi que la relation entre les signes induits par l'instrument et les signes mathématiques, peuvent être obtenus en coordonnant l'activité dans deux domaines. Dans un domaine en lien avec l'informatique, nous avons l'activité qui est liée à l'instrument. C'est le cas où l'élève doit résoudre une tâche avec l'instrument et son système spécifique de signes. En revanche, dans un domaine mathématique, l'activité consiste à comprendre la tâche et la solution au niveau mathématique et à travailler avec des signes mathématiques, justifiant mathématiquement la solution obtenue.

Suite à ses observations, se pose alors la question du choix d'un cadre théorique qui peut aider à donner du sens au travail de l'élève dans des activités impliquant plusieurs domaines et leurs coordinations, en tenant compte de la dimension sémiotique ainsi que de l'utilisation des instruments et des contenus et des raisonnements propres à chaque domaine ?

2. Vers un nouveau cadre théorique. Pourquoi ?

Tenant compte des observations rapportées dans la section précédente, nous proposons que ce nouveau cadre théorique soit basé pour l'essentiel sur les recherches en didactique des mathématiques, entreprises depuis plus de quinze ans, autour des *Espaces de Travail Mathématique* (ETM).

[22] http://math.unipa.it/~grim/YESS-5/Mariotti_Explantion&Proof.pdf (Consulté le 20 septembre 2017)

Dans un premier temps, les *Espaces de Travail* étaient liés au domaine de la géométrie. En effet, Kuzniak et Houdement présentèrent en 2006 la construction d'un modèle d'*Espaces de Travail Géométrique* (ETG) et de leurs paradigmes. Depuis, l'évolution de la recherche en didactique des mathématiques autour des *Espaces de Travail* a fait sentir la nécessité de développer ce cadre théorique initial à d'autres domaines des mathématiques et à ainsi conduit à l'élaboration d'*Espaces de Travail Mathématique spécifiques* (ETM_s) associés à l'algèbre, l'arithmétique, l'analyse, les probabilités, etc.

Tenant compte de cette évolution sur la recherche autour des ETM, nous sommes partis de l'hypothèse qu'une série de tâches dans le domaine de l'algorithmique pouvait être aussi analysé dans le cadre d'un *Espace de Travail* du type ETG. En effet, la réalisation d'un algorithme comme représentation d'une suite finie d'opérations élémentaires, à appliquer dans un ordre déterminé, à des données, peut être comparée aux différentes étapes à mettre en place lors de la construction d'une figure géométrique donnée. Ainsi, tant pour la réalisation d'un algorithme que pour la construction d'une figure géométrique, le processus associé permet de résoudre un même type de problème.

Cependant, se pose alors le fait de voir l'algorithmique comme un sous-domaine des mathématiques et, par conséquent, de poser comme objectif d'étendre les ETM_s à un ETM_{algo} sans chercher à construire un nouvel *Espace de Travail* propre à l'algorithmique, ou bien de voir l'algorithmique comme une discipline à part entière qui pourrait être en interaction avec des ETM spécifiques. Dans ce second cas, s'avère alors la nécessité de construire et de justifier l'existence d'un *Espace de Travail Algorithmique* comme *Espace de Travail* à part entière.

Nous faisons le choix de favoriser cette deuxième approche. En effet, nous partons du fait qu'une recherche autour d'un enseignement de l'algorithmique dans le système éducatif peut se voir tant d'un point de vue didactique de l'informatique comme l'ont évoqué les travaux de Lagrange, Rogalski, Samurçay, … mais aussi d'un point de vue didactique des mathématiques comme l'a présenté Modeste en signalant que *l'algorithmique est* […] *objet mathématique*. Nous proposons ainsi de créer un nouvel *Espace de Travail* que nous nommons *Espace de Travail Algorithmique* (ETA).

Nous référant aux travaux de Montoya Delgadillo et Vivier (2014), nous souhaitons aussi étudier les articulations des différents domaines mathématiques et algorithmiques lors de

travaux de type *mathématico-algorithmique*. Cette étude théorique va se compléter par l'analyse d'ingénieries didactiques (cf. partie 2) permettant d'étudier les conséquences de ces interactions de domaines tant mathématique qu'algorithmique sur l'élève débutant en informatique et n'ayant pas nécessairement des compétences mathématiques lui permettant de rester dans le domaine spécifique des mathématiques en lien avec l'ingénierie mise en place. Notre étude sur les changements de domaine s'appuie plus spécifiquement sur les ETA en distinguant un domaine initial, ou source, et un domaine d'arrivée, ou de résolution.

Les ingénieries que nous développons et expérimentons dans des classes de lycées français nous permettent aussi de proposer des grilles d'analyse pour l'étude des interactions entre domaines mathématique et algorithmique dans le cadre des ETM$_s$ et des ETA.

De plus, les recherches autour d'un modèle de *paradigmes géométriques* (Houdement et Kuzniak (2006, 2009) et (Kuzniak, 2011)) ont permis d'ouvrir la voie pour une extension à d'autres ETM relatifs à des domaines spécifiques des mathématiques. Nous proposons ainsi de les étendre aux ETA (Laval, 2014, 2016).

Une question récurrente va se poser lors des analyses de nos ingénieries, sur les articulations de ces différents domaines spécifiques tant mathématiques qu'algorithmique au sens d'*Espaces de Travail*. Pour cela, nous nous référerons aux travaux présentés lors des colloques ETM3 (Montréal, octobre 2012), ETM4 (El Escorial, Juillet 2014) et ETM 5 (Florina, juillet 2016).

3. Les Espaces de Travail Mathématique

3.1 Les Espaces de Travail Géométrique et les paradigmes géométriques

3.1.1 Les Espaces de Travail Géométrique

Afin de mieux saisir le cheminement qui a conduit à la construction du modèle des *Espaces de Travail Mathématique*, il nous semble important de présenter l'*Espace de Travail* qui fut le point de départ de ce type de modèle. En effet, le modèle des *Espaces de Travail Géométrique* (ETG) en est l'origine. Il se propose de décrire les différentes formes du travail géométrique effectué par les élèves dans un cadre scolaire. Il place le travail géométrique au centre de la réflexion sur l'enseignement et l'apprentissage. Dans ce contexte, l'institution et les enseignants de mathématiques ont pour objectif de développer un environnement qui doit

permettre aux élèves de résoudre de manière adaptée des problèmes mathématiques de géométrie.

Pour décrire ce type d'activité des élèves, les ETG sont organisés en deux niveaux. Un premier niveau dit épistémologique (cf. fig.7 avec le *plan épistémologique*) où est défini les attentes *a priori* sur l'activité géométrique. A ce niveau, nous identifions trois composantes caractéristiques de l'activité géométrique dans une dimension purement mathématique :

- un *espace réel et local* comme support matériel constitué d'un ensemble d'objets concrets et tangibles qui dans le cas de la géométrie sont des figures ou des dessins ;
- un *ensemble d'artefacts* tels que les instruments de dessin ou des environnements informatiques ;
- un *système théorique de référence* basé sur des définitions, des propriétés et des théorèmes.

La géométrie enseignée ne se réduit pas à un corpus désincarné de propriétés et d'objets se limitant à des signifiants manipulables par des systèmes formels. En effet, elle est avant tout et pour l'essentiel une activité humaine. Il est alors primordial de comprendre comment des groupes d'élèves, mais aussi l'élève en particulier utilisent et s'approprient les compétences géométriques pour une mise en pratique de celles-ci. En conséquence, nous sommes conduits à introduire un deuxième niveau que nous appelons *plan cognitif* (cf. fig. 7 avec le *plan cognitif*) qui représente le travail effectué par l'élève pendant l'activité de résolution d'un problème de géométrie. Il est retenu trois processus cognitifs qui sont en interaction :

- un *processus de visualisation* qui est en relation avec la représentation de l'espace et le support matériel ;
- un *processus de construction* qui est déterminé par les instruments de géométrie utilisés, comme les règles, l'équerre, le compas,… et les configurations géométriques ;
- un *processus discursif* qui permet de produire des argumentations et des preuves.

Cet ensemble de relations est alors visualisé grâce au schéma donné ci-dessous (fig. 7). Il fait aussi apparaître les relations entre les deux niveaux avec les différentes *genèses* :

sémiotique, *discursive*, *instrumentale*.

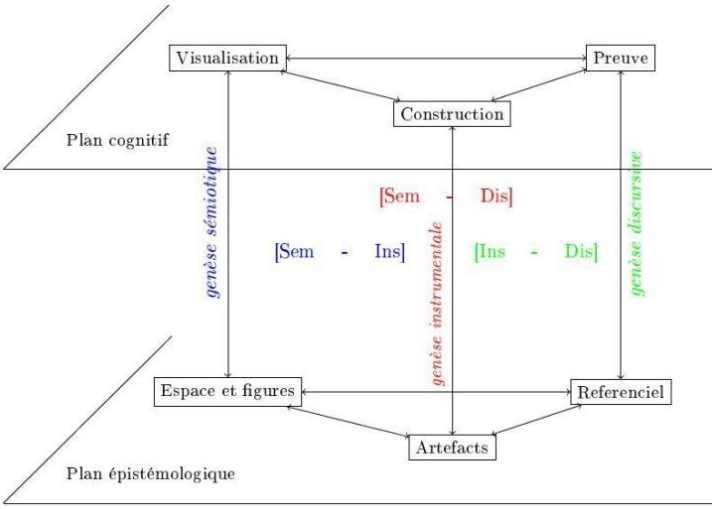


Figure 7 (Les plans épistémologique et cognitif des ETG)

3.1.2 Les paradigmes géométriques

Pour terminer notre présentation sur les ETG, il nous reste à préciser le rôle des *paradigmes géométriques* en lien avec le modèle. En effet, sous le terme « géométrie » *se cache une grande diversité de conceptions et d'idées différentes qui renvoient à ce que Kuhn (1966) appelle des paradigmes*. Selon Kuhn, un *paradigme* désigne un ensemble de croyances, de techniques et de valeurs que peut partager une communauté scientifique.

Au sens littéral, un paradigme *désigne les exemples typiques à partir desquels les élèves apprennent les règles de fonctionnement de certains schémas linguistiques (par exemple, la suite « rosa, rosam, rosae, etc. » est le paradigme des mots qui obéissent à la première déclinaison latine)*[23]. Ainsi, cette notion est *parfaitement adaptée pour caractériser métaphoriquement le genre d'objets qui, dans les sciences, joue le même rôle que les modèles de déclinaison dans les langues, à savoir les exercices types*[24] et elle permet de regrouper les connaissances d'un même groupe qui travaille sur un sujet particulier.

Quand des personnes se situent dans un même paradigme, elles peuvent se comprendre. En effet, elles utilisent un code et des concepts communs qui leur permettent de traiter un problème de manière analogue. En revanche, quand des individus ne se réfèrent pas à un même paradigme géométrique, des méprises peuvent apparaître qui pourront être source de

[23] http://www.universalis.fr/encyclopedie/thomas-kuhn/ (Consulté le 20 juin 2017)
[24] Ibid.

véritables incompréhensions.

Dans le cadre de l'enseignement de la géométrie en classe de mathématiques, Houdement et Kuzniak (2006) identifient alors *trois niveaux de paradigmes géométriques* :

- Niveau « *Géométrie I* » (GI) : une *géométrie naturelle* s'intéressant au monde de la pratique, des figures, des objets réels ayant pour *source de validation la réalité et le monde sensible*. Les types d'arguments proposés par l'apprenant ou l'utilisateur sont autorisés pour justifier une affirmation et convaincre son interlocuteur.

- Niveau « *Géométrie 2* » (GII) : une géométrie construite sur la démonstration, prenant en compte les définitions, les propriétés, les théorèmes, et les relations entre les différentes définitions des objets. C'est une *géométrie axiomatique naturelle* fondée *sur une schématisation de la réalité mais une fois les axiomes fixés, les démonstrations doivent se situer à l'intérieur du système* axiomatique *pour être certaines*.

- Niveau « *Géométrie 3* » (GIII) : une *géométrie axiomatique formaliste*, *qui privilégie essentiellement les relations entre les axiomes définissant les objets sans se préoccuper de leur relation avec la réalité*.

Au cours de la scolarité obligatoire, seuls, les deux premiers niveaux de paradigmes géométriques (GI et GII) sont pris en compte dans l'enseignement de la géométrie.

Cependant, ceux-ci ne sont pas hiérarchisés. En effet, l'un ne peut être considéré comme meilleur que l'autre. Ils n'ont pas les mêmes fonctions et la même finalité : *aspect pratique et technologique* dans le cas de la GI, *aspect axiomatique et logique* dans le cas de la GII.

Selon Kuzniak et Nechache (2014), ces paradigmes servent *de boussole permettant d'identifier la nature épistémologique du travail réellement effectué dans le cadre de l'enseignement*. Ils permettent de caractériser *les ETG mis en place, mais également la circulation du travail géométrique en fonction des différentes entrées*.

3.2 Les Espaces de Travail Mathématique

Selon Kuzniak et Richard (2014), le travail mathématique et son fonctionnement dans le

cadre scolaire sont à l'origine de l'approche par les *Espaces de Travail Mathématique* (ETM).

Le modèle des ETM prolonge celui des ETG. Nous proposons dans cette section de rappeler le principe de l'organisation de ces ETM. Par ETM, nous définissons un environnement pensé et organisé afin de permettre le travail des élèves résolvant des problèmes mathématiques. L'ETM va être une structure évolutive destinée à recueillir les activités mathématiques. Selon Alain Kuzniak & al[25], *de la distinction féconde établie dans les théories de l'activité entre tâche et activité*, il est retenu que cette évolution dépend *des tâches prescrites à un individu et des activités qu'il développera*. *Dans le cas des mathématiques scolaires, ces individus ne seront généralement pas des experts, mais des élèves ou étudiants, confirmés ou débutants*.

Les recherches menées par Kuzniak et Richard sur l'introduction des ETM dans le cadre de travaux en didactique viennent d'une part, *de la mise en évidence de la diversité du travail du mathématicien vu comme l'acteur principal de la progression mathématique* (Kuzniak & al, 2014) et, d'autre part, de *l'idée pédagogique de promouvoir l'activité de l'élève pour le rendre plus apte à développer ses connaissances dans un contexte de résolution de problèmes* (Ibid.). Ainsi dans les deux situations*, le travail mathématique est au cœur de l'évolution, ce qui conduit logiquement à donner à cette notion une place centrale en didactique des mathématiques* (Ibid.). Le travail auquel nous allons nous référer porte sur des activités rationnelles ayant un objectif précis et *pouvant s'appuyer ou non sur l'usage d'un certain nombre d'instruments et d'artefacts spécifiques* (Ibid.).

Dans nos ingénieries, sur le plan mathématique, les activités étudiées sont centrées sur des objets étudiés par des apprentis mathématiciens. Ainsi, en nous référant aux travaux de Kuzniak, nous partons de l'hypothèse que nous devons avoir une double approche en tant que chercheur. En effet, nous devons nous interroger sur les apprentissages de nouvelles compétences mathématiques par des élèves débutants dans des domaines comme l'analyse, la *théorie élémentaire de nombres*, les probabilités et les statistiques mais aussi sur *l'organisation* de ces apprentissages par l'enseignant, *dans le cadre d'un enseignement favorisant le développement du travail mathématique de l'élève* (Ibid.).

[25] Alain Kuzniak, Elizabeth Montoya, Fabrice Vandebrouck et Laurent Vivier (Projet ECOS (2014 - 2016))

3.2.1 Organisation des Espaces de Travail Mathématique

Les *Espaces de Travail Mathématique* ont pour objectif de faciliter la compréhension des enjeux didactiques dans le cadre du travail mathématique dans l'enseignement du premier et du second degré. Cet *espace va désigner un environnement pensé et organisé pour permettre le travail des individus résolvant des problèmes mathématiques* (Ibid.). Ainsi, dans le cas des mathématiques enseignées au primaire et au secondaire, les individus sont des élèves débutants acquérant au fur et à mesure des années de leur scolarité des compétences sur lesquelles ils peuvent s'appuyer pour étudier de nouveaux concepts mathématiques.

Kuzniak (2011) propose d'étendre le modèle des ETG aux ETM, en articulant deux niveaux, l'un *de nature épistémologique, en rapport étroit avec les contenus mathématiques du domaine étudié, et l'autre, de nature cognitive,* concernant *la pensée du sujet résolvant des tâches mathématiques*.

En accord avec Kuzniak et Richard (2014), nous pouvons observer que le travail mathématique résulte d'un processus qui permet *de donner progressivement un sens, d'une part, à chacun des niveaux épistémologique et cognitif et, d'autre part, d'articuler ces deux niveaux grâce à différentes genèses*.

3.2.1.1. Le plan épistémologique et ses composantes

Comme pour les ETG, nous avons trois composantes en interaction, mais non juxtaposées qui doivent être organisées selon un but précis dépendant du domaine mathématique spécifique dans sa dimension épistémologique. Selon Kuzniak, les trois composantes, dépendantes du domaine mathématique étudié, sont constituées :

- d'un *espace réel et local* associé au support matériel ;
- d'un *ensemble d'artefacts* ;
- d'un *système théorique de référence* qui va être entre autres basé sur des définitions, des propriétés et des théorèmes.

De plus, quand l'accent va être mis sur le processus d'apprentissage de l'élève dans une situation didactique, le plan épistémologique va pouvoir être aussi considéré comme un *milieu épistémologique* (Coutat et Richard, 2011).

Les *artefacts et le référentiel théorique restent deux composantes de base de tout plan épistémologique associé à un domaine mathématique particulier* (Kuzniak et Richard, 2014). La composante liée à l'espace et aux configurations géométriques que nous avons présentées dans les ETG (cf. section 1.1.1) doit être modifiée afin qu'elle puisse être étendue à d'autres domaines mathématiques. Selon Kuzniak et Richard (2014), *en accord avec une conception des mathématiques fondées sur des représentations sémiotiques qui va au-delà de la seule considération de systèmes de représentation, il semble pertinent d'utiliser la notion de signe ou representamen, au sens de Peirce*. Nous rappelons que nous entendons par *signe* ou *representamen*, *une « chose » qui en représente une autre que ce soit son objet ou peut-être aussi lui-même* (Kuzniak et Richard, 2014). *Le signe est un representamen, quelque chose qui est mis pour quelque chose, pour quelqu'un. Il crée dans l'esprit de ce dernier un signe équivalent ou plus développé qui est l'interprétant du premier signe. Il est mis pour quelque chose qui est son objet. Mais pas à tous égards, seulement par rapport à une sorte d'idée qui est le fondement du representamen*[26].

Ainsi, en fonction du domaine mathématique étudié, les signes vont pouvoir être des figures géométriques, *des symboles algébriques ou des graphiques, voire des jetons, des maquettes ou des photos dans le cas de problèmes qui mettent en jeu de la modélisation* (Kuzniak à Richard). De même, Kuzniak souligne qu'*à la différence des signes de structure dyadique qui ne retiennent que la relation de référence entre le signifiant et l'objet représenté, l'idée d'un signe qui est aussi sa propre représentation invite à revisiter le processus sémiotique lorsque le travail mathématique est en jeu*. Par exemple, ceci prend tout son sens quand une figure géométrique qui est en elle-même une forme, *est à la fois representamen et modèle de représentation* (Coutat, Laborde et Richard, 2013).

Nous avons ainsi comme plan épistémologique des ETM la représentation suivante (fig. 8) :

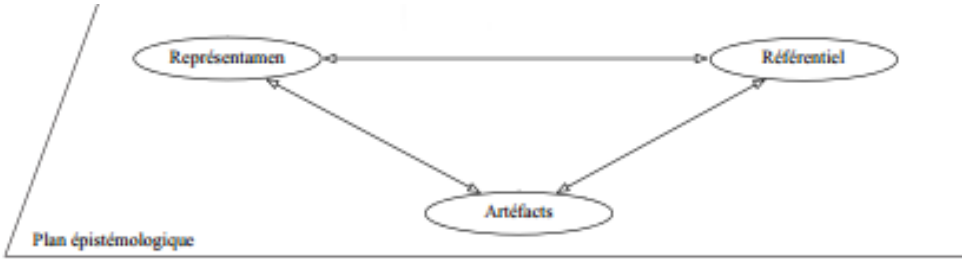


Figure 8 (Plan épistémologique des ETM)

[26] http://www.universalis.fr/encyclopedie/charles-sanders-peirce/ (Consulté le 20 juin 2017)

Comme nous l'avions déjà observé avec les ETG, n'importe quel domaine mathématique enseigné n'est pas réduit à un corpus désincarné de propriétés et d'objets se limitant à des signifiants manipulables par des systèmes formels. En effet, le domaine mathématique concerné est avant tout un champ d'activités humaines. Par conséquent comme le rappelle Kuzniak et Richard, il est essentiel de comprendre comment des groupes d'élèves, mais aussi l'élève en tant qu'individu, *utilisent et s'approprient les connaissances mathématiques dans leur pratique de la discipline*. Il est aussi essentiel de comprendre comment ces élèves vont donner du *sens à tous ces signes et objets tangibles*. De ce fait, comme pour les ETG, nous avons la nécessité de construire un deuxième niveau : le plan cognitif pour l'ETM qui est *centré sur le sujet vu comme un sujet cognitif*.

3.2.1.2. Le plan cognitif

En prenant appui sur les travaux mis en place dans les ETG (Kuzniak, 2011) et les perspectives faites sur les ETM (Kuzniak & Richard, 2014), nous avons donc un deuxième niveau de l'ETM *centré sur le sujet vu comme un sujet cognitif* (fig. 9). Cette ouverture sur le champ cognitif *[va] se faire en étroite relation avec les composantes du niveau épistémologique et, pour rester dans un cadre didactique, il est possible d'adapter l'approche sémiotique de Duval* (2006). Ainsi, Kuzniak propose pour toute activité mathématique trois processus : - un *processus de visualisation* ; - un *processus de construction* ; - un *processus discursif*.

Cependant, dans le cadre d'une extension à des ETM, nous devons préciser le *processus de visualisation*. En effet, selon Kuzniak et Richard (2014), celui-ci *doit être associé à des schèmes et des opérations d'usage des signes dont rien ne prouve a priori qu'ils relèvent tous de la visualisation en tant que telle, même dans une conception étendue de celle-ci* (Fig.9). Ce « *processus de visualisation étendue* » ne peut être associé qu'à *une simple vision ou perception des objets* (Ibid.). En effet, il peut être envisagé comme un *processus de structuration des informations apportées par les diagrammes et les signes. Il nourrit l'intuition des propriétés et il contribue parfois à fonder cognitivement la validité de ces propriétés* (Ibid.). Selon Richard (2004), sous certaines conditions, ce processus de visualisation *peut s'apparenter à un raisonnement de type discursivo-graphique*. Il peut ainsi *s'exprimer à l'intérieur de registres de représentation sémiotique déterminés* (Ibid.).

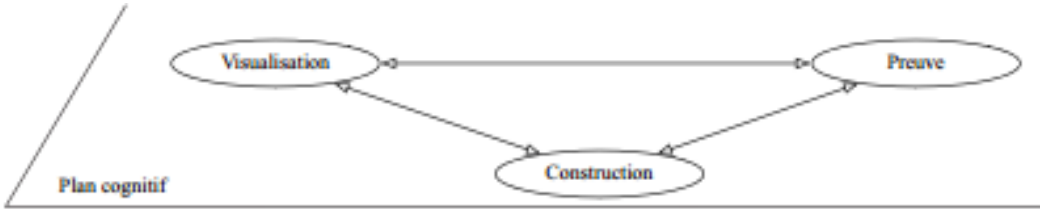


Figure 9 (Plan cognitif des ETM)

3.2.2 Les genèses dans les ETM

Comme il a été souligné de façon implicite lors de la présentation des plans épistémologique et cognitif qui structurent les ETM en deux niveaux (cf. section précédente), nous souhaitons introduire les genèses fondamentales permettant de mieux comprendre les articulations possibles, de façon opératoire, entre les deux niveaux. En effet, ces genèses aident à une meilleure compréhension de la *circulation des connaissances au sein du travail mathématique* (Kuzniak, Richard, 2014).

Selon Kuzniak (2006, 2014) et Richard (2014), qui s'appuient sur les travaux déjà mis en place sur les ETG, nous avons trois genèses :

- une *genèse instrumentale* qui *permet de rendre opératoire les artefacts dans le processus constructif qui contribue à l'accomplissement du travail mathématique* ;
- une *genèse sémiotique* qui est en partie basée sur des *registres de représentation sémiotique.* Elle *donne* du sens *aux objets de l'ETM et leur confère leur statut d'objets mathématiques opératoires*. Elle permet ainsi d'assurer *la mise en relation entre syntaxe, sémantique, fonction et structure des signes véhiculés* ;
- une *genèse discursive de la preuve* qui utilise les définitions, les propriétés, les théorèmes éléments d'un référentiel théorique afin de *les mettre au service du raisonnement mathématique et d'une validation non exclusivement iconique, graphique ou instrumentée*.

Comme pour les ETG, nous introduisons ensuite trois plans verticaux qui peuvent être reliés aux différentes phases du travail mathématique mis en œuvre dans l'exécution de la tâche demandée à l'élève : *découverte et exploration, justification et raisonnement, présentation et communication*. La réalisation effective de ces phases nous permet alors de définir, *un certain nombre de compétences mathématiques cognitives fondées sur la coordination des genèses dans leurs relations avec le plan épistémologique* (Kuzniak & Richard, 2014) (fig. 10).

Nous pouvons observer trois plans d'interactions qui sont identifiés par les genèses qu'ils mettent en œuvre [Sem-Ins] (en bleu), [Ins-Dis] (en rouge) et [Sem-Dis] (en vert) (fig. 10 & 11).

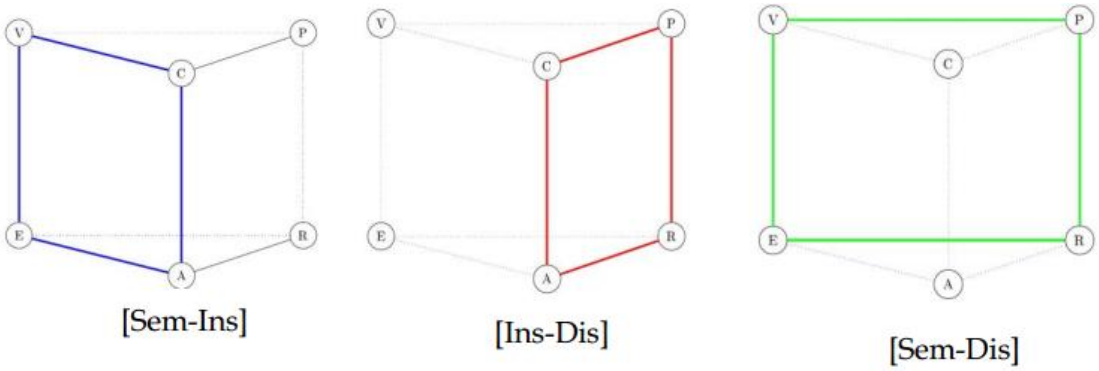


Figure 10 (Les 3 plans d'interactions identifiés par les genèses mises en œuvre dans les ETM)

Le premier s'intéresse à l'identification et l'exploration des objets. Il s'appuie sur les *genèses sémiotique et instrumentale* afin de développer une (ou des) compétence(s) liée(s) à la résolution du problème mathématique.

Le second se base sur le *raisonnement mathématique* fondé sur la justification des réponses au problème mathématique en articulant les *genèses instrumentale et discursive*.

Le troisième et dernier plan est dirigé *vers la communication mathématique des résultats* et se réfère pour l'essentiel sur les *genèses sémiotique et discursive*.

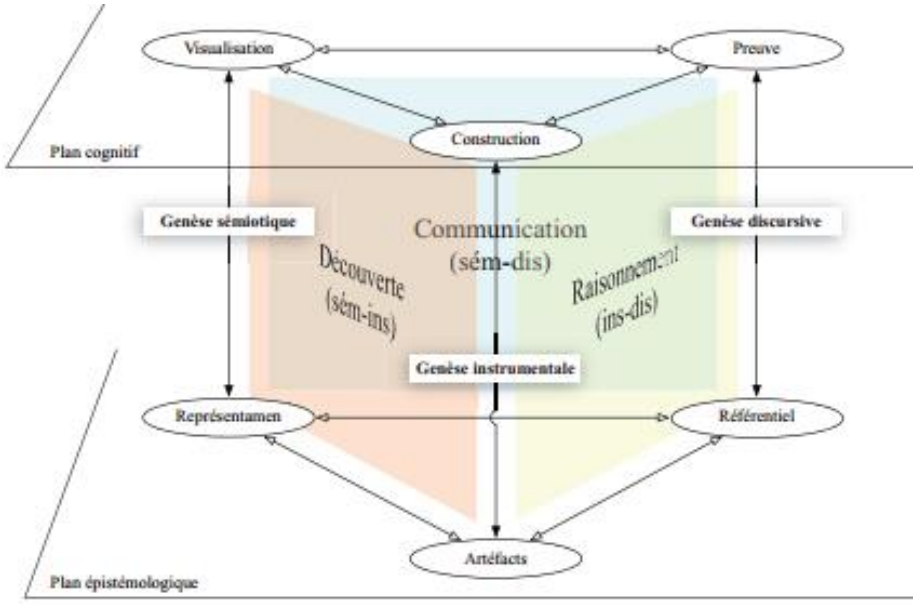


Figure 11 (Plans épistémologique et cognitif et les genèses dans le cas des ETM)

Cependant, comme le soulignent Kuzniak et Richard (2014), les définitions de ces trois plans d'interactions et la description de leurs interrelations dépendent pour l'essentiel du domaine mathématique spécifique auquel est rattachée la tâche demandée.

3.2.3 Notion de paradigmes – ETM de référence – ETM idoines – ETM personnels

Comme pour les *paradigmes géométriques* (cf. section 1.1.2), le cadre théorique des ETM est aussi lié à la notion de paradigme. Cette notion oriente et structure l'organisation des composantes qui, par leurs fonctions différentes, participe *à la spécificité des divers paradigmes en jeu* (Kuzniak, Richard, 2014).

En effet, dans le cas de l'enseignement, un paradigme va s'instituer quand un groupe d'élèves s'accorde *pour formuler des problèmes et organiser leurs solutions en privilégiant certains outils ou certaines formes de pensée* (Ibid.).

Selon Kuzniak, l'*espace de travail « paradigmatique »* tel qu'il est défini par le groupe d'élèves va correspondre à des *ETM de référence*.

De même, pour une institution scolaire donnée, la résolution d'un problème de mathématique va supposer qu'un *ETM idoine* peut être organisé afin d'aider l'élève à s'engager dans la résolution de ce problème. Selon Kuzniak, cet *ETM idoine* doit obligatoirement remplir deux conditions : *d'une part permettre de travailler dans le paradigme correspondant à la problématique visée, d'autre part être « bien construit », dans le sens où ses différentes composantes sont organisées de manière valide*. Le concepteur joue dans ce cas un rôle que nous pourrions comparer à celui que mène l'architecte *qui conçoit un espace de travail pour des utilisateurs potentiels*. En classe, la conception de cet espace va dépendre d'un troisième ETM : *l'ETM personnel* de l'enseignant. Ainsi, toujours selon Kuzniak, lorsqu'un problème de mathématique est posé à un élève, la compréhension des consignes et le traitement mathématique du problème par l'élève vont être conduits dans l'*ETM personnel* de l'élève.

Par conséquent, l'*ETM idoine* doit sans cesse évoluer afin d'être ajusté *aux contraintes locales*.

Tout travail mathématique scolaire peut alors être décrit grâce à ces trois niveaux d'ETM. En effet, selon Kuzniak et Richard (2014), *la mathématique visée par l'institution est décrite dans l'ETM de référence. Celui-ci doit être aménagé par le professeur en ETM idoine pour permettre une mise en place effective dans les classes, où chaque élève* va travailler *dans son ETM personnel*.

Le choix et l'organisation des tâches proposées par l'enseignant à ses élèves sont essentiels dans la constitution de l'*ETM idoine*, afin de donner la possibilité aux élèves de répondre aux questions proposées, *de manière conforme aux attentes institutionnelles décrites de façon plus ou moins explicite dans l'ETM de référence* (Ibid.). Ces choix, ainsi que la gestion des tâches dépendent pour l'essentiel de l'*ETM personnel* de l'enseignant. Kuzniak et Richard complètent cette approche en faisant l'observation que le déroulement de l'activité de l'élève va permettre d'identifier ses *ETM personnels en y repérant d'éventuels sous-ensembles de pratiques stables*.

2.3 Les Espaces de Travail Mathématique spécifiques

Dans les sections précédentes, nous avons présenté l'organisation du modèle théorique définie par les ETG et plus généralement par les ETM. L'application de ce modèle à différents domaines mathématiques nécessite d'introduire le principe d'*Espaces de Travail Mathématiques spécifiques*, associés à des domaines mathématiques *d* particuliers. Nous adoptons la notation ETM_d définie par Kuzniak et Richard (2014) où *d* représente un des domaines mathématiques dans lesquels nous menons nos ingénieries (cf. partie 2). Nous avons ainsi les $ETM_{arithmétique}$, $ETM_{algèbre}$, $ETM_{analyse}$ et $ETM_{probabilités}$.

De plus, pour chacune de nos ingénieries, l'ETM est vu *comme une mise en réseau des diverses fibres que constituent les* ETM_d (Kuzniak & Richard, 2014) mais aussi des *Espaces de Travail algorithmique* (Laval, 2015, 2016) que nous présentons dans la section suivante.

Le travail didactique que nous menons lors des analyses de nos ingénieries didactiques (cf. partie 2), va nous permettre d'étudier les différentes interactions entre des *ETM spécifiques* en articulation avec des *Espaces de Travail Algorithmique* (Laval, 2014, 2015, 2016). Ainsi, selon Kuzniak et Richard (2014), une des questions qui va se poser tout au long de ces analyses, consistera à étudier l'organisation de la *fibration* entre les différentes *Espaces de Travail* ou le

feuilletage des plans.

Ces interactions entre divers domaines mathématiques (Kuzniak et Richard, 2014 et 2016) et algorithmiques (Laval, 2015 et 2016) sont essentielles pour analyser le fonctionnement global du travail mathématique et/ou algorithmique mené en classe par des élèves débutants en informatique et n'ayant pas nécessairement toutes les compétences mathématiques nécessaires, ceci permettant de justifier auprès des élèves l'emploi de l'algorithmique afin de palier à ces compétences mathématiques incomplètes. Ces analyses peuvent aussi nécessiter de prendre en considération des *processus de modélisation* dans le cadre des ETM ou des *Espaces de Travail Algorithmique*, au-delà des seules questions sémiotiques.

4. Les Espaces de Travail Algorithmique (ETA)

Depuis Laval (2015), nous nous intéressons à une transposition des ETG aux ETA. En effet, cette transposition peut être utile pour analyser des tâches spécifiques dans le domaine de l'algorithmique proposées aux élèves de lycée. La mise en place d'ETA va aider *à analyser les positions respectives des élèves et des enseignants, sous l'influence des curricula. En effet, l'ETA idoine prévu par l'enseignant n'est pas nécessairement celui de l'élève* (Laval, 2015)

Comme pour les ETG et les ETM, nous avons deux organisations des ETA. Une première qui est constituée d'un plan épistémologique schématisé ainsi (fig. 11) et une seconde qui est due au fait que l'ouverture sur le plan cognitif des ETA (fig. 12) se fait en interaction avec le niveau épistémologique et les composantes que nous allons présenter.

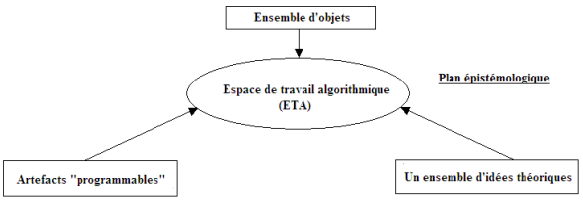


Figure 12 (Plan épistémologique des ETA)

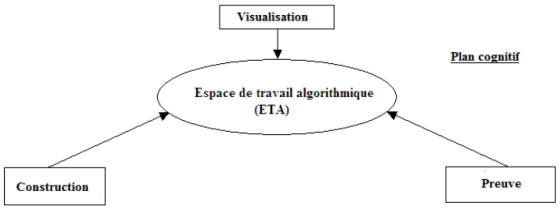


Figure 13 (Plan cognitif des ETA)

4.1 Le plan épistémologique et ses composantes

Comme pour les ETG, nous avons trois composantes en interaction, mais non juxtaposées qui sont organisées selon un but précis dépendant du domaine algorithmique spécifique dans sa dimension épistémologique. Les trois composantes, dépendantes du domaine

algorithmique étudié, sont constituées :

- d'un *ensemble d'« objets »* associé au support matériel ;
- d'un *ensemble d'« artefacts programmables »* ;
- d'un *ensemble d'« idées théoriques »* permettant *de créer et de justifier les algorithmes comme objets pour l'exécution par des artefacts programmables* (Laval, 2015).

De plus, quand l'accent est mis sur le processus d'apprentissage de l'élève dans une situation didactique, ce plan épistémologique peut être aussi considérer comme un *milieu épistémologique* (Coutat et Richard, 2011).

Les éléments des ensembles des « artefacts programmables » et des idées théoriques vont rester deux composantes de base pour tout plan épistémologique associé à un domaine algorithmique *particulier*.

L'*ensemble des « artefacts programmables »* est constitué entre autres de la calculatrice, des objets programmables, d'un langage naturel, d'un langage pseudo-code, des langages algorithmiques, des organigrammes et d'environnements numériques permettant de programmer des algorithmes afin de les tester. *La construction d'algorithmes passant par un langage, plusieurs types de langages peuvent être employés pour exprimer des algorithmes. Cependant, il ne faut pas prendre le mot « langage » au sens technique de programmation. En effet, l'objet de l'algorithmique est de comprendre si l'on peut résoudre tel ou tel problème par le calcul, et si oui, de quelle manière, et à quel coût en termes de temps et de mémoire* (Laval, 2017).

L'*ensemble d'« idées théoriques »* est constitué de concepts de modélisation mathématique et d'algorithmes, de l'étude de la structure de l'algorithme et du choix des variables itératives permettant de construire l'algorithme, ainsi que de l'« effectivité » et du « coût » de l'algorithme. Une première vérification consiste souvent à pratiquer « manuellement » quelques essais. En effet, *le concepteur et l'utilisateur peuvent exécuter l'algorithme « à la main » ou après l'avoir programmé sur l'ordinateur avec quelques données dont ils peuvent connaître le résultat. Si le résultat n'est pas conforme à l'attente, ils prouvent ainsi que l'algorithme est incorrect* (Ibid.).

De nombreux outils formels ou théoriques sont développés afin de décrire les algorithmes, de les étudier et d'exprimer leurs « qualités », et de permettre de les comparer. En effet, pour

décrire des algorithmes, des structures algorithmiques sont mises en évidence : – « structures de contrôle » (séquences, conditionnelles, boucles) ; – « structures de données » (constantes, variables, tableaux, structures récursives du type listes, arbres, graphes). Pour justifier de leur qualité, des notions de correction, de complétude et de terminaison sont mises en place. Pour comparer des algorithmes, une théorie de la complexité des algorithmes est définie. (Ibid.)

L'*ensemble d'« objets »* est le lieu *où un algorithme est un traitement sur des objets qui peuvent être des objets mathématiques ou des objets du monde, notamment lorsqu'il s'agit de résoudre un problème* (Ibid.). Par exemple, l'« algorithme glouton » (*Greedy algorithm*) vise à optimiser le « rendu des monnaies » ou encore de l'« algorithme de Dijkstra » permet de déterminer un chemin optimal dans un graphe.

Comme pour les ETG, les *objets* sont ceux sur lesquels va porter le problème. Ils aident ainsi à distinguer des artefacts qui servent à les manipuler.

Nous verrons dans la partie 2, que pour l'« *algorithme de Kaprekar*[27] » qui se situe dans le domaine de la *théorie élémentaire des nombres*, le graphique représentant le plan épistémologique (fig. 14) peut se présenter ainsi :

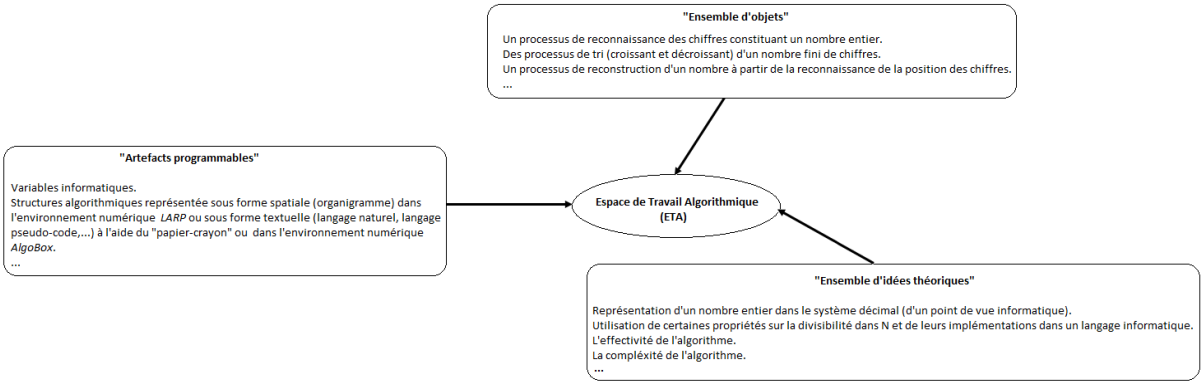


Figure 14 (Plan épistémologique des ETA associé à l'Algorithme de Kaprkar)

[27] Procédure de calcul :
1. Choisir un nombre entier de trois chiffres.
2. Former un nouvel entier obtenu en rangeant les chiffres du nombre choisi à l'instruction 1. dans l'ordre croissant.
3. Former un nouvel entier obtenu en rangeant les chiffres du nombre choisi à l'instruction 1. dans l'ordre décroissant.
4. Calculer la différence des nombres obtenus aux instructions 2. et 3.
5. Recommencer le processus (à partir de l'instruction 2.) avec le nombre obtenu à l'instruction 4. jusqu'à obtenir un nombre déjà obtenu.

De même, pour l' « *algorithme de dichotomie discret*[28] » (cf. Partie 2, chapitre 4), le plan épistémologique peut se présenter comme en figure 15.

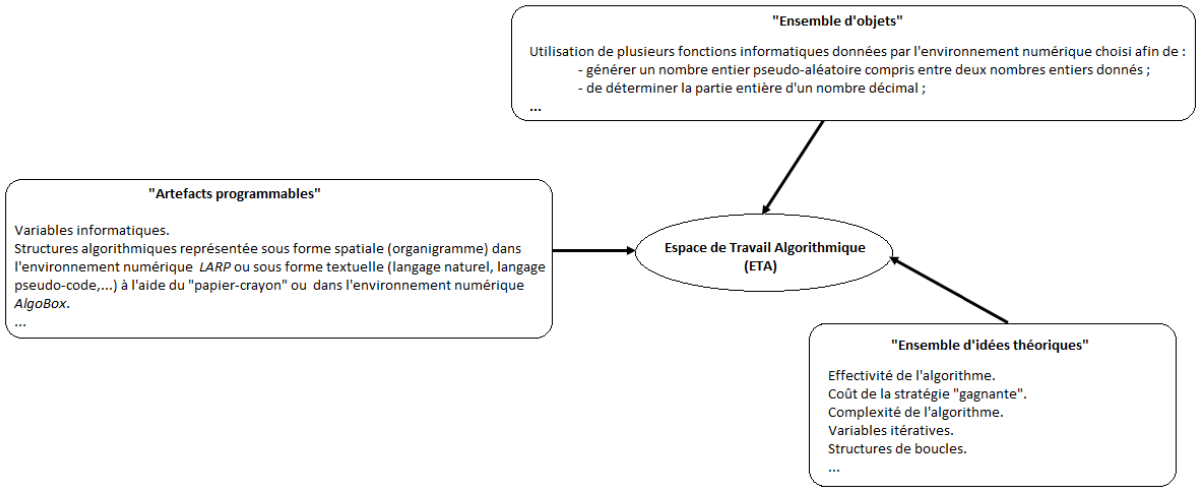


Figure 15 (Plan épistémologique des ETA associé à la « dichotomie discrète »)

Cependant, nous observons une certaine difficulté à décrire les « ensembles d'objets » dans le cadre des ETA. En effet, comme nous l'observerons lors des analyses des ingénieries expérimentées en classe, cet ensemble peut être constitué d'« objets » qui peuvent plus renvoyer à des *Espaces Mathématiques spécifiques* qu'à des ETA. Par exemple, dans le cas de l'« algorithme de Kaprekar », le fait de choisir des nombres entiers constitués d'un nombre fini de chiffres permet d'envisager l'écriture de position des chiffres de ce nombre ce qui renvoie à un $ETM_{arithmétique}$.

4.2 Le plan cognitif

Dans un second temps, nous nous intéressons à une ouverture sur le niveau cognitif des ETA en interaction avec le niveau épistémologique. En effet, l'algorithmique enseignée dans les classes des lycées français ne se réduit pas à un simple corpus qui consisterait à n'aborder que l'aspect théorique et technique de l'algorithmique.

> *La démarche algorithmique est […] une composante essentielle de l'activité mathématique. […] Ce qui est proposé dans le programme est une formalisation en langage naturel propre à donner lieu à traduction […] à l'aide d'un logiciel. Il s'agit de familiariser les élèves avec les grands principes d'organisation d'un*

[28] Stratégie « gagnante » et « rapide » afin de déterminer un nombre entier « secret » compris entre deux nombres entiers connus.

algorithme : gestion des entrées-sorties, affectation d'une valeur et mise en forme d'un calcul[29].

Comme pour les mathématiques enseignées dans le secondaire, l'algorithmique « *enseignée au lycée n'est pas un corpus désincarné de propriétés et d'objets réduits à des signifiants manipulables par des systèmes formels, elle est d'abord et principalement une activité humaine* » (Kuzniak & Richard, 2014, p. 4). Il est essentiel de comprendre comment les élèves utilisent et s'approprient des connaissances algorithmiques dans leur pratique des mathématiques, et comment ils donnent du sens aux structures algorithmiques et aux variables itératives. Il en est de même pour les diverses notions mathématiques utilisées dans les ingénieries (cf. partie 2), comme celui de « nombre entier aléatoire », de « fonction partie » entière, de « calculs de moyennes », d'« espérances mathématiques »,… et les objets tangibles que sont les langages utilisés pour construire des algorithmes.

En prenant appui sur les travaux de recherche mis en place dans les ETG (Kuzniak, 2011) et les perspectives faites sur les ETM (Kuzniak & Richard, 2014), nous proposons alors un deuxième niveau des ETA *centré sur le sujet vu comme un sujet cognitif*. Cette ouverture sur le champ cognitif *[va] se faire en étroite relation avec les composantes du niveau épistémologique et, pour rester dans un cadre didactique, il est possible d'adapter l'approche sémiotique de Duval* (2006). Ainsi, de Duval, nous adoptons pour l'activité algorithmique l'idée de trois processus cognitifs :

- *un processus de visualisation* (Kuzniak & Richard, 2014) en relation avec la représentation de l'algorithme et le support matériel ;
- *un processus de construction* (Ibid.) déterminé par les langages et les instruments utilisés comme les organigrammes, le langage naturel, un langage pseudo-code, un langage de programmation, les ordinateurs, des environnements numériques de type algorithmique, les calculatrices,… ;
- un *processus discursif* (Ibid.) qui permet d'étudier la *terminaison*, la *correction*, l'*efficacité* et la *complexité* de l'algorithme, mais aussi produit des argumentations ainsi que des preuves.

[29] http://media.education.gouv.fr/file/30/52/3/programme_mathematiques_seconde_65523.pdf (Consulté le 7 mai 2017)

Lors de l'exécution d'un algorithme, trois questions clés se posent. En effet, quelles que soient les données : (1) L'algorithme se termine-t-il ? ; (2) L'algorithme résout-il correctement le problème posé ? ; (3) Suivant la « taille » des données, en combien de temps se termine l'algorithme ? Ainsi, la première question pose le problème de terminaison de l'algorithme, la seconde celle du problème de la correction (soit la démonstration) de l'algorithme et le troisième pose le problème de la complexité de l'algorithme. L'objectif est de prouver que l'algorithme résout le problème posé en un nombre fini d'étapes.

4.3 Les paradigmes algorithmiques

Pour compléter ce cadre théorique, nous nous intéressons à la possibilité de transposer les *paradigmes de la géométrie* à des *paradigmes de l'algorithmique*. Nous définissons ainsi trois niveaux de *paradigmes de l'algorithmique* (Laval, 2015).

Un premier niveau algorithmique (AI), où nous avons une approche intuitive des algorithmes issus de situations de la vie réelle. A ce niveau, par exemple, l'efficacité de l'algorithme suit naturellement sa description.

Ce premier niveau se complète d'un deuxième niveau (AII), où il est fait référence à l'axiomatique naturel des algorithmes. A ce niveau, l'efficacité de l'algorithme est utilisée. Son efficacité et sa complexité sont interrogées. L'algorithme peut alors devenir un objet d'un travail mathématique. Ces deux niveaux sont ceux que nous retrouvons dans le cadre d'un enseignement algorithmique dans le secondaire, premier et second cycle.

Dans le cadre d'un enseignement de l'algorithmique dans l'enseignement supérieur et d'une utilisation de théories en lien avec l'algorithmique dans l'industrie informatique, en particulier dans le cadre de la recherche dans ce domaine, nous définissons un troisième niveau (AIII), où l'intérêt de la science algorithmique porte aussi sur le traitement formel des algorithmes. C'est le cas des *machines de Turing*.

Afin de préciser notre choix de transposer les trois niveaux des *paradigmes de la géométrie* à des *paradigmes de l'algorithmique*, nous proposons ci-dessous le tableau (Tableau 1) de comparaison :

Niveaux impliqués	Résoudre un problème géométrique	Etude d'un algorithme
Niveau AI Approche intuitive des algorithmes issus de situations de la vie réelle	**Preuve spatiale**	**Preuve de l'efficacité** de l'exécution de l'algorithme ou raisonnement intuitif
Niveau AII Axiomatique *naturelle* des algorithmes	**Déduction de postulats implicites** Exemple : Les axiomes de base de la géométrie euclidienne	**Preuve mathématique de l'efficacité** Exemple : La preuve de l'algorithme d'Euclide repose sur des propriétés du PGCD et des suites positives d'entiers strictement décroissantes jusqu'à atteindre zéro
Niveau A III Axiomatique formelle	**Différentes géométries** Quelles propriétés initiales sont nécessaires ? Axiome des parallèles	**Calculabilité** Définition équivalente d'une *fonction calculable* Exemple : Machine de Turing

Tableau 1 (Les différents niveaux de paradigmes)

5. Espaces de Travail Mathématique de référence

Pour terminer cette présentation sur notre cadre théorique qui est pour l'essentiel basé sur les modèles des *Espaces de Travail*, il nous semble important de revenir sur certains des *Espaces de Travail Mathématique de référence* que nous mettrons en place dans la partie 2 afin d'aider à optimiser les résultats des analyses de nos différentes ingénieries en fonction des domaines mathématiques auxquelles elles réfèrent.

En effet, notre travail de recherche didactique étudie entre autres les interactions qui peuvent y avoir entre des ETA et des ETM spécifiques.

Par exemple, dans le cas de certains algorithmes du domaine de l'arithmétique, une fois que l'élève a conjecturé un résultat après avoir testé un processus de calcul donné sur quelques exemples pris au hasard, celui-ci va chercher à établir une preuve « complète » de la conjecture. Ainsi, il peut être amené à proposer une preuve empirique consistant à implémenter l'algorithme correspondant au processus de calcul numérique donné dans un environnement numérique, afin de le tester sur tous les cas possibles à l'aide d'une structure de type « Pour ». Cependant, le choix d'expérimenter une telle situation dans des classes de fin du cycle terminal, peut faire que ce même élève ne se satisfasse pas d'une telle preuve et cherche ainsi à diminuer le nombre nécessaire de vérifications à faire pour valider la conjecture en combinant preuve « algébrico-arithmétique » et preuve « informatique ». Ainsi, dans le cas de l'*algorithme de Kaprekar*, comme nous le verrons dans le chapitre 2 de la partie 2, l'apport des $ETM_{algèbre}$ peut être résumé dans le tableau (Tableau 2) ci-dessous :

Paradigme	Un algorithme *qui tourne*	Des réponses *sûres* à des questions
Espace d'objets	Structures algorithmiques, modularité, conversion, représentation des nombres, répétition, tri	Numération décimale (nombre, unité, dizaine, centaine)
Artefacts	Dispositif, langage	Formes algébriques
Référentiel théorique	Contraintes du langage	Théorie élémentaire des nombres

Tableau 2 (Apport des $ETM_{algèbre}$)

De même, dans le cas d'un algorithme associé à une stratégie « gagnante » et « rapide » permettant de déterminer en un nombre minimum de coups, un nombre entier « secret » compris entre deux nombres entiers donnés, le travail de recherche didactique va nous permettre d'étudier les interactions entre des ETA et $ETM_{analyse}$, mais aussi entre des ETA et $ETM_{probabilités}$. En effet, l'élève peut partir de l'hypothèse qu'un type particulier d'algorithme va s'imposer comme répondant à une stratégie « gagnante » et « rapide », implémentable dans un environnement informatique. Nous traduisons cela par le fait que la consigne d'obtenir le nombre entier « secret » en un minimum de propositions peut aider l'élève à interpréter cela comme un objectif de performance comparée. De plus, si le choix de l'algorithme porte sur la *méthode de dichotomie*, nous pouvons supposer que les élèves vont chercher à établir des liens entre calcul de moyenne de deux nombres entiers et partie entière de la moyenne obtenue lors de la programmation sur « machine » de l'algorithme. Dans ce cas, l'apport des ETM spécifiques va se résumer ainsi (Tableau 3) :

	Un algorithme « qui tourne »	Des réponses « sûres » à des questions
Espace d'objets	Structures algorithmiques, partie entière, répétition, comparaison, calcul de moyenne, représentation des nombres	Numération décimale, nombre entier, écriture fractionnaire
Artefacts	Dispositif, langage, variables itératives, corps de boucles	Formes algébriques
Référentiel théorique	Contraintes du langage	Nombre aléatoire, partie entière

Tableau 3

6. Une approche « algorithmique » du concept de modélisation selon Blum et Leiss

Le concept de modèle n'a pas une définition unique. En effet, celle-ci varie suivant la discipline scientifique. Pour établir le lien entre une mathématisation de certaines situations

issues du monde « réel », et l'élaboration de modèles « algorithmiques » associés à ces situations, nous choisissons le cycle de modélisation (Blum & Leiss, 2006) qui prend la forme d'un cercle de résolution de problèmes (fig. 16 « *The modeling cycle according to Blum and Leiß* »). Ainsi, partant d'une situation du monde « réel », épurée et précisée, nous proposons de formuler un modèle « algorithmique », passant par l'élaboration d'un modèle mathématique, afin d'effectuer un traitement mathématique avec production de résultats, suivi d'une interprétation de ces résultats en fonction de la situation réelle d'origine et enfin de valider le modèle en fonction de la pertinence des résultats obtenus.

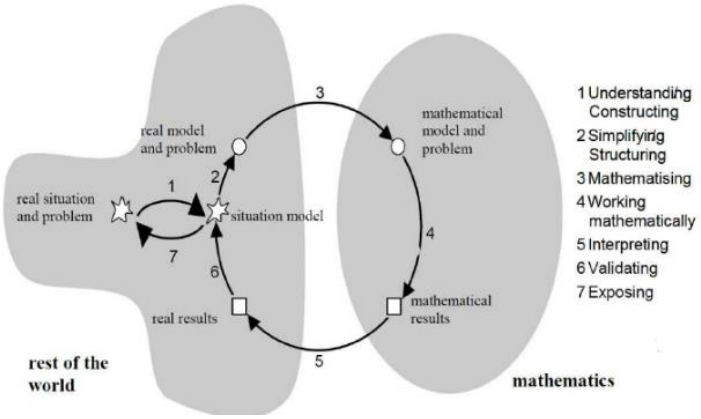


Figure 16 (« The modeling cycle according to Blum and Leiß (2006) »)

7. Conclusion du chapitre 2 : Un retour sur nos choix de cadres théoriques

Dans le cadre de nos ingénieries, bien que le modèle des *Espaces de Travail* corresponde pour essentiel au cadre théorique que nous utilisons pour mener à bien les analyses de nos ingénieries, nous pouvons être aussi conduits à exploiter ponctuellement d'autres cadres théoriques afin d'affiner les déroulements des différentes phases de certaines de nos ingénieries et par conséquent les analyses *a posteriori* qui en découlent.

Ainsi, les outils théoriques que nous mettons en pratique sont essentiellement empruntés aux modèles des *Espaces de Travail Algorithmique* et à des *Espaces de Travail Mathématique* spécifiques comme l'algèbre, l'arithmétique, l'analyse et les probabilités. Cependant, nous devons tenir compte suivant la tâche demandée, du fait que l'élève peut se trouver dans le cas d'une *situation* faisant appel à des éléments de la *Théorie des Situations Didactiques* (Brousseau, 1998), afin d'affiner nos analyses.

De même, les connaissances mathématiques et algorithmiques des élèves étant représentées par différents systèmes sémiotiques, nous pourrons aussi être amenés à prendre en compte les travaux de Duval (2006) sur les registres d'écriture mathématique, ainsi que sur des registres associés à des environnements algorithmiques spécifiques.

Nous complétons nos analyses des différentes tâches que les élèves ont à réaliser, à l'aide des trois genèses : *sémiotique, instrumentale* et *discursive*.

De plus, suivant le domaine mathématique où l'ingénierie est mise en place, nous avons aussi à utiliser les ETM et les ETA comme outil d'analyse du travail tant mathématique qu'algorithmique lors des phases de *cycle de modélisation* (Blum & Leiss, 2005). Ainsi, certaines étapes de modélisation vont pouvoir être nécessaires. Pour cela, nous utilisons comme outil d'analyse du travail mathématique ou algorithmique pour les différentes phases du cycle de modélisation le modèle suivant (fig. 17) :

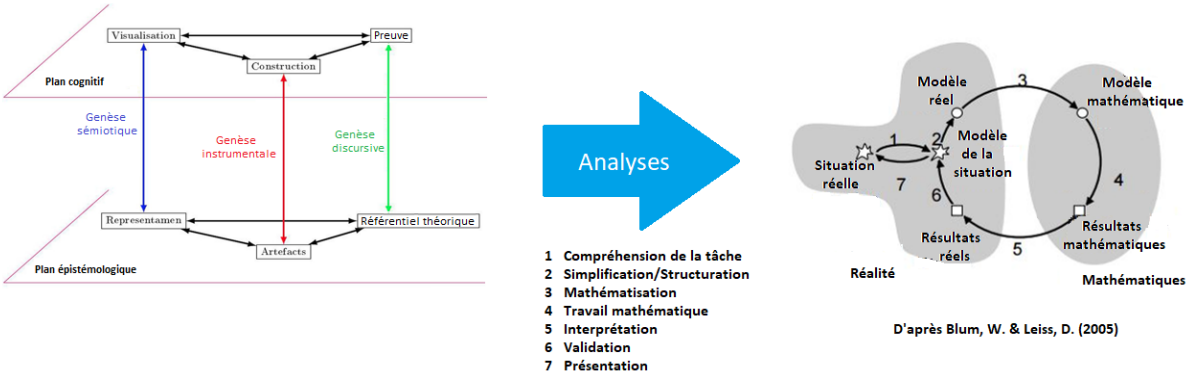


Figure 17 (ETM/ETA associés au cylce de modélisation de Blum et Leiss (2005))

Par exemple, lors de la détermination d'une stratégie « gagnante » et « rapide » permettant de déterminer un nombre entier « secret » compris entre deux nombres entiers donnés, nous pouvons décrire le travail de modélisation nécessaire et associée à la situation algorithmique avec le modèle des ETM et des ETA pour l'analyse de la partie mathématique (Laval, 2016) en ne gardant du graphique de Blum et Leiss que les phases 2, 3 et 4 (fig. 18).

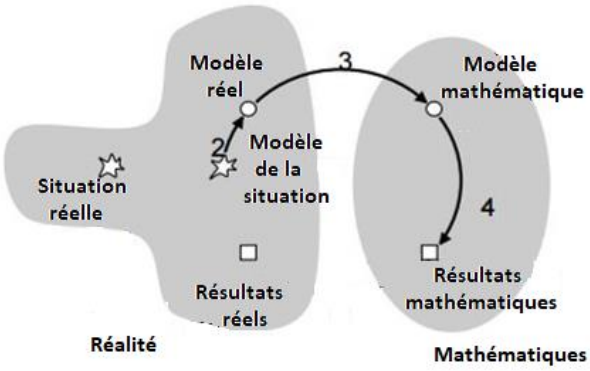


Figure 18 (Un exemple de cycle de modélisation : « dichotomie discète » (Laval, 2016))

Chapitre 3 : Problématique et méthodologie de recherche.

Comme nous l'avons décrit au chapitre 1, le problème se pose avec l'irruption des discours valorisant l'algorithmique et mettant en avant la « notion d'algorithme **»** en l'opposant à la programmation. Ces discours se positionnent par rapport aux programmes de mathématiques dans l'enseignement secondaire français depuis 2009. En effet, l'institution introduit une composante « algorithmique » dans l'enseignement des mathématiques dans le secondaire sans proposer autre chose que des exemples qui peuvent être vus comme sous-tendant une activité tournée vers la programmation. Ainsi, selon Ouvrier-Buffet, Modeste et Gravier (2010), l'algorithmique est *souvent vu comme un objet de l'informatique* que l'on *associe à la programmation*. Ils rappellent aussi que *la notion d'algorithme précède beaucoup l'informatique* et les définitions qu'ils proposent du concept d'algorithme reposent pour l'essentiel sur des « primitives » non définies, au sens de « règles à appliquer » qui les rendent peu opérationnelles, et que celles-ci posent la question de la terminaison comme une contrainte, alors que, d'un point de vue algorithmique, ce serait plutôt le contraire (ex. l'*algorithme correspondant à la suite de Syracuse*[30]). Cependant, Lagrange signale que la programmation est péjorée sans que l'activité qu'elle sous-tend, et sa contribution possible à une compréhension « basique » de l'algorithmique comme domaine, soit analysée. Le discours fait ainsi l'économie de la formalisation, rejetant l'idée de langage sur la programmation, ce qui peut paraitre un contre-sens. Se référant aux propos de Dijkstra (1976, p. 7), Lagrange écrit alors que Dijkstra note que « *tant qu'un algorithme n'est administré que de manière informelle, ce n'est pas un objet approprié pour un traitement formel* » et que par conséquent, « *une notation formelle appropriée* » est nécessaire « *pour étudier les algorithmes comme objets mathématiques* ». Cette notation formelle pour les algorithmes ou « langage » est alors un « *véhicule* » pour aller vers l'abstraction plutôt que pour une exécution dans un environnement numérique.

Dans le cadre de notre recherche, nous mettons en place des ingénieries didactiques. Ces ingénieries doivent nous permettre de tester l'hypothèse que mener un travail approfondi sur des algorithmes par des élèves de Seconde et des classes du cycle scientifique terminal leur

[30] La *suite de Syracuse* est définie de la façon suivante : on choisit un entier naturel non nul, s'il est pair on le divise par 2 sinon on lui applique la fonction $x \rightarrow 3x + 1$, et l'on réitère le processus. On conjecture que, quel que soit le nombre entier non nul choisi initialement, ce processus aboutit à 1.

permet : **(1)** *de faciliter l'apprentissage de nouvelles notions mathématiques* ; **(2)** *de développer des démarches scientifiques dans différents domaines mathématiques* ; **(3)** *d'obtenir des compétences dans des langages de programmation*.

Les situations qui sont présentées à travers ces ingénieries reflètent les choix implicites décrit précédemment. Certes, elles vont mettre en jeu des questions centrales de l'algorithmique, mais d'une part on voit bien la maturité qui va être nécessaire chez les élèves pour qu'elles fonctionnent, et d'autre part que le parti pris de non formalisation des éléments de l'algorithme peut faire que les actions et les conceptualisations des élèves peuvent être peu visibles et objectivables.

1. Spécificités des ETA et ETM permettant d'affiner l'étude du travail de l'élève dans certains domaines mathématiques

1.1 Quelles interactions entre ETA et ETM dans divers domaines des mathématiques scolaires ?

Comme nous avons pu le voir au chapitre précédent, les premières recherches menées par Houdement & Kuzniak (2006) et Kuzniak (2009) sur le modèle des *paradigmes géométriques* et des *Espaces de Travail Géométrique* (Houdement & Kuzniak, 2006 ; Kuzniak, 2009), Kuzniak (2011) ont ouvert la possibilité d'une extension de ces modèles et espaces à d'autres *Espaces de Travail Mathématique (ETM),* relatifs à des domaines comme l'algèbre, l'analyse et les probabilités. La notion de domaine nous semble très importante car c'est à ce niveau que nous pouvons identifier les paradigmes qui caractérisent ces *Espaces de Travail*. En effet, ils permettent de préciser les représentations, les artefacts, les propriétés des objets, ...

Selon Montoya et Vivier (2014), nous distinguons la notion de *domaine* de celle de *cadre* au sens de Douady (1986). En effet, le *domaine* dans notre étude est essentiellement mathématique ou algorithmique contrairement à un *cadre* qui contiendrait des *images mentales des élèves* ainsi que *leurs connaissances*. De ce fait, dans les modèles des ETA/ETM que nous mettons en place, le cognitif est considéré dans le plan cognitif et les images mentales d'un sujet peuvent être considérées à travers la notion d'*ETA/ETM personnel*.

Par ailleurs, nous avançons l'idée d'*Espaces de Travail Algorithmique (ETA)* pour rendre compte des spécificités des artefacts et des règles de référence lors de l'élaboration d'un

travail mathématique que mettent en place des algorithmes permettant à l'élève de structurer sa pensée et d'aborder des savoirs mathématiques qui ne font pas nécessairement partie des compétences de l'élève à l'instant où est mise en place l'ingénierie.

Cette approche a pour conséquence l'étude des questions des articulations possibles entre les différents *Espaces de Travail*. De fait, lorsque nous posons une tâche mathématique à un élève, elle est fréquemment posée à travers plusieurs domaines et surtout il peut arriver qu'au cours du travail mis en place par l'élève, il faille qu'il évolue parmi ces domaines. Par exemple, si nous prenons l'exercice suivant :

> « *Une personne dispose de planches de carton de 110 cm de long et de 88 cm de large. Elle souhaite découper dans ces planches des carrés tous identiques, les plus grands possibles, de façon à ne pas avoir de perte. Quelle est alors la longueur d'un carré et combien peut-elle en découper par planches ?* »

Nous observons qu'il permet un travail dans plusieurs domaines. En effet, un premier *domaine pratique* basé sur le découpage de planches doit aider l'élève à la compréhension du problème. Cette première approche se complète d'un *domaine géométrique*, où le travail peut être réalisé par la construction d'« algorithmes géométriques ». De plus, l'élève doit se situer aussi dans un *domaine des grandeurs*. En effet, il doit aussi considérer les relations entre les objets géométriques. Dans le cadre de la recherche de la solution du problème l'élève va avoir besoin de mettre en place un travail dans le *domaine de l'arithmétique* en échafaudant un calcul de PGCD afin de reconnaitre des conditions de divisibilité. Enfin, le passage à un domaine de type algorithmique va pouvoir s'avérer judicieux pour l'élève dans le cadre de la formalisation et la généralisation de l'« algorithme géométrique ». C'est l'articulation entre ces différents domaines qui, selon notre hypothèse va donner, pour l'élève, du sens à son travail. Par exemple, l'exercice cité mène à un travail de preuve de l'« algorithme géométrique »-optimalité (en effet, c'est bien le carré le plus grand), d'effectivité (il pave bien le rectangle) et de terminaison – qui se mène conjointement dans les différents domaines.

Des études sur les articulations entre ETM de plusieurs domaines ont déjà été introduites lors des trois derniers symposiums ETM. Le rôle de l'enseignant nous semble important et en particulier dans les *ETA/ETM idoines* qu'il va être amené à élaborer pour ses élèves. Ceci inclut aussi les adaptations qu'il devra faire en classe lors des différentes séances. Ainsi, pour chacune de nos ingénieries où il s'agit aussi de situations de classe, nous sommes amenés à utiliser des enregistrements de types vidéo ou audio afin de mieux comprendre et d'analyser

les *ETA/ETM idoines*, ainsi que les responsabilités de chacun des acteurs : l'enseignant et ses élèves, lors de ces changements de domaine de travail.

Ceci implique pour l'enseignant et l'enseignement de nouvelles notions mathématiques, la nécessité d'établir des articulations entre les *Espaces de Travail* dans plusieurs domaines. Nous nous intéressons particulièrement aux articulations entre ETA et ETM et donc à la conception d'espaces de travail *idoines* pour cela.

Afin de favoriser la lecture, conformément aux travaux de Kuzniak (2009), nous rappelons que nous distinguons trois types d'*ETA/ETM* :

- les *ETA/ETM de référence* qui sont définis selon des relations à des savoirs spécifiques en se basant sur des critères relatifs à l'algorithmique et aux mathématiques ;
- les *ETA/ETM idoines* qui dépendent d'institutions et qui sont définis selon les manières dont ces savoirs spécifiques en algorithmique et en mathématiques sont enseignés dans le cadre de ces institutions avec des fonctions propres ;
- les *ETA/ETM personnels* qui dépendent de l'individu (ici l'élève) et qui sont définis par les manières dont cet individu va se confronter à un problème mathématique, l'amenant à utiliser l'algorithmique, et en se basant sur ses propres connaissances et capacités cognitives.

Notre travail implique de considérer ces trois types d'ETA/ETM, tout en centrant notre questionnement sur les *Espaces de Travail idoines*. Nous déterminons aussi des domaines mathématiques particulièrement propices à notre recherche, ce qui nous amène à nous intéresser à des *ETA/ETM de référence* relevant de domaines spécifiques. A ce niveau, nous identifions un choix précis des domaines parmi les plus propices à l'étude des articulations des ETA/ETM.

Ainsi, dans le domaine de la *théorie élémentaire des nombres*, les élèves travaillent sur des questions de divisibilité, de multiplicité, de décomposition d'un entier dans le système décimal, … (cf. Partie 2 – Chapitre 2). De même, dans le domaine de l'*analyse*, les élèves sont amenés à utiliser de façon intuitive ou formelle suivant le niveau scolaire où ils se trouvent, des concepts mathématiques comme la *continuité* et la *monotonie* d'une fonction sur un intervalle pour la détermination de conditions d'existence et d'unicité d'une solution d'une équation, mais aussi la connaissance de la notion de fonction partie entière afin de déterminer

une valeur entière d'une moyenne de deux nombres entiers,… (cf. Partie 2 – Chapitres 3 et 4). Dans d'autres domaines, comme celui des *probabilités et des statistiques*, la connaissance de calculs de fréquences, de probabilités, de moyennes, d'espérances… mais aussi la mise en pratique de lois discrètes font partie des savoirs ou savoir-faire nécessaires pour les élèves afin de répondre aux attentes de l'institution représentée par l'enseignant (cf. Partie 2 – Chapitre 5).

Dans le « domaine » de l'*algorithmique*, nous pouvons observer quelles compétences doivent acquérir les élèves afin de bâtir des algorithmes nécessitant des connaissances sur les structures de boucles, les variables itératives, le tri,… mais aussi une utilisation adéquate de divers environnements numériques autour de l'algorithmiques et de la programmation.

1.2 Hypothèses de travail sur les ETA-ETM idoines

Pour chacun des *ETA-ETM idoines* nous nous référons aux programmes en fonction des domaines mathématiques et algorithmique nécessaires à la bonne conduite des tâches demandées.

Dans le cadre du domaine mathématique correspondant au domaine de la *théorie élémentaire des nombres*, de nombreuses notions mathématiques peuvent être étudiées à l'aide de l'algorithmique, comme l'*algorithme d'Euclide*, des algorithmes sur la formulation d'un test de primalité, des algorithmes de changement de bases numériques, des algorithmes associés à la cryptographie, … Dans le cas d'algorithmes comme celui d'Euclide, l'élève de Terminale Scientifique a les connaissances mathématiques afin de démontrer la validité de cet algorithme à l'aide de savoirs qui sont étudiées en cours d'enseignement spécifique. Ainsi, les *ETA* et *ETM* idoines dépendent d'institutions, comme les programmes, les manuels, les documents d'accompagnement, etc. Ils sont définis selon les différentes manières que les savoirs spécifiques sont enseignés dans le cadre de ces institutions avec leurs fonctions propres.

Dans le cadre du programme mathématique correspondant au domaine de l'analyse, l'algorithmique peut par exemple contribuer aux apprentissages sur la détermination d'un zéro ou la recherche d'un antécédent d'une fonction sur un intervalle donné. En effet, après avoir justifié l'existence d'un unique zéro sur un intervalle pour une fonction donnée, soit par

des considérations graphiques au niveau Seconde ou une utilisation du calcul différentiel complétée d'une observation graphique au niveau Première Scientifique ou encore une validation des conditions du *théorème des valeurs intermédiaires* et de son corollaire au niveau Terminale Scientifique, l'élève peut ensuite à l'aide d'un algorithme d'approximation déterminer une valeur approchée ou un encadrement de ce zéro. Ici aussi, l'*ETA-ETM idoine* va dépendre des programmes et par conséquent des savoirs spécifiques enseignés dans le cadre institutionnel avec ses fonctions propres.

Dans le domaine des probabilités et des statistiques, l'algorithmique peut contribuer à favoriser l'approche « fréquentiste » de phénomènes aléatoires, à travers la détermination de fréquences relatives des caractères représentatifs de ces phénomènes, afin d'aider à la compréhension de la notion de probabilité théorique d'un événement, et l'observation de moyennes statistiques aidant ainsi l'élève à la prise de conscience de la notion d'espérance d'une variable aléatoire (Parzysz). La mise en place d'une simulation aléatoire dans un environnement numérique peut être un moyen pratique d'obtenir des séries statistiques de taille suffisante pour observer la convergence des fréquences relatives et des moyennes statistiques. Par ailleurs, la simulation d'un phénomène aléatoire issu de situations extra-mathématiques suppose une modélisation de cette situation. Le modèle ainsi obtenu peut alors être mis à profit pour fonder le calcul de probabilités théoriques (Kiet, 2015). Ici aussi, les *ETA-ETM idoines* vont dépendre des programmes et des savoirs spécifiques enseignés dans un cadre institutionnel avec ses fonctions propres.

Quels que soient les domaines mathématiques, leurs articulations avec un domaine algorithmique vont dépendre aussi du choix des environnements algorithmiques qui peuvent être multiples, tant sur le plan des langages utilisés : naturel, pseudo-code, informatique, … que sur les choix de représentations des algorithmes : spatial (organigramme), textuel, … ainsi que dans le choix des environnements numériques : algorithmique, tableur, calculatrice, … Les apprentissages en algorithmique et en programmation nécessitent des savoirs sous-jacents comme dans le cas des algorithmes complexes qui peuvent nécessiter la mise en place de plusieurs procédures mais que la plupart des environnements algorithmiques utilisés dans les lycées ne permettent pas et par conséquent obligent l'élève à une progressivité dans la compréhension des différentes étapes de ces algorithmes afin de pallier cette impossibilité de mettre en place des sous-algorithmes.

2. Méthodologie pour la conception et la mise en œuvre d'ingénieries didactiques

Comme nous le disions dans la section précédente, pour les ingénieries didactiques que nous avons bâties et analysées dans la partie 2, nos choix se sont portés sur des concepts mathématiques issus de divers domaines mathématiques : la *théorie élémentaire des nombres*, l'analyse, les statistiques-probabilités pour les *Espaces de Travail Mathématique* de référence, et l'utilisation de variables informatiques, d'instructions conditionnelles et de structures itératives[31] pour les *Espaces de Travail algorithmique*.

2.1 Notre méthodologie

Nous souhaitons étudier et voir s'il est possible d'améliorer et de développer des compétences mathématiques chez les élèves sur des domaines particuliers des mathématiques à travers une utilisation de l'algorithmique. Pour cela, nous revenons au cadre théorique sur les *Espaces de Travail*, afin de déterminer les interactions et les articulations possibles entre ETM spécifique et ETA.

Nous souhaitons aussi mettre en évidence les différentes relations qui peuvent exister entre ces ETM et ETA, et analyser les points forts dans les *Espaces de Travail de référence*, et dans les *Espaces de Travail idoine* en fonction des problèmes que vont rencontrer les élèves afin d'étudier ensuite ce que cela donne au niveau des différentes dimensions des genèses et des interactions au niveau des élèves. Ainsi, nous pouvons observer les dynamiques qui doivent exister dans les ETM/ETA sur les différents points étudiés.

Pour cela, il est nécessaire que nous allions voir ce qui se passe dans les ETM idoines et aussi d'accéder au travail personnel des élèves, d'où la nécessité de développer des ingénieries didactiques dans un certain nombre de domaines mathématiques.

Ainsi les ETM et les ETA servent de guide d'observation méthodologique pour définir les objets, de bien définir les relations entre ETM et ETA grâce aux ingénieries observées.

[31] Nous entendons par *structure itér*ative, une instruction permettant de répéter un certain nombre de fois une série d'instructions simples ou composées constituant un bloc. On parle aussi de *boucle*.

2.2 Nos ingénieries

2.2.1 Les attentes

Pour répondre à notre problématique et dans le cadre de notre méthodologie, nous choisissons d'élaborer trois scénarios représentant chacun une ingénierie associée à un domaine particulier des mathématiques : la *théorie élémentaire des nombres*, l'analyse et les statistiques-probabilités.

Chacune de ces ingénieries est associée à une étude précise en termes d'ETM et ETA sur l'apport de l'algorithmique à l'apprentissage de nouveaux concepts mathématiques. Ainsi, nous choisissons d'élaborer le thème de la *preuve* en situant une première ingénierie dans le domaine de la *théorie élémentaire des nombres*, puis une seconde dans le domaine de l'analyse afin d'étudier chez l'élève quelle pourrait être la contribution d'une approche algorithmique de la preuve tant en *théorie élémentaire des nombres* qu'en analyse.

De plus, dans le cadre d'une tâche associée au domaine des statistiques-probabilités, nous souhaitons diversifier les analyses sur les apports de l'algorithmique aux apprentissages de phénomènes aléatoires. Pour cela, une situation aléatoire est proposée à des élèves afin qu'ils mettent en place un modèle mathématique permettant d'étudier la situation et de valider ou non un premier ressenti sur l'évolution de la situation. Nous partons du fait que les conditions d'évolution de la situation sont connues des élèves. Nous proposons alors une étude du travail des élèves en nous basant sur une utilisation des ETM et des ETA comme outil d'analyse du travail tant mathématique qu'algorithmique lors des différentes phases du *cycle de modélisation* (Blum & Leiss, 2005).

2.2.2 Dans le domaine de la théorie élémentaire des nombres

Dans le cadre de la *théorie élémentaire des nombres*, quel peut être l'apport de l'algorithmique au concept *de preuve* ? En effet, ce domaine propose une grande diversité de situations permettant d'organiser des preuves algorithmiques de problèmes associés à des propriétés issues du domaine de l'arithmétique sans pour autant poser de grandes difficultés d'un point de vue des compétences mathématiques.

Ainsi, dans le cas d'un processus de calcul mettant en jeu une suite finie de nombres entiers, les élèves peuvent au cours d'une première phase conjecturer un résultat répondant au processus, puis dans une seconde phase proposer une preuve par exhaustion de cas pour prouver la conjecture. Cependant, en faisant le choix de nous placer dans un cadre

institutionnel où les élèves auraient des compétences sur un certain nombre de propriétés de l'arithmétique, comme la décomposition d'un nombre entier dans le système décimal, la divisibilité dans **N**, … nous pensons que ces derniers peuvent souhaiter chercher à minimiser le nombre de cas à vérifier pour valider la conjecture.

La question qui se pose alors : comment, dans une telle situation, le travail de preuve peut-il s'articuler pour des élèves débutants en informatique avec le travail sur un algorithme ?

Nous choisissons ainsi d'interpréter le questionnement évoqué ci-dessus à l'aide des *Espaces de Travail Algorithmique* et des interactions observées entre ces ETA et un ETM spécifique au domaine de la théorie élémentaire des nombres.

2.2.3 Dans le domaine de l'analyse

Dans le domaine de l'analyse, nos choix portent sur un travail autour de la preuve d'un théorème clé sur les fonctions : le *théorème des valeurs intermédiaires*. Quel peut être l'apport de l'algorithmique pour donner du sens aux conditions de validité du théorème et pour bâtir une démarche de preuve permettant de construire une preuve de ce théorème ?

Nous nous interrogeons aussi sur le travail que doit mettre en place des élèves débutants en informatique pour organiser un test de sortie de boucle quand celui-ci n'est pas défini par une égalité à un nombre donné. Pour cela, nous souhaitons débuter par un travail autour de la création d'un test d'arrêt associé à une problématique de jeu afin de générer un algorithme implémentable dans un environnement numérique où le test de sortie de boucle peut sembler assez « naturel » pour des élèves débutants. En effet, nous pensons que l'étape d'un test d'arrêt qui ne serait pas défini par une égalité à un nombre donné peut-être source d'une difficulté informatique chez l'élève débutant, qui ne s'observe pas nécessairement lors d'un travail « papier-crayon ».

Ceci nous amène alors à distinguer deux domaines de l'analyse : celui que nous appelons « discret » car nous travaillons sur des suites de nombres dans **N** et celui que nous nommons « continu » car nous travaillons sur des fonctions numériques définies sur des intervalles de **R**.

Dans le cas « discret », nous nous intéressons à des concepts mathématiques issus du domaine de l'analyse mais aussi du domaine des probabilités pour les ETM de référence et d'une structure itérative pour l'ETA.

Dans le cas « continu », nous faisons le choix que les élèves aient des connaissances non institutionnalisées sur les suites adjacentes afin de procéder à une preuve algorithmique du *théorème des valeurs intermédiaires* (selon l'institution, ce théorème est admis au niveau du lycée).

2.2.3.1. Cas « discret »

Dans la continuité des travaux présentés au symposium EMT 5 (Florina, juillet 2016), nous nous questionnons, dans le cadre d'une genèse, sur ce que serait *la prise de conscience de la part de l'élève de la nécessité d'une stratégie « gagnante » et « rapide » répondant à une problématique de recherche d'un nombre entier « secret » compris entre deux nombres entiers donnés et de la programmer dans un environnement informatique afin d'en vérifier sa validité* (Laval, 2017). De plus, nous nous intéressons à ce que serait la contribution d'un travail de construction d'algorithmes sur une suite de nombres dans **N** pour la compréhension du concept d'un *test de sortie de boucle* en informatique.

En effet, les élèves n'ont pas nécessairement l'intuition d'une stratégie particulière pouvant répondre à une problématique donnée. De plus, ils ne distinguent pas forcément l'opérateur humain d'une machine sur laquelle est implémenté l'algorithme associée à cette stratégie afin d'en contrôler sa pertinence et sa validité. De même, ils n'ont pas nécessairement conscience du type de traitement que cette implémentation peut solliciter. La prise de conscience souhaitée renvoie à des questions liées à la représentation du traitement dans un environnement informatique avec la mise en place d'ETA qui doit, dans le cadre d'un travail sur des fonctions numériques, interagir avec des ETM spécifiques caractérisés par ces ETA en lien avec des notions mathématiques issus du domaine de l'analyse : continuité, monotonie, …

Nous pouvons supposer que nous allons observer chez les élèves une première série d'interactions entre les genèses discursives mises en place dans un ETA sur les représentations des traitements des données, des structures informatiques, des variables, mais aussi sur les observables (production et interprétation) lors de l'exécution de l'algorithme dans l'environnement informatique et un ETM spécifique mettant en lumière une réflexion de la

part de l'élève sur des concepts mathématiques comme celui de *nombre entier aléatoire*, d'*intervalle*, de *moyenne* et de *fonction partie entière*. Dans le cadre de l'ETA, nous nous intéressons plus particulièrement sur le choix des *variables itératives* fait par les élèves et les concepts mathématiques qui vont être alors sollicités. Quelles vont être les interactions entre ces concepts mathématiques sollicités et leurs utilisations dans des structures de boucles ? En effet, une utilisation de variables dans une structure de boucle, plutôt que lors de simple déclaration ou d'alternative, peut être favorisée par le fait que, dans la boucle, le concept de variable informatique peut apparaître différent de celui de variable mathématique.

Nous pouvons ainsi identifier trois opérations sur les variables dans une boucle, renvoyant à trois aspects cognitifs : la *mise à jour*, le *test d'arrêt* et l'*initialisation*. De plus, comme le souligne Samurçay (1985), nous avons également deux types de variables intervenant dans des problèmes de programmation : celles qui sont des données explicites du problème et celles qui sont rendues nécessaire par la solution informatique.

Nous pouvons aussi faire l'hypothèse que plus le traitement informatique des variables va s'éloigner de l'exécution « à la main » de l'algorithme, plus les élèves vont être susceptibles de rencontrer des difficultés, comme lors de la gestion d'un invariant de boucle par exemple.

Plusieurs questions concernant le perçu des élèves sur l'initialisation des variables, sur les variables d'accumulation ou sur les compteurs, vont pouvoir alors se poser. Pouvons-nous attendre à ce que les variables d'accumulation soient moins bien traitées par des élèves débutants que les autres ? En effet, nous pouvons supposer que les variables avec lesquelles les élèves éprouvent le moins de difficultés sont vraisemblablement les compteurs. Ceci est peut-être dû au fait que ce type de variable est relativement institutionnalisé. Une question peut alors se poser : en est-il de même avec la structure de boucle « Répéter… Jusqu'à … » ? Nous privilégions d'interpréter ce questionnement général à l'aide des ETA, cependant, nous nous intéressons aussi à la productivité de ce choix.

2.2.3.2. Cas « continu »

Nous partons du fait qu'un « algorithme d'approximation d'un zéro d'une fonction » peut rendre deux valeurs aussi proches que l'on veut, encadrant un réel annulant la fonction (effectivité), et éventuellement qu'il soit garanti qu'il n'existe pas de zéros en dehors de cet encadrement (unicité). Nous nous intéressons ainsi particulièrement à la *dichotomie* qui, dans ce contexte, peut être mise en place dès la classe de Seconde.

Nous rappelons que si nous avons deux nombres réels *a* et *b* et une fonction *f sur* l'intervalle [*a* ; *b*] telle que *f*(*a*) et *f*(*b*) ne sont pas de même signe (Fig. 19), alors le *processus de dichotomie* va consister à découper l'intervalle [*a* ; *b*] en deux intervalles de même amplitude. Si la valeur médiane annule la fonction, l'algorithme termine, sinon on conserve l'intervalle où *f* n'a pas le même signe aux deux bornes. On recommence ensuite ce processus en découpant de nouveau l'intervalle conservé, et ainsi de suite. On obtient alors soit un zéro, soit des intervalles emboîtés tels que *f* n'a pas le même signe aux deux bornes et dont les amplitudes forment une suite géométrique de raison $\frac{1}{2}$. On considère que l'algorithme est effectif si les intervalles encadrent un zéro. De plus, pour atteindre une précision ε, arbitrairement petite, il suffit d'itérer ce processus *n* fois, où *n* est le plus petit entier $\frac{|a-b|}{2^n} \leq \varepsilon$. On peut alors démontrer que *n* est le plus petit entier naturel supérieur ou égal à $log_2\left(\frac{|a-b|}{\varepsilon}\right)$. Dans notre ingénierie, ce résultat ne va pas être exploité par les élèves, mais il va permettre de prévoir qu'avec des données couramment utilisées comme une différence *a* – b égale à 1 ou 2 et ε choisi jusqu'à 10^{-10}, le nombre d'itération est au maximum de 34.

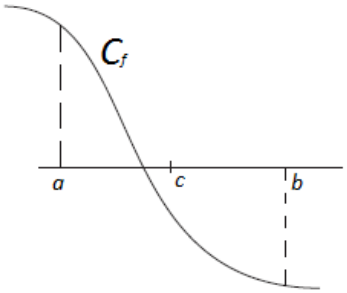


Figure 19 (Représentation graphique d'une fonction continue f sur [a ; b] avec f(a)×f(b)<0)

Après implémentation dans un environnement informatique d'un algorithme répondant à la consigne, nous attendons de la part de l'élève qu'il teste cet algorithme, puis qu'il se questionne sur la validité ou non des résultats obtenus par rapport à l'objectif fixé par la consigne. En effet, nous pensons que ce travail peut aider l'élève à comprendre la nécessité de travailler sur des articulations entre ETM et ETA. Par exemple, un travail algorithmique sur la résolution d'équations à partir de certaines propriétés connues sur une fonction *f* ne peut être qu'utile que si on a la connaissance de certaines conditions vérifiées par la fonction *f* sur l'intervalle d'étude avant d'émettre une conclusion sur le résultat trouvé ? Ainsi, les élèves vont prendre conscience des conditions nécessaires du *théorème des valeurs intermédiaires* (TVI) ou du *théorème de la bijection* (TBI). Nous visons aussi à mettre en évidence la

« monotonie » sur l'intervalle initial comme une condition suffisante d'unicité d'un zéro cherché.

A travers ces différentes étapes, nous nous attendons à ce que l'élève de fin de cycle terminal scientifique puisse prendre conscience de la possibilité de construire une preuve de type algorithmique de la version restreinte[32] du TVI au cas où $f(x) = 0$. Nous nous posons ainsi la question de la reconnaissance par l'élève de fin du cycle terminal scientifique, de propriétés sur les suites des bornes d'intervalles emboîtés et comment il peut utiliser ces propriétés afin de proposer une preuve de type algorithmique du TVI restreint au cas $f(x) = 0$. Ces questions de recherche vont être analysées autour des interactions possibles entre des ETM et des ETA.

Il n'est pas évident que les élèves puissent produire effectivement une preuve de type algorithmique du TVI. En effet, les travaux des élèves qui peuvent être recueillis peuvent ne pas permettre de savoir si les « stratégies » pensées sur des fonctions particulières sont généralisables. Selon Lagrange, nous avons aussi la question implicite : « *Pourquoi ne pas parler de programme d'action plutôt que d'algorithme ?* ». Finalement, nous pouvons observer une difficulté chez l'élève débutant en informatique à enclencher une démarche de preuve, et par conséquent à réellement « *basculer du côté de l'algorithmique* » (Lagrange).

2.2.4 Dans le domaine des statistiques-probabilités

Pour compléter et terminer le travail sur les articulations possibles entre ETA et ETM spécifiques, nous proposons une dernière situation sur l'apport de l'algorithmique à l'apprentissage des mathématiques dans le domaine des statistiques-probabilités. A la différence de l'analyse et de l'arithmétique, les probabilités dans le secondaire, associées aux statistiques, constituent un domaine où les phénomènes aléatoires issus de situations extra-mathématiques jouent un rôle fondamental. Les programmes récents insistent sur une approche « fréquentiste » à travers l'observation de fréquences relatives de caractères représentatifs de ces phénomènes, préparant la compréhension de la notion de probabilité théorique d'un événement, et l'observation de moyennes statistiques préparant à la compréhension de la notion d'espérance d'une variable aléatoire (Parzysz, 2009).

La simulation sur ordinateur de ces phénomènes peut être un premier moyen pratique

[32] La généralisation à k quelconque est un travail algébrique sans rapport avec notre problématique.

d'obtenir des séries statistiques de taille suffisante pour observer la convergence des fréquences relatives et des moyennes statistiques. Par ailleurs, la simulation d'un phénomène aléatoire issu de situations extra-mathématiques suppose une modélisation de la situation. Le modèle ainsi obtenu peut alors être mis à profit pour fonder le calcul de probabilités théoriques (Kiet, 2015).

Ainsi, selon Lagrange et Kiet (2016), dans le cas où la distribution modélise un phénomène aléatoire, comme le lancer d'une pièce de monnaie, la loi de probabilité est présupposée à partir du principe de raison suffisante, comme la similitude des deux faces de la pièce et du poids absolument semblable de la pièce, *il n'y a aucune raison (nulla sit ratio) qu'une des faces soit plus encline à tomber qu'une autre* (Bernoulli, *Ars conjectand*). *L'étude des fréquences relatives dans des essais répétés débouche sur la loi des grands nombres selon laquelle les caractéristiques statistiques d'un échantillon aléatoire se rapprochent d'autant plus des caractéristiques de la population dont l'échantillon est issu, que la taille de l'échantillon augmente. Formalisée en théorème, la loi forte des grands nombres est le fondement théorique des statistiques inférentielles* (Lagrange & Kiet, 2016).

Nous nous intéressons comme Kiet (2015) à la construction par les élèves de tels modèles. Nous situons pour notre part cette construction d'un modèle et son exploitation dans le cadre des ETM et de leurs apports au cycle de modélisation.

Les questions spécifiques que nous considérons alors sont les suivantes : (1) quel est le caractère « algorithmique » d'un ETA intervenant dans le cycle de modélisation (au sens de la construction d'une simulation) ? (2) quelle est alors la spécificité de cet ETA dans le modèle proposé par les élèves ? (3) quelles relations entretient-il au cours de ce travail avec d'autres Espaces de Travail Mathématique ?

Nous souhaitons ainsi étudier les réactions des élèves devant une situation concrète mettant en jeu une double approche : mathématique avec une étude probabiliste du problème et algorithmique avec une simulation permettant d'expérimenter la problématique du problème donné afin d'émettre des conjectures sur la validité ou non de cette problématique.

3. Conclusion du chapitre 3 : Des ETA-ETM idoines et personnels

Comme nous l'avons présenté tout au long de la section précédente, nos expérimentations se situent au niveau des *ETA-ETM idoines*. En effet, ces *ETA-ETM* sont construits à partir de l'étude des programmes de Seconde et des deux niveaux du cycle scientifique terminal, mais aussi à partir de certains documents ressources proposés par l'institution en lien avec l'Education Nationale ou les IREM, ainsi qu'un choix non exhaustif de manuels scolaires.

Quant aux *ETA-ETM personnels*, ils sont dépendants du niveau scolaire de l'élève et de choix de gestion de la classe par l'enseignant : classe entière ou en demi-groupes, salle informatique ou salle classique, en binôme ou seul, outils à disposition … C'est à ce niveau que vont se situer les observations. Ces observations vont nous permettre de faire un retour sur les *ETA-ETM idoines*, par l'étude des manières dont l'élève s'est confronté à un problème mathématique mettant en pratique des tâches en lien avec l'algorithmique, en se basant sur ses propres connaissances tout en restant dans un cadre institutionnel avec des savoirs spécifiques enseignés.

Afin de répondre aux questions spécifiques du (ou des) domaine(s) mathématiques(s) étudié(s), pour chacune de nos ingénieries, notre approche et notre méthode vont être les suivantes :

- mettre en œuvre et évaluer une unité d'enseignement, suivant l'ingénierie, en Seconde et/ou dans le cycle scientifique terminal des lycées français, sur de nouvelles démarches autour de la conception de structures d'algorithmes. Nous souhaitons ainsi étudier les interactions chez l'élève débutant en informatique entre un travail dans le domaine de l'algorithmique et le (ou les) domaines des mathématiques étudié(s), ainsi que la dépendance par rapport au niveau scolaire ;
- construire et analyser des algorithmes permettant de modéliser les observations faites au « papier-crayon » et nécessitant l'utilisation de concepts mathématiques comme : partie entière, moyenne, comparaison, nombre entier aléatoire, etc. ;
- écrire quand cela est nécessaire les algorithmes modélisant la (ou les) tâche(s) demandée(s), afin que l'élève puisse travailler sur les structures et les expressions des algorithmes mis en jeu et présenter une démarche de modélisation « algorithmique » d'une situation pouvant être issu du monde « réel ».

Nous attendons que l'élève mette en place des raisonnements et des stratégies pour répondre à la tâche demandée. Dans le cadre de ces raisonnements et de ces stratégies, nous attendons que l'élève construise et analyse des algorithmes dont les représentations peuvent être sous forme « spatiale » à l'aide d'organigrammes ou « textuelle » à l'aide d'écritures en langage naturel. La transcription en un langage pseudo-code de ces algorithmes doit permettre une implémentation dans des environnements numériques comme *AlgoBox* ou *LARP*[33] (cf. partie 2) afin de tester leurs validités et d'étudier les observations et les conjectures faites manuellement.

Suivant l'ingénierie, diverses structures algorithmiques peuvent être ainsi proposées, mais aussi des remédiations par l'enseignant quand cela s'avère nécessaire pour répondre aux difficultés, tant techniques que théoriques, rencontrées par les élèves.

[33] *AlgoBox* et *LARP* sont des logiciels libres d'accès permettant l'élaboration et l'exécution d'algorithmes.

Conclusion de la partie 1.

Dans cette première partie, nous avons présenté la constitution du terrain de recherche qui va nous permettre de mettre en place des analyses de différentes tâches dans divers domaines mathématiques à travers une approche algorithmique de ces tâches et des concepts mathématiques en jeu, en particulier le concept de preuve algorithmique. Pour cela, nous avons construit un cadre théorique basé pour l'essentiel sur le modèle des *Espaces de Travail Mathématiques* et des *Espaces de Travail Algorithmique* nous permettant ainsi d'affiner les analyses des ressentis et des travaux des élèves.

De plus, afin de mieux comprendre les choix instrumentaux autour des environnements numériques qui ont été présentés dans la section 2 du chapitre 2, nous souhaitons aborder l'hypothèse que l'algorithmique et la programmation font parties de l'activité humaine (Lagrange, 2016). En effet, quand nous mettons en place des activités nécessitant le recours à la programmation, il nous semble utile de prendre du recul, et que l'élève ait une réflexion de nature algorithmique. De même, quand nous nous intéressons aux questions centrales de l'algorithmique que sont la *terminaison*, l'*effectivité* et la *complexité,* la programmation peut constituer un champ expérimental utile, justifiant ainsi le fait que les élèves aient été amenés à expérimenter leurs travaux sur des algorithmes « papier-crayon » dans des environnements numériques comme *AlgoBox* et *LARP*.

Afin d'entrer plus en détail dans le *domaine d'activité* commun à l'algorithmique et à la programmation, il nous semble important d'en préciser les éléments constitutifs qui sont au nombre de trois types et étroitement associés : (1) les *traitements* ; (2) les *données* ; (3) les *dispositifs*. Ces derniers opèrent en différé sans nécessairement considérer dans un premier temps l'utilisateur, en l'occurrence l'élève. La compréhension de ce dernier élément constitutif est fort importante pour les premiers apprentissages, comme le note Lagrange (2016). De plus, il est possible de modéliser ce *domaine d'activité* comme *Espace de Travail Algorithmique* avec des paradigmes spécifiques inspirés des paradigmes géométriques.

Nous rappelons que les *traitements* et les *données* sont des éléments de base qui se conçoivent à une multitude de niveaux différents d'activité et de formalisation. En effet, dans le cas de l'*algorithme d'Euclide*, nous avons un traitement portant sur des nombres entiers

(ou sur des polynômes). De même, la description d'un parcours pour aller d'un lieu à un autre lieu dans une ville va s'appuyer sur des traitements portant sur l'orientation, la direction, le kilométrage, la nature du parcours, … tandis que dans le cas de la recette de cuisine, nous avons un traitement portant sur des ingrédients. Pour le cas d'un moteur de recherche, comme GOOGLE, le traitement va s'effectuer à partir d'une requête…

Nous rappelons aussi que selon Lagrange, un ordinateur est un dispositif de ce type, mais il est très rare qu'il soit vu au niveau de ses composants matériels. Entre ces composants et le programmeur, il existe ainsi une ou le plus souvent plusieurs couches logicielles qu'on appelle *langages de programmation*.

Les analyses des travaux d'élèves (cf. partie 2) faites à travers le cadre théorique basé sur des articulations possibles entre des ETM spécifiques et des ETA dans plusieurs domaines des mathématiques, les *traitements* et les *données* ne vont avoir d'autre finalité que d'être exécutés en différé sur un ou des *dispositifs*. Ceux-ci sont aussi concevables à une multitude de niveaux différents d'activité et de formalisation.

Pour conclure cette première partie, nous proposons de compléter nos propose sur l'opérationnalité du triptyque *traitement, donnée, dispositif*. Pour cela, nous nous appuyons sur les remarques de Lagrange (2017). En effet, il propose de considérer deux paradigmes : le *paradigme fonctionnel* mis en avant, notamment dans les premiers cycles universitaires de mathématiques en opposition au *paradigme impératif* qui domine dans l'enseignement secondaire. Afin d'expliquer la nature de ces deux paradigmes, nous proposons de prendre comme exemple, l'*algorithme des soustractions* permettant le calcul du PGCD de deux nombres entiers.

Dans le cas du *paradigme impératif*, nous avons :

```
Entrer A et B
TantQue A différent de B
  Si A > B alors
    A prend la valeur B – A
  Sinon
    B prend la valeur A – B
Resultat A
```

Pour le cas du *paradigme fonctionnel*, nous obtenons :

```
Pour PGCD(A ; B)
Si A > B alors
  PGCD(A – B ; B)
Sinon
  Si B > A
      PGCD(B – A ; B)
    Sinon
      A
```

- **Analyses des deux algorithmes**

Dans l'écriture « *impérative* », nous observons que les états successifs du *dispositif* (ici, les valeurs du couple de données (A ; B)) et la façon dont il évolue (c'est-à-dire le *traitement*) sont manifestes. Une *preuve de terminaison* est possible en considérant une donnée non traitée, la somme A + B dont il est facile de voir qu'elle est strictement décroissante. En revanche une *preuve d'effectivité* suppose de considérer une « donnée invariante » (le PGCD des deux nombres entiers A et B), ce qui est loin d'être trivial.

En ce qui concerne l'interprétation de l'écriture *fonctionnelle*, nous constatons qu'il ne suppose pas *a priori* de considérer le triptyque *traitement*, *donnée*, *dispositif*. En effet, c'est une écriture mathématique qui se suffit à elle-même. D'un point de vue algorithmique, la *preuve d'effectivité* est directe. En effet, l'écriture ne fait que retraduire les deux propriétés :

$$PGCD(A ; B) = PGCD(A – B ; B) \text{ et } PGCD(A ; A) = A.$$

En revanche, une *preuve de terminaison* ne peut pas se faire directement.

D'un point de vue programmation, l'écriture *fonctionnelle* a aussi des avantages. En effet, la fonction ainsi définie peut être appelée dans un programme au même titre que les fonctions natives d'un langage. Par exemple, pour le calcul du PPCM de deux nombres entiers A et B, nous avons :

```
Pour PPMC(A ; B)
A × B/PGCD(A ; B)
```

Lagrange constate qu'auprès d'étudiants professeurs de mathématiques, le passage d'*écritures impératives* à des *écritures fonctionnelles* peut constituer un saut cognitif que la plupart ne franchissent pas.

Afin d'éviter ce type de difficultés avec des élèves débutants en informatique, nous faisons le choix de favoriser l'*écriture impérative* et pour cela ne choisissons que des environnements numériques le permettant.

Comment interpréter cette difficulté ? L'*écriture fonctionnelle* masque la référence directe au triptyque *traitement*, *donnée*, *dispositif* pour favoriser une interprétation conforme aux habitudes mathématique. Mais sa compréhension suppose de restituer le fonctionnement du dispositif. Dans le cas de l'*algorithme d'Euclide*, il faut imaginer que le dispositif crée à chaque

appel de la fonction une instance de celle-ci avec son couple de données, qui est mémorisée dans l'attente de la finalisation du traitement. Il s'agit en quelque sorte de « dé-récursiver » la fonction[34]. Pour un sujet « avancé » ce n'est pas un problème. En effet, il dispose de structures mentales lui permettant de lire le triptyque *traitement*, *donnée*, *dispositif* à partir d'une *écriture fonctionnelle*. Pour un sujet « débutant », l'enjeu est au contraire de construire ces structures[35].

Suite aux travaux sur les *Système de Représentation et de Traitement* (Hoc, 1987) pour caractériser les structures mentales, Lagrange (1990) montre qu'il existe des précurseurs, encore mal identifiés. Chez la grande majorité des sujets, la construction de ces structures impose une exposition au triptyque : *quel traitement veut-on effectuer, sur quelles données, avec quel dispositif ?*

Distinguer programmation et algorithmique peut ainsi amener à des nuances, à condition de ne pas les opposer. Par exemple, dans l'analyse de l'*algorithme d'Euclide*, nous avons considéré le triptyque *traitement, donnée, dispositif* à un niveau assez général, afin d'aider à la compréhension du lecteur. Les choses sont certainement différentes par exemple pour un élève de fin du cycle 4 et de Seconde. Doit-on alors refuser à cet élève de se confronter à un langage précis sur un ordinateur précis sous prétexte qu'il soit en classe de mathématiques et que la séance soit sur l'algorithmique, alors que cela peut lui fournir un champ expérimental et ainsi être un point d'appui précieux pour sa compréhension de nouvelles compétences mathématiques ?

[34] Par exemple dans le cas de l'*algorithme d'Euclide*, il s'agit de récursivité terminale, et donc l'instance créée est exécutée directement et peut être « oubliée », ce qui revient à l'itération en impératif. C'est donc un cas particulièrement simple, pour lequel Rouchier a montré qu'on pouvait créer des situations favorisant sa compréhension. Ce serait différent avec la récursivité généralisée comme par exemple pour la génération de suites récurrentes doubles (cas des suites de Fibonacci).

[35] On peut supposer que cela soit le reflet d'activités en algorithmique et programmation trop sporadiques et trop peu étayées par des approches didactiques.

Partie 2 : Les ingénieries didactiques

- **Chapitre 1 :** Des ingénieries didactiques.
- **Chapitre 2 :** Articulation Espace de Travail Algorithmique et Espace de Travail *Mathématique spécifique* à la *théorie élémentaire des nombres*.
- **Chapitre 3** : Vers une ingénierie didactique autour de l'Algorithme de dichotomie.
- **Chapitre 4** : Une ingénierie articulant ETA et ETM spécifique (partie 1) : *Algorithme de dichotomie* « discrète » – Une stratégie « rapide » et « gagnante ».
- **Chapitre 5** : Une ingénierie articulant ETA et ETM spécifique (partie 2) : *Algorithme de dichotomie* « continue » – Vers une preuve du théorème des valeurs intermédiaires.
- **Chapitre 6** : Une ingénierie articulant ETA et ETM spécifique : Une politique des naissances.
- **Conclusion de la partie 2**.

Chapitre 1 : Des ingénieries didactiques

La notion d'*ingénierie didactique* (Artigue, 1988) est apparue au début des années 1980 comme une méthodologie pour tester des hypothèses de recherche en didactique. Elle amène à un travail comparable à celui d'un ingénieur. En effet, ce dernier, pour réaliser un projet, doit s'appuyer sur des connaissances scientifiques issues de son domaine de connaissances et de savoirs et accepter de se soumettre à un contrôle de type scientifique.

1. Les différentes phases de la méthodologie

Selon Artigue (1990), pour chacune de nos ingénieries, nous sommes amenés à distinguer quatre temps :

(1) Un premier temps constitué d'**analyses *a priori***, *destinées à faire émerger les connaissances liées au domaine* mathématique *étudié, afin de préparer* les phases *de conception* qui s'ensuivent. Ces analyses vont prendre *en compte les objectifs spécifiques* de la recherche concernée.

(2) Un deuxième temps où, en tant que chercheurs, nous prenons, **la décision d'agir** *sur un certain nombre de variables du système non fixées par les contraintes : variables de commande* dont nous supposons *qu'elles sont des variables pertinentes par rapport* aux problèmes étudiés. Les principaux objectifs de ce deuxième temps vont consister à *déterminer en quoi les choix effectués* vont nous permettre *de contrôler les comportements des élèves et leur sens.* Pour cela, ces analyses vont se fonder *sur des hypothèses dont la validation* est*, en principe, indirectement en jeu dans la confrontation opérée* au cours des différentes phases du déroulement, afin d'évaluer les hypothèses faites lors des analyses *a priori* avec l'observation du « réel » et permettre ainsi d'affiner les analyses *a posteriori* qui vont en découler.

(3) Un troisième temps constitué des différentes phases du déroulement de l'**expérimentation sur le terrain**. Ce temps va nous permettre d'élaborer un *recueil de données* afin d'éclairer l'objet de nos recherches.

(4) Un quatrième temps d'**analyses *a posteriori*** qui vont précéder des évaluations s'appuyant sur l'ensemble des données recueillies lors des expérimentations, c'est-à-dire les observations pratiquées lors de l'expérimentation et les productions des élèves en classe, voire hors classe.

Ainsi, c'est à travers cette *confrontation des deux analyses : analyse a priori et analyse a posteriori* que va se fonder *essentiellement la validation des hypothèses engagées* (Artigue, 1988).

2. Nos choix généraux

Nous choisissons de réaliser deux ingénieries dites *locales* et une dite *globale* dans des classes de Secondes et du cycle scientifique terminal.

Nous rappelons que pour chacune de ces ingénieries, nous désirons donner un nouvel éclairage sur les interactions possibles entre *Espaces de Travail Mathématiques* spécifiques et *Espaces de Travail Algorithmique* à travers un nombre réduit de séances et suivant le niveau scolaire de l'élève. Chacune de ces ingénieries doit ainsi donner lieu à une *confrontation entre analyse a priori et analyse a posteriori*, permettant de répondre de façon contextualisée aux questions de recherche. Une synthèse de ces analyses sera proposée afin de tenter des réponses plus globales.

De plus, pour chacune de ces ingénieries, il va aussi s'agir d'organiser un travail sur un (ou plusieurs) algorithme(s) dans un domaine mathématique particulier, sur une durée plus ou moins longue d'observation (c'est-à-dire une ou plusieurs séances) et sur plusieurs niveaux scolaires du lycée de façon à étudier l'évolution des différents *Espaces de Travail Mathématique spécifique* et *Algorithmique*, ainsi que leurs interactions qui peuvent exister à différents moments de la scolarité de l'élève.

3. Les ingénieries didactiques : les domaines, les tâches et les phases

3.1 Les domaines mathématiques spécifiques

De nombreux domaines mathématiques enseignés au lycée semblent être favorables à une étude des interactions possibles entre différents domaines mathématiques spécifiques et le domaine de l'algorithmique. En effet, les programmes précisent que l'*algorithmique a une place naturelle dans tous les domaines des mathématiques et les problèmes posés doivent être en relation avec les autres parties du programme (analyse, géométrie, statistiques et probabilités, logique...)*.

Comme nous l'avons précisé précédemment, des domaines comme la *théorie élémentaire*

des nombres, l'analyse, les probabilités-statistiques sont presque tous accessibles aux trois niveaux scolaires des lycées et permettent ainsi de mettre en place l'étude des interactions possibles avec l'algorithmique. Au niveau du lycée, les élèves ont suffisamment de connaissances mathématiques dans les domaines de l'analyse et des probabilités-statistiques sur les trois niveaux scolaires du lycée et pour la *théorie élémentaire des nombres* plus particulièrement au niveau de la Terminale Scientifique, pour mieux comprendre ce que peut être un apport de l'algorithmique à certains apprentissages mathématiques. En effet, ils peuvent plus précisément interpréter et définir une preuve heuristique à l'aide d'un algorithme par exemple ou analyser les différentes étapes d'un algorithme complexe,…

3.1.1 La théorie élémentaire des nombres

Pour les algorithmes considérés dans le domaine de la *théorie élémentaire des nombres*, nous pouvons nous intéresser à ceux qui permettent un travail constructif sur le concept de preuve, mais aussi sur l'efficacité et la complexité de l'algorithme, et non se limiter à une simple acquisition de compétences nouvelles dans ce domaine mathématique.

C'est pourquoi nous choisissons de centrer le travail, non sur les algorithmes classiques comme celui d'Euclide ou sur un test de primalité, mais plutôt sur un processus de calcul mettant en jeu des connaissances plus élémentaires comme les différentes représentations d'un nombre entier naturel dans le système décimal d'un point de vue mathématique mais aussi informatique ainsi que sur le concept de divisibilité dans **N**. Ces compétences font partie du programme de l'enseignement de spécialité de Terminale Scientifique. De plus, elles ont déjà fait l'objet de travaux dans les classes antérieures de la part des élèves. Nous proposons ainsi de mettre en place une ingénierie *locale* autour d'un algorithme complexe et permettant d'obtenir une preuve non explicative d'une conjecture : l'*algorithme de Kaprekar* pour des nombres entiers constitués de trois chiffres dans le système décimal.

3.1.2 L'analyse

Les algorithmes considérés dans le domaine de l'analyse concernent des méthodes d'approximation issues du domaine de l'*analyse numérique* comme la résolution d'équations, la détermination d'un maximum local d'une fonction, l'intégration numérique, les équations différentielles,...

Faisant le choix d'étudier un travail algorithmique sur un même sujet comme contribution

à l'apprentissage en analyse au cours des trois années de lycée de notions aussi diverses qu'approximations, résolution d'équations à une inconnue dans **R** ou sur des intervalles de **R**, étude de la continuité sur un intervalle de **R**, application du *Théorème des valeurs intermédiaires*, et de son corollaire, le *Théorème de la bijection*, nous choisissons d'élaborer et d'expérimenter une ingénierie *globale* autour d'un algorithme d'approximation : l'*algorithme de dichotomie*.

3.1.3 Les statistiques et les probabilités

Amener les élèves à travailler des algorithmes en lien avec les lois de probabilités qu'elles soient « discrètes » ou « continues », ainsi que sur des marches aléatoires, sont des points forts de l'enseignement des statistique-probabilités des classes des lycées français. Ainsi, comme nous le verrons dans le chapitre 6 de cette partie, de nombreux documents ressources proposés par l'institution ainsi que l'ensemble des manuels scolaires proposent différentes tâches sur ces notions en lien avec l'utilisation de l'algorithmique permettant ainsi de procéder à des simulations de situations aléatoires avec l'intégration de concepts comme « nombre aléatoire », « fréquences théoriques », « probabilités », « répétitions », « échantillonnage », « estimation », « statistiques inférentielles », « moyennes », « espérance mathématique »,… L'introduction de ces différents concepts se fait de façon progressive, tout au long des trois années de lycée. Ils permettent entre autres que les statistiques et les probabilités, en articulation avec l'algorithmique, trouvent une place naturelle dans des tâches en *relation* […] *avec les autres disciplines ou le traitement de problèmes concrets*.

A l'occasion de l'écriture d'algorithmes autour de la simulation aléatoire et de leurs implémentations dans des environnements numériques afin de tester leurs validités et d'interpréter les résultats obtenus, les élèves peuvent acquérir des habitudes de rigueur et s'entraîner *aux pratiques systématiques de vérification et de contrôle.* De même, l'*utilisation d'instructions élémentaires (affectation, calcul, entrée, sortie)* ainsi que des structures algorithmiques faisant partie des demandes institutionnelles leur permettent aussi d'acquérir des compétences supplémentaires en langage de programmation. Par ailleurs, comme nous l'avons évoqué plus haut, la simulation de phénomènes aléatoires issus de situations extra-mathématiques va supposer que l'élève puisse acquérir des compétences sur le plan modélisation d'une situation aléatoire. Le modèle ainsi obtenu peut alors être mis à profit pour fonder le calcul de probabilités théoriques, comme l'a témoigné Kiet (2015) dans ces

travaux de recherche sur l'enseignement des probabilités.

Ainsi, dans le cadre d'une ingénierie se situant dans le domaine des statistiques et des probabilités, nous choisissons de faire une expérimentation sur une *politique des naissances* et de voir comment les élèves peuvent ressentir dans un premier temps, avant toute expérimentation, les conséquences que pourraient avoir sur l'évolution d'une population la mise en place d'une telle politique.

3.2 Les tâches

Comme nous l'avons déjà évoqué, l'étude des algorithmes s'inscrit dans les programmes scolaires du lycée, depuis 2010 et depuis 2016 dans ceux des collèges et en moindre mesure des cycles 1, 2 et 3 des écoles primaires. Tenant compte de la demande institutionnelle, les tâches que nous proposons pour nos ingénieries sont inspirées par les directives communes venant de l'institution (programmes, documents ressources, manuels,…) pour les trois années du lycée. Ceci spécifie alors de mettre en place des tâches en algorithmique afin de :

- *décrire certains algorithmes en langage naturel ou dans un langage symbolique ;*
- d'*en réaliser quelques-uns à l'aide d'un tableur ou d'un programme sur calculatrice ou avec un logiciel adapté ;*
- ***d'****interpréter des algorithmes plus complexes.*
- *[…]*

De même, les programmes précisent aussi les compétences à mettre en œuvre au cours de travaux en lien avec ces tâches (Guy, 2013) :

- *Etre capable d'écrire une formule permettant un calcul ;*
- *Etre capable d'écrire un programme calculant et donnant la valeur d'une fonction ;*
- *ainsi que les instructions d'entrées et sorties nécessaires au traitement ;*
- *Etre capable d'utiliser des boucles et itérateur, des instructions conditionnelles ;*
- *Etre capable de programmer un calcul itératif, le nombre d'itérations étant donné ;*
- *Etre capable de programmer une instruction conditionnelle, un calcul itératif, avec une fin de boucle conditionnelle.*

Cependant, nous ne nous contentons pas de rester dans un cadre purement institutionnel pour certaines de nos ingénieries, en particulier pour celle que nous mettons en place dans le domaine de la *théorie élémentaire des nombres* (cf. Chapitre 2). En effet, nous verrons que celle-ci est basée sur le concept de preuve dans le domaine de l'algorithmique. Or ce concept

n'apparaît pas explicitement dans les programmes bien que cela puisse être pertinent de le mettre dans un travail sur l'interprétation d'algorithmes complexes.

3.3 Les Phases

Chacune de nos ingénieries va être découpée en plusieurs phases. Cependant, avant d'étudier chacune des phases constituant l'ingénierie, nous choisissons de présenter en premier lieu le domaine mathématique spécifique retenu pour l'expérimentation algorithmique Ceci va nous amener à étudier le programme du niveau scolaire où l'ingénierie va être mis en place, ainsi que mener une étude approfondie de manuels scolaires et des documents ressources en lien avec les compétences mathématiques en jeu dans l'ingénierie. Toutefois, pour des raisons d'efficacité le choix des manuels scolaires étudiés se fait sur un nombre non exhaustif d'ouvrages pour chaque niveau scolaire. Ce choix vient aussi du fait des similitudes entre les différents manuels se trouvant sur le marché de l'édition, tant sur le plan du contenu que de l'organisation des chapitres. De plus, lors des analyses des phases, nous choisissons de nous intéresser qu'aux hypothèses relatives aux interactions entre *ETA* et *ETM* spécifiques en lien avec le choix des tâches pour les élèves.

Sachant que le nombre d'heures réellement attribuées à l'enseignement de l'algorithmique dans les lycées français et, a fortiori à l'utilisation d'environnements numériques autres que la calculatrice en classe est relativement mineur, malgré les objectifs pédagogiques et didactiques imposés par l'institution, nous devons aussi examiner les différents concepts fondamentaux de l'algorithmique que sont les données, les variables informatiques, les structures de données, les bases de données et leurs représentations,… Cette réflexion va se faire en fonction des phases étudiées pour chacune des ingénieries, en particulier lors des analyses *a priori* et *a posteriori*. En effet, lors de des analyses *a priori*, nous allons nous intéresser plus particulièrement aux différentes variables de commande, ainsi qu'aux instruments utilisés par les élèves pour mener à bien les tâches correspondantes à chacune des phases. Le langage, les connaissances à réactiver, les médiations tant en mathématique qu'en algorithmique sont aussi des paramètres qui doivent être pris en compte lors de la description et des analyses des différents phases. Nous souhaitons aussi observer des phases où les élèves travaillent de façon individuelle ou en en binôme, mais aussi des phases collectives où un retour sur ces phases individuelles sont faites afin d'étudier les interactions entre élèves en particulier lorsque ceux-ci se trouvent dans des situations d'adidacticité. Nous

allons aussi favoriser des périodes où les élèves vont être regroupés en classe entière ou en demi-groupes afin de voir les influences que ces regroupements vont pouvoir avoir sur la progressivité des apprentissages chez l'élève.

Le choix des tâches est fait en lien avec la progressivité des apprentissages en algorithmique, en particulier dans la connaissance et la maitrise des itérations, des structures informatiques et des variables informatiques.

Pour les différentes analyses *a priori* que nous allons mener, nous choisissons d'étudier plus précisément l'influence du choix des tâches sur les comportements des élèves et le sens qu'ils vont donner à ces tâches en lien avec des domaines mathématiques spécifiques dans le cadre d'une approche algorithmique de ces domaines. De façon générale, nous souhaitons étudier comment cela doit contribuer à l'évolution des ETM/ETA personnels chez des élèves débutants en informatique tant sur le plan conception d'algorithmes que sur le plan de la programmation.

Ces analyses *a priori* sont mises à l'épreuve au cours d'expérimentations sur le terrain dans diverses classes sur les trois niveaux du lycée : Seconde générale et les deux niveaux du cycle scientifique terminal. Nous faisons le choix de prendre des classes « ordinaires », mais aussi une classe où une sélection des élèves est faite à l'entrée de la Seconde. Cependant, pour des raisons logistiques, cette classe n'est prise en compte que dans le cadre de l'ingénierie *globale*.

Dans toutes les ingénieries mises en place, nous sommes amenés à exploiter du matériel d'enregistrement pour obtenir des prises d'images et de son, afin d'étudier au plus près les échanges qui ont lieu entre les élèves d'un même binôme, d'un groupe, d'une classe mais aussi observer les interactions entre les élèves et leur enseignant. De plus, chaque enseignant participant aux ingénieries, en accord avec le chercheur, récupère à la fin de chaque séance les travaux écrits de ses élèves, ainsi que les algorithmes implémentés dans les environnements numériques choisis pour l'ingénierie. Les différents algorithmes implémentés sur des logiciels algorithmiques sont enregistrés sur des clés USB. Le chercheur récupérant ces clés USB, peut ainsi analyser les algorithmes programmés en se référant au cadre théorique des ETM/ETA. De même, des copies d'écran d'ordinateur représentant les résultats obtenus après que l'élève ou le binôme ait testé l'algorithme programmé dans tel ou tel environnement numérique, sont aussi recueillis par le chercheur afin de faciliter une meilleur exploitation des travaux des élèves et par conséquent d'aider aux analyses du déroulement des différentes phases permettant ainsi de formuler des analyses *a posteriori* fines et de mieux

caractériser les ETM/ETA personnels des élèves, ainsi que de confronter les résultats des observations faites avec ce qui avait été anticipé lors des analyses *a priori*.

4. Les deux principaux environnements numériques utilisés dans le cadre de nos ingénieries

Les élèves travaillent dans plusieurs environnements algorithmiques : langage naturel, pseudo-code et langage informatique. Ils ont à leur disposition les logiciels *AlgoBox* et *LARP*. Nous présentons brièvement ce deux logiciels afin d'aider le lecteur à se familiariser avec ces environnements algorithmiques qui font l'objet d'une reconnaissance institutionnelle en particulier par les auteurs de manuels et les documents ressources proposées par l'Inspection.

4.1 Présentation du logiciel AlgoBox

AlgoBox (fig. 20) est un logiciel libre, multiplateforme et gratuit permettant la création d'algorithmes utilisant :

- des déclarations de variables informatiques de type NOMBRE (nombre entier ou décimal), de type CHAINE (chaîne de caractères), de type LISTE (liste de nombres) ;
- des opérations élémentaires d'affectation, de lecture et d'affichage de variables ou de chaînes de caractères ou de listes de nombres ;
- des structures conditionnelles : « Si…Alors » ou « Si… Alors…Sinon » ;
- des structures de boucles « Pour…Faire » et « TantQue… Faire ».

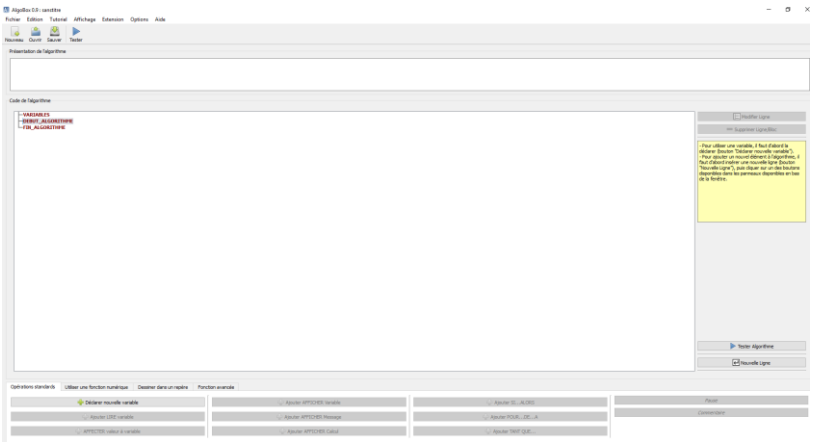


Figure 20 (Ecran AlgoBox)

Cet environnement numérique permet aussi de définir et d'utiliser une fonction numérique. Pour cela, l'utilisateur ouvre l'onglet « Utiliser une fonction numérique » qui lui permet de définir une fonction F1 à variable réelle du type F1(x) = < expression en fonction de *x* >. Une fois que la fonction est définie par l'utilisateur, celle-ci peut s'utiliser comme n'importe quelle fonction prédéfinie au sein d'expressions numériques.

De plus, cet environnement possède deux onglets qui permettent respectivement de « Dessiner dans un repère » et de construire une « Fonction avancée », notée F2, dépendante d'un paramètre. En effet, on peut ainsi définir :

- une fonction définie par « morceaux » ;
- une fonction dépendant de plusieurs paramètres (contrairement à la fonction F1 qui ne doit dépendre que de *x*) ;
- une fonction définie de façon récursive.

Pour cela, l'utilisateur doit :

- préciser les paramètres dont dépend la fonction ;
- ajouter une à une les conditions permettant de définir la fonction ;
- indiquer ce que doit retourner la fonction quand aucune des conditions n'est remplie.

Un texte écrit en *AlgoBox* peut être vu soit comme un algorithme si on s'intéresse à sa logique et à ses performances, soit comme un programme si on s'intéresse à son implémentation.

4.2 Présentation du logiciel LARP

LARP[36] (fig. 21) est un logiciel libre et gratuit correspondant à un langage de programmation permettant le prototypage rapide d'algorithmes. Il donne la possibilité d'écrire et d'implanter des algorithmes en langage pseudo-code qui être aussi formulé sous forme d'organigramme. L'objectif est ainsi de permettre la comparaison entre deux modes d'écriture des algorithmes : « textuel » avec le pseudo code et spatial avec « l'organigramme » afin de faire ressortir l'intérêt de l'écriture d'un pseudo-code. Cet environnement permet aussi de concevoir des sous-programmes pour des algorithmes plus complexes. En effet, ces sous-programmes peuvent être appelés par le programme principal lors de son exécution. De plus, *LARP* permet d'aller plus loin en autorisant l'appel d'un sous-programme par lui-même, c'est-à-dire il favorise la récursivité.

[36] *LARP* est un acronyme qui vient de la compression de la phrase « Logiciel d'Algorithmes et de Résolution de Problèmes ».

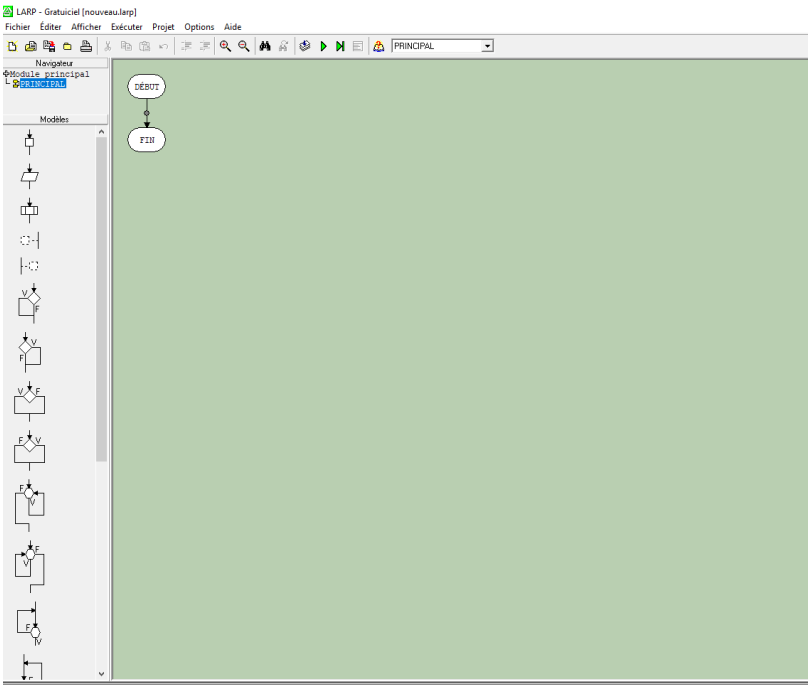


Figure 21 (Ecran LARP)

LARP possède des ressources de programmation très proches de celles d'AlgoBox. Cependant, il permet aussi de concevoir des algorithmes nécessitant l'utilisation d'une boucle « Répéter… Jusqu'à ».

5. Conclusion du chapitre 1

Ayant présenté nos choix généraux et notre problématique, nous souhaitons dans les chapitres suivants de cette deuxième partie, présenter les différentes ingénieries didactiques que nous avons mises en place dans des classes de Seconde et du cycle terminal scientifique de l'enseignement secondaire français. Pour chacune de ces ingénieries, nous proposons des analyses des différentes phases constituant l'ingénierie étudiée en nous référant au cadre théorique présenté dans la partie 1.

Pour conclure ce manuscrit, nous aurons une dernière partie où nous procéderons à une conclusion globale du travail décrit dans les différents chapitres des ingénieries de la partie 2.

Chapitre 2 : Articulation *Espace de Travail Algorithmique* et *Espace de Travail Mathématique spécifique* à la *théorie élémentaire des nombres*

Comme nous l'avons présenté dans le chapitre précédent, nous souhaitons diriger notre recherche sur ce que peuvent s'apporter mutuellement des travaux en algorithmique et l'apprentissage des mathématiques au lycée. Nous nous intéressons en particulier à la dépendance de cet apport relativement au domaine étudié et au niveau d'enseignement.

Ce chapitre étudie la conception et l'analyse d'une ingénierie autour d'un algorithme en lien avec un domaine particulier des mathématiques : la *théorie élémentaire des nombres* et mettant en jeu la preuve en algorithmique. L'intervention de ce domaine inscrit bien l'ingénierie dans l'étude des interactions entre *ETM et ETA spécifiques.*

Cette ingénierie est *locale* : nous nous contentons d'observer une seule classe et par conséquent un seul niveau scolaire.

1. Pourquoi parler de *théorie élémentaire des nombres* et non d'*arithmétique* ?

1.1 La *théorie des nombres*

Traditionnellement, la *théorie des nombres* est considérée comme étant une branche des mathématiques qui traite des propriétés des nombres entiers, naturels ou relatifs, et qui contient de nombreux problèmes mettant en jeu au départ des notions simples. Elle occupe une place particulière dans les domaines des mathématiques. En effet, elle est en connexion avec de nombreux autres domaines des mathématiques.

Elle a fait de grands progrès au cours des dernières décennies et elle est encore en plein développement. Ainsi, au début de la préface de son ouvrage *Algebraische Zahlentheorie*, Neukirch (2006) écrit : « *Die Zahlentheorie nimmt unter den mathematischen Disziplinen eine ähnlich idealisierte Stellung ein wie die Mathematik selbst unter den anderen*

Naturenwissenschaften.[37] ». Les problèmes peuvent être reliés à des applications pratiques (cryptage…) ou à des domaines ludiques.

1.2 L'arithmétique

Le terme *arithmétique* est aussi utilisé pour faire référence à la *théorie des nombres*. C'est un terme ancien qui est moins usité que par le passé. Pour éviter des confusions, jusqu'au début du vingtième siècle, quand on parlait de *théorie des nombres*, on utilisait en général le terme d'*arithmétique supérieure*. Cependant, l'adjectif *arithmétique* reste encore assez répandu de nos jours. En effet, il peut être utilisé en particulier pour désigner des domaines mathématiques, comme la *géométrie algébrique arithmétique* ou encore l'*arithmétique des courbes* et des *surfaces elliptiques*,... Cependant, ce sens du terme arithmétique ne doit pas être confondu avec celui utilisé dans le domaine de la *logique* lors de l'étude des systèmes formels axiomatisant les nombres entiers, comme il en est avec l'*arithmétique de Peano* (Mathématicien et linguiste italien, fin 19^{e} et début 20^{e}). En effet, dans le cadre de l'*arithmétique de Peano*, nous avons un système logique destiné à fournir une axiomatisation formelle de l'arithmétique. Ici, la notion de nombre entier naturel est primitive. Le langage logique permet de construire les autres notions sur cette base. De plus, il donne un cadre logique dans lequel il est possible d'étudier les fonctions calculables. Ceci permet ensuite d'établir les célèbres *théorèmes d'incomplétude* de Gödel (Logicien et Mathématicien du 20^{e}).

1.3 La *théorie élémentaire des nombres*

La *théorie élémentaire des nombres* est une branche de la *théorie des nombre* où les nombres entiers sont étudiés sans utiliser de techniques mettant en jeu d'autres domaines des mathématiques. Ainsi, les questions de divisibilité, l'*algorithme d'Euclide* pour calculer le PGCD[38] de nombre entiers, les nombres premiers et la décomposition en produit de facteurs premiers de nombres entiers, la recherche des nombres parfaits et des congruences, etc., appartiennent à ce domaine et mettent en jeu de nombreux algorithmes.

2. Choix principaux pour une ingénierie dans le domaine de la

[37] La théorie des nombres occupe parmi les disciplines mathématiques une position idéalisée analogue à celle qu'occupent les mathématiques elles-mêmes parmi les autres sciences de la nature (N.D.L.R.)

[38] Plus Grand Commun Diviseur

théorie élémentaire des nombres

Notre hypothèse, cohérente avec les buts énoncés au début de ce chapitre, est que le domaine de la *théorie élémentaire des nombres* permet un travail en algorithmique centré sur la *preuve* en mettant en jeu des outils mathématiques simples. L'algorithme retenu est l'*algorithme de Kaprekar*. Il répond à un processus de calcul dont nous allons présenter les différentes étapes dans le paragraphe suivant. Nous avons choisi cet algorithme, bien qu'il ne s'agisse pas d'un des algorithmes classiques de la *théorie élémentaire des nombres* dont nous avons donné quelques exemples ci-dessus. En effet, comme nous l'avons précisé au début de ce chapitre, nous visons ici une ingénierie *locale* et désirons limiter les interférences avec des contenus centraux dans la théorie[39].

Précisons d'emblée que cette ingénierie s'inscrit dans un travail de collaboration avec M.N. Guy, étudiante de Master en Didactique des Mathématiques. En effet, dans le cadre de son mémoire de Master (Guy, 2013), une ingénierie portant sur la construction de l'algorithme a été mise en place et exploitée, dénommée « ingénierie-Guy » dans la suite. Nous développons ici une ingénierie, qui fait suite à celle de Guy et qui porte sur l'efficacité et la preuve.

3. Présentation de l'algorithme de Kaprekar

3.1 Description du processus de calcul

Un nombre entier naturel non nul p étant choisi, nous considérons les nombres entiers naturels inférieurs à 10^p écrits avec p chiffres dans la numération de position dans le système décimal, les premiers chiffres pouvant être des zéros. Le processus utilise une fonction K associant à un de ces nombres n un autre nombre $K(n)$ généré de la façon suivante : on forme le nombre n_1 en rangeant les chiffres du nombre n dans l'ordre croissant et le nombre n_2 en rangeant toujours les chiffres du nombre n dans l'ordre décroissant. On pose $K(n) = n_2 - n_1$. Il est trivial de voir que $K(n)$ est nécessairement positif et inférieur à 10^p et donc qu'il est possible d'itérer le processus avec $K(n)$. On poursuit l'itération jusqu'à obtenir 0 ou un nombre entier constant ou un cycle de nombres entiers suivant la taille du nombre entier initial n. L'ensemble des nombres entiers positifs et inférieurs à 10^p étant fini, le traitement termine alors

[39] Par exemple, l'ingénierie globale sur la dichotomie (ch. 3 et 4) interfère notablement avec le TVI.

nécessairement.

Ce processus de calcul a été proposé et décrit par le mathématicien d'origine indienne D.R. Kaprekar (1905-1986) pour des nombres entiers naturels de quatre chiffres et lui a donné son nom. Cependant, ce processus peut être généralisé à tous les nombres entiers naturels *p*.

Nous le décrivons ici comme un traitement « manuel », un des enjeux étant pour les élèves de l'exprimer comme un algorithme, c'est-à-dire un traitement adapté à un langage respectant des contraintes formelles. C'est pourquoi nous désignons par la suite ce traitement par le terme de « recette » et nous dirons *recette de Kaprekar*.

3.2 Étude selon le nombre entier initial choisi[40] - Choix du nombre de chiffres du nombre entier initial pour notre ingénierie

Les résultats obtenus, quand on met en place l'*algorithme de Kaprekar*, sont différents suivant le nombre de chiffres constituant le nombre initial. Par exemple, pour des nombres entiers naturels constitués de moins de cinq chiffres, nous avons :

- pour un nombre entier compris entre 0 à 9 : le point fixe 0 est obtenu dès le premier pas ;
- pour un nombre entier compris entre 00 à 99 : au premier pas le nombre obtenu est le point fixe 00 si tous les chiffres sont égaux, sinon c'est un multiple non nul de 9 et le traitement termine selon le cycle {27, 45, 09, 81, 63} ;
- pour un nombre entier compris entre 000 à 999 : au premier pas le nombre obtenu est le point fixe 000 si tous les chiffres sont égaux, sinon c'est un multiple non nul de 99 inférieur à 1000 et en considérant chacun de ces multiples, on constate que le point fixe 495 est toujours atteint ;
- pour un nombre entier compris entre 0000 à 9999 : on montre de même que dans tous les cas, un des points fixes 0000 ou 6174 est atteint ;
- pour un nombre entier compris entre 00000 à 99999 : on montre que l'on atteint le

[40] Nous nous limitons ici au système décimal

point fixe 00000 ou que le traitement termine sur un des cycles {53955, 59994}, {62964, 71973, 83952, 74943} ou {61974, 82962, 75933, 63954}.

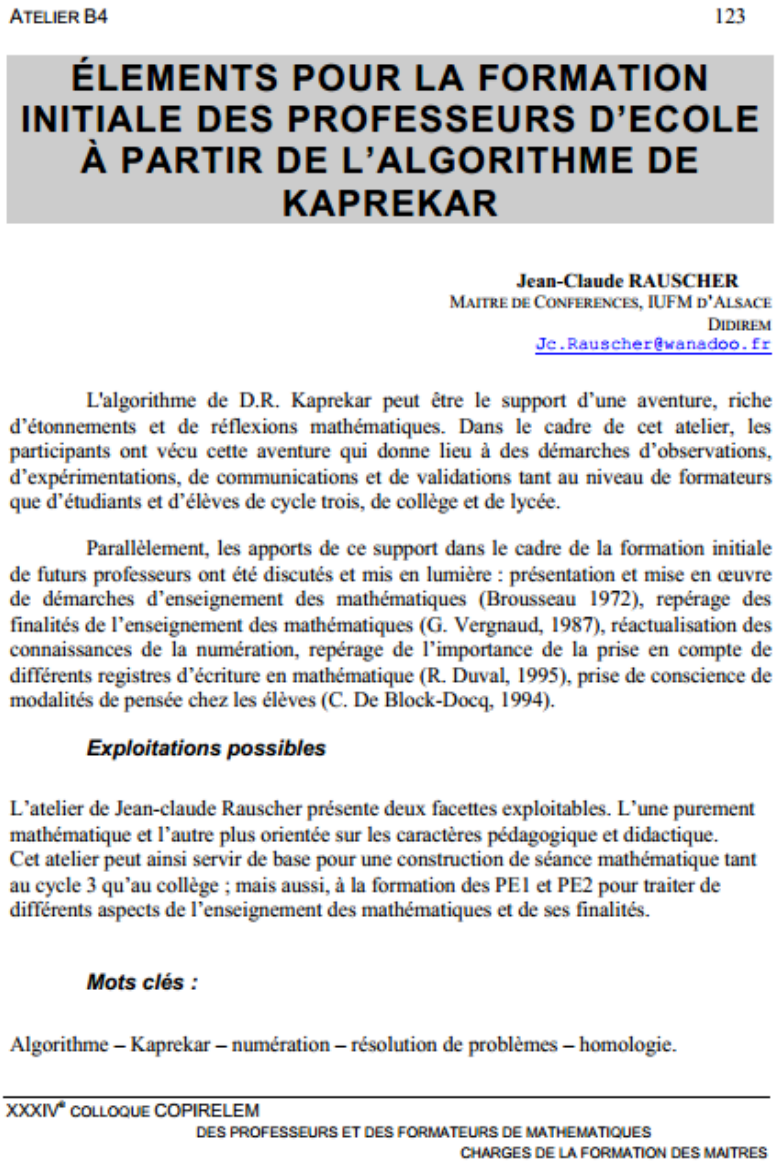
3.3 Un algorithme qui trouve sa place à tous les niveaux scolaires de l'élémentaire au supérieur – Le choix du niveau

Dans un premier temps, nous avons cherché à étudier un algorithme qui pouvait avoir sa place dans différents niveaux de la scolarité obligatoire. Ainsi, cette étude nous a amené à choisir un algorithme qui pouvait être exploité à différents niveaux allant du cycle 3 de l'enseignement élémentaire, à l'enseignement supérieur. En effet, comme nous le verrons par la suite, cet algorithme engendre des phénomènes remarquables, qui peuvent conduire à un travail de preuves faisant interagir algorithmique et raisonnement mathématique dans des domaines de l'algèbre et de l'arithmétique. Les questions que suscitent ces phénomènes font de cet algorithme un support pour différentes activités d'apprentissage et par conséquent de compétences en classe.

Au cours des années 80, Chauvat utilise cet algorithme pour proposer des tâches dans le cadre d'un apprentissage dans le domaine de la *théorie élémentaire des nombres* à des classes de Cinquième. Pour cela, il utilise la notion de *classe d'équivalence*.

Bien que de nos jours ce concept de *classe d'équivalence* ne soit plus inscrit dans les programmes du secondaire, le travail de numération mis en place avec cet algorithme le rend toujours très attractif dans l'enseignement des classes de Collège. Par ailleurs, l'introduction à partir de la rentrée 2016, d'un enseignement de l'algorithmique au Collège justifie aussi l'emploi de cet algorithme à ce niveau, en particulier pour les différents aspects algorithmique (« planification » de plusieurs sous-algorithmes nécessitant l'utilisation de variables itératives et de différentes structures de boucles, et « preuve de terminaison » de l'algorithme) qu'il permet d'aborder. Le fait que cet algorithme nécessite l'élaboration de plusieurs sous-algorithmes indépendants peut permettre de procéder à un travail collaboratif entre les élèves. Des phases « papier-crayon », algorithmique et programmation peuvent alterner et permettre de faire que le contrat didactique laisse à la charge des élèves une découverte et une compréhension en totale autonomie de l'algorithme comme nous verrons lors du déroulement des phases de l'ingénierie.

Par ailleurs, dans le cadre de la formation initiale des professeurs des écoles, Rauscher a proposé des tâches en liaison avec cet algorithme (fig. 22) :



ATELIER B4 123

ÉLEMENTS POUR LA FORMATION INITIALE DES PROFESSEURS D'ECOLE À PARTIR DE L'ALGORITHME DE KAPREKAR

Jean-Claude RAUSCHER
MAITRE DE CONFERENCES, IUFM D'ALSACE
DIDIREM
Jc.Rauscher@wanadoo.fr

L'algorithme de D.R. Kaprekar peut être le support d'une aventure, riche d'étonnements et de réflexions mathématiques. Dans le cadre de cet atelier, les participants ont vécu cette aventure qui donne lieu à des démarches d'observations, d'expérimentations, de communications et de validations tant au niveau de formateurs que d'étudiants et d'élèves de cycle trois, de collège et de lycée.

Parallèlement, les apports de ce support dans le cadre de la formation initiale de futurs professeurs ont été discutés et mis en lumière : présentation et mise en œuvre de démarches d'enseignement des mathématiques (Brousseau 1972), repérage des finalités de l'enseignement des mathématiques (G. Vergnaud, 1987), réactualisation des connaissances de la numération, repérage de l'importance de la prise en compte de différents registres d'écriture en mathématique (R. Duval, 1995), prise de conscience de modalités de pensée chez les élèves (C. De Block-Docq, 1994).

Exploitations possibles

L'atelier de Jean-claude Rauscher présente deux facettes exploitables. L'une purement mathématique et l'autre plus orientée sur les caractères pédagogique et didactique.
Cet atelier peut ainsi servir de base pour une construction de séance mathématique tant au cycle 3 qu'au collège ; mais aussi, à la formation des PE1 et PE2 pour traiter de différents aspects de l'enseignement des mathématiques et de ses finalités.

Mots clés :

Algorithme – Kaprekar – numération – résolution de problèmes – homologie.

Figure 22 (Kaprekar – Extrait du 34[e] colloque COPIRELEM)

En 2010, lors d'une épreuve d'admissibilité du CRPE[41], le sujet comportait un exercice sur cet algorithme numérique. En effet, les candidats devaient expérimenter dans une première phase le processus de calcul décrit sous forme algébrique :

> *« L'algorithme de Kaprekar consiste à associer à tout nombre entier naturel n le nombre K(n) généré de la façon suivante : – On considère les chiffres de l'écriture en base 10 du nombre n. On forme le nombre n_1 en rangeant ces chiffres dans*

[41] Concours de recrutement des Professeurs des Écoles

l'ordre croissant et le nombre n_2 en les rangeant dans l'ordre décroissant. – On pose $K(n) = n_2 - n_1$. On itère ensuite le processus en repartant du nombre K(n). »

Ensuite, un exemple numérique, en l'occurrence n = 634, était proposé faisant remarquer qu'au bout d'un certain nombre d'itération du processus, tous les résultats obtenus étaient égaux à 495. Ce résultat débutait à partir du moment où K(n) = n.

K(297) = 693 ; K(693) = 594 ; K(594) = 495 ; K(495) = 495.

Ensuite, tous les résultats sont égaux à 495.

1. Montrer que l'algorithme appliqué au nombre 5 294 conduit aussi à un nombre entier p tel que K(p) = p.

2. On considère maintenant un nombre m qui s'écrit en base dix avec les trois chiffres a, b et c, tels que : $m = \overline{abc}$ et $0 < a < b < c$.

a) Montrer que le nombre K(m) est un multiple de 99.

b) Montrer alors que l'algorithme appliqué au nombre m conduit au nombre 495 en cinq itérations au plus. »

Nous observons dans cet extrait d'exercice que l'objectif principal de l'auteur est d'amener le candidat à démontrer à l'aide d'un raisonnement algébrico-arithmétique certaines propriétés du processus de calcul. Il n'est pas demandé aux candidats d'utiliser un environnement numérique particulier afin de tester le processus décrit par l'auteur. L'auteur questionne le candidat sur le nombre maximal d'itérations pour obtenir le nombre 495 qui arrête le processus dans le cas où le nombre initial *m* est constitué de trois chiffres tous différents. Nous pouvons aussi observer que ce nombre *m* a ses trois chiffres triés dans l'ordre croissant. Ainsi, il suffit de permuter les chiffres *a* et *c* du nombre *m* pour obtenir ce deuxième nombre. La principale difficulté pour justifier que *K*(*m*) est un multiple de 99 va consister à reconnaitre l'écriture d'un nombre entier à trois chiffres dans le système décimal, c'est-à-dire qu'un tel nombre *m* s'écrit sous la forme $100a + 10b + c$ et que le deuxième nombre où les chiffres *a* et *c* auront été permutés s'écrit $100c + 10b + a$. De plus, le passage à l'étape sur la différence de ces deux nombres entiers va demander aux candidats de connaître des propriétés sur la factorisation et son interprétation en termes de divisibilité. Il faudra aussi, pour la question 2.b), que le candidat pense à appliquer le processus à chacun des multiples de 99, un raisonnement par disjonction de cinq cas qui n'est pas courant en mathématiques :

dans la majorité des raisonnements, on considère deux cas seulement, exceptionnellement trois.

Par ailleurs, l'introduction de l'algorithmique dans les nouveaux programmes du Secondaire (Lycée et Collège) rend l'étude de cet algorithme particulièrement pertinente dans les classes de Troisième, Seconde et du cycle Terminal Scientifique. Il en est de même des programmes des spécialités mathématiques ou d'informatique et sciences du numérique en Terminale Scientifique.

4. Analyse des programmes et de deux manuels en lien avec l'*algorithme de Kaprekar*

Nous souhaitons présenter les concepts de la *théorie élémentaire des nombres* se trouvant dans les programmes de Troisième et de Terminale Scientifique *enseignement de spécialité*, ainsi que sur l'approche algorithmique de ces concepts que propose l'institution. En effet, seuls ces deux niveaux scolaires instruisent officiellement un enseignement en classe de concepts en lien avec la *théorie élémentaire des nombres*.

Afin de compléter cette étude, nous étudions à travers certains manuels scolaires de Terminale Scientifique, ainsi que des documents ressources pour les classes de Terminale proposés par l'institution avec *Eduscol* et les IREM, comment des tâches en lien avec une approche algorithmique de la *théorie élémentaire des nombres* peuvent faire l'objet d'un travail en classe.

4.1 Le programme sur « Nombres et calculs » en classe Troisième

La classe de Troisième n'étant pas prise en compte directement dans notre ingénierie, nous nous contentons de proposer seulement une présentation de la rubrique « Nombres et calculs », située dans la partie intitulée « Nombres entiers et relationnels » pour le programme de 2008, et de la rubrique « Comprendre et utiliser les notions de divisibilité et de nombres premiers », située dans la partie intitulée « Nombres et calculs » pour celui de 2016.

Nous présentons ci-dessous (Tableau n° 4) les points importants du programme de 2008 qui sont en lien avec la *théorie élémentaire des nombres*.

Connaissances	Capacités	Commentaires
Nombres entiers et Rationnels Diviseurs communs à deux entiers, PGCD.	• *Connaître et utiliser un algorithme donnant le PGCD de deux entiers (algorithme des soustractions, algorithme d'Euclide).* • Calculer le PGCD de deux entiers. • *Déterminer si deux entiers donnés sont premiers entre eux.* Simplifier une fraction donnée pour la rendre irréductible.	*Plusieurs méthodes peuvent être envisagées.* La connaissance de relations arithmétiques entre nombres – que la pratique du calcul mental a permis de développer – permet d'identifier des diviseurs communs de deux entiers. Le recours à une décomposition en produits de facteurs premiers est possible dans des cas simples mais ne doit pas être systématisée. Les tableurs, calculatrices et logiciels de calcul formel sont exploités. […]

Tableau 4

A partir de la rentrée 2016, dans la continuité de ce qui est mis en place depuis de nombreuses années dans l'enseignement Primaire, le Collège va être découpé en cycle. Ainsi, la classe de Troisième va correspondre à la dernière année du cycle 4 qui est constitué des niveaux : Cinquième, Quatrième et Troisième. Dans la rubrique « Nombres et calculs » du cycle 4, les élèves vont être amenés « à consolider le sens des nombres et conforter la maitrise des procédures de calcul ». Les différentes composantes de ce thème (« Utiliser les nombres pour comparer, calculer et résoudre des problèmes – Comprendre et utiliser les notions de divisibilité et de nombres premiers – Utiliser le calcul littéral ») sont reliées entre elles. Par exemple, les élèves « peuvent prendre conscience du fait qu'un même nombre peut avoir plusieurs écritures (notamment écritures fractionnaire et décimale) et à l'occasion d'activités de recherche, ils peuvent rencontrer la notion de nombres irrationnels ». Nous résumons les directives sur les compétences attendues par l'institution sur les notions de divisibilité et de nombres premiers dans le tableau ci-dessous (Tableau n° 5).

Connaissances et compétences associées	Exemples de situations, d'activités et de ressources pour l'élève
[…]	[…]
Comprendre et utiliser les notions de divisibilité et de nombres premiers	
Déterminer si un entier est ou n'est pas multiple ou diviseur d'un autre entier. Simplifier une fraction donnée pour la rendre irréductible. • Division euclidienne (quotient, reste). • Multiples et diviseurs. • Notion de nombres premiers.	Recourir à une décomposition en facteurs premiers dans des cas simples. Exploiter tableurs, calculatrices et logiciels, par exemple pour chercher les diviseurs d'un nombre ou déterminer si un nombre est premier. Démontrer des critères de divisibilité (par exemple par 2, 3, 5 ou 10) ou la preuve par 9. Etudier des problèmes d'engrenages (par exemple braquets d'un vélo, rapports de transmission d'une boîte de vitesses, horloge), de conjonction de phénomènes périodiques (par exemple éclipses ou alignements de planètes).

Tableau 5

Ainsi, nous pouvons observer que dans le domaine de la *théorie élémentaire des nombres*, le terme d'« algorithme » n'est plus mentionné dans le programme de 2016. En revanche, l'exploitation de logiciels et du tableur sont toujours mentionnés en 2016. Cependant, en 2016 il n'est plus fait allusion au « calcul formel » comme en 2008. Nous pouvons penser que cela peut venir du fait que dans le cadre de la rénovation du Primaire et du Collège, un enseignement de l'algorithmique et de la programmation doit être explicitement donné aux élèves dès le cycle 3[42] dans le cadre d'un enseignement des mathématiques. Pendant la scolarité du cycle 4[43], les élèves poursuivent leur initiation à la programmation. Ainsi, « en créant un programme, ils développent des méthodes de programmation, revisitent les notions de variables et de fonctions sous une forme différente, et s'entraînent au raisonnement ».

Pour l'institution, les attendus de fin de cycle en algorithmique et en programmation consistent à ce que les élèves « sachent écrire, mettre au point et exécuter un programme simple ». Ceci se résumé dans le tableau (Tableau n° 6) ci-dessous.

[42] Les deux dernières années de l'Ecole Primaire : CM 1 et CM 2, et la première année du Collège : Sixième.
[43] Les trois dernières années du Collège : Cinquième, Quatrième et Troisième.

Connaissances et compétences associées	Exemples de situations, d'activités et de ressources pour l'élève
Décomposer un problème en sous-problèmes afin de structurer un programme ; reconnaître des schémas. Écrire, mettre au point (tester, corriger) et exécuter un programme en réponse à un problème donné. Écrire un programme dans lequel des actions sont déclenchées par des événements extérieurs. Programmer des scripts se déroulant en parallèle. • Notions d'algorithme et de programme. • Notion de variable informatique. • Déclenchement d'une action par un évènement, séquences d'instructions, boucles, instructions conditionnelles. • Notion de message échangé entre objets.	Jeux dans un labyrinthe, jeu de Pong, bataille navale, jeu de Nim, tic-tac-toe. Réalisation de figure à l'aide d'un logiciel de programmation pour consolider les notions de longueur et d'angle. Initiation au chiffrement (Morse, chiffre de César, code ASCII…) Construction de tables de conjugaison, de pluriels, jeu du cadavre exquis… Calculs simples de calendrier. Calculs de répertoire (recherche, recherche inversée, etc.). Calculs de fréquences d'apparition de chaque lettre dans un texte pour distinguer sa langue d'origine : français, anglais, italien, etc.

Tableau 6

Sachant que ce programme englobe les trois dernières années de la scolarité du collégien, nous pouvons supposer que selon celui-ci, il est attendu de la part les élèves qu'à l'entrée du lycée, ils aient eu une initiation « à la programmation évènementielle », puis développer progressivement « de nouvelles compétences, en programmant des actions en parallèle, en utilisant la notion de variable informatique, en découvrant les boucles et les instructions conditionnelles qui complètent les structures de contrôle liées aux événements », et qu'ils aient abordé la « gestion des objets », en mettant en pratique des échanges de messages.

Ainsi, tenant compte de ce nouvel apport sur l'algorithmique et ceci étant approfondi au cours des années de lycée, nous pouvons supposer qu'une approche algorithmique de certains concepts issus du domaine de la *théorie élémentaire des nombres* va pouvoir être approfondi dans les années futures. Ainsi, dans le programme de 2008, nous pouvons observer que la dimension algorithmique ne semble guère exister autrement que par une utilisation de méthodes numériques dont seuls les intitulés comportent le terme d'*algorithme* sans que pour autant l'aspect algorithmique soit explicitement prise en compte. En effet, il n'est nullement cité les aspects de structures algorithmiques, de variables informatiques,… dans le programme de 2008. Les environnements numériques proposés sont entre autres le tableur et le calcul formel qui ne favorisent pas nécessairement la prise de conscience de ces objets informatiques. En revanche, le nouveau programme de 2016 insiste sur l'algorithmique et la

programmation, bien qu'en même temps l'*algorithme d'Euclide* disparaît, alors que ce dernier aurait pu faire l'objet d'un travail liant mathématiques et algorithmique dès la classe de Troisième.

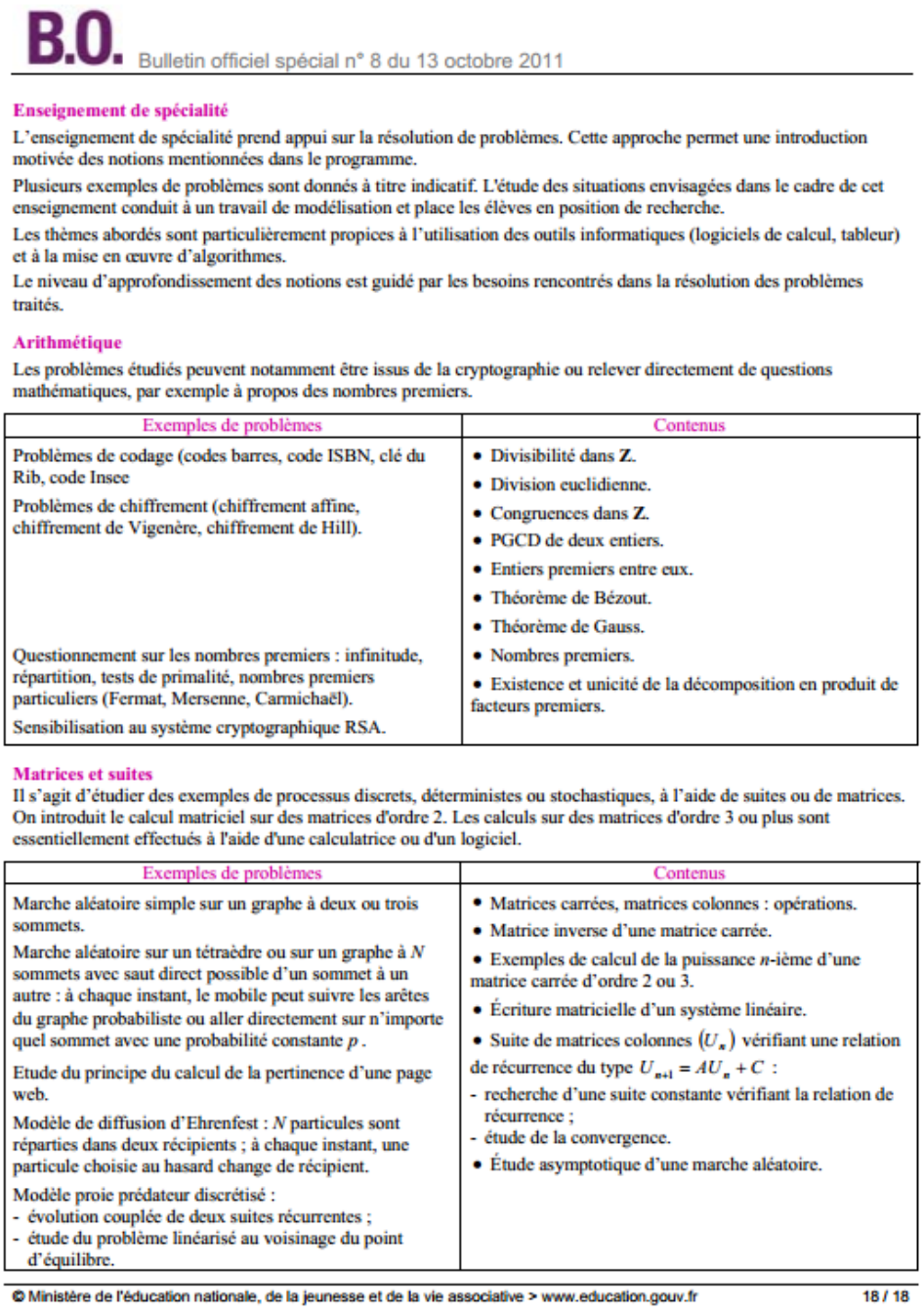
4.2 Le programme sur l'arithmétique en Terminale Scientifique enseignement de spécialité

4.2.1 Les objectifs généraux du programme de l'enseignement de spécialité

Comme le présente le programme (fig. 23) de l'*enseignement de spécialité* (*Bulletin officiel spécial n°8 du 13 octobre 2011*), l'enseignement de spécialité doit prendre « appui sur la résolution de problèmes ». De même, conformément à l'esprit de l'enseignement spécifique, le cycle Terminal Scientifique doit procurer à l'élève un bagage mathématique solide en particulier pour celui qui désire s'engager dans des études post-bac scientifiques.

L'étude des situations envisagées dans le cadre d'un tel enseignement de spécialité « conduit à un travail de modélisation et place l'élève en position de recherche. Les thèmes abordés en enseignement de spécialité sont particulièrement propices à l'utilisation d'outils informatiques (logiciels de calcul, logiciels algorithmiques, tableur) et à la mise en œuvre d'algorithmes » dans les domaines mathématiques comme l'arithmétique, le calcul matriciel et les suites. Le niveau d'approfondissement des notions traitées en enseignement de spécialité « est guidé par les besoins rencontrés dans la résolution des problèmes abordés ».

L'objectif principal est de former le futur étudiant à la pratique d'une démarche scientifique, en renforçant son goût pour des activités de recherche. En effet, l'apprentissage des mathématiques doit permettre à l'élève d'acquérir et de cultiver des compétences qui devront faciliter sa formation tout au long de sa vie et l'aider à mieux appréhender une société en perpétuelle évolution. Au-delà du cadre scolaire, le programme de mathématiques pour la Terminale Scientifique s'inscrit dans une perspective de formation de l'individu.



B.O. Bulletin officiel spécial n° 8 du 13 octobre 2011

Enseignement de spécialité

L'enseignement de spécialité prend appui sur la résolution de problèmes. Cette approche permet une introduction motivée des notions mentionnées dans le programme.

Plusieurs exemples de problèmes sont donnés à titre indicatif. L'étude des situations envisagées dans le cadre de cet enseignement conduit à un travail de modélisation et place les élèves en position de recherche.

Les thèmes abordés sont particulièrement propices à l'utilisation des outils informatiques (logiciels de calcul, tableur) et à la mise en œuvre d'algorithmes.

Le niveau d'approfondissement des notions est guidé par les besoins rencontrés dans la résolution des problèmes traités.

Arithmétique

Les problèmes étudiés peuvent notamment être issus de la cryptographie ou relever directement de questions mathématiques, par exemple à propos des nombres premiers.

Exemples de problèmes	Contenus
Problèmes de codage (codes barres, code ISBN, clé du Rib, code Insee Problèmes de chiffrement (chiffrement affine, chiffrement de Vigenère, chiffrement de Hill). Questionnement sur les nombres premiers : infinitude, répartition, tests de primalité, nombres premiers particuliers (Fermat, Mersenne, Carmichaël). Sensibilisation au système cryptographique RSA.	• Divisibilité dans **Z**. • Division euclidienne. • Congruences dans **Z**. • PGCD de deux entiers. • Entiers premiers entre eux. • Théorème de Bézout. • Théorème de Gauss. • Nombres premiers. • Existence et unicité de la décomposition en produit de facteurs premiers.

Matrices et suites

Il s'agit d'étudier des exemples de processus discrets, déterministes ou stochastiques, à l'aide de suites ou de matrices. On introduit le calcul matriciel sur des matrices d'ordre 2. Les calculs sur des matrices d'ordre 3 ou plus sont essentiellement effectués à l'aide d'une calculatrice ou d'un logiciel.

Exemples de problèmes	Contenus
Marche aléatoire simple sur un graphe à deux ou trois sommets. Marche aléatoire sur un tétraèdre ou sur un graphe à N sommets avec saut direct possible d'un sommet à un autre : à chaque instant, le mobile peut suivre les arêtes du graphe probabiliste ou aller directement sur n'importe quel sommet avec une probabilité constante p. Etude du principe du calcul de la pertinence d'une page web. Modèle de diffusion d'Ehrenfest : N particules sont réparties dans deux récipients ; à chaque instant, une particule choisie au hasard change de récipient. Modèle proie prédateur discrétisé : - évolution couplée de deux suites récurrentes ; - étude du problème linéarisé au voisinage du point d'équilibre.	• Matrices carrées, matrices colonnes : opérations. • Matrice inverse d'une matrice carrée. • Exemples de calcul de la puissance n-ième d'une matrice carrée d'ordre 2 ou 3. • Écriture matricielle d'un système linéaire. • Suite de matrices colonnes (U_n) vérifiant une relation de récurrence du type $U_{n+1} = AU_n + C$: - recherche d'une suite constante vérifiant la relation de récurrence ; - étude de la convergence. • Étude asymptotique d'une marche aléatoire.

Figure 23 (Extrait du programme obligatoire et spécialité de la Terminale Scientifique)

La lecture de ce programme, nous permet d'observer qu'un certain nombre de tâches proposées aux élèves peuvent combiner plusieurs notions mathématiques autour de la *théorie élémentaire des nombres* : divisibilité dans **Z**, division euclidienne, congruences dans **Z**, PGCD de deux entiers, …. Il semble aussi que la liste des exemples de problèmes mettant en lien l'algorithmique et la *théorie élémentaire des nombres* proposés par l'institution ne soit pas exhaustive : problèmes de codage, problèmes déchiffrement, test de primalité, cryptographie,… Cependant, il n'est pas explicitement dit pour chacun de ces thèmes quel

pourrait être l'apport de l'algorithmique pour la résolution de tels problèmes. L'aspect algorithmique est présenté au début de ce programme sous une forme littérale assez *vague* :

> « *Les thèmes abordés sont particulièrement propices à l'utilisation des outils informatiques (logiciels de calcul, tableur) et la mise en œuvre d'algorithmes* ».

Ainsi beaucoup de libertés pédagogique et didactique sont laissées à la responsabilité de l'enseignant quant aux choix des outils informatiques, des langages algorithmiques, … Nous verrons comment cela est interprété par les auteurs d'ouvrages scolaires lors de l'analyse de deux manuels que nous présentons plus loin.

4.2.2 Le programme et le domaine de l'arithmétique[44]

Dans le domaine de la *théorie élémentaire des nombres*, « les problèmes étudiés peuvent notamment être issus de la cryptographie ou relever directement de questions mathématiques, par exemple à propos des nombres premiers ». Nous rappelons ci-dessous (Tableau n° 7) les directives données par le programme pour l'enseignement de la *théorie élémentaire des nombres*.

Exemples de problèmes	Contenus
Problèmes de codage (codes-barres, code ISBN, clé du RIB, code INSEE. Problèmes de chiffrement (chiffrement affine, chiffrement de Vigenère, chiffrement de Hill). Questionnement sur les nombres premiers : infinitude, répartition, tests de primalité, nombres premiers particuliers (Fermat, Mersenne, Carmichaël). Sensibilisation au système cryptographique RSA.	• Divisibilité dans **Z**. • Division euclidienne. • Congruences dans **Z**. • PGCD de deux entiers. • Entiers premiers entre eux. • Théorème de Bézout. • Théorème de Gauss. • Nombres premiers. • Existence et unicité de la décomposition en produit de facteurs premiers.

Tableau 7

A la lecture de cet extrait du programme sur l'*arithmétique*, nous pouvons observer que l'introduction d'un enseignement de l'algorithmique en classe de mathématiques en Terminale Scientifique va nous permettre en tant que chercheur de favoriser la possibilité de diversifier les choix d'articulations entre divers *Espaces de Travail Mathématique spécifiques* : *algèbre* et *arithmétique* et *Espace de Travail Algorithmique* tout en faisant que notre ingénierie

[44] Pour le titre de ce paragraphe, nous choisissons de garder le terme institutionnel au lieu de parler *de Théorie élémentaires des nombres.*

va être porté par certains faits issus de la *Théorie des Situations Didactiques* (Brousseau, 1998).

Il semble aussi que la liste des exemples de problèmes proposés par l'institution ne soit pas exhaustive. En effet, dans le cadre d'un enseignement de la *théorie élémentaire des nombres*, une analyse de certains manuels scolaires et de documents ressources proposées par certaines instances proches de l'institution peut nous conforter dans ce constat.

4.3 Etude de deux manuels scolaires de Terminale proposant des tâches en lien avec l'algorithme de Kaprekar

Nous allons centrer cette étude sur l'algorithme que nous avons choisi comme support pour l'ingénierie. C'est pourquoi nous n'avons retenu que deux manuels ayant entre autres comme particularité de proposer une tâche sur la « recette de Kaprekar ». Il se trouve que ce sont des manuels de Terminale, car nous n'en avons pas trouvé pour la classe de Troisième. Nous considérons ainsi qu'il s'agit d'un éventail assez large, nous permettant de situer nos choix pour l'ingénierie parmi les travaux qui peuvent être proposés aux élèves dans le contexte de notre recherche. Pour ces deux manuels, notre objectif est d'étudier le type d'exercices proposés sur la « recette de Kaprekar » et son introduction tant dans la partie cours que dans celle des *activités* proposées. Pour cela, nous nous limitons à une présentation du cours et à un repérage des tâches liées à notre problématique.

Afin de compléter ces présentations sur ces deux manuels, nous faisons aussi un rapide point sur la façon dont d'autres ressources reconnues par l'institution considèrent une introduction de la « recette de Kaprekar » comme objet d'apprentissage de la *planification* et de la *preuve* chez des élèves en fin de cycle terminal Scientifique.

4.3.1 Manuel Hyperbole (Edition 2011), enseignement de spécialité

Ce manuel est découpé en cinq chapitres traitant des deux domaines de l'enseignement de spécialité : (1) *Arithmétique* ; (2) *Matrices et suites*. La partie « Arithmétique » est constituée de trois chapitres : – *Divisibilité dans Z* ; – *Théorème de Bézout, Théorème de Gauss* ; – *Nombres premiers*. La structure de chaque chapitre est la suivante : – une rubrique « ouverture » constituée d'un document ouvrant le chapitre sur le monde qui entoure l'élève en fonction du contenu du chapitre, ainsi que d'énigmes aidant l'élèves à réactiver ses connaissances anciennes nécessaires pour entrer dans le chapitre. Cette « ouverture » est

complétée de thèmes. Pour chaque thème, conformément au programme, est proposée une entrée par « problème ». Ces thèmes sont suivis du cours qui présente les définitions et les propriétés illustrées d'exemples et d'exercices résolus afin d'aider l'élève à dégager des méthodes de résolution de problèmes. Le chapitre se termine par de nombreux exercices de difficultés progressives le tout étant partagé en exercices d'application, d'entraînement et d'approfondissement.

Dans le chapitre « Divisibilité dans Z », les auteurs proposent un exercice (fig. 24) intitulé « Un algorithme à explorer » dans le cadre du B2i[45]. Cet exercice se situe dans la rubrique « Porter un regard critique » de la série « Exercices d'entraînement ». Il est extrait des « Epreuves pratiques ».

Porter un regard critique

114 Un algorithme à explorer B2i

N désigne un nombre entier naturel de **trois chiffres** dont deux au moins sont distincts. On range ses chiffres dans l'ordre croissant, on obtient un nombre P. On range les chiffres de N dans l'ordre décroissant, on obtient un nombre G. On calcule les différences D = G – P.
Le nombre D devient le nombre initial et on recommence le calcul. L'algorithme s'arrête lorsqu'on obtient deux fois de suite le même nombre D.
Exemple sur les deux premières étapes : N = 878 donne P = 788 et G = 887 d'où D = 887 – 788 = 99. On remplace N par 99, soit N = 099 qui donne P = 99 et G = 990, d'où D = 891…

a) Entrer un nombre à trois chiffres dans la cellule A1 d'un tableur. Tester les formules :
=ENT(A1/100) et =MOD(ENT(A1/10);10).
Que retournent ces formules ? Expliquer.

b) Réaliser à l'aide du tableur le tableau suivant en utilisant les fonctions ENT, MOD, MIN, MAX, MEDIANE.

N	Centaines	Dizaines	Unités	P	G	D
878	8	7	8	788	887	99
99	0	9	9	99	990	891

c) Appliquer l'algorithme aux nombres 787, 877, 794, 425, puis à différents nombres de trois chiffres.

Qu'observe-t-on ? Émettre des conjectures sur la suite des nombres obtenus dans la colonne A.

d) Valider ou invalider les conjectures émises.

D'après *Épreuves pratiques*

Figure 24 (Extrait d'un exercice du manuel Hyperbole (Ed. 2011) – Kaprekar)

Comme nous pouvons l'observer, l'exercice démarre par une présentation d'un processus

[45] Le Brevet Informatique et Internet (Les B2i école, collège et lycée sont mis en œuvre depuis la rentrée 2012. Ils permettent d'attester le niveau acquis par les élèves dans la maîtrise des outils multimédia et de l'internet. Ils permettent aussi de mieux préparer les élèves à un usage responsable des technologies de l'information et de la communication.)

générique de calcul :

> « N désigne un nombre entier naturel de trois chiffres dont deux au moins sont distincts.
> On range ses chiffres dans l'ordre croissant, on obtient un nombre P.
> On range les chiffres de N dans l'ordre décroissant, on obtient un nombre G.
> On calcule les différences D = G – P.
> Le nombre D devient le nombre initial et on recommence le calcul.
> L'algorithme s'arrête lorsqu'on obtient deux fois de suite le même nombre D. […]
> »

Nous pouvons observer que les auteurs semblent considérer certaines étapes de l'algorithme comme étant implicites. En effet, ils ne mentionnent pas les étapes de déconstruction et de reconstruction des trois nombres entiers qui interviennent dans le processus : décomposition de l'entier initial, reconstruction des deux nombres entiers après les processus de tri dans l'ordre croissant, puis décroissant des chiffres constituant le nombre initial. De plus, la proposition d'un test d'arrêt « L'algorithme s'arrête lorsqu'on obtient deux fois de suite le même nombre D » par les auteurs de l'exercice laisse supposer qu'ils n'attendent pas une phase où l'élève expérimenterait le processus manuellement afin d'émettre une conjecture sur la terminaison de l'algorithme. Un exemple est proposé par les auteurs sur un nombre entier donné qui permet d'illustrer partiellement le processus de calcul à travers une description des « deux premières » étapes : « […] N = 878 donne P = 788 et G = 887 d'où D = 887 – 788 = 99. On remplace N par 99, soit N = 099 qui donne P = 099 et G = 990, d'où D = 891… ».

Dans cet exemple, nous constatons que les auteurs considèrent toujours la phase de déconstruction et de reconstruction des entiers comme implicites. Le choix et la gestion des variables itératives nécessaires au fonctionnement de l'algorithme dans un langage informatique semblent aussi rester du domaine de l'implicite par les auteurs. Le choix du logiciel de traitement du processus est imposé par les auteurs. En effet, dans le cadre d'une approche de type B2i, ils guident l'élève à mener un travail sur tableur en donnant deux formules `=ENT(A1/100)` et `=MOD(ENT(A1/10);10)`, où A1 correspond à la cellule contenant le nombre entier initial à trois chiffres. Ils attendent que l'élève explique le rôle de ces formules en les entrant dans le tableur. Par tâtonnement, l'élève doit reconnaitre le rôle de chacune de ces formules puisque les auteurs demandent de tester les formules sans préciser cependant les cellules où doivent être inscrites ces deux formules. Nous pouvons supposer que la

connaissance du tableur en place depuis le Collège chez les élèves, leur permet d'assumer cette prise d'initiative sur le choix des cellules. Une fois que ces formules sont inscrites dans les cellules correspondantes, les élèves reconnaissent après les avoir testée (tâche demandée par les auteurs) sur plusieurs nombres entiers de trois chiffres entrer dans la cellule A1 que la première formule donne le chiffre des centaines du nombre inscrit dans A1 et le chiffre des dizaines de ce même nombre. Cependant, il est demandé aux élèves par les auteurs de ne pas se contenter d'une série de tests sur ces formules pour conclure. En effet, les auteurs attendent que les élèves testent puis expliquent le pourquoi des résultats obtenus. Cet aspect peut être considéré d'une certaine manière comme une approche algorithmique du processus de décomposition d'un nombre entier à trois chiffres. Une fois que cela est fait, les auteurs demandent aux élèves de réaliser un tableau à l'aide du tableur en inscrivant différentes colonnes représentant le nombre entier N, puis la décomposition de ce nombre N et les reconstructions des deux nombres entiers constitués des chiffres triés de N respectivement dans l'ordre décroissant (pour le nombre P) et croissant (pour le nombre G). Pour ces deux reconstructions, les auteurs précisent les fonctions qui doivent être utilisées. En effet, ils donnent les fonctions « ENT, MOD, MIN, MAX, MEDIANE » sans pour autant expliquer à quoi correspondent chacune de ces fonctions. Cet aspect est laissé à l'initiative des élèves. Le tableau donné par les auteurs montre aussi que ces derniers choisissent de donner une organisation, sous forme de colonnes, des différents résultats pour chaque étape du processus : les décompositions des nombres entiers en chiffre des centaines, des dizaines et des unités, complétés des nombres P, G et D pour chaque étape de calcul. La gestion des variables itératives nécessaires à l'exécution de l'algorithme reste ainsi toujours de l'ordre de l'implicite. De plus, chaque étape semble être constituée de plusieurs sous-étapes qui peuvent ne pas être reconnues par un élève débutant en informatique. On peut supposer que les auteurs attendent une application d'un processus de calcul au lieu d'une introduction à un raisonnement de type algorithmique. Nous constatons aussi que le choix des fonctions imposées par les auteurs renvoie l'élève à utiliser des connaissances qu'il côtoie d'habitude dans divers domaines des mathématiques comme les statistiques, l'analyse, l'arithmétique,…

Le tableau proposé par les auteurs peut être aussi une source d'incompréhension par l'élève débutant en informatique. En effet, l'élève peut ne pas penser en voyant le tableau donné par les auteurs que celui-ci peut nécessiter d'autres lignes de calcul pour décrire le

processus de calcul à mener jusqu'à ce que l'on obtienne par une série de « copier-glisser » le « test d'arrêt » proposé par les auteurs, c'est-à-dire « L'algorithme s'arrête lorsque l'on obtient deux fois de suite le même nombre D. »

Nous pouvons faire l'hypothèse que la formulation de cet exercice peut ne pas favoriser une prise de conscience chez l'élève du rôle de l'algorithmique dans cette tâche en particulier sur une *planification* nécessaire permettant de construire un algorithme complexe, ainsi que pour la mise en place *d'une preuve empirique, puis algébrique justifiant le choix de la condition de* terminaison de l'algorithme. De même que le choix des variables itératives et de la structure algorithmique nécessaire pour le bon fonctionnement de l'algorithme semblent être totalement implicite et totalement contextualisé n'offrant pas ainsi à l'élève des outils facilement réutilisables ou transposables à d'autres types de tâches en lien avec des mises en relation entre structures algorithmiques et organisation de variables itératives sur des algorithmes complexes. Nous pouvons supposer que les validations attendues par les auteurs en fin d'exercice, d'un certain nombre de conjectures sur les nombres entiers obtenus à chaque étape semblent rester dans le registre algébrique sans pour autant renvoyer l'élève à une preuve de type algorithmique. Les auteurs ne proposent pas une preuve « par exhaustion »[46] où les élèves auraient fait tourner l'algorithme sur tous les nombres entiers compris entre 000 et 999, dont deux des trois chiffres constituant le nombre entier seraient au moins distincts.

De plus, nous pouvons observer que cette dernière condition sur le choix des chiffres du nombre entier initial ne permet pas d'approfondir un test de sortie de l'algorithme. En effet, si les auteurs avaient autorisé le fait que les trois chiffres soient égaux, le test de sortie de l'algorithme aurait été différent et permis à l'élève de mieux « porter un regard critique » (intitulation de cette rubrique d'exercices) sur des conditions faisant intervenir les connecteurs logiques « ou » et « et ». Comme nous pourrons le voir lors de l'analyse de notre ingénierie, ce choix de laisser les trois chiffres du nombre entier initial égaux a été l'objet d'échanges et de débat entre les élèves.

Dans le tableau ci-dessous (Tableau 5), nous proposons une étude de l'exercice en fonction des *paradigmes algorithmiques* et des articulations entre *Espaces de Travail*

[46] Un prédicat P étant donné sur un intervalle fini E, nous appelons « preuve par exhaustion » de la proposition « quel que soit *e* appartenant à E, P(*e*) », la vérification une par une de toutes les propositions P(*e*), *e* appartenant à E. Ne pas confondre avec la méthode d'exhaustion ou méthode d'Archimède pour le calcul d'aire.

Mathématique/Algorithmique privilégiés qu'il met en jeu.

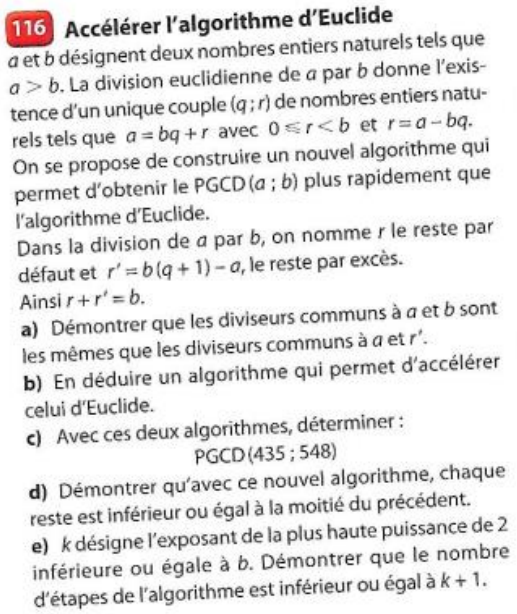
<table>
<tr><td>Thème de l'exercice</td><td colspan="2">Un algorithme à explorer (fig. 24)</td></tr>
<tr><td>Paradigmes</td><td colspan="2">Tout au long de l'exercice, nous nous situons au niveau 1 des paradigmes de l'algorithmique. En effet, dans cet exercice, le travail demandé à l'élève suppose une représentation des objets et des actions élémentaires et la construction d'un traitement :
- Il faut distinguer le nombre et ses chiffres et créer des formules permettant de passer d'une représentation à une autre.
- Il faut itérer le traitement.
Cependant, cette représentation et ce traitement restent dans un cadre non axiomatique, ce qui caractérise le niveau 1:
- L'itération s'opère dans le tableur, et utilise la recopie incrémentée de cellules, l'arrêt étant géré par l'utilisateur et non par l'algorithme.
- Il est demandé aux élèves « de valider ou d'invalider » une conjecture, mais ceci n'est pas mis en relation avec une propriété de terminaison de l'algorithme.
- Implicitement, « valider ou invalider » va conduire à considérer le sous-ensemble des nombres obtenus à la première itération. Cette « réduction » n'est pas exploitée en relation avec la complexité d'une preuve par algorithme.</td></tr>
<tr><td rowspan="2">Espaces de Travail spécifiques</td><td>ETA</td><td>ETM</td></tr>
<tr><td>L'élève travaille avec le tableur. Des formules lui sont données. Il doit expliquer ce qu'elles retournent.
Un tableau type est aussi proposé avec la description de chacune des colonnes utilisées.
Le travail attendu dans les quatre premières questions (sur cinq) fait que l'élève reste dans des genèses instrumentale et sémiotique.
Une genèse discursive qui porterait sur une preuve de terminaison n'est pas envisagée. Elle serait compliquée à mettre en place avec un tableur.</td><td>Dans la dernière question l'élève doit « valider ou invalider » la conjecture faite dans l'ETA. Puisqu'il n'est pas fait référence au travail dans l'ETA des questions précédentes, et dans l'esprit de l'épreuve pratique, il semble que le travail attendu par les auteurs dans cette question cantonne les élèves dans une genèse discursive dans l'ETM. On aurait pu s'attendre à ce que les genèses instrumentale et sémiotique développées dans l'ETA soient exploitées pour complémenter cette genèse discursive dans l'ETM.</td></tr>
</table>

Tableau 8

Pour compléter notre analyse de ce manuel sur cet aspect algorithmique dans le cadre de la divisibilité dans **N**, nous pouvons observer que conformément aux attentes du programme, les auteurs proposent une démonstration de l'*algorithme d'Euclide* pour le calcul du PGCD de deux nombres entiers. Cette démonstration doit permettre à l'élève de prendre conscience de l'importance de construire une preuve sur la validité d'un algorithme. Elle se situe dans la partie « cours » de l'ouvrage. Elle est découpée en deux étapes : une démonstration du lemme d'Euclide : « a, b, q et r désignent des nombres entiers relatifs non nuls. Si $a = b\,q + r$, alors PGCD (a ; b) = PGCD (b ; r) » suivie d'une démonstration de la propriété : « Soient a et b deux nombres entiers naturels non nuls tels que $a > b$. Lorsque b ne divise pas a, le PGCD des nombres entiers naturels non nuls a et b est égal au dernier reste non nul obtenu par l'*algorithme d'Euclide*. ». Ainsi, le lemme donné précédemment est une proposition dont la

démonstration est préliminaire à cette dernière propriété. De plus, bien que le paragraphe du cours sur le PGCD soit donné sur deux nombres entiers relatifs non nuls, la dernière propriété ainsi que sa démonstration est faite sur **N***. En effet, les auteurs précisent que « deux nombres entiers opposés ont le même ensemble de diviseurs. C'est pourquoi on ne s'intéresse […] qu'au cas de nombres entiers naturels ».

Dans la rubrique « Porter un regard critique » des « exercices d'entrainement » où nous avions trouvé l'exercice sur l'*algorithme de Kaprekar*, les auteurs proposent un exercice (fig. 25) en lien avec l'*algorithme d'Euclide*, où les élèves doivent déterminer, après justification, un algorithme qui permettrait d'accélérer celui d'Euclide.



116 Accélérer l'algorithme d'Euclide

a et b désignent deux nombres entiers naturels tels que $a > b$. La division euclidienne de a par b donne l'existence d'un unique couple $(q ; r)$ de nombres entiers naturels tels que $a = bq + r$ avec $0 \leqslant r < b$ et $r = a - bq$.
On se propose de construire un nouvel algorithme qui permet d'obtenir le PGCD $(a ; b)$ plus rapidement que l'algorithme d'Euclide.
Dans la division de a par b, on nomme r le reste par défaut et $r' = b(q + 1) - a$, le reste par excès.
Ainsi $r + r' = b$.

a) Démontrer que les diviseurs communs à a et b sont les mêmes que les diviseurs communs à a et r'.

b) En déduire un algorithme qui permet d'accélérer celui d'Euclide.

c) Avec ces deux algorithmes, déterminer :
PGCD (435 ; 548)

d) Démontrer qu'avec ce nouvel algorithme, chaque reste est inférieur ou égal à la moitié du précédent.

e) k désigne l'exposant de la plus haute puissance de 2 inférieure ou égale à b. Démontrer que le nombre d'étapes de l'algorithme est inférieur ou égal à $k + 1$.

Figure 25 (Extrait d'un exercice du manuel Hyperbole – Algorithme d'Euclide)

Cet exercice se situe deux exercices après celui de Kaprekar que nous avons décrit précédemment. Nous pouvons observer que celui-ci ne porte pas la référence B2i[47] et qu'il est explicitement inscrit dans la *preuve de terminaison* d'un algorithme. De plus, il n'est pas demandé aux élèves, une fois que le nouvel algorithme est construit, de l'implémenter dans un environnement informatique afin de le tester. Les élèves tout au long de cet exercice restent dans un registre algébrico-arithmétique, et l'aspect algorithmique semble ne faire appel qu'au langage « naturel », voire « pseudo-code ». Ainsi, les auteurs ne semblent pas attendre des élèves qu'ils fassent référence à tel ou tel langage informatique.

A travers une analyse en termes de *paradigmes algorithmiques* et d'articulations entre *ETA*

[47] Brevet informatique et internet

et *ETM*, nous pouvons observer que bien que ces deux exercices (Kaprekar (Fig. 24) et Euclide (Fig. 25)) se situent dans le même domaine de la *théorie élémentaire des nombres*, ils ne renvoient pas exactement aux mêmes *paradigmes algorithmiques* et aux mêmes *Espaces de Travails* spécifiques (Tableau 9).

<table>
<tr><td>Thème de l'exercice</td><td colspan="2">Accélérer l'algorithme d'Euclide (fig. 25)</td></tr>
<tr><td>Paradigmes</td><td colspan="2">Il s'agit clairement d'une tâche située au niveau 2 des paradigmes de l'algorithmique, tels que nous les avons définis au chapitre 2 (cadre théorique) de la partie 1, en rupture avec l'approche de niveau 1 de l'exercice précédent. Dans ce paradigme de niveau 2, la source de validation se fonde sur des lois hypothético-déductives dans un « système axiomatique » aussi précis que possible. Mais le problème du choix des « axiomes » utilisés se pose. La relation avec la « réalité » subsiste encore dans ce niveau 2 des paradigmes algorithmiques, dans la mesure où elle s'est constituée pour organiser les connaissances algorithmiques issues ici de problèmes sur la détermination du PGCD[48] de deux nombres entiers. L'« axiomatisation » proposée est certes une formalisation, mais elle n'est pas formelle car ici la syntaxe n'est pas coupée de la sémantique qui renvoie à la « réalité ». D'où la conservation du qualificatif de « naturelle » pour ce niveau de paradigme. En effet, dans cet exercice, l'algorithmique peut s'exercer moyennant une « axiomatisation partielle », voire des « îlots d'axiomatisation » (Kuzniak, Houdement, 2006).</td></tr>
<tr><td rowspan="2">Espaces de Travail spécifiques</td><td>ETA</td><td>ETM</td></tr>
<tr><td>Dans cet exercice, l'élève n'est pas sollicité par les auteurs à mettre en place un travail dans un ETA spécifique.
La genèse utilisée ici est du type discursif.
La question sur la détermination du PGCD des entiers 435 et 548, peut très bien être faite au « papier-crayon » sans une utilisation d'un environnement informatique spécifique.</td><td>L'élève devant démontrer que les diviseurs communs aux nombres entiers a et b sont les mêmes que les diviseurs communs à a et r', où r' est le reste par excès de la division euclidienne de a par b, justifie la mise en action d'une genèse discursive dans un ETM. Il en est de même pour le travail demandé sur le nombre d'étapes de l'algorithme.</td></tr>
</table>

Tableau 9

Ainsi, les articulations entre différents *Espaces de Travail* mis en jeu dans ce type d'exercices ne semblent pas favoriser une articulation entre *ETA et ETM spécifique* qui serait ici : *théorie élémentaire des nombres*.

Dans le premier exercice (Fig. 24), comme nous avons pu l'observer, l'élève centre son travail pour l'essentiel dans un *ETA*, autour d'une genèse instrumentale. Cependant, dans le cadre de l'élaboration d'une preuve de la conjecture ce dernier devra articuler celle-ci entre *ETA* et *ETM* afin de formaliser une preuve en limitant le nombre de cas à vérifier. Ceci conduit l'élève à mener aussi un travail dans l'*ETM*, autour d'une genèse discursive, bien qu'aucune indication, dans la conception d'une telle preuve, ne soit donnée par les auteurs de l'exercice.

Dans le second exercice (Fig. 25), l'*ETA* ne vient qu'en second plan. En effet, le travail

[48] Plus Grand Commun Diviseur

demandé se situe plus dans un *ETM*, en particulier pour la genèse discursive. La genèse instrumentale à laquelle pourrait se référer l'élève dans un *ETA* spécifique semble ne pas être réellement prise en compte par les auteurs de ce deuxième exercice.

En revanche, mener un travail sur ce type d'exercices peut avoir une implication pertinente pour notre ingénierie (Tableau 10).

Implication pour notre ingénierie.	Nous constatons que : 1. Le travail reste au niveau 1 en algorithmique : itération « naturelle », pas de problématique de terminaison et de coût d'une preuve. 2. Les genèses instrumentale et sémiotique dans l'*ETA* ne viennent ni supporter la genèse discursive dans l'*ETM*, ni engager une genèse discursive dans l'*ETA*. Nous souhaitons, dans notre ingénierie, élaborer des tâches tirant parti des genèses instrumentale et sémiotique dans l'*ETA* et articulant des genèses discursives dans l'*ETM* et l'*ETA*, autour de la terminaison et de l'efficacité donc vers le niveau 2 de l'algorithmique. Ceci sera présenté en 5.2.

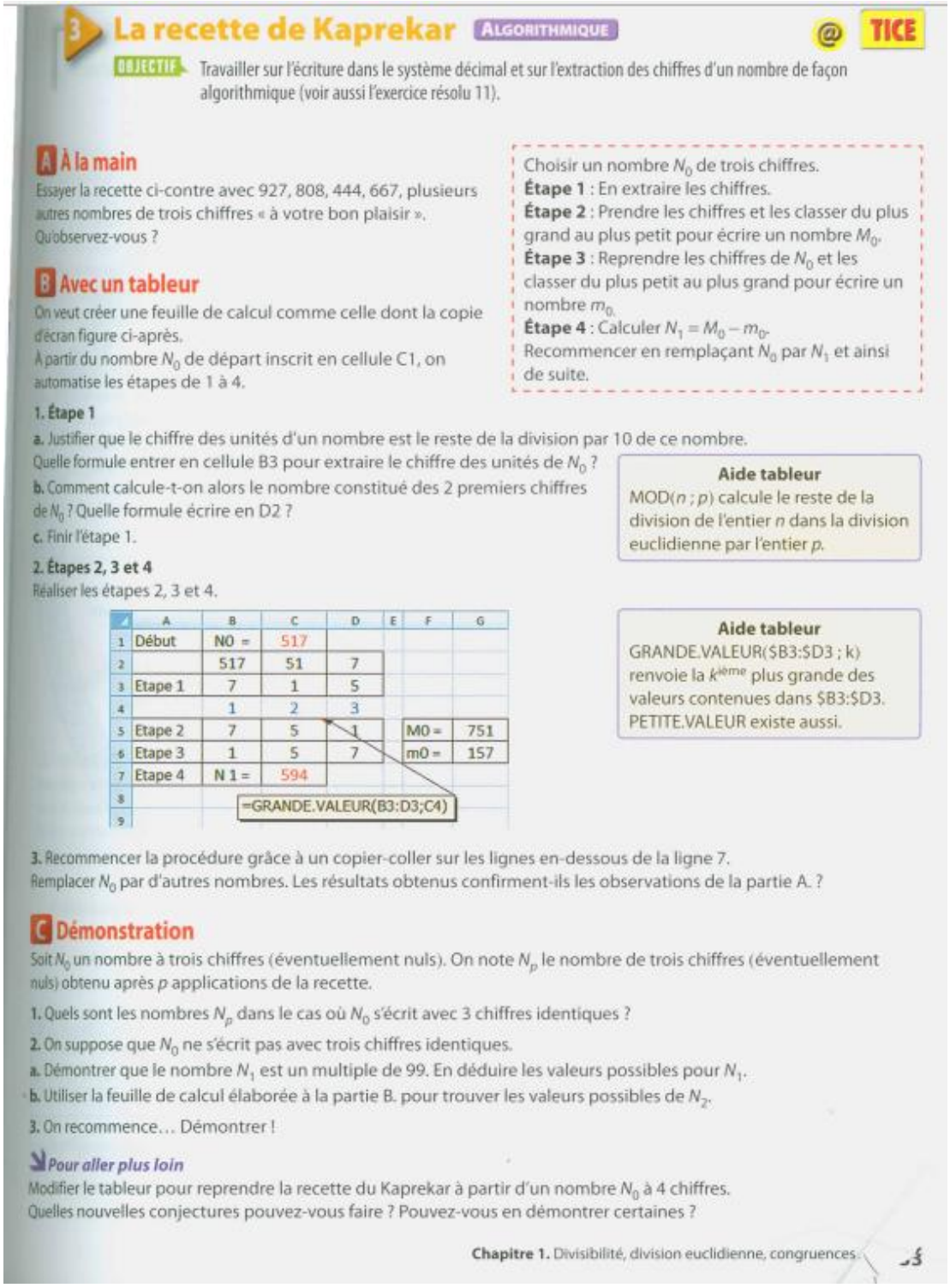
Tableau 10

4.3.2 Manuel Math'x, enseignement de spécialité

Ce manuel est aussi constitué de cinq chapitres recouvrant l'ensemble du programme. La partie sur l'arithmétique se découpe en trois chapitres : (1) *Divisibilité, division euclidienne, congruences* ; (2) *Nombres premiers* ; (3) *PGCD, théorème de Bézout, théorème de Gauss*. La structure de chaque chapitre est classique. Elle est constituée d'une « ouverture » et d'une rubrique « Résolution de problèmes » introduisant les différents concepts qui vont être mis en place dans le chapitre. Les auteurs proposent ensuite un « cours » donnant les définitions et les propriétés complétées de leurs démonstrations ainsi que des exercices résolus permettant à l'élève d'acquérir des méthodes de raisonnement suivant les problématiques mises en jeu. La rubrique « exercices » venant après le « cours » est constituée dans un premier temps d'exercices guidés dont certains sont accompagnés d'éléments de réponses, puis de nombreux problèmes sont proposés dans le cadre de travaux dirigés où certains peuvent être en lien avec d'autres sciences et enfin une liste d'exercices classés par thèmes et dont la difficulté est progressive clos le chapitre.

Dans le chapitre « Divisibilité, division euclidienne, congruences », à la rubrique « Résolution de problèmes », nous trouvons un énoncé intitulé « La recette de Kaprekar ». Les auteurs situent cet exercice (Fig. 26) dans le cadre d'un enseignement des TICE et de

l'algorithmique.



3 La recette de Kaprekar ALGORITHMIQUE @ TICE

OBJECTIF Travailler sur l'écriture dans le système décimal et sur l'extraction des chiffres d'un nombre de façon algorithmique (voir aussi l'exercice résolu 11).

A À la main

Essayer la recette ci-contre avec 927, 808, 444, 667, plusieurs autres nombres de trois chiffres « à votre bon plaisir ».
Qu'observez-vous ?

Choisir un nombre N_0 de trois chiffres.
Étape 1 : En extraire les chiffres.
Étape 2 : Prendre les chiffres et les classer du plus grand au plus petit pour écrire un nombre M_0.
Étape 3 : Reprendre les chiffres de N_0 et les classer du plus petit au plus grand pour écrire un nombre m_0.
Étape 4 : Calculer $N_1 = M_0 - m_0$.
Recommencer en remplaçant N_0 par N_1 et ainsi de suite.

B Avec un tableur

On veut créer une feuille de calcul comme celle dont la copie d'écran figure ci-après.
À partir du nombre N_0 de départ inscrit en cellule C1, on automatise les étapes de 1 à 4.

1. Étape 1

a. Justifier que le chiffre des unités d'un nombre est le reste de la division par 10 de ce nombre.
Quelle formule entrer en cellule B3 pour extraire le chiffre des unités de N_0 ?
b. Comment calcule-t-on alors le nombre constitué des 2 premiers chiffres de N_0 ? Quelle formule écrire en D2 ?
c. Finir l'étape 1.

Aide tableur
MOD(n ; p) calcule le reste de la division de l'entier n dans la division euclidienne par l'entier p.

2. Étapes 2, 3 et 4

Réaliser les étapes 2, 3 et 4.

	A	B	C	D	E	F	G
1	Début	N0 =	517				
2		517	51	7			
3	Etape 1	7	1	5			
4		1	2	3			
5	Etape 2	7	5	1		M0 =	751
6	Etape 3	1	5	7		m0 =	157
7	Etape 4	N 1 =	594				
8							
9							

=GRANDE.VALEUR(B3:D3;C4)

Aide tableur
GRANDE.VALEUR($B3:$D3 ; k) renvoie la $k^{ième}$ plus grande des valeurs contenues dans $B3:$D3.
PETITE.VALEUR existe aussi.

3. Recommencer la procédure grâce à un copier-coller sur les lignes en-dessous de la ligne 7.
Remplacer N_0 par d'autres nombres. Les résultats obtenus confirment-ils les observations de la partie A. ?

C Démonstration

Soit N_0 un nombre à trois chiffres (éventuellement nuls). On note N_p le nombre de trois chiffres (éventuellement nuls) obtenu après p applications de la recette.

1. Quels sont les nombres N_p dans le cas où N_0 s'écrit avec 3 chiffres identiques ?

2. On suppose que N_0 ne s'écrit pas avec trois chiffres identiques.
a. Démontrer que le nombre N_1 est un multiple de 99. En déduire les valeurs possibles pour N_1.
b. Utiliser la feuille de calcul élaborée à la partie B. pour trouver les valeurs possibles de N_2.

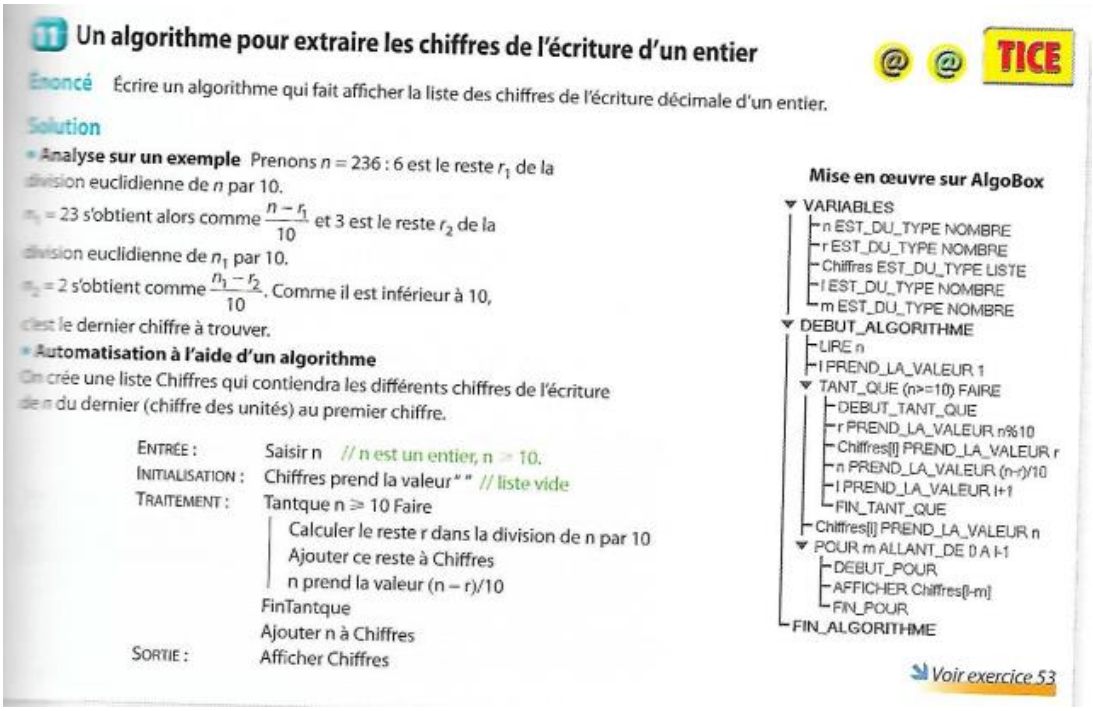
3. On recommence… Démontrer !

Pour aller plus loin

Modifier le tableur pour reprendre la recette du Kaprekar à partir d'un nombre N_0 à 4 chiffres.
Quelles nouvelles conjectures pouvez-vous faire ? Pouvez-vous en démontrer certaines ?

Figure 26 (Extrait d'un TP du manuel Math'x – La recette de Kaprekar)

Cet exercice renvoie à un exercice corrigé (Fig. 27) se trouvant à la fin de la série d'« exercices résolus » qui suit la partie « Congruences dans Z » du cours.



11 Un algorithme pour extraire les chiffres de l'écriture d'un entier TICE

Énoncé Écrire un algorithme qui fait afficher la liste des chiffres de l'écriture décimale d'un entier.

Solution

• **Analyse sur un exemple** Prenons $n = 236$: 6 est le reste r_1 de la division euclidienne de n par 10.

$n_1 = 23$ s'obtient alors comme $\frac{n - r_1}{10}$ et 3 est le reste r_2 de la division euclidienne de n_1 par 10.

$n_2 = 2$ s'obtient comme $\frac{n_1 - r_2}{10}$. Comme il est inférieur à 10, c'est le dernier chiffre à trouver.

• **Automatisation à l'aide d'un algorithme**

On crée une liste Chiffres qui contiendra les différents chiffres de l'écriture de n du dernier (chiffre des unités) au premier chiffre.

```
ENTRÉE :        Saisir n    // n est un entier, n ⩾ 10.
INITIALISATION : Chiffres prend la valeur " "  // liste vide
TRAITEMENT :    Tantque n ⩾ 10 Faire
                    Calculer le reste r dans la division de n par 10
                    Ajouter ce reste à Chiffres
                    n prend la valeur (n – r)/10
                FinTantque
                Ajouter n à Chiffres
SORTIE :        Afficher Chiffres
```

Mise en œuvre sur AlgoBox

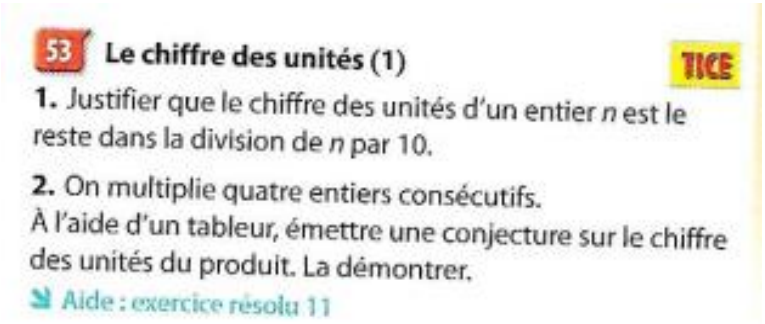
```
VARIABLES
    n EST_DU_TYPE NOMBRE
    r EST_DU_TYPE NOMBRE
    Chiffres EST_DU_TYPE LISTE
    I EST_DU_TYPE NOMBRE
    m EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
    LIRE n
    I PREND_LA_VALEUR 1
    TANT_QUE (n>=10) FAIRE
        DEBUT_TANT_QUE
        r PREND_LA_VALEUR n%10
        Chiffres[I] PREND_LA_VALEUR r
        n PREND_LA_VALEUR (n-r)/10
        I PREND_LA_VALEUR I+1
        FIN_TANT_QUE
    Chiffres[I] PREND_LA_VALEUR n
    POUR m ALLANT_DE 0 A I-1
        DEBUT_POUR
        AFFICHER Chiffres[I-m]
        FIN_POUR
FIN_ALGORITHME
```

Voir exercice 53

Figure 27 (Extrait d'un TP du manuel Math'x – Algorithme d'extraction des chiffres d'un entier)

Dans cet exercice résolu, les auteurs proposent une tâche sur un algorithme pour extraire les chiffres de l'écriture d'un nombre entier naturel. Cette tâche peut être considérée comme préparatoire à l'*algorithme de Kaprekar*, comme le signalent les auteurs dans l'objectif de l'exercice sur Kaprekar : « Travailler sur l'écriture dans le système décimal et sur l'extraction des chiffres d'un nombre de façon algorithmique. » Malgré une solution complète proposée par les auteurs, l'énoncé est donné sous forme de problème ouvert : « Ecrire un algorithme qui fait afficher la liste des chiffres de l'écriture d'un entier ». La correction est constituée dans une première approche d'une analyse sur un exemple numérique basée sur le reste dans la division euclidienne, puis d'une automatisation à l'aide d'un algorithme du raisonnement mis en place lors de l'analyse précédente. Un algorithme écrit en langage naturel est alors donné par les auteurs. Cet algorithme nécessite de créer une liste « Chiffres » qui contient les différents chiffres de l'écriture de l'entier n ($n \geq 10$) constitué du dernier chiffre « de droite » (chiffre des unités) au premier chiffre « de gauche ». Enfin, les auteurs donnent une mise en œuvre de cet algorithme sur un logiciel de programmation *AlgoBox* utilisant une variable de type LISTE.

De plus, il est à noter que les auteurs, en fin d'exercice, renvoient à un exercice ayant pour thème « Le chiffre des unités » d'un entier n (Fig. 28).



53 **Le chiffre des unités (1)** TICE

1. Justifier que le chiffre des unités d'un entier *n* est le reste dans la division de *n* par 10.

2. On multiplie quatre entiers consécutifs.
À l'aide d'un tableur, émettre une conjecture sur le chiffre des unités du produit. La démontrer.

Aide : exercice résolu 11

Figure 28 (Extrait d'un TP du manuel Math'x – Les chiffres des unités)

En effet, dans ce nouvel exercice, il est demandé aux élèves de justifier que le chiffre des unités d'un entier *n* est le reste dans la division euclidienne de *n* par 10. Puis, en multipliant quatre entiers consécutifs, à l'aide d'un tableur, les élèves doivent émettre une conjecture sur le chiffre des unités de ce produit et la démontrer. Cet exercice se situe dans une perspective TICE. Ces deux exercices (fig. 27 et 28) sont préparatoires à l'étape de décomposition d'un nombre entier nécessaire dans l'*algorithme de Kaprekar*. Ils permettent d'approcher l'algorithme complexe qu'est celui de Kaprekar en préparant un début de *planification* par l'introduction d'un algorithme de décomposition nécessaire au bon déroulement de l'*algorithme de Kaprekar*. Comme pour le manuel précédent, les auteurs n'insistent pas sur les différents concepts algorithmiques nécessaires pour la construction de cet algorithme d'extraction des chiffres de l'écriture d'un nombre entier. Les variables itératives et les structures de boucles « TantQue… Faire » et « Pour » données dans l'exercice corrigé (Fig. 27) ne font pas l'objet de questionnement de la part des auteurs. En effet, il n'est pas demandé à l'élève de construire cet algorithme puisque sa mise en œuvre dans un langage informatique est déjà proposée et de plus, les auteurs ne leur demandent pas de l'implémenter dans un environnement informatique afin de le tester. Cela est laissé à la curiosité de l'élève. Quant à l'autre exercice sur la reconnaissance du chiffre des unités d'un produit de nombres entiers (Fig. 28), les élèves doivent utiliser un tableur pour émettre une conjecture sur le chiffre des unités du produit de quatre nombres entiers consécutifs, puis la démontrer. On peut penser alors que les auteurs considèrent que la preuve doit mettre en œuvre un calcul algébrique éventuellement appuyé par un raisonnement mettant en œuvre la congruence modulo 5. En effet, aucune question sur la structure de l'algorithme associée à ce produit et à la reconnaissance du chiffre des unités de cet entier n'est demandée par les auteurs de l'exercice.

Dans le problème sur la « recette de Kaprekar » (Fig. 26), un processus de calcul est

donné par les auteurs :

> « Choisir un nombre N_0 de trois chiffres.
> **Etape 1 :** En extraire les chiffres ;
> **Etape 2 :** Prendre les chiffres et les classer du plus grand au plus petit pour écrire un nombre M_0 ;
> **Etape 3 :** Reprendre les chiffres de N_0 et les classes du plus petit au plus grand pour écrire un nombre m_0 ;
> **Etape 4 :** Calculer $N_1 = M_0 – m_0$;
> Recommencer en remplaçant N_0 par N_1 et ainsi de suite. »

Le problème est ensuite découpé en trois parties : (A) *A la main* ; (B) *Avec un tableur* ; (C) *Démonstration*. Nous observons que cet énoncé est très détaillé. En effet, il est précisé qu'il faut tout d'abord extraire les chiffres du nombre entier initial. Par ailleurs, après une première partie où les élèves prennent un premier contact « manuel » avec l'algorithme lors de son exécution « à la main », les élèves sont invités à exploiter un tableur. Ainsi, le choix du logiciel n'est pas laissé aux élèves. Ensuite, les auteurs annoncent que l'objectif de la partie B est non pas de faire exécuter l'algorithme, mais de créer une feuille de calcul comme celle dont la copie d'écran (Fig. 29) est donnée par les auteurs. Nous constatons aussi que l'étape d'extraction des chiffres du nombre entier initial est extrêmement détaillée. En effet, les auteurs ne demandent aux élèves que de justifier que le chiffre des unités d'un nombre entier est le reste de la division par 10 de ce nombre. Pour cela, une utilisation de la fonction « MOD » est suggérée dans un encadré intitulé « Aide tableur ». De plus, nous observons que les étapes de tri sont également très guidées. En effet, les auteurs leur suggèrent dans un deuxième encadré toujours intitulé « Aide tableur » d'utiliser les fonctions « GRANDE.VALEUR » et « PETITE.VALEUR » suivant le tri choisit. Seule, pour cette phase de « planification », les reconstructions des nombres, dont les chiffres sont triés, sont à la charge des élèves.

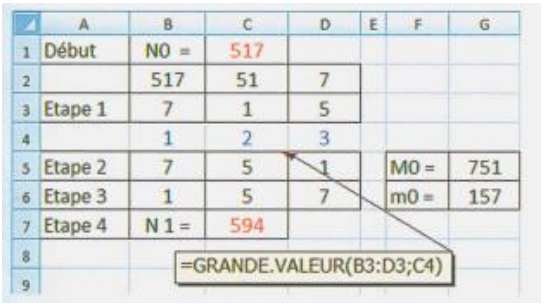


Figure 29 (Copie d'écran d'une feuille de calcul – Tris décroissant et croissant de trois chiffres)

Pour compléter notre analyse de la partie « Avec un tableur », nous observons que pour l'étape 1 (extraction des chiffres), les auteurs attendent que les élèves justifient que le chiffre des unités d'un nombre entier est le reste de la division euclidienne par 10 de ce nombre,

avant d'utiliser une fonction (proposée par les auteurs) permettant l'extraction de ce chiffre des unités et de l'inscrire dans une cellule du tableur. Cette étape peut renvoyer les élèves aux exercices que nous avons décrits précédemment, mais, comme nous le disions cela n'est pas nécessaire vu les aides proposées par les auteurs du manuel. Ensuite, les auteurs demandent aux élèves d'expliquer comment il est possible de calculer le nombre constitué des deux premiers chiffres du nombre et de proposer une formule permettant d'obtenir ce nombre des dizaines (expression non donnée par les auteurs) et enfin de terminer le processus de calcul permettant de finir l'étape 1.

Dans la partie « Démonstration », les auteurs guident les élèves vers un raisonnement algébrique permettant d'obtenir une preuve de la conjecture par limitation à cinq cas.

Nous pouvons observer que les auteurs n'attendent pas une preuve « par exhaustion » qui utiliserait la feuille du tableur pour appliquer le traitement aux mille entiers considérés. L'utilisation du tableur renvoie à un travail s'inscrivant dans le cadre d'une « application numérique » de cette « recette de Kaprekar ». La structure algorithmique concernant cette « recette » n'est pas considérée comme pouvant faire l'objet d'un questionnement de la part des élèves. Tout le travail autour de la démonstration est aussi fortement guidé. En effet, dans un premier temps, il est demandé aux élèves d'étudier le résultat que l'on obtient avec un nombre entier de trois chiffres égaux puis de démontrer le cas où au moins un des chiffres est différent des autres. Dans cette deuxième situation, les auteurs guident encore les élèves en leur demandant de justifier que le premier nombre N_1 obtenu après une exécution des quatre étapes du processus de calcul est bien un multiple de 99, puis d'en déduire les valeurs possibles de ce nombre N_1. Ensuite, les auteurs renvoient à une utilisation de la feuille de calcul de la partie B pour trouver le deuxième nombre N_2 après une deuxième exécution des quatre étapes du processus de calcul. Les auteurs terminent cet aspect de la démonstration en demandant aux élèves de recommencer ce qu'ils viennent de décrire précisément dans les questions précédentes et de démontrer la conjecture. Nous pouvons supposer que cette dernière étape du raisonnement est laissée à l'initiative des élèves car aucune indication particulière n'est donnée à ce stade de l'énoncé.

Les articulations entre différents *ETM* spécifiques mis en jeu dans cet exercice ne semblent pas favoriser un travail de genèse discursive dans un *ETA*. Seul, un *Espace de Travail Mathématique* dans les domaines de l'algèbre et de l'arithmétique sur le plan genèse

discursive semble pris en compte par les auteurs de cet exercice.

Cependant, l'exercice se termine par une question :

> « Modifier le tableur pour reprendre la recette de Kaprekar à partir d'un nombre N_0 à 4 chiffres. Quelles nouvelles conjectures pouvez-vous faire ? Pouvez-vous en démontrer certaines ? »

La formulation « ouverte » de cette question pourrait laisser supposer que l'élève, contrairement aux premières parties sur le cas de nombres entiers à trois chiffres va être amené à prendre des initiatives dans sa démarche de preuve de la conjecture. Cependant, il est fort probable que les élèves vont se contenter de transposer le travail guidé fait sur des nombres à trois chiffres, à celui à mener pour des nombres à quatre chiffres.

4.4 Que retenir des analyses des programmes en lien avec le domaine de l'arithmétique et des deux manuels proposant des tâches sur l'*algorithme de Kaprekar*, voire de l'*algorithme d'Euclide*, dans la perspective des analyses concernant notre ingénierie, en fonction de notre cadre théorique ?

A la lecture du programme sur l'*arithmétique* en Terminale Scientifique et des analyses faites sur des tâches proposées par certains manuels en lien avec notre ingénierie, nous pouvons supposer qu'une introduction d'un enseignement de l'algorithmique en classe de mathématiques en Terminale Scientifique peut favoriser les analyses de tâches issues du domaine de la *théorie élémentaire des nombres*, à travers la possibilité de diversifier les choix des *Espaces de Travail Mathématique spécifiques* (*ETM$_s$*). De plus, ces *ETMs* vont permettre d'observer les articulations entre les genèses dans des *ETA*, et des ETM *algébrique* et *arithmétique*.

5. L'ingénierie

Nous rappelons que notre ingénierie se base sur la « recette » suivante :

(1)	Choisir un nombre entier naturel de trois chiffres (c.-à-d. compris entre 000 et 999.
(2)	Former le nombre entier obtenu en rangeant les chiffres du nombre choisi en (1) dans l'ordre croissant.
(3)	Former le nombre entier obtenu en rangeant les chiffres du nombre choisi en (1) dans l'ordre décroissant.
(4)	Calculer la différence des nombres entiers obtenus en (2) et (3).
(5)	Recommencer le processus à partir de l'instruction (2), avec le résultat obtenu

en (3). Etc.

Nous précisons d'abord quelques éléments de l'« ingénierie-Guy » en justifiant notre choix de placer notre ingénierie à sa suite. Puis nous présentons notre ingénierie.

5.1 L'« ingénierie-Guy »

5.1.1 Problématique et tâches dans l'« ingénierie-Guy »

La problématique de l'« ingénierie-Guy » concerne la capacité des élèves à déterminer des étapes dans un traitement complexe, compatibles avec une exécution par un dispositif informatique (Rogalski), en mobilisant des connaissances mathématiques. Elle s'inscrit dans la problématique de la planification au sens de Rogalski et Samurçay (1990).

Au cours de cette « ingénierie-Guy » les élèves doivent d'abord déterminer une conjecture correspondant à la « recette » donnée ci-dessus. Les phases de l'« ingénierie-Guy » sont : **(1)** détermination d'une conjecture à l'aide du papier-crayon ; **(2)** planification des différents étapes en lien avec la « recette » Kaprekar ; **(3)** conception et implémentation de ces sous-algorithmes dans un environnement de programmation ; **(5)** assemblage de ces sous-algorithmes afin d'obtenir l'algorithme complet pour l'implémenter dans l'environnement de programmation ; **(6)** test de l'algorithme complet sur des nombres entiers compris entre 000 et 999, choisis de façon arbitraire. Dans cette « ingénierie-Guy » la preuve de la conjecture n'est pas demandée aux élèves: elle est l'objet de notre ingénierie.

Lors de la première phase, les élèves établissent une conjecture qui nécessite de leur part une compréhension de la « recette » de Kaprekar. Puis, dans la seconde phase, ils doivent reconnaitre, pour un algorithme à effectuer par un dispositif informatique (Rogalski, 1986), la nécessité d'étapes non présentes dans un traitement manuel : décomposition du nombre en chiffres, recomposition à partir des chiffres. Dans la troisième phase, ils doivent concevoir ces étapes, ainsi que d'autres déjà présentes dans le traitement manuel (entrée, tri des chiffres, soustraction et sortie du résultat) comme des sous-algorithmes. La présence d'étapes non existantes dans le traitement manuel (décomposition – recomposition) impose de mobiliser des connaissances en numération normalement acquises au niveau de la Terminale Scientifique (choix fait pour l'« ingénierie-Guy »), mais qui pourraient être un objectif d'apprentissage si l'on veut exploiter cet algorithme au niveau des classes du cycle 4

(Collège) : les élèves doivent concevoir le calcul du chiffre des unités comme reste dans la division euclidienne, puis celui des dizaines (resp. des centaines) par le même calcul, cette fois sur nombre divisé par 10 (resp. 100). Concernant l'étape de la recomposition du nombre à partir de ses chiffres, la stratégie va consister à déterminer à partir des chiffres *a*, *b*, et *c* le nombre $\overline{abc}$ = 100 × *a* + 10 × *b* + *c*.

Par ailleurs, le tri de trois chiffres qui est immédiat dans un traitement manuel impose une construction tenant compte des possibilités de la machine : plusieurs sont possibles, mais nous ne les détaillons pas car elles n'ont pas d'incidence sur notre ingénierie.

5.1.2 La mise en œuvre : la classe, la salle informatique, les binômes

L'« ingénierie-Guy » est mise en place dans une classe de Terminale Scientifique, enseignement de spécialité, avec des élèves ayant un bon niveau en Mathématiques et en informatique. Elle a nécessité trois heures de travail découpés en une première séance de deux heures suivie d'une seconde séance d'une une heure à une semaine de décalage. Pendant, la totalité de l'ingénierie les élèves sont en salle informatique avec un ordinateur pour deux élèves. Lors du travail sur la conjecture (environ 10 minutes), les ordinateurs de la salle sont en mode « veille ». Une fois que cette conjecture est testée sur quelques exemples à l'aide du « papier-crayon » et que l'ensemble de la classe a approuvé la conjecture formulée, les ordinateurs sont activés et les élèves ouvrent le logiciel *AlgoBox* installé sur tous les PC. Ce logiciel est le seul autorisé pour cette ingénierie.

5.1.3 Les connaissances des élèves à l'issue de cette de l'« ingénierie-Guy »

Suite au travail accompli par les binômes au cours de l'« ingénierie-Guy », les binômes ont reconnu les différentes étapes nécessaires pour un algorithme conçu pour une machine, ainsi que les variables informatiques nécessaires pour la conception des sous-algorithmes et leur assemblage dans un seul algorithme. Ils ont ainsi connaissance de procédures de décomposition et de recomposition d'un nombre entier à trois chiffres, impliquant notamment la présence de trois variables pour les chiffres et d'une formule algébrique traduisant la décomposition-recomposition du nombre en chiffres.

La conjecture ayant été aussi vérifiée sur un certain nombre d'entiers, les élèves peuvent d'eux-mêmes proposer une modification de l'algorithme afin d'aller vers une preuve de la conjecture. Ceci est l'objet de notre ingénierie.

5.2 Notre ingénierie

5.2.1 Nos motivations (choix généraux)

5.2.1.1. Le point de vue « algorithme »

Nous avons justifié notre intérêt pour la « recette de Kaprekar » par les liens qui existaient entre celui-ci et des notions de la *théorie élémentaire des nombres* enseignées au lycée. Cette « recette » peut s'inscrire dans le curriculum de Terminale Scientifique comme un algorithme permettant un travail de recherche dans un domaine des mathématiques reconnu par l'institution. Ainsi, nous parlerons d'*algorithme de Kaprekar* et non plus de « recette ». En effet, celui-ci respecte les diverses propriétés (au sens de Knuth) que doivent vérifier une procédure de calcul afin d'être considérée comme étant un algorithme :

- **La validité (effectivité) :** « La validité d'un algorithme est son aptitude à réaliser exactement la tâche pour laquelle il a été conçu ». Ici, la tâche à réaliser consiste à obtenir l'égalité entre deux nombres entiers, à partir d'un nombre entier de trois chiffres choisi par l'utilisateur.

- **La robustesse :** « La robustesse d'un algorithme est son aptitude à se protéger de conditions anormales d'utilisation ». Ici, la robustesse de l'algorithme se pose si le nombre choisi initialement est un nombre de quatre chiffres.

- **La réutilisabilité d'un algorithme :** « La réutilisabilité d'un algorithme est son aptitude à être réutilisé pour résoudre des tâches équivalentes à celle pour laquelle il a été conçu ». Ici, l'algorithme est-il réutilisable tel quel si le nombre initial est donné en base 2 ?

- **La complexité d'un algorithme :** « C'est le nombre d'instructions élémentaires à exécuter pour réaliser la tâche pour laquelle l'algorithme a été conçu ». Ici, la complexité de l'algorithme peut se compter en nombre d'étapes maximum à faire pour

vérifier la conjecture.

- **L'efficacité d'un algorithme :** « C'est son aptitude à utiliser de manière optimale les ressources du matériel qui l'exécute ». Si, l'on cherche à déterminer une preuve de la conjecture n'existerait-il pas un nombre restreint d'entiers à vérifier ?

Par conséquent, cet algorithme peut donner lieu à un travail de preuve. Contrairement à des algorithmes comme celui de l'*algorithme d'Euclide*, il ne fait pas partie intégrante du cours, ce qui convient bien à notre choix d'une ingénierie locale pour ce chapitre.

Le choix de placer notre ingénierie à la suite de l'« ingénierie-Guy » en classe de Terminale présente de nombreux avantages quant aux prérequis des élèves. Les structures algorithmiques mises en place lors de l'élaboration de l'*algorithme de Kaprekar* ne doivent pas poser de difficulté. En effet, l'utilisation de boucles « TantQue » et « Pour » font partie des connaissances anciennes chez des élèves de Terminale. De plus, celles-ci ont fait l'objet d'un travail approfondi lors de l'« ingénierie Guy », les élèves ont au début de notre ingénierie un algorithme qui tourne et qu'ils ont eu l'occasion de tester sur des nombres entiers pris au hasard. De même, les difficultés autour de la planification des différents sous-algorithmes constituant la « recette » de Kaprekar et l'élaboration d'un test de sortie lors de l'« ingénierie-Guy » ne posent plus de problèmes aux élèves lors de notre ingénierie, ainsi les élèves peuvent concentrer leurs efforts sur la preuve.

5.2.1.2. Le point de vue « preuve »

Les problématiques liées à la preuve sont nombreuses en algorithmique. Nous distinguons une première problématique interne à l'algorithmique. Des preuves sont construites pour répondre à des questions telles que : cet algorithme termine-t-il ? Si oui, le résultat obtenu est-il bien une solution au problème posé (effectivité) ? Une mesure de complexité étant précisée, pour le même problème, peut-on comparer deux algorithmes (efficacité) ? Bien qu'internes à l'algorithmique, ces preuves peuvent mettre en jeu des éléments mathématiques. Par exemple, les preuves de l'*algorithme d'Euclide* mettent en jeu le formalisme des suites indexées et des propriétés de **N**, comme « *Toute partie non vide de* N *admet un plus petit élément* ».

Une seconde problématique est celle des « preuves assistées ». Dans beaucoup de domaines, la puissance de calcul des ordinateurs et l'élaboration de langages formels pour représenter un domaine, sont mises à profits pour des « preuves assistées ». En Mathématiques, ceci a des conséquences du fait de l'obtention de résultats nouveaux, mais aussi du fait de la structuration et de la formalisation nécessaire pour un traitement algorithmique. Ainsi, par exemple, la conjecture des quatre couleurs est devenue un théorème, et la coloration des cartes a été formalisée en graphe. Le travail d'élaboration de preuves assistées conduit aussi à des développements en informatique théorique, du fait de la nécessité de valider les algorithmes à l'œuvre dans une preuve. Les deux problématiques sont donc liées.

L'hypothèse à la base de notre ingénierie est qu'il est possible de faire travailler des élèves sur ces problématiques et que ce travail met en jeu les interactions entre *Espaces de Travail Mathématique* et *Algorithmique*. Nous allons préciser ce travail en nous situant dans une perspective de « transposition didactique » de pratiques de recherche (Lagrange, 2007). La pratique que nous allons transposer est relative à la recherche de la distance maximale dans le problème du *Rubik's Cube*[49].

Le résultat est le suivant : « Every position of *Rubik's Cube*™ can be solved in twenty moves or less[50] ». Nous rappelons que le nombre de positions du *Rubik's Cube* est de 43 252 003 274 489 856 000. La méthodologie est ainsi énoncée:

1) Les 43 252 003 274 489 856 000 positions du cube ont été réparties en 2 217 093 120 ensembles de 19 508 428 800 configurations chacune (Ceci permet à chaque sous-problème de tenir dans la mémoire d'un PC moderne).
2) Le nombre d'ensemble a été réduit à 55 882 296 en utilisant des symétries et ensembles couvrants.
3) Les solutions recherchées n'étaient pas les optimales pour chaque position, mais celles de 20 mouvements ou moins (étant donné que la limite inférieure était déjà à 20 depuis 1995).
4) Un programme (au sens de *code source*) permet de résoudre chaque position (donc de trouver le nombre de coup minimal) en 20 secondes environ.
5) 35 années de temps CPU (offertes par Google), on permit de trouver toutes les solutions pour chacun des 55 882 296 ensembles.

[49] http://www.cube20.org/ (Consulté le 10 août 2017)
[50] Chaque position du Rubik's Cube™ peut être résolu en vingt mouvements ou moins.

La partition à l'étape 1 utilise la théorie des groupes (classes suivant un sous-groupe, ou *cosets*). Elle permet d'une part la réduction des classes à l'étape 2 grâce aux relations de symétrie qui permet un gain d'efficacité de l'ordre de 40. Elle permet un autre gain d'efficacité grâce à la conception d'un algorithme opérant un traitement spécifique de chaque classe : en utilisant cet algorithme, les ordinateurs considérés opèrent un million de vérification par seconde à l'intérieur d'une classe donnée, alors qu'une vérification sans considération de classe ne permet d'en traiter que 3 900 par seconde. La partition rend aussi possible à l'étape 5 de « diviser le travail » de façon à utiliser un nombre d'ordinateurs en parallèle suffisant (mais non précisé) pour obtenir la vérification en un temps raisonnable. Les auteurs déclarent avoir construit leur algorithme « *using a combination of mathematical tricks and careful programming* ».

Pour nous, cet exemple montre la pertinence des interactions entre *Espaces de Travail Mathématique* et *Algorithmique* dans une démarche de preuve. Les étapes 1 et 2 sont d'ordre mathématique, tout en étant orientées par le souci d'efficacité dans la construction d'un algorithme. Les étapes 3, 4 et 5 sont d'ordre algorithmique, tout en combinant des « artifices mathématiques » à la conception.

Nous avons observé plus haut que les preuves classiques concernant l'*algorithme de Kaprekar* telles qu'elles existent dans les manuels ou l'épreuve de concours analysée plus haut, utilisent aussi la réduction des cas grâce à un travail mathématique. Nous montrerons plus loin comment s'opère ce travail dans le cas de nombres à trois ou quatre chiffres. Le problème de la terminaison de cet algorithme et celui de la preuve de la distance maximale dans le cas du *Rubik's Cube* appartiennent donc à même classe : celle des problèmes portant sur un nombre fini de configurations initiales, où le traitement automatisé de chaque configuration est possible et où il est aussi possible, par un travail mathématique, de diminuer le nombre de configurations à traiter[51].

Nous pouvons donc fixer à un travail mathématique sur l'*algorithme de Kaprekar*, un enjeu comparable au travail des chercheurs sur le *Rubik's Cube*, après avoir pris conscience du coût d'une preuve par exhaustion, construire une preuve par réduction des cas, ce qui justifie la perspective de transposition que nous adoptons. Notons deux effets de cette transposition. D'une part, dans le cas du *Rubik's Cube* la preuve par exhaustion n'est pas matériellement

[51] Le problème de coloriage des cartes n'appartient pas à cette classe car le nombre de configurations est infini.

possible, alors qu'elle l'est dans le cas de Kaprekar, ce qui implique une dévolution particulière de la démarche de réduction des cas. D'autre part, les preuves portant sur des ensemble finis sont rares dans l'enseignement, et une preuve par exhaustion, bien que mathématiquement correcte, n'apporte pas d'éclairage particulier sur la proposition à prouver, ou, selon les termes de D. Grenier et C. Payan (1998) n'a pas « de fonction explicative », alors que cette fonction est généralement valorisée dans l'enseignement. La dévolution aux élèves d'une démarche de réduction des cas peut donc s'appuyer sur un besoin ressenti par ces élèves d'une preuve ayant une « fonction explicative » plus manifeste.

5.2.1.3. La continuité avec l'« ingénierie Guy »

Comme nous l'avons dit ci-dessus, les élèves au cours de l'« ingénierie-Guy » ont d'abord élaboré une conjecture par simple application papier-crayon du processus à quelques entiers. Lors de la transposition du travail fait au « papier-crayon » dans un environnement informatique, les élèves ont pris conscience de différentes représentations d'un nombre entier et de certaines propriétés qui en découlaient. Ils ont aussi pu travailler sur les différentes étapes qui constituaient la procédure de calcul : décomposition, tri et reconstruction.

Ainsi, avant notre ingénierie, les élèves ont déjà étudié la construction de l'algorithme correspondant à la « recette Kaprekar » et certaines propriétés sur les représentations des nombres entiers et ils peuvent donc concentrer leurs réflexions sur l'écriture d'une preuve de la conjecture. Deux possibilités d'organisation de la preuve vont pouvoir être mises en jeu lors de notre ingénierie.

Une première consiste à reprendre l'algorithme construit pendant l'« ingénierie Guy » et lui ajouter une structure « Pour » afin d'obtenir une vérification de la conjecture pour tous les nombres entiers compris entre 000 et 999. Cette preuve est une *preuve par exhaustion* et elle peut être considérée comme n'ayant pas de « fonction explicative ». La nécessité d'une preuve ayant une fonction plus explicative peut être ressentie par des élèves de Terminale Scientifique. En effet, dans le cas où les chiffres constituant le nombre entiers sont tous différents, les élèves de ce niveau peuvent partir de considérations faites pendant l'« ingénierie Guy », sur les résultats obtenus dès le premier calcul de la différence des deux nombres entiers. Ils ont observé qu'après la détermination de la différence le nouvel entier

obtenu avait pour chiffres des dizaines 9. Ce résultat peut les amener à se poser la question de la possibilité de diminuer le nombre de tests. Pour cela, les élèves vont articuler des *ETM* : algorithmique et arithmético-algébrique.

5.2.2 Les choix

Notre ingénierie comporte trois phases : **(1)** une adaptation de l'algorithme obtenu lors de l'« ingénierie-Guy » afin d'obtenir une preuve « par exhaustion » de la conjecture par une étude automatisée de vérification des 1 000 nombres entiers compris entre 000 et 999 ; **(2)** la justification de ce qu'il suffit de vérifier la conjecture sur six entiers ; **(3)** une adaptation de l'algorithme pour une exécution sur une liste d'entiers non consécutifs.

5.2.2.1. Le choix de garder la classe de l'« ingénierie Guy »

Nous choisissons de mener cette ingénierie dans la classe où l'« ingénierie Guy » avait déjà eu lieu. En effet, cette Terminale Scientifique connait ainsi la « recette de Kaprekar ». Les élèves ont énoncé une conjecture manuellement et ils ont construit un algorithme permettant de l'implémenter dans un environnement informatique, qu'ils ont testé sur quelques nombres entiers choisis de façon arbitraire. Ils ont aussi consolidé des compétences sur des propriétés dans **N**.

S'agissant d'une Terminale Scientifique, nous supposons que le contrat didactique en place à ce niveau scolaire permet de penser que les élèves ne vont pas se contenter de quelques tests, mais vont aussi chercher à déterminer une preuve de la conjecture. En effet, des élèves de ce niveau scolaire savent que pour prouver qu'une conjecture est vérifiée pour toutes les valeurs possibles (ici, des nombres entiers compris entre 000 et 999), il est nécessaire de procéder à une vérification pour toutes ces valeurs.

Ce choix permet aussi de compléter l'« ingénierie Guy » en présentant une ingénierie *locale*, découpée en trois séances, autour d'une tâche dans le domaine de la *théorie élémentaire des nombres* mettant en jeu des compétences en algorithmique (planification, preuve de terminaison) et en mathématiques (représentation des nombres, preuve d'un processus de calcul).

5.2.2.2. Le choix de l'ensemble des nombres

Guy (2013) a observé que pour des nombres entiers constitués de cinq ou six chiffres, on obtient plusieurs comportements différents (points stationnaires ou cycles) et que les cycles sont une difficulté pour l'élève de Terminale tant du point de vue de la conception, que du point de vue de leur repérage par un algorithme. Elle indique qu'elle a choisi de faire travailler les élèves sur des nombres à trois chiffres plutôt que quatre pour simplifier la procédure de tri.

Pour notre ingénierie, les nombres entiers à trois ou quatre chiffres nous semblent pertinents pour la même raison. Mais diminuer le nombre de cas pour les nombres à quatre chiffres est plus difficile.

En nous limitant aux nombres entiers s'écrivant $n_1 = \overline{abcd}$, avec $a \geq b \geq c \geq d$ tel que $a \neq d$, nous avons : $n_2 = \overline{dcba}$, avec $a \geq b \geq c \geq d$ et $a \neq d$.
D'où : $n_1 - n_2 = \overline{abcd} - \overline{dcba}$. Ecrivons ce nombre $\overline{ABCD}$.
Or « *a* ôté de *d* » est impossible, par conséquent, nous devons faire « *a* ôté de 10 + *d* », et alors, on écrit : $D = 10 + d - a$, puis on retient 1.
De plus, « *b* + 1 ôté de *c* » est impossible, car $b + 1 > c$, alors nous devons faire « *b* + 1 ôté de 10 + *c* », et on écrit : $C = 9 + c - b$, puis on retient 1. Quant à « *c* + 1 ôté de *b* » soit :

→ c'est possible (si $b \geq c + 1$ c.-à-d. si $b > c$), d'où : $B = b - c - 1$ et $A = a - d$ (nous faisons « *d* ôté de *a* », qui est toujours possible) ;
→ c'est impossible (si $b = c$), d'où : $D = 10 + d - a$, $C = 9$, « *b* + 1 ôté de *c* » est impossible, donc nous faisons « *b* + 1 ôté de 10 + *b* », et on écrit : $B = 9$ et on retient 1. De plus, « *d* + 1 ôté de *a* » est possible, alors on écrit $A = a - d - 1$.

Après récapitulation, on obtient :

- Si $b \neq c$, alors pour la différence des nombres entiers, on obtient :

$$A = a - d \ ; \ B = b - c - 1 \ ; \ C = 9 + c - b \ ; \ D = 10 + d - a$$

d'où : $A + D = 10$ et $B + C = 8$.

On a alors 25 nombres entiers qui conviennent, écrits avec leurs chiffres rangés du plus grand au plus petit : 9810 ; 9711 ; 9621 ; 9531 ; 9441 ; 8820 ; 8730 ; 8721 ; 8640 ; 8622 ; 8550 ; 8532 ; 8432 ; 7731 ; 7641 ; 7632 ; 7551 ; 7533 ; 7443 ; 6642 ; 6552 ; 6543 ; 6444 ; 5553 et 5544.

- Si $b = c$, alors pour la différence des nombres entiers, on obtient :

$$A + D = 9 \text{ et } B = C = 9.$$

On a alors 5 nombres entiers qui conviennent, écrits avec leurs chiffres rangés du plus grand au plus petit : 9990 ; 9981 ; 9972 ; 9963 ; 9954.

Pour conclure, il suffit faire fonctionner l'*algorithme de Kaprekar* pour chacun de ces 25 + 5 = 30 nombres entiers. On peut vérifier que, en au plus 7 itérations on arrive toujours à 6174.

Cette étude sur l'ensemble des nombres à quatre chiffres implique donc plusieurs étapes et une gestion de différents cas et nous parait ainsi délicate à traiter avec les élèves dans une séance de quatre-vingt-dix minutes d'autant plus que ceux-ci ont programmé le tri sur trois chiffres seulement.

Avec les nombres de 000 à 999 comme nous le verrons plus loin, on peut démontrer en une étape que l'ensemble des résultats après la première itération n'a que cinq éléments (six, si on compte les nombres entiers de la forme $\overline{aaa}$, avec a entier compris entre 0 et 9), ce qui répond bien à l'objectif de réduire le coût de la preuve faite avec l'algorithme. Donc, nous choisissons, comme pour l'« ingénierie-Guy », les nombres entiers à trois chiffres. L'extension à quatre chiffres, qui correspond à l'étude initiale de Kaprekar[52] reste cependant, une tâche tout à fait accessible aux élèves de Terminale Scientifique.

5.2.2.3. Le choix de l'environnement informatique

Nous choisissons de garder l'environnement de programmation utilisé pour l'« ingénierie-Guy ». En effet, comme le signale Guy (2013), un texte écrit en *AlgoBox* peut être vu soit comme un algorithme si on s'intéresse à sa logique et à ses performances, soit comme un programme si on s'intéresse à son implémentation. De plus, ce logiciel permet d'utiliser des listes numérotées de nombres et de procéder à des calculs dont les résultats seront du type NOMBRE. Le choix de ce logiciel pour notre ingénierie permet aussi aux élèves de repartir de l'algorithme implémenté lors de l'ingénierie précédente et les binômes peuvent plus facilement procéder aux modifications et aux améliorations possibles de l'algorithme obtenu lors de l'« ingénierie Guy ».

[52] https://fr.wikipedia.org/wiki/Algorithme_de_Kaprekar (Consulté le 18 octobre 2017)

5.2.3 Les objectifs

Le premier objectif de cette ingénierie est d'amener les élèves à adapter l'algorithme obtenu lors de l'« ingénierie-Guy » en utilisant une structure « Pour » afin que l'implémentation de cet algorithme « complet » dans un l'environnement *AlgoBox* permette d'obtenir une preuve *par exhaustion* de la conjecture.

De plus, comme l'« ingénierie-Guy » a permis à l'élève de consolider des compétences sur des propriétés concernant les nombres entiers, un deuxième objectif est d'amener les élèves à exploiter ces compétences. En effet, ces acquis ou ces consolidations sur des compétences concernant le système décimal et la divisibilité dans **N**, permettront aux élèves de se situer dans un ETM. Nous avons ainsi comme objectif d'installer un ETM favorable à la construction d'une preuve par réduction des cas.

Ainsi, nous avons comme objectif que les élèves puissent utiliser la décomposition d'un nombre entier N = $\overline{abc}$ sous la forme N = $100\,a + 10\,b + c$ afin de construire une preuve justifiant que la vérification puisse se faire sur un nombre limité d'entiers.

5.2.4 Les attentes

Lors de la première phase, nous attendons que les élèves utilisent l'algorithme obtenu à la fin de l'« ingénierie-Guy » et procèdent à des améliorations de cet algorithme en l'incluant dans une structure « Pour » afin de mettre en place une preuve par exhaustion. Puis, en seconde phase, tenant compte du contrat didactique en classe de Terminale Scientifique, nous attendons que les élèves proposent une preuve se situant dans un ETM, afin de limiter le nombre de cas à vérifier. L'ensemble des nombres entiers à vérifier qu'ils auront déterminés par des raisonnements mathématiques ne suit pas des règles simples, comme des nombres consécutifs par exemple. Nous nous attendons donc à ce que les élèves mettent en place en troisième phase une stratégie utilisant le type LISTE du logiciel *AlgoBox* afin de parcourir cet ensemble.

5.2.5 Détermination de l'ensemble des nombres entiers à vérifier

Pour faciliter la lecture de ce qui va suivre, et sachant que cela ne va rien enlever à la pertinence de la preuve proposée, nous pouvons supposer que l'entier N = $\overline{abc}$ vérifie les inégalités suivantes $9 \geq a \geq b \geq c \geq 0$.

Soit alors G le nombre entier obtenu lors de l'étape (2) de la « recette de Kaprekar », c'est-à-dire l'entier G = $\overline{abc} = 100\ a + 10\ b + c$, et soit P le nombre entier obtenu lors de l'étape (3), c'est-à-dire P = $\overline{cba} = 100\ c + 10\ b + a$.

Alors, nous obtenons à l'étape (4), le nombre entier D tel que :

$$D = G - P = 100\ a + 10\ b + c - (100\ c + 10\ b + a) = 99\ a - 99\ c = 99\ (a - c).$$

Par conséquent, le nombre D est un multiple de 99.

Or, le nombre ($a - c$) est un entier compris entre 0 et 9, car $9 \geq a \geq c \geq 0$.

Par conséquent, il ne reste plus qu'à vérifier les dix nombres entiers suivants : 000, 099, 198, 297, 396, 495, 594, 693, 792 et 891.

Cependant, nous pouvons observer que les nombres 198 et 891, les nombres 297 et 792, les nombres 396 et 693, et les nombres 495 et 594 sont respectivement constitués des mêmes chiffres, et que par conséquent il suffit de vérifier la conjecture pour les nombres 000, 099, 198, 297, 396 et 495 afin de valider celle-ci.

Nous pouvons observer que ce raisonnement utilise le calcul algébrique et des propriétés de divisibilité. Il se situe donc dans un ETM.

5.2.6 Les tâches pour chacune des phases de notre ingénierie et le mode de travail

Comme nous l'avons vu plus haut, notre ingénierie présente trois phases : **(1)** une adaptation de l'algorithme obtenu lors de l'« ingénierie-Guy » afin d'obtenir une preuve par exhaustion des cas de la conjecture par une étude automatisée de vérification des 1 000 nombres entiers compris entre 000 et 999 ; **(2)** la justification qu'il suffit de vérifier la conjecture seulement sur un ensemble de six nombres entiers ; **(3)** une adaptation de l'algorithme pour une exécution sur cet ensemble d'entiers.

Il s'agit d'amener les élèves à construire deux preuves de cette conjecture. En effet, nous attendons que les élèves proposent dans un premier temps une preuve par exhaustion des cas. Cette preuve nécessite que les élèves modifient l'algorithme de l'« ingénierie-Guy » introduisant une boucle « Pour » qui va permettre que l'environnement informatique procède à une vérification de la conjecture pour tous les nombres entiers compris entre 000 et 999,

soit 1000 cas à vérifier. Puis, dans un second temps, nous pensons que les élèves de ce niveau scolaire (Terminale Scientifique, enseignement de spécialité) vont se pencher sur la nécessité ou non de faire que l'environnent informatique procède à une vérification sur cet ensemble de nombres entiers. En effet, en amont de cette phase de notre ingénierie, les élèves ont eu la possibilité de lire un article sur le *Rubik's Cube* (cf. Annexe 2). A la lecture de cet article, les élèves ont pu constater que depuis 1995, on savait que 20 était une limite inférieur du nombre de mouvements requis dans le pire des cas pour résoudre la configuration du *Rubik's Cube*. C'est-à-dire qu'*il y a des positions* qui ne peuvent être résolues en moins de 20 coups. Mais, les scientifiques qui s'intéressaient à ce problème de résolution se sont interrogés si toutes les solutions pouvaient être résolues en 20 rotations ? Il faut savoir que le nombre de positions de départ est de 43 252 003 274 489 856 000.

Moves required to solve Rubik's Cube	Number of positions
0	1
1	18
2	243
3	3,240
4	43,239
5	574,908
6	7,618,438
7	100,803,036
8	1,332,343,288
9	17,596,479,795
10	232,248,063,316
11	3,063,288,809,012
12	40,374,425,656,24
13	531,653,418,284,628
14	6,989,320,578,825,358
15	91,365,146,187,124,313
16	about 1,100,000,000,000,000,000
17	about 12,000,000,000,000,000,000
18	about 29,000,000,000,000,000,000
19	about 1,500,000,000,000,000,000
20	about 300,000,000

Figure 30[53] *(Nombre de nœuds du graphe du Rubik's Cube ayant une distance donnée à la configuration initiale)*

Les élèves prennent alors conscience que si l'on pouvait passer en revue un milliard de combinaison par seconde, il faudrait plus de 1 200 ans pour les visiter toutes. Ou encore, si

[53] Source de l'image : Daily Mail

nous disposions de cubes de taille normale associés à toutes les configurations possibles, nous pourrions alors recouvrir 275 fois la Terre. Ainsi, les élèves reconnaissent que ce nombre est énorme et que par conséquent la résolution du problème de résoudre le *Rubik's Cube* n'est pas triviale. Nous espérons ainsi que les élèves vont s'interroger sur la pertinence d'une résolution qui demanderait moins de cas nécessaire pour résoudre ce problème en un minimum de coups. De ce fait, l'article proposé aux élèves, leur apprend qu'en 2010, une équipe composée de Tomas Rokicki (programmeur de Palo Alto), Herbert Kociemba (Professeur de Mathématiques de Darmstadt), Morlay Davidson (Mathématicien de Kent State University) et de John Dethridge (Ingénieur chez Google) a pu prouver que les 43 252 003 274 489 856 000 positions de départ peuvent être réparties en 2 217 093 120 ensembles de 19 508 428 800 configurations chacune. Le nombre d'ensemble est ainsi réduit à 55 882 296 configurations en utilisant des symétries et ensembles couvrants. Ils ont pu tester ces configurations, et prouver que chacune des configurations initiales pouvaient se résoudre en 20 coups ou moins.

Cette lecture, toute proportion gardée, doit inciter les élèves à retravailler le problème de Kaprekar en se fixant comme objectif de déterminer dans un premier temps un ensemble limité d'entiers qui seraient suffisants pour valider la conjecture, avant de passer à l'étape consistant à vérifier à l'aide de l'environnement informatique cette conjecture sur cet ensemble limité.

Pour cela, les élèves doivent travailler d'abord dans le domaine de la *théorie élémentaire des nombres* pour prouver que seul six nombres entiers doivent faire l'objet d'une vérification à l'aide de l'algorithme de l'« ingénierie-Guy ». Notre ingénierie ayant pour objectif aussi d'automatiser dans un environnement informatique cette vérification, nous attendons que les élèves adaptent alors l'algorithme de l'« ingénierie-Guy » pour procéder à cette vérification automatisée sur les six nombres entiers obtenus. Cette adaptation va nécessiter de la part des élèves d'utiliser une variable de type LISTE pour représenter l'ensemble des nombres qui devront être réellement testés par l'exécution de l'algorithme.

5.2.6.1. Phase 1 : Adapter l'algorithme obtenu lors de l'« ingénierie Guy » pour une preuve par exhaustion des cas

Les élèves doivent utiliser l'algorithme qu'ils ont obtenu lors de l'ingénierie précédente En le modifiant éventuellement afin de vérifier la conjecture sur tous les entiers de 0 à 999.

5.2.6.2. Phase 2 : Justifier qu'il suffit de vérifier la conjecture que sur six nombres entiers

Pour cette seconde phase, il est prévu qu'elle débute par un échange entre l'enseignant et l'ensemble de la classe. Il est demandé aux élèves de lire entre les deux ingénieries un article sur le *Rubik's Cube* (cf. Annexe n° 2), distribué à la fin de la séance précédant cette nouvelle ingénierie. A la lecture de l'article, les élèves doivent prendre conscience du fait qu'une preuve a été proposée justifiant qu'un nombre maximal de mouvements de coups pour résoudre la problématique de reconstruction du *Rubik's Cube* a été déterminée et ceci quelle que soit la configuration initiale du *Rubik's Cube*. Les élèves doivent alors interpréter ce nombre maximal de mouvements, comme une réduction des rotations à faire pour résoudre le problème de reconstruction, ce qui doit les conduire, dans notre cas, vers l'élaboration d'une preuve « par réduction des cas » de la conjecture en remplacement de la preuve par exhaustion des cas obtenue lors la phase précédente. Les élèves doivent interpréter l'article sur le *Rubik's Cube* comme justification que le nombre de vérifications qui est à faire lors d'une preuve par exhaustion des cas ne peut avoir de sens que si ce nombre de cas à vérifier reste raisonnable d'un point temporel. Ainsi, au cours de cette phase, les élèves doivent par un raisonnement arithmético-algébrique, démontrer que la preuve obtenue lors de la phase précédente peut être améliorée (au sens de gain en temps) et rendre ainsi l'algorithme plus efficace. Afin de construire une preuve « par réduction des cas » de la conjecture, les élèves doivent aussi réutiliser certains résultats de l'« ingénierie-Guy », comme le fait que tout entier de la forme N = $\overline{abc}$, où *a*, *b* et *c* appartiennent à l'ensemble {0 ; 1 ; 2 ; … ; 9}, puisse s'écrire sous la forme N = 100 *a* + 10 *b* + *c*. Ainsi, au cours de cette phase, les élèves doivent aboutir à l'aide de raisonnements arithmético-algébrique à une égalité de la forme 99 (*a* – *c*), avec *a* ≥ *c*. Puis, une fois que celle-ci est justifiée, les élèves doivent démontrer qu'il suffit de prendre en considération les seuls nombres 000, 099, 198, 297, 396 et 495 pour une vérification à l'aide de l'algorithme. Les élèves peuvent considérer cette nouvelle preuve comme étant plus

« explicative », car elle s'appuie sur des raisonnements demandant un formalisme mathématique.

5.2.6.3. Phase 3 : Modifier l'algorithme pour une exécution sur une liste d'entiers non consécutifs.

Au cours de cette dernière phase les élèves doivent reprendre l'algorithme de la première phase et le modifier en concevant le parcours d'un ensemble de nombres d'entiers non consécutifs. Ensuite, les élèves doivent exécuter l'algorithme afin de justifier la conjecture.

5.2.7 Les analyses *a priori*

Pour aider le lecteur à la compréhension des analyses qui vont suivre, nous rappelons la recette de Kaprekar qui avait été donnée au début de l'« ingénierie-Guy ».

(1) Choisir un nombre entier naturel de trois chiffres (c.-à-d. compris entre 000 et 999).
(2) Former le nombre entier obtenu en rangeant les chiffres du nombre choisi en (1) dans l'ordre croissant.
(3) Former le nombre entier obtenu en rangeant les chiffres du nombre choisi en (1) dans l'ordre décroissant.
(4) Calculer la différence des nombres entiers obtenus en (2) et (3).
(5) Recommencer le processus à partir de l'instruction (2), avec le résultat obtenu en (3). Etc.

5.2.7.1. Phase 1 : Adapter l'algorithme obtenu lors de l'« ingénierie Guy » afin de déterminer une preuve par exhaustion des cas (Durée prévue : 20 minutes)

Lors de cette première phase, nous prévoyons qu'après ouverture du fichier correspondant à l'algorithme qu'ils ont obtenu lors de l'« ingénierie-Guy », ils vont chercher à vérifier la conjecture pour tous les nombres entiers de 000 à 999. Pour comprendre le travail fait en amont (lors de l'« ingénierie-Guy ») et étudier l'importance des modifications qui sont nécessaires lors de cette première phase pour que l'algorithme initial puisse permettre une vérification « automatisée » de la conjecture pour les 1 000 nombres entiers à vérifier, nous rappelons les différents algorithmes obtenus par les élèves lors de l'« ingénierie-Guy ». Puis, nous analysons le comportement attendu des élèves selon l'algorithme obtenu.

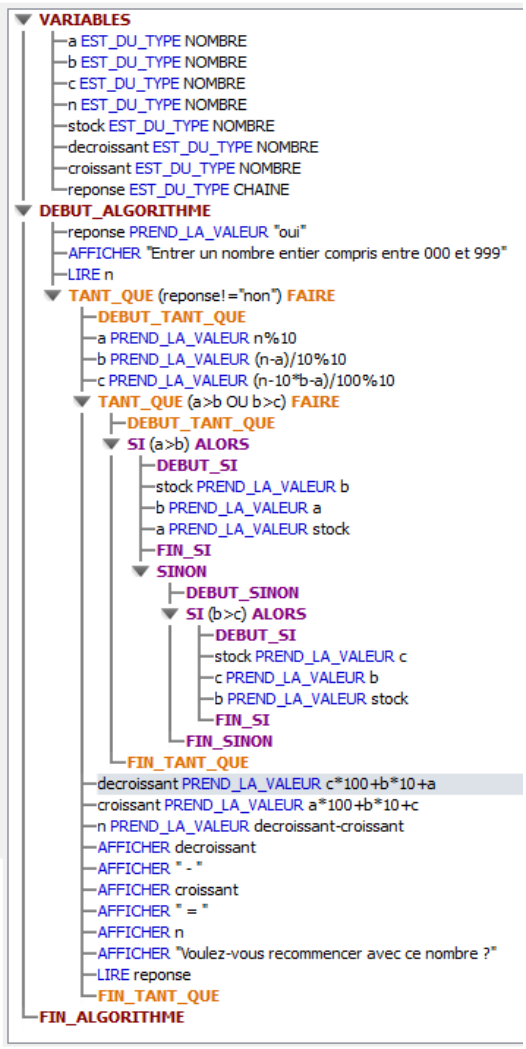
a) Un rappel sur les algorithmes obtenus à la fin de l'« ingénierie-Guy »

A la fin de l'« ingénierie-Guy », les élèves ont rendu des algorithmes du type (a), (b), (c) et (d) (fig. 31, extraits de travaux d'élèves). Les algorithmes (a), (b) et (c) donnés par trois binômes sont représentatifs de 30 % des algorithmes obtenus à la fin de l'« ingénierie Guy ».

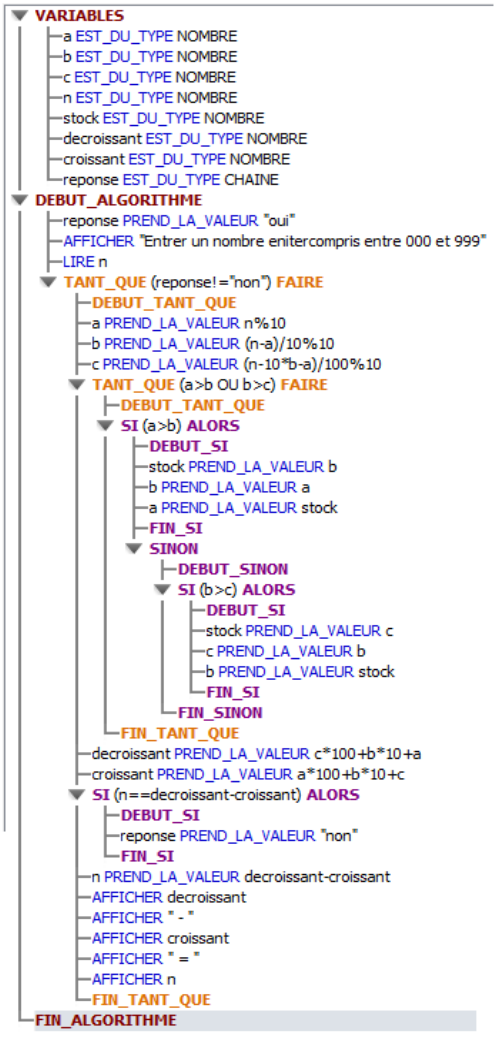
Dans les algorithmes (a) et (b), nous observons que les élèves utilisent deux types de variables : NOMBRE et CHAINE. Les élèves mettent en place une reconnaissance des trois chiffres constituants le nombre entier N = $\overline{cdu}$ afin de déterminer la décomposition du nombre N en chiffres pour répondre aux étapes (2) et (3) de la « recette de Kaprekar » permettant de calculer la différence de l'étape (4). Pour cela, les élèves utilisent les fonctions que propose l'environnement informatique utilisé. Par exemple, cette reconnaissance des chiffres se fait à l'aide de la fonction « reste de la division euclidienne » du nombre N par 10 pour obtenir le chiffre des unités qu'ils stockent dans la variable *a*, puis après la détermination du nombre des dizaines en faisant le calcul (N – a)/10, l'algorithme détermine le chiffre des dizaines du nombre N en réutilisant la fonction « reste de la division euclidienne » de l'environnement informatique sur ce nombre des dizaines, et pour terminer cette décomposition du nombre N réutilise à nouveau ce processus de calcul afin de d'obtenir le chiffre des centaines. Cependant, les élèves qui utilisent ce procédé ne donnent aucune indication explicite quant à la nature de ces trois chiffres. Ainsi, le lecteur doit reconnaitre dans les calculs proposés que les chiffres obtenus correspondent aux chiffres des unités, des dizaines et des centaines du nombre N. S'ensuit alors l'étape de tri de ces trois chiffres obtenus afin de répondre aux étapes de reconstruction de deux nombres entiers demandés aux étapes (2) et (3) de la recette. Ceci se termine par le calcul de la différence demandée à l'étape (4). Ensuite, nous pouvons observer que les algorithmes (a) et (b) différent par le fait que le premier ne propose pas une automatisation complète de la recette contrairement au second. En effet, dans le cas de l'algorithme (a), les élèves introduisent à nouveau un échange entre l'utilisateur et l'environnement informatique en faisant que le processus de calcul de la recette ne peut recommencer avec le nouveau nombre obtenu à l'étape (4) que si l'utilisateur le confirme, et ainsi l'étape (5) du processus se fait comme si l'élève transpose une action de type « copier-coller » pratiquée lors de travaux sur tableur. En revanche, dans le cas de l'algorithme (b), cet échange n'existe pas, suite à une automatisation de l'algorithme, et par conséquent l'étape (5) se fait de façon autonome par l'environnement informatique.

Pour l'algorithme (c), les élèves utilisent trois types de variables : NOMBRE, CHAINE et LISTE. Le processus de décomposition du nombre entier N = $\overline{cdu}$ permettant d'obtenir les chiffres des unités, des dizaines et des centaines du nombre est identique à celui mis en place dans les deux algorithmes précédents. En revanche, dans ce cas, les élèves stockent ces trois

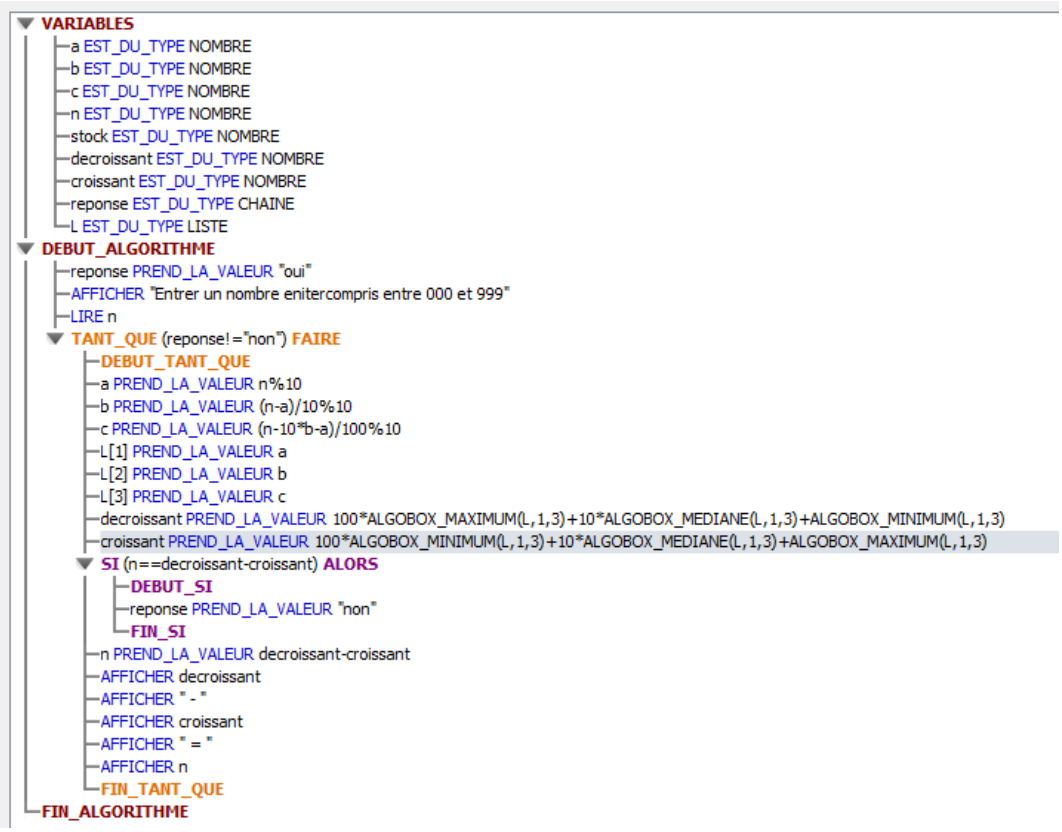
chiffres dans une variable de type LISTE. Comme pour les algorithmes décrient précédemment, la signification des trois chiffres obtenus après décomposition du nombre N puis stockés dans une liste, est donnée de façon implicite. En effet, les auteurs de ce type d'algorithme ne donnent aucune indication concernant la nature des trois chiffres obtenus : chiffre des unités, chiffre des dizaines, chiffre des unités. Ensuite, à l'aide de fonctions disponibles dans l'environnement informatique utilisé, les élèves déterminent les nombres demandés aux étapes (2) et (3). Ces fonctions utilisées renvoient les élèves à des connaissances issues du domaine des statistiques descriptives, où les élèves ont l'habitude de déterminer les valeurs maximale et minimale d'une série statistique ainsi que le calcul d'une médiane d'une telle série. Comme pour l'algorithme (b), ce dernier est aussi entièrement « autonome » dans le fait que pour un entier donné par l'utilisateur le processus de calcul de la recette est complètement automatisé. En effet, l'étape (5) est faite par l'environnement informatique et non par l'utilisateur.



Algorithme (a)

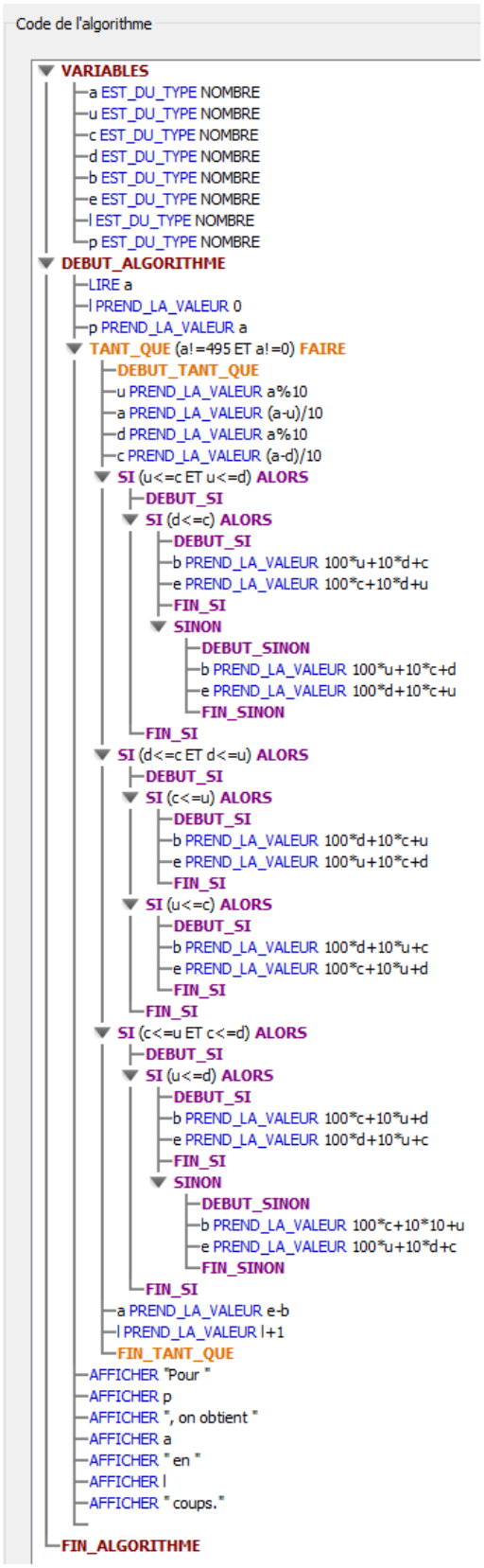


Algorithme (b)



```
VARIABLES
    a EST_DU_TYPE NOMBRE
    b EST_DU_TYPE NOMBRE
    c EST_DU_TYPE NOMBRE
    n EST_DU_TYPE NOMBRE
    stock EST_DU_TYPE NOMBRE
    decroissant EST_DU_TYPE NOMBRE
    croissant EST_DU_TYPE NOMBRE
    reponse EST_DU_TYPE CHAINE
    L EST_DU_TYPE LISTE
DEBUT_ALGORITHME
    reponse PREND_LA_VALEUR "oui"
    AFFICHER "Entrer un nombre enitercompris entre 000 et 999"
    LIRE n
    TANT_QUE (reponse!="non") FAIRE
        DEBUT_TANT_QUE
        a PREND_LA_VALEUR n%10
        b PREND_LA_VALEUR (n-a)/10%10
        c PREND_LA_VALEUR (n-10*b-a)/100%10
        L[1] PREND_LA_VALEUR a
        L[2] PREND_LA_VALEUR b
        L[3] PREND_LA_VALEUR c
        decroissant PREND_LA_VALEUR 100*ALGOBOX_MAXIMUM(L,1,3)+10*ALGOBOX_MEDIANE(L,1,3)+ALGOBOX_MINIMUM(L,1,3)
        croissant PREND_LA_VALEUR 100*ALGOBOX_MINIMUM(L,1,3)+10*ALGOBOX_MEDIANE(L,1,3)+ALGOBOX_MAXIMUM(L,1,3)
        SI (n==decroissant-croissant) ALORS
            DEBUT_SI
            reponse PREND_LA_VALEUR "non"
            FIN_SI
        n PREND_LA_VALEUR decroissant-croissant
        AFFICHER decroissant
        AFFICHER " - "
        AFFICHER croissant
        AFFICHER " = "
        AFFICHER n
        FIN_TANT_QUE
FIN_ALGORITHME
```

Algorithme (c)



Algorithme (d)

Figures 31 (Exemples d'algorithmes obtenus à la fin de l' « ingénierie-Guy » sur Kaprekar)

De plus, l'automation des algorithmes (b) et (c), se fait à l'aide d'un test de sortie de la boucle « TantQue » basée sur une égalité entre deux nombres : celui connu avant l'étape (2)

et celui obtenu à la fin de l'étape (4). En effet, tant que cette égalité n'est pas vérifiée le processus complet de la recette est reconduit. L'interprétation par les élèves de la conjecture dans ce cas est faite sur l'égalité entre deux nombres : celui du départ avant l'entrée dans le processus et celui de l'arrivée une fois que le processus de calcul des étapes (2) à (4) est fait.

Les autres algorithmes proposés par les élèves (soit 70 % des binômes) mis en place utilisent un test de sortie basé sur une égalité à 0 ou 495 (fig. 7d), suivant le nombre entier initialement choisi. Ce test renvoie au travail fait lors de l'ingénierie précédente sur la détermination de la conjecture. Contrairement aux algorithmes précédents, nous observons ici que les élèves sont bien dans l'esprit de justifier une conjecture.

Pour la construction de l'algorithme, ils n'utilisent qu'un seul type de variable : NOMBRE. La décomposition du nombre entier est identique à celle mise en place dans les algorithmes précédents. Cependant, nous observons que les élèves utilisent des variables qui permettent de mieux associer les résultats obtenus en tant que chiffres des unités, chiffres des dizaines et chiffres des centaines. De plus, le tri des trois chiffres se fait à l'aide d'une succession de « Si …Alors » contrairement aux algorithmes (a) et (b) qui ont fait appel à deux types de structures une « TantQue » contenant une structure « Si …Alors ». De plus, la reconstruction des nombres entiers demandés aux étapes (2) et (3) se fait comme pour les algorithmes précédents, en utilisant des résultats revus lors de l'« ingénierie-Guy » sur la structure d'un nombre dans le système décimal : $\overline{abc} = 100\,a + 10\,b + c$. Pour finir sur l'analyse de ce dernier algorithme, les élèves proposent d'introduire une variable de type NOMBRE et notée *p* qui compte le nombre de coups nécessaires afin d'obtenir le résultat de la conjecture. Ceci est le fait d'une prise d'initiative des élèves. En effet, cela n'est pas questionné dans l'« ingénierie-Guy » ainsi que dans la nôtre. Toutefois, vu que 70 % des binômes proposent un tel algorithme à la fin de l'« ingénierie-Guy », nous pouvons supposer que les élèves prennent cette initiative par habitude ou questionnement sur une problématique de type « temps de parcours ».

b) Comportement attendu selon l'algorithme obtenu dans l'« ingénierie Guy ».

Comme nous l'avons vu dans le paragraphe précédent, les élèves ont terminé l'ingénierie précédente avec quatre types d'algorithmes. Ainsi, suivant l'algorithme obtenu par le binôme lors l'« ingénierie-Guy », ce dernier doit apporter plus ou moins de modifications afin de répondre à la consigne.

- **Elèves ayant obtenu L'algorithme (a)**

Dans le cas de l'algorithme (a), nous nous attendons à ce que les binômes concernés procèdent à des modifications de l'algorithme afin que ce dernier puisse répondre à la consigne de manière automatisée. En effet, la lecture de cet algorithme montre qu'il est nécessaire qu'à chaque procédure complète de l'algorithme, l'utilisateur indique s'il continue ou non le processus avec le nombre obtenu. Par conséquent, nous nous attendons à ce que les binômes ayant mis en place cet algorithme, procèdent à plusieurs adaptations. Ainsi, ils doivent supprimer toute interaction entre l'utilisateur et la machine. En effet, pour notre problématique de vérification de la conjecture pour tous les entiers compris entre 0 et 999, ces binômes doivent supprimer la ligne « Lire *n* » dans le sens que maintenant l'algorithme ne doit plus travailler sur un entier choisi de manière arbitraire par l'utilisateur. Ils doivent aussi mettre en place une structure de type « TantQue…Faire » pour que la résolution de la « recette de Kaprekar » soit totalement automatisée. Puis, la vérification de la conjecture ne devant plus être faite sur un nombre entier arbitraire compris en 0 et 999, les binômes doivent alors utiliser une structure de type « Pour *k* = 0 à 999 » afin que l'algorithme implémenté dans l'environnement informatique procède à une vérification automatisée de la conjecture donnant ainsi une preuve par exhaustion des cas. Nous supposons que le travail fait au cours de l'ingénierie précédente va permettre à ces binômes de ne pas éprouver de difficultés particulières pour mettre en place ces modifications. En revanche, nous pensons que certains binômes peuvent oublier que toute variable introduite dans le traitement de l'algorithme nécessite d'être déclarée. En effet, l'environnement informatique choisi demande une telle déclaration. Cela peut poser des difficultés aux élèves. En effet, lors de la conception de l'algorithme au « papier-crayon », ceci ce fait de façon implicite, et de plus, malgré le travail fait lors de l'« ingénierie-Guy », cette étape de déclarer les variables n'est pas nécessairement automatisée dans l'esprit d'élèves débutant avec l'environnement informatique utilisé. Cependant, nous nous attendons à ce que lors de l'introduction des nouvelles procédures, les binômes concernées réagissent à cet oubli possible. En effet, l'environnement informatique utilisé ne permet pas de créer une nouvelle structure associée à une variable informatique si celle-ci n'a pas été explicitement déclarée.

- **Elèves ayant obtenu L'algorithme (b)**

Dans le cas de l'algorithme (b), nous nous attendons à ce que les binômes concernés éprouvent moins de difficultés que ceux qui ont proposé un algorithme du type (a). En effet, pour ces binômes l'algorithme du départ est déjà automatisé pour un nombre entier arbitraire choisi par l'utilisateur. Nous supposons alors que ces binômes vont procéder soit à une vérification fastidieuse sur tous les nombres entiers compris entre 000 et 999 sans apporter de modifications à l'algorithme, soit à la création d'une boucle permettant d'automatiser aussi la vérification de la conjecture pour tous les nombres entiers compris entre 000 et 999. Pour cela, nous nous attendons à ce qu'ils suppriment dans un premier temps la ligne « Lire *n* », correspondant au nombre entier arbitraire choisi par l'utilisateur pour la remplacer par une variable informatique permettant d'introduire une structure du type « Pour *k* = 0 à 999 ». Nous attendons alors à ce que les élèves prennent conscience que cette introduction n'entraîne pas une refonte complète de l'algorithme mais seulement l'introduction de la boucle existante du type « TantQue » dans la boucle « Pour ». Tenant compte du travail fait en amont lors de l'« ingénierie-Guy », nous ne nous attendons pas à ce que les élèves concernés éprouvent des difficultés particulières pour procéder à ces modifications.

- **Elèves ayant obtenu L'algorithme (c)**

En ce qui concerne l'algorithme (c), nous nous attendons à ce que les binômes concernés agissent de façon analogue à ceux de l'algorithme (b). En effet, pour l'algorithme (c) obtenu lors de l'ingénierie précédente, seule la technique mise en place pour déterminer le tri des chiffres constituant le nombre entier de trois chiffres diffère. Or ceci ne nécessite pas une modification dans le cadre des attentes de cette phase. Pour le reste, c'est-à-dire l'automatisation de l'algorithme permettant de vérifier la conjecture pour tout nombre entier compris entre 000 et 999 sans qu'il soit choisi de manière arbitraire par l'utilisateur, nous supposons qu'aucune difficulté nouvelle par rapport à ce qui a été décrit pour l'algorithme (b) ne va être à relever. Ainsi, nous nous attendons à ce que ces élèves introduisent une structure du type « Pour *k* = 0 à 999 » pour procéder à une vérification automatisée de la conjecture sur l'ensemble des entiers compris entre 0 et 999. Cependant, nous pouvons aussi penser que certains vont soit essayer de tester l'algorithme obtenu lors de l'ingénierie précédente sans le modifier pour chaque entier compris entre 0 et 999, soit essayer de créer de nouveau une

variable de type LISTE permettant d'afficher tous les résultats obtenus pour les 1 000 nombres entiers à vérifier.

- **Elèves ayant obtenu L'algorithme (d)**

Dans le cas de l'algorithme (d), où les élèves ont construit un test de sortie sur une égalité à 0 ou à 495 traduisant le résultat de la conjecture définie au « papier-crayon », nous pouvons supposer que les difficultés que vont connaitre les élèves, vont être semblables à celles abordées dans le cas des algorithmes automatisés (b) et (c).

Cependant, nous pouvons supposer que le travail fait lors de l'« ingénierie-Guy » va permettre de faire que les tâches nécessaires sur le plan algorithmique pour automatiser la vérification de la conjecture afin d'obtenir une preuve par exhaustion des cas ne va pas poser des difficultés majeures pour une grande majorité des binômes concernés par cet algorithme (d).

D'une manière générale, bien qu'il soit prévu une vingtaine de minutes pour cette phase, nous pouvons penser que certains binômes vont rester moins de temps sur cette phase.

5.2.7.2. Phase 2 : Justifier qu'il suffit de vérifier la conjecture que sur six nombres entiers (Durée prévue : 30 minutes)

Bien que la preuve par exhaustion des cas de la conjecture ne puisse pas être vue par les élèves comme pouvant poser problème d'un point de vue de l'exécution qui est en effet, obtenue dans un temps raisonnable dans le cas de Kaprekar, nous souhaitons que les élèves prennent conscience du problème à un niveau plus général, de ce que ce temps d'exécution est à prendre en compte car il peut se manifester de façon cruciale pour d'autres traitements. Nous nous attendons ainsi à ce que la lecture de l'article sur le *Rubik's Cube* puisse contribuer à cette prise de conscience chez les élèves de ce niveau scolaire. De plus, les élèves peuvent se référer à des compétences en numération décimale, et plus particulièrement autour de la décomposition d'un nombre entier comme combinaison linéaire de puissances de dix, et mettre en pratique cette décomposition afin d'obtenir une preuve « plus efficace » de la conjecture. Nous nous attendons à ce que les binômes soient encouragés à aller vers la construction d'une telle preuve, à la fois par la lecture de l'article qui leur montre qu'il y a un nombre maximal de cas à considérer pour résoudre la problématique de la configuration du

Rubik's cube et par le travail fait sur un certain nombre de propriétés sur les nombres entiers lors de l'« ingénierie-Guy ». En effet, sur ce dernier point, nous nous attendons à ce que les élèves observent certaines propriétés lors de l'expérimentation de la « recette de Kaprekar » à l'aide du « papier-crayon ». Par exemple, ils peuvent observer qu'après que les étapes de calculs aient été faites complétement pour un nombre entier quelconque compris 000 et 999, la différence des deux nombres obtenus aux étapes (2) et (3) de la recette donne toujours un nombre dont le chiffre des dizaines est 9, dans le cas où au moins un des chiffres du nombre du départ est différent des deux autres.

Nous pensons aussi que les élèves ne vont pas éprouver de difficulté à trouver et justifier l'égalité 99 ($a - c$) avec le nombre N = $\overline{abc}$, où $0 \leq c \leq b \leq a \leq 9$ et a, b et c entiers lors du calcul de la première différence après que la procédure complète de l'algorithme ait lieu.

Les élèves devant ensuite déterminer les différentes valeurs possibles de (a – c), nous pensons que pour cela ils vont pouvoir se référer à des compétences en lien avec le concept de divisibilité acquises dans le cadre de l'enseignement de l'arithmétique tel qu'il est pratiqué dans l'enseignement de spécialité des Terminales Scientifiques.

Ensuite, nous supposons que les binômes vont proposer une liste de nombres entiers multiples de 99 compris entre 0×99 et 9×99, c'est-à-dire entre 0 et 891. Il est possible que certains élèves n'aient pas conscience que les10 nombres ainsi obtenus peuvent être réduit à 6 nombres. En effet, nous pensons que ces élèves vont procéder à une vérification de la conjecture pour ces dix nombres et considérer que cela peut suffire pour optimiser le nombre de vérifications à faire à l'aide de l'environnement informatique. Cependant, nous pensons qu'il est probable que la lecture de l'article sur le *Rubik's cube* qui montre les différentes étapes « historiques » qui ont conduit à des améliorations dans la détermination du nombre de rotations nécessaires pour optimiser la reconstruction de la configuration ordonnée et complète du cube, et ceci quelle que soit sa configuration initiale, puisse favoriser un nouveau questionnement sur la détermination de l'ensemble des entiers obtenus après un premier travail sur 99 ($a - c$). En effet, cet ensemble est constitué des nombres 0, 99, 198, 297, 396, 495, 594, 693, 792 et 891. Nous pensons alors que ces élèves vont observer le fait que les nombres 198, 297, 396, 495, 594, 693, 792 et 891 peuvent être regroupés par deux dès les étapes (2) et (3), et que par conséquent, il ne suffit de procéder à la vérification que pour 198,

297, 396 et 495. Pour conclure, nous nous attendons à ce que les élèves concluent que l'algorithme doit être exploité pour une vérification concernant seulement six nombres entiers : 000, 099, 198, 297, 396 et 495.

5.2.7.3. Phase 3 : Modifier l'algorithme obtenu lors de l'« ingénierie Guy » (Durée prévue : 40 minutes)

Pour cette dernière phase, nous pensons que certains élèves vont vouloir vérifier la « recette de Kaprekar » pour les nombres entiers 0, 99, 198, 297, 396, 495, 594, 693, 792 et 891, ou plus simplement pour les nombres 0, 99, 198, 297, 396 et 495 sans procéder à une modification de l'algorithme obtenu lors de l'ingénierie précédente. En effet, comme ce nombre de valeurs est très restreint, nous pouvons supposer que certains élèves ne vont pas avoir conscience d'une utilité de modifier l'algorithme initial. Même parmi ceux-ci, certains peuvent se contenter de proposer une vérification au « papier-crayon ». En effet, le travail fait lors de la phase 2 étant fait au « papier-crayon », il est possible que certains élèves restent dans le cadre d'un travail « papier-crayon » pour justifier la conjecture, en particulier si le nombre d'entiers que l'élèves veulent vérifier se limitent bien à 6 cas. Cependant, nous pensons qu'une majorité d'élèves va travailler sur une adaptation de l'algorithme initial, afin d'automatiser la vérification de la conjecture pour les nombres entiers déterminés lors de la phase 2.

Suivant l'algorithme obtenu lors de l'« ingénierie-Guy », les élèves vont procéder différemment pour adapter leur algorithme au problème posé. En effet, nous pensons que ceux qui ont travaillé lors de l'ingénierie précédente sur un type LISTE (voir l'algorithme (d)), vont probablement choisir d'adapter leur liste de nombres afin d'obtenir un algorithme permettant de procéder à une vérification automatisée sur un ensemble fini de nombres entiers.

Quant aux élèves qui ont choisi une stratégie différente lors de l'ingénierie précédente (algorithmes (a) et (b)), nous pensons qu'ils vont soit vérifier l'algorithme de façon « semi-automatisé » sur les nombres entiers retenus, c'est-à-dire vérifier méthodiquement pour chacun des 6 (ou dix) entiers, soit implémenter l'ensemble de ces nombres à l'aide d'une liste. Cependant, cette compétence algorithmique n'étant pas spécifiquement donnée par le

programme, nous pensons que cette étape d'utilisation d'une liste va poser des difficultés à des élèves relativement débutants en informatique.

5.2.8 Mise en place de l'expérimentation, déroulement et analyse *a posteriori*

5.2.8.1. Mise en place de l'expérimentation : la classe, les supports d'observations

Notre ingénierie est expérimentée à la suite de l'« ingénierie-Guy » avec les élèves ayant participé à l'expérimentation de cette ingénierie. Ces élèves forment un groupe homogène de 24 élèves d'une classe de Terminale Scientifique, enseignement de spécialité.

L'expérimentation se fait en présence du chercheur. Pour notre ingénierie, l'enseignant et le chercheur sont une seule et unique personne. Cependant, Madame Guy, agrégée de Mathématiques et étudiante en Master de didactique des mathématiques, participe à la gestion de la classe et assume d'une certaine manière le rôle de l'enseignant.

Les élèves sont en salle informatique et travaillent en binôme (12 binômes en tout), de façon qu'il y ait un ordinateur par binôme équipé d'un environnement informatique permettant d'utiliser un langage pseudo-code.

Il est précisé aux élèves qu'ils doivent utiliser l'algorithme qu'ils ont écrit lors de l'« ingénierie-Guy ». Pour les phases 1 et 3, chaque binôme doit rendre ces travaux sous forme de fichiers *AlgoBox*. Pour la phase 2, il leur est précisé que le travail se fait à l'écrit.

L'expérimentation est prévue pour une durée maximale de 2 heures. L'ensemble des phases est faite sur deux séances de travaux dirigés d'une heure se succédant à une semaine d'intervalle. La première phase débute juste après l'« ingénierie-Guy ».

L'analyse *a posteriori* est construite à partir de plusieurs supports : les enregistrements vidéo et audio des deux séances, les observations du travail des binômes et des réactions de leur enseignant. Ces divers supports favorisent l'étude et l'analyse des algorithmes produits lors des deux séances par les binômes. A la demande du chercheur, les binômes rendent à leur professeur un écrit, correspondant au travail fait pendant la phase 2, et un enregistrement sur clé USB des deux algorithmes écrits en langage Pseudo-code et implémentés sur *AlgoBox*.

5.2.8.2. Déroulement et premières analyses *a posteriori*

a) Phase 1 : Adapter l'algorithme obtenu lors de l'« ingénierie Guy » afin de déterminer une preuve par exhaustion des cas

Le temps prévu (20 minutes) lors de la préparation de cette phase a été globalement respecté. Les élèves ont repris les algorithmes qu'ils avaient obtenus lors de l'« ingénierie-Guy ».

Intéressons-nous d'abord à ceux qui ont un algorithme automatisé (du type « algorithme (b) », soit 70% des binômes) et qui n'utilisent pas une variable de type LISTE. Comme nous l'avions envisagé, une fois que leur fichier correspondant à un de ces algorithmes automatisés est ouvert, certains binômes (soit 20% des binômes « algorithme (b) ») commencent à tester l'algorithme obtenu lors de l'ingénierie précédente sans porter de modifications, successivement sur les nombres entiers en partant de 0.

Parmi les binômes « algorithme (b) », nous observons des échanges comme celui-ci :

> **Elève 1 :** *Pour les 20 premiers entiers, cela marche à chaque fois… mais c'est long.*
> **Elève 2 :** *Veux-tu que je te remplace ? On peut faire à tour de rôle les vérifications. Allez…*
> **Elève 1 :** *A ton tour alors !!*
> **Elève 2 :** *Ok.*
> Celui-ci vérifie pour les premiers nombres entiers qui suivent 20.
> **Professeur :** *Où en êtes-vous ?*
> **Elève 2 :** *Nous vérifions à tour de rôle pour des groupes de nombres, mais c'est très long. Monsieur, y-a-t-il une astuce ?*
> **Elève 1 :** *Et si nous faisions un nouveau programme, où nous mettrions une boucle « Pour de compteur = 0 à 1000 ».*
> **Professeur :** *Ta proposition est pertinente. Mais est-il nécessaire de refaire tout l'algorithme ? Et, puis pourquoi vérifier jusqu'à 1 000 ? Quelle est la consigne sur le nombre de chiffre constituant l'entier ?*
> **Elève 1 :** *3 chiffres. Donc on s'arrête à 999.*
> Le professeur laisse les élèves pour passer à un autre binôme.
> **Elève 1** *: On va ajouter de nouvelles lignes au programme. Le prof. a raison, nous n'avons pas besoin de tout refaire.*

Nous observons que ce binôme propose un premier algorithme (fig. 32) qui affiche les résultats avec tous les détails des différentes étapes du processus (extrait des résultats affichés fig. 33).

```
VARIABLES
    a EST_DU_TYPE NOMBRE
    b EST_DU_TYPE NOMBRE
    c EST_DU_TYPE NOMBRE
    n EST_DU_TYPE NOMBRE
    stock EST_DU_TYPE NOMBRE
    decroissant EST_DU_TYPE NOMBRE
    croissant EST_DU_TYPE NOMBRE
    reponse EST_DU_TYPE CHAINE
    compteur EST_DU_TYPE NOMBRE
    nombre_entier EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
    nombre_entier PREND_LA_VALEUR 0
    POUR compteur ALLANT_DE 0 A 999
        DEBUT_POUR
        n PREND_LA_VALEUR nombre_entier
        reponse PREND_LA_VALEUR "oui"
        AFFICHER "Pour n = "
        AFFICHER n
        TANT_QUE (reponse!="non") FAIRE
            DEBUT_TANT_QUE
            a PREND_LA_VALEUR n%10
            b PREND_LA_VALEUR (n-a)/10%10
            c PREND_LA_VALEUR (n-10*b-a)/100%10
            TANT_QUE (a>b OU b>c) FAIRE
                DEBUT_TANT_QUE
                SI (a>b) ALORS
                    DEBUT_SI
                    stock PREND_LA_VALEUR b
                    b PREND_LA_VALEUR a
                    a PREND_LA_VALEUR stock
                    FIN_SI
                SINON
                    DEBUT_SINON
                    SI (b>c) ALORS
                        DEBUT_SI
                        stock PREND_LA_VALEUR c
                        c PREND_LA_VALEUR b
                        b PREND_LA_VALEUR stock
                        FIN_SI
                    FIN_SINON
                FIN_TANT_QUE
            decroissant PREND_LA_VALEUR c*100+b*10+a
            croissant PREND_LA_VALEUR a*100+b*10+c
            SI (n==decroissant-croissant) ALORS
                DEBUT_SI
                reponse PREND_LA_VALEUR "non"
                FIN_SI
            n PREND_LA_VALEUR decroissant-croissant
            AFFICHER decroissant
            AFFICHER " - "
            AFFICHER croissant
            AFFICHER " = "
            AFFICHER n
            FIN_TANT_QUE
        nombre_entier PREND_LA_VALEUR nombre_entier+1
        FIN_POUR
FIN_ALGORITHME
```

Figure 32 (Cas d'un premier algorithme sous AlgoBox permettant de déterminer une preuve par exhaustion des cas de la conjecture Kaprekar)

```
***Algorithme lancé***
Pour n = 0
0 - 0 = 0
Pour n = 1
100 - 1 = 99
990 - 99 = 891
981 - 189 = 792
972 - 279 = 693
963 - 369 = 594
954 - 459 = 495
954 - 459 = 495
Pour n = 2
200 - 2 = 198
981 - 189 = 792
972 - 279 = 693
963 - 369 = 594
954 - 459 = 495
954 - 459 = 495
Pour n = 3
300 - 3 = 297
972 - 279 = 693
963 - 369 = 594
```

Figure 33 (Extrait des résultats obtenus lors de l'exécution sous AlgoBox de l'algorithme de la figure 32)

Un des élèves (élève 1) du binôme fait remarquer que ces résultats sont difficilement analysables et suggèrent à son camarade une nouvelle modification de l'algorithme. Nous avons ainsi l'échange suivant :

Elève 1 : *Regarde, nous pouvons pas vérifier facilement la conjecture avec ces résultats.*
Elève 2 : *Oui. Mais je vois pas ce qu'il faudrait faire.*

Elève 1 : *Je pense qu'il est pas nécessaire d'afficher les étapes de calcul pour un nombre entier. Enlevons les lignes en trop.*

L'élève 1 fait exécuter le programme et observe de nouveau que les résultats restent difficilement exploitables (Fig. 34 et 35).

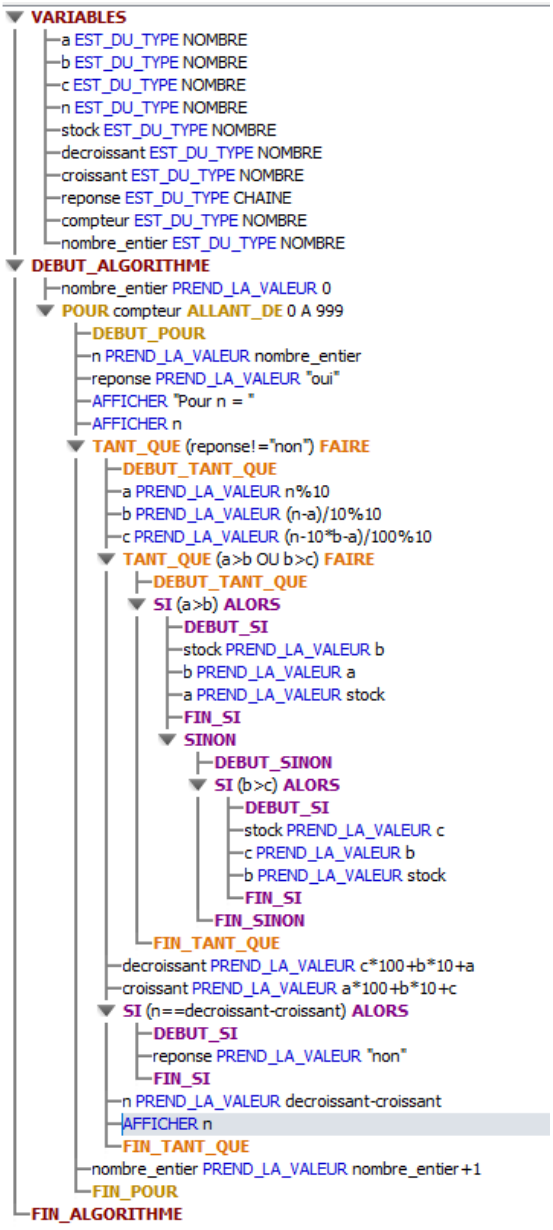


Figure 34 (Sous AlgoBox, une première modification de l'algorithme de la figure 32)

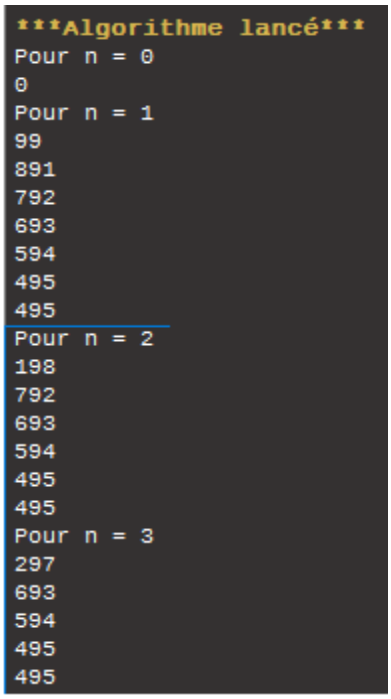


Figure 35 (Extrait des résultats obtenus lors de l'exécution sous AlgoBox de l'algorithme de la figure 34)

Son camarade (élève 2) propose alors de modifier les dernières lignes de l'algorithme afin de faire afficher par l'environnement numérique des résultats plus lisibles (Fig. 36) :

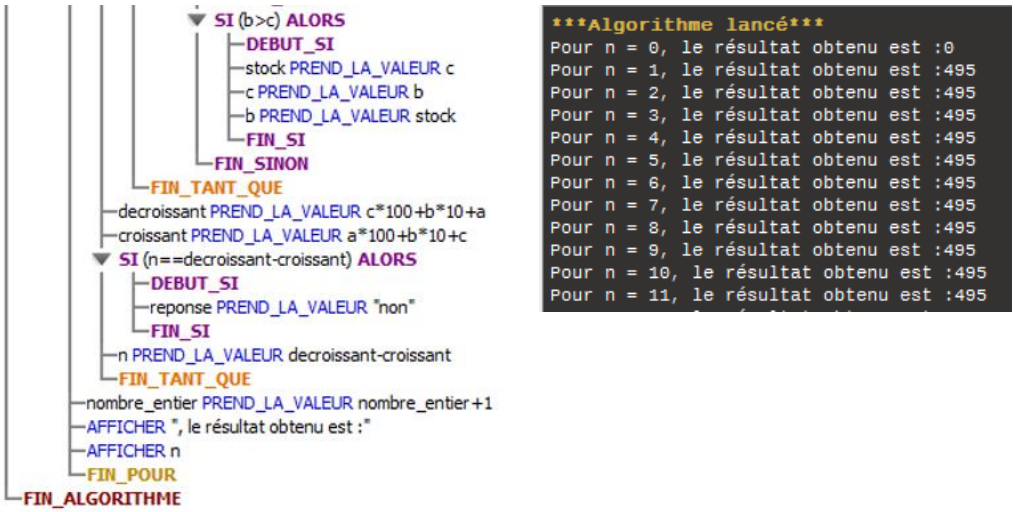


Figure 36 (Sous AlgoBox, une modification de l'algorithme de la figure 34)

Ce type d'évolution dans le travail est observé chez l'ensemble des binômes ayant obtenu, lors de l'ingénierie précédente, un algorithme semblable à celui que nous venons de voir (algorithme (b)) et qui ont commencé par traiter les nombres entiers un par un, en partant de zéro. Au cours de cette phase, pour ces binômes, l'enseignant n'intervient quasiment pas, si ce n'est pour indiquer aux élèves, une fois que ceux-ci pensent à mettre en place une boucle « Pour », qu'il ne leur est pas nécessaire de réécrire l'algorithme en entier. De plus, seul le groupe, dont on a retranscrit un extrait du dialogue entre les élèves, propose dans un premier temps de faire que la boucle vérifie de 0 à 1 000 et non de 0 à 999, ce qui nécessite un complément d'intervention de l'enseignant comme nous l'avons vu ci-dessus.

Nous observons aussi que ces élèves qui choisissent cet algorithme, utilisent deux variables de type NOMBRE, telles que l'une représente un « compteur » parcourant l'ensemble de tous les nombres entiers allant de 0 à 999 et l'autre, intitulée « nombre_entier », représente le nombre entier initial, ainsi que la succession des nombres obtenus lors de l'étape (4) à chaque procédure complète de calculs. Ces variables ont les mêmes valeurs tout au long de l'exécution, et sont donc sémantiquement identiques. Néanmoins, il semble que pour les élèves le « compteur » de boucle a un rôle particulier qui ne permet pas de le considérer dans les calculs.

Les binômes qui utilisent un algorithme initial semblable à l'algorithme (b) obtenu à la fin de l'« ingénierie-Guy », et qui ne passent pas par l'étape de vérification non automatisée pour l'ensemble des entiers de 000 à 999, éprouvent aussi des difficultés semblables à celles

décrites ci-dessus pour mettre en place une modification de l'algorithme permettant un affichage « lisible » lors de son exécution dans l'environnement numérique *AlgoBox*.

Pour ceux qui mettent en place un algorithme (algorithme (a), voir fig. 31) non automatisé (soit environ 10 % des binômes de la classe) lors de l'« ingénierie-Guy », la difficulté supplémentaire consiste tout d'abord à supprimer le fait que l'exécution de celui-ci demande à l'utilisateur s'il veut reprendre le processus de calcul de la « recette de Kaprekar » avec le nouveau nombre entier obtenu après l'étape (4) de ce processus. En effet, ces élèves, ayant eu des difficultés à mettre un test d'arrêt automatisé, ont choisi une telle procédure semi-automatisée afin de vérifier la conjecture. Ainsi, le travail débute par une première transformation de leur algorithme afin que celui-ci soit totalement automatisé lors de la vérification de la conjecture pour un nombre entier arbitraire choisi par l'utilisateur entre 0 et 999.

Pour cela, les binômes concernés doivent supprimer une variable de type CHAINE qui correspond au fait que l'environnement informatique demande à l'utilisateur s'il souhaite ou non reprendre la « recette de Kaprekar » avec le nouvel entier obtenu (fig. 37)

Figure 37 (Extrait d'un algorithme Kaprekar sous Algobox)

Figure 38 (Extrait d'un algorithme Kaprekar sous Algobox)

Les élèves doivent alors introduire une structure conditionnelle permettant à l'algorithme de reprendre de façon automatisé le processus de calcul avec le nombre entier obtenu,

jusqu'à obtenir la non vérification du test de sortie (Fig. 38). Ils utilisent aussi une variable de type CHAINE (intitulée « réponse ») afin d'enregistrer la réponse proposée par l'algorithme qui lui permet de reprendre ou non le processus de calcul avec le dernier nombre entier obtenu. Nous observons que ces élèves n'ont finalement éprouvé aucune difficulté à faire cette adaptation.

Le fait que ces élèves élaborent un algorithme semi-automatisé lors de l'« ingénierie-Guy », nous conduit à avoir un échange avec ces derniers, qui n'est pas prévu initialement dans le cadre de notre ingénierie. Ainsi lors de cet échange, en aval de notre expérimentation, à la question sur le fait qu'ils choisissent de faire que l'algorithme initial de l'« ingénierie-Guy » soit semi-automatisé, ils répondent que ceci est dû à ce que l'enseignant a proposé un test bâti sur deux conditions : *l'égalité à 0 ou à 495* suivant le nombre entier initial choisi. En effet, il leur semble alors qu'ils ne sont pas en mesure de procéder à une transposition du « OU » dans l'environnement informatique utilisé (nous renvoyons pour cela aux analyses faites par Guy (2013) dans le cadre de son mémoire de M2). Nous constatons ainsi qu'un certain nombre de binômes est confronté à ce problème de test de sortie. Cependant, dans le cadre de cette phase de notre ingénierie, les élèves concernés savent modifier leur premier algorithme afin de le rendre automatisé pour répondre au problème de traitement complet sur un nombre entier choisi arbitrairement. Une fois ce problème résolu, nous observons qu'ils sont eux-aussi confrontés aux problèmes de construction et de modification de l'algorithme automatisé sur un nombre entier arbitraire afin de permettre à celui-ci, après son implémentation dans un environnement informatique, de vérifier la conjecture pour tous les nombres entiers compris 000 à 999 dans le cadre d'une preuve par exhaustion des cas.

Ceux qui ont construit, lors de l'« ingénierie Guy », un algorithme automatisé mettant en jeu un test d'arrêt du type « nombre entier obtenu différent à 0 et à 495 » (voir l'algorithme (d)), n'ont qu'à construire une boucle « Pour » pour obtenir une automatisation de la vérification de la conjecture pour tous les nombres entiers compris entre 000 et 999. En effet, ces élèves qui ont une bonne maîtrise du langage pseudo-code observée lors de l'introduction de l'opérateur « ET » dans le test d'arrêt de l'algorithme initial, proposent des algorithmes complets avec une boucle « Pour » permettant de vérifier la conjecture pour tous les nombres entiers naturels inférieurs à 999 (fig. 39).

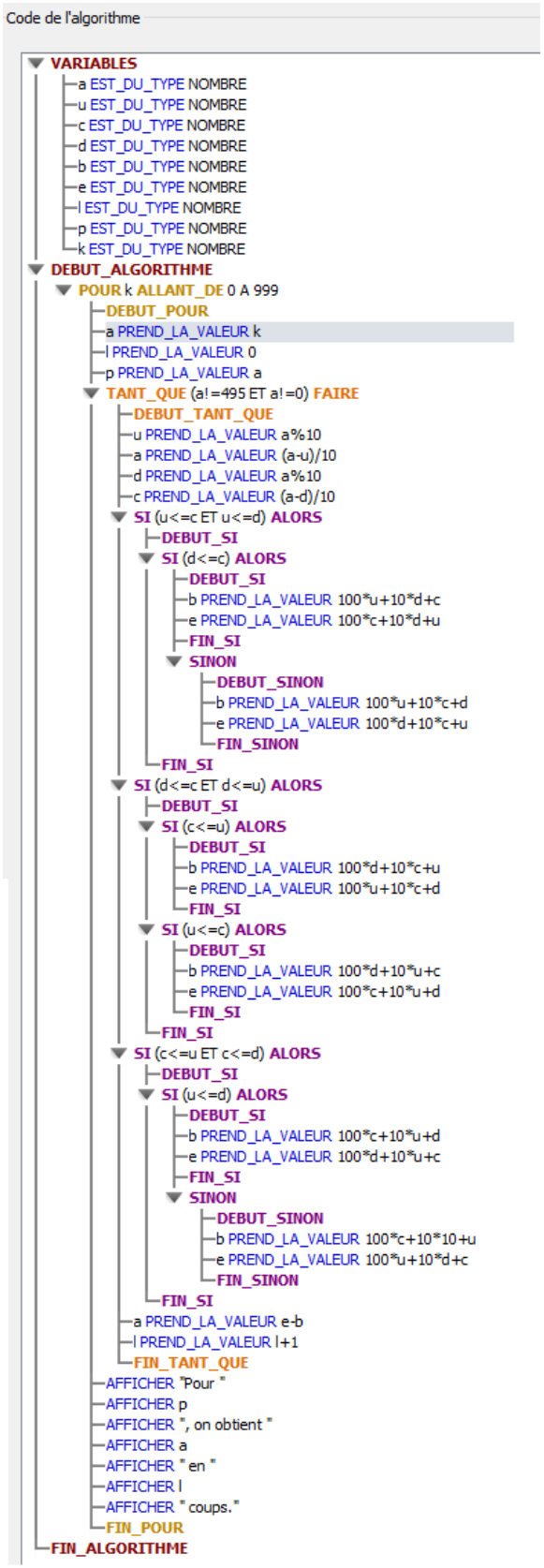


Figure 39 (Sous AlgoBox, un algorithme automatisé permettant une preuve par exhaustion des cas de la conjecture Kaprekar, pour tous les entiers naturels inférieurs à 999)

Quant à ceux qui ont travaillé lors de l'« ingénierie-Guy » sur un algorithme introduisant une variable de type LISTE pour stocker les trois chiffres qui constituent le nombre entier, nous observons qu'une très grande majorité de ces derniers ne modifient pas leur algorithme (fig. 40) (seul un binôme propose l'utilisation d'une deuxième liste (fig. 41)) en introduisant une nouvelle liste qui est associée à l'ensemble des nombres entiers compris entre 0 et 999 pour générer une validation automatisée de la conjecture.

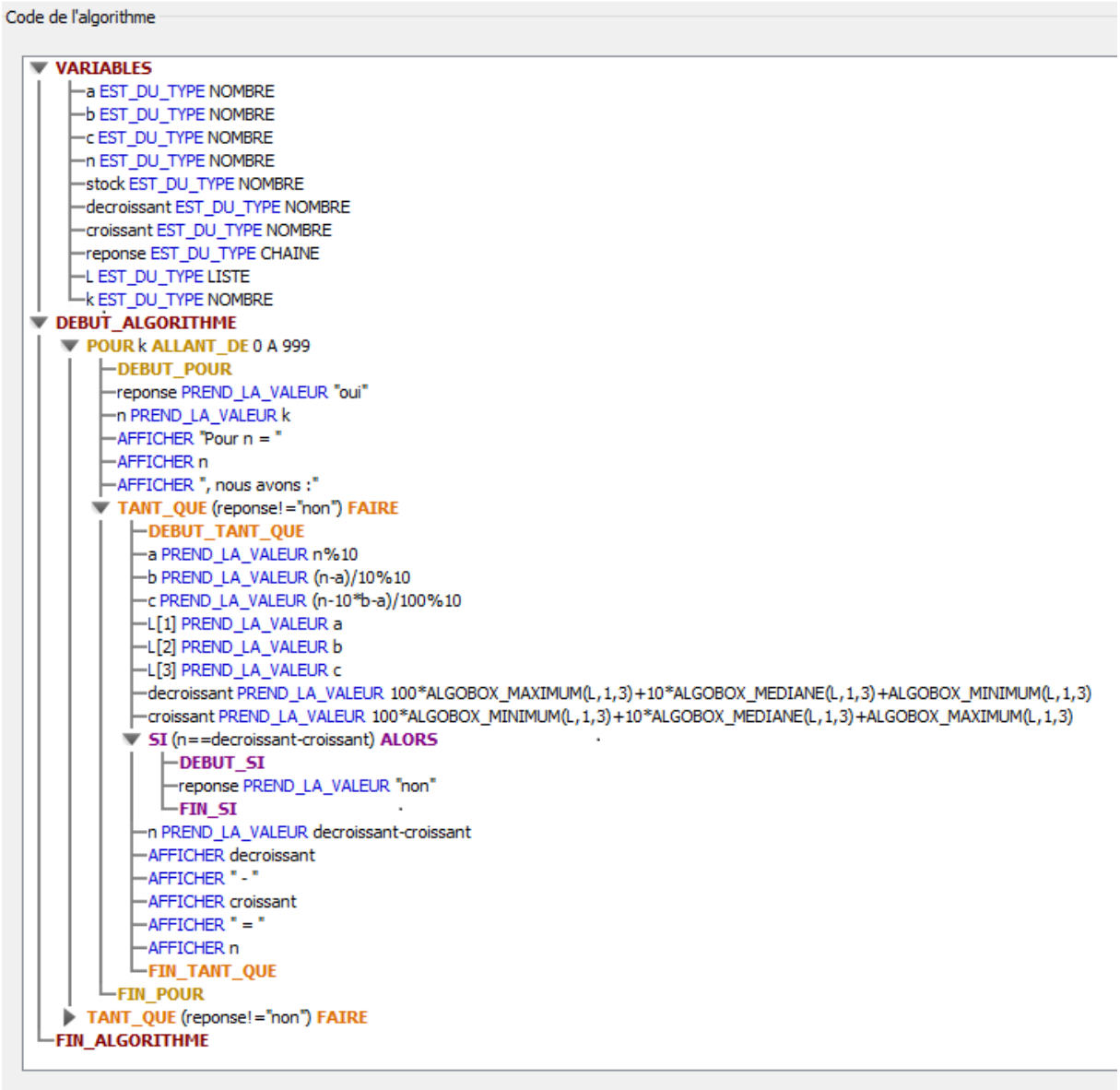


Figure 40 (Sous AlgoBox, un algorithme automatisé avec utilisation d'une liste, permettant une preuve par exhaustion des cas de la conjecture Kaprekar, pour tous les entiers naturels inférieurs à 999)

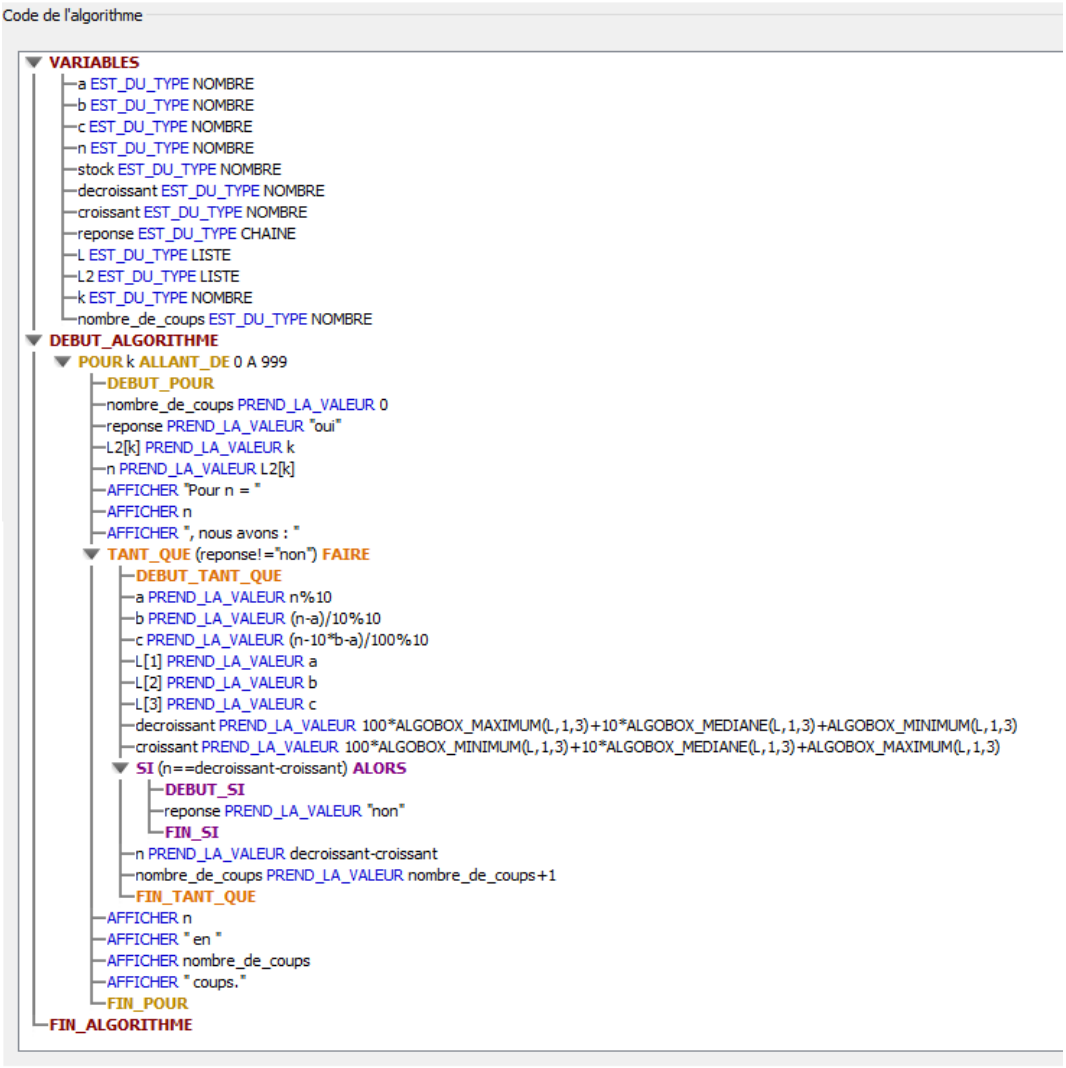


Figure 41 (Sous AlgoBox, un algorithme automatisé avec utilisation de deux listes, permettant une preuve par exhaustion des cas de la conjecture Kaprekar, pour tous les entiers naturels inférieurs à 999)

Nous pouvons aussi observer que dans le cas du binôme qui utilise une deuxième liste, notée L2 (fig. 10b), afin de parcourir l'ensemble des entiers naturels compris entre 0 et 999, celui-ci aussi prend en compte le nombre de coups qui est nécessaire pour obtenir une des valeurs de la conjecture, pour chaque entier compris entre 0 et 999. Ce binôme montre une bonne connaissance des possibilités qu'offre l'environnement informatique. Néanmoins, les valeurs stockées dans la seconde liste L2 ne sont pas réutilisées.

b) Phase 2 : Justifier qu'il suffit de vérifier la conjecture que sur six nombres entiers

Pendant cette phase, les élèves regroupés en binôme travaillent sur une simple feuille de papier. Chaque élève, ayant sa propre feuille de papier, justifie le fait que le nombre d'entiers à vérifier peut être limité à un nombre de cas précis qu'il doit déterminer. Sachant que tous

les élèves ont lu en amont de cette phase l'article sur le *Rubik's Cube*, l'enseignant propose un temps d'échange collectif autour de la finalité de l'article en rapport aux attentes de la phase en court.

Nous observons (sans surprise) que dans un premier temps une majorité d'élèves (plus de 90 %) ne comprend pas réellement la relation possible entre le contenu de l'article et les attentes de la phase. Cependant, un élève émet l'hypothèse que l'article permettrait de se poser la question du bien fondé de procéder à une vérification pour l'ensemble des mille nombre entiers inférieurs à 999. En effet, il souligne que dans la problématique sur Kaprekar, cette vérification à un « coût » informatique négligeable mais que si on doit le faire au « papier-crayon » cela pourrait prendre plusieurs heures. De plus, suite à cette intervention, d'autres signalent que des nombres comme « *358, 385, 538, 583, 835 et 853 sont constitués des mêmes chiffres et que par conséquent après la décomposition du nombre initial en chiffres, l'ensemble des étapes qui suivent, va consister à refaire le même type de calcul en un nombre de coups équivalents pour aboutir au final au même résultat attendu, c'est-à-dire 495.* » Ce temps d'échange dure environ 5 minutes.

Une fois que cet échange autour de l'article sur le *Rubik's Cube* est clos et que la totalité des élèves s'est ralliée à l'idée de procéder à une étude du nombre de cas à faire vérifier par l'algorithme implémenté dans un environnement informatique, nous observons que l'ensemble de la classe comprend la consigne, et que chacun des élèves utilise alors une écriture du type $N = \overline{abc} = 100\,a + 10\,b + c$, où *a*, *b* et *c* sont des entiers compris entre 0 et 9 pour élaborer une preuve qui justifie qu'elles sont les nombres entiers qu'il faut réellement vérifier afin de valider la conjecture.

Les élèves calculent sans difficulté la différence entre les nombres entiers dont les trois chiffres sont triés. En général, il parte de l'hypothèse que les trois chiffres *a*, *b* et *c* du nombre initial N vérifient les inégalités suivantes $a \leq b \leq c$, puis ils déterminent la différence $\overline{cba} - \overline{abc}$ (fig. 11a à 11d) est égale à $99\,(c - a)$, en utilisant le fait que cette différence revient à calculer $(100\,c + 10\,b + a) - (100\,a + 10\,b + c)$. Pour ce calcul, les élèves réutilisent un résultat qu'ils avaient revu lors de l'« ingénierie-Guy » sur la décomposition d'un nombre entier dans le système décimal.

(2) Soient n_0 un nombre à trois chiffres, éventuellement nuls, que l'on écrit $\overline{abc}$ où a, b et c sont donc des entiers compris entre 0 et 9 qui représentent respectivement le chiffre des centaines, le chiffre des dizaines et le chiffre des unités de n_0, et n_p le nombre de trois chiffres, éventuellement nuls, obtenu après p applications de l'algorithme de Kaprekar.

(a) Expliquer pourquoi on peut, sans perdre de généralité, supposer que $a \leq b \leq c$ tels que deux au moins sont distincts.

(b) Démontrer la conjecture faite à la **question 2. (1)**.

En déduire toutes les valeurs possibles de n_1.

Peut-on encore réduire le nombre de cas à vérifier ?

(c) Modifier l'algorithme que vous avez proposé à la **question 1.**, en utilisant les observations faites à la question précédente. On présisera le nombre maximum d'itérations nécessaires pour l'obtention du résultat remarquable. Programmer cet algorithme sous *AlgoBox*.

→ Joindre le programme écrit sous *AlgoBox*.

Figure 42 (Extrait d'une copie sur Kaprekar d'un élève de Terminale Scientifique)

(2) Soient n_0 un nombre à trois chiffres, éventuellement nuls, que l'on écrit $\overline{abc}$ où a, b et c sont donc des entiers compris entre 0 et 9 qui représentent respectivement le chiffre des centaines, le chiffre des dizaines et le chiffre des unités de n_0, et n_p le nombre de trois chiffres, éventuellement nuls, obtenu après p applications de l'algorithme de Kaprekar.

(a) Expliquer pourquoi on peut, sans perdre de généralité, supposer que $a \leq b \leq c$ tels que deux au moins sont distincts.

Même si l'inégalité n'est pas respectée (ex: $b_0 \leq a_0 \leq c_0$), la seconde étape de l'algorithme retournera de toute manières les chiffres ordonnés (ici: $a_1 = b_0$; $b_1 = a_0$; $c_1 = c_0$)
$a_1 \leq b_1 \leq c_1$

(b) Démontrer la conjecture faite à la **question 2. (1)**.

$\overline{cba} \rightarrow 100c + 10b + a - 100a - 10b - c = -99a + 99c = 99(-a + c)$ or $c - a \in \mathbb{Z}$
↳ divisible par 99

En déduire toutes les valeurs possibles de n_1.

$99 \times 1 = 99$
$\times 2 : 198$
$\times 3 : 297$
$\times 4 : 396$
$\times 5 : 495$
$\times 6 : 594$
$\times 7 : 693$
$\times 8 : 792$
$\times 9 : 891$
$\times 10 : 990$

Peut-on encore réduire le nombre de cas à vérifier ?

On trouve des nombres équivalents après "réadaptation". Les valeurs de n_1 composés de 2 chiffres sont équivalents dans l'algorithme.
On passe de 10 groupes : $(c-a) = 0, 1, 2, \ldots, 10$ à 5 groupes : $(c-a) = 0, 1, 2, \ldots, 5$

car

(c) Modifier l'algorithme que vous avez proposé à la **question 1.**, en utilisant les observations faites à la question précédente. On présisera le nombre maximum d'itérations nécessaires pour l'obtention du résultat remarquable. Programmer cet algorithme sous *AlgoBox*.

→ Joindre le programme écrit sous *AlgoBox*.

Figure 43 (Extrait d'une copie sur Kaprekar d'un élève de Terminale Scientifique)

(2) Soient n_0 un nombre à trois chiffres, éventuellement nuls, que l'on écrit $\overline{abc}$ où a, b et c sont donc des entiers compris entre 0 et 9 qui représentent respectivement le chiffre des centaines, le chiffre des dizaines et le chiffre des unités de n_0, et n_p le nombre de trois chiffres, éventuellement nuls, obtenu après p applications de l'algorithme de Kaprekar.

(a) Expliquer pourquoi on peut, sans perdre de généralité, supposer que $a \leq b \leq c$ tels que deux au moins sont distincts.

L'ordre des lettres dans le premier nombre ne change pas la suite de l'algorithme

(b) Démontrer la conjecture faite à la **question 2. (1)**.

$c \times 100 + b \times 10 + a - a \times 100 + b \times 10 + c$

$= a(1 - 100) + b(10 - 10) + c(100 - 1)$

$= -99a + 99c$

$= 99(c - a)$ donc cette suite est divisible par 11

En déduire toutes les valeurs possibles de n_1.

99, 198, 297, 396, 495, 594, 693, 792, 891, 990

$n_1 = 99k$, $k \in [\![1, 10]\!]$

Peut-on encore réduire le nombre de cas à vérifier ?

On divise l'ensemble des 1000 nb à tester en 11 groupes selon le premier nb obtenu.
Tester un nb de chaque groupe revient à tester l'ensemble des membres de ce groupe.

(c) Modifier l'algorithme que vous avez proposé à la **question 1.**, en utilisant les observations faites à la question précédente. On présisera le nombre maximum d'itérations nécessaires pour l'obtention du résultat remarquable. Programmer cet algorithme sous *AlgoBox*.

→ Joindre le programme écrit sous *AlgoBox*.

Figure 44 (Extrait d'une copie sur Kaprekar d'un élève de Terminale Scientifique)

(2) Soient n_0 un nombre à trois chiffres, éventuellement nuls, que l'on écrit $\overline{abc}$ où a, b et c sont donc des entiers compris entre 0 et 9 qui représentent respectivement le chiffre des centaines, le chiffre des dizaines et le chiffre des unités de n_0, et n_p le nombre de trois chiffres, éventuellement nuls, obtenu après p applications de l'algorithme de Kaprekar.

(a) Expliquer pourquoi on peut, sans perdre de généralité, supposer que $a \leq b \leq c$ tels que deux au moins sont distincts.

l'ordre des chiffres dans le 1er nbre importe peu.

(b) Démontrer la conjecture faite à la **question 2. (1).**

$cx100 + bx10 + a - (ax100 + bx10 + c)$
$= a(1-100) + c(100-1)$
$= -99a + 99c$
$= 99(c-a)$ de cette - cette est divisible par 99.
$\in \mathbb{Z}$

En déduire toutes les valeurs possibles de n_1.

n_1 peut être égal à $99k$, avec $k \in [\![1;10]\!]$
c'est à dire : 99, 198, 297, 396, 495, 594, 693, 792, 891, 990.

Peut-on encore réduire le nombre de cas à vérifier ?

On sait que certains n_1 sont donnés de fin n_0.
On peut donc les recopier.
Ce qui donnera finalement 5 cas à vérifier :
99, 198, 297, 396, 495.

(c) Modifier l'algorithme que vous avez proposé à la **question 1.**, en utilisant les observations faites à la question précédente. On présisera le nombre maximum d'itérations nécessaires pour l'obtention du résultat remarquable. Programmer cet algorithme sous *AlgoBox*.

→ Joindre le programme écrit sous *AlgoBox*.

Figure 45 (Extrait d'une copie sur Kaprekar d'un élève de Terminale Scientifique)

En étudiant ces extraits de travaux de binômes, nous observons que malgré l'hypothèse $a \leq b \leq c$ pour le nombre entier N = $\overline{abc}$, les élèves considèrent que la différence $c - a$ peut représenter un nombre entier relatif. En revanche, l'ensemble des élèves constatent que la

différence $\overline{cba} - \overline{abc}$ est égale à un nombre entier multiple de 99. Ils écrivent en général ce nombre sous la forme 99 *k*, avec *k* représentant un nombre entier.

Nous observons aussi qu'une confusion existe chez 90 % des binômes qui considèrent que la différence *c* – *a* peut être égale à 10, et que 20 % des binômes ne prend pas compte le cas où cas où *c* – *a* serait égal à zéro, c'est-à-dire le cas où *a* = *c* (= *b*).

Une fois que cette première étape de la preuve est faite, les binômes proposent en général la liste suivante des nombres entiers à vérifier : 099, 198, 297, 396, 495, 594, 693, 790, 891, 990. 80 % des binômes ajoutent le cas 000 dans les nombres entiers à vérifier.

Dans un second temps, nous constatons qu'à la question « Peut-on encore réduire le nombre de cas à vérifier ? », 70 % des binômes répondent que certains nombres de la liste donnée ci-dessus vont, lors du renouvellement du processus de calcul, donner des résultats identiques, ce qui confirmerait les échanges qu'ils ont eu au début de cette phase sur les apports de l'article sur le *Rubik's Cube*. Ils concluent alors que la liste des nombres à vérifier peut se réduire à 099, 198, 297, 396 et 594, soit un total de cinq nombres entiers. Nous constatons ainsi que leur choix se porte sur 594 et non 495. L'enseignant leur demande alors : « *A propos, pourquoi choisissez-vous 594 au lieu de 495 ?* ». Les élèves répondent alors que « *495 n'est pas à prendre en compte puisque c'est une des valeurs possibles de la conjecture* ». Cependant, ce questionnement interpelle certains élèves qui après une brève discussion informelle entre eux signalent qu'en fait prendre le nombre 495 ou 594 n'a pas d'importance puisqu'ils sont tous les deux constitués des mêmes chiffres et que par conséquent le calcul de la première différence va donner le même résultat. Ainsi, certains concluent que l'importance est de prendre des nombres qui ne sont pas constitués des trois mêmes chiffres et que l'ordre de ces chiffres n'a pas d'importance pour la validation de la conjecture. De plus, les élèves constatent que l'hypothèse que 495 serait un des deux résultats possibles de la conjecture et que par conséquent ce nombre ne serait pas à prendre en compte est erroné car ils indiquent que l'on ne peut pas savoir que ce résultat correspond à la conjecture lors de la vérification. Cependant, nous observons que certains élèves critiquent alors le choix d'un test de sortie qui prendrait comme acquis que la conjecture serait vérifiée avant même sa démonstration. Ceci va faire l'objet d'un temps d'échange lors de la dernière phase. Pour finir, un nombre conséquent d'élèves (environ 30 % des binômes) considèrent que le cas 000 doit être aussi

pris en compte pour la validation de la conjecture. En effet, ils disent que « *ce nombre est aussi un multiple de 99 (99 × 0) inférieur à 999 et il représente les nombres dont tous les chiffres seraient égaux* ». De plus, comme le soulignent ces élèves, il permet de valider le choix du test de sortie portant sur les valeurs finales qui pourraient être soit 0, soit 495 après que la procédure de calcul soit complètement exécutée pour un nombre entier arbitraire compris entre 00 et 999.

c) Phase 3 : Modifier l'algorithme pour une exécution sur une liste d'entiers non consécutifs

Conformément à ce qui avait été prévu lors de l'analyse *a priori*, les élèves travaillent sur l'ensemble des nombres entiers déterminés à la phase précédente.

- **Validation au « papier-crayon » de la conjecture**

8 % des binômes se contentent de vérifier au « papier-crayon » la conjecture sur les six nombres déterminés à la phase 2. Ces élèves signalent que le fait d'avoir très peu de cas à vérifier ne nécessitent pas de travailler dans l'environnement numérique, d'autant que pour chaque binôme concerné, ils se répartissent équitablement les six nombres entiers à vérifier. Nous constatons aussi que ces binômes avaient éprouvé des difficultés lors de l'« ingénierie-Guy » pour la transcription en langage pseudo-code de la « recette de Kaprekar ».

- **Exécution « non automatisée » de l'algorithme obtenu à la fin de l'« ingénierie-Guy » sur les six nombres déterminés à la phase 2**

Nous observons que certains binômes (environ 30 %) vont se contenter, dans un premier temps, de faire tourner l'algorithme obtenu à la fin de l'« ingénierie-Guy » et implémenter dans l'environnement numérique retenu, les six nombres entiers déterminés à la phase précédente. Une fois cette vérification faite sur chacun des six nombres entiers, ces élèves concluent à la validité de la conjecture.

Toutefois parmi ces binômes, une très forte majorité va se référer, dans un second temps, à la modification de l'algorithme pratiquée lors de la phase 1. En effet, ne souhaitant pas se contenter d'une telle validation, ils proposent de reprendre le travail fait précédemment en mettant en place une nouvelle démarche basée sur l'utilisation d'une boucle « POUR ». Ces binômes émettent l'hypothèse que bien que les 6 nombres ne soient pas consécutifs, il doit

être possible d'adapter l'algorithme de la phase 1 afin d'automatiser la validation. Cependant, nous constatons qu'à la fin de la séance 5 % d'entre eux n'auront pas réussi cette automatisation faute de réussir à adapter la boucle « POUR ». En effet, la boucle « POUR » de l'environnement numérique retenu ne permet pas de mettre en place un « pas » autre que 1.

- **Mise en place d'une boucle « POUR » associée à la multiplicité par 99**

Parmi les binômes ayant utilisé au départ l'algorithme implémenté obtenu lors de l'« ingénierie-Guy » et souhaité dans un second temps procéder à une modification de l'algorithme de la phase 1 de notre ingénierie avec l'introduction d'une boucle « POUR », 90% observent que les six nombres à vérifier sont des multiples de 99. Pour illustrer ce fait, nous avons par exemple cet échange entre deux élèves d'un même binôme :

> **Elève a :** *Je pense qu'il suffit de reprendre l'algorithme de la dernière fois*.
> **Elève b :** *Oui. L'algorithme d'avant il a eu une boucle pour vérifier pour tous les entiers de 0 à 999, mais je ne vois pas comment créer une boucle permettant de vérifier que pour certains entiers qui ne se suivent pas.*
> **Elève a :** *Attends… Nous avons 0, 99, 198, 297, etc. Ils se suivent mais avec un écart de 99.*
> **Elève b :** *Tu as raison. T'as vu : c'est les premiers multiples de 99. Donc, il suffirait de faire un pas de 99.*
> Les élèves regardent si le logiciel *AlgoBox* permet d'utiliser un « pas » autre que 1 pour une boucle de type « Pour ».
> **Elève b :** *Regarde, la variable utilisée pour la boucle « POUR » ne peut-être que du type NOMBRE et la valeur de cette variable est automatiquement augmentée de 1 à chaque boucle, alors nous pouvons utiliser ici cette boucle. Donc, tu as raison nous devons reprendre l'algorithme de la dernière fois*.

En tenant compte des binômes qui dès le début de cette phase 3 utilisent le caractère de multiplicité par 99, nous observons que 75 % des binômes de la classe reprennent l'algorithme de la phase 1 en utilisant une boucle « POUR » et le fait que les nombres entiers à vérifier pour valider la conjecture sont de la forme $99 \times k$, où k est un entier parcourant l'ensemble {0 ; 1 ; 2 ; 3 ; 4 ; 5 }, voire {0 ; 1 ; 2 ; 3 ; 4 ; 5 ; 6 ; 7 ; 8 ; 9} si le binôme ne tient pas compte des équivalences de comportement à la fin de la première procédure de calcul de la « recette de Kaprekar » sur certains nombres entiers. En effet, à la fin de cette première procédure, ces binômes constatent que l'on obtient le même résultat pour 198 et 891, pour 297 et 792, pour 396 et 693, ainsi que pour 495 et 594. Nous pouvons observer que cette stratégie axée sur l'utilisation des multiples de 99 n'a pas été envisagée lors de l'analyse *a priori*. Ainsi, ils reprennent dans un premier temps un algorithme de type (b) ou (d) obtenu à la fin de

l'« ingénierie-Guy » qu'ils modifient en créant une nouvelle variable de type NOMBRE, notée *k*, puis une boucle « Pour *k* allant de 0 à 5 » ou « Pour *k* allant de 0 à 9 ». Ces binômes produisent des algorithmes comme ceux donnés ci-dessous (fig. 46) où ils affectent 99**k* à la variable représentant le nombre à tester de façon à ce que celle-ci parcourt l'ensemble déterminé à la phase 2.

Comme nous le signalions ci-dessus cette stratégie autour de la multiplicité à 99 n'a pas été retenue lors de l'analyse *a priori*. Les élèves qui ont fait ce choix indiquent qu'ils ont été dans l'impossibilité de construire une variable de type LISTE. En revanche, leur compréhension de l'ensemble des nombres trouvé à la phase 2 comme multiples successifs de 99, et leur habileté à utiliser le compteur de boucle pour parcourir cet ensemble leur permet de construire un algorithme correct. Nous interprétons cela, certes comme un déficit dans l'*ETA* (pas de disponibilité du type LISTE), mais aussi comme une coordination efficace de l'*ETM* et de l'*ETA* (multiples de 99 et boucle « POUR »).

Nous notons aussi que pour ces binômes, l'ensemble des multiples de 99 est parcouru en affectant à la variable *n* le produit du compteur par 99, et non pas accumulation (*n* prend la valeur *n* + 99).

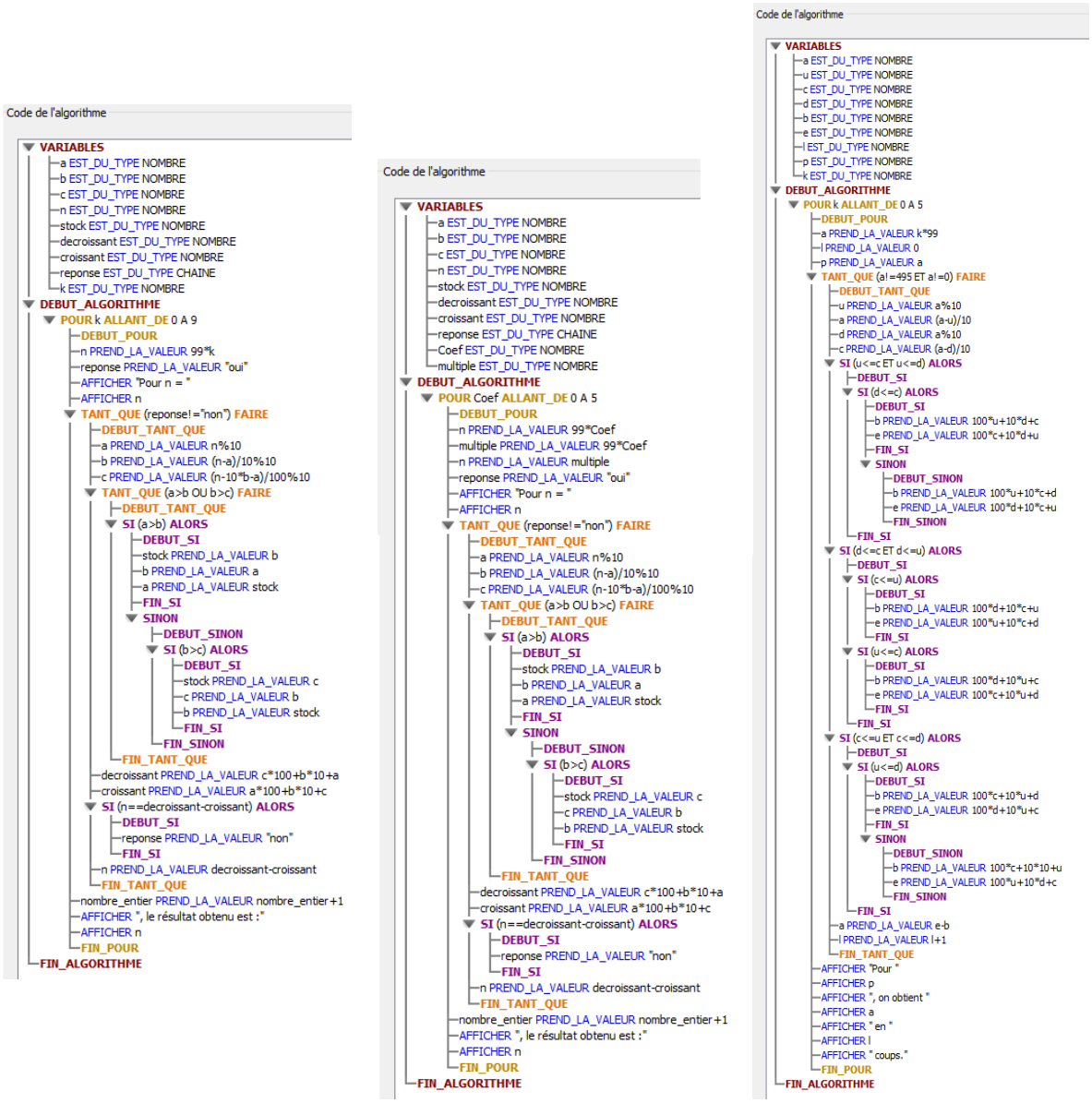


Figure 46 (Divers algorithmes d'élèves sur Kaprekar implémentés dans AlgoBox)

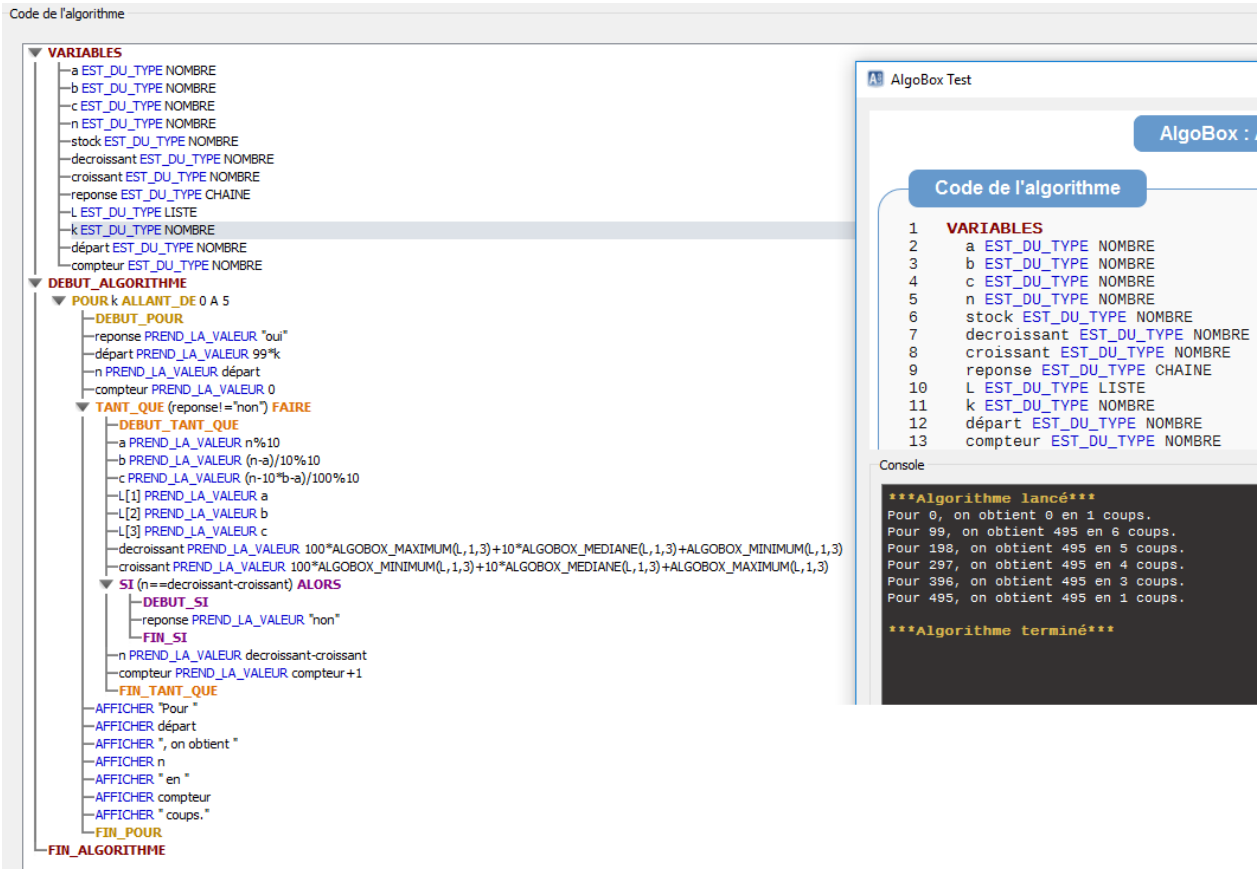


Figure 47 (Cas d'un algorithme Kaprekar « automatisé » sous Algobox avec utilisation d'une LISTE de six entiers de la forme 99 k, où k est un entier allant de 0 à 5)

Dans un cas, les élèves utilisent une LISTE notée L2 à six éléments et la peuplent avec les multiples de 99 (fig. 48). Ces élèves utilisaient déjà une liste pour les 1000 entiers en première phase, et ils remplacent ici la ligne « POUR *k* ALLANT de 0 A 999 » par la ligne « POUR *k* ALLANT de 0 à 5 » et la ligne « L2[k] PREND LA VALEUR *k* », « L2[*k*] PREND LA VALEUR 99**k* ». Mais, comme en première phase, cette liste n'est pas utilisée et donc l'algorithme produit est le même que celui des élèves précédents.

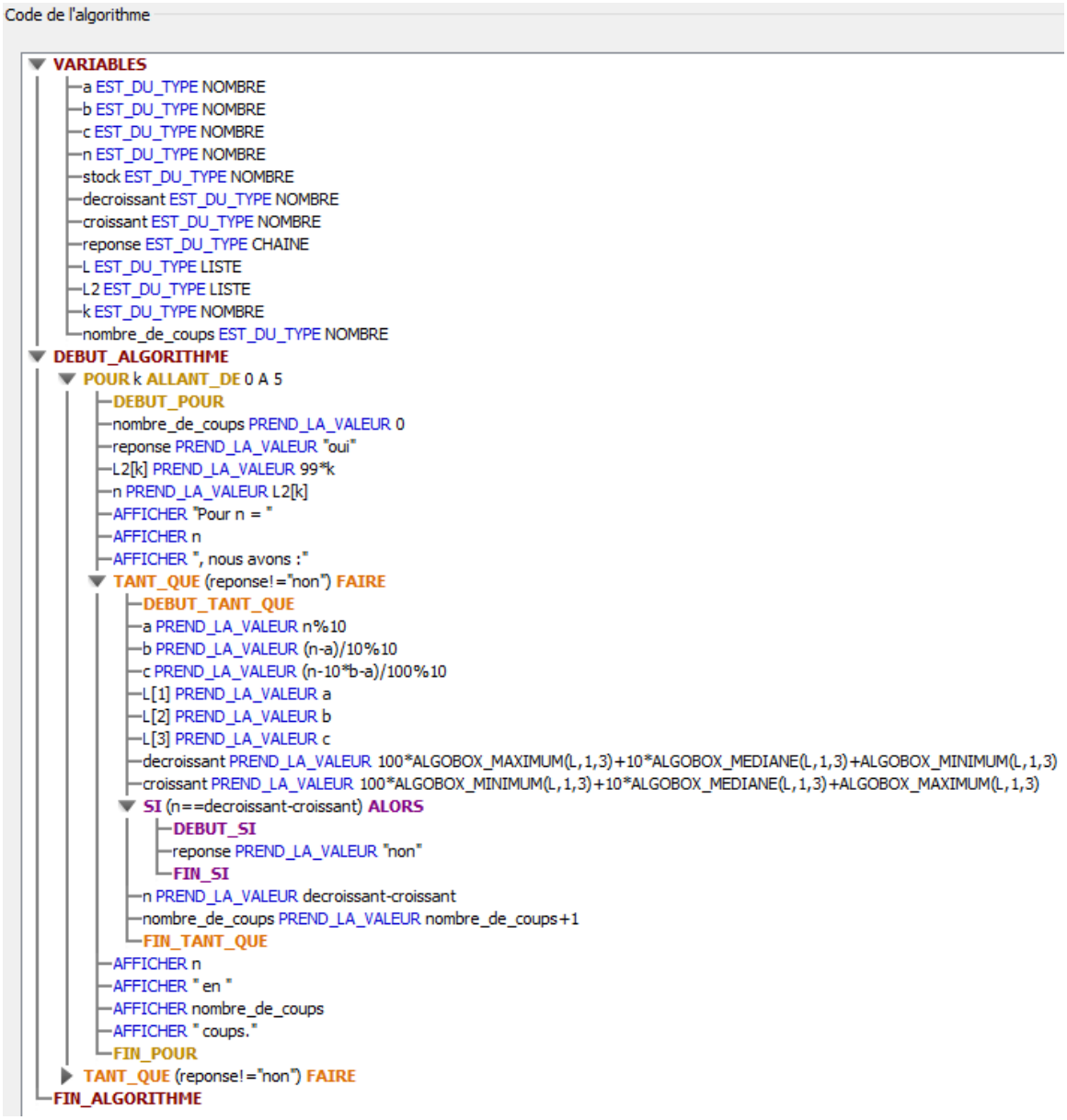
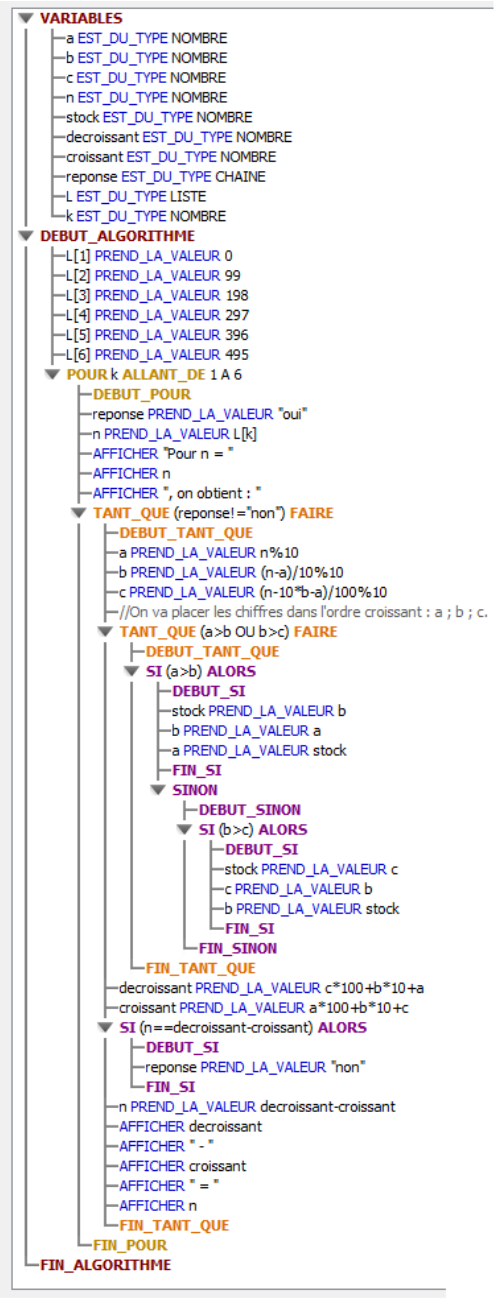


Figure 48 (Cas d'un algorithme Kaprekar « automatisé » sous Algobox avec une utilisation de deux LISTE pour obtenir une preuve « explicative » de la conjecture)

- **Mise en place d'une variable « LISTE » construite *a priori* pour le parcours des 6 nombres**

D'autres déclarent de même une LISTE, mais ils la « peuplent » avant l'itération sans référence aux multiples de 99 (fig. 49). Ces binômes représentent 5 % de l'ensemble des binômes. Ce sont les seuls dont la procédure est conforme à l'analyse *a priori*.



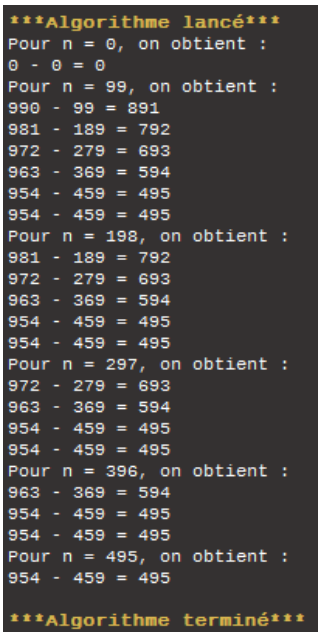


Figure 49 (Cas d'un algorithme Kaprekar « automatisé » sous Algobox avec utilisation d'une LISTE complétée avant l'itération sans référence aux multiples de 99)

Nous résumons dans le tableau (Tableau n° 11) ci-dessous, les différents choix qui sont mis en place par les binômes.

Choix	Validation au « papier-crayon »	Réutilisation sans modification de l'algorithme implémenté à la fin de l'« ingénierie-Guy »	Parcours des cas comme multiples de 99	Mise en place d'une variable « LISTE » peuplée *a priori* pour le parcours des 6 nombres
Effectif en %	8%	2%	85%	5%

Tableau 11

5.2.8.3. Analyses *a posteriori* en lien avec notre cadre théorique

L'analyse *a posteriori* de cette ingénierie reprend les résultats principaux observés plus haut dans la partie déroulement, en se référant aux *Espaces de Travail* décris au chapitre 2 de la partie I sur les cadres théoriques. Nous souhaitons analyser ces résultats en termes de genèses dans un *ETA* et un *ETM*, et vérifier notre hypothèse selon laquelle, au cours de cette ingénierie, ces genèses fonctionnent de façon convergente.

a) *Phase 1 : Adapter l'algorithme obtenu lors de l'ingénierie Guy afin de déterminer une preuve par exhaustion.*

Dans le tableau n° 12, les observations (colonne de gauche) renvoient à des genèses dans l'ETA (colonne de droite). Nous les groupons en trois grandes genèses selon la conceptualisation en jeu : **(1)** les rapports entre utilisateur du programme et dispositif informatique ; **(2)** l'itération ; **(3)** la modularité.

Toutes les genèses impliquent une connaissance de l'environnement informatique. Elles peuvent par conséquent être qualifiées d'instrumentales. Certaines concernent la façon dont les élèves se représentent le traitement par le dispositif informatique. Nous les qualifions d'« instrumentales/discursives ». Les autres impliquent plutôt le maniement de certains éléments du langage. Nous les qualifions alors d'« instrumentales/sémiotiques ».

Observations	Genèses dans l'ETA
Tous les élèves ne conçoivent pas d'emblée que la vérification peut être entièrement automatisée, sans entrée de donnée.	Genèse « instrumentale/discursive » des rapports entre utilisateur du programme et dispositif informatique : comprendre que, dans la tâche demandée, le contrôle de l'exécution passe de l'utilisateur au dispositif.
Les élèves dont l'algorithme obtenu lors de la l'« ingénierie-Guy » demandait une intervention de l'utilisateur à chaque pas de l'itération prennent conscience avec difficulté de la nécessité de supprimer l'instruction de lecture correspondante et d'aménager la condition d'arrêt de la boucle de type « TantQue ».	Genèse « instrumentale/sémiotique » des rapports entre utilisateur du programme et dispositif informatique : comprendre la différence entre l'affectation d'une variable d'une part par lecture et d'autre part par un calcul à partir de valeurs précédentes.
Tous les algorithmes obtenus affichent une série de 1000 résultats. Cela permet de contrôler le bon déroulement de l'itération, mais rend la vérification difficilement exploitable. Aucun élève ne propose de limiter l'affichage à des valeurs réfutant la conjecture.	Genèse « instrumentale/discursive » des rapports entre utilisateur du programme et dispositif informatique : dans la production des élèves, la vérification proprement dite est laissée à l'utilisateur.

La vérification automatique étant conçue comme la répétition 1000 fois du traitement par une boucle « POUR », les élèves ne savent pas comment traiter la variable désignant le nombre à tester. Dans l'algorithme précédent, cette variable est affectée par une instruction de lecture.	Genèse « instrumentale/discursive » de l'itération : concevoir l'automatisation de la répétition d'un traitement sur un ensemble de données.
Les élèves ne comprennent pas d'emblée que, dans la vérification automatique, la variable désignant le nombre à tester est simplement la variable de contrôle de la boucle principale « POUR ». Dans certains algorithmes, deux variables distinctes sont sémantiquement équivalentes : la variable de contrôle de la boucle « POUR » et le nombre à traiter à chaque passage dans la boucle. Les élèves qui produisent ces algorithmes appellent « compteur » cette variable de contrôle. Elle a pour eux une signification particulière.	Genèse instrumentale/sémiotique de l'itération : concevoir la variable de contrôle d'une boucle « POUR » comme une variable itérative et non un simple « compteur ».
Les élèves n'ont pas besoin d'apporter d'autres aménagements à l'algorithme obtenu lors de la l'« ingénierie-Guy » (décomposition, tri, recomposition).	Genèse instrumentale/discursive de la « modularité » : les élèves conçoivent le traitement de chaque nombre comme une « boîte noire ». Les élèves n'ont pourtant pas à leur disposition les éléments sémiotiques (procédures) qui leur permettraient d'exprimer cette modularité.

Tableau 12

b) Phase 2 : Justifier qu'il suffit de vérifier la conjecture que sur six nombres entiers

Comme nous l'avions évoqué précédemment, lors de cette deuxième phase, les élèves doivent déterminer les six nombres entiers qui sont suffisants pour justifier que la conjecture est exacte. Pour cela les élèves utilisent le papier et le crayon et se retrouvent ainsi à raisonner dans un *ETM*. Cependant, bien que le travail se fasse dans un *ETM*, l'*ETA* reste « en toile de fond » sur deux aspects : c'est lui qui justifie de mener un travail en mathématiques, et c'est lui qui sert de référence en ce qui concerne les représentations des nombres. Dans la colonne *Espaces de Travail* (Tableau n° 13), nous montrons comment ces genèses se développent et interagissent.

Observations	***Espaces de travail***
Dans l'exploitation de l'article sur le *Rubik's Cube*, les élèves donnent du sens à la lecture d'un article ne semblant pas avoir dans un	Genèse discursive de la preuve par vérification automatisée dans l'*ETA* :

premier temps un rapport direct avec le problème à résoudre.	situer le problème à résoudre dans une classe plus générale.
Un binôme observe que les nombres obtenus par permutation de 3 chiffres vont nécessairement conduire au même résultat ce qui pourrait aboutir à diminuer le nombre de vérification. Cette observation n'est pas exploitée.	Genèse discursive : • dans l'*ETM* : les nombres obtenus par permutation des chiffres donnent le même résultat. • dans l'*ETA* : le parcours pourrait par exemple être construit sur l'ensemble des entiers dont les chiffres vont croissant. Implicitement, pour les élèves et l'enseignant, c'est trop difficile à mettre en œuvre. Nous avons ainsi une genèse « instrumentale/discursive » non aboutie dans l'*ETA*.
Les élèves réutilisent le principe de décomposition-recomposition vu lors de l'« ingénierie-Guy » et de la phase 1 d'un nombre entier dans le système décimal.	Genève « sémiotique/discursive » par transfert de l'*ETA* vers l'*ETM*. Les élèves transfèrent le principe de la « décomposition/recomposition » et les variables qui permettent de l'exprimer.
Les manipulations sur les nombres obtenus font ensuite intervenir des propriétés issues de l'algèbre élémentaire comme la factorisation, ainsi que des propriétés issues de la *théorie élémentaire des nombres* comme la multiplicité : *être multiple de 99*.	Le raisonnement met en jeu des calculs et propriétés mathématiques. Il s'agit donc d'une genèse « sémiotique/discursive » dans l'*ETM*. Elle est dirigée en arrière-plan par une motivation issue de l'*ETA* : diminuer le nombre de vérifications.
Une partie seulement des élèves considère seulement les 6 premiers multiples de 99.	Il s'agit de la genèse discursive dans l'*ETM* concernant les nombres obtenus par permutation. Elle n'est pas aboutie chez tous les élèves.

Tableau 13

c) Phase 3 : Modifier l'algorithme pour une exécution sur une liste d'entiers non consécutifs

Dans cette dernière phase, nous pouvons observer un conflit intéressant entre deux conceptions du parcours de l'ensemble des 6 nombres. En effet, les élèves peuvent le voir comme un ensemble fini de multiples de 99 (voir phase 2), ou comme une liste d'entiers sans propriété particulière. Cependant, au vu du choix des modifications apportées par les élèves à leurs algorithmes, nous pouvons observer que la création d'une liste constituée de ces six nombres entiers sans propriété particulière est retenue par une petite minorité d'élèves, même parmi ceux qui utilisent des variables de type « LISTE ».

Ainsi, la première conception basée sur « la multiplicité », est retenue par la grande majorité des élèves, même chez ceux qui ont mis en place une nouvelle variable de type « LISTE ». Il est à noter que cette procédure n'a pas été envisagée dans l'analyse *a priori*. Notons qu'elle articule *ETM* et *ETA*, alors que la procédure de l'analyse *a priori* définissant une « LISTE » de nombre entiers sans propriété particulière, privilégie l'*ETA*.

Nous proposons alors d'analyser ces observations en termes de genèses suivant les *ETA* et *ETM* utilisés. Cette analyse est présentée dans le tableau (Tableau n° 14) qui suit.

Phase 3	**ETA**
Procédure papier crayon ou exécution 6 fois de l'algorithme.	Ces élèves considèrent que le petit nombre de cas ne justifie la construction d'une itération. Ils ne sont pas entrés dans la problématique de « réduction des cas » transposée du *Rubik's Cube*. (Nous avons une genèse « Instrumentale-Discursive » dans l'*ETA* non aboutie.) Par ailleurs, ils rencontrent des difficultés à mettre en place une itération sur des entiers non consécutifs. (Nous avons une genèse « Instrumentale-Sémiotique » dans l'*ETA* non aboutie.)
Parcours des 6 nombres comme multiples de 99.	Compréhension de l'ensemble des nombres trouvé à la phase 2 comme multiples successifs de 99 dans l'*ETM*. Utilisation du compteur de boucle pour parcourir cet ensemble. (Nous avons une genèse « Instrumentale-Sémiotique » dans l'*ETA*.)
Mise en place d'une variable « LISTE » peuplée *a priori* pour le parcours des 6 nombres.	Compréhension de l'ensemble des nombres à tester comme quelconque transposée du *Rubik's Cube*. (Nous avons une genèse « Instrumentale-Discursive » dans l'*ETA*.) Mise en place d'une initialisation et d'un parcours adéquat. (Nous avons une genèse « Instrumentale-Sémiotique » dans l'*ETA*.)

Tableau 14

6. Conclusion du chapitre 2

Un travail en algorithmique, centré sur la *preuve* mettant en jeu des outils mathématiques et algorithmique, témoigne que suivant le cas, différentes genèses peuvent apparaître tant dans un *ETM* que dans un *ETA*.

La lecture de l'article du *Rubik's Cube* montre aussi que des élèves ayant un certain niveau de connaissances dans l'abstraction et le raisonnement en mathématique, ont pu donner du sens à cette lecture en transposant l'idée centrale de la problématique de « réduction des cas » à l'idée d'une preuve « explicative » sur un ensemble de nombre à tester. Nous pouvons penser qu'une telle approche peut favoriser des articulations entre *ETM* et *ETA*.

Les articulations *Espaces de Travail Algorithmique* et *Espaces de Travail Mathématiques spécifiques* participent à l'élaboration de la « pensée mathématique » chez l'élève car bien que le travail autour de la preuve se fasse pour l'essentiel dans un *ETM* avec la maitrise de nombres multiples de 99 et des outils algébriques, l'*ETA* joue un rôle important dans cette approche de la preuve. En effet, les genèses instrumentales et sémiotiques de l'*ETA* restent continuellement « en toile de fond » sur les deux aspects que nous avons observés au cours de l'expérimentation. Dans le cadre d'articulations *ETM/ETA*, l'élève comprend qu'il mène un travail en mathématiques, et que l'*ETA* va servir de référence en ce qui concerne les représentations sémiotiques (au sens de Duval) du nombre entier. Le nombre entier n'est pas un objet réel ou physique. Pour le manipuler, l'élève doit passer par des représentations mentales et sémiotiques.

Pour conclure, l'observation faite avec cette première ingénierie sur un travail se situant autour d'*ETA* et *ETT* spécifique à la *théorie élémentaire des nombres*, met en valeur les représentations mentales d'un objet mathématique comme le « nombre entier ». Ce sont des constructions mentales faites dans un contexte particulier et à des fins spécifiques. Ces représentations correspondent aux conceptions de l'élève sur l'objet « nombre entier », mais aussi sur une situation et ce qui lui est associé. Ainsi, les représentations sémiotiques sont définies comme des productions constituées par l'emploi de signes, d'écritures appartenant à un système de représentations qui a ses propres contraintes de signifiance et de fonctionnement que ce soit dans le domaine des mathématiques ou le domaine de

l'algorithmique. L'élève approfondit ainsi la prise de conscience que l'objet mathématique « nombre entier » peut avoir plusieurs représentations sémiotiques.

Chapitre 3 : Vers une ingénierie didactique autour de l'*Algorithme de dichotomie*

Comme nous l'écrivions dans le premier chapitre de cette partie, nous nous intéressons ici au domaine de l'Analyse enseignée au lycée. En effet, l'enseignement de ce domaine se retrouve à tous les niveaux de la scolarité des trois années du lycée.

1. Introduction

Notre proposons une ingénierie en lien avec le concept de dichotomie que nous mettons en place dans des classes de Seconde et des classes du cycle terminal scientifique. Afin de faciliter le travail de l'élève et tenant compte des compétences acquises de la part de l'élève suivant son niveau scolaire, nous souhaitons construire et expérimentée une ingénierie *globale* que nous découpons en deux parties.

La première partie est centrée sur une approche autour d'un jeu de la méthode de dichotomie. Le jeu consiste à mettre les élèves en binôme afin qu'ils puissent déterminer une stratégie « rapide » et « gagnante » dans le cadre de la recherche d'un nombre entier « secret » compris entre deux nombres entiers donnés. Cette stratégie doit être interprétée par les élèves comme une mise en action de la méthode de dichotomie dans un cas « discret ». C'est pour cela, que nous souhaitons appeler cette première partie de notre ingénierie : « *dichotomie discrète* ». Elle fait l'objet du chapitre 4 qui suit.

Lors de la deuxième partie de notre ingénierie, les élèves doivent exploiter le travail algorithmique fait lors de l'expérimentation du jeu sur la structure algorithmique mise en place dans le cas « discret ». En effet, nous attendons que les élèves puissent interpréter dans un premier temps les résultats possibles donnés par l'utilisation de l'*algorithme de dichotomie* appliquée à des fonctions, entre autres dans le cadre de la recherche d'une valeur approchée d'une solution d'une équation algébrique. Puis dans un second temps, plus spécialement pour les élèves de Terminale, nous attendons qu'ils puissent proposer une preuve du *théorème des valeurs intermédiaires (*TVI) à l'aide de la dichotomie. Sachant que cette deuxième partie s'inscrit sur un travail en lien avec les fonctions définies sur R ou un intervalle de R, nous souhaitons nommer cette partie de notre ingénierie : « *dichotomie continue* ». Elle fait l'objet du chapitre 5 de cette deuxième partie du manuscrit.

Bien que nous revenions plus loin sur nos choix retenues pour cette ingénierie sur la dichotomie, nous souhaitons présenter rapidement dans cette introduction l'origine de ce choix d'organiser un travail sur la dichotomie, afin de faciliter la lecture de la présentation de notre expérimentation. En effet, nous sommes d'abord partis du fait que certains documents ressources, voire manuels, proposent des « activités » autour d'une approche « discrète » de la dichotomie. Nous pensons aussi, que cela peut permettre à l'élève de prendre connaissance, à travers une approche ludique, du concept de la dichotomie.

Plus globalement, nous tenons à rappeler aussi que l'intérêt que nous portons à l'« *algorithme de dichotomie* » vient des liens qui peuvent exister entre la version « continue » de l'algorithme et les notions d'analyse enseignées au lycée, en particulier dans les classes scientifiques. En effet, la « *dichotomie continu* » répond à un problème classique posé dans le domaine de l'analyse : l'approximation des valeurs des antécédents par une fonction donnée, notamment la recherche de solutions approchées d'une équation numérique du type $f(x) = 0$ (*), où f est une fonction numérique à variable réelle. En effet, bien qu'il y ait plusieurs méthodes pour résoudre une telle équation, comme balayage, sécante, tangente[54], …, nous choisissons la dichotomie, car parmi ces méthodes, elle permet de mettre en jeu une notion d'analyse centrale dans les classes scientifiques, en particulier au niveau de la Terminale, des lycée français : la *continuité*. De plus, comme nous le verrons au cours de l'ingénierie décrite dans le chapitre 4, la méthode *de balayage*, bien qu'elle soit facile à mettre en œuvre à la main, n'est pas aisée à systématiser dans un *algorithme*. De même, la *méthode de la tangente*, suppose que l'élève ait déjà calculé la dérivée f' de la fonction f de l'équation (*) et qu'il connaisse, pour un x_i donné, l'équation de la tangente à la courbe de f en x_i. Ces connaissances n'étant étudiées qu'à partir du milieu de l'année scolaire de Première Scientifique, elles ne permettent pas d'étudier l'évolution du savoir chez l'élève au cours des trois années du lycée sur une même méthode. Quant aux autres méthodes du point fixe, elles supposent des notions abordées seulement au cours de la Terminale Scientifique pour leurs justifications. En ce qui concerne la *méthode de la sécante* elle n'a pas été retenue car elle n'apparaît pas explicitement dans le programme du lycée, contrairement à la dichotomie qui est mentionnée dans le programme de Seconde.

[54] On dit aussi « *Méthode de Newton* ».

Pour conclure cette introduction et afin de rendre plus aisé la lecture et la compréhension de notre ingénierie *globale* sur la dichotomie, nous rappelons que dans le cadre d'une analyse en lien avec l'étude des articulations possibles entre *ETA* et $ETM_{analyse}$, nous avons fait le choix de découper la présentation de cette analyse en deux chapitres indépendants associés respectivement à la « *dichotomie discrète* » et à la « *dichotomie continue* ».

2. Analyse des programmes de la Seconde et des classes du cycle terminal scientifique en lien avec notre recherche

Nous faisons le choix de présenter une brève analyse des compétences sur les fonctions et les suites ainsi que sur les propriétés qui leur sont associées en fonction des niveaux programmes des niveaux scolaires retenus. Nous souhaitons aussi compléter cette présentation par une description des attentes des auteurs des programmes sur les tâches algorithmiques dans le domaine de l'Analyse et plus particulièrement sur la dichotomie.

2.1 Les fonctions

En Seconde, l'enseignement de l'Analyse a pour nom « Fonctions ». On attend de l'élève qu'il soit capable d'étudier[55] :

- *un problème se ramenant à une équation du type f(x) = k et de le résoudre dans le cas où la fonction est donnée (définie par une courbe, un tableau de données, une formule) et aussi lorsque toute autonomie est laissée pour associer au problème divers aspects d'une fonction ;*
- *un problème d'optimisation ou un problème du type f(x) > k et de le résoudre, selon les cas, en exploitant les potentialités de logiciels, graphiquement ou algébriquement, toute autonomie pouvant être laissée pour associer au problème une fonction.*

[…] *Par ailleurs, la résolution de problèmes vise aussi à progresser dans la maîtrise du calcul algébrique et à approfondir la connaissance des différents types de nombres, en particulier pour la distinction d'un nombre de ses valeurs approchées.*[56]

De plus, l'élève doit prendre conscience que des *dessins peuvent suffire pour répondre de*

[55] Extrait du BO n° 30 du 23 juillet 2009
[56] Ibid.

façon satisfaisante à un problème concret mais qu'ils ne suffisent pas à démontrer des propriétés de la fonction.[57]

A ce niveau scolaire, l'élève approfondit ses connaissances sur les fonctions de référence vues au Collège : *fonctions linéaire et affine*. Celles-ci sont complétées par l'introduction des *fonctions carrée et inverse*, ainsi qu'une première approche des *fonctions polynômes de degré 2 et des fonctions homographiques*.

Dans les classes du cycle scientifique terminal, l'enseignement des fonctions et des concepts qui lui sont rattachées prend explicitement le nom d'« Analyse ». L'enseignement de l'Analyse s'inscrit dans le cadre de la résolution de problème et ceci dans un esprit de continuité et d'approfondissement des apprentissages faits dans la classe antérieure. L'objectif attendu est de doter les élèves du cycle scientifique d'outils mathématiques *leur permettant de traiter des problèmes relevant de la modélisation de phénomènes continus ou discrets*.[58] En Première Scientifique, les élèves doivent consolider *l'ensemble des fonctions mobilisables.*[59] De même, en Terminale Scientifique, on attend de l'élève qu'il consolide et enrichisse des *notions relatives aux suites et aux fonctions*[60] afin qu'il puisse *étudier un plus grand nombre de phénomènes discrets ou continus*[61].

Au cours du cycle scientifique terminal, sont introduits de nouvelles fonctions de référence : les *fonctions racine carrée et valeur absolue* en Première, et les *fonctions exponentielle, logarithme, sinus et cosinus* en Terminale.

Au niveau de la Première un nouvel outil est introduit : la *fonction dérivée*. Cette introduction s'accompagne d'une approche intuitive de la notion de limite finie en un point. En Terminale, les compétences dans le domaine de l'Analyse sont complétées par l'introduction de la *continuité sur un intervalle* et le *théorème de valeurs intermédiaires* (TVI) ainsi que de son corollaire (le *théorème de la bijection* (TB)).

De plus, en Terminale, une étude approfondie du concept de limite d'une fonction et de

[57] Ibid.
[58] Extrait du BO spécial n° 9 du 30 septembre 2010
[59] Ibid.
[60] Extrait du BO spécial n° 8 du 13 octobre 2011
[61] Ibid.

l'étude du comportement asymptotique d'une fonction fait partie des compétences attendues chez l'élève dans ce domaine des mathématiques.

Enfin, au niveau de la classe de Terminale *s'ajoute le nouveau concept d'intégration qui, bien que modestement abordé et développé, demeure un concept fondamental de l'analyse*[62].

2.2 Les suites

Seulement à partir de la classe de Première, l'enseignement de l'Analyse introduit le concept de *Suite numérique*. En effet, à ce niveau scolaire, *l'étude de phénomènes discrets fournit un moyen d'introduire les suites et leur génération en s'appuyant sur des registres différents (algébrique, graphique, numérique, géométrique) et en faisant largement appel à des logiciels. Les interrogations sur leur comportement amènent à une première approche de la notion de limite qui* [est] *développée en classe de Terminale*[63]. Les auteurs du programme signalent que l'étude des suites se prête tout particulièrement à la mise en place d'activités algorithmiques.

Au niveau de la classe de Terminale, la notion de limite de suite fait l'objet d'une étude approfondie. Cette étude approfondie doit ainsi permettre de préparer la présentation des limites de fonctions.

2.3 L'algorithmique dans le domaine de l'Analyse

De façon générale, les algorithmes considérés en analyse concernent des méthodes d'approximation issues du domaine de l'*analyse numérique* comme la résolution d'équations, la détermination d'un maximum local d'une fonction, l'intégration numérique, les équations différentielles, … En effet, l'*analyse numérique* s'intéresse à ces méthodes, de façon à en caractériser l'efficacité, la complexité (ou coût) et leurs champs d'application.

Nous rappelons qu'au collège, les élèves ont rencontré quelques algorithmes (*algorithmes opératoires, algorithme des différences, algorithme d'Euclide, algorithmes de construction en géométrie*). Au lycée, dès la classe de Seconde, dans le domaine de l'Analyse, en parallèle de la formalisation des notions sur les fonctions, il est demandé aux élèves une formalisation en langage naturel d'algorithmes, propre à donner lieu à une expression en langage

[62] Ibid.

[63] Extrait du BO spécial n° 9 du 30 septembre 2010

algorithmique afin d'être implémentable dans un environnement numérique, que celui-ci soit une calculatrice ou un logiciel algorithmique.

Ces tâches de formalisation sont poursuivies tout au long du lycée. En Seconde, comme nous l'avons vu précédemment, les élèves vont travailler sur des problèmes se ramenant à une équation du type *f (x) = k*. Pour résoudre ces problèmes, les élèves ont à leur disposition soit la représentation graphique de la fonction, soit un tableau de valeurs associé à la fonction, soit l'expression de cette fonction. Pour résoudre ces problèmes, on attend aussi des élèves qu'ils élaborent des algorithmes[64] afin qu'ils identifient *le calcul ou le traitement qui est à répéter* et qu'ils l'automatisent *un nombre donné de fois ou un nombre de fois soumis à un test*.

Dans le cas de la recherche d'une valeur approchée d'une solution d'une équation, le programme[65] de Seconde indique le fait que les élèves doivent acquérir des compétences spécifiques à la *résolution graphique et algébrique d'équations*. Dans le cadre de ces compétences, les capacités attendues sont explicitement données. Les élèves doivent pouvoir *encadrer une racine d'une équation grâce à un algorithme de dichotomie* (cf. tableau ci-dessous).

Contenus	**Capacités attendues**	**Commentaires**
[…]	[…]	[…]
Équations Résolution graphique et algébrique d'équations.	• Mettre un problème en équation. • Résoudre une équation se ramenant au premier degré. ◊ Encadrer une racine d'une équation grâce à un algorithme de dichotomie.	Pour un même problème, combiner résolution graphique et contrôle algébrique. Utiliser, en particulier, les représentations graphiques données sur écran par une calculatrice, un logiciel.

Tableau 15

Au niveau du cycle scientifique terminal, le programme précise que les élève doivent se procurer *un bagage mathématique solide*, en particulier si ces derniers désirent *s'engager dans des études supérieures scientifiques*. En effet, ils doivent être formés *à la pratique d'une démarche scientifique* […] *en renforçant leur goût pour des activités de recherche*. […] *Outre*

[64] Extrait des « Ressources pour la classe de seconde »

[65] Extrait de la partie « *Fonctions* » du programme de Seconde

l'apport de nouvelles connaissances, le programme vise le développement des compétences suivantes : – mettre en œuvre une recherche de façon autonome ; – mener des raisonnements et utiliser le *langage mathématiques* ; *– avoir une attitude critique vis-à-vis des résultats obtenus* ; *– communiquer à l'oral et à l'écrit*. […] *L'utilisation de logiciels, d'outils de visualisation et de simulation, de calcul (formel ou scientifique) et de programmation* doit permettre de favoriser chez les élèves *une démarche d'investigation*. *En particulier, lors de la résolution de problèmes, l'utilisation de logiciels* […] *peut limiter le temps consacré à des calculs très techniques afin de se concentrer sur la mise en place de raisonnements*. Les élèves doivent chercher, expérimenter, modéliser, à l'aide d'outils nécessitant des travaux dans des environnements numériques. Les élèves doivent mettre en œuvre des algorithmes. Cependant, dès la Première Scientifique, les programmes précisent qu'il est aussi attendu des élèves qu'ils raisonnent, démontrent et trouvent des résultats partiels qu'ils puissent mettre en perspective.

Comme nous le signalions plus haut, l'*étude de phénomènes discrets* à travers des problèmes mettant en place des suites numériques peut fournir un Espace de Travail qui *se prête tout particulièrement à la mise en place d'activités algorithmiques*. Ainsi, l*e tableur, les logiciels […] de calcul* et algorithmique peuvent être *des outils adaptés à l'étude des suites, en particulier* au niveau de la classe de Première, *pour l'approche expérimentale de la notion de limite* où on n'attend pas que les élèves aient à leur disposition une *définition formelle de la limite*. Ce point est étudié au niveau de la classe de Terminale.

Au niveau de la classe de Première, il est attendu que les élèves modélisent et étudient des situations à l'aide de suites. Pour cela, ils doivent construire des algorithmes permettant d'obtenir une liste de termes d'une suite, ainsi que le calcul d'un terme de rang donné. Selon le programme, la mise en œuvre d'algorithmes doit être l'occasion d'étudier des suites générées par une relation de récurrence. Les élèves peuvent aussi utiliser un algorithme, voire un tableur, *pour traiter des problèmes de comparaison d'évolutions et de seuil*.

Au niveau de la classe de Terminale, *l'acquisition d'automatismes de calcul demeure un objectif*. Les élèves peuvent recourir si besoin à des logiciels *de calcul formel ou scientifique* dans le cadre de la résolution de problèmes. En revanche, il n'est pas fait mention de l'usage d'un environnement de programmation spécifique. Les « activités » de type algorithmique sont plutôt vues comme des « applications numériques » (recherche de résultats approchés) plutôt que de la résolution de problème.

A ce niveau scolaire, *dans le cas d'une limite infinie, étant donnés une suite croissante (u_n) et un nombre réel A,* les élèves doivent savoir *déterminer à l'aide d'un algorithme un rang à partir duquel u_n est supérieur à A*. De plus, selon le programme, des activités algorithmiques peuvent réalisés réalisées dans le cadre de la recherche de solutions de l'équation $f(x) = k$, où k est une constante réelle.

Tout au long des trois années de lycée, les élèves conçoivent et mettre en œuvre des algorithmes. Lors d'activités algorithmiques, les élèves vont *décrire certains algorithmes en langage naturel ou dans un langage symbolique* et en implémenter quelques-uns sur calculatrice ou sur ordinateur muni d'un logiciel adapté. Ils doivent aussi *interpréter des algorithmes plus complexes*. *Aucun langage, aucun logiciel n'est imposé*. Ils doivent apprendre à maitriser les *instructions élémentaires* : *affectation, calcul, entrée, sortie*, ainsi que les structures algorithmiques : *boucle et itérateur*, *instruction conditionnelle*.

2.4 Vers une preuve algorithmique du *théorème des valeurs intermédiaires* en Terminale scientifique

Depuis la classe de Première scientifique, les élèves acquièrent de nouveaux savoirs sur les suites et en particulier sur les problématiques de *convergence*. Nous faisons alors l'hypothèse que des preuves de terminaison d'algorithmes d'approximation d'une solution d'une équation peuvent faire aussi l'objet de tâches de recherche dans le cadre de travaux menés en classe. Par exemple, une démonstration du *théorème des valeurs intermédiaires* peut notamment être mise en place dans le cadre d'un travail de recherche mené par les élèves, à partir de l'*algorithme de dichotomie* et de considérations sur les suites. Cependant, le programme n'adopte pas totalement cette démarche scientifique puisqu'il indique que *le théorème des valeurs intermédiaires est admis*. En effet, il est nécessaire de compléter les compétences institutionnelles sur les suites par des apports supplémentaires autour des suites adjacentes et des propriétés qui leur sont associées. Cependant, nous pensons qu'un tel travail va s'avérer être bénéfique quant aux analyses en termes d'articulations d'*Espaces de Travail Mathématique* et *Algorithmique*, ainsi que l'étude des genèses qui vont y être associées. Ceci va faire l'objet du chapitre 5 de cette seconde partie.

3. Etude de manuels scolaires et d'autres ressources

Pour compléter la présentation et l'analyse des programmes de Seconde et du cycle terminal scientifique sur le domaine de l'Analyse, nous proposons une étude sur la façon dont

certains manuels scolaires et documents ressources proposés par l'institution avec *Eduscol* et les IREM abordent des activités algorithmiques sur la dichotomie dans le domaine de l'analyse.

3.1 Au niveau de la Seconde

Nous faisons le choix de ne sélectionner qu'un nombre limité de manuels scolaires en mathématiques pour notre étude : *Nathan* avec la collection *Hyperbole* et *Hachette* avec la collection *Maths-repères*. En ce qui concerne une introduction et des approches de l'*algorithme de dichotomie*, bien que les manuels scolaires de Seconde puissent paraître assez dissemblables, nous considérons qu'il s'agit d'un éventail assez large, nous permettant de situer nos choix pour l'ingénierie parmi les travaux qui peuvent être proposés aux élèves dans le contexte de notre recherche.

Pour ces deux manuels, notre objectif est d'étudier le type d'exercices proposés sur l'*algorithme de dichotomie* et son introduction tant dans la partie cours que dans celle des activités proposées. Pour cela, nous nous limitons à une présentation du cours et à un repérage des tâches liées à notre problématique. Nous faisons aussi le point sur la façon dont d'autres ressources, telles que celles disponibles sur *Eduscol* ou élaborées dans les IREM considèrent l'*algorithme de dichotomie* à ce niveau scolaire.

3.1.1 Le manuel Hyperbole (Edition 2010)

3.1.1.1. Présentation rapide du manuel

Ce manuel est constitué de treize chapitres, augmentés d'une première partie que nous nommerons *chapitre 0* : « *Premiers pas en algorithmique* » comportant des activités d'approche : – *automatiser un calcul et dégager les étapes de la rédaction d'un algorithme simple* ; – *introduction à la notion de variable et à l'instruction d'affectation* ; – *découverte de la structure alternative (avec ou sans « sinon »)* ; – *dégager la structure itérative dans le cas où le nombre d'itérations est connu à l'avance* ; – *aborder la structure itérative dans le cas où le nombre d'itérations dépend d'une condition* ; et des exercices abordables dès le début de la Seconde, afin de permettre aux élèves débutants de comprendre le but de l'algorithmique et de présenter ses principaux outils : – *les instructions d'entrée-sortie, l'affectation, les variables* ; – *la structure alternative ou test* ; – *la structure itérative ou boucle*. Ce *chapitre 0* se termine par une partie intitulée « *Méthodes* » proposant la donnée et la résolution de six exercices ayant chacun un objectif précis et signalé par les auteurs du manuel comme : –

comprendre un algorithme écrit en langage naturel ou semi-formalisé ; *– comprendre le déroulement d'un algorithme et suivre l'évolution du contenu des variables* ; *– écrire puis compléter un algorithme, puis coder le programme correspondant dans un langage algorithmique afin de l'implémenter dans une machine* ; *– lire et comprendre un programme écrit dans un langage algorithmique puis le modifier afin de répondre à une problématique donnée* ; *– écrire un algorithme pour résoudre un problème donné et le transcrire en langage machine*.

Les autres chapitres sont tous découpés sur le schéma suivant : – *Vérifier les acquis* suivi de *deux activités* permettant d'introduire les nouvelles notions du cours ; – *Le cours* qui présentent les définitions et les propriétés en jeu, illustrées de nombreux exemples, puis des méthodes données sous forme d'exercices résolus afin de permettre de dégager des méthodes de résolution de problèmes en lien avec le cours étudié ; – Les *exercices* organisés en trois rubriques permettant de soigner la progressivité et de diversifier l'activité mathématique. En effet, afin de créer des automatismes chez l'élève une première série d'exercices dits *de bases* est proposée. Elle est suivie d'une deuxième série d'exercices dits *d'entraînement* permettant de développer chez l'élève des compétences. Cette deuxième série se termine par une double page intitulée « *Se préparer au contrôle* », proposant un QCM sur la leçon, un « Vrai – Faux » et enfin des exercices « incontournables » avec divers conseils de révisions. S'en suit après une dernière série d'exercices dits d'*approfondissement* pour que l'élève puisse aller plus loin dans son apprentissage des nouveaux concepts vus dans le chapitre. Conformément aux attentes du programme, dans chaque chapitre sont aussi proposés de nombreux exercices d'algorithmique et de logique.

3.1.1.2. Trois exercices autour de la résolution d'une équation

Ce manuel est constitué d'un chapitre proposant un travail spécifique sur la *dichotomie*. En effet, dans le chapitre intitulé « *Fonctions, expressions algébriques et problèmes* », bien que nous n'observons aucune information dans la partie cours sur la résolution approchée de valeurs d'antécédents ou la détermination de solutions approchées d'équations, nous trouvons dans la partie « *Raisonner, démontrer* » des exercices d'entraînement, un unique exercice (fig. 50) faisant explicitement référence à l'*algorithme de dichotomie*. Il s'agit toutefois de l'algorithme « continu » au sens vu dans l'introduction de ce chapitre. De plus, cet exercice est le seul exercice du chapitre proposant un travail sur la problématique :

résolution approchée d'une équation de la forme f(x) = 0, sachant que cette équation admet une unique solution x_0 dans un intervalle donné [*a* ; *b*].

Algorithmique

43 Un algorithme de dichotomie

f est la fonction définie sur un intervalle $[a;b]$ représentée ci-contre. On suppose que l'équation $f(x) = 0$ admet une solution unique x_0 dans l'intervalle $[a;b]$.
On considère l'algorithme suivant :

Entrées
Saisir
a, *b* : bornes de l'intervalle de définition
f : fonction étudiée
N : entier naturel, N ⩾ 1
Traitement
Pour *k* de 1 jusqu'à N
  m prend la valeur $\frac{a+b}{2}$
  Si $f(m)$ et $f(a)$ sont de même signe alors
    a prend la valeur *m*
  sinon
    b prend la valeur *m*
  FinSi
FinPour
Sorties
Afficher *a*, *b*

a) On applique cet algorithme à la fonction *f* définie sur l'intervalle $[0;1]$ par $f(x) = x^3 + 2x - 2$.
Prendre N = 4 et compléter le tableau suivant.

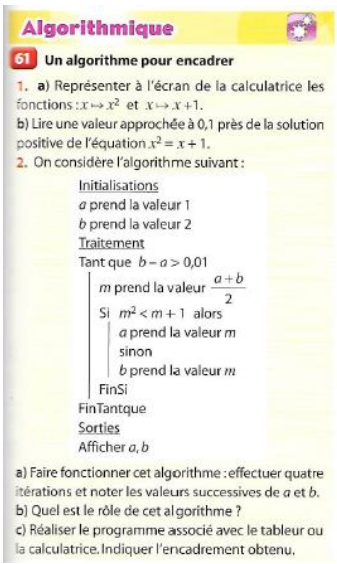
k		1	2	3	4
m		0,5			
a	0	0,5			
b	1	1			

b) Quel est le rôle de cet algorithme ? Expliquer en particulier la fonction de la variable N.
c) Traduire cet algorithme dans un langage de programmation et tester le programme obtenu.

Figure 50 (Extrait d'un exercice du manuel Hyperbole de 2nde : Un algorithme de dichotomie)

Comme nous pouvons l'observer, un algorithme, écrit dans un langage semi-formalisé, est donné aux élèves. On leur demande d'appliquer cet algorithme à une fonction *f*, dont on connait son expression, et qui est définie sur un intervalle donné. Pour cela, l'élève doit remplir un tableau de valeurs puis répondre à différentes questions sur le rôle de l'algorithme et expliquer en particulier le rôle d'une variable donnée. Ensuite, il doit « traduire » cet algorithme dans un langage de programmation et l'implémenter dans un environnement numérique afin de le tester.

Cependant, dans le chapitre « *Fonctions de référence et problèmes* » (fig. 51) et dans la partie finissant les chapitres sur les fonctions « *Algorithmique et fonctions* » (fig. 52), deux autres exercices en lien avec la recherche d'une valeur approchée d'une solution d'une équation sont aussi proposés. Il s'agit de variantes de l'algorithme « continu », mais la méthode de dichotomie n'est pas explicitement mentionnée.



Algorithmique

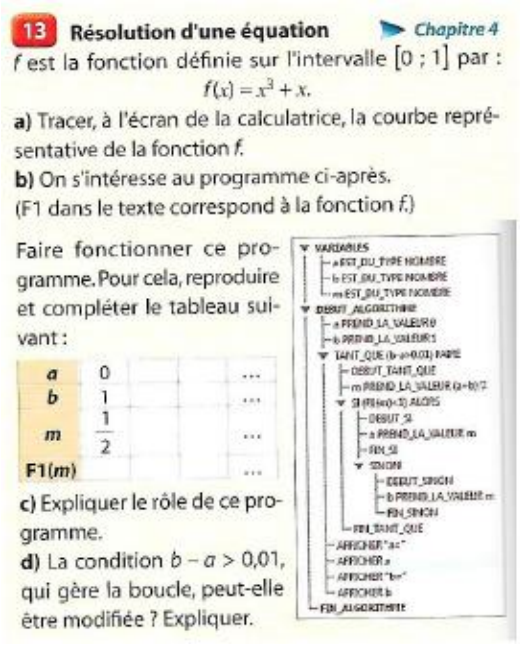
61 Un algorithme pour encadrer

1. **a)** Représenter à l'écran de la calculatrice les fonctions : $x \mapsto x^2$ et $x \mapsto x+1$.
b) Lire une valeur approchée à 0,1 près de la solution positive de l'équation $x^2 = x + 1$.
2. On considère l'algorithme suivant :

Initialisations
a prend la valeur 1
b prend la valeur 2
Traitement
Tant que $b - a > 0{,}01$
  m prend la valeur $\frac{a+b}{2}$
  Si $m^2 < m + 1$ alors
    a prend la valeur *m*
  sinon
    b prend la valeur *m*
  FinSi
FinTantque
Sorties
Afficher *a*, *b*

a) Faire fonctionner cet algorithme : effectuer quatre itérations et noter les valeurs successives de *a* et *b*.
b) Quel est le rôle de cet algorithme ?
c) Réaliser le programme associé avec le tableur ou la calculatrice. Indiquer l'encadrement obtenu.

Figure 51 (Extrait d'un exercice du manuel Hyperbole de 2nde : Un algorithme pour encadrer)



13 **Résolution d'une équation** *Chapitre 4*

f est la fonction définie sur l'intervalle [0 ; 1] par :

$f(x) = x^3 + x$.

a) Tracer, à l'écran de la calculatrice, la courbe représentative de la fonction *f*.

b) On s'intéresse au programme ci-après.

(F1 dans le texte correspond à la fonction *f*.)

Faire fonctionner ce programme. Pour cela, reproduire et compléter le tableau suivant :

a	0			...
b	1			...
m	$\frac{1}{2}$			...
F1(*m*)				...

c) Expliquer le rôle de ce programme.

d) La condition $b - a > 0{,}01$, qui gère la boucle, peut-elle être modifiée ? Expliquer.

Figure 52 (Extrait d'un exercice du manuel Hyperbole de 2nde : Résolution d'une équation)

Nous observons que dans le troisième exercice (fig. 52), l'algorithme donné est écrit en langage algorithmique (ici *AlgoBox*) contrairement aux deux autres exercices. De plus, nous constatons aussi que ces trois exercices sont très proches dans les attentes des auteurs : – rôle de l'algorithme ; – implémentation de l'algorithme sur machine afin de le tester. Cependant, un des exercices proposent une structure itérative du type « *Pour … Faire* » tandis que pour les deux autres exercices, les élèves travaillent sur une structure itérative « *TantQue… Faire* », où le nombre d'itérations dépend d'une condition.

3.1.1.3. Approche de ces trois exercices à l'aide du cadre théorique sur les ETA/ETM

Nous souhaitons revenir sur les différents points concernant les exercices présentés dans la section précédente, ainsi que sur les paradigmes concernés (Tableaux n° 16 et 17).

En effet, nous proposons une présentation de ces exercices en termes de *paradigmes algorithmiques* et d'articulations entre *Espaces de Travail Mathématique* et *Algorithmique* privilégiés qu'ils mettent en jeu. Ces trois exercices ne renvoient pas exactement au même *paradigme algorithmique* bien que l'*Espace de Travail* spécifique soit le même pour les deux premiers exercices (fig. 50 et 51) et légèrement différent pour le troisième (fig. 52).

Thème de l'exercice	Un algorithme à explorer (fig. 50 et 51)
Paradigmes	Tout au long des deux exercices, nous nous situons au niveau 1 des paradigmes de l'algorithmique. En effet, dans ces deux exercices, le travail demandé aux élèves suppose qu'à partir d'une représentation spécifique des objets et des actions élémentaires, ils construisent un traitement permettant de répondre à une

<table>
<tr><td></td><td colspan="2">problématique qui doit être reconnue par les élèves.
- Dans le cas de l'exercice de la fig. 50, l'élève doit choisir le nombre d'itérations qui doit être fait afin d'obtenir les deux bornes de l'intervalle encadrant l'unique solution x_0 sans pour autant qu'il ait conscience que cet intervalle correspond à une amplitude précise. Il n'a pas à construire de formules permettant d'obtenir les bornes des intervalles emboitées. De plus, il doit itérer le traitement.
- Dans le cas de la fig. 51, l'élève doit interpréter les valeurs du premier intervalle d'encadrement, puis déterminer une suite d'intervalles emboités dont le dernier qui doit être affiché par la machine correspond à une amplitude donnée par les auteurs de l'exercice. Pour cela, l'élève doit aussi interpréter la condition « Tant Que $b - a > 0{,}01$ » comme test d'arrêt permettant de sortir de la boucle sans que pour autant il en saisisse totalement sa signification, puisqu'il n'est pas proposé de modifier la valeur du second membre de l'inégalité, contrairement à l'exercice de la figure 52.
De plus, la lecture graphique demande une valeur approchée à 0,1 près de la solution de l'équation donnée, contrairement à l'algorithme qui propose un intervalle d'encadrement correspondant à une amplitude fixée. Comme pour le cas précédent, il n'a pas à construire de formules permettant d'obtenir les bornes des intervalles emboités. De plus, là aussi, il doit itérer le processus.

Cependant, dans le cas de l'exercice de la figure 50, ce traitement reste dans un cadre non axiomatique tant sur le plan algorithmique que mathématique (en effet, pour cet aspect il est admis que la fonction admet un unique zéro sur un intervalle donné), ce qui caractérise le niveau 1:
- L'itération s'opère dans le tableur, et utilise la recopie incrémentée de cellules, l'arrêt étant géré par l'utilisateur et non par l'algorithme.
- Il est demandé aux élèves de reconnaitre le « rôle » de l'algorithme et d'expliquer la fonction de la variable informatique N, mais ceci n'est pas mis explicitement en relation avec une propriété de terminaison de l'algorithme.
- Implicitement, cette reconnaissance va conduire à considérer les suites des valeurs des variables *a* et *b* obtenues lors des N itérations. Ainsi, que le rôle de la variable *m* moyenne des valeurs de *a* et *b*.

Dans le cas de l'exercice de la figure 51, le traitement reste aussi dans un cadre non axiomatique d'un point de vue algorithmique, malgré le fait que sur le plan mathématique, il soit demandé aux élèves de représenter les graphiques des deux fonctions dont les expressions respectives sont connues et d'interpréter l'abscisse du point commun à ces deux courbes comme étant la solution de l'équation algébrique obtenue en faisant l'égalité entre les deux fonctions. Dans le domaine de l'Analyse se second exercice se situe donc à la « frontière » des niveaux 1 et 2 des paradigmes de l'Analyse.</td></tr>
<tr><td rowspan="2">Espaces de Travail spécifiques</td><td>ETA</td><td>ETM</td></tr>
<tr><td>Dans l'exercice de la figure 50, l'élève travaille dans un premier temps avec le</td><td>Pour l'exercice de la figure 50, il n'est pas demandé à l'élève de représenter une</td></tr>
</table>

	tableur. Cependant, les formules ne lui sont pas données explicitement. Il doit interpréter l'algorithme proposé afin de déterminer ces formules pour qu'il puisse les implémenter dans le tableur et définir ce qu'elles retournent. Un tableau type est aussi proposé avec une description partielle de chacune des colonnes et lignes utilisées. Le travail attendu dans les deux premières questions (sur trois) fait que l'élève reste ici dans des genèses instrumentale et sémiotique. La question qui porte sur le rôle de l'algorithme et la signification de la variable N permet d'envisager un début de genèse discursive, sans que pour autant l'étude de la terminaison ne soit envisagée. Elle serait compliquée à mettre en place avec un tableur. Dans la dernière question, il est demandé à l'élève de traduire cet algorithme en langage de programmation et de l'implémenter dans un environnement numérique afin de le tester. Ici, l'élève reste encore dans des genèses instrumentale et sémiotique. Dans l'exercice de la figure 51, il n'est plus demandé à l'élève de passer par une première approche autour du tableur permettant de tester l'algorithme. Le travail attendu en algorithmique dans la question 2a) (sur les trois de type algorithmique) fait que l'élève reste ici dans des genèses instrumentale et sémiotique. Cependant, dans la question 2a), on attend que l'élève puisse interpréter le rôle de l'algorithme et par conséquent, il se trouve ici dans des genèses instrumentale et discursive. Dans la dernière question, comme pour l'exercice précédent, il est demandé à l'élève d'implémenter cet algorithme sur	courbe représentative d'une fonction sur un intervalle donné, ni de lire la valeur approchée d'une abscisse ou d'un encadrement du point où la courbe coupe l'axe des abscisse, et correspondant à la solution de l'équation $f(x) = 0$. Ici, le travail ne se fait que dans un ETA. Pour l'exercice de la figure 51, les deux premières questions 1a) et 1b), l'élève doit représenter les courbes associées à deux fonctions dont les expressions sont données afin de lire une valeur approchée de la solution de l'équation obtenue en faisant l'égalité de ces deux fonctions. Cependant, à ce niveau de la scolarité, il n'est pas demander de justifier l'existence et l'unicité de cette solution. L'intervalle d'encadrement est obtenu pour une amplitude donnée dans l'*ETA*. Il semble que le travail attendu par les auteurs dans ces deux questions 1a) et 1b) cantonne les élèves dans une approche d'une genèse discursive dans l'*ETM*. On aurait pu s'attendre à ce que les genèses instrumentale et sémiotique

	un tableur ou de le traduire en langage de programmation afin de l'implémenter dans un environnement numérique pour le tester. Ici, l'élève reste encore dans des genèses instrumentale et sémiotique.	développées dans l'*ETA* soient exploitées pour complémenter cette approche d'une genèse discursive dans l'*ETM*.

Tableau 16

Thème de l'exercice	Un algorithme en langage pseudo-code (fig. 52)	
Paradigmes	Il s'agirait plus d'une tâche située au niveau 2 des paradigmes de l'algorithmique, tels que nous les avons définis au chapitre 2 (cadre théorique) de la partie 1, en rupture avec l'approche de niveau 1 des exercices précédents. Dans ce paradigme de niveau 2, la source de validation se fonde sur la compréhension explicite de deux conditions : (1) « F1(m) < 1 » correspondant à la « nature » de l'équation (en effet, cette condition doit être associée à la résolution d'une équation et non d'une inéquation), c'est-à-dire $f(x) = 1$ sur l'intervalle [0 ; 1], avec F1 représentant la fonction f ; (2) « $b - a > 0{,}01$ » correspondant à l'amplitude de l'intervalle cherché ; le tout dans un « système axiomatique » aussi précis que possible. Mais le problème du choix des « axiomes » utilisés se pose. La relation avec la « réalité » subsiste encore dans ce niveau 2 des paradigmes algorithmiques, dans la mesure où elle s'est constituée pour organiser les connaissances algorithmiques issues ici de problèmes sur les approximations des bornes d'un intervalle encadrant une solution d'une équation algébrique. L'« axiomatisation » proposée est certes une formalisation, mais elle n'est pas formelle car ici la syntaxe n'est pas coupée de la sémantique qui renvoie à la « réalité ». D'où la conservation du qualificatif de « naturelle » pour ce niveau de paradigme. En effet, dans cet exercice, l'algorithmique peut s'exercer moyennant une « axiomatisation partielle », voire des « îlots d'axiomatisation » (Kuzniak, Houdement, 2006). Toutefois, la question correspondant au fonctionnement de l'algorithme permettant de compléter le tableau proposé, renvoie de nouveau l'exercice à un paradigme de niveau 1 de l'algorithmique. En effet, l'élève peut se « contenter » de faire tourner l'algorithme au « papier-crayon ». En revanche, pour compléter le tableau en implémentant l'algorithme dans l'environnement numérique correspondant au langage pseudo-code de l'algorithme donné, l'élève doit procéder à une légère modification de l'algorithme en inscrivant l'affichage des valeurs de a et b dans la boucle « TantQue ». Cependant, nous restons encore au niveau 1 de l'algorithmique.	
Espaces de Travail spécifiques	*ETA*	*ETM*
	Dans cet exercice, l'élève est sollicité par les auteurs à mettre en place un travail dans des *ETA* spécifiques : - la calculatrice pour représenter sur un intervalle donné la fonction dont on connait son expression ; - le « papier-crayon » pour déterminer les bornes des intervalles emboités	L'élève doit donner la représentation graphique d'une fonction sans pour autant expliquer le processus de construction qui reste dans un ETA correspondant à l'écran de la calculatrice. Nous situons cette construction dans des genèses instrumentale et

	d'encadrement de la solution d'une équation que l'élève doit reconnaitre (en l'occurrence $f(x) = 1$), à partir de la transposition d'un algorithme donné dans un langage pseudo-code (*AlgoBox*) ; - le logiciel AlgoBox si l'élève veut compléter le tableau en procédant à une modification de l'algorithme afin que les bornes des intervalles emboités soient toutes affichées et non seulement le dernier intervalle correspondant à la sortie de boucle. Les genèses utilisées ici sont pour l'essentielle de types instrumentale et sémiotique, mais aussi de type discursive. En effet, l'élève doit reconnaitre l'équation à résoudre en interprétant la condition « F1(m) < 1 » (sachant que F1 correspond à la fonction *f*), puis expliquer la condition $b - a > 0{,}01$ en justifiant que celle-ci peut être modifiée.	sémiotique pour l'*ETM*.

Tableau 17

A partir des analyses présentées dans les deux tableaux précédents, nous en déduisons ce que peuvent être les conséquences de ce type d'exercices vis à vis de notre ingénierie (tableau n° 18).

Implication pour notre ingénierie.	Nous constatons que : 1. Pour l'essentiel du travail proposé dans les trois exercices, celui-ci reste au niveau 1 en algorithmique : itération « naturelle », pas de problématique de terminaison,… 2. Cependant, une première approche du niveau 2 en algorithmique peut être observée dans le troisième exercice même si celle-ci peut être vue encore au niveau 1. 3. Les genèses instrumentale et sémiotique dans l'*ETA* ne viennent ni supporter la genèse discursive dans l'*ETM* quand celle-ci existe même que « partiellement » (cf. le deuxième exercice), ni engager une réelle genèse discursive dans l'*ETA*. Nous souhaitons, dans notre ingénierie, élaborer des tâches tirant parti des genèses instrumentale et sémiotique dans l'*ETA* et articulant des genèses discursives dans l'*ETM* et l'*ETA*, autour de la terminaison et de l'efficacité donc vers le niveau 2 de l'algorithmique. Ceci sera présenté dans les chapitres 4 et 5.

Tableau 18

3.1.2 Le manuel Maths-repères (Edition 2010)

3.1.2.1. Présentation rapide du manuel

Le manuel débute par un *cahier* intitulé *« A la découverte des algorithmes et de la calculatrice* ». Ce *cahier* est découpé en trois parties nommées *A la découverte des algorithmes*, *A la découverte de la calculatrice* et *Programme*.

La partie « *A la découverte des algorithmes* » est constituée de neuf fiches : (1) *Qu'est-ce qu'un algorithme ?* ; (2) *Vocabulaire des algorithmes* ; (3) *Logiciels* ; (4) *Entrées et sorties d'un algorithme* ; (5) « *Si… alors…Sinon…* » ; (6) *Boucle « Pour »* ; (7) *Boucle « « TantQue »/« Répète »* ; (8) *Exercices bilan* ; (9) *Glossaire*. La partie « *A la découverte de la calculatrice* » est constituée de trois fiches : (1) *Fonctions* ; (2) *Calcul* ; (3) *Programmation*. Quant à la dernière partie, l'élève peut y trouver les points clés du programme de mathématiques.

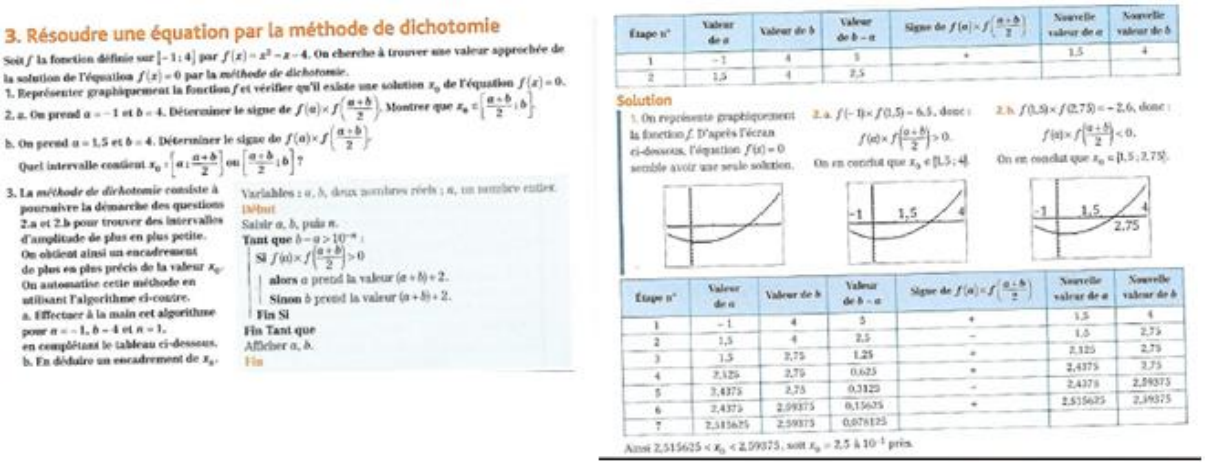
Ensuite, le manuel est constitué de trois parties traitant respectivement des *fonctions*, des *statistiques et probabilités* et de la *Géométrie*. Chacune de ces parties sont elles-mêmes découpées en plusieurs chapitres. Ainsi, la partie sur les *fonctions* est divisée en trois chapitres intitulés *Généralités sur les fonctions*, *Fonctions de référence* et *Compléments sur les fonctions*.

La structure de chaque chapitre est la suivante : (1) une *entrée dans le chapitre* ; (2) une *activité de découverte* ; (3) un *cours* clair et synthétique accompagnée d'applications ; (4) une double page de *raisonnement mathématique* ; (5) des *exercices résolus*, considérés par les auteurs comme incontournables pour comprendre les objectifs et les méthodes attendus dans le cours ; (6) une double page de *tests* afin de permettre à l'élève de faire une première évaluation et de faire le point sur les nouveaux savoirs mis en place dans le cours ; (7) des *exercices*, classés par rubriques et de difficultés progressives ; (8) une double page de *travaux pratiques* faisant appel aux outils TICE ; (9) une *activité de recherche* permettant aux élèves d'aller plus loin dans leur réflexion et dans le raisonnement ; (10) des *profils professionnels* en lien avec les mathématiques sont proposés en fin de chapitre.

3.1.2.2. Présentation des trois exercices en lien avec la dichotomie

Dans la partie « *Exercices résolus* » du chapitre « *Fonctions de référence* », les auteurs proposent un exercice intitulé « *Résoudre une équation par la méthode de dichotomie* » (fig.

53). Cet exercice propose de déterminer une valeur approchée de la solution x_0 d'une équation $f(x) = 0$, où f est une fonction trinôme définie sur un intervalle $[a ; b]$ donné dont on connait son expression. Il est demandé aux élèves de représenter graphiquement la fonction et de vérifier l'existence d'une solution unique sur l'intervalle $[a ; b]$. Puis, il est proposé aux élèves un processus de calcul qu'ils doivent répéter de façon à obtenir un encadrement de plus en plus précis de la valeur x_0. Pour cela, un algorithme associé à cette méthode est donné en langage pseudo-code. Les élèves doivent faire tourner cet algorithme « *à la main* » afin de compléter un tableau décrivant les différentes étapes. Une solution complète est donnée par les auteurs.



3. Résoudre une équation par la méthode de dichotomie

Soit f la fonction définie sur $[-1 ; 4]$ par $f(x) = x^2 - x - 4$. On cherche à trouver une valeur approchée de la solution de l'équation $f(x) = 0$ par la *méthode de dichotomie*.

1. Représenter graphiquement la fonction f et vérifier qu'il existe une solution x_0 de l'équation $f(x) = 0$.

2. a. On prend $a = -1$ et $b = 4$. Déterminer le signe de $f(a) \times f\left(\frac{a+b}{2}\right)$. Montrer que $x_0 \in \left[\frac{a+b}{2} ; b\right]$.

b. On prend $a = 1,5$ et $b = 4$. Déterminer le signe de $f(a) \times f\left(\frac{a+b}{2}\right)$.

Quel intervalle contient x_0 : $\left[a ; \frac{a+b}{2}\right]$ ou $\left[\frac{a+b}{2} ; b\right]$?

3. La *méthode de dichotomie* consiste à poursuivre la démarche des questions 2.a et 2.b pour trouver des intervalles d'amplitude de plus en plus petite. On obtient ainsi un encadrement de plus en plus précis de la valeur x_0. On automatise cette méthode en utilisant l'algorithme ci-contre.
a. Effectuer à la main cet algorithme pour $a = -1$, $b = 4$ et $n = 1$, en complétant le tableau ci-dessous.
b. En déduire un encadrement de x_0.

```
Variables : a, b, deux nombres réels ; n, un nombre entier.
Début
Saisir a, b, puis n.
Tant que b − a > 10^−n :
    Si f(a) × f((a+b)/2) > 0
    alors a prend la valeur (a+b)/2.
    Sinon b prend la valeur (a+b)/2.
    Fin Si
Fin Tant que
Afficher a, b.
Fin
```

Étape n°	Valeur de a	Valeur de b	Valeur de $b - a$	Signe de $f(a) \times f\left(\frac{a+b}{2}\right)$	Nouvelle valeur de a	Nouvelle valeur de b
1	−1	4	5	+	1,5	4
2	1,5	4	2,5			

Solution

1. On représente graphiquement la fonction f. D'après l'écran ci-dessous, l'équation $f(x) = 0$ semble avoir une seule solution.

2. a. $f(-1) \times f(1,5) = 6,5$, donc : $f(a) \times f\left(\frac{a+b}{2}\right) > 0$. On en conclut que $x_0 \in [1,5 ; 4]$.

2. b. $f(1,5) \times f(2,75) = -2,6$, donc : $f(a) \times f\left(\frac{a+b}{2}\right) < 0$. On en conclut que $x_0 \in [1,5 ; 2,75]$.

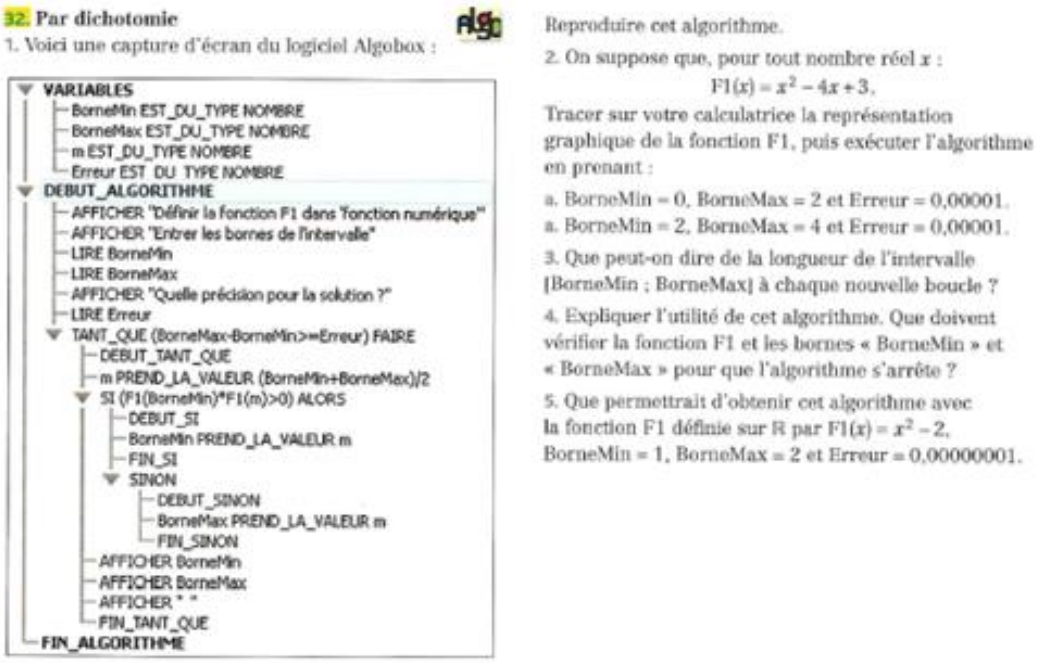
Étape n°	Valeur de a	Valeur de b	Valeur de $b - a$	Signe de $f(a) \times f\left(\frac{a+b}{2}\right)$	Nouvelle valeur de a	Nouvelle valeur de b
1	−1	4	5	+	1,5	4
2	1,5	4	2,5	−	1,5	2,75
3	1,5	2,75	1,25	+	2,125	2,75
4	2,125	2,75	0,625	+	2,4375	2,75
5	2,4375	2,75	0,3125	−	2,4375	2,59375
6	2,4375	2,59375	0,15625	+	2,515625	2,59375
7	2,515625	2,59375	0,078125			

Ainsi $2,515625 < x_0 < 2,59375$, soit $x_0 \approx 2,5$ à 10^{-1} près.

Figure 53 (Extrait d'un exercice du manuel de 2nde Maths-repères : Méthode de dichotomie)

L'algorithme donné utilise une structure de boucle « TantQue », et on peut constater qu'il n'est pas demandé aux élèves de programmer cet algorithme dans un environnement numérique. Les élèves peuvent l'exécuter seulement « *à la main* ». On peut noter qu'aucun autre exercice utilisant cette méthode n'est demandé dans ce chapitre.

Dans la partie « *exercices* » du chapitre « *Compléments sur les fonctions* », les auteurs proposent un exercice sur une utilisation de l'*algorithme de dichotomie* (fig. 54) appliquée sur des fonctions trinômes.



32. Par dichotomie

1. Voici une capture d'écran du logiciel Algobox :

```
VARIABLES
    BorneMin EST_DU_TYPE NOMBRE
    BorneMax EST_DU_TYPE NOMBRE
    m EST_DU_TYPE NOMBRE
    Erreur EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
    AFFICHER "Définir la fonction F1 dans fonction numérique"
    AFFICHER "Entrer les bornes de l'intervalle"
    LIRE BorneMin
    LIRE BorneMax
    AFFICHER "Quelle précision pour la solution ?"
    LIRE Erreur
    TANT_QUE (BorneMax-BorneMin>=Erreur) FAIRE
        DEBUT_TANT_QUE
        m PREND_LA_VALEUR (BorneMin+BorneMax)/2
        SI (F1(BorneMin)*F1(m)>0) ALORS
            DEBUT_SI
            BorneMin PREND_LA_VALEUR m
            FIN_SI
        SINON
            DEBUT_SINON
            BorneMax PREND_LA_VALEUR m
            FIN_SINON
        AFFICHER BorneMin
        AFFICHER BorneMax
        AFFICHER " "
        FIN_TANT_QUE
FIN_ALGORITHME
```

Reproduire cet algorithme.

2. On suppose que, pour tout nombre réel x :

$$F1(x) = x^2 - 4x + 3.$$

Tracer sur votre calculatrice la représentation graphique de la fonction F1, puis exécuter l'algorithme en prenant :

a. BorneMin = 0, BorneMax = 2 et Erreur = 0,00001.
a. BorneMin = 2, BorneMax = 4 et Erreur = 0,00001.

3. Que peut-on dire de la longueur de l'intervalle [BorneMin ; BorneMax] à chaque nouvelle boucle ?

4. Expliquer l'utilité de cet algorithme. Que doivent vérifier la fonction F1 et les bornes « BorneMin » et « BorneMax » pour que l'algorithme s'arrête ?

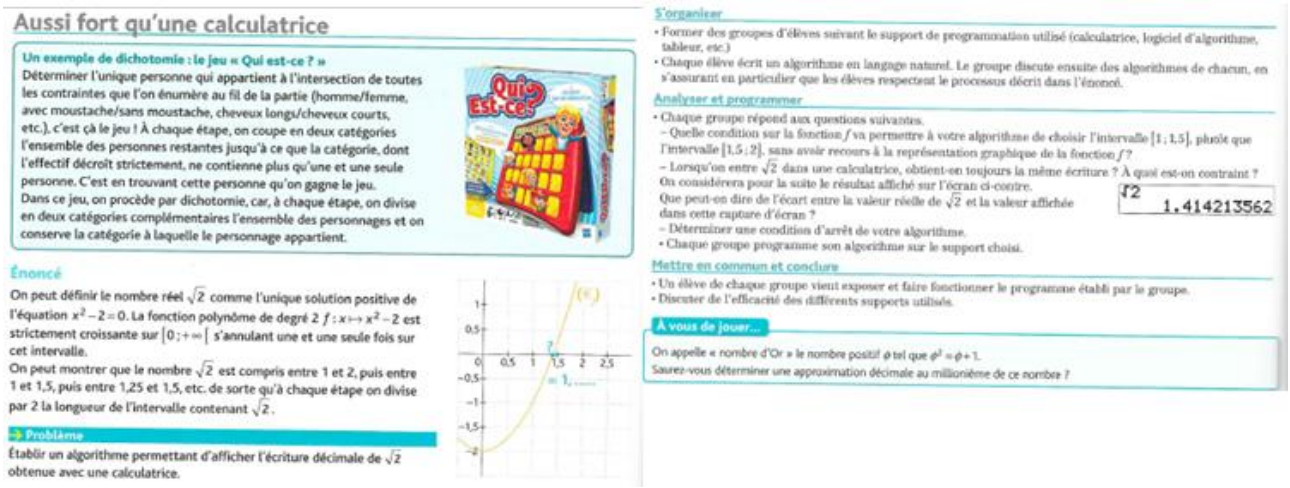
5. Que permettrait d'obtenir cet algorithme avec la fonction F1 définie sur $\mathbb{R}$ par $F1(x) = x^2 - 2$, BorneMin = 1, BorneMax = 2 et Erreur = 0,00000001.

Figure 54 (Extrait d'un exercice du manuel de 2nde Maths-repères : Par dichotomie et AlgoBox)

Comme nous pouvons le constater, l'algorithme est donné sous forme de langage pseudo-code. Les élèves, pouvant se référer au titre de l'exercice : *Par dichotomie*, peuvent anticiper l'objectif principal de cet exercice et anticiper les attentes des auteurs de l'exercice avant de lire totalement l'énoncé. En effet, pour un élève de Seconde, la *dichotomie* est associée à une *application numérique* ayant pour objectif de déterminer une approximation ou un encadrement d'une solution d'une équation de la forme $f(x) = 0$ (voire $f(x) = k$, k nombre réel donné). Ils savent aussi qu'avant de passer à cette *application numérique*, ils doivent observer que la fonction f vérifie certaines conditions permettant de supposer que l'équation admet une unique solution sur un intervalle fixé. Ce travail leur permet alors de justifier le recours à l'algorithmique pour obtenir des intervalles emboités ayant des amplitudes de plus en plus « fines ». Cependant, le fait de mettre l'algorithme en début de l'énoncé et de demander dès la première question de le reproduire, réduit cette première étape à une simple « tâche de copie » sur machine, en particulier si le logiciel utilisé est celui correspondant au langage machine proposé par la capture de l'écran. A ce niveau de l'exercice, l'adidacticité laissée à l'élève relève pour l'essentiel du choix de l'instrument de programmation. En effet, suivant le choix que fera l'élève, il devra tenir compte ou pas de contraintes d'ordre syntaxique dues au langage machine utilisé. De plus, lors de cette tâche d'implémentation de l'algorithme dans la machine, nous pouvons relever que l'algorithme proposé n'est pas défini pour une fonction

particulière mais pour une fonction générique notée F1. Les questions qui suivent, font intervenir des fonctions polynomiales de degré deux ainsi que les intervalles de recherche où les fonctions n'admettent qu'un seul zéro. Ce dernier point fait l'objet d'un questionnement de la part des auteurs : *Expliquer l'utilité de cet algorithme*. Nous rappelons qu'au niveau de la Seconde, les élèves n'ont pas la connaissance de la technique de résolution algébrique des équations du second degré dans **R**. De plus, la justification de l'existence des solutions semble se faire dans un premier temps à l'aide de la représentation graphique de la fonction. Cependant, dans la question 4 : « […] *Que doivent vérifier la fonction F1 et les bornes « BorneMin » et « BorneMax » pour que l'algorithme s'arrête ? »*, nous pouvons supposer que les auteurs attendent des élèves qu'ils se réfèrent de façon implicite à des conditions de validité de l'algorithme proches de celles du TVI et de son corollaire, savoirs qui ne font pas partie des compétences des élèves de Seconde. Nous constatons aussi que dans l'algorithme donné par les auteurs, la condition sur le *Si… alors…Sinon* est conforme à ce que les élèves connaissent sur le fonctionnement de la *méthode de dichotomie*. Ce choix imposé par les auteurs ne nécessite pas une monotonie particulière de la fonction étudiée autre que le fait qu'elle soit strictement monotone sur l'intervalle où l'équation $f(x) = 0$ admet une unique solution. Dans le domaine algorithmique, les auteurs insistent aussi sur le choix des bornes des intervalles considérés et plus particulièrement sur l'amplitude de ces intervalles. Le fait d'utiliser dans la boucle « TantQue » comme contrainte une inégalité du type « supérieure ou égale » peut aider l'élève à mieux comprendre que le nom choisi pour la variable correspondante soit « Erreur » car elle peut le renvoyer à des raisonnements vus en probabilités sur le concept d'événement contraire. Cet exercice est aussi accompagné d'un commentaire intitulé « *Le saviez-vous ?* » et indiquant : « *Cet algorithme porte le nom d'algorithme de dichotomie et permet d'obtenir des approximations de nombres tels que* $\sqrt{2}, \sqrt{3}, \sqrt{5}$*, etc. En revanche, ce n'est pas cet algorithme qui équipe vos calculatrices : c'est l'algorithme de Héron, bien plus rapide ! On dit que la convergence de cet algorithme est quadratique* ». Nous pouvons aussi constater que dans le commentaire, les auteurs font référence aux concepts de convergence et d'optimalité de l'algorithme, sans que pour autant soit proposé une réelle étude de comparaison sur deux méthodes de calculs de valeurs approchées de racines carrées de nombres premiers. De plus, aucun exercice sur la l'*algorithme de Héron* n'est donné à la suite de ce commentaire.

Cependant, dans ce même chapitre, à la partie « *Activité de recherche* », est proposé un travail de recherche (fig. 55) sur une application de la *méthode de dichotomie* permettant de déterminer une écriture décimale du nombre réel $\sqrt{2}$.



Aussi fort qu'une calculatrice

Un exemple de dichotomie : le jeu « Qui est-ce ? »

Déterminer l'unique personne qui appartient à l'intersection de toutes les contraintes que l'on énumère au fil de la partie (homme/femme, avec moustache/sans moustache, cheveux longs/cheveux courts, etc.), c'est ça le jeu ! À chaque étape, on coupe en deux catégories l'ensemble des personnes restantes jusqu'à ce que la catégorie, dont l'effectif décroît strictement, ne contienne plus qu'une et une seule personne. C'est en trouvant cette personne qu'on gagne le jeu.
Dans ce jeu, on procède par dichotomie, car, à chaque étape, on divise en deux catégories complémentaires l'ensemble des personnages et on conserve la catégorie à laquelle le personnage appartient.

Énoncé

On peut définir le nombre réel $\sqrt{2}$ comme l'unique solution positive de l'équation $x^2 - 2 = 0$. La fonction polynôme de degré 2 $f : x \mapsto x^2 - 2$ est strictement croissante sur $[0 ; +\infty[$ s'annulant une et une seule fois sur cet intervalle.
On peut montrer que le nombre $\sqrt{2}$ est compris entre 1 et 2, puis entre 1 et 1,5, puis entre 1,25 et 1,5, etc. de sorte qu'à chaque étape on divise par 2 la longueur de l'intervalle contenant $\sqrt{2}$.

Problème

Établir un algorithme permettant d'afficher l'écriture décimale de $\sqrt{2}$ obtenue avec une calculatrice.

S'organiser

- Former des groupes d'élèves suivant le support de programmation utilisé (calculatrice, logiciel d'algorithme, tableur, etc.)
- Chaque élève écrit un algorithme en langage naturel. Le groupe discute ensuite des algorithmes de chacun, en s'assurant en particulier que les élèves respectent le processus décrit dans l'énoncé.

Analyser et programmer

- Chaque groupe répond aux questions suivantes.
  – Quelle condition sur la fonction f va permettre à votre algorithme de choisir l'intervalle $[1 ; 1,5]$, plutôt que l'intervalle $[1,5 ; 2]$, sans avoir recours à la représentation graphique de la fonction f ?
  – Lorsqu'on entre $\sqrt{2}$ dans une calculatrice, obtient-on toujours la même écriture ? À quoi est-on contraint ?
  On considérera pour la suite le résultat affiché sur l'écran ci-contre.
  Que peut-on dire de l'écart entre la valeur réelle de $\sqrt{2}$ et la valeur affichée dans cette capture d'écran ?
  – Déterminer une condition d'arrêt de votre algorithme.
- Chaque groupe programme son algorithme sur le support choisi.

√2 1.414213562

Mettre en commun et conclure

- Un élève de chaque groupe vient exposer et faire fonctionner le programme établi par le groupe.
- Discuter de l'efficacité des différents supports utilisés.

À vous de jouer...

On appelle « nombre d'Or » le nombre positif ϕ tel que $\phi^2 = \phi + 1$.
Saurez-vous déterminer une approximation décimale au millionième de ce nombre ?

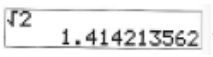
Figure 55 (Extrait d'une « activité de recherche » du manuel de 2nde Maths-repères : Ecriture décimale du nombre réel $\sqrt{2}$)

Les auteurs justifient ce travail en énonçant que ce nombre réel est l'unique solution positive de l'équation $x^2 - 2 = 0$. Pour cela, ils donnent une représentation de la fonction polynôme de degré 2 : $f : x \mapsto x^2 - 2$ sur l'intervalle [– 0,5 ; 2,5] et affirment sans aucune justification autre que graphique, que cette fonction est strictement croissante sur l'intervalle [0 ; + ∝[et qu'elle s'annule une et une seule fois sur celui-ci. La mise en place d'une approche dichotomique du problème est indiquée par une introduction autour de la description d'un jeu intitulé « *Qui est-ce ?* » qui est située avant l'énoncé du problème. L'énoncé du problème se termine par une question que l'on peut qualifier de semi-ouverte : *Etablir un algorithme permettant d'afficher l'écriture décimale de $\sqrt{2}$ obtenue avec un calculatrice*. Comme nous pouvons aussi le constater, l'organisation du travail de recherche demandé aux élèves est très claire et fortement guidée par les auteurs. En effet, les élèves doivent former des groupes en fonction des supports algorithmiques qu'ils utiliseront : calculatrice, logiciel algorithmique, tableur,… Ainsi, les élèves travaillent en autonomie « *instrumentale* ». Dans chaque groupe, tous les élèves doivent d'abord proposer un algorithme en langage naturel répondant au processus décrit dans l'énoncé : « *On peut montrer que le nombre $\sqrt{2}$ est compris entre 1 et 2, puis entre 1 et 1,5, puis entre 1,25 et 1,5, etc. de sorte qu'à chaque étape on divise par 2 la longueur de l'intervalle contenant $\sqrt{2}$.* ». On peut supposer que ce processus va limiter les choix dans les méthodes de calcul des élèves. Les diverses formulations des algorithmes

proposées par les élèves font alors l'objet d'un échange oral dans chaque groupe afin de dégager une unique formulation de l'algorithme décrivant le processus décrit dans l'énoncé. Aucune contrainte de temps n'est indiquée par les auteurs de l'activité. On peut supposer que cet aspect temporel est laissé à la charge de l'enseignant encadrant la classe. Au cours du travail sur l'algorithme à construire en langage naturel, une organisation des tâches à faire est indiquée par les auteurs. En effet, dans le paragraphe intitulé « *Analyser et programmer* », les auteurs donnent les consignes auxquelles doivent se référer les élèves de chaque groupe, c'est-à-dire :

- condition sur la fonction permettant à l'algorithme de choisir tel ou tel sous-intervalle à chaque étape du processus ;
- détermination d'une condition d'arrêt de l'algorithme ;
- programmation de l'algorithme sur le support matériel choisi par le groupe.

Aucune indication sur la structure de l'algorithme ainsi que sur les variables itératives utilisées n'est donnée par les auteurs. Lors de notre ingénierie, nous verrons que ce sont deux « objets » qui peuvent poser problème à des élèves débutant en algorithmique et en programmation.

Les auteurs demandent aux élèves d'étudier ce que renvoie la calculatrice lorsque l'on rentre le nombre $\sqrt{2}$. Ainsi, ils proposent comme possibilité de capture d'écran : 

. Nous pouvons supposer que les questions : *« Lorsqu'on entre* $\sqrt{2}$ *dans une calculatrice, obtient-on toujours la même écriture ? A quoi est-on contraint ? Que peut-on dire de l'écart entre la valeur réelle* $\sqrt{2}$ *et la valeur affichée* » lors d'une capture d'écran peuvent être interprétées comme une aide aux élèves à formaliser une condition d'arrêt de l'algorithme sous forme d'une inégalité de longueur.

Pour terminer la description de l'organisation du travail, les auteurs demandent à chaque groupe de désigner un rapporteur qui aura la charge d'exposer à l'ensemble de la classe, l'algorithme et de le faire tourner après implémentation dans un environnement numérique. Lors de cette mise en commun, une discussion entre les différents groupes doit aussi avoir lieu sur l'efficacité des différents supports « instrumentaux » utilisés et non sur l'efficacité de l'algorithme en lui-même. Pour conclure l'activité de recherche proposée, les auteurs demandent d'adapter l'algorithme afin d'obtenir une approximation décimale au millionième

du « nombre d'or » en utilisant le fait que ce nombre réel positif, noté Φ, est l'unique solution positive de l'équation $x^2 = x + 1$.

3.1.2.1. Approche de ces trois exercices à l'aide du cadre théorique sur les ETA/ETM

Nous souhaitons compléter les analyses faites sur les trois exercices par une approche en termes de *paradigmes algorithmiques* et d'articulations entre *ETA/ETM* (tableaux n° 19 et 20).

<table>
<tr><td>Thème de l'exercice</td><td colspan="2">Découverte de la méthode de dichotomie : « application numérique »
(Fig. 53)</td></tr>
<tr><td>Paradigmes</td><td colspan="2">Les questions en lien avec la construction de la représentation graphique de la fonction et la vérification d'une unique solution de l'équation f(x) = 0 à partir du graphique se situent au niveau 1 de l'algorithmique.

L'interprétation du signe du produit des images par la fonction f des nombres a et (a + b)/2, la conséquence de l'étude de ce signe sur la détermination de la suite des intervalles emboités contenant le nombre x_0, unique solution de l'équation f(x) = 0 se situe à la frontière entre les niveaux 1 et 2 de l'algorithmique.
En effet, le travail demandé aux élèves suppose qu'à partir d'une représentation spécifique des objets et des actions élémentaires, ils construisent un traitement permettant de répondre à une problématique qui doit être interprétée par les élèves comme méthode permettant de définir une suite d'intervalles emboités encadrant l'unique solution x_0 de l'équation f(x) = 0.

Cependant, un algorithme est proposé donnant la structure de la boucle ainsi que le test de sortie de cette boucle. Le fait que les élèves ne doivent tester cet algorithme qu'au papier-crayon afin de remplir le tableau proposé par les auteurs de l'exercice où les intitulés des colonnes sont déjà donné, font que l'exercice dans son ensemble se situe au niveau 1 de l'algorithmique.
Les élèves n'ont pas à construire de formules permettant d'obtenir les bornes des intervalles emboitées. De plus, ils doivent se contenter d'itérer le traitement.

Il n'est pas précisé explicitement à quoi correspond l'amplitude du dernier intervalle obtenu.

On peut constater que dans la solution de l'exercice, les auteurs proposent une valeur approchée de x_0 qui n'est demandée dans l'énoncé.</td></tr>
<tr><td rowspan="2">Espaces de Travail spécifiques</td><td>ETA</td><td>ETM</td></tr>
<tr><td>La question portant sur la représentation graphique laisse le choix à l'élève, entre une utilisation d'un instrument numérique ou de la feuille de papier. A ce niveau scolaire, nous pouvons supposer que le travail de l'élève se situe dans une genèse instrumentale.

Dans cet exercice, les questions autour</td><td>La reconnaissance et l'interprétation du signe de $f(a) \times f\left(\frac{a+b}{2}\right)$ pour la détermination de la suite des bornes des intervalles emboités, se situent dans des genèses instrumentale/discursive.</td></tr>
</table>

	de l'algorithmique ne nécessitent qu'une compréhension de l'algorithme proposé en langage pseudo-code et de son utilisation au papier-crayon en interprétant la fonction *f* de l'algorithme comme fonction générique. Nous nous situons dans des genèses instrumentale/sémiotique.	

Tableau 19

Thème de l'exercice	Méthode de dichotomie : « analyse numérique » (Fig. 54 et 55)
Paradigmes	Nous choisissons d'étudier ces deux exercices ensemble car ils se situent dans le cadre d'une situation d' « analyse numérique » au sens que nous donnerons dans le chapitre suivant. • <u>Exercice de la figure 54</u> Les auteurs donnent une capture d'écran du logiciel *AlgoBox* correspondant à la méthode de dichotomie pour une fonction « générique » F1. Les élèves peuvent prendre conscience du rôle de cet algorithme en se référant d'abord au titre de l'exercice « Par dichotomie ». L'algorithme laisse aussi aux élèves le choix de fixer les valeurs des bornes de l'intervalle initial d'étude où se situe l'unique solution de l'équation F1(*x*) = 0 sur cet intervalle. Dans la question 1, les élèves doivent reproduire cet algorithme sans que pour autant soit préciser explicitement le choix de l'environnement numérique. Cependant, on peut supposer que le fait que cet algorithme soit écrit dans le langage pseudo-code d'AlgoBox va faire que les élèves vont majoritairement choisir cet environnement pour reproduire l'algorithme. Dans la question 2, les élèves ont l'expression d'une fonction F1. La représentation graphique de cette fonction doit être faite sur la calculatrice. Ainsi, les élèves peuvent travailler dans deux environnements numériques : la calculatrice pour la représentation graphique et *AlgoBox* pour la méthode de dichotomie. Ensuite, les auteurs fixes deux intervalles que les élèves doivent successivement utiliser dans l'algorithme avec la fonction F1 définie précédemment. A ce stade de l'exercice, il n'est pas demandé aux élèves de justifier le pourquoi du choix prédéfini des valeurs des bornes des deux intervalles. Nous restons au niveau 1 de l'algorithmique tout au long des questions 1 et 2. Dans les questions 3 et 4, les élèves doivent travailler sur la validité des choix faits sur les bornes initiales des deux intervalles, ainsi que sur l'amplitude des intervalles cherchés après exécution de l'algorithme. De plus, les élèves sont amenés à s'interroger sur les justifications d'existence et d'unicité d'une solution de l'équation F1(*x*) = 0 sur chacun des intervalles initiaux choisis. A ce niveau de scolarité, nous pouvons penser que ces justifications vont essentiellement se baser

<table>
<tr><td></td><td colspan="2">sur des observations graphiques.

Ces questions 3 et 4 mènent les élèves à un travail de recherche et de justifications se situant au niveau 2 de l'algorithmique.

• <u>Exercice de la figure 55</u>

Cet exercice se situe dans un cadre dit « activité de recherche ». En effet, l'exercice ne propose pas des questions explicites et semble se contenter d'un travail autour de la construction d'un algorithme associé à la méthode dichotomie sous la forme d'une simple reprise du travail fait à l'exercice de la figure 2b, en particulier lors de la dernière question de cet exercice. Cependant, les différentes attentes décrites par les auteurs : s'organiser, analyser et programmer, mettre en commun et conclure font que les élèves doivent justifier les différentes étapes des différents raisonnements, tant mathématique avec la justification de que $\sqrt{2}$ est l'unique solution positive de l'équation x² – 2 = 0, qu'algorithmique avec la construction de l'algorithme et en particulier le travail sur une condition d'arrêt de l'algorithme.

L'ensemble du travail attendu se situe donc au niveau 2 de l'algorithmique.</td></tr>
<tr><td rowspan="2">Espaces de Travail spécifiques</td><td>ETA</td><td>ETM</td></tr>
<tr><td>• <u>Exercice de la figure 54</u>
La question portant sur la représentation graphique impose que l'élève utilise sa calculatrice. Le travail se situe dans des genèses instrumentale/sémiotique.

De même, l'algorithme étant fourni et écrit dans le langage pseudo-code du logiciel AlgoBox, la reproduction de cet algorithme fait que le travail se situe dans une simple genèse instrumentale. L'aspect genèse sémiotique ne semble pas être nécessaire vu que si les élèves choisissent de travailler dans l'environnement numérique correspondant à l'algorithme donné, ils restent dans une tâche de reproduction ne nécessitant pas de comprendre le rôle et les différentes étapes de l'algorithme.

Cependant, le fait que l'algorithme utilise la fonction générique F1, et que celui-ci doit être testé pour des fonctions particulières, nous avons aussi une</td><td>• <u>Exercice de la figure 54</u>
La reconnaissance et l'interprétation du signe de $F1(BorneMin) \times F1(m)$ pour la détermination de la suite des bornes des intervalles emboités, ainsi que le choix de la précision pour l'amplitude de l'intervalle d'encadrement de la solution, font que le travail dans le champ de l'analyse se situe dans des genèses instrumentale/discursive.
Nous rappelons qu'à ce stade de la scolarité, les élèves n'ayant pas la technique de la résolution d'une équation du second degré fait que la première partie de l'exercice sur la fonction $x \rightarrow$ F1(x) = $x^2 - 4x + 3$ peut être vue comme une tâche situant l'utilisation de la méthode de dichotomie comme relevant de l'analyse numérique[66] et non une simple application numérique.
En revanche, les élèves ayant la connaissance de la factorisation de</td></tr>
</table>

[66] Nous reviendrons dans le chapitre suivant sur les sens que nous donnons à « application numérique » et « analyse numérique ».

	genèse sémiotique qui intervient. De plus, le travail sur les conditions de terminaison de l'algorithme fait que le travail des élèves se situe aussi dans des genèses instrumentale/discursive. • Exercice de la figure 55 Les élèves devant justifier entre autres la condition d'arrêt de leur algorithme avant de construire l'algorithme et l'implémenter dans l'environnement numérique de leur choix, nous situons ici dans des genèses instrumentale/sémiotique pour le choix de l'environnement, mais aussi discussive pour la justification de la condition d'arrêt.	l'expression $a^2 - b^2$, la question portant sur la fonction $x \rightarrow F1(x) = x^2 - 2$, peut être vue d'un point de vue purement mathématique et permettre à l'élève de mieux comprendre le rôle de l'algorithme en lien avec la remarque qui conclue l'exercice sur les approximations de nombres comme $\sqrt{2}$ à l'aide de la méthode de dichotomie. Ici, les élèves sont dans une genèse discursive de l'$ETM_{analyse}$. • Exercice de la figure 55 Le fait que les élèves doivent justifier les conditions que doivent vérifier la fonction $f : x \rightarrow x^2 - 2$ pour permettre de choisir l'intervalle [1 ; 2] comme intervalle initiale d'étude, puis de réduire cet intervalle à [1 ; 1,5] plutôt qu'à l'intervalle [1,5 ; 2] dans l'algorithme, sans avoir recours à la représentation graphique de la fonction *f* permet de situer le problème dans une genèse discursive.

Tableau 20

A partir des analyses présentées dans les deux tableaux précédents, nous en déduisons ce que peuvent être les conséquences de ce type d'exercices vis à vis de notre ingénierie (tableau n° 21).

Implication pour notre ingénierie	Nous constatons que : 1. Pour l'essentiel du travail proposé dans l'exercice de la figure 53, celui-ci reste au niveau 1 en algorithmique : itération « naturelle », pas de problématique de terminaison,… Tout est prédéfini. Le travail est du type « application numérique » sur le plan mathématique. 2. Cependant, une première approche du niveau 2 en algorithmique peut être observée dans les exercices 54 et 55, en particulier sur le travail concernant plus précisément la terminaison de l'algorithme. 3. Dans les exercices 54 et 55, les genèses instrumentale/sémiotique/discursive dans l'*ETA* viennent supporter la genèse discursive dans l'*ETM*. Ces deux exercices permettent une approche de type « analyse numérique » d'un travail sur la dichotomie. Nous souhaitons, dans notre ingénierie, élaborer des tâches de type « application numérique » ou de type « analyse numérique ». Ceci sera présenté dans les chapitres 4 et 5.

Tableau 21

3.1.3 Autres ressources : Documents ressources – Documents IREM

De nombreux documents proposés par *Eduscol* et les *IREM* fournissent des éléments pour l'étude en classe de l'*algorithme de dichotomie*. *Eduscol* est le site pédagogique du Ministère de l'Education Nationale, où toute personne peut y trouver toutes les informations officielles essentielles relatives à l'enseignement à l'Ecole, au Collège et au Lycée. Il a été conçu par la direction de l'enseignement scolaire du Ministère de l'Éducation Nationale, de l'Enseignement Supérieur et de la Recherche. Quant aux *IREM*, ils sont des lieux de formation et de recherche où des professeurs de différents niveaux (primaire, secondaire, supérieur) consacrent du temps pour réfléchir en commun aux mathématiques et aux problèmes que l'on rencontre pour les enseigner aujourd'hui.

Nous choisissons d'étudier des extraits de deux *documents ressources* proposés par *Eduscol* de façon à situer les attentes de l'institution, ainsi qu'une proposition d'activités pour la classe de Seconde élaborée par l'IREM d'Aix-Marseille. Nous ne visons pas l'exhaustivité. Ces documents nous ont paru adaptés pour préparer notre ingénierie sur la dichotomie.

3.1.3.1. Deux documents ressources *Eduscol*

L'institution, à travers le site *Eduscol*, donne accès à deux *documents ressources* intitulés respectivement « *Algorithmique* » et « *Fonctions* », où l'enseignant peut trouver diverses tâches en lien avec la *dichotomie.*

Le document « *Algorithmique »* est découpé en huit parties : (1) *Présentation générale* ; (2) *Une initiation à l'algorithmique* ; (3) *Exemples de dispositifs de classe* ; (4) *Algorithmes et géométrie* ; (5) *Algorithmes et fonctions* ; (6) *Algorithmes et probabilités* ; (7) *Bibliographie* ; (8) *Présentation rapide des logiciels*. Chacune de ces parties sont elles-mêmes partagées en plusieurs sous-parties. Les trois premières parties proposent une approche pédagogique générale et une initiation à l'algorithmique, puis des propositions de mise en œuvre dans différents domaines mathématiques. L'*algorithme de dichotomie* apparaît dans le domaine « *fonctions* ».

Avant d'analyser la proposition relative à cet algorithme, nous résumons l'approche pédagogique générale de ce document ressource. Selon cette approche, un enseignement de l'algorithmique fournit l'occasion de diversifier les dispositifs de classe et offre aux élèves la possibilité de modéliser d'une manière « *semi-formelle* » des algorithmes issus de la vie quotidienne et de les faire vivre sous formes d'activités de communication. Le document

précise aussi que dans le cadre d'une progression pédagogique diversifiée, il est pertinent de passer par des phases de tâtonnement, d'expression orale, d'échanges sans chercher à aborder trop rapidement une formalisation écrite qui peut s'avérer bloquante pour certains élèves. Il mentionne que les structures algorithmes et les variables itératives utilisées peuvent être source de rupture chez les élèves par rapport à leur conception des variables qu'ils ont l'habitude d'utiliser en mathématique.

Dans la partie « *Algorithmes et fonctions* » du document, nous avons quatre sous-parties : (1) *Recherche des extremums sur un segment : fenêtrage vertical* ; (2) *Tester la monotonie* ; (3) *La question du fenêtrage horizontal : comportement asymptotique* ; (4) *Recherche de solution d'équation et d'extremum*. La sous-partie intitulée « *Recherche de solution d'équation et d'extremum* » est divisée en deux paragraphes : un premier traitant de « *La dichotomie* » et un deuxième traitant d'un algorithme appelé « *Monter plus haut* » que nous n'exposons pas ici car n'ayant pas de lien direct avec notre ingénierie. Le paragraphe sur la *dichotomie* est introduit par une courte explication sur l'intérêt de mettre en place un travail sur la dichotomie. Contrairement à ce qui est observé dans les manuels scolaires, ce paragraphe propose d'abord une tâche sur l'algorithme « *discret* » de la *dichotomie* : « *Programmer un jeu : deviner le nombre en six essais* ». En effet, il est demandé aux élèves de deviner en moins de six essais un nombre entier tiré au hasard entre 10 et 100. L'algorithme doit indiquer à chaque proposition par l'utilisateur si le nombre proposé est supérieur ou inférieur au nombre cherche. Le document signale que, « *sans stratégie* », il est difficile d'y parvenir. De plus, il est signalé que le choix des valeurs 10 et 100 qui encadrent le nombre à trouver, ainsi que le nombre d'essais est à mettre en débat dans la classe. En effet, comme le signale les auteurs du document, selon le choix de ces valeurs, il sera ou non possible de déterminer à coup sûr la solution avec une bonne stratégie, ou on pourra seulement optimiser les chances de gagner. Les auteurs proposent alors un algorithme (Fig. 56) écrit en langage naturel, puis concluent sans donner d'explication précise qu'*une bonne stratégie conduit à l'algorithme utilisant la dichotomie*. *Cette méthode consiste, en choisissant à chaque fois la valeur située au milieu de l'intervalle en cours, à réduire de moitié à chaque fois l'amplitude de l'intervalle dans lequel se trouve le nombre et comme* 2^6 *est égal a 64, le dernier intervalle, sur cet exemple, est d'amplitude 1*. Le deuxième algorithme proposé (Fig. 57) correspond à l'algorithme « continu » sur la « *recherche d'un zéro par dichotomie* ». Il est décrit comme une

transposition de l'algorithme « discret ». Les auteurs indiquent en remarque, après l'algorithme, que les variables *a* et *b*, désignant au départ les bornes de l'intervalle d'étude initial, changent de valeur au fur et à mesure de l'exécution de la boucle « *Pour* ». Les auteurs précisent que comme l'algorithme est exécuté 50 fois, la largeur de l'intervalle initial est divisée par 2^{50} (soit environ 10^{15}), ce qui donne un bon encadrement de la valeur cherchée. Une implémentation de l'algorithme est proposée dans l'environnement numérique *SCRATCH*[67] avec la fonction $f : x \mapsto -x^2 + 10\,x - 23$.

```
Variables
    N nombre choisi par l'utilisateur
Initialisation
    S, un nombre entier au hasard entre 10 et 100
    essai prend la valeur 1
Traitement
    Tant que essai est inférieur ou égal à 6
        Saisir N
        Si N est supérieur à S alors
            Affiche « c'est moins »
        Si N est inférieur à S
            Affiche « c'est plus »
        Si n=S alors
            Affiche « gagné »
            fin de programme
        essai prend la valeur essai+1
Sortie
    Affiche « perdu ».
```

Figure 56 (Un algorithme écrit en langage naturel)

```
Variables
    m , valeur milieu de l'intervalle « courant »
Initialisation
    a et b, les bornes de l'intervalle [a ; b]
    f, la fonction (rappel : f change de signe entre a et b)
Traitement
    Pour i variant de 1 à 50
        m prend la valeur (a+b)/2
        Si f(m) et f (a) sont de même signe alors
            a prend la valeur m
        sinon
            b prend la valeur m
Sortie
    Affiche a et b
```

Figure 57 (Un algorithme de « dichotomie continue »)

Nous observons que le document « *Fonctions* » est découpé en six chapitres : (1) *Quels sont les objectifs à atteindre ?* ; (2) *La notion de fonction : une notion à travailler dans la durée ;* (3) *Une incitation pédagogique* ; (4) *Notations et raisonnement en analyse ;* (5) *Place de l'algorithmique en analyse* ; (6) *Quelques précisions sur des points particuliers du programme*, suivis de cinq *illustrations* (au sens d'exemples) : (1) *Une histoire de diviseurs* ; (2) *Le quadrilatère tournant* ; (3) *Patrons de récipients* ; (4) *Une formule de physique concernant la puissance électrique* ; (5) *Mesure de l'épaisseur d'un cheveu par diffraction*. Ce document se termine par des annexes proposant des exemples de *raisonnement à valoriser*, à *faire vivre en classes*.

Comme pour le document précédent, avant d'analyser la proposition relative à la dichotomie, nous résumons l'approche pédagogique générale de ce document ressource. Les auteurs du document rappellent que le programme fixe pour objectif la maîtrise de deux familles de problèmes : – une première famille où les problèmes se ramènent *à une équation du type f (x) = k dans le cas où la fonction est donnée mais aussi dans le cas où toute autonomie*

[67] Nous rappelons que ce logiciel n'a pas été retenu pour nos ingénieries (cf. Partie 1)

est laissée pour associer au problème divers aspects d'une fonction ; – une seconde famille constituée de *problèmes d'optimisation ou du type « f (x) > k ». Dans un premier temps un élève doit pouvoir résoudre un tel problème, de façon exacte ou approchée, à l'aide d'un graphique et de façon exacte si les variations de la fonction et les antécédents de k sont connus. Dans un second temps cette étude peut être faite, selon les cas, en exploitant les potentialités de logiciels, graphiquement ou algébriquement, toute autonomie pouvant être laissée pour associer au problème une fonction*. De plus, les auteurs rappellent que *la notion de fonction est, pour beaucoup d'élèves de seconde, une notion difficile à appréhender*. Cependant, il est nécessaire que sa maîtrise soit effective chez les élèves pour la poursuite de leurs études. Le travail sur les fonctions est amorcé dès le collège. *Un objectif essentiel de ce travail consiste à faire émerger progressivement, et sur des exemples concrets, « un processus faisant correspondre à un nombre un autre nombre »*. […] *Pour beaucoup d'élèves, la notion de fonction ne fait pas encore sens en début de seconde. Il importe donc qu'avant toute formalisation nouvelle, les élèves soient* […] *le plus souvent possible confrontés à des situations dans lesquelles il y ait besoin, pour répondre à une question posée* […] *d'identifier les avantages et les inconvénients de tel ou tel aspect d'une fonction*. […] *Le programme encourage une programmation moins centrée sur les notions elles-mêmes*, mais *davantage sur la nature des problèmes que les élèves doivent savoir résoudre*.

Quant à la place d'un enseignement de l'algorithmique en *analyse*[68], les auteurs signalent qu'une *première approche de l'algorithmique en analyse peut être l'automatisation d'une représentation graphique d'une fonction*. Ils indiquent que *les calculatrices graphiques et de nombreux logiciels* […] *donnent un tracé de la courbe représentative d'une fonction déterminée par une formule algébrique*, mais que *ces tracés sont faits de façon opaque*. Ils observent qu'il peut être *plus fructueux de conduire les élèves à tracer aussi une courbe « à la main » en partant d'un tableau de valeurs pour obtenir un nuage de points), de les inciter à se poser la question de la manière de joindre les points du nuage*, puis de *proposer ensuite aux élèves d'augmenter par étapes le nombre des points du nuage* afin de *renforcer leur compréhension de ce qu'est la courbe représentative d'une fonction en les aidant à mieux distinguer l'objet mathématique des dessins que l'on peut en faire*. Les auteurs du document proposent par exemple l'algorithme suivant, écrit en langage naturel (fig. 58).

[68] Terme employé par les auteurs du document

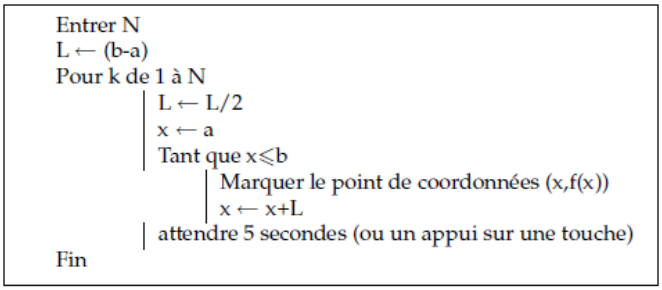


Figure 58 (Un algorithme de tracé d'un nombre fini de points de la représentation graphique d'une fonction générique)

Il s'agit d'un algorithme de tracé d'un nombre fini de points de la représentation graphique d'une fonction générique. Le choix de division de l'intervalle de définition par « dichotomie » (terme employé par les auteurs) viserait à une prise de conscience par les élèves de l'influence du pas sur le tracé. Cet algorithme est accompagné d'un bref commentaire sur la signification des données (la fonction *f*, les bornes *a* et *b*, le nombre d'itérations *N*), ainsi que sur les variables (variable entière dans la boucle avec *k*, longueur de l'intervalle entre deux points avec *L*, abscisse du point marqué avec *x*).

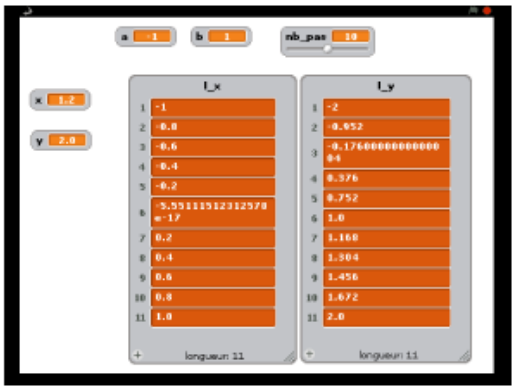
Parmi les exemples proposés sur la place de l'algorithmique dans le domaine de l'*analyse*, les auteurs proposent la recherche d'une valeur approchée d'une solution d'une équation de la forme $f(x) = 0$ par la méthode de dichotomie (fig. 59).

Comme nous pouvons l'observer, ils présentent divers registres : graphiques, tableau de valeurs, et logiciels algorithmiques sans pour autant donner la moindre justification concernant ces choix. Cependant, nous pouvons supposer que cela est dû au fait de ne pas favoriser tel ou tel langage conformément aux directives du programme : « *Aucun langage, aucun logiciel n'est imposé* ». Nous notons aussi que ce document « *Fonctions* » à la différence du document « *Algorithmique* » mentionne le changement de signe aux bornes et la stricte monotonie de la fonction. Il s'agit de propriétés parmi celles généralement retenues comme conditions suffisantes du *théorème des valeurs intermédiaires* (TVI) et de son corollaire (théorème de la bijection[69]), théorème que les auteurs ne mentionnent pas. En effet, ces propriétés de monotonie et de changement de signes sont au programme de seconde, tandis que le TVI et son corollaire n'interviennent qu'en Terminale. Ainsi, au niveau de la Seconde la question de la continuité reste implicite, les élèves n'ont qu'une approche « intuitive » des

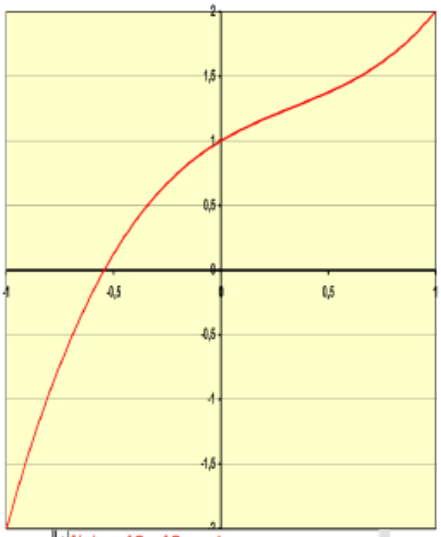
[69] Le **théorème de la bijection** est un corollaire du TVI qui affirme que *si une fonction f est continue et strictement monotone sur un intervalle I, alors elle définit une bijection entre cet intervalle I et son image f(I).*

justifications de l'existence et de l'unicité d'un zéro d'une fonction sur un intervalle de **R** donné.

Exemple : Considérons la fonction $x \mapsto x^3 - x^2 + x + 1$ sur l'intervalle $[-1;1]$. Un tableau de valeurs peut être obtenu avec le logiciel Scratch :

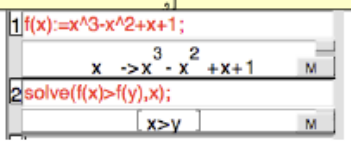


La fonction « passe de $f(-1) = -2$ à $f(1) = 2$ ». Intuitivement, elle va donc s'annuler. Une représentation graphique, donnée ci-contre, permet de conjecturer qu'elle est strictement croissante, ce qui peut être confirmé par un logiciel de calcul formel (voir, sous la courbe, le résultat donné par le logiciel Xcas).



Par exemple sur la calculatrice TI-nspire, l'algorithme de recherche d'une valeur approchée de la solution de l'équation $f(x) = 0$ peut se traduire par le programme :

```
Define dichot(a, b, n) =
Func
Local x, y, c :
x := a
y := b
While y − x > 10^−n
  c := (x + y)/2 :
  If f(x) * f(c) > 0 Then
    x := c
  Else
    y := c
  EndIf
EndWhile
Return x :
EndFunc
```



L'utilisation de la fonction **dichot**[1] ainsi définie se traduit par :

$dichot(-1, 1, 12)$	-0.54368901269208
$solve(f(x) = 0, x)$	-0.54368901269208

La dernière ligne servant à vérifier le résultat trouvé.

1. Notons au passage que l'algorithmique permet, conformément au programme de confronter les élèves à des fonctions autres que des fonctions d'une variable réelle : fonctions d'une variable entière, fonctions de plusieurs variables.

Direction générale de l'enseignement scolaire — Fonctions — 8 / 28

Figure 59 (Valeur approchée d'une solution d'une équation de la forme f(x) = 0 par la méthode de dichotomie)

3.1.3.2. Un document IREM

L'IREM d'Aix-Marseille propose un document de quatre pages (fig. 60) introduisant l'*algorithme de dichotomie* par une première approche *discrète* autour du jeu « *Le jeu du nombre caché* » : « *Albert a choisi un nombre compris entre 1 et 100, Bertrand doit le deviner. Bertrand fait des propositions et Albert répond « trop grand », « trop petit » ou « gagné ». Le*

jeu s'arrête lorsque Bertrand a trouvé le nombre. Faire fonctionner le jeu avec un camarade, archiver le déroulement du jeu dans un tableau sur papier. On veut maintenant remplacer Albert par un ordinateur et écrire l'algorithme à faire exécuter à l'ordinateur dans ce cas. »

L'algorithme de dichotomie

Henri ROLAND
Mai 2010

1) Le jeu du nombre caché

Albert a choisi un nombre compris entre 1 et 100, Bertrand doit le deviner.
Bertrand fait des propositions et Albert répond "trop grand", "trop petit" ou "gagné". Le jeu s'arrête lorsque Bertrand a trouvé le nombre.
Faire fonctionner le jeu avec un camarade, archiver le déroulement du jeu dans un tableau sur papier.
Que faut-il mémoriser ?
Que faut-il archiver dans le tableau ?
On décide d'appeler N le nombre caché et R la réponse de Bertrand.
Albert a choisi 63. On obtient le tableau suivant :

N	R	Test	Réponse d'Albert
63	40	R<N	Trop petit
	80	R>N	Trop grand
	60	R<N	Trop petit
	65	R>N	Trop grand
	62	R<N	Trop petit
	63	R=N	Gagné

On veut maintenant remplacer Albert par un ordinateur et écrire l'algorithme à faire exécuter à l'ordinateur dans ce cas.
Pour répondre au nombre R proposé par Bertrand on pourra envisager l'instruction conditionnelle :

```
Si R>N alors
    Afficher("Trop grand")
Sinon
    si R<N alors
        Afficher("Trop petit")
    Sinon
        Afficher("Gagné")
```

On obtient ainsi l'algorithme suivant

```
N⟵ Nombre aléatoire compris entre 1 et 100.
Lire(R)
Tant que R≠N Faire
    Si R>N Alors
        Afficher("Trop grand")
    Sinon
        si R<N Alors
            Afficher("Trop petit")
        Sinon
            Afficher("Gagné")
    Si R≠N Alors
        Lire(R)
```

Le `Lire(R)` en fin de boucle permet à Bertrand de taper le nombre suivant sauf dans le cas où il a gagné. On aurait préféré avoir le `Lire (R)` en début de boucle, par contre dans ce cas on aurait deux valeurs de R saisies avant le premier `Si···Alors`. Une solution satisfaisante sera d'initialiser R avec une valeur telle que R≠N dans tous les cas.

On obtient l'algorithme suivant :

```
N⟵ Nombre aléatoire compris entre 1 et 100.
R⟵0
Tant que R≠N Faire
    Lire(R)
    Si R>N Alors
        Afficher("Trop grand")
    Si R<N Alors
        Afficher("Trop petit")
Afficher("Gagné")
```

Programmation sur TI 82

```
:EntAléat(1,100)→N
:0→R
:While R≠N
:Prompt R
:If R>N
:Then
:Disp("TROP GRAND")
:End
:If R<N
:Then
:Disp("TROP PETIT")
:End
:End
:Disp("GAGNE")
```

Programmation en Python 2.6

```
N=randint(1,100)
R=0
while R!=N :
    R=input('R=')
    if R>N:
      print 'Trop grand'
    if R<N:
      print 'Trop petit'
print 'Gagné'
```

2) Initiation à la dichotomie

Bertrand joue contre la machine. Archiver le jeu de Bertrand dans un tableau sur papier.
Quelle technique de jeu peut employer Bertrand pour gagner le plus rapidement possible ?

- Première méthode : déterminer dans quel intervalle [A ; B] se trouve le nombre caché et proposer un entier aléatoire dans cet intervalle.
- Deuxième méthode : déterminer dans quel intervalle [A ; B] se trouve le nombre caché et proposer un entier C proche du milieu de [A;B].

En employant la première méthode, si le nombre caché est 72 :

A	B	C	R=Réponse d'Albert
1	100	53	Trop petit
54	100	80	Trop grand
54	79	57	Trop petit
58	79	65	Trop petit
66	79	78	Trop grand
66	77	71	Trop petit
72	77	77	Trop grand
72	76	72	Gagné

On veut maintenant replacer Bertrand par un ordinateur et écrire l'algorithme à faire exécuter à l'ordinateur dans ce cas.

Remarque
Les calculatrices programmables "non formelles" ne gèrent que les variables de type numérique et pas les chaînes de caractères. Il faut donc trouver un codage pour la réponse d'Albert.
Pour simplifier, Albert tapera -1 pour "Trop petit" ou 1 pour "Trop grand" ou 0 pour "Gagné".
On obtient l'algorithme suivant :

Première méthode avec un entier aléatoire

```
A⟵1
B⟵100
R⟵2
Tant que R≠0 Faire
    C⟵Entier Aléatoire entre A et B
    Afficher(C)
    Lire(R)
    Si R=1 Alors
        B⟵C-1
    Si R=-1 Alors
        A⟵C+1
```

Deuxième méthode par dichotomie

```
A⟵1
B⟵100
R⟵2
Tant que R≠0 Faire
    C⟵PartieEntière((A+B)/2)
    Afficher(C)
    Lire(R)
    Si R=1 Alors
        B⟵C-1
    Si R=-1 Alors
        A⟵C+1
```

Programmation de la deuxième méthode

Programmation sur TI 82

```
:1→A
:100→B
:2→R
:While R≠0
:PartEnt((A+B)/2)→C
:Disp("PROPOSITION=",C)
:Input("REPONSE ?",R)
:If R=1
:Then
:C-1→B
:End
:If R=-1
:Then
:C+1→A
:End
:End
```

Programmation en Python 2.6

```
from math import *
A=1
B=100
R=2
while R !=0 :
      C=floor((A+B)/2.)
      print 'Proposition = ',C
      R=input('Réponse ?  ')
      if R==1:
            B=C-1
      if R==-1:
            A=C+1
```

3) Utilisation de la dichotomie pour résoudre une équation

f est une fonction définie sur l'intervalle $[a;b]$ et strictement monotone sur $[a;b]$. On cherche à résoudre numériquement l'équation $f(x) = 0$.

- On peut éliminer tout d'abord les cas où $f(a) = 0$ ou $f(b) = 0$.
- L'existence d'une racine α sur $]a;b[$ est subordonnée au fait que $f(a)$ et $f(b)$ sont de signes contraires, ce qui équivaut à $f(a) \times f(b) < 0$, et que f est continue sur $[a\,;\,b]$.
- Si c est un réel de l'intervalle $]a,b[$, la position de α par rapport à c peut être testée par l'intruction suivante :


```
Si f(a)×f(c)⩽0 alors
    Rechercher α sur ]a;c]
Sinon
    Rechercher α sur ]c;b[
```

Dans la pratique on prendra pour c le milieu de l'intervalle $[a;b]$.
On itérera le processus jusqu'à obtenir par exemple $b - a < \varepsilon$ pour ε donné.

Algorithme

```
A, B, E et f sont donnés.
Si f(A)×f(B)≥0 alors
    Afficher('Pas de racine sur ]A;B[')
Sinon
    Tant que B-A≥E Faire
        C⟵(A+B)/2
        Si f(A)×f(C)⩽0 alors
            B⟵C
        Sinon
            A⟵C
    Afficher(A,B)
```

Ecriture du programme
Pour le programme sur TI 82 on suppose que la fonction $(x \longmapsto x^3 + x + 1)$ a été préalablement entrée dans la variable Y1.

Programmation sur TI 82

```
:Prompt A
:Prompt B
:Prompt E
:If Y1(A)*Y1(B)≥0
:Then
:Disp("PAS DE RACINE")
:Else
:While B-A≥E
:(A+B)/2→C
:If Y1(A)*Y1(C)⩽0
:Then
:C→B
:Else
:C→A
:End
:End
:Disp("A=",A)
:Disp("B=",B)
```

Programmation en Python 2.6

```
def f(x):
    return x**3+x+1
A=input('A=')
B=input('B=')
E=input('Précision =')
if f(A)*f(B)>=0 :
    print('pas de racine entre ',A,' et ',B)
else:
    while B-A>=E :
        C=(A+B)/2.
        if f(A)*f(C)⩽0 :
            B=C
        else :
            A=C
    print('Une racine entre ',A, 'et ',B)
```

Note
Pour une étude plus générale de problèmes liés à la dichotomie vous pouvez consulter les documents suivants disponibles sur le site de l'IREM d'Aix-Marseille :http://www.irem.univ-mrs.fr/

- Une méthode pour élaborer des algorithmes itératifs. Auteur : F.Didier.
- Algorithmes et logique au lycée. Auteurs : P.Bouttier, A.Crumière, F.Didier, J-M.Fillia, M.Quatrini, H.Roland.

Figure 60 (L'algorithme de dichotomie (Roland, 2010) d'après un document IREM)

Nous pouvons observer que peu de dévolution est laissée à l'élève, si l'enseignant prend la situation proposée « clé en main ». Comme pour les documents *ressources*, cette fiche est pour l'enseignant et non l'élève. Les conditions suffisantes du TVI sont présentées comme « subordonnant » l'existence d'une solution unique.

3.2 Au niveau de la Première Scientifique

Comme pour le niveau Seconde, nous faisons le choix de ne sélectionner qu'un nombre limité de manuels scolaires en mathématiques : *Hatier* avec la collection *Odyssée* et *Belin* avec la collection *Symbole*. Nous considérons qu'il s'agit d'un éventail suffisamment large, qui peut nous permettre de situer nos choix pour notre ingénierie parmi les activités qui peuvent être proposés aux élèves dans le domaine de l'*Analyse*.

Ainsi, pour ces deux manuels, notre objectif est d'étudier le type d'exercices proposés sur les algorithmes, en particulier d'approximation, et son introduction tant dans la partie cours que dans celle des activités proposées. Pour cela, nous nous limitons à une présentation du cours et à un repérage des tâches liées à notre problématique.

Nous faisons aussi le point sur la façon dont d'autres ressources, telles que celles disponibles sur *Eduscol* ou élaborées dans les IREM considèrent l'*algorithme de dichotomie* en Première Scientifique.

3.1.1 Le manuel Odyssée (Edition 2011)

Ce manuel est constitué de trois parties découpées en plusieurs chapitres, précédés de deux parties introductives et transversales aux domaines mathématiques étudiés : une première sur le « *raisonnement logique* » et une seconde sur l'« *algorithmique* ».

La partie sur l'algorithmique permet de revoir les compétences sur l'algorithmique abordées en Seconde, comme les *instructions* avec l'*affectation*, l'*instruction conditionnelle*, la *répétition en boucle*, la *boucle itérative*. Pour chacune de ces instructions de nombreux exemples sont proposés. Ces connaissances sont complétées de plusieurs sections sur l'application de l'algorithmique aux divers domaines mathématiques étudiés en Première. Par exemple, dans celui intitulé « *Fonctions ou procédures* », les élèves peuvent aussi y trouver un exemple de décomposition d'un algorithme complexe en sous-algorithmes plus simples à lire et à interpréter auxquels l'algorithme principal peut y faire plusieurs fois appels.

Les autres chapitres du manuel sont composés de deux parties. Une première présentant une *ouverture* au chapitre complétée d'*activités* introductives permettant le réinvestissement

des compétences anciennes, suivi d'un *cours* accompagné d'exemples facilitant les nouvelles acquisitions. Cette partie ce complète de *savoir-faire*, de *travaux pratiques* autour de l'outil informatique, puis de pages *culture maths*. La deuxième partie de chacun des chapitres est constituée de nombreux *exercices d'application*, d'une page « *Se tester sur…* » où l'élève peut vérifier son niveau d'acquisition et de compréhension des nouveaux concepts étudiés dans le chapitre et de « *Problèmes* » dont certains sont issus de la vie courante ou en lien avec d'autres disciplines scientifiques.

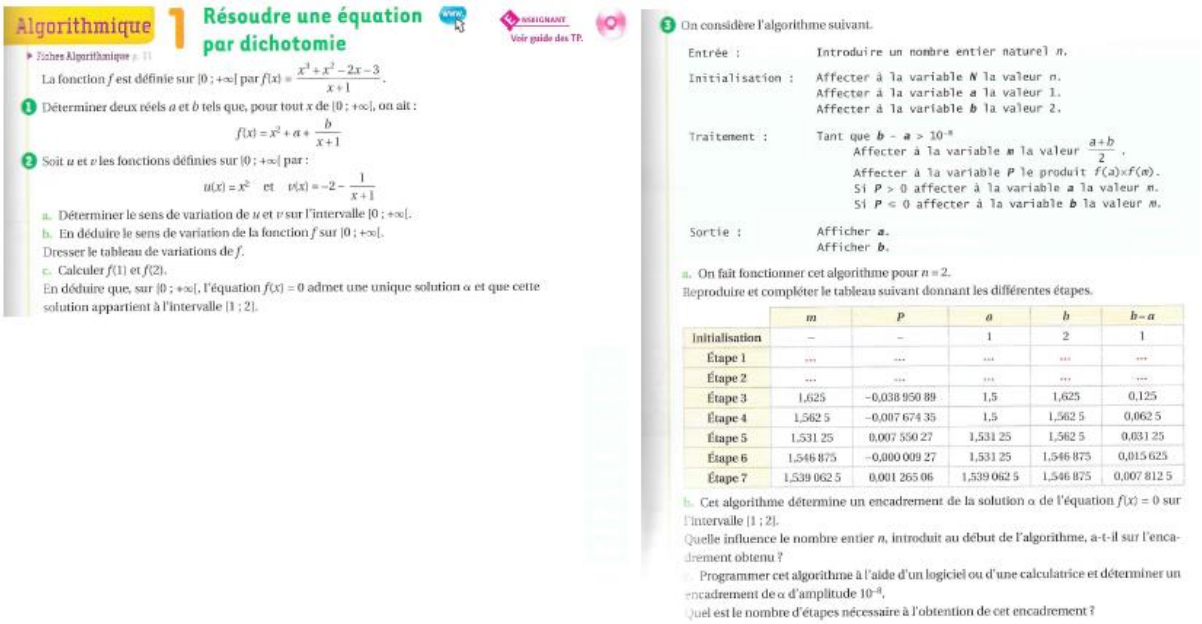
La partie sur l'*analyse* est découpée en quatre chapitres nommés respectivement « *Second degré* », « *Etude de fonctions* », « *Dérivation* » et « *Les suites* ». Seul le chapitre sur « *Etude de fonction*s » propose dans la rubrique « *Travaux pratiques* », une tâche en lien avec la problématique « *Résoudre une équation par dichotomie* » (fig. 61). En effet, une fonction rationnelle *f* définie sur intervalle de R est donnée par son expression algébrique et dans la partie concernant la résolution de l'équation $f(x) = 0$, il est demandé aux élèves de justifier que cette équation admet bien une unique solution sur le domaine où est défini la fonction. Pour cela, l'élève est amené à répondre à plusieurs questions conduisant à dresser le tableau de variation de la fonction *f* à l'aide d'une décomposition de la fonction *f* en deux sous-fonctions dont il est possible de déterminer le sens de variation en utilisant des propriétés sur les fonctions associées et d'en déduire ainsi le sens de variation de la fonction *f* sur son ensemble de définition. Cet exercice étant proposé avant le chapitre sur la dérivation, les auteurs n'attendent pas ici que les élèves étudient le sens de variation de la fonction *f* à l'aide du signe de la dérivée de la fonction. Ceci s'inscrit dans l'esprit du programme de Première Scientifique (Tableau n° 22) où il est demandé que l'élève sache aussi déterminer le sens à l'aide de considérations opératoires sur les fonctions associées.

Contenus	**Capacités attendues**	**Commentaires**
Étude de fonctions […] Sens de variation des fonctions $u + k$, $\lambda\, u$, $\sqrt{u}$ et $\frac{1}{u}$, la fonction u étant connue, k étant une fonction constante et λ un réel.	[…] • Exploiter ces propriétés pour déterminer le sens de variation de fonctions simples.	[…] On nourrit la diversité des raisonnements travaillés dans les classes précédentes en montrant à l'aide de contre-exemples qu'on ne peut pas énoncer de règle générale donnant le sens de variation de la somme ou du produit de deux fonctions. […]

Tableau 22 (Extrait du programme de Première Scientifique)

Ensuite les élèves doivent en déduire que l'équation $f(x) = 0$ admet une unique solution

sur son ensemble de définition. Puis, les élèves déterminent les valeurs de deux entiers consécutifs afin d'en déduire un intervalle d'amplitude 1 contenant cette solution. Une fois cette justification faite, un algorithme écrit dans un langage algorithmique est proposé. Les élèves ont pour consigne de tester cet algorithme pour une amplitude fixée afin de compléter les deux premières étapes de l'algorithme dont les résultats des étapes cinq étapes suivantes sont donnés. Ces résultats étant présentés par une capture d'écran représentant une page de tableur, il est probable que les auteurs attendent de l'élève qu'il teste l'algorithme à « la main ». A ce niveau de l'exercice aucune précision sur le rôle de l'algorithme n'est indiquée. Cependant, une fois le tableau complété, dans la question qui suit, les auteurs précisent la nature de l'algorithme. Ils demandent alors de préciser à quoi correspond la valeur du nombre entier *n* d'une des données variables et qu'elle est son influence sur l'encadrement de la solution. Cet exercice se termine par la demande d'implémenter cet algorithme dans un environnement numérique et de le tester pour une amplitude donnée, en précisant le nombre d'étapes nécessaire à l'obtention de l'encadrement.



Algorithmique

1 Résoudre une équation par dichotomie

▸ Fiches Algorithmique p. 11

ASSIGNANT Voir guide des TP.

La fonction f est définie sur $[0 ; +\infty[$ par $f(x) = \dfrac{x^3+x^2-2x-3}{x+1}$.

1. Déterminer deux réels a et b tels que, pour tout x de $[0 ; +\infty[$, on ait :
$$f(x) = x^2 + a + \frac{b}{x+1}$$

2. Soit u et v les fonctions définies sur $[0 ; +\infty[$ par :
$$u(x) = x^2 \quad \text{et} \quad v(x) = -2 - \frac{1}{x+1}$$

a. Déterminer le sens de variation de u et v sur l'intervalle $[0 ; +\infty[$.
b. En déduire le sens de variation de la fonction f sur $[0 ; +\infty[$.
Dresser le tableau de variations de f.
c. Calculer $f(1)$ et $f(2)$.
En déduire que, sur $[0 ; +\infty[$, l'équation $f(x) = 0$ admet une unique solution α et que cette solution appartient à l'intervalle $[1 ; 2]$.

3. On considère l'algorithme suivant.

```
Entrée :        Introduire un nombre entier naturel n.
Initialisation : Affecter à la variable N la valeur n.
                 Affecter à la variable a la valeur 1.
                 Affecter à la variable b la valeur 2.
Traitement :    Tant que b - a > 10^-N
                     Affecter à la variable m la valeur (a+b)/2.
                     Affecter à la variable P le produit f(a)×f(m).
                     Si P > 0 affecter à la variable a la valeur m.
                     Si P < 0 affecter à la variable b la valeur m.
Sortie :        Afficher a.
                Afficher b.
```

a. On fait fonctionner cet algorithme pour $n = 2$.
Reproduire et compléter le tableau suivant donnant les différentes étapes.

	m	P	a	b	$b-a$
Initialisation	–	–	1	2	1
Étape 1	...	...	...	...	...
Étape 2	...	...	...	...	...
Étape 3	1,625	−0,038 950 89	1,5	1,625	0,125
Étape 4	1,562 5	−0,007 674 35	1,5	1,562 5	0,062 5
Étape 5	1,531 25	0,007 550 27	1,531 25	1,562 5	0,031 25
Étape 6	1,546 875	−0,000 009 27	1,531 25	1,546 875	0,015 625
Étape 7	1,539 062 5	0,001 265 06	1,539 062 5	1,546 875	0,007 812 5

b. Cet algorithme détermine un encadrement de la solution α de l'équation $f(x) = 0$ sur l'intervalle $[1 ; 2]$.
Quelle influence le nombre entier n, introduit au début de l'algorithme, a-t-il sur l'encadrement obtenu ?
c. Programmer cet algorithme à l'aide d'un logiciel ou d'une calculatrice et déterminer un encadrement de α d'amplitude 10^{-9}.
Quel est le nombre d'étapes nécessaire à l'obtention de cet encadrement ?

Figure 61 (Extrait d'un exercice du manuel Odyssée de 1ère Scientifique : Résolution d'une équation par dichotomie)

A ce niveau de la scolarité, il est possible de mieux voir les différentes articulations possibles entre ETM spécifique et ETA. Pour cela, nous proposons de revenir sur certains points concernant l'exercice, ainsi que sur les *paradigmes algorithmiques* concernés (Tableau n° 23).

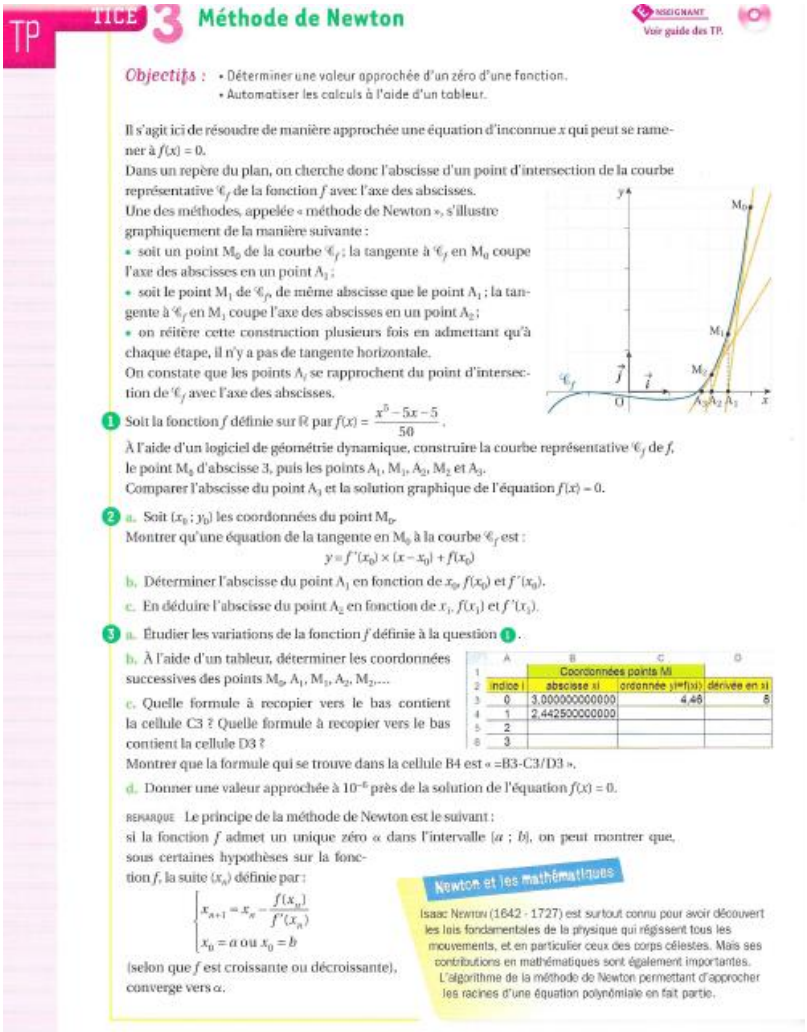
<table>
<tr><td>Thème de l'exercice</td><td colspan="2">Résolution d'une équation et méthode de dichotomie : « application numérique » (Fig. 61)</td></tr>
<tr><td>Paradigmes</td><td colspan="2">Les deux premières questions ont pour objectif de justifier que l'équation $f(x) = 0$ admet une unique solution α sur un sous-intervalle d'amplitude 1 de l'ensemble de définition de la fonction f, à partir de propriétés issues du domaine de l'analyse.

La question 3. a pour objectif de déterminer une approximation de cette solution à l'aide de raisonnement se situant dans le domaine de l'algorithmique. Les élèves ayant l'algorithme écrit en langage naturel, doivent dans un premier temps le tester au « papier-crayon » afin de compléter un tableau où sont précisés les étapes et le rôle des différentes colonnes. A ce stade de la question, nous nous situons au niveau 1 de l'algorithmique. En effet, les élèves n'ont pas à justifier la terminaison de l'algorithme, ni sa validité. Cependant, dans la seconde partie de cette question, les élèves devant aborder la question de l'influence du nombre entier n sur l'encadrement obtenu, les élèves sont alors confrontés à une analyse du problème les situant au niveau 2 de l'algorithmique.</td></tr>
<tr><td rowspan="2">Espaces de Travail spécifiques</td><td>ETA</td><td>ETM</td></tr>
<tr><td>Pour compléter le tableau donné, les élèves pouvant se limiter à un travail au papier-crayon sur l'algorithme sans pour autant justifier leurs réponses, nous observons alors que le travail de l'élève renvoie à des genèses instrumentale/sémiotique.
Cependant, la question portant sur l'influence de l'entier n sur l'encadrement permet une analyse plus fine de l'algorithme par l'élève et ainsi de faire que nous nous trouvions dans des genèses sémiotique/discursive.</td><td>Les deux premières questions de l'exercice font que les élèves doivent articuler deux ETM spécifiques : un d'algèbre pour le calcul des valeurs a et b de la nouvelle expression algébrique de la fonction et l'autre d'analyse pour l'étude du sens de variation de la fonction se situant au niveau 2 de l'$ETM_{analyse}$. Ainsi, les élèves articulent des genèses sémiotique/discursive de l'algèbre et de l'analyse.
Cependant, nous observons que la tâche se situant dans le domaine de l'algorithmique ne permet pas une réelle articulation entre un ETA et des ETM spécifiques.</td></tr>
<tr><td>Implication pour notre ingénierie</td><td colspan="2">Nous constatons que :
1. pour l'essentiel du travail proposé dans cet exercice (fig. 5a), celui-ci reste au niveau 1 en algorithmique : itération « naturelle », pas de problématique de terminaison,…</td></tr>
</table>

	Tout est prédéfini. Le travail algorithmique est du type « application numérique » de la problématique mathématique qui par contre se situe au niveau 2 de l'analyse ; 2. cependant, une approche du niveau 2 de l'algorithmique peut être observée lors de l'étude de l'influence de la condition de sortie de l'algorithme ; 3. les genèses instrumentale/sémiotique dans l'*ETA* viennent supporter la genèse discursive dans l'*ETM*. *Dans notre ingénierie, nous souhaitons approfondir le choix des structures de boucle mises en place, ainsi que la construction des tests de sortie de ces boucles.*

Tableau 23

Ce manuel propose aussi dans le chapitre qui suit et portant sur la dérivation une deuxième tâche utilisant des outils TICE (fig. 62) autour du calcul d'une valeur approchée d'un zéro d'une fonction mettant en place la *méthode de Newton* basée sur le concept de dérivée.



TP TICE 3 Méthode de Newton

ENSEIGNANT Voir guide des TP.

Objectifs : • Déterminer une valeur approchée d'un zéro d'une fonction.
• Automatiser les calculs à l'aide d'un tableur.

Il s'agit ici de résoudre de manière approchée une équation d'inconnue x qui peut se ramener à $f(x) = 0$.
Dans un repère du plan, on cherche donc l'abscisse d'un point d'intersection de la courbe représentative $\mathscr{C}_f$ de la fonction f avec l'axe des abscisses.
Une des méthodes, appelée « méthode de Newton », s'illustre graphiquement de la manière suivante :
• soit un point M_0 de la courbe $\mathscr{C}_f$; la tangente à $\mathscr{C}_f$ en M_0 coupe l'axe des abscisses en un point A_1 ;
• soit le point M_1 de $\mathscr{C}_f$, de même abscisse que le point A_1 ; la tangente à $\mathscr{C}_f$ en M_1 coupe l'axe des abscisses en un point A_2 ;
• on réitère cette construction plusieurs fois en admettant qu'à chaque étape, il n'y a pas de tangente horizontale.
On constate que les points A_i se rapprochent du point d'intersection de $\mathscr{C}_f$ avec l'axe des abscisses.

1. Soit la fonction f définie sur $\mathbb{R}$ par $f(x) = \dfrac{x^5 - 5x - 5}{50}$.
À l'aide d'un logiciel de géométrie dynamique, construire la courbe représentative $\mathscr{C}_f$ de f, le point M_0 d'abscisse 3, puis les points A_1, M_1, A_2, M_2 et A_3.
Comparer l'abscisse du point A_3 et la solution graphique de l'équation $f(x) = 0$.

2. a. Soit $(x_0 ; y_0)$ les coordonnées du point M_0.
Montrer qu'une équation de la tangente en M_0 à la courbe $\mathscr{C}_f$ est :
$y = f'(x_0) \times (x - x_0) + f(x_0)$
b. Déterminer l'abscisse du point A_1 en fonction de x_0, $f(x_0)$ et $f'(x_0)$.
c. En déduire l'abscisse du point A_2 en fonction de x_1, $f(x_1)$ et $f'(x_1)$.

3. a. Étudier les variations de la fonction f définie à la question 1.
b. À l'aide d'un tableur, déterminer les coordonnées successives des points M_0, A_1, M_1, A_2, M_2,...
c. Quelle formule à recopier vers le bas contient la cellule C3 ? Quelle formule à recopier vers le bas contient la cellule D3 ?

	A	B	C	D
1		Coordonnées points Mi		
2	indice i	abscisse xi	ordonnée yi=f(xi)	dérivée en xi
3	0	3,000000000000	4,46	8
4	1	2,442500000000		
5	2			
6	3			

Montrer que la formule qui se trouve dans la cellule B4 est « =B3-C3/D3 ».
d. Donner une valeur approchée à 10^{-6} près de la solution de l'équation $f(x) = 0$.

REMARQUE Le principe de la méthode de Newton est le suivant :
si la fonction f admet un unique zéro α dans l'intervalle $[a ; b]$, on peut montrer que, sous certaines hypothèses sur la fonction f, la suite (x_n) définie par :

$$\begin{cases} x_{n+1} = x_n - \dfrac{f(x_n)}{f'(x_n)} \\ x_0 = a \text{ ou } x_0 = b \end{cases}$$

(selon que f est croissante ou décroissante), converge vers α.

Newton et les mathématiques

Isaac Newton (1642 - 1727) est surtout connu pour avoir découvert les lois fondamentales de la physique qui régissent tous les mouvements, et en particulier ceux des corps célestes. Mais ses contributions en mathématiques sont également importantes. L'algorithme de la méthode de Newton permettant d'approcher les racines d'une équation polynômiale en fait partie.

Figure 62 (Extrait d'un exercice du manuel Odyssée de 1ère Scientifique : Méthode de Newton)

L'exercice ne met pas explicitement une étude autour d'une stratégie de type algorithmique, mais renvoie l'élève à une utilisation du tableur. Ainsi, contrairement à

l'exercice précédent, le terme « algorithme » n'est pas utilisé dans ce second exercice. Les auteurs emploient ainsi le terme de « méthode ».

Pour conclure sur ce manuel, nous observons aussi qu'aucun exercice ne mettant en place une étude en termes de vitesse de convergence des deux processus d'approximation (méthode de dichotomie et méthode de Newton) n'est proposé par les auteurs.

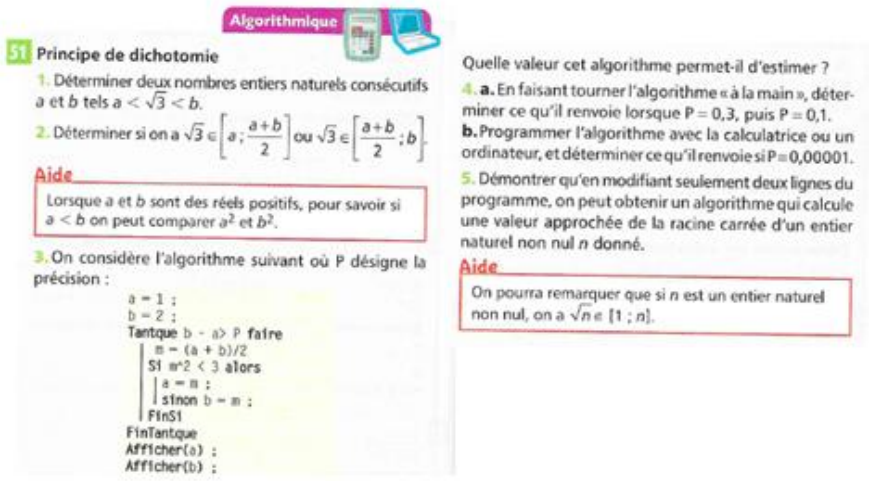
3.1.2 Le manuel Symbole (Edition 2011)

Ce manuel est découpé en dix chapitres traitant l'ensemble du programme de Première Scientifique. Il est aussi complété d'une partie indépendante intitulée « *Guide pratique – Algorithmique et TICE* ». La partie « *Algorithmique* » est découpée en quatre sous-parties traitant de *la notion d'état de la mémoire et de variable*, des *quatre instructions*, des *principaux types de données* et de l'écriture de programmes.

Chaque chapitre du manuel est découpé en plusieurs sections présentant entre autres des *activités d'introduction* suivies d'un *cours* énonçant les nouvelles propriétés et leurs démonstrations. Celles-ci sont complétées de *capacités attendues* qu'illustrent divers exercices résolus pour aborder les savoir-faire des concepts étudiés mais aussi des savoir-faire algorithmique en lien avec les nouveaux objets mathématiques abordés. Dans la section « *capacités attendues* », des *travaux pratiques* constitués de *TP algorithmiques* permettent à l'élève d'expérimenter les nouvelles notions apprises à l'aide de logiciels divers. Chacun des chapitres se terminent par une série d'exercices de difficultés croissantes.

Dans le chapitre « *Variations d'une fonction* » sont traités les *Fonctions de référence* et les *Opérations sur une fonction et variations*. Dans la partie exercices du chapitre, un exercice intitulé « *Principe de dichotomie* » est signalé comme faisant partie de la famille des exercices algorithmiques. Dans cet exercice (fig. 63), l'algorithme est donné en langage algorithmique. Il permet de déterminer un encadrement de $\sqrt{3}$ en appliquant la *méthode de dichotomie* sur l'intervalle initial [1 ; 2] (intervalle que doit déterminer l'élève). Puis, il est proposé un processus de calcul proposant une découpe en deux sous-intervalles de même amplitude selon la *méthode de dichotomie*, faisant intervenir la moyenne des deux bornes 1 et 2, afin de déterminer un intervalle d'amplitude plus petite que la précédente contenant $\sqrt{3}$. Pour cela, il est proposé une « Aide » de la part des auteurs que « *lorsque a et b sont des réels positifs, pour savoir si a < b, on peut comparer a² et b²* ». Cette aide est aussi mise en pratique dans l'algorithme donné par les auteurs car la fonction carrée $x \mapsto x^2$ est de nouveau exploité pour

le processus itératif décrit dans la question 2.



Algorithmique

51 Principe de dichotomie

1. Déterminer deux nombres entiers naturels consécutifs a et b tels $a < \sqrt{3} < b$.

2. Déterminer si on a $\sqrt{3} \in \left[a ; \frac{a+b}{2}\right]$ ou $\sqrt{3} \in \left[\frac{a+b}{2} ; b\right]$.

Aide

Lorsque a et b sont des réels positifs, pour savoir si $a < b$ on peut comparer a^2 et b^2.

3. On considère l'algorithme suivant où P désigne la précision :

```
a = 1 ;
b = 2 ;
Tantque b - a> P faire
    m = (a + b)/2
    SI m^2 < 3 alors
        a = m ;
    sinon b = m ;
    FinSI
FinTantque
Afficher(a) ;
Afficher(b) ;
```

Quelle valeur cet algorithme permet-il d'estimer ?

4. **a.** En faisant tourner l'algorithme « à la main », déterminer ce qu'il renvoie lorsque P = 0,3, puis P = 0,1.
b. Programmer l'algorithme avec la calculatrice ou un ordinateur, et déterminer ce qu'il renvoie si P = 0,00001.

5. Démontrer qu'en modifiant seulement deux lignes du programme, on peut obtenir un algorithme qui calcule une valeur approchée de la racine carrée d'un entier naturel non nul n donné.

Aide

On pourra remarquer que si n est un entier naturel non nul, on a $\sqrt{n} \in [1 ; n]$.

Figure 63 (Extrait d'un exercice du manuel Symbole de 1ère Scientifique : Principe de dichotomie)

Les élèves doivent ensuite déterminer le rôle de l'algorithme sans qu'il leur soit proposé de le programmer sur machine dans un premier temps, puis le faire « *tourner à la main* » afin d'aller vers une amplitude de plus en plus petite de l'intervalle d'encadrement. Les premières amplitudes demandées sont du ressort d'une application « manuelle » de l'algorithme. Cependant, les auteurs souhaitant obtenir une amplitude d'ordre 10^{-5}, ils demandent aux élèves de le programmer dans un environnement numérique afin de le tester. Conformément aux directives du programme, aucun langage informatique particulier n'est demandé. Les élèves sont libres de leur choix du langage informatique et par conséquent du choix l'environnement numérique. L'exercice se termine par une demande de modification de l'algorithme, avec une précision du nombre de lignes concernées sans pour autant les citer, afin d'adapter l'algorithme au calcul d'une valeur approchée de la racine carrée d'un entier naturel non nul donné.

Afin de mieux observer les différentes articulations possibles entre *ETM* spécifique et *ETA*, nous proposons de revenir sur certains points concernant l'exercice, ainsi que sur les *paradigmes algorithmiques* concernés (Tableau n° 24).

Thème de l'exercice	Valeur approchée de la racine carrée d'un entier naturel non nul donné et principe de dichotomie (Fig. 63)
Paradigmes	Les deux premières questions ont pour objectif de déterminer un premier encadrement par deux nombres entiers de la racine carrée d'un nombre entier donné, puis de déduire de cet encadrement le premier sous-intervalle emboité encadrant la racine carrée du nombre entier choisi. Dans ces deux questions nous nous situons au niveau 1 de l'algorithmique.

<table>
<tr><td></td><td colspan="2">Dans la question 3, les élèves doivent interpréter en termes d'estimation un algorithme écrit en langage pseudo-code. Pour cela, ils doivent reconnaitre le rôle de chacune des variables informatiques et comprendre le test de sortie de la boucle « TantQue », ainsi que la condition « $m^2 < 3$ » dans l'instruction conditionnelle. Ainsi, le travail se situe à la frontière des niveaux 1 et 2 de l'algorithmique.
Dans la question 4, les élèves font tourner l'algorithme dans un premier temps « à la main », puis ils l'implémentent dans un environnement numérique, afin de le tester sur des précisions données. L'algorithme donné étant écrit en langage pseudo-code, la difficulté est d'ordre sémantique. En effet, les élèves vont devoir le traduire en fonction de la syntaxe de l'environnement numérique choisi. Pour cette question, il suffit que la variable générique de la précision P soit remplacée successivement par les différentes valeurs de P sans nécessairement garder créer un « appel » de l'algorithme afin de donner une valeur de P. Le travail attendu se situe ainsi au niveau 1 de l'algorithmique.
Dans la question 5, les élèves doivent modifier précisément deux lignes du programme afin de le rendre générique pour obtenir un encadrement de n'importe quelle racine carrée d'un nombre entier non nul n donné. Le travail se situe encore au niveau 1 de l'algorithmique.</td></tr>
<tr><td rowspan="2">Espaces de Travail spécifiques</td><td>ETA</td><td>ETM</td></tr>
<tr><td>Les élèves pouvant se limiter à un travail au « papier-crayon », puis à une implémentation de l'algorithme dans un environnement numérique en transcrivant la syntaxe du langage pseudo-code au langage machine sans pour autant justifier leurs réponses, nous observons alors que le travail de l'élève renvoie à des genèses instrumentale/sémiotique.
Les questions portant sur l'influence du nombre P désignant la précision de l'encadrement et le fait de rendre « générique » l'algorithme à un entier naturel n non nul, permettent cependant, une analyse plus fine de l'algorithme par l'élève et ainsi de faire que le travail puisse se situer aussi dans des genèses sémiotique/discursive.</td><td>Les deux premières questions de l'exercice font que les élèves doivent articuler deux ETM spécifiques : un d'algèbre pour la détermination des valeurs a et b du premier encadrement et l'autre d'analyse pour affiner cet encadrement. Ainsi, les élèves articulent des genèses sémiotique/discursive de l'algèbre et de l'analyse.
Cependant, nous observons que la tâche se situant dans le domaine de l'algorithmique ne permet pas une réelle articulation entre ETA et ETM spécifiques.</td></tr>
<tr><td>Implication pour notre ingénierie</td><td colspan="2">Nous constatons qu'il n'est pas attendu de la part des élèves qu'ils se questionnent explicitement quant à la validité de l'algorithme et de sa terminaison ainsi que de son efficacité. Ainsi, l'essentiel du travail reste au niveau 1 de l'algorithmique : itération « naturelle », pas de problématique de terminaison,… Tout est prédéfini. Le travail algorithmique est du type « application numérique » de la problématique mathématique qui par contre se situe au niveau 2 de l'analyse.
Cependant, une approche du niveau 2 de l'algorithmique peut être observée lors de</td></tr>
</table>

	l'étude de l'influence de la condition de sortie de la boucle « TantQue » et sa généralisation à un nombre entier naturel non nul quelconque. Les genèses instrumentale/sémiotique dans l'*ETA* viennent supporter la genèse discursive dans l'*ETM*. *Dans notre ingénierie, nous souhaitons approfondir le choix des structures de boucle mises en place, ainsi que la construction des tests de sortie de ces boucles.*

Tableau 24

Dans le chapitre « *Dérivation et applications* », les auteurs présentent dans la rubrique « *Travaux pratiques* », une méthode de résolution d'une équation $f(x) = 0$ basée sur les nouvelles connaissances acquises par les élèves de Première Scientifique : *dérivée* et *tangente*. Cet exercice (cf. annexe) présenté dans le cadre d'un enseignement de l'algorithmique, renvoie à la *méthode de Newton-Raphson*. Cette méthode n'étant pas prise en compte dans notre ingénierie, nous ne proposons pas une analyse du travail demandé dans cet exercice.

3.1.3 Autres ressources : Documents ressources – Documents IREM

Bien que l'institution propose un document ressource pour la Première Scientifique dans le champ de l'analyse, nous pouvons observer que la résolution approchée d'équations ne fait pas l'objet d'une étude spécifique introduisant des algorithmes comme la dichotomie ou la méthode de Newton. Lors de l'étude sur le programme (cf. le paragraphe 2.1.), nous avons constaté que ceci était conforme à l'esprit du programme tant sur l'enseignement des fonctions, malgré une introduction du calcul différentiel qui pourrait permettre de considérer la méthode de Newton, que sur l'enseignement des suites où on attend en particulier que l'élève puisse travailler sur des algorithmes (ou un tableur) *pour traiter des problèmes de comparaison d'évolutions et de seuils*[70]. En effet, il n'est pas précisé explicitement dans le programme, contrairement à celui de Seconde, qu'une compétence attendue de l'élève soit de travailler sur des méthodes d'approximation d'une solution d'une équation ou d'encadrement d'irrationnel, même si ce travail est abordé par un certain nombre de manuels, comme nous avons pu le rapporter précédemment.

Dans les documents proposés en ligne par les principaux IREM, nous n'avons pas trouvé de ressources en lien avec la résolution approchée d'équations. L'IREM de la Réunion par exemple se limite en analyse et algorithmique à des contenus en lien direct avec des objets

[70] Bulletin officiel spécial n° 9 du 30 septembre 2010.

du programme, comme la valeur absolue, les équations du second degré, le calcul de dérivées et de ses applications, et une application des suites numériques à des situations concrètes.

3.3 Au niveau de la Terminale Scientifique

Comme pour les niveaux de Seconde et de Première Scientifique, nous faisons le choix de ne sélectionner qu'un nombre limité de manuels scolaires en mathématiques pour notre étude : *Didier* avec la collection *Math'x*, *Hachette* avec la collection *Déclic* et *Belin* avec la collection *Radial*. Nous considérons qu'il s'agit d'un éventail assez large, qui va nous permettre de situer nos choix pour l'ingénierie parmi les travaux qui peuvent être proposés aux élèves dans le contexte de notre recherche. Pour ces trois manuels retenus, notre objectif est d'étudier le type d'exercices proposés sur les algorithmes d'approximation et son introduction tant dans la partie cours, que celle des activités proposées. Pour cela, nous nous limitons à une présentation du cours et à un repérage des tâches liées à notre problématique. Nous faisons aussi le point sur la façon dont d'autres ressources, telles que celles disponibles sur *Eduscol* ou élaborées dans les IREM considèrent les méthodes d'approximation au niveau de la Terminale Scientifique.

3.3.1 Le manuel Math'x (Edition 2012)

Le manuel commence par deux premières parties indépendantes du reste de l'ouvrage. La première intitulée « *Rappels d'algorithmique* » est découpée en quatre parties : – *Algorithme et programme informatique* ; – *Variables : entrée, sortie et affectation* ; – *Structure alternative : « Si… Alors… Sinon »* ; *Structures itératives : boucles*. Chacune de ces parties permettent au lecteur de revoir les principales bases de l'algorithmique vue au cours des deux années précédentes. La deuxième est constituée de plusieurs fiches présentant différents logiciels et environnements numériques. Ensuite, nous avons trois parties : *Analyse*, *Géométrie* et *Probabilités-Statistique*. Chacune de ces parties est découpée en plusieurs chapitres. La présentation de chaque chapitre est classique et diffère peu de celle présentée dans les manuels de Seconde et de Première : activités, cours, travaux pratiques et exercices. Chaque chapitre se termine par une partie intitulée *Accompagnement personnalisée* proposant des exercices sur l'organisation des connaissances et les démarches à élaborer, ainsi que des exercices d'approfondissement.

Conformément aux directives du programme, les auteurs proposent de nombreux

exercices en lien avec des méthodes d'approximation d'une solution d'une équation dans les chapitres d'analyse. Ainsi, dans le chapitre « *Etude de fonctions. Continuité et dérivabilité* », nous trouvons dans la partie « *Travaux pratiques* », un exercice (fig. 64) sur la comparaison des vitesses de convergence entre deux algorithmes d'approximation d'une solution d'équation de la forme $f(x) = 0$: l'*algorithme de dichotomie* et l'*algorithme de Newton-Raphson*. Bien que ces deux algorithmes soient normalement familiers des élèves à ce niveau de la scolarité, les auteurs rappellent les processus de calcul de chacune de ces méthodes, en présentant ces processus avec le formalisme des suites. L'expression algébrique de la fonction f utilisée dans le cadre de l'exercice est donnée, bien que les processus soient décrits de façon générique. Après une série de questions permettant d'obtenir la valeur approchée de la solution de l'équation $f(x) = 0$ sur un intervalle donné, à l'aide des deux algorithmes, les élèves doivent comparer ces deux algorithmes en termes de vitesse de convergence. Pour cela, il leur est proposé une formulation incomplète de l'*algorithme de dichotomie* en langage algorithmique, puis dans un encart intitulé « *Aide à la programmation* », diverses écritures en langage machine de ce même algorithme. Quant à la *méthode de Newton-Raphson*, les auteurs ne proposent aucun algorithme, tant partiel que complet, et attendent des élèves qu'ils programment sur leur calculatrice la suite (x_n) donnant les termes de chacune des abscisses x_{n+1} des points successifs d'intersection de l'axe des abscisses et des tangentes successives à la courbe représentative de la fonction f en son point d'abscisse x_n, après avoir justifié que cette suite (x_n) vérifie la relution de récurrence $x_{n+1} = x_n - f(x_n)/f'(x_n)$, où f est une fonction générique et dérivable en x_n. L'étude de la comparaison des deux algorithmes ce traduit par une approche algorithmique autour de la solution de l'équation $f(x) = 0$, dans le cas où f est la fonction donnée en début d'énoncé. En effet, les auteurs demandent à partir de quel rang obtient-on une première décimale exacte de cette solution (ici $\sqrt{2}$) par chacun des deux algorithmes. Cette question sous-entend que les élèves l'interprètent en terme de « tester les deux algorithmes ». Puis, les auteurs demandent de prolonger cette comparaison pour davantage de premières décimales exactes, afin dans déduire une conjecture sur les vitesses de convergence de ces deux algorithmes. Ensuite, les auteurs donnent une série de questions ayant pour objectifs de guider les élèves vers une justification formalisée en termes de suites et de distances de cette conjecture. Nous pouvons observer, que comme pour les niveaux de Seconde et de Première Scientifique, que l'algorithmique ne trouve sa place dans l'esprit des auteurs que comme outil d'*analyse numérique* et non comme objet de preuve. De

plus, les questions du choix des variables itératives et des structures algorithmiques ne semblent pas faire l'objet de questionnement chez les auteurs pour les élèves, en particulier dans le cas de l'*algorithme de dichotomie* dont la structure de l'algorithme est déjà donnée comme nous l'avons fait remarquer plus haut.

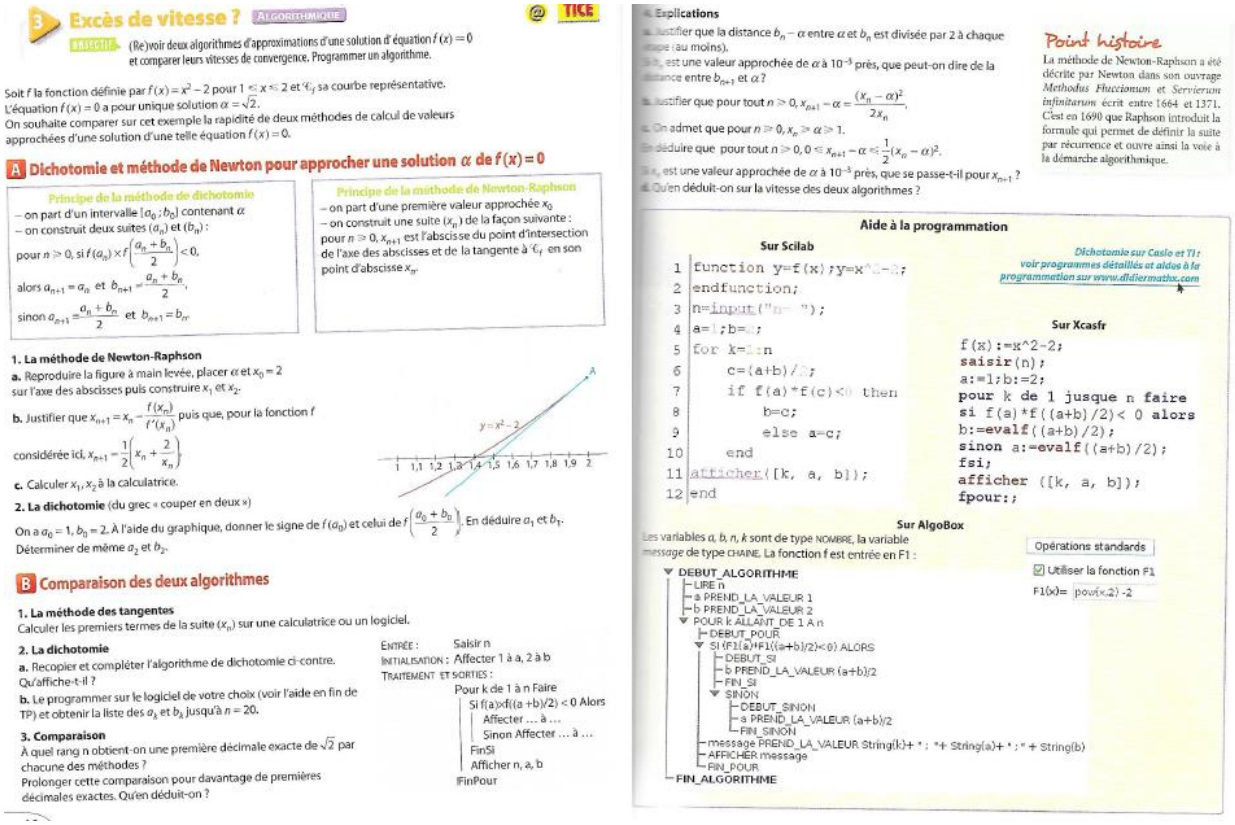


Figure 64 (Extrait d'un exercice du manuel Math'x de Terminale Scientifique : Comparaison de deux algorithmes)

Contrairement aux manuels de Seconde et de Première Scientifique, il est aussi à noter que ce manuel propose dans ce même chapitre, dans une rubrique intitulée « *Equation f(x) = k* » (fig. 65) de nombreux exercices, où il est demandé aux élèves de justifier par des considérations graphiques ou mathématiques de l'existence de solutions, du nombre de ces solutions, et dans certains exercices de la détermination des valeurs approchées de ces solutions en mettant en place des algorithmes afin de les implémenter dans des environnements numériques, puis de les tester.

Pour ces exercices de calculs de valeur approchée d'une solution de l'équation considérée, un algorithme écrit en langage algorithmique peut être proposé et modifiable à la demande des auteurs si celui-ci ne correspond pas à un de ceux connus des élèves, sinon les auteurs ne proposent que le nom de la méthode (*dichotomie* ou *Newton*) et laissent la liberté à l'élève de

construire l'algorithme dans le langage qu'il souhaite et qu'il le programme sur machine afin de le tester et d'obtenir un encadrement de la solution pour une amplitude fixée ou une valeur approchée à 10^{-n} près, où *n* est un entier naturel fixé par les auteurs.

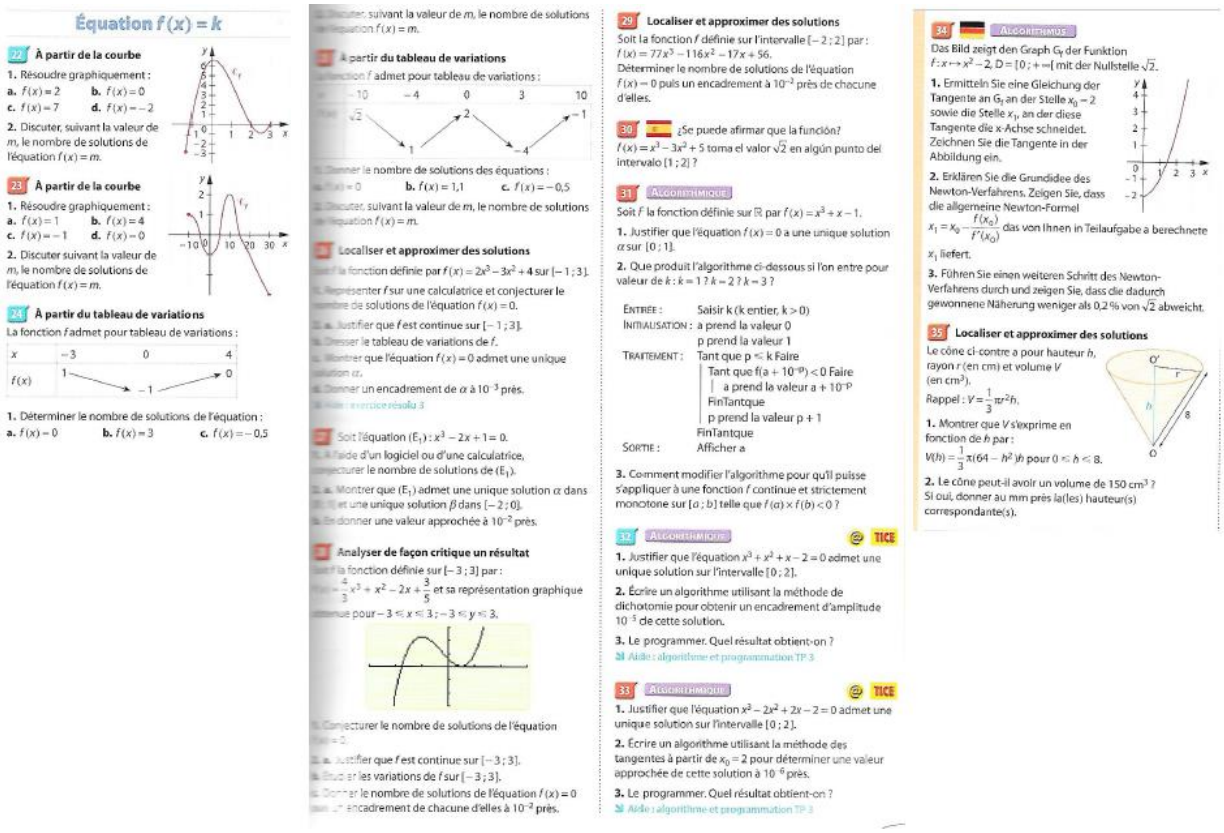


Figure 65 (Extrait d'une série d'exercices du manuel Math'x de Terminale Scientifique : Equations de la forme f(x) = k)

Là encore, nous pouvons observer que l'algorithmique renvoie à un travail dans le champ de l'*analyse numérique* sans pour autant amener l'élève à des questionnements guidés sur les choix d'une structure algorithmique et de la mise en place de variables itératives ne soient formulés ou guidés. Une totale dévolution semble ainsi laissée à l'élève sur ces questions algorithmiques.

Dans le chapitre « *Fonction exponentielle* », fonction nouvelle pour des élèves de Terminale Scientifique, dans la rubrique « *Travaux pratiques* », les auteurs proposent une tâche (fig. 66) sur l'*algorithme de dichotomie* afin d'approcher l'image d'un nombre réel *x* par la fonction $x \mapsto e^x$. Pour cela, les auteurs supposent que le nombre réel e est entré dans un calculateur qui ne possède que les opérations de base : +, –, ×, ÷ et √. On peut alors déterminer e^n, pour tout entier relatif *n* (en effet, $e^n = \underbrace{e \times e \times e \times \ldots \times e}_{n \text{ fois le nombre e}}$ et $e^{-n} = \frac{1}{e^n}$), mais se pose alors le problème d'obtenir des valeurs approchées de e^x pour un nombre réel *x* non entier. Les

auteurs font alors observer que pour tout nombre réel *x*, $e^x = e^{x-n} \times e^n$, avec *n* nombre entier car représentant la partie entière du réel *x* et $0 \leq x - n < 1$, il suffit de chercher à calculer des valeurs approchées de e^x pour $0 \leq x < 1$. Ensuite, en utilisant le registre algébrique, les élèves déterminent une nouvelle relation sur exp, donnant $\exp\left(\frac{a+b}{2}\right)$ en fonction de exp(*a*) et exp(*b*), pour tous nombres réels *a* et *b*. Puis, à l'aide du registre graphique, les élèves placent sur la courbe représentative de la fonction exp sur l'intervalle [*a* ; *b*], l'abscisse $\frac{a+b}{2}$, ainsi que son image $\exp\left(\frac{a+b}{2}\right)$. Enfin, à l'aide du registre analyse, justifie des encadrements images des intervalles $\left]a\ ;\ \frac{a+b}{2}\right[$ et $\left[\frac{a+b}{2}\ ;\ b\right[$ par la fonction exp avec *a* et *b* réels tels que $a < b$. Une fois que cette approche théorique basée sur des considérations mathématiques est faite, les auteurs demandent un travail en algorithmique afin de mettre en place un tel calculateur et de le tester après l'avoir implémenté dans un environnement numérique. Les élèves sont supposés reconnaître la *méthode de dichotomie* par le fait que les intervalles emboités qui interviennent sont obtenus par une coupure en deux sous-intervalles de même amplitude de l'intervalle précédent. L'algorithme écrit par les auteurs en langage naturel est incomplet, les élèves doivent vérifier que les conditions sur un encadrement de *x* et de son image par exp sont correctes lors de l'initialisation puis le compléter afin que ces encadrements soient vérifiés à chaque passage dans la boucle. La condition sur l'amplitude de l'intervalle n'étant pas forcément acquise par des élèves relativement débutant en informatique, nous observons que les auteurs de l'exercice demandent des précisions sur celle-ci. Enfin, les élèves doivent transposer cet algorithme en langage de programmation afin de le programmer puis de le tester pour une valeur donnée avec une amplitude choisie et interpréter le résultat obtenu. Nous pouvons aussi observer que dans les exercices de ce chapitre le choix des boucles reste toujours du ressort des auteurs et que peu de liberté dans cette prise d'initiative est laissée à l'élève, en total contraste avec le chapitre « *Etude de fonctions. Continuité et dérivabilité* » où aucune indication n'était donnée relativement à la structure et aux variables.

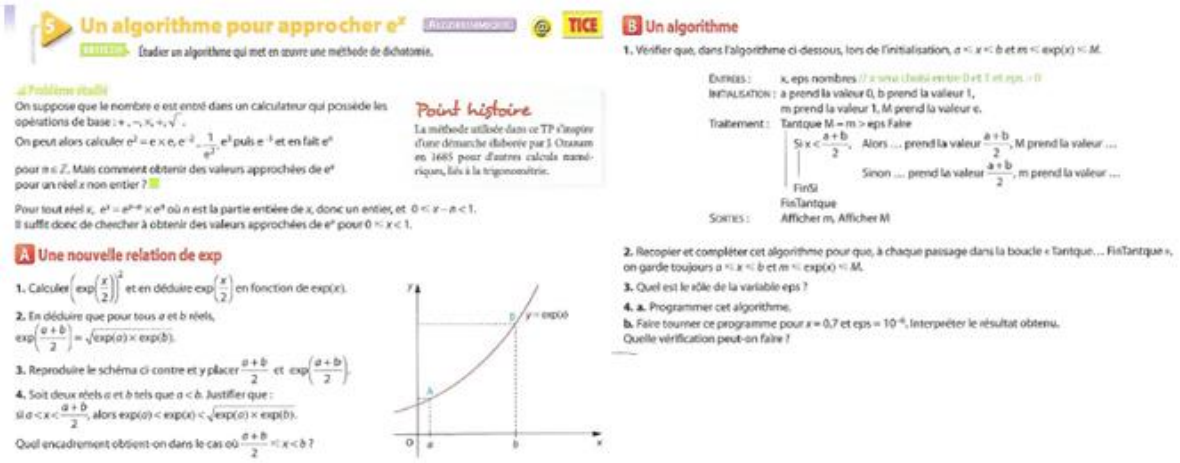


Figure 66 (Extrait d'un TP du manuel Math'x de Terminale Scientifique : l'algorithme de dichotomie afin d'approcher l'image d'un nombre réel x par la fonction $x \mapsto e^x$)

Dans le chapitre « Suites numériques », nous trouvons un exercice (fig. 67) permettant de de construire une *preuve de terminaison* de l'*algorithme de dichotomie « continue »* pour la détermination d'un encadrement d'amplitude donnée de l'unique solution de l'équation $f(x) = 0$, où f est une fonction générique vérifiant la continuité et la stricte monotonie (ici croissance) sur **R**, ainsi que le changement de signe sur un intervalle d'amplitude 1, donné de **R** (ici [1 ; 2]). Cette preuve se construit à l'aide de deux suites (a_n) et (b_n) adjacentes (termes inconnus des élèves de Terminale Scientifique dans le cadre du nouveau programme) dont les relations de récurrence sont données. Après que les élèves aient justifié la convergence de ces deux suites vers une même limite, les auteurs demandent de construire l'algorithme (de dichotomie) qui demande une amplitude *e* strictement positive et affiche un encadrement de l'unique solution d'amplitude *e*. Cependant, il n'est pas précisé de l'implémenter en machine afin de le tester. Toutefois, les auteurs demandent de conclure en justifiant que l'algorithme s'arrête bien pour toute valeur de *e* strictement positive choisie.

65 ALGORITHMIQUE

Justifier l'algorithme de dichotomie

La fonction f est continue et strictement croissante sur $\mathbb{R}$ avec $f(1) < 0$ et $f(2) > 0$. L'équation $f(x) = 0$ admet donc une unique solution α dans $[1 ; 2]$.

On construit deux suites (a_n) et (b_n) telles que :

$a_0 = 1$ et $b_0 = 2$ et, pour tout n de $\mathbb{N}$,

si $f(a_n) f\left(\frac{a_n + b_n}{2}\right) < 0$, alors $a_{n+1} = a_n$ et $b_{n+1} = \frac{a_n + b_n}{2}$,

sinon, $a_{n+1} = \frac{a_n + b_n}{2}$ et $b_{n+1} = b_n$.

Par construction, $1 \leqslant a_n \leqslant \alpha \leqslant b_n \leqslant 2$ pour tout $n \geqslant 0$.

1. a. Montrer que $(b_n - a_n)$ est une suite géométrique.

b. Déterminer l'expression de $b_n - a_n$ en fonction de n et sa limite quand n tend vers $+\infty$.

2. a. Montrer que la suite (a_n) est croissante puis qu'elle converge. Soit ℓ sa limite.

b. Montrer de même que (b_n) converge. Soit ℓ' sa limite.

c. Montrer que $\ell \leqslant \alpha$, $\alpha \leqslant \ell'$ et $\ell' - \ell = 0$.
Qu'en déduit-on pour les suites (a_n) et (b_n) ?

3. a. Écrire un algorithme qui demande une amplitude $e > 0$ et affiche un encadrement de α d'amplitude e.

b. Justifier que cet algorithme s'arrête bien pour tout valeur de e strictement positive.

Figure 67 (Extrait d'un exercice du manuel Math'x de Terminale Scientifique : Justifier l'algorithme de dichotomie)

On peut ainsi observer une approche de l'algorithme objet d'apprentissage (au sens de Douady) en Terminale Scientifique de l'algorithme de dichotomie qui n'était pas vue dans les classes antérieures bien que les connaissances sur les suites géométriques aient fait l'objet d'apprentissages en Première.

Afin de terminer la présentation de l'exercice de la figure 67, nous proposons de présenter une étude (tableau n° 25) des compétences mises en place en termes d'articulations d'*ETA* et d'*ETM* spécifiques, complétée d'une approche en fonction des niveaux de paradigmes algorithmiques utilisés.

Thème de l'exercice	Preuve de l'*algorithme de dichotomie* : analyse numérique (Fig. 67)
Paradigmes	Les élèves travaillent sur une fonction générique notée f et vérifiant certaines propriétés permettant d'appliquer le théorème de la bijection sur un intervalle donné, corollaire du TVI. Les deux premières questions ont pour objectif de déterminer deux suites (a_n) et (b_n) qui convergent vers une même limite qui correspond à l'unique solution α de l'équation $f(x) = 0$ dans l'intervalle donné. La question 3 consiste à construire un algorithme permettant de déterminer les valeurs respectives des suites (a_n) et (b_n) qui encadrent la solution α d'une équation de la forme $f(x) = 0$, pour une amplitude fixée, puis de justifier que cet algorithme

<table>
<tr><td></td><td colspan="2">s’arrête pour toute amplitude strictement positive. Ainsi, dans cette question le travail se situe au niveau 2 de l’algorithmique.</td></tr>
<tr><td rowspan="2">Espaces de Travail spécifiques</td><td>ETA</td><td>ETM</td></tr>
<tr><td>Les élèves travaillant une preuve de l’algorithme de dichotomie à l’aide de l’algorithmique et de deux suites adjacentes. On attend qu’ils justifient l’existence de ces suites et qu’ils construisent l’algorithme permettant de calculer les valeurs de ces deux suites pour une amplitude donnée. Ce travail nécessite de créer une boucle de la forme « TantQue » permettant de sortir de la boucle contenant une instruction conditionnelle permettant d’obtenir les différents termes des deux suites adjacentes. La construction de l’algorithme en langage naturel ou pseudo-code afin de l’implémenter dans un environnement numérique pour le tester pour une amplitude choisie par l’utilisateur, fait que le travail de l’élève renvoie à des genèses instrumentale/sémiotique. De plus, la preuve de la validité et de la terminaison de l’algorithme associée à une preuve de la méthode de dichotomie permet que le travail se situe aussi autour de genèses sémiotique/discursive.</td><td>L’ensemble des questions de l’exercice font que les élèves doivent articuler un ETM d’analyse et un ETA. En effet, dans les deux premières questions le travail attendu se situe dans un ETM d’analyse. Celui-ci permet la construction de l’algorithme et de justifier dans le cadre d’un ETA la validité et la terminaison de l’algorithme.
Contrairement aux observations faites lors des analyses des manuels de Seconde et de Première Scientifique, nous observons donc ici que les tâches demandées à l’élève font bien que nous avons une réelle articulation entre ETA et $ETM_{analyse}$.</td></tr>
<tr><td>Implication pour notre ingénierie</td><td colspan="2">Nous constatons qu’il est attendu de la part des élèves qu’ils se questionnent explicitement quant à la validité de l’algorithme et de sa terminaison. Ainsi, l’essentiel du travail reste au niveau 1 de l’algorithmique pour l’itération « naturelle » et le choix des structures de boucles, mais aussi au niveau 2 pour la problématique de terminaison.
Le travail algorithmique est donc du type « analyse numérique » de la problématique mathématique qui se situe elle-aussi au niveau 2 de l’analyse.
Les genèses instrumentale/sémiotique/discursive dans l’ETA viennent par conséquent supporter les genèses sémiotique/discursive dans l’ETM.
Cet exercice s’inscrit donc dans l’esprit de notre ingénierie, où nous souhaitons approfondir le choix des structures de boucle mises en place, ainsi que la construction des tests de sortie de ces boucles, mais aussi construire une preuve du TVI à l’aide de l’algorithme de dichotomie.</td></tr>
</table>

Tableau 25

3.3.2 Le manuel Déclic (Edition 2012)

Le manuel propose deux encarts sur l'algorithmique et la logique qui se trouvent en fin d'ouvrage. Le manuel est découpé en douze chapitres. Les six premiers chapitres portent sur l'ensemble des compétences requises à ce niveau scolaire dans le domaine de l'analyse. La structure de chaque chapitre est classique, composée de diverses parties : cours, travaux dirigés, exercices,... et se terminant par une partie permettant entre autres de faire le lien avec les autres sciences.

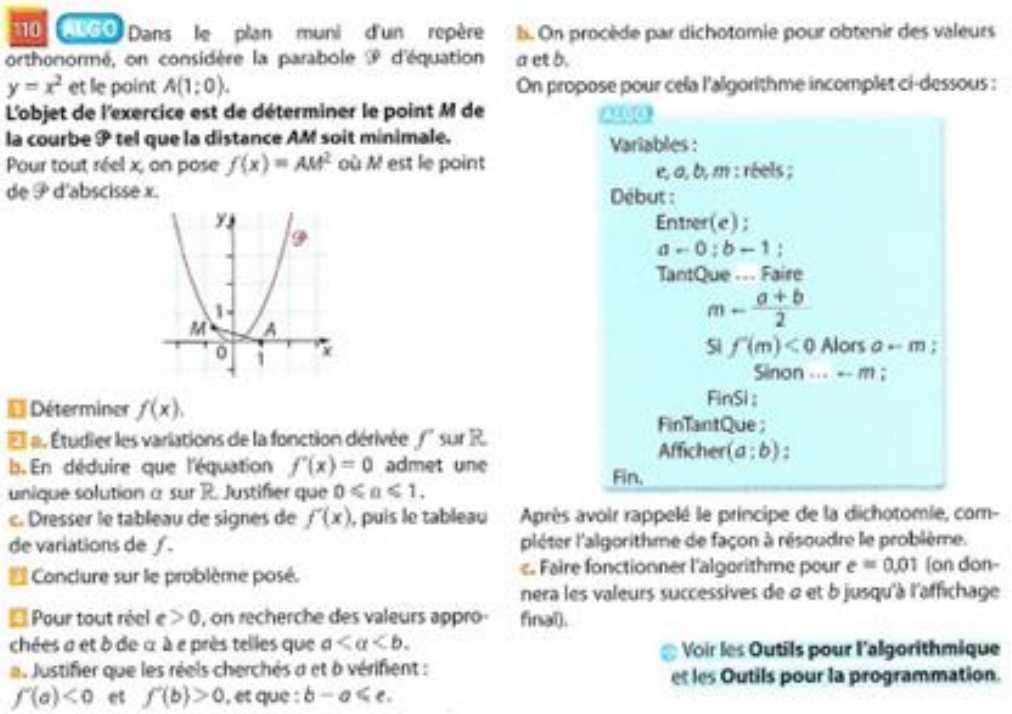
Dans la partie cours du chapitre « *Limites et fonctions continues* », à la section « *Continuité* », se trouve une sous-section intitulée : « *Théorème des valeurs intermédiaires* » (cf. annexe). Dans cette sous-section, le théorème est cité avec le commentaire « *Admis* » et un graphique illustre son contenu. En revanche, une démonstration de son corollaire, le *théorème de la bijection*, est donnée par les auteurs. Cette sous-section se termine par un point de méthodologie de recherche de la détermination d'une valeur approchée de la solution, sur un intervalle donné I, d'une équation du type $f(x) = k$. Cette méthode repose sur les étapes suivantes :

(1) vérifier que la fonction est continue sur l'intervalle I ;
(2) étudier les variations de la fonction sur I;
(3) s'assurer de l'existence d'au moins une solution en vérifiant que le nombre réel k appartient à l'intervalle $[m ; M]$, où m et M sont respectivement les minimum et maximum de la fonction de la fonction sur I ;
(4) déterminer ensuite une valeur approchée (ou un encadrement) de chacune des solutions de l'équation, en mettant en place, sur chacun des intervalles de monotonie un *algorithme de dichotomie* écrit en langage naturel ou pseudo-code.

Notons que, dans cette présentation, l'*algorithme de dichotomie* est présenté comme une simple « méthode », sans qu'il ne soit fait mention de propriétés des suites obtenues à partir des bornes de l'encadrement à la différence de plusieurs manuels. Clairement, l'algorithme ne met en jeu ni des savoirs en analyse ni des savoirs en algorithmique, mais sert seulement à une « application numérique ».

Dans la partie « *Exercices* » de ce chapitre, à la rubrique « *Prépa Bac* », les auteurs proposent un exercice (fig. 68) sur la détermination du point mobile M d'une parabole P d'équation $y = x^2$ telle que la distance AM, avec A(1 ; 0), soit minimale. La nouveauté de cet exercice par rapport à ceux qui ont été présentés jusqu'à maintenant, réside dans le fait que l'étude sur la recherche d'un encadrement d'une solution à une équation ne va pas se faire

sur la fonction *f* définie par *f*(*x*) = AM², où M est le point de P d'abscisse *x*, mais sur la dérivée de *f*, c'est-à-dire de l'unique solution de l'équation *f* '(*x*) = 0, *f* ' étant la fonction dérivée de *f* sur **R**, avant de conclure au problème posé initialement. Le choix de la méthode d'approximation de cet encadrement est imposé par les auteurs. En effet, ils procèdent par dichotomie et donnent un algorithme incomplet écrit en langage pseudo-code. Nous nous situons toujours dans le cadre d'une *application numérique*.



110 ALGO Dans le plan muni d'un repère orthonormé, on considère la parabole $\mathscr{P}$ d'équation $y = x^2$ et le point $A(1;0)$.

L'objet de l'exercice est de déterminer le point *M* de la courbe $\mathscr{P}$ tel que la distance *AM* soit minimale.

Pour tout réel x, on pose $f(x) = AM^2$ où M est le point de $\mathscr{P}$ d'abscisse x.

1 Déterminer $f(x)$.

2 a. Étudier les variations de la fonction dérivée f' sur $\mathbb{R}$.

b. En déduire que l'équation $f'(x) = 0$ admet une unique solution α sur $\mathbb{R}$. Justifier que $0 \leqslant \alpha \leqslant 1$.

c. Dresser le tableau de signes de $f'(x)$, puis le tableau de variations de f.

3 Conclure sur le problème posé.

4 Pour tout réel $e > 0$, on recherche des valeurs approchées a et b de α à e près telles que $a < \alpha < b$.

a. Justifier que les réels cherchés a et b vérifient : $f'(a) < 0$ et $f'(b) > 0$, et que : $b - a \leqslant e$.

b. On procède par dichotomie pour obtenir des valeurs a et b.

On propose pour cela l'algorithme incomplet ci-dessous :

ALGO

```
Variables :
    e, a, b, m : réels ;
Début :
    Entrer(e) ;
    a ← 0 ; b ← 1 ;
    TantQue ... Faire
        m ← (a + b)/2
        Si f'(m) < 0 Alors a ← m ;
            Sinon ... ← m ;
        FinSi ;
    FinTantQue ;
    Afficher(a ; b) ;
Fin.
```

Après avoir rappelé le principe de la dichotomie, compléter l'algorithme de façon à résoudre le problème.

c. Faire fonctionner l'algorithme pour $e = 0{,}01$ (on donnera les valeurs successives de a et b jusqu'à l'affichage final).

Voir les **Outils pour l'algorithmique** et les **Outils pour la programmation**.

Figure 68 (Extrait d'un exercice du manuel Déclic de Terminale Scientifique : Détermination du point mobile M d'une parabole P d'équation $y = x^2$ telle que la distance AM, où A(1 ; 0), soit minimale)

Nous souhaitons compléter l'analyse de cet exercice (Fig. 68) en termes de *paradigmes algorithmiques* et d'articulations entre *ETA* et *ETM*, que nous présentons de nouveau sous la forme d'un tableau (n° 26).

Thème de l'exercice	Un algorithme à explorer (fig. 68) – Une situation d'*analyse numérique*
Paradigmes	La partie algorithmique de l'exercice concerne la question 4. En effet, dans les trois premières questions, l'élève doit déterminer l'expression algébrique de la fonction *f*, puis répondre au problème posé à l'aide de raisonnements mathématiques qui font que l'élève travaille au niveau 2 des paradigmes de l'analyse. En ce qui concerne l'algorithmique, bien que les élèves aient un algorithme incomplet correspondant à la *méthode de dichotomie*, nous observons qu'ils doivent travailler sur le test de sortie de la boucle « TantQue ». Ainsi, nous nous situons aux niveaux 1 et 2 des paradigmes de l'algorithmique. En effet, le travail demandé à l'élève suppose une représentation des objets et des actions élémentaires et la construction d'un traitement :

<table>
<tr><td></td><td colspan="2">- Il faut distinguer le rôle des différentes variables informatiques et créer un test de sortie de la boucle « TantQue ».
- De plus, les élèves doivent interpréter la fonction « générique » f comme devant être représentée par son expression algébrique.
Cette représentation et ce traitement se situent dans un cadre « axiomatique », ce qui caractérise le niveau 2 :
- Les élèves doivent rappeler le principe de dichotomie (dans le cas « continu »).
- Il est demandé aux élèves de valider diverses inégalités concernant les valeurs approchées a et b de la solution de α à e près. Ceci est aussi mis en relation avec une propriété de terminaison de l'algorithme.
Dans cet exercice, l'algorithmique peut s'exercer moyennant une « axiomatisation partielle », voire des « îlots d'axiomatisation » (Kuzniak, Houdement, 2006).</td></tr>
<tr><td rowspan="2">Espaces de Travail spécifiques</td><td>ETA</td><td>ETM</td></tr>
<tr><td>L'élève doit expliciter le principe de dichotomie puis compléter un algorithme écrit en langage pseudo-code transposant ce principe. La fonction proposée dans l'algorithme donnée est générique. L'élève doit par conséquent remplacer cette fonction générique par l'expression algébrique de la fonction qu'il a déterminée dans la première partie de l'exercice. Il doit aussi compléter les « trous » de l'algorithme en tenant compte du fait que celui doit être représentatif de la méthode de dichotomie.
Un tableau type est aussi proposé avec la description de chacune des colonnes utilisées.
Le travail attendu dans cette question 3 fait que l'élève se déplace suivant les attendus de la sous-question dans des genèses instrumentale/sémiotique, des genèses instrumentale/discursive et dans des genèses sémiotique/discursive.
En effet, une genèse discursive qui porte sur une preuve de terminaison est envisagée.</td><td>Dans les trois premières questions, l'élève doit construire la fonction f permettant de répondre au problème posé, mais aussi justifier l'existence et l'unicité de la solution répondant au problème posé. Ceci se situe dans un ETM d'analyse pour l'existence et l'unicité de la solution, mais aussi dans des ETM de géométrie et d'algèbre pour la construction de la fonction.
Le travail demandé sur les inégalités concernant les valeurs approchées de la solution semble laisser penser que les auteurs souhaitent mener les élèves dans une genèse discursive dans les différents ETM qui va s'articuler avec celle de l'ETA correspondant au principe de dichotomie.
Ainsi, on peut s'attendre à ce que les genèses instrumentale et sémiotique développées dans l'ETA soient exploitées pour complémenter cette genèse discursive dans les ETM.</td></tr>
<tr><td>Implication pour notre ingénierie</td><td colspan="2">Nous constatons que :
1. Le travail reste au niveau 1 en algorithmique pour l'itération « naturelle », mais le questionnement sur la problématique de terminaison et la connaissance de la méthode de dichotomie font que nous avons un début d'« axiomatisation » permettant aussi un travail au niveau 2 de l'algorithmique.
2. Les genèses instrumentale/sémiotique dans l'ETA viennent ici supporter la genèse discursive dans les ETM, et engager une genèse discursive dans l'ETA.

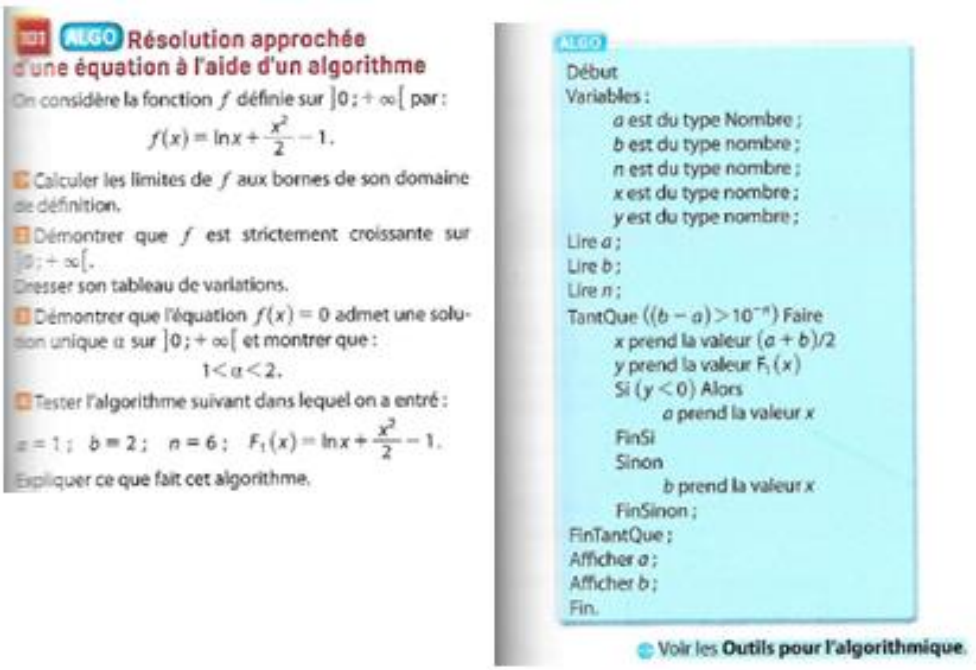
Nous rappelons que nous souhaitons, dans notre ingénierie, élaborer des tâches tirant parti des genèses instrumentale et sémiotique dans l'ETA et articulant des genèses discursives dans l'ETM et l'ETA, autour de la terminaison et de l'efficacité</td></tr>
</table>

	donc vers le niveau 2 de l'algorithmique. Ceci sera présenté dans les deux chapitres suivant.

Tableau 26

D'autres exercices dans ce chapitre sont aussi proposés sur la problématique d'approximation d'une solution d'une équation, sans qu'aucune méthode ou approche algorithmique du problème ne soit imposée voire conseillée. En effet, la dévolution de ces choix semble toujours laissée à l'élève. Cependant, pour tous ces exercices, nous nous situons dans un esprit d'*application numérique* de la méthode retenue par l'élève. En effet, seules des questions sur l'itération « naturelle » et le choix des structures de boucles semblent être demandées aux élèves. Les problématiques de terminaison et de validité des algorithmes utilisés semblent ne pas faire par partie des questionnements envisagés par les auteurs du manuel.

De même, nous observons qu'au chapitre « *Fonction logarithme népérien* », un exercice (fig. 69) sur la résolution approchée d'une équation à l'aide d'un algorithme est proposé. Cet exercice est dans l'esprit des autres exercices abordant ce thème d'approximation trouvés dans les différents manuels étudiés. En effet, la partie algorithmique arrive à la fin de l'exercice et se présente comme une *application numérique* à la détermination d'un encadrement d'une solution d'une équation. Cependant, les élèves doivent expliquer le rôle de l'algorithme donné en langage pseudo-code. Le rôle de cet algorithme vient après qu'il ait été demandé aux élèves de tester l'algorithme.



ALGO **Résolution approchée d'une équation à l'aide d'un algorithme**

On considère la fonction f définie sur $]0;+\infty[$ par :

$$f(x) = \ln x + \frac{x^2}{2} - 1.$$

1. Calculer les limites de f aux bornes de son domaine de définition.
2. Démontrer que f est strictement croissante sur $]0;+\infty[$.
Dresser son tableau de variations.
3. Démontrer que l'équation $f(x) = 0$ admet une solution unique α sur $]0;+\infty[$ et montrer que :
$1 < \alpha < 2$.
4. Tester l'algorithme suivant dans lequel on a entré :
$a = 1$; $b = 2$; $n = 6$; $F_1(x) = \ln x + \frac{x^2}{2} - 1$.
Expliquer ce que fait cet algorithme.

ALGO

```
Début
Variables :
    a est du type Nombre ;
    b est du type nombre ;
    n est du type nombre ;
    x est du type nombre ;
    y est du type nombre ;
Lire a ;
Lire b ;
Lire n ;
TantQue ((b − a) > 10^−n) Faire
    x prend la valeur (a + b)/2
    y prend la valeur F1(x)
    Si (y < 0) Alors
        a prend la valeur x
    FinSi
    Sinon
        b prend la valeur x
    FinSinon ;
FinTantQue ;
Afficher a ;
Afficher b ;
Fin.
```

Voir les **Outils pour l'algorithmique**.

Figure 69 (Extrait d'un exercice du manuel Déclic de Terminale Scientifique : Résolution d'une équation à l'aide d'un algorithme)

3.3.3 Le manuel Radial (Edition 2006)

Bien que ce manuel soit ancien et qu'il se réfère au programme précédent de Terminale Scientifique, il nous a semblé pertinent, dans le cadre de notre ingénierie, sur la dichotomie de présenter cet ouvrage sur les tâches et les compétences qu'il propose en lien avec la dichotomie. En effet, pendant la période de 2006 à 2012, le programme de Terminale Scientifique en mathématiques signalait, dans le cadre des modalités de mise en œuvre des contenus théoriques, que le *théorème des valeurs intermédiaires* pouvait être admis ou démontré à l'aide de suites adjacentes (celles-ci étaient au programme ainsi que les propriétés de convergence de ces suites). De plus, le programme indiquait que la démonstration du corollaire « *si f est une fonction continue strictement monotone sur [a; b], alors, pour tout réel k compris entre f (a) et f (b), l'équation f (x) = k a une solution unique dans [a ; b]* » faisait partie des démonstrations à connaitre de l'élève. Cette propriété pouvait faire l'objet d'une question de cours au baccalauréat, dans le cadre d'un R.O.C.[71]. Le programme de l'époque, signalait que ce corollaire pouvait être étendu au cas où la fonction *f* serait « *définie sur un intervalle ouvert ou semi-ouvert, borné ou non, les limites de f aux bornes de l'intervalle étant supposées connues* ». Pour compléter le cadre des modalités de mise en œuvre des contenus en lien avec

[71] N.B. : *Restitution Organisée des Connaissances*

le *théorème des valeurs intermédiaires*, le programme indiquait aussi qu'il était possible d'approcher la solution d'une équation du type $f(x) = k$ par dichotomie ou balayage avec la calculatrice ou l'ordinateur.

Pour des raisons de relative ancienneté de l'ouvrage, nous ne proposons pas une étude détaillée de cet ouvrage « ancien ». Nous nous contentons seulement de présenter deux exercices donnés par les auteurs de l'ouvrage dans le chapitre intitulé « *Suites et récurrences* » qui vient chronologiquement après les chapitres : « *Fonctions : limites et continuité* », « *Dérivation – Primitives d'une fonction* », « *La fonction exponentielle* », « *Fonction logarithme népérien et autres fonctions* ». En effet, ces deux exercices sont en étroit lien avec notre problématique sur la *dichotomie* et le *théorème des valeurs intermédiaires* que nous verrons lors de la présentation et l'étude de notre ingénierie.

Dans le premier exercice (fig. 70) intitulé « *Une démonstration du théorème des valeurs intermédiaires* », les auteurs rappellent dans un premier temps l'énoncé du TVI, puis pour une fonction f définie et continue sur un intervalle de **R**, ils construisent, par récurrence, deux suites adjacentes et après calcul de la limite commune de ces deux suites en déduisent la fin de la démonstration du TVI. Nous pouvons observer que les auteurs ne proposent aucune indication qui permettrait à l'élève de voir si une approche algorithmique de cette démonstration pourrait être une aide à la compréhension de la tâche demandée.

86 Une démonstration du théorème des valeurs intermédiaires

Le but de cet exercice est de démontrer le théorème des valeurs intermédiaires (théorème 5 du chapitre 1) : « Soit f une fonction définie sur un intervalle I de $\mathbb{R}$, et a, b deux réels appartenant à I, $a < b$.
Si f est continue sur $[a\,;\,b]$, alors pour tout réel k compris entre $f(a)$ et $f(b)$, il existe au moins un réel c appartenant à $[a\,;\,b]$ tel que $f(c) = k$. »

Soit une fonction f définie et continue sur un intervalle I de $\mathbb{R}$, a et b deux réels appartenant à I tels que $a < b$, et k un réel compris entre $f(a)$ et $f(b)$.
On construit, par récurrence, deux suites (a_n) et (b_n) en posant : $a_0 = a$ et $b_0 = b$, et pour tout entier naturel n,

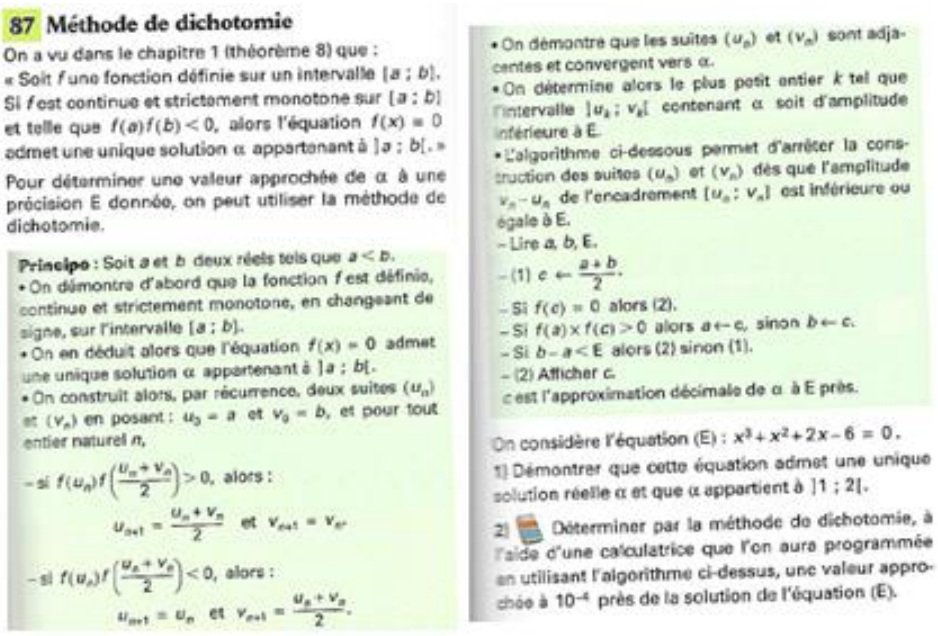
– si $f\left(\frac{a_n + b_n}{2}\right) \leqslant k$, alors $a_{n+1} = \frac{a_n + b_n}{2}$ et $b_{n+1} = b_n$;

– si $f\left(\frac{a_n + b_n}{2}\right) > k$, alors $a_{n+1} = a_n$ et $b_{n+1} = \frac{a_n + b_n}{2}$.

1) a) Démontrer que : $\forall n \in \mathbb{N},\ a_n < b_n$.
b) Démontrer que : $\forall n \in \mathbb{N},\ f(a_n) \leqslant k \leqslant f(b_n)$.
c) Démontrer que (a_n) est une suite croissante et que (b_n) est une suite décroissante.
d) Démontrer que : $\forall n \in \mathbb{N},\ b_n - a_n = \frac{1}{2^n}(b_0 - a_0)$.
e) En déduire que les suites (a_n) et (b_n) sont adjacentes. On appelle L leur limite commune.
2) a) Déterminer $\lim\limits_{n \to +\infty} f(a_n)$ et $\lim\limits_{n \to +\infty} f(b_n)$.
b) Achever la démonstration.

Figure 70 (Extrait d'un exercice du manuel Radial de Terminale Scientifique : Une démonstration du théorème des valeurs intermédiaires)

Le deuxième exercice (fig. 71), intitulé « *Méthode de dichotomie* », suit juste après celui que nous venons de décrire.



87 Méthode de dichotomie

On a vu dans le chapitre 1 (théorème 8) que :
« Soit f une fonction définie sur un intervalle $[a\,;\,b]$. Si f est continue et strictement monotone sur $[a\,;\,b]$ et telle que $f(a)f(b) < 0$, alors l'équation $f(x) = 0$ admet une unique solution α appartenant à $]a\,;\,b[$. »

Pour déterminer une valeur approchée de α à une précision E donnée, on peut utiliser la méthode de dichotomie.

Principe : Soit a et b deux réels tels que $a < b$.
• On démontre d'abord que la fonction f est définie, continue et strictement monotone, en changeant de signe, sur l'intervalle $[a\,;\,b]$.
• On en déduit alors que l'équation $f(x) = 0$ admet une unique solution α appartenant à $]a\,;\,b[$.
• On construit alors, par récurrence, deux suites (u_n) et (v_n) en posant : $u_0 = a$ et $v_0 = b$, et pour tout entier naturel n,

– si $f(u_n)f\left(\frac{u_n + v_n}{2}\right) > 0$, alors :

$$u_{n+1} = \frac{u_n + v_n}{2} \quad \text{et} \quad v_{n+1} = v_n.$$

– si $f(u_n)f\left(\frac{u_n + v_n}{2}\right) < 0$, alors :

$$u_{n+1} = u_n \quad \text{et} \quad v_{n+1} = \frac{u_n + v_n}{2}.$$

• On démontre que les suites (u_n) et (v_n) sont adjacentes et convergent vers α.
• On détermine alors le plus petit entier k tel que l'intervalle $]u_k\,;\,v_k[$ contenant α soit d'amplitude inférieure à E.
• L'algorithme ci-dessous permet d'arrêter la construction des suites (u_n) et (v_n) dès que l'amplitude $v_n - u_n$ de l'encadrement $[u_n\,;\,v_n]$ est inférieure ou égale à E.
– Lire a, b, E.
– (1) $c \leftarrow \frac{a+b}{2}$.
– Si $f(c) = 0$ alors (2).
– Si $f(a) \times f(c) > 0$ alors $a \leftarrow c$, sinon $b \leftarrow c$.
– Si $b - a <$ E alors (2) sinon (1).
– (2) Afficher c.
c est l'approximation décimale de α à E près.

On considère l'équation (E) : $x^3 + x^2 + 2x - 6 = 0$.
1) Démontrer que cette équation admet une unique solution réelle α et que α appartient à $]1\,;\,2[$.
2) Déterminer par la méthode de dichotomie, à l'aide d'une calculatrice que l'on aura programmée en utilisant l'algorithme ci-dessus, une valeur approchée à 10^{-4} près de la solution de l'équation (E).

Figure 71 (Extrait d'un exercice du manuel Radial de Terminale Scientifique : Méthode de dichotomie)

Dans cet exercice, les auteurs présentent la *méthode de dichotomie* comme un moyen de déterminer l'unique solution qui résulte du corollaire du TVI. Cette présentation utilise les mêmes suites adjacentes que celles utilisées pour la démonstration du TVI vue dans l'exercice précédent, sans que le lien soit fait explicitement par les auteurs entre les deux exercices. Les auteurs rappellent l'intérêt qui doit être portée sur la convergence des deux suites, puis sur la distance entre les termes de même rang des deux suites et indiquent que ces derniers vont encadrer la solution.

L'algorithme présenté comme un moyen « d'arrêter la construction » pour une amplitude fixée est écrit en langage naturel. Les auteurs semblent n'attendre des élèves qu'ils se « contentent » à une simple transcription de l'algorithme en langage pseudo-code afin de l'implémenter dans un environnement numérique pour le tester sans que toutefois soit proposé une réflexion sur la validation et la terminaison de l'algorithmique. Le travail se situe ainsi au niveau 1 de l'algorithmique. Cependant, si l'élève fait le lien entre les deux exercices (fig. 70 et 71), nous pouvons observer que les genèses instrumentale/sémiotique dans l'*ETA* vont venir supporter la genèse discursive dans l'$ETM_{analyse}$, et engager une genèse discursive dans l'*ETA*.

3.3.4 Autres ressources

Contrairement aux niveaux précédents, les documents ressources proposés par l'institution, en l'occurrence *Eduscol*, au niveau la Terminale Scientifique ne semblent pas revenir sur des problématiques d'approximation de solutions d'équation. En effet, à ce niveau, les exercices, ainsi que les tâches proposées par les auteurs de ces documents autour du concept d'approximation sont en lien avec le calcul intégral, nouvelle compétence de l'élève à ce niveau, et des approximations de calcul d'aire, en particulier lors de travaux sur les lois continues dans le domaine des probabilités.

Quant aux IREM, ils semblent aussi favoriser une telle « politique » au niveau de la Terminale Scientifique au vue des ressources trouvées sur les sites concernés.

4. Synthèse sur l'analyse des programmes et l'étude de manuels et documents ressources de Seconde et du cycle scientifique terminal

4.1 Au niveau de la Seconde

En Seconde, programme, documents ressources et manuels considèrent un « champ » fonction, plutôt que des contenus en analyse. Les algorithmes concernant les fonctions permettent d'une part la tabulation ou le tracé de courbe, et d'autre part la résolution approchée d'équations. Cette dernière se limite à la dichotomie ; la méthode de balayage ne fait pas l'objet d'un traitement algorithmique. Le travail sur ces algorithmes est privilégié dans un premier temps, avant l'introduction du formalisme algébrique : notre interprétation est que, implicitement, les variables informatiques impliquées dans ce travail sont vues comme des précurseurs des variables de fonction.

Nous avons observé aussi que le programme ne précise pas si les capacités attendues concernent l'implémentation d'algorithmes pour la résolution approchée effective d'une équation donnée ou l'étude d'algorithmes « génériques » opérant sur toutes les équations d'un type donné, ce qui rejoint l'analyse de Modeste (2012). Les manuels ne se situent pas clairement, prenant l'un ou l'autre point de vue, sans que l'objectif soit précisé.

Concernant l'*algorithme de dichotomie* les manuels étudiés, bien que dissemblables, sont conformes au programme. En effet, l'étude de cet algorithme se trouve toujours dans un ou

plusieurs chapitres en lien avec les fonctions, et son objectif est relatif à l'encadrement d'une solution d'une équation. Ainsi, seule l'étude de l'algorithme « *continu* » est proposée par les manuels. Il en est de même pour le document ressource « *fonctions* » provenant d'*Eduscol*. En revanche, le document ressource sur l'algorithmique provenant aussi d'*Eduscol* ainsi que celui de l'IREM proposent une introduction à l'*algorithme de dichotomie* par une approche « *discrète* », autour du jeu : *recherche d'un nombre entier secret dans un intervalle à bornes entières*. Bien que ce ne soit pas dit explicitement, l'étude de l'algorithme « *discret* » prépare celle de l'algorithme continu. Nous discuterons plus loin l'intérêt d'une telle préparation pour notre ingénierie.

Les documents « ressources » et le document IREM diversifient le type de langages algorithmiques et de programmation utilisés contrairement à ce qui est observé dans les manuels étudiés. Le programme ne donne aucune indication à ce sujet. Nous notons aussi que les propriétés des fonctions servant de conditions suffisantes pour l'existence et l'unicité de solutions ne sont mentionnées que dans le document ressource « *fonctions* » et le document IREM. Les propriétés de stricte monotonie et le changement de signe aux bornes sont connues des élèves de Seconde, alors que la continuité intervient seulement en Terminale. Ces propriétés sont évoquées pour l'existence de solutions et non pour justifier l'effectivité de l'algorithme. Le document « *algorithmique* » insiste sur la taille de l'intervalle à chaque itération, ce qui justifie la terminaison. Le programme ne donne aucune indication concernant des conditions pour l'effectivité et la terminaison de l'algorithme.

4.2 Au niveau de la Première Scientifique

Les manuels de Première Scientifique étudiés proposent peu d'exercices sur des algorithmes en lien avec l'approximation d'une solution d'une équation. Nous observons que deux algorithmes peuvent être mis en jeu, contrairement au niveau Seconde où seul l'*algorithme de dichotomie* était concerné. En effet, au niveau de la Première Scientifique, les élèves ayant comme nouveau savoir la *dérivation*, deux algorithmes d'approximation peuvent être proposés : l'*algorithme de Newton* avec utilisation du concept de « dérivée » et l'*algorithme de dichotomie*.

Notre étude montre que, bien que la *méthode de dichotomie* « continue » n'apparaisse pas explicitement, ni même implicitement dans le programme, elle est mise en pratique dans les

manuels étudiés pour l'approximation de solution d'une équation de la forme $f(x) = k$ en lien avec des objectifs liés aux fonctions et à l'algorithmique, ainsi que pour l'obtention d'un encadrement d'une racine carrée. Nous pouvons la trouver dans des chapitres sur les fonctions et dans le chapitre concernant les généralités sur les suites, ce qui permet d'utiliser la notation indexée pour désigner les valeurs successives des variables de boucle. De plus, comme nous avons pu le voir dans le descriptif des manuels choisis, qu'au niveau Première Scientifique certains exercices donnés par les auteurs peuvent être vus aussi comme une approche implicite de conditions nécessaires et suffisantes pour l'existence et l'unicité d'une solution en lien avec le corollaire du TVI qui n'est pas au programme de Première Scientifique, en particulier quand l'exercice se trouve dans un chapitre concernant la dérivation et ses applications. Nous pouvons noter que c'était déjà le cas au niveau de la Seconde dans les documents ressources pour l'enseignant mais pas dans les manuels scolaires étudiés de ce niveau. De plus, au niveau de la Première Scientifique, cette approche implicite de conditions nécessaires et suffisantes pour l'existence et l'unicité d'une solution en lien avec le corollaire du TVI n'est pas systématiquement mise en valeur par les auteurs comme nous avons pu le constater sur les manuels choisis pour notre étude.

Cependant, le concept de continuité avec changement de signes pour l'existence d'une solution à une équation du type $f(x) = 0$ n'est abordé que de façon implicite et essentiellement par une observation graphique de la fonction sur un sous-intervalle en général d'amplitude 1 de l'intervalle de définition, où la solution de l'équation va se trouver. Cette approche bien que moins développée en Seconde, ne semble pas normalement être une nouveauté pour l'élève. L'aspect nouveau vient pour l'essentiel sur le concept d'unicité de la solution sur l'intervalle étudié. En effet, l'élève de Première Scientifique ayant pu mettre en place un travail sur la dérivation, peut ainsi justifier la stricte monotonie de la fonction sur l'intervalle étudié. Ce dernier point, ne fait pas en général l'objet d'un questionnement en Seconde. De plus, il nous semble intéressant de rapporter que parmi les manuels étudiés, seul le manuel *Symbole* présente l'*algorithme de dichotomie* dans le cas d'un encadrement d'une racine carrée irrationnelle sans que soit mentionné l'aspect résolution d'une équation de la forme $x^2 = c$ avec c désignant un nombre entier naturel premier. Nous ne constatons aussi qu'aucun des manuels, donnant des tâches sur les deux algorithmes d'approximation que sont la dichotomie et la méthode de Newton basée sur les équations de tangente, ne propose une

étude quant à la vitesse de convergence de ces méthodes pouvant permettre aussi de les comparer. Quant aux documents IREM, comme nous l'avions signalé précédemment, nous n'avons pu trouver au niveau de la Première Scientifique, un réel texte présentant des tâches nouvelles par rapport à celles vues en Seconde, sur l'*algorithme de dichotomie*. Il semble en être de même pour l'*algorithme de Newton*, malgré l'introduction à ce niveau scolaire du *calcul différentiel*.

Comme nous l'avons décrit dans la partie 1, les logiciels algorithmiques que nous avons choisis pour nos ingénieries ne permettent pas d'exploiter la technique de sous-programmes. Ce choix se justifie par le fait que les documents ressources, ainsi que le programme ne proposent pas l'utilisation d'une telle technique. Le choix des structures proposées par les manuels et les documents ressources étudiés permettent d'observer que les auteurs favorisent une utilisation de boucle de type « *TantQue* ». Ceci fait l'objet d'une analyse détaillée dans la cadre de notre « sous-ingénierie » « *dichotomie discrète* » que nous présentons dans le chapitre suivant.

Les auteurs des manuels semblent aussi favoriser la recherche d'un intervalle encadrant la solution et définissent ainsi une condition d'arrêt sur une inégalité à un nombre arbitraire et non la détermination d'une valeur approchée de la solution. Certains auteurs proposent aussi comme condition dans le cas de l'instruction conditionnelle « *Si… Alors… Sinon* » une inégalité sur la fonction qui renvoie l'élève à une monotonie précise de la fonction étudiée, contrairement à l'étude du signe du produit des images par la fonction choisie des bornes de l'intervalle étudié, ce qui renvoie l'élève à un schéma plus classique du *principe de la dichotomie*.

4.3 Au niveau de la Terminale Scientifique

Dans le champ de l'analyse, nous pouvons observer que les tâches algorithmiques proposées par les manuels étudiés privilégient l'approximation d'une solution d'une équation de la forme $f(x) = k$, où f est une fonction continue et strictement monotone sur un intervalle donné plutôt que d'autres thèmes proposés par le programme tels que l'approximation de nombres réels. Tous les manuels étudiés étudient la dichotomie comme méthode d'approximation. Ainsi, de nombreux exercices proposent de justifier dans un premier temps l'existence et l'unicité d'une solution de l'équation étudiée en travaillant sur la validation des

conditions du TVI et de son corollaire par la fonction étudiée, puis dans un second temps sur l'écriture d'un algorithme de dichotomie permettant de déterminer une valeur approchée ou un encadrement de cette solution. Cette approche de l'algorithmique semble consister à la mise en œuvre de méthodes d'approximation à des résultats montrés au préalable de façon théorique, ce que nous désignerons par *application numérique.* Nous opposons cette approche à celle où l'élève serait amené à s'intéresser à des méthodes d'approximation d'un point de vue aussi bien pratique que théorique : obtention de valeurs approchées, mais aussi discussion sur le domaine d'application, comparaison de méthodes, etc. Cette seconde approche s'apparente à l'*analyse numérique* telle qu'elle est pratiquée dans la recherche et enseignée à l'université et met en jeu des connaissances en analyse (propriétés des fonctions, suites numériques…) aussi bien qu'algorithmiques (itération, variables itératives, effectivité, efficacité**…)**

Bien que les connaissances nécessaires à la preuve de l'effectivité de l'algorithme, et donc du TVI fassent partie du programme de Terminale, celui-ci exclut cette preuve. Cependant, certains des manuels étudiés à ce niveau présentent dans le cadre de travaux pratiques certaines tâches où cette question est présente à travers une *forme atténuée de validation de l'algorithme*. De même, nous avons pu voir que dans le manuel *Radial*, correspondant au programme de 2006, les deux exercices (fig. 8a et 8b) présentés mettent en jeu une preuve du TVI à l'aide de propriétés sur les suites générées par dichotomie. Cependant, l'articulation entre la preuve et la méthode nous a semblé peu maîtrisée : les suites interviennent dans le premier exercice dédié à la preuve sur une fonction générique, puis dans le second dédié à l'approximation du zéro d'une fonction dont l'expression est donnée, sans qu'aucun lien ne soit fait entre ces deux interventions. Par ailleurs, dans le second exercice (fig. 8b), la convergence des suites est montrée à nouveau, alors que seules des considérations d'encadrement sont en jeu pour l'approximation. Nous pouvons penser alors que la conséquence d'une telle approche est que les élèves puissent ne pas distinguer le point de vue « encadrement » qui intervient dans l'approximation, et le point de vue « convergence » qui intervient dans la preuve. Ainsi, l'élève se situe dans une approche de type « application numérique » de l'algorithme de dichotomie, alors que le thème permettrait de passer à une étape au-dessus, c'est-à-dire de type « analyse numérique », au sens que nous avons présenté précédemment.

Le programme ne propose pas d'algorithme particulier pour la détermination d'une valeur

approchée ou d'un encadrement d'une solution d'une équation de la forme $f(x) = k$, contrairement à celui de la Seconde où seul celui de la dichotomie était mentionné. Une problématique de comparaison d'algorithme pourrait donc exister et poser notamment la question de l'efficacité de l'algorithme, ainsi que celui du choix des structures et de l'utilisation de telle ou telle variable itérative. Nous pouvons constater que deux algorithmes d'approximation sont le plus souvent proposés aux élèves : la *dichotomie* et la *méthode de Newton*, mais peu de manuels étudiés propose un travail de comparaison de vitesse de convergence de ces deux algorithmes où l'élève est confronté implicitement à une problématique d'efficacité.

Dans la plupart des cas, la *méthode de dichotomie* garantit la convergence vers un zéro lorsque la fonction est continue sur l'intervalle étudié. Cependant, sa progression dans la recherche de ce zéro (encadrement ou valeur approchée) est plutôt lente. En effet, sa vitesse de convergence est de type linéaire. Quant à la *méthode de Newton*, elle peut ne pas converger si la valeur initiale est trop loin d'un zéro. Cependant, lorsqu'elle converge, elle est beaucoup plus rapide que la *méthode de dichotomie*. En effet, sa vitesse de convergence est de type quadratique.

Nous pouvons aussi poser l'hypothèse qu'au niveau de la Terminale, les élèves ayant des connaissances en analyse plus diversifiées (dérivée, tangente, calcul de limites de suites et de fonctions,…) que celles des niveaux scolaires précédents, le choix d'utiliser tel ou tel algorithme d'approximation d'une solution d'une équation soit laissé à l'initiative de l'élève. De plus, au regard du programme de la Terminale Scientifique, il serait possible d'amener l'élève à s'interroger sur des questions de comparaison de vitesses de convergence de certains algorithmes d'approximation.

4.4 Synthèse globale

Pour conclure, nous notons que l'*algorithme de dichotomie* lui-même n'est mentionné que par le programme et les documents ressources de Seconde où il s'agit plutôt de le mettre en œuvre sur des exemples particuliers. En dépit de son absence du programme, l'*algorithme de dichotomie* apparaît dans les manuels de Première pour des équations pour lesquelles les élèves ne disposent pas d'une méthode de résolution exacte. Il est considéré ainsi comme une méthode générale, ce qui élargit le point de vue. Le programme de Terminale Scientifique

inscrit dans le même paragraphe des notions d'analyse (continuité et TVI) et des « activités algorithmiques » sans cependant préciser comment lier dans les apprentissages ces notions et ces « activités ». Alors que nous pourrions avoir un point de vue « analyse numérique », c'est le point de vue « application numérique » qui domine dans le programme et les manuels étudiés.

Sur les trois niveaux très peu de tâches sont relative à l'effectivité et à l'efficacité. Aucune tâche sur le choix d'utiliser telle ou telle structure algorithmique et sur la détermination des variables itératives nécessaires à l'élaboration d'un algorithme d'approximation n'est proposée tant par les programmes, que les documents « ressources » et les manuels.

5. Nos motivations

Dans une première approche, nous avons justifié notre intérêt pour l'*algorithme de dichotomie* par les liens qui existaient entre celui-ci et les notions d'analyse enseignées au lycée. En effet, celui-ci :

- répond à un problème classique et institutionnel posé en analyse : approximation des valeurs des antécédents par une fonction donnée. Parmi d'autres algorithmes, il nous semble plus directement mettre en jeu les notions d'analyse au lycée. Par exemple, l'*algorithme de balayage* est facile à mettre en œuvre à la main, mais n'est pas aisé à systématiser. Quant à la *méthode de Newton* ou d'autres méthodes du point fixe, elle suppose la connaissance par les élèves de notions pour leur justification correspondant au niveau de fin de Terminale ;
- donne une succession d'encadrements emboités et à bornes rationnelles, jusqu'à une amplitude arbitrairement petite, offrant une représentation accessible aux élèves d'un nombre réel. Le corps de boucle construit les bornes des intervalles et le choix d'une condition d'arrêt correspond à l'amplitude du dernier intervalle.

Mais aussi, nous observons que :

- son champ d'application dépend des propriétés de la fonction : la question de son effectivité renvoie notamment à la notion de continuité et de monotonie sur un intervalle. Ainsi, les élèves peuvent observer des cas où l'algorithme termine, mais n'encadrerait pas un antécédent ou n'en encadrerait qu'un seul alors qu'il en existerait

plusieurs ;

- il permet aussi d'illustrer le Théorème des Valeurs Intermédiaires (TVI) et fournit une démonstration[72] de celui-ci. De façon générale, le TVI repose sur la complétude[73] de **R**. On utilise ici une conséquence : la convergence des suites adjacentes[74]. Les bornes successives des intervalles emboités forment des suites adjacentes.

Cependant, tenant compte de certaines difficultés que pourraient entraîner, auprès d'élèves de lycée, une approche directe de l'*algorithme de dichotomie* dans le cadre « continu », nous choisissons de procéder à une première « sous-ingénierie » qui serait accès sur une approche « discrète » de cet algorithme avant de travailler le cas « continu ». En effet, dans l'algorithme « discret », la structure algorithmique est la même, mais les deux algorithmes n'opèrent pas sur les mêmes données et ne répondent pas au même problème. Dans l'algorithme « discret » la condition d'arrêt et l'alternative dans le corps de boucle sont plus simples à construire pour des élèves débutants en informatique, ce qui permet de graduer la difficulté dans la construction. Enfin, nous faisons l'hypothèse qu'un travail sur l'algorithme « discret » peut préparer les élèves à exprimer l'algorithme « continu ». Cependant, nous anticipons qu'il ne s'agit pas d'un simple transfert de structure. En effet, la condition d'arrêt porte sur une valeur numérique dans le cas « discret », tandis que dans le cas « continu » elle porte sur une amplitude. Elle est construite comme une égalité à un nombre fixé dans le cas « discret », tandis que dans le cas « continu », il s'agit d'une inégalité à un nombre arbitraire.

6. Conclusion du chapitre 3 : la structure de l'ingénierie

Comme nous l'avons mentionné dans la section 3.1.3, à la différence de la majorité des manuels, deux ressources analysées considèrent l'algorithme « discret » en préalable à l'introduction de l'algorithme « continu ». L'*algorithme de dichotomie* « continu », le seul mentionné par le programme conduit à un encadrement par des rationnels aussi proches que l'on veut.

Nous rappelons que dans les deux approches de l'*algorithme de dichotomie*, tant

[72] Bolzano (1781-1848), dans *Une preuve analytique*... (1817) et Cauchy (1789-1857), dans son *Cours d'analyse à l'École polytechnique* (1821), utilisent la dichotomie pour démontrer le TVI.

[73] Concept non connu au niveau du lycée.

[74] Les suites adjacentes ne font plus officiellement partie des programmes des lycées depuis 2012.

« discret » que « continu », la structure algorithmique est la même, mais ils n'opèrent pas sur les mêmes données et ne répondent pas au même problème. Nous souhaitons alors structurer notre ingénierie en deux « sous-ingénieries » :

(a) Une première qui va porter sur le cas « discret » mettant en jeu la structure et les variables, ainsi qu'un test d'arrêt comme égalité à un nombre fixé (cf. Chapitre 4).

(b) Une seconde sur le cas « continu » mettant en jeu la compréhension de la méthode comme moyen d'obtenir une approximation d'un antécédent par une fonction donnée et de prendre conscience de la propriété des valeurs intermédiaires, et la construction d'un test d'arrêt différent de celui élaboré dans le cas « discret »**.** Ce travail dans le cas « continu » va se compléter par un questionnant sur le domaine de validité de l'algorithme « continu » et l'identification des conditions suffisantes pour cette validité et donc pour la propriété des valeurs intermédiaires (cf. Chapitre 5).

Ces deux « sous-ingénieries » vont se décliner différemment dans les trois niveaux de classe. En seconde et en première, notre ingénierie va comporter les « sous-ingénieries » **(a)** et **(b)** sur des fonctions prédéfinies. Tandis, qu'en terminale, notre ingénierie va se centrer sur la sous-ingénierie **(b)**. Cependant, les élèves vont pouvoir faire la sous-ingénierie **(a),** en particulier s'ils sont novices vis-à-vis de l'algorithme.

Pour conclure, nous rappelons que nous choisissons de découper notre présentation en deux chapitres distincts. Ainsi, nous pensons que le lecteur peut mieux distinguer les approches et le travail liés à la situation « discrète » de celle de la situation « continue ».

Chapitre 4 : Une ingénierie articulant ETA et ETM spécifique (partie 1) : *Algorithme de dichotomie* « discrète » – Une stratégie « rapide » et « gagnante »

1. Nos objectifs et nos choix

1.1 Les objectifs

1.1.1 Le problème

Dans le cadre d'un jeu pratiqué par deux joueurs sur la recherche d'un nombre entier secret compris entre deux nombres entiers donnés, les élèves doivent élaborer une stratégie « gagnante » et « rapide » permettant de trouver ce nombre entier secret en un minimum de coups. Afin d'étudier la pertinence de leur stratégie, il est demandé aux élèves de la formaliser sous la forme d'un algorithme exécutable de façon automatique.

1.1.2 Les attentes

Lors de la mise en pratique de cette tâche, nous nous attendons à ce que les élèves s'approprient, par des essais successifs, le problème d'identification d'un nombre entier inconnu dans un intervalle donné auquel participent deux joueurs. Nous espérons que chaque essai va donner lieu à un diagnostic de comparaison de l'essai et du nombre entier secret, pour découvrir que la *méthode de dichotomie* peut s'avérer être une solution à ce problème.

Nous nous attendons aussi à ce qu'en mathématique les élèves mobilisent des connaissances anciennes comme le calcul d'une moyenne de deux nombres entiers, et qu'ils acquièrent aussi des connaissances nouvelles sur les concepts de partie entière d'un nombre réel et de nombre entier aléatoire sur un intervalle, ainsi que sur la *méthode de dichotomie*. L'utilisation de l'algorithmique doit permettre aux élèves : – d'interpréter la partie entière comme une fonction numérique ; – de définir un nombre (pseudo-)aléatoire ; – d'utiliser des variables itératives et de choisir une structure algorithmique pour formaliser en langage algorithmique la méthode d'approximation qu'est la dichotomie.

1.2 Les choix

1.2.1 La question posée

Nous décidons de ne pas poser la question de l'optimalité qui ne peut être traitée à ces niveaux scolaires (Seconde à Terminale Scientifique). Nous faisons l'hypothèse que la dichotomie va s'imposer aux élèves comme une stratégie « rapide » et « gagnante ». Nous traduisons cela dans la tâche donnée aux élèves par le fait que le nombre entier « secret », doit être découvert « *en faisant le minimum de propositions* » et en faisant l'hypothèse que les élèves vont interpréter cette condition comme étant un objectif de performance comparée qui ne sera pas exprimé en termes d'optimalité.

1.2.2 Le mode de travail

Quel que soit le niveau scolaire, les élèves sont en salle informatique et travaillent en binôme, de façon qu'il y ait un ordinateur par binôme équipé de deux logiciels permettant d'exploiter des organigrammes (avec *LARP*) ou un langage informatique (avec *LARP* et *AlgoBox*). Les élèves de chaque binôme sont libres de choisir tel ou tel artefact : « papier-crayon » pour les essais et l'approche ludique du problème, algorithmes écrits en langage naturel, organigrammes permettant une représentation *graphique* d'un algorithme, langage *pseudo-code* et langage machine.

De plus, quand la nécessité de passer à l'algorithmique pour modéliser le jeu « *recherche d'un nombre entier secret* » va apparaitre dans l'esprit de l'élève, afin d'étudier l'optimalité de la stratégie « rapide » conjecturée, la transposition du travail, fait à l'aide du « papier-crayon » à un algorithme associé à cette stratégie « rapide » va aussi nous permettre d'observer si la programmation d'un algorithme peut aider les élèves à changer leur conception des variables, ou les aider à améliorer leurs raisonnements mathématiques ou participer à la formation de leur esprit critique.

1.2.3 Les algorithmes en jeu

Quel que soit le niveau scolaire, la règle du jeu proposée aux élèves est la suivante :

Le jeu proposé ci-dessous se joue à deux joueurs A et B.
- Le joueur A choisit « secrètement » et au hasard un nombre entier « secret » compris entre 1 et 1000.
- Le joueur B doit deviner ce nombre entier « secret » en faisant le minimum de propositions.

A chaque proposition du joueur B, le joueur A répond par « le nombre cherché est plus grand », « le nombre cherché est plus petit » ou « bravo, tu as gagné » selon la position de la proposition par rapport au nombre entier « secret » à atteindre.

Lors de cette ingénierie les élèves ont pour objectif final la production d'un algorithme dit du joueur B, c'est-à-dire conçu pour être exécuté par une machine émettant de façon automatique des propositions en fonction des réponses d'un joueur A, sur le *principe de la dichotomie*.

Nous considérons aussi dans cette section l'algorithme dit du joueur A, c'est-à-dire conçu pour être exécuté par une machine mémorisant un nombre « secret » et émettant de façon automatique des réponses aux propositions d'un joueur B.

En effet, notre hypothèse est qu'un travail sur les deux algorithmes peut permettre de clarifier les positions respectives de la machine (dispositif automatique) et de l'utilisateur, et que leurs similarités et différences peuvent être mises à profit dans la construction de l'ingénierie.

1.2.3.1. Structures et expressions possibles de l'algorithme du joueur A

Nous présentons ici les différentes expressions possibles de l'*algorithme* du joueur A.

a) La structure « Boucle… FinBoucle » avec sortie à l'intérieur de la boucle (QuitterBoucle)

- **Un peu de théorie**

Nous appelons *« Boucle…FinBoucle »* une structure de contrôle capable d'exécuter un certain nombre de fois une même série d'instructions formant le corps de la boucle, en fonction d'une ou plusieurs conditions à vérifier. L'utilisateur doit toujours vérifier que les conditions deviennent fausses à un moment donné afin de sortir de la boucle. En effet, si ceci n'est pas vérifié la boucle devient infinie et fait « bugger » le programme.

Ainsi, selon Arsac (1983), la boucle doit comporter une ou plusieurs instructions « SORTIE » que nous notons « *f*(SORTIE) », à savoir une suite d'instructions ayant au moins une occurrence de l'instruction « SORTIE ». « SORTIE » a la signification : sortir de la boucle et continuer en séquence. Une telle boucle est ainsi équivalente à :

```
L : f(ALLER en L') ; ALLER EN L
L' : …
```

Comme pour la boucle « TantQue…Faire » et selon Arsac, la *post-assertion* de *f* est *pré-assertion* de « ALLER EN L », et doit alors impliquer la *pré-assertion* de L. Par conséquent, cette assertion est invariante sur la boucle.

Ainsi, nous avons alors :

```
FAIRE
        f(SORTIE)
BOUCLER
```

Soit encore, selon Arsac :

```
⟦'P⟧    FAIRE
        ⟦P⟧ f(⟦Q⟧ SORTIE)
        BOUCLER ;
⟦Q⟧
```

où $'P$ est la *pré-assertion* de la boucle et Q celle de l'instruction « SORTIE » (ou encore l'union de celles des instructions « SORTIE » s'il y en a plusieurs).

Une telle boucle étant beaucoup plus générale qu'une boucle « TantQue…Faire », nous ne possédons pas de relation simple et générale entre *P* et *Q*. Ceci dépend de la structure de la suite d'instructions *f*. Cependant, la pratique montre qu'elle se prête bien à une construction raisonnée de programmes.

- **Retour au joueur A (celui qui pense le nombre entier « secret »)**

Cette structure est isomorphe au traitement « manuel » tel qu'il est proposé aux élèves. Ainsi, on peut avoir les représentations suivantes[75] comme expressions « *textuelle* » (langage pseudo-code) et « *spatiale* » (organigramme) de l'algorithme du joueur A (fig. 72) :

```
Début algorithme
Initialiser A
Boucle
    Lire VP (#valeur proposée)
    Si VP < A
        Afficher « trop petit »
    Sinon
        Si VP > A
            Afficher « trop grand »
        Sinon
            Afficher « bonne
valeur »
            QuitterBoucle
        FinSi
    FinSi
Fin Boucle
Fin algorithme
```

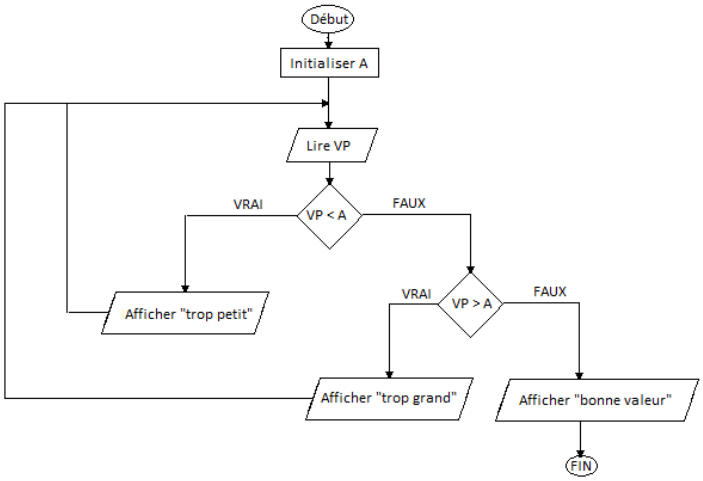


Figure 72 (Un organigramme (format « spatial » de l'algorithme du joueur A, celui qui pense le nombre entier « secret »))

Cette structure a fait l'objet d'un débat parmi les informaticiens. Une majorité a considéré qu'elle entrainait des difficultés, spécialement pour les débutants, à structurer efficacement leur programme. Pour cette raison, elle n'est généralement plus proposée dans les

[75] Une variante possible consiste à faire successivement le test à VP=A avec sortie si la valeur est « vraie », puis VP<A avec une l'alternative.

environnements de programmation pour les débutants. Néanmoins nous ne l'écartons pas dans notre ingénierie, car elle exprime directement le traitement « manuel ».

b) La structure « Répéter… Jusqu'à »

- **Un peu de théorie**

La boucle « Répéter…Jusqu'à » répète un traitement jusqu'à ce qu'une condition (expression booléenne) soit vraie. Son formalisme est représenté par le principe suivant :

```
Répéter
        < Traitement > // une instruction simple ou un bloc d'instructions
Jusqu'à une condition d'arrêt
```

Ainsi, le traitement est exécuté puis la condition est vérifiée. Si elle n'est pas vraie, on retourne au début de la boucle et le traitement est répété. Si la condition est vraie, on sort de la boucle et le programme continue séquentiellement. A chaque fois que le traitement est exécuté, la condition d'arrêt est de nouveau vérifiée à la fin. Ce traitement est exécuté au moins une fois.

- **Conséquence pour le joueur A**

Avec cette structure, l'algorithme du joueur A peut ainsi s'exprimer sous la forme « textuelle » suivante (fig. 73), où « A » représente le nombre entier« secret » et « VP » la valeur proposée par le joueur B. De même, l'expression « spatiale » associée comporte elle aussi une condition de sortie après le corps boucle (fig. 74).

```
Initialiser A
Répéter
   Choisir VP
   Si A<VP
         Ecrire « le nombre cherché est
plus petit »
   Sinon
         Si A>VP
               Ecrire « le nombre cherché
est plus grand »
         Fin Si
   Fin Si
Jusqu'à A=VP
Ecrire « Bon nombre »
```

Figure 73 (Un algorithme au format « textuel » du joueur A)

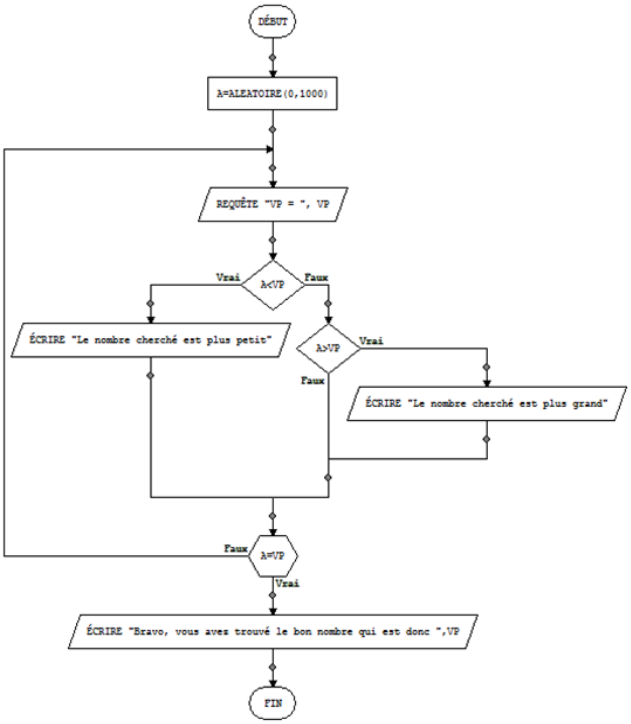


Figure 74 (Un algorithme au format « spatial » du joueur A)

- **Comparaison avec la structure « Répéter » avec sortie à l'intérieur de la boucle**

Cette structure « Répéter … Jusqu'à » oblige à un test supplémentaire par rapport à la structure « *Boucle...FinBoucle* ». On peut considérer qu'elle découle moins directement du traitement « manuel ».

Cependant, ce test final peut aider l'élève à mieux comprendre ce qu'il sera amené à construire quand il passera à l'algorithme « continu » de dichotomie, où en plus des tests d'inégalité dans la boucle, il va exister une condition d'arrêt relative à la précision souhaitée

c) La structure « TantQue… Faire »

- **Un peu de théorie**

La boucle « TantQue… Faire » répète un traitement tant qu'une expression conditionnelle est vraie. Si d'emblée, la condition n'est pas vraie, le traitement n'est pas exécuté. On voit donc que la boucle « TantQue … Faire » a un point commun avec la structure conditionnelle, ou si la condition n'est pas vraie, le traitement n'est pas exécuté.

La syntaxe d'une telle boucle est la suivante :

```
TantQue < Condition d'exécution > Faire
         < Traitement > // instruction simple ou bloc
d'instructions
FinTantQue
```

- **Cas du joueur A**

Ainsi, nous avons dans notre cas, nous avons comme forme « textuelle » de l'algorithme l'expression suivante (fig. 75) :

```
A ← entier aléatoire dans la plage 0, 1 000
Choisir VP
TantQue A différent de VP
    Si A<VP
        Ecrire « le nombre cherché est plus petit »
    Sinon
        Ecrire « le nombre cherché est plus grand »
    FinSi
    Choisir VP
FinTantQue
```

Figure 75 (Un algorithme au format « textuel » du joueur A)

A la différence de la structure « Répéter … Jusqu'à », cette structure « TantQue … Faire » oblige à répéter l'instruction choisir VP, avant l'itération et dans le corps de boucle, mais elle n'utilise qu'une alternative dans le corps de boucle.

L'expression « spatiale » de l'algorithme (fig. 76) comporte elle aussi une condition de continuation en entrée de boucle :

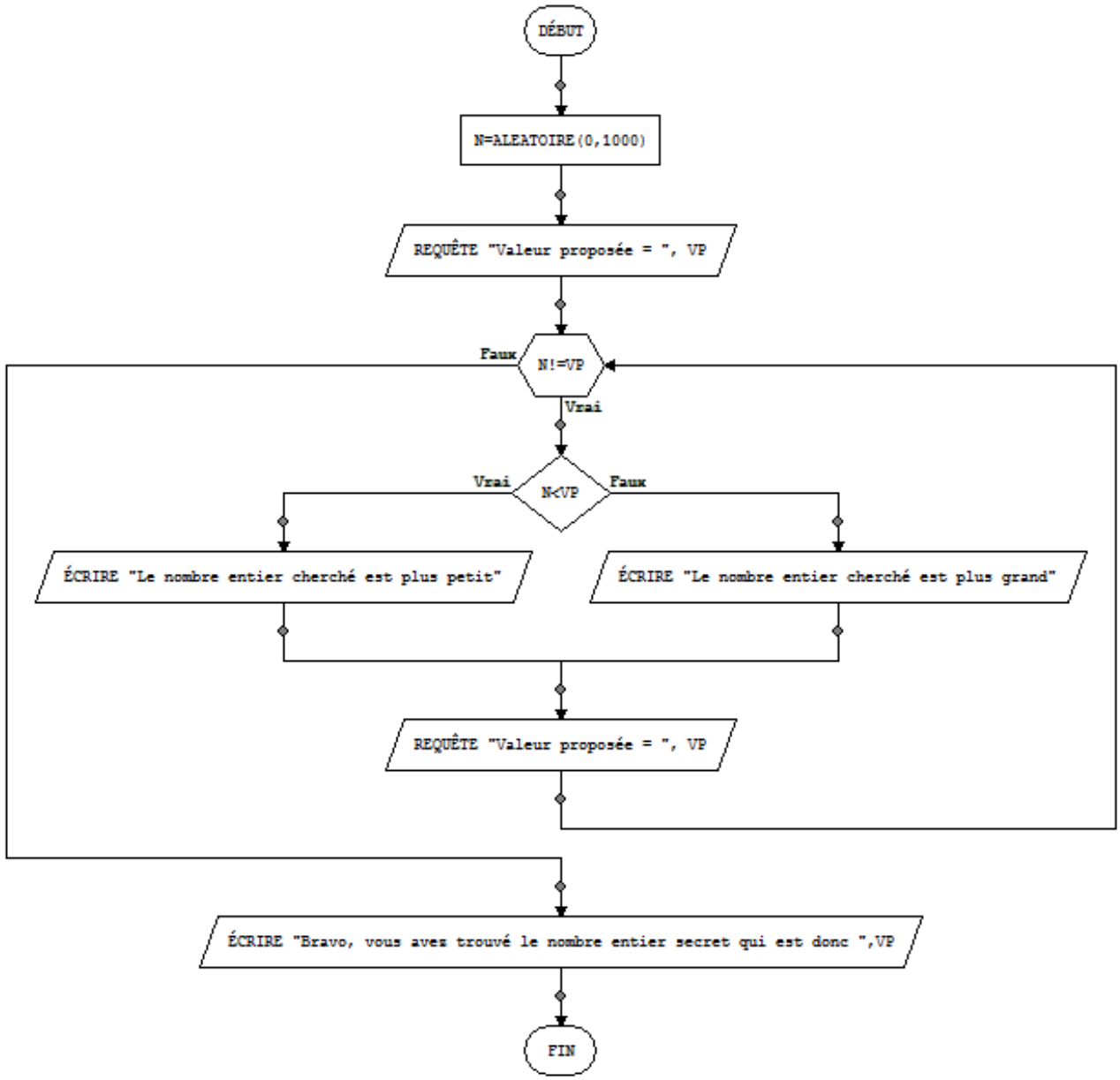


Figure 76 (Un algorithme au format « spatial » du joueur A)

- **Comparaison avec la structure « Répéter…Jusqu'à »**

Nous avons ainsi le tableau (Tableau n° 27) suivant qui permet de synthétiser les différences entre les boucles « Répéter… Jusqu'à » et « TantQue … Faire » :

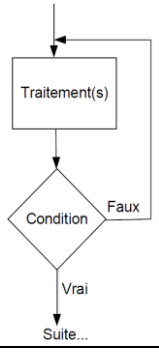
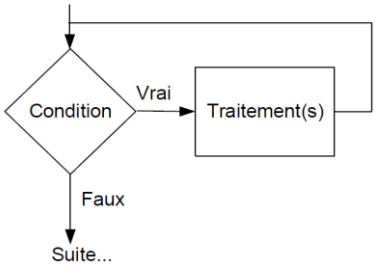
« Répéter… Jusqu'à »	« TantQue…Faire »
Condition vérifiée après le traitement : le traitement est forcément exécuté une fois.	Condition vérifiée avant le traitement : le traitement peut ne pas être exécuté.
Condition d'arrêt : le traitement est répété si la condition est fausse.	Condition de continuation : le traitement est répété si la condition est vraie.
	

Tableau 27

Dans la culture des lycées français, il n'est souvent proposé qu'une seule structure pour les boucles à nombre d'itérations non connu. Comme la structure « TantQue … Faire » est plus générale puisqu'elle n'implique pas au minimum un passage dans la boucle, c'est celle qui est retenue par la quasi-totalité des enseignants et des auteurs de manuels (cf. les analyses des différents manuels faites au Chapitre 3). Ici aussi, nous prenons en compte ce choix, en ce qui concerne l'expression textuelle, sans le discuter puisque ce n'est pas l'objet de cette thèse. Néanmoins, nous pensons intéressant que lors de l'expression sous forme d'une disposition spatiale (organigramme), les élèves puissent avoir le choix des deux structures. Nous pensons que cela leur donne plus de liberté pour passer du traitement manuel à une expression algorithmique, tout en nous conformant aux usages en ce qui concerne l'expression textuelle.

De plus, le choix de proposer aux élèves une disposition spatiale peut permettre de diversifier les possibilités d'expression.

1.2.3.2. Structures et expressions possibles de l'algorithme du joueur B (celui qui propose des réponses)

Nous souhaitons présenter les expressions de l'algorithme du joueur B pour les structures « Boucle… FinBoucle » et « Répéter … Jusqu'à », ainsi que pour l'« expression textuelle » « TantQue … Faire ».

a) La structure « Boucle… FinBoucle »

Comme pour l'algorithme du joueur A, cette structure reste isomorphe au traitement

« manuel » tel qu'il est proposé aux élèves. Ainsi, on peut avoir comme expression « textuelle » de l'algorithme du joueur B, la formulation suivante (fig. 77).

```
Début algorithme
VP1 ← 0 ; VP2 ← 1000
Boucle
      DébutBoucle
          VP ← ENT((VP1+VP2)/2)
          Afficher VP
          Ecrire #trop petit ou trop grand ou
          égal#
          Lire Réponse
          Si Réponse = « égal »
              QuitterBoucle
          Sinon
              Si Réponse = « trop petit »
                    VP1=VP
                    ALLER EN DébutBoucle
              SINON
                    VP2=VP
                    ALLER EN DébutBoucle
              FinSi
          FinSi
      FinBoucle
Fin algorithme
```

Figure 77 (Un algorithme au format « textuel » du joueur B, celui qui propose des réponses)

b) La structure « Répéter… Jusqu'à »

La structure globale est la même que pour l'algorithme du joueur A, la boucle « Répéter…Jusqu'à » répète le traitement jusqu'à ce que la condition (expression booléenne) choisie soit vraie. Cette condition correspond à la même situation que celle l'algorithme du joueur A. Le corps de boucle contient une seule alternative au lieu de deux imbriquées comme dans l'algorithme du joueur A, la seconde branche de cette alternative pouvant être exécutée dans le cas où la réponse est « égal », alors qu'elle n'a de sens que pour une réponse « trop grand ».

L'algorithme que nous considérons ici, diffère aussi de l'algorithme du joueur A car il implique des variables entières itératives qui évoluent selon une stratégie prédéfinie : les bornes de l'intervalle courant (VP1 et VP2 dans les figures) dans lequel se trouve le nombre secret et une variable VP calculée à partir de VP1 et VP2 en fonction de la stratégie choisie (la partie entière de la moyenne dans les figures). Il est important que VP1 et VP2 soient initialisées avant la boucle. VP peut prendre sa valeur initiale dans le corps de boucle.

Avec cette structure et ces variables, l'algorithme du joueur B peut s'exprimer de la façon suivante qu'elle soit « textuelle » ou « spatiale » (fig. 78) :

```
VP1←0 ; VP2←1000
Répéter
    VP←ENT ((VP1+VP2)/2)
    Afficher VP
    Ecrire #trop petit ou trop grand ou égal#
    Lire Réponse
    Si Réponse= « trop petit »
          VP1←VP
    Sinon
          VP2←VP
    Fin Si
Jusqu'à Réponse= « égal »
Ecrire « J'ai trouvé le bon nombre », VP
```

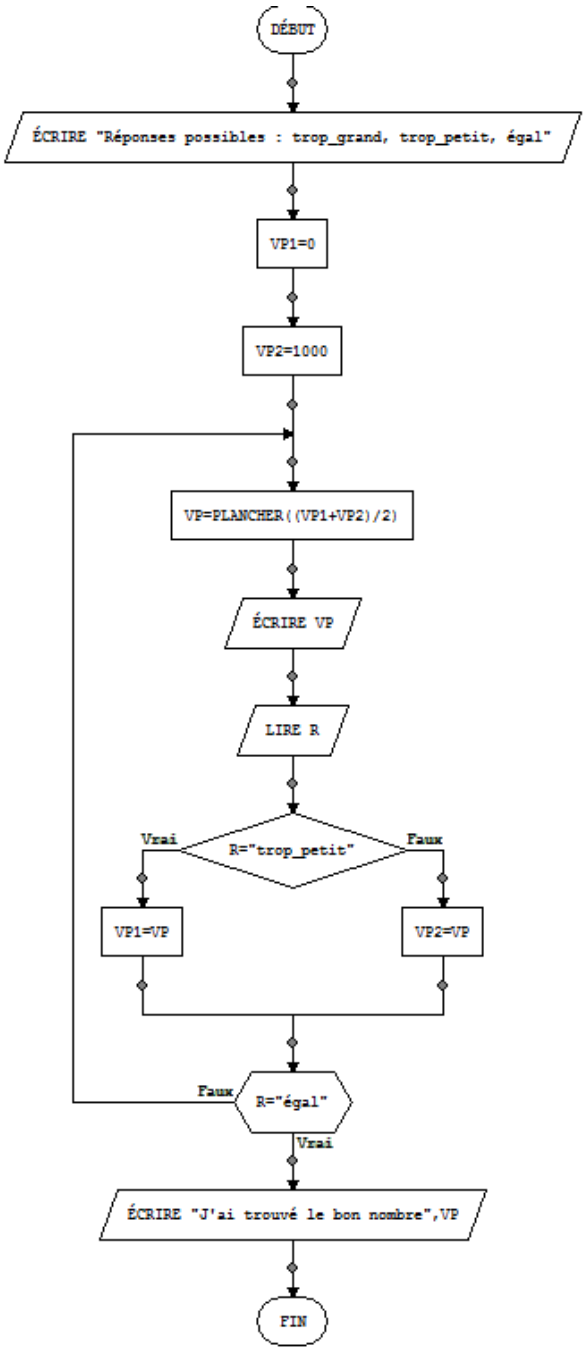


Figure 78 (Deux algorithmes possibles du joueur B aux formats « textuel » et « spatial »)

c) La structure « TantQue… Faire »

La structure globale est ici aussi la même que pour l'algorithme du joueur A, la boucle « TantQue…Faire » répète le traitement tant que la condition (expression booléenne) choisie est fausse. Comme pour le « Répéter… Jusqu'à », le corps de boucle contient une seule alternative au lieu de deux imbriquées comme dans l'algorithme du joueur A.

Ici, aussi l'algorithme implique des variables entières itératives VP1, VP2 et VP qui évoluent selon une stratégie prédéfinie. Comme il est classique dans la structure « TantQue... Faire », une partie du traitement doit être réalisée avant le test d'entrée dans la boucle. Il est donc présent à la fois dans la partie initialisation et à la fin du corps de boucle. Ainsi, l'algorithme du joueur B peut avoir comme expression « textuelle » ou « spatiale » (fig. 79) :

```
VP1←0 ; VP2←1000
VP← (VP1+VP2)/2
Afficher VP
Ecrire #trop petit ou trop grand ou égal#
Lire Réponse
TantQue Réponse différent de « égal »
        Si Réponse est « trop petit »
                VP1←VP
        Sinon
                VP2←VP
        FinSi
        VP←ENT ((VP1+VP2)/2)
        Afficher VP
        Ecrire #trop petit ou trop grand ou
égal#
        Lire Réponse
FinTantQue
```

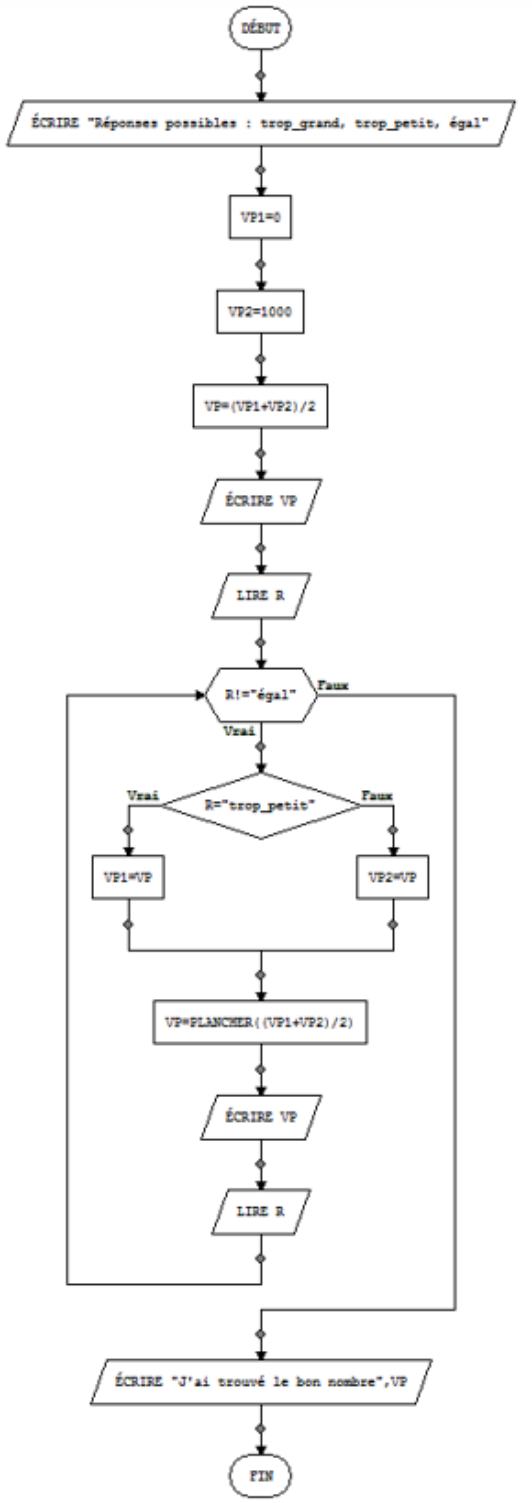


Figure 79 (Deux algorithmes possibles du joueur B aux formats « textuel » et « spatial »)

1.2.4 Choix pour l'ingénierie

En fonction des analyses faites sur les algorithmes précédents, nous prévoyons une étape de préparation centrée sur la structure ; pour cette étape nous demandons aux élèves non pas de réaliser l'algorithme gagnant du joueur B, mais l'algorithme réalisant les actions du joueur A : *choix du nombre « secret » et réponses aux propositions du joueur B*. L'intérêt de ce choix vient du fait que l'algorithme du jouer A possède la même structure que celle du joueur B, mais n'implique pas la gestion de variables de boucle. Ainsi, nous séparons la difficulté pour les élèves.

2. Les phases

L'ingénierie est prévue pour une durée de deux heures. La première phase concerne l'algorithme du joueur A (celui qui choisit un nombre entier « secret » et qui répond aux réponses proposées par le joueur B), et les deux autres phases l'algorithme du joueur B. Nous concevons la première phase comme une prise de conscience de la structure algorithmique commune aux deux algorithmes, et par conséquent cette phase ne va pas jusqu'à la formulation dans un environnement de programmation. Les phases suivantes concernent l'algorithme du joueur B. nous choisissons de ne pas laisser les élèves d'emblée se confronter à l'ordinateur. En effet, compte-tenu de la complexité de l'algorithme, il ne serait pas aisé pour eux de tirer parti des résultats d'exécution dans le temps limité de la « sous-ingénierie ». Nous prévoyons donc une phase 2 de conception générale au « papier-crayon ». Puis, dans la phase 3, les élèves vont rencontrer les contraintes de l'expression dans un environnement de programmation. Nous choisissons de retarder l'exécution sur ordinateur jusqu'à la fin de cette phase.

2.1 Phase 1 (algorithme du joueur A – conception générale)

Dans cette phase, l'objectif consiste à ce que les élèves se familiarisent dans un premier temps avec le jeu en jouant à deux puis conçoivent la structure d'un algorithme écrit au « papier-crayon » sous une forme « textuelle » ou « spatiale » représentant le joueur A, c'est-à-dire celui qui choisit « secrètement » et au hasard un nombre entier « secret » compris entre 1 et 1000.

Les élèves sont regroupés en binôme. Pendant cette phase, les ordinateurs sont éteints.

Chaque binôme fait fonctionner le jeu proposé oralement. Pour chaque binôme, les élèves sont amenés à choisir qui joue le rôle du joueur A et du joueur B, afin de s'approprier le jeu. Puis, ils doivent déterminer à quelle structure algorithmique va correspondre l'action du joueur A afin de pouvoir écrire un algorithme « papier-crayon » correspond à la tâche du joueur A. Ils n'ont à leur disposition, pendant cette phase, qu'une simple feuille de papier et un crayon.

Les élèves terminent cette phase par l'écriture au « papier-crayon » d'un algorithme du joueur A, en le présentant soit sous une forme « textuelle » soit sous forme « spatiale ». Cette phase est prévue durer 30 minutes.

2.2 Phase 2 (algorithme de joueur B – conception générale et écriture d'un organigramme papier-crayon)

Au début de cette phase, les élèves doivent imaginer les stratégies possibles pour déterminer ce nombre entier « secret ». En fonction de l'avancée du travail de l'ensemble des binômes lors de la première phase, l'enseignant peut décider, si cela s'avère nécessaire, de rappeler la consigne « *Le joueur B doit deviner ce nombre entier « secret » en faisant le minimum de propositions* ». Ainsi, il peut inciter les élèves à réfléchir aux stratégies mises en place dans la classe par le joueur B de chaque binôme pour trouver le nombre secret. Il peut s'ensuivre alors des discussions entre les joueurs de chaque binôme, les joueurs devant arriver à élaborer une stratégie « gagnante » et « rapide » permettant au joueur B de minimiser le nombre de propositions pour trouver le nombre entier « secret » auquel pense le joueur A. Pour cela, une stratégie basée sur la dichotomie peut se construire chez les élèves.

Une fois que cette stratégie est perçue par les élèves, chaque binôme doit proposer un algorithme « papier-crayon » correspondant à l'action du joueur B en pouvant s'inspirer de la structure algorithmique mise en place lors de la phase 1. Le cas échéant, l'enseignant peut intervenir pour inciter les élèves à actualiser à chaque étape du processus, les informations évoquées lors des essais précédents, mais sans aucune stratégie particulière pour stocker les nouvelles informations de façon à conserver, au fur et à mesure du jeu, les réponses données aux tests par le joueur A.

Les élèves ont donc à décrire une stratégie gagnante en un minimum de coups où le joueur B va être joué par la machine. Pour préparer la phase 3 sur la formalisation de l'algorithme du

joueur B dans un environnement de programmation, les élèves doivent mettre en évidence, la réduction de l'intervalle grâce à la moyenne des bornes (dichotomie), comme assurant une stratégie « gagnante » et « rapide » sans toutefois l'exprimer avec les contraintes d'un langage formalisé donné. Les élèves doivent aussi considérer des nombres entiers même si la moyenne n'est pas entière.

Cette phase se termine par l'écriture d'un algorithme, sous la forme d'un organigramme dessiné au « papier-crayon » représentant l'action du joueur B décrit au début de la phase. Cette phase est prévue pour une durée de 60 minutes.

2.3 Phase 3 (algorithme de joueur B – formalisation sous formes « textuelle » et « spatiale » – implémentation dans un environnement numérique de l'algorithme)

Lors de cette phase, les élèves procèdent à une formalisation, sous formes « textuelle » et « spatiale » pouvant être implémentable sur un ordinateur, de l'algorithme obtenu à la phase 2 pour le joueur B, puis ils doivent le programmer dans deux environnements numériques *AlgoBox* et *LARP* (cf. Chapitre 1), afin de le tester. Nous inscrivant dans la culture du lycée, nous utilisons un environnement de programmation « textuelle » n'autorisant que la structure « TantQue ». En ce qui concerne l'organigramme, nous souhaitons donner plus de choix à l'élève, et nous choisissons d'utiliser un environnement implémentant les deux structures « TantQue » et « Répéter … Jusqu'à ». Lors de cette phase, les élèves doivent, si cela s'avère nécessaire, procéder à des modifications ou à des adaptations de l'algorithme « papier-crayon » de la phase 2, tout en gardant le même traitement des variables et le même type de structure algorithmique afin que l'algorithme soit bien implémentable dans un environnement numérique.

Cette phase se termine par des essais sur l'ordinateur de l'algorithme du joueur B afin de vérifier si la stratégie du joueur B est « gagnante » et « rapide ». Cette phase est prévue durer 30 minutes.

3. Analyse *a priori*

3.1 Phase 1 (algorithme du joueur A – conception générale)

Lors de cette phase, les élèves s'approprient le jeu et écrivent un algorithme pour le joueur A en « papier-crayon » soit sous une forme « textuelle » soit sous une forme « spatiale »

(organigramme). Nous supposons que le passage du jeu manuel à la conception d'un algorithme pour une exécution automatique peut poser des difficultés chez un certain nombre d'élèves débutants en informatique. En effet, on s'attend à ce que les élèves éprouvent des difficultés à établir la structure algorithmique rendant compte du déroulement des échanges qui ont lieu entre les deux joueurs, ici pour les actions du joueur A. Nous prévoyons la mise en place par des élèves débutants d'alternatives successives plutôt qu'imbriquées et des expressions de boucle de type « Si *condition* sortir sinon *recommencer* », c'est-à-dire du type « GOTO ». De plus, suivant le niveau de compétences en informatique des élèves, nous pensons que dans le cas d'une représentation de l'algorithme sous forme d'un organigramme, certains élèves favorisent le choix d'une structure algorithmique de type « Boucle…FinBoucle » directement déduite de la règle du jeu. Pour les élèves déjà initiés aux formes d'itération standard, nous attendons plutôt la structure « Répéter… Jusqu'à » qui nous semble moins s'éloigner de la règle du jeu que la structure « Tant Que…Faire ». La mise en place des variables intervenant dans le cas du joueur A peut aussi poser problème à l'élève spécialement en Seconde ; le nombre « secret » est une constante ; le choix du joueur B est lu à chaque pas de l'itération.

Selon Rogalski et Samurçay (1986), les constantes posent moins de difficultés aux débutants que les variables évoluant au cours de l'itération ; ainsi, pour le choix du joueur B, des élèves peuvent être tentés d'introduire une nouvelle variable à chaque pas de l'itération.

3.2 Phase 2 (algorithme de joueur B – conception générale et écriture d'un organigramme « papier-crayon »)

Lors de cette phase, les élèves doivent d'abord déterminer une stratégie « rapide » et « gagnante » pour deviner le nombre (stratégie du joueur B) puis écrire à l'aide d'un organigramme « papier-crayon » un algorithme correspondant à cette stratégie.

Nous prévoyons que les élèves vont comprendre la rapidité comme étant le fait de trouver le nombre « en un minimum de coups ». Nous supposons que la première difficulté pour les élèves sera de penser le choix d'un nombre à proposer par le joueur B à chaque étape du jeu. En effet, dans un premier temps, il est possible que les élèves pensent que le joueur B propose un nombre au hasard qui tient plus ou moins compte des réponses du joueur A. Cependant, nous pensons que le choix d'un nombre à l'intérieur d'intervalles imbriqués à chaque étape

jouée par le joueur B, va être une « méthode » qui va très rapidement s'imposer aux élèves. En effet, tenant compte des réponses du joueur A, cette méthode va leur sembler permettre d'évoluer plus vite vers la découverte de l'entier secret qu'un choix entièrement aléatoire. Il leur faudra déterminer ces intervalles puis un principe de choix d'un nombre dans l'intervalle courant. Pour cela, nous pensons que prendre la moyenne des bornes de l'intervalle va s'imposer aux élèves. Cependant, nous pensons que ce calcul de moyennes va leur poser une nouvelle difficulté. En effet, le résultat obtenu pourra ne pas être un nombre entier contrairement au cas où ils proposeraient un nombre entier de manière aléatoire. Nous pensons alors qu'ils vont dans un premier temps proposer un arrondi à l'unité près du nombre décimal obtenu lors du calcul de la moyenne

Ensuite, les élèves ont à écrire un algorithme du joueur B sous la forme d'un organigramme, dessiné au « papier-crayon », qu'ils ont considéré dans la première partie de cette phase sans tenir compte de contraintes qui seraient le fait d'un langage formalisé. Nous supposons tout d'abord que cela va faciliter le travail des élèves concernant l'expression du nombre à proposer, celle-ci pouvant ne pas être formalisée avec la notation fonctionnelle, par exemple « (a+b)/2 arrondi » plutôt que « Partie Entière ((a+b)/2) ». Nous pensons aussi que, concernant la structure à concevoir, les élèves pourront réinvestir la structure élaborée de façon formalisée en phase 1 pour le joueur A. En revanche, nous prévoyons que certains élèves vont éprouver des difficultés à mettre en place des variables itératives correspondant aux bornes des intervalles imbriqués et au nombre à proposer : le nombre choisi à chaque étape (par la suite, nous désignerons ce nombre par NbCh), et les bornes de l'intervalle (par la suite, nous désignerons par Binf et Bsup ces deux bornes). Nous référant à la littérature sur les difficultés conceptuelles des débutants relativement aux variables informatiques présentée au chapitre 1 de la partie I, nous nous attendons à ce que l'introduction de ces variables itératives soit source de difficultés. En effet, NbCh est calculé à chaque étape à partir de Binf et Bsup. L'évolution de Binf et Bsup est complexe, car chacune peut selon le cas ne pas changer de valeur, ou prendre la valeur de NbCh.

3.3 Phase 3 (algorithme de joueur B – formalisation sous formes « textuelle » et « spatiale » – implémentation de l'algorithme dans l'environnement numérique)

La formalisation de l'algorithme du joueur B sous forme d'un organigramme ayant été faite lors de la phase précédente, les élèves doivent procéder à son implémentation sur ordinateur

via l'utilisation d'un environnement de programmation sur ordinateur qui sera choisi par chaque binôme parmi deux propositions (*LARP* et *AlgoBox*). Pour cela, ils doivent transformer l'algorithme « dessiné » à la phase précédente sous forme d'un organigramme en un algorithme formalisé pour l'environnement choisi. Lors de cette phase, nous pensons que les élèves peuvent tenir compte des possibilités qu'offre l'environnement de programmation, notamment les structures « Tant Que…Faire » ou « Répéter… Jusqu'à », mais nous nous attendons aussi à des hésitations, les élèves pouvant ne pas voir directement quelle structure correspond au dessin fait en phase 2.

Nous supposons aussi que l'élève va rencontrer des difficultés d'ordre syntaxique dues au langage adopté dans l'environnement de programmation. Ainsi, nous nous attendons lors de cette phase à ce que l'enseignant soit mis dans l'obligation d'intervenir pour donner des indications sur la syntaxe du langage, afin de faire que les difficultés de l'élève ne soient pas dues à cette syntaxe. Nous pensons que le problème d'obtenir une valeur entière à chaque calcul d'une moyenne va être ici une réelle difficulté pour les élèves. Nous nous attendons à ce que les élèves recherchent parmi les fonctionnalités proposées par le logiciel celle qui permettent de résoudre ce problème. Nous supposons que les élèves, n'ayant pas le concept de fonction partie entière ne vont pas voir que ces valeurs entières peuvent être données par une fonction et vont rester bloqués du fait de leur conception d'un calcul d'arrondi à l'unité près au cas par cas.

4. Mise en place de l'expérimentation, déroulement et analyse *a posteriori*

4.1 Les classes, les supports d'observations

L'expérimentation dans son ensemble a été faite en présence du chercheur. Pour chaque classe y participant, les élèves ont été en salle informatique avec leur professeur de mathématiques. Les trois classes concernées par cette expérimentation (une Seconde, une Première Scientifique et une Terminale Scientifique) étaient constituées du même nombre d'élèves, soit trente élèves par classe. L'expérimentation en classe était prévue durer entre une centaine de minutes et deux heures (suivant le niveau scolaire de la classe) : soit une vingtaine de minutes pour la phase 1, entre 40 et 70 minutes pour la phase 2 et entre 30 et 40 minutes pour la phase 3. L'ensemble de ces phases a été faite sur deux séances de travaux

dirigés d'une heure se succédant sur une même après-midi avec une pause de 10 minutes entre les deux séances.

Les classes de Seconde et Terminale Scientifique étaient des classes d'un même lycée (lycée E. Galois à Sartrouville) avec un public très hétérogène, issu de différentes origines sociales et ethniques. Quant à la classe de Première Scientifique, elle était au lycée Richelieu de Rueil-Malmaison. Cette classe était constituée d'un public très homogène, enfants de cadres ou enseignants, en bonne réussite scolaire, une particularité de la classe était son bilinguisme (Français-Allemand).

L'analyse *a posteriori* a pu être construite à partir de plusieurs supports : les enregistrements vidéo et audio des séances, les observations du travail des élèves et des réactions de leur enseignant. Ces divers supports ont favorisé l'étude et l'analyse des algorithmes produits lors des deux séances par les élèves. A la demande du chercheur, les élèves ont rendu à leur professeur un écrit avec un enregistrement sur clé USB de leurs algorithmes implémentés dans l'environnement numérique.

4.2 Phase 1 (algorithme du joueur A – conception générale. Durée : 20 minutes en Seconde – 15 minutes en Première et Terminale Scientifique)

4.2.1 Le déroulement

Dans cette phase, les enseignants ont réparti les élèves en binômes sans tenir compte de singularités particulières. Dans chaque binôme, les élèves avaient pour tâches de se familiariser avec la situation en jouant à deux, puis de concevoir un algorithme « papier-crayon », sous une forme « textuelle » (langage naturel ou pseudo-code) ou « spatiale » (organigramme), représentant les actions du joueur A, c'est-à-dire le fait de penser au nombre entier « secret » et de répondre aux propositions du joueur B. Dans le déroulement de cette phase, nous ne distinguerons pas les niveaux scolaires des classes observées car les différences dans la conduite des élèves et des enseignants sont peu significatives.

Dans chaque binôme, les élèves se sont répartis les actions des joueurs A et B. Pendant cinq minutes, ils ont joué au jeu en respectant la règle. Après ces cinq premières minutes de jeu, pour chaque classe observée, l'enseignant a demandé aux élèves de se diriger vers un traitement algorithmique correspondant à l'action du joueur A.

Sachant que, les manuels scolaires utilisés par les élèves des classes observées ne donnent aucune indication sur la construction d'un organigramme, les enseignants avaient distribué la semaine précédant les séances un polycopié (voir Annexe 1) fourni par le chercheur, donnant les principes de base de construction d'un organigramme, illustré de quelques exemples. Tout au long de cette phase, ils ont rappelé régulièrement aux élèves que l'algorithme « papier-crayon » proposé doit être écrit soit sous forme « textuelle » soit sous une forme « spatiale ».

Les élèves de Seconde ont généralement choisi une représentation de l'algorithme « papier-crayon » sous sa forme « spatiale » via l'organigramme, tandis que les élèves du cycle scientifique ont privilégié une représentation « textuelle ».

Lors de la construction de l'algorithme certains binômes ont eu d'emblée des difficultés avec la représentation des données : dans un cas, le nombre aléatoire n'était pas affecté à une variable, mais rappelé pour chaque comparaison, les élèves n'ayant pas conscience qu'ainsi ce nombre changeait à chaque test ; dans un autre cas, la réponse du joueur B était affectée à une variable en dehors de la boucle et était donc constante au cours de l'exécution.

Voici par exemple (Tableau 28) un algorithme rendu en fin de phase. Le binôme utilise correctement une variable « Nbrpr » pour la réponse du joueur B, mais pas pour le nombre secret. Nous pensons que les élèves de ce binôme considèrent que le choix de ce nombre par la machine pourrait reste constant chaque fois que la machine le « réécrit ».

La machine choisit un nombre entier secret de façon aléatoire entre 0 et 1000
Répéter
    Le joueur B propose un nombre entier compris entre 0 et 1 000 qu'il stocke dans Nbrpr
    Si le nombre Nbrpr < au nombre entier secret que la machine choisit de façon aléatoire
    La machine répond : « Trop Petit »
    Sinon si le Nbrpr > au nombre entier secret que la machine choisit de façon aléatoire
    La machine répond « Trop Grand »
Jusqu'à ce que Nbrpr soit égal au nombre entier secret que la machine choisit de façon aléatoire
La machine répond alors « Bon nombre » qui est égal au dernier Nbrpr

Tableau 28

Voici un autre exemple (Tableau 29). Le binôme considère correctement deux variables, mais place l'instruction de lecture en dehors de la boucle:

La machine choisit aléatoirement un nombre entier secret entre 0 et 1000 qu'elle mémorise dans NbrSec *Le joueur B propose un nombre entier compris entre 0 et 1 000 qu'il mémorise dans NbrProp* *Répéter* *Si le nombre NbrProp < NbrSec* *La machine répond : « Trop Petit »* *Sinon La machine répond « Trop Grand »* *Jusqu'à ce que NbrProp = NbrSec* *La machine répond « Bon nombre » : NbrProp*

Tableau 29

Nous avons pu observer que parmi les binômes de Seconde une grande majorité (près de 70%) conçoive le choix du nombre entier « secret » et sa mémorisation « nombre entier secret » comme une « action » de la machine, plutôt que comme l'affectation à cette variable du résultat rendu par une fonction. Par exemple, un binôme écrit : « La machine choisit aléatoirement un nombre entier secret entre 0 et 1000 qu'elle mémorise dans NbrSec ». Dans d'autres cas, il y a bien affectation, mais la donnée à affecter n'a pas de statut informatique (par exemple l'élève affecte « nombre entier secret » à la variable « NbrSec » (fig. 80)).

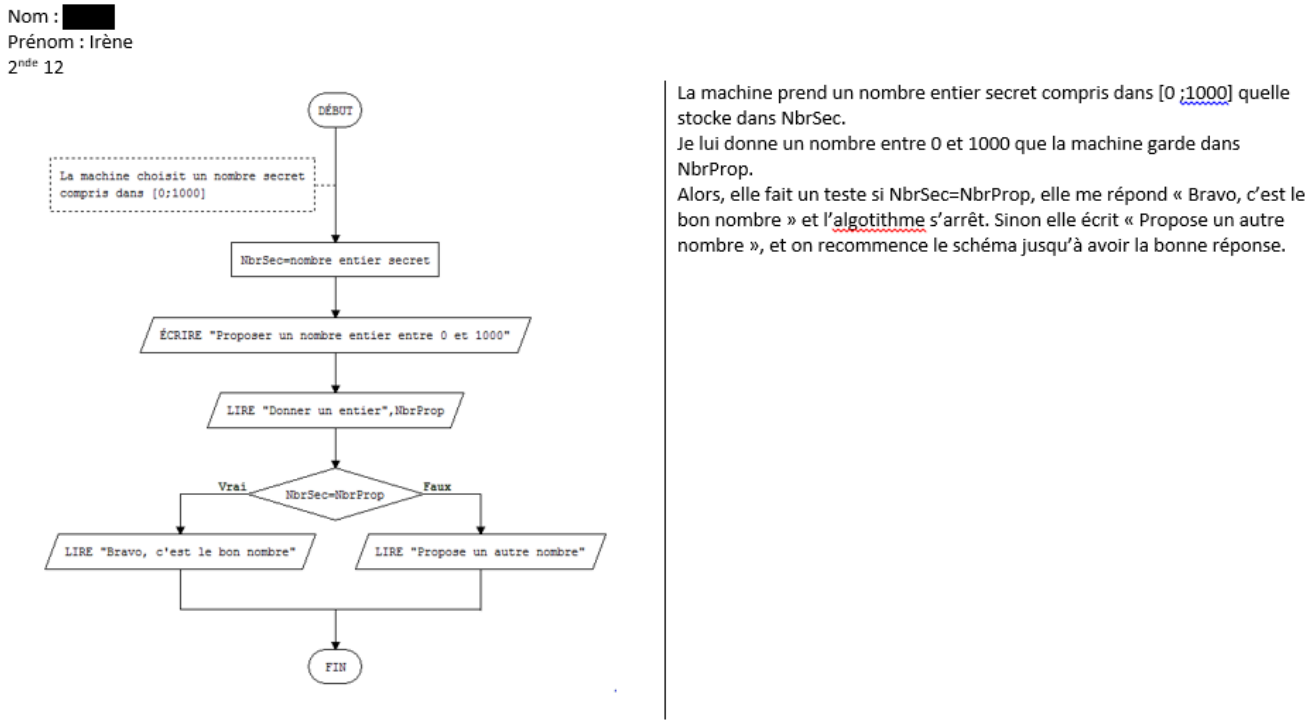


Figure 80 (Une copie d'élève où celui-ci affecte « nombre entier secret » à la variable « NbrSec »)

Dans l'exemple ci-dessus, nous observons aussi que l'élève suppose que le renvoi à la proposition par l'utilisateur du programme se fait de façon implicite à la sortie de la structure conditionnelle dans le cas où la réponse proposée est erronée.

Lors des premières ébauches d'algorithme du joueur A, 60% des binômes de Seconde et

20% des binômes du cycle terminal ont produit une écriture où l'itération est exprimée avec un nombre indéterminé de blocs lecture, alternatives imbriquées, test d'égalité. Dans le cas d'un organigramme, cela se traduit dans un schéma (fig. 81) où ce bloc est répété un petit nombre de fois et suivi d'une case « … » ou « etc. ». Pour le test, ces élèves n'ont pas utilisé une alternative complète qui aurait permis de renvoyer à l'instruction initiale. Dans le cas d'une représentation « textuelle », nous avons de même un bloc répété comprenant deux tests en alternative simple, suivi d'une ligne « … » ou « etc. » (fig. 82). Ces élèves n'ont pas utilisé spontanément les marqueurs d'itération « Tant que » ou « Répéter ».

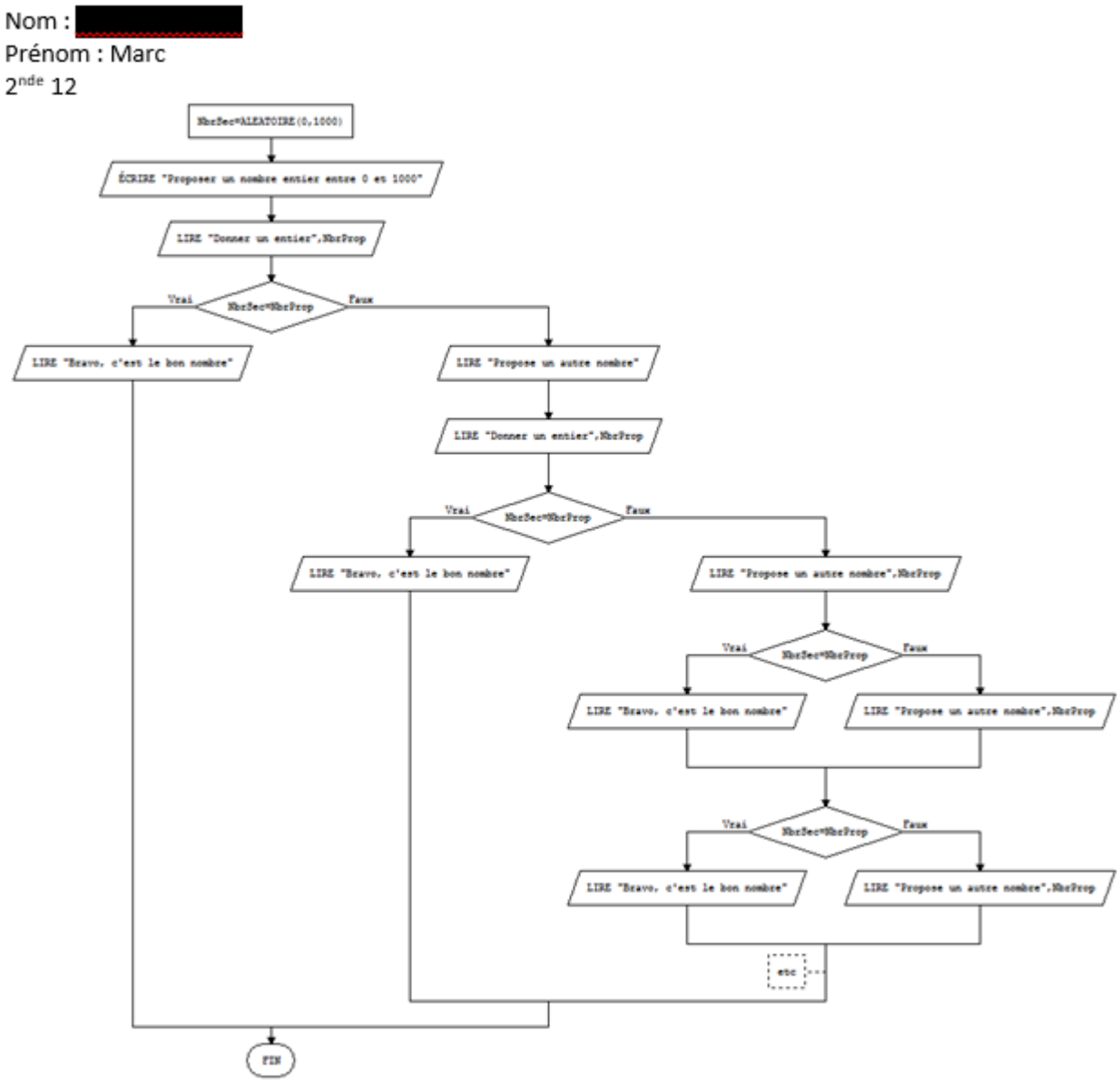
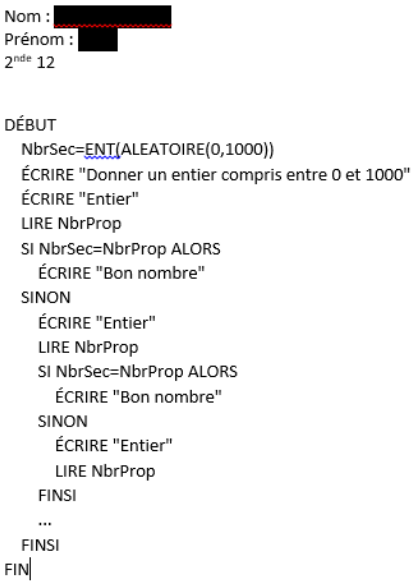


Figure 81 (Extrait d'un organigramme obtenu par un élève n'ayant pas utilisé les marqueurs d'itération « Tant que » ou « Répéter » pour simuler le joueur B)



Nom : [redacted]
Prénom : [redacted]
2nde 12

```
DÉBUT
  NbrSec=ENT(ALEATOIRE(0,1000))
  ÉCRIRE "Donner un entier compris entre 0 et 1000"
  ÉCRIRE "Entier"
  LIRE NbrProp
  SI NbrSec=NbrProp ALORS
    ÉCRIRE "Bon nombre"
  SINON
    ÉCRIRE "Entier"
    LIRE NbrProp
    SI NbrSec=NbrProp ALORS
      ÉCRIRE "Bon nombre"
    SINON
      ÉCRIRE "Entier"
      LIRE NbrProp
    FINSI
    ...
  FINSI
FIN
```

Figure 82 ((Extrait d'un algorithme au format « textuel » obtenu par un élève n'ayant pas utilisé les marqueurs d'itération « Tant que » ou « Répéter » pour simuler le joueur B)

Ayant observé ces difficultés, à la moitié du temps prévu pour cette phase, soit environ 15 minutes après le début de la phase, les enseignants ont pris l'initiative de mener une intervention en collectif. Ainsi, ils proposèrent qu'un élève d'un binôme en difficulté passe au tableau afin de présenter la succession des étapes du jeu selon les points de vue des joueurs A et B. En effet, les enseignants demandèrent à l'élève désigné pour cette intervention collective de schématiser les tâches des joueurs A et B sous forme d'un tableau à deux colonnes, où la première représenterait les propositions du joueur B et la deuxième les réponses du joueur A en fonction des propositions du joueur B. Ainsi, nous avons eu comme représentation, le tableau (Tableau n° 30) suivant :

Nombres proposés par le joueur B	Réponses du joueur A
R_1	« Trop petit » ou « Trop grand »
…	…
R_n	« Bon nombre »

Tableau 30

Dans la colonne associée au joueur B, la suite de nombres entiers proposés par le joueur B est constituée des nombres R_1, R_2, R_3, … Quant à la colonne du joueur A, nous avons ses réponses possibles en fonction des nombres proposés par le joueur B. De plus, pour compléter cette information, les enseignants proposèrent que le nombre entier secret pensé par le joueur A soit nommé N. Pour compléter cette approche, les enseignants signalèrent aux élèves, qu'ils devaient voir les nombres N, R_1, R_2, R_3,… comme les variables devant intervenir dans l'algorithme du joueur A. Quant aux réponses « Trop petit », « Trop grand » ou « Bon

nombre », les enseignants demandèrent dans chaque classe, à l'ensemble des élèves à quoi correspondait ces réponses pour l'algorithme du joueur A.

Les ordinateurs devaient être tous en veille au cours de cette phase. Cependant, certains binômes ont souhaité vérifier si le travail qu'ils faisaient sur l'algorithme « papier-crayon » était réellement programmable et correspondait aux attentes de cette phase sur le joueur A. Nous avons eu par exemple comme retour des travaux de deux binômes une image de leurs écrans (fig. 83 et 84) représentant l'algorithme du joueur A et non un algorithme « papier-crayon » comme il l'avait été demandé au début de la phase et rappelé par les enseignants tout au long de cette phase.

Par la suite, les binômes ayant des difficultés avec la représentation des données, ont généralement utilisé des variables de façon appropriée et à l'issue de la phase, la grande majorité des binômes a produit une itération correcte. Tous les élèves de Seconde et 25% des élèves de Première ont produit des organigrammes présentant une boucle « Répéter… Jusqu'à » (fig. 85). Il s'est avéré que seuls ces élèves de Première avaient suivi la consigne de lire au préalable le polycopié sur les organigrammes. Les autres élèves ont utilisé une boucle « TantQue…Faire » dans une présentation « textuelle » (fig. 86).

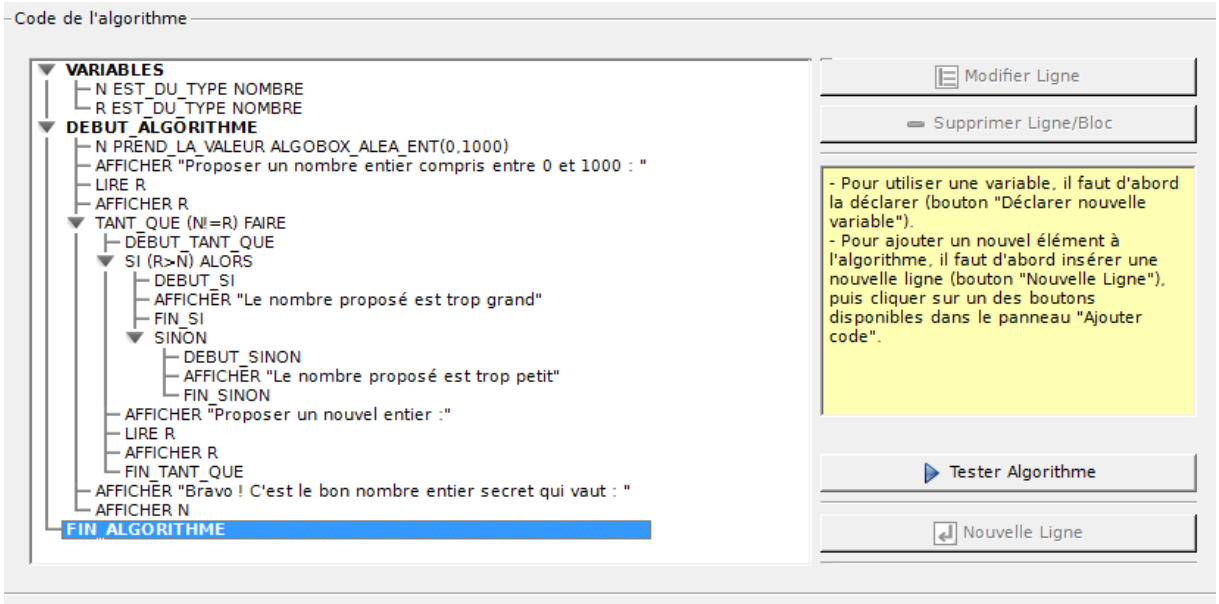


Figure 83 (Algorithme proposé par un élève représentant les réponses du joueur A (machine) en fonction des réponses du joueur B (utilisateur))

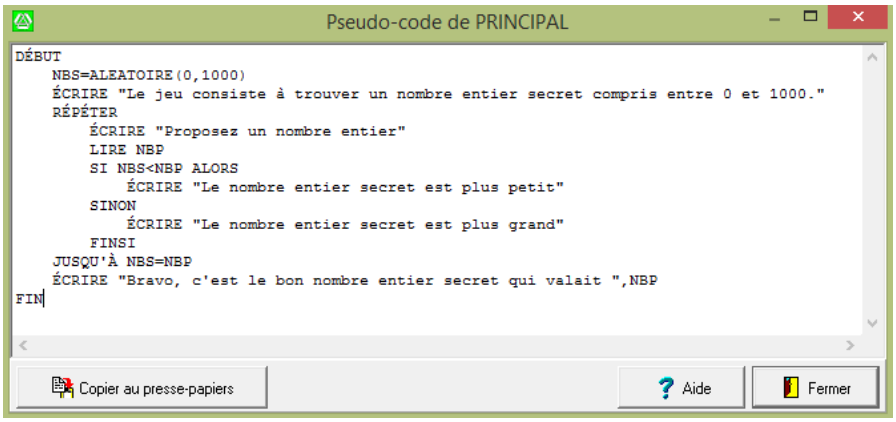


Figure 84 (Algorithme proposé par un élève représentant les réponses du joueur A (machine) en fonction des réponses du joueur B (utilisateur))

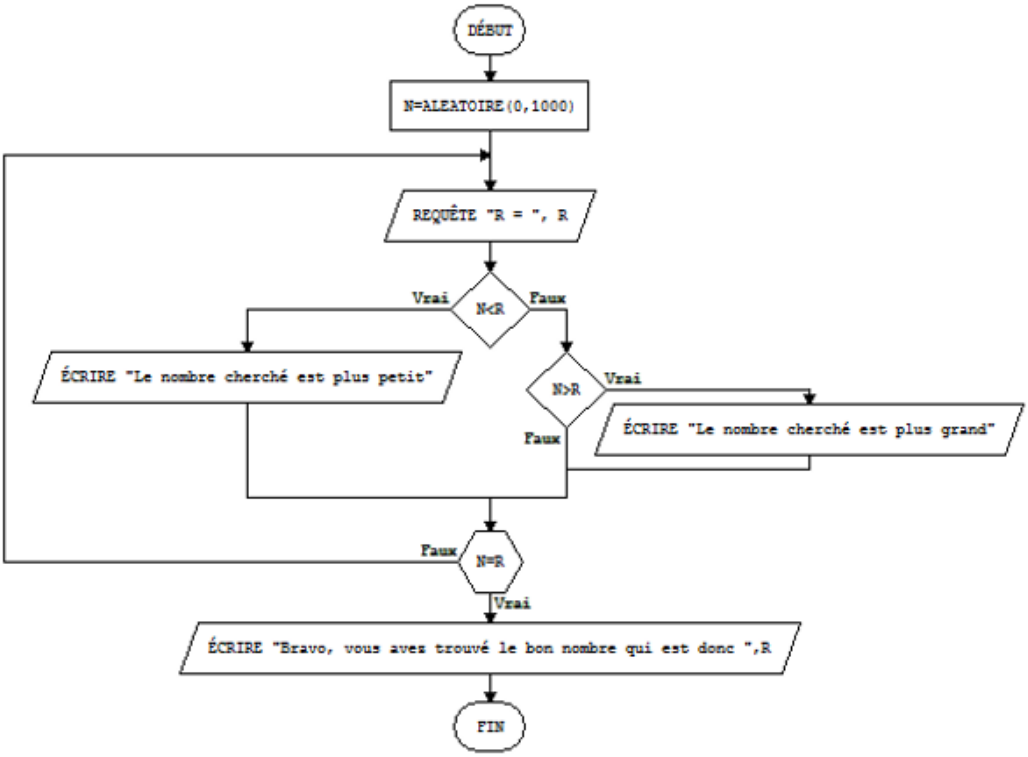


Figure 85 (Organigramme proposé par un élève représentant les réponses du joueur A (machine) en fonction des réponses du joueur B (utilisateur))

Nom : [redacted] 1ère S 4
Prénom : [redacted]

Algorithme du joueur A

```
DEBUT
  Nbsec=ALEATOIRE(0,1000)
  REQUETE "R =", R
  TANTQUE Nbsec!=R FAIRE
    SI R>Nbsec ALORS
      ECRIRE "Le nombre entier secret est plus petit"
    SINON
      ECRIRE "Le nombre entier secret est plus grand"
    FINSI
    REQUETE "Proposer un nouveau nombre entier secret", R
  FINTANTQUE
  ECRIRE "Bravo ! C'est le bom nombre entier secret"
FIN
```

Figure 86 (Algorithme du joueur A proposé par un élève)

Dans les classes du cycle scientifique terminal, après l'intervention collective au tableau, le travail est allé plus loin que ce qui avait été prévu pour cette phase. Les enseignants ayant souligné la nécessite de « stocker les réponses du joueur B », certains élèves du cycle terminal ont alors posé le problème d'une stratégie qui serait à mettre en place afin de trouver en un minimum de coups le nombre entier « secret ». Cependant, cette problématique ne faisait pas partie des actions du joueur A. Mais les enseignants ne firent aucune remarque sur ce point et laissèrent les binômes concernés travailler sur cette problématique. Lors des débriefings qui eurent lieu pendant la pause, les enseignants signalèrent que leur intervention collective avait été guidée par la lecture d'une ressource proposée par l'IREM qu'ils avaient trouvée sur Internet dans les jours qui ont précédé l'expérimentation.

4.2.2 Analyse *a posteriori*

Comme pour le déroulement de la phase, nous ne distinguons les trois niveaux scolaires que quand cela s'avère pertinent.

4.2.2.1. La sous-phase de jeu

Globalement, le moment de jeu est conforme à l'analyse *a priori*. Les élèves de Seconde jouent sans envisager une stratégie systématique à la différence des deux autres niveaux Dans la pratique, la résolution d'équations par dichotomie est vue au mois de décembre de la classe de Seconde, alors que l'expérimentation a eu lieu en septembre. On peut inférer de cette observation sur la partie jeu de la phase, que la dichotomie n'est en rien pour des élèves de Seconde un processus spontané. En revanche, les élèves de Première et Terminale Scientifiques transfèrent facilement au cas discret la méthode vue en Seconde pour la résolution approchée d'équations. Notons aussi que cette sous-phase ne particularise pas l'algorithme du joueur A, ce qui peut contribuer aux difficultés à démarrer la sous-phase de construction de cet algorithme en particulier au niveau de la Seconde.

4.2.2.2. La sous-phase de construction de l'algorithme joueur A

a) La formalisation de l'algorithme

Lors de l'expression de l'algorithme écrit au « papier-crayon », nous observons qu'en Seconde les élèves privilégient l'organigramme, alors que dans les autres classes, une formulation « textuelle » de l'algorithme est largement dominante. Nous interprétons cela comme étant une conséquence du fait que les élèves du cycle scientifique aient déjà mis en

pratique plusieurs algorithmes en Seconde, et en Première pour la Terminale, et que conformément à ce qui est majoritairement proposé par les manuels scolaires des trois niveaux scolaires, une présentation « textuelle » d'un algorithme est privilégiée à celle d'une présentation « spatiale ». En revanche, comme les élèves de Seconde, n'ont pas encore été trop « formatés » à telle formalisation d'un algorithme, ils sont plus libres dans leur choix pour représenter un algorithme.

b) Données et variables

Lors de la construction de l'algorithme, la mise en place d'une variable initiale concernant le nombre entier « secret » choisi par le joueur A est bien envisagée par la majorité des élèves, et ceci quel que soit leur niveau scolaire. Cependant, nous pouvons observer chez certains élèves de Seconde, des difficultés avec la représentation de cette donnée. Nous citons le cas d'un binôme qui, dans la boucle, compare le nombre proposé par le joueur B avec la donnée d'un nombre secret que la machine choisit de façon aléatoire. Ainsi, ce binôme ne semble pas avoir conscience que cette donnée va alors changer à chaque test mis en place dans la boucle « Répéter … Jusqu'à ». Nous supposons qu'il y a une interprétation erronée de la part de ces élèves sur le principe de stockage en mémoire dans une machine. Ceci rejoint certaines analyses sur les difficultés des débutants à se représenter les traitements des données dans un algorithme (Lagrange, 1991), ainsi que des observations recueillies dans l'usage d'un tableur pour des simulations dans l'enseignement des probabilités-statistiques (Bui Ahn, 2015).

Nous rapportons aussi une difficulté assez générale en Seconde à concevoir le choix du nombre entier « secret » comme l'affectation du résultat rendu par une fonction. Certains élèves de Seconde et tous les élèves du cycle terminal réussissent néanmoins, et nous interprétons cela comme résultant de l'usage de la fonction ALEA du tableur lors de travaux en probabilités-statistiques.

Nous observons aussi en Seconde des difficultés relatives à la position de l'affectation par lecture du choix de la proposition du joueur B à une variable : celle-ci se trouve alors avant l'itération plutôt que dans le corps de boucle. Ceci renvoie aux difficultés conceptuelles relatives à l'itération repérées dans la littérature (cf. Chapitre 1 de la partie 1).

c) L'itération

Nous observons que les élèves de Seconde et certains de Première se dirigent initialement vers la construction d'une série d'alternatives en nombre indéterminé (fig. 87), ce à quoi nous nous attendions pour une ingénierie qui se déroule au cours du mois de septembre de l'année scolaire. Cette construction, comme nous l'avons vu dans l'analyse *a priori* ne peut aboutir à la différence de situations où le nombre d'itérations est connu à l'avance.

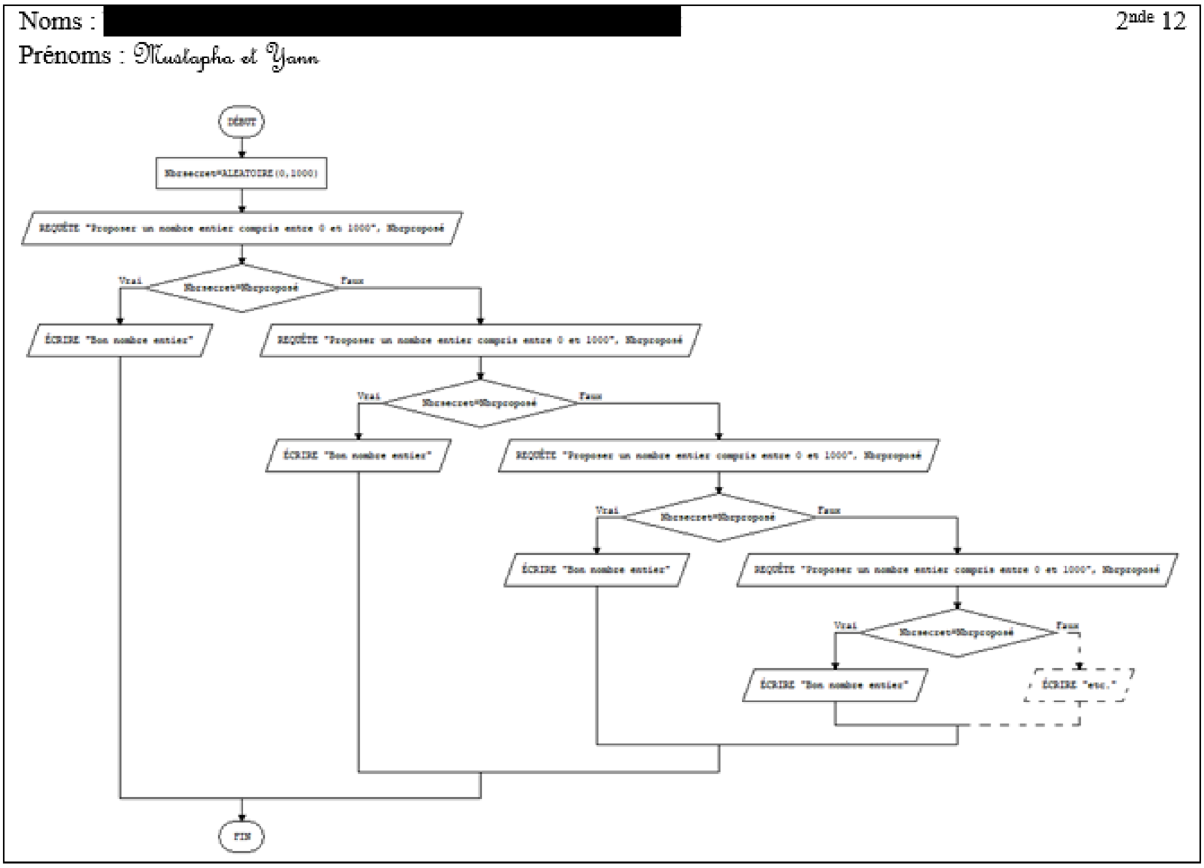


Figure 87 (Dans le cadre d'un algorithme du joueur A : construction proposée par deux élèves d'une série d'alternatives en nombre indéterminé)

Ainsi, une utilisation appropriée de structures itératives nous montre que celle-ci nécessite de la part de l'élève une identification et une construction d'un invariant de boucle (« Mise à jour »), d'une condition de continuation (« Test ») et d'une initialisation des variables.

d) Le type d'itération dans les solutions

Comme nous l'avons observé, les élèves de Seconde privilégient une structure où la sortie de boucle est placée à la fin de la boucle. Cela se traduit par une branche d'alternative comportant un renvoi en début de boucle, ou l'utilisation de la structure « Répéter …

Jusqu'à ». Ce type de structure est encore bien présent au niveau de la Première en particulier chez les élèves qui utilisent l'organigramme. Ceci confirme notre analyse *a priori* selon laquelle une sortie de boucle placée à la fin de boucle est la plus naturelle pour l'algorithme considéré.

Bien que la structure « TantQue » impose une écriture peu naturelle, notamment la répétition dans la partie initialisation, d'une partie du corps de boucle, les élèves qui la retiennent, ne rencontrent pas de difficultés particulières. Il s'agit principalement d'élèves de Première et Terminale qui ont un bon niveau et qui maitrisent cette structure.

e) La présence de l'ordinateur dans la salle et le rôle de l'enseignant

Du fait que les trois classes aient été au cours des séances toujours en salle informatique et malgré une interdiction formelle d'utiliser l'ordinateur au cours de cette phase, certains binômes n'ont pu s'empêcher d'utiliser les ordinateurs qui n'étaient pas éteints mais seulement mis en veille. Selon l'analyse de Rogalski et Samurçay (1990), la prise en compte des contraintes du dispositif informatique est nécessaire pour passer d'une série « indéterminée » d'alternatives à une itération avec condition d'arrêt. Bien que nous ne disposions pas d'enregistrements attestant un rôle joué par l'utilisation d'un ordinateur, nous pensons que les contraintes syntaxiques et les fonctionnalités proposées dans le langage de l'ordinateur ont pu aider à construire une structure adéquate.

Dans les trois classes, les enseignants sont intervenus avec la présentation d'un tableau pour clarifier le déroulement du jeu. Ils ont de ce fait mis l'accent sur les variables du programme. Le nombre secret est noté par une lettre, tandis que la réponse du joueur B est exprimée en notation indexée, comme pour une suite, pour marquer son caractère évolutif. Nous nous interrogeons sur l'influence de ce choix concernant la conception de cette variable puisque, en informatique, une variable ne change pas de nom au cours de l'évolution du processus. Nous serons amenés à observer lors de l'analyse de la phase suivante quelle en aura pu être la conséquence chez les élèves pour leur conception de l'algorithme du joueur B.

4.3 Phase 2 (algorithme de joueur B – conception générale et écriture d'un organigramme papier-crayon. Durée : 70 minutes en Seconde – 40 minutes en Première et Terminale Scientifique)

4.3.1 Le déroulement

Pour décrire le déroulement de cette phase, contrairement à la phase 1, nous sommes obligés de distinguer la classe de Seconde, des classes de Première et Terminale Scientifiques. Cette phase a été abordée dans les trois classes, une fois que la phase 1 fut considérée par l'enseignant en charge de la classe, comme totalement terminée.

Nous rappelons que les élèves doivent d'abord travailler sur une stratégie « gagnante » et « rapide » à mettre en place par le joueur B lors du jeu pour trouver le nombre entier « secret » en « un minimum de coups ». Puis, ils doivent élaborer un algorithme « papier-crayon » du joueur B sous forme d'un organigramme.

4.3.1.1. En Seconde (Durée : 70 minutes au lieu des 60 minutes prévues initialement)

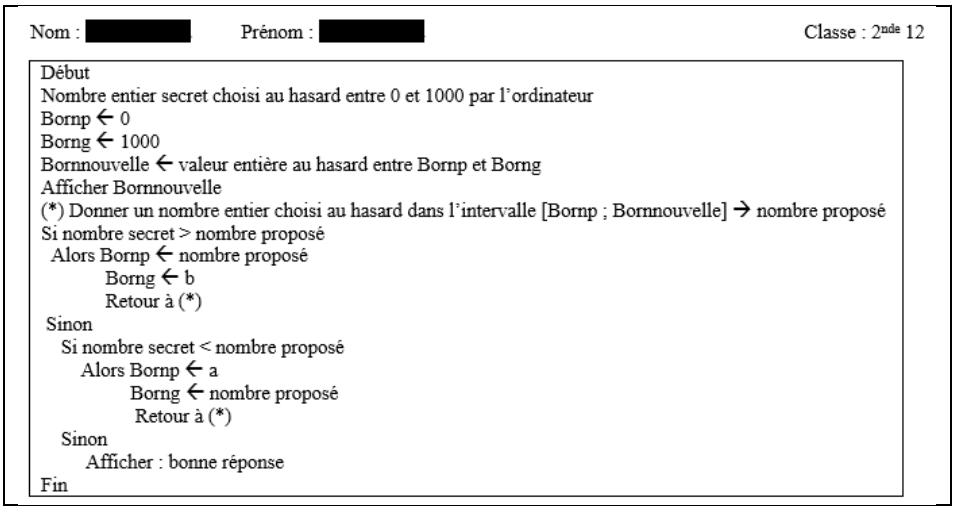
a) Le choix d'une stratégie

Au début de cette phase, malgré la courte intervention de l'enseignant au début de la première phase lors de la pratique manuelle du jeu, un certain nombre d'élèves persistent à concevoir l'action du « joueur B » comme consistant à proposer un nombre entier en tenant compte de la réponse du joueur A (« plus petit » ou « plus grand ») et donc de l'intervalle contenant le nombre secret, mais sans règle précise de choix d'un nombre à proposer. Cette absence de prise de conscience de la nécessité d'une règle précise les empêche d'élaborer un algorithme. Cependant, au cours de cette seconde phase axée sur l'action du joueur B, apparait assez rapidement l'idée qu'une « stratégie » doit s'imposer sinon, comme le dit un élève, « le jeu devient trop long ». Au bout d'une dizaine de minutes, un binôme s'adressant à l'enseignant entame un échange à voix haute avec lui. Le reste de la classe écoute dans un calme relatif. Afin de faciliter la transcription de cet échange, nous notons les élèves de ce binôme « Bin1 » et « Bin2 », et l'enseignant « EnsSec ».

Bin1 : *Monsieur, nous avons trouvé une stratégie.*
EnsSec : *Nous vous écoutons.*
[L'enseignant demande aux autres binômes d'interrompre leur travail et d'écouter]
Bin1 : *Nous pouvons couper l'intervalle du départ en deux sous-intervalles de même amplitude à une unité près.*
EnsSec : *Oui, et après ?*

Bin1 : [Temps d'hésitation de 5 secondes environ] *Euh… On prend un entier au hasard dans un des deux sous-intervalles obtenus et on le compare au nombre entier secret*.
Bin2 : *Nous avons un test, Monsieur.*
EnsSec : *C'est-à-dire ?*
Bin2 : [Temps de silence de quelques secondes] *Eh bien,… euh… si le nombre entier secret est plus grand que ce nombre proposé, alors on garde comme nouvel intervalle de recherche l'intervalle ayant pour borne inférieure le nombre proposé et comme borne supérieure l'ancienne borne supérieure de l'intervalle de recherche précédant et…* [Temps d'hésitation]
EnsSec : *Et alors ?*
Bin1 : *Monsieur, si le nombre entier secret est plus petit que le nombre proposé, alors le nouvel intervalle de recherche a pour borne inférieure celle de l'intervalle de recherche précédant et pour borne supérieure le nombre proposé, et si le nombre proposé est le nombre entier secret, alors on arrête là.* [Temps de silence]
Bin2 : *Si ce n'est pas le bon nombre secret, on recommence ce que nous venons de dire.*
EnsSec : *On recommence quoi ?*
Bin1 : *Ben, le processus de découpage et de comparaison.* […]

L'enseignant demande alors aux autres binômes de dire ce qu'ils en pensent. Les élèves partent dans un premier temps sur cette stratégie et ainsi deux binômes proposent un début d'algorithme « textuel » écrit au « papier-crayon » du joueur B basée sur cette stratégie (fig. 88).


Nom : [redacted] Prénom : [redacted] Classe : 2nde 12

Début
Nombre entier secret choisi au hasard entre 0 et 1000 par l'ordinateur
Bornp ← 0
Borng ← 1000
Bornnouvelle ← valeur entière au hasard entre Bornp et Borng
Afficher Bornnouvelle
(*) Donner un nombre entier choisi au hasard dans l'intervalle [Bornp ; Bornnouvelle] → nombre proposé
Si nombre secret > nombre proposé
Alors Bornp ← nombre proposé
Borng ← b
Retour à (*)
Sinon
Si nombre secret < nombre proposé
Alors Bornp ← a
Borng ← nombre proposé
Retour à (*)
Sinon
Afficher : bonne réponse
Fin

Figure 88 (Un algorithme du joueur B proposé par un élève de 2nde)

Nous pouvons supposer que la structure de l'algorithme du « joueur A » semble être réinvestie, et que le choix du nombre à proposer reste une difficulté majeure chez certains élèves.

Au bout d'une vingtaine de minutes, la classe semble ne pas avoir évolué dans l'expression de la stratégie sous forme d'un algorithme, quand un des élèves (Bin1) du binôme à l'origine du choix de cette première stratégie interpelle l'enseignant et dit à voix haute : *Il ne faut pas prendre au hasard un nombre à proposer dans un des deux sous-intervalles, mais plutôt la*

frontière entre les deux sous-intervalles. L'enseignant demande alors : *Qu'entends-tu par frontière ?* . Après une brève hésitation, l'élève reprend la parole et dit : *le milieu de l'intervalle* (Fig. 89).

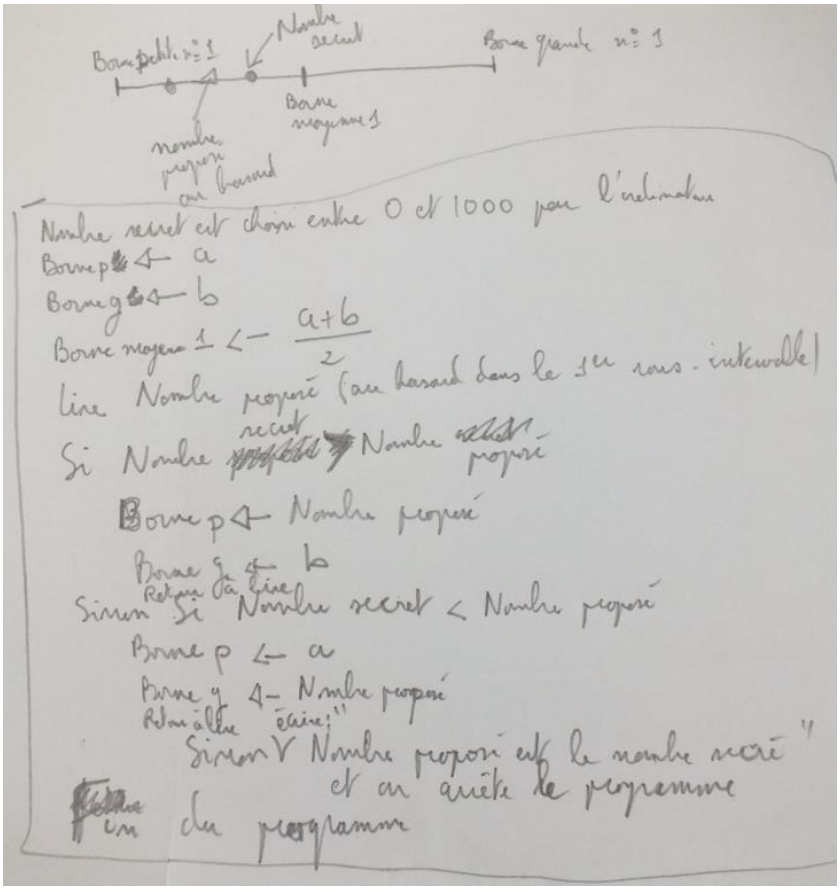


Figure 89 (Une copie d'élève avec représentation graphique du découpage d'un intervalle et algorithme)

L'enseignant demande alors aux autres élèves ce qu'ils en pensent. Les élèves ne répondent pas dans un premier temps. L'ensemble des binômes semblent faire des essais avec ce que vient de proposer l'élève sur le « milieu » de l'intervalle. Un autre élève prend alors la parole et dit : *Monsieur, est-ce que logiciel que nous allons utiliser après possède une fonction « Milieu » ?* . L'enseignant répond : *Je ne peux te répondre pour l'instant à ta question. Mais, penses-tu nécessaire d'avoir besoin d'une fonction « Milieu » pour calculer le « milieu » d'un intervalle ?*. Un autre élève prend alors la parole et dit : *On n'en a pas besoin, car finalement le milieu de l'intervalle est la moyenne des bornes de l'intervalle*. L'enseignant profitant de cette dernière remarque, demande à l'ensemble de la classe : *Avez-vous bien entendu ce que vient de dire J. ? Qu'en pensez-vous ?*. L'ensemble de la classe approuve l'intervention sur la « moyenne des bornes de l'intervalle » de leur camarade.

Quelques minutes après cette intervention, plusieurs élèves de différents binômes lèvent le doigt. L'enseignant donne la parole a un de ces élèves qui répond : *Monsieur, que faisons-nous quand nous obtenons un nombre à virgule après le calcul de la moyenne des bornes ?*. En effet, la majorité des binômes observant ce fait pense alors que cette stratégie leur paraît inadéquate car elle peut conduire la machine tenant le rôle du joueur B, à proposer des nombres non entiers, contrairement à la première stratégie qui avait été présentée et où on ne travaillait que sur des nombres entiers. L'enseignant reprend alors la parole et demande aux élèves de garder quand même cette stratégie basée sur le calcul des moyennes et leur demande de déterminer une « technique de calcul » qui permettrait de travailler à la fois avec les moyennes des bornes et des nombres entiers.

b) Rendre entière la moyenne des bornes quand cela est nécessaire

Les élèves cherchent alors une « technique de calcul » qui permettrait de corriger ce fait en faisant donner par la machine un nombre entier à l'intérieur du dernier intervalle connu, dans le cas où la moyenne n'est pas entière. Après 10 minutes de tâtonnements autour de ce problème de valeur numérique entière, un élève prend spontanément la parole devant l'ensemble des binômes pour proposer une « technique de calcul » reposant sur l'entier le plus proche de la valeur moyenne, se référant explicitement au principe de valeur approchée par excès et par défaut, comme ils ont l'habitude de le voir en cours de Sciences Physique ou en Technologie lors des années de Collège. L'enseignant n'intervient pas et laisse les élèves échanger sur ce point ce qui explique le fait que cette phase va dépasser la durée qui a été prévue initialement (cf. l'analyse *a priori* de cette phase). L'ensemble des élèves accepte alors cette « technique de calcul » comme « solution » permettant de résoudre le problème d'une suite de nombres entiers terminant sur le nombre entier « secret ».

Pendant une dizaine de minutes, les élèves exécutent à la main cette « technique de calcul ». L'enseignant intervient alors en demandant : *comment allez-vous pouvoir transcrire cette « technique de calcul » dans la construction de l'algorithme du joueur B ?* . Ne voyant pas de réponse pertinente émerger dans les réponses données par les élèves, l'enseignant prend alors l'initiative d'aborder pendant 5 minutes, le concept de *fonction partie entière* en signalant que les logiciels qu'ils ont à leur disposition pour la phase 3 ont la possibilité d'utiliser cette fonction. Pour aider à l'élaboration d'algorithme du joueur B aux formats « textuel » ou

« spatial », l'enseignant écrit alors au tableau : *On note cette fonction « ENT »*. Ainsi, après les 55 premières minutes de cette phase, les élèves peuvent commencer à entrer réellement dans l'élaboration d'un algorithme « papier-crayon ».

Un binôme se trouvant très impliqué dans l'écriture d'un algorithme du joueur B et n'ayant pas pris conscience de l'intervention de l'enseignant sur la *fonction partie entière*, appelle l'enseignant à la fin de sa brève intervention et lui présente un algorithme « textuel » du joueur B (fig. 90) ne tenant pas compte du fait que la moyenne des bornes d'un intervalle n'est pas systématiquement un nombre entier.

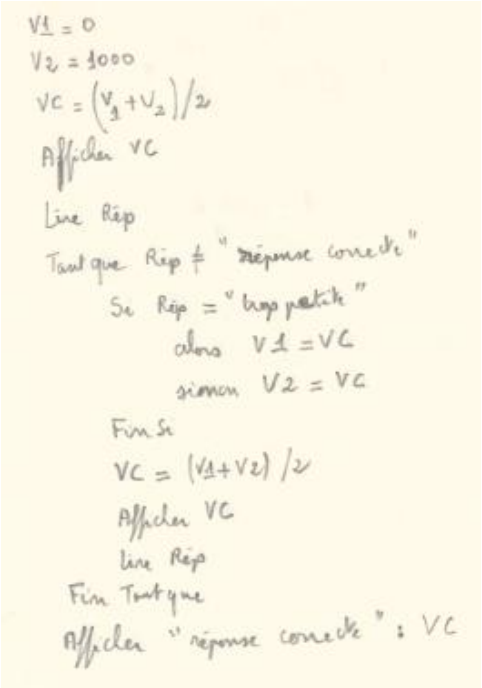


Figure 90 (Un algorithme « textuel » du joueur B ne tenant pas compte du fait que la moyenne des bornes d'un intervalle n'est pas systématiquement un nombre entier)

S'ensuit alors un dialogue entre l'enseignant (EnsSec) et ces deux élèves (El1 et El2) que nous retranscrivons ci-dessous.

EnsSec : *Que pensez-vous de la nature des valeurs numériques des moyennes que vous allez obtenir avec votre algorithme ?*
El1 : *Des entiers, car si on fait (0+1000)/2, on obtient 500, puis si l'intervalle validé est [500 ; 1000], alors la moyenne est 750, puis si c'est [500 ; 750], on a 625 et ainsi de suite. Donc, je ne vois pas où est le problème.*
EnsSec : *Supposons que l'algorithme tourne depuis un petit moment et qu'à un certain instant t l'intervalle valide contenant le nombre secret est* $[125\,;250]$*. Vous refaites alors la procédure. Que donne alors la nouvelle moyenne des bornes ?*
[Un des deux élèves procède au calcul de la nouvelle moyenne.]
El1 : *La moyenne est 187,5.*
El2 : *Oups ! Mais ce n'est pas un nombre entier !*

EnsSec : *En effet ! Et si vous aviez été plus attentif, vous auriez entendu la remarque que je viens de faire à l'ensemble de la classe. Il existe une fonction mathématique qui peut rendre entier tout nombre réel. On la note ENT.*
[Tout en disant cela, l'enseignant leur montre le tableau afin qu'ils puissent prendre connaissance de ce qu'il vient d'écrire sur la fonction partie entière.]
EL1 : *Monsieur, avec cette fonction, 187,5 deviendrait 187 ?*
EnsSec : *Oui. Et à propos qu'elle serait la valeur de tous les nombres réels compris entre 187 et 188 (exclu) ?*
El1 et EL2 simultanément : *187*.
[L'enseignant laisse alors ces deux élèves pour se déplacer vers d'autres binômes]

Cependant, un des deux élèves semble perplexe. En effet, une fois que l'enseignant quitte le binôme, il y a un échange entre les deux élèves qui clôture la discussion sur ce problème de valeur entière :

El2 : *Je peux comprendre que la valeur entière des nombres compris entre 187 et 187,5 soit 187. Mais je ne comprends pas comment la valeur entière d'un nombre compris entre 187,5 et 188 ne soit pas 188.*
El1 : *On vérifiera cela tout à l'heure quand nous pourrons utiliser le logiciel.*

L'ensemble des binômes ayant défini une stratégie en lien avec le calcul de moyennes des bornes des intervalles, les élèves passent à la conception d'un algorithme « papier-crayon » associé à cette stratégie. Pour cela, nous voyons diverses expressions de cet algorithme, même si celle qui leur était demandée initialement, était de dresser un organigramme. En effet, n'ayant pas toujours une maîtrise totale du symbolisme graphique associé aux organigrammes, l'ensemble des binômes a recours en parallèle à une représentation « textuelle » de l'algorithme, mais ils éprouvent des difficultés à l'écrire. En effet, comme nous allons le voir, ils vont éprouver des difficultés sur le choix des structures algorithmiques et la mise en place de variables informatiques.

c) La structure

L'algorithme du joueur B nécessite une itération, structure nouvelle pour les élèves en début de Seconde. Par crainte de manquer de temps, l'enseignant décide alors d'intervenir auprès des binômes en les renvoyant au travail qu'ils avaient mis en place lors de la phase 1 sur l'algorithme du joueur A et en leur précisant que tant qu'ils travaillent au format « papier-crayon », il ne leur est pas nécessaire de tenir compte des contraintes du symbolisme graphique des organigrammes, ni des contraintes du langage informatique pour l'algorithme textuel.

Ainsi, on trouve aux formats « spatial » et « textuel », les algorithmes suivants (Fig. 91 et 92) proposés par certains binômes :

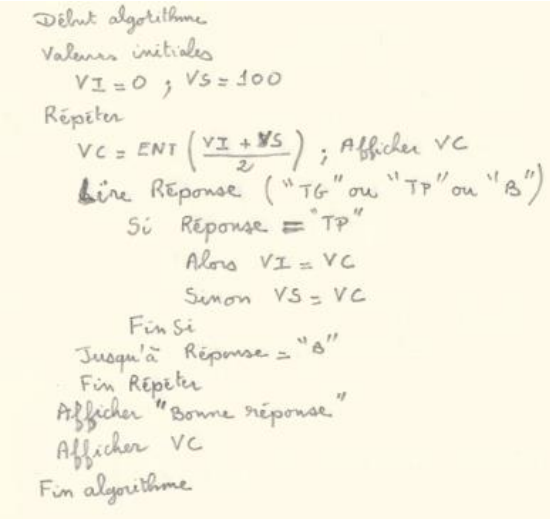


Figure 91 (Structure « Répéter… Jusqu'à », Algorithme du joueur B au format « textuel »)

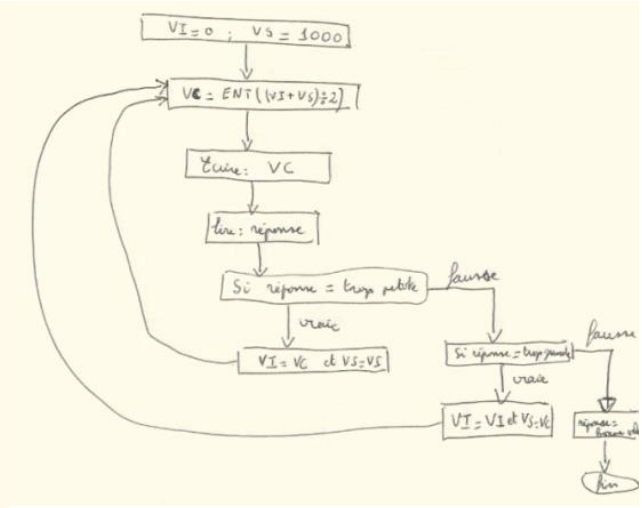


Figure 92 (Algorithme du joueur B au format « spatial »)

Pour l'algorithme textuel (Fig. 91), nous pouvons constater qu'une boucle « Répéter… Jusqu'à » est correctement écrite. Le corps de boucle comporte une seule alternative complète, ce qui fait penser que l'élève a vu que l'instruction dans la partie « SINON » n'entraîne pas de difficulté pour le cas « Réponse = B ». Quant à l'algorithme spatial (Fig. 92), il est du type « Boucle…FinBoucle ». La branche « Si » des deux alternatives pointe sur un bloc qui comporte deux affectations, dont l'une est inutile.

Cependant, malgré le travail fait pendant la phase précédente et la première partie de cette phase, nous trouvons aussi des binômes qui proposent des algorithmes incorrects (fig. 93) et 94) avec une absence de boucle.

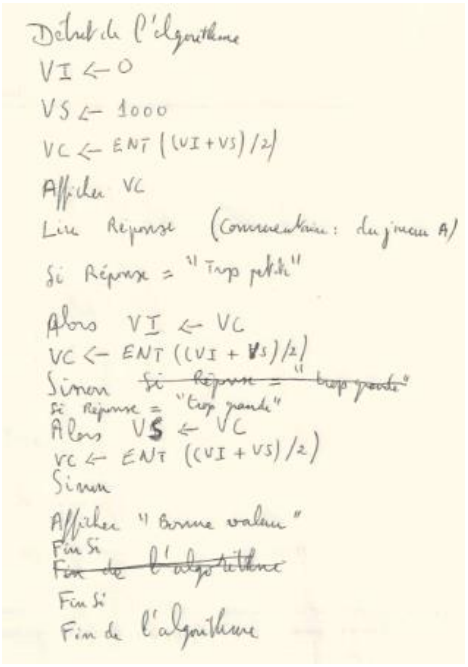


Figure 93 (Un algorithme du joueur B ne mettant pas en place une boucle de type « TantQue » ou « Répéter … Jusqu' »)

Noms : [redacted] et [redacted]

Prénoms : [redacted] et [redacted]

Classe : 2nde 12

Algorithme du joueur B

```
Affecter à Vinf la valeur 0
Affecter à Vsup la valeur 1000
Afficher Vcentre qui prend la valeur de ENT((Vinf+Vsup)/2)
Demander Réponse (du joueur A)
Si Réponse est « trop petit »
Alors
        Affecter à Vinf la valeur Vcentre
        Affecter à Vcentre la valeur de ENT(Vinf+Vsup/2)
Sinon
        Si Réponse est « trop grand »
        Alors
                Affecter à Vsup la valeur Vcentre
                Affecter à Vcentre la valeur de ENT(Vinf+Vsup/2)
        Sinon
                Si Réponse est « bonne valeur »
                Alors
                        Afficher « c'est le nombre entier secret »
                Sinon
                        Si Réponse est « trop petit »
                        Alors
                                Affecter à Vinf la valeur Vcentre
                                Affecter à Vcentre la valeur de ENT((Vinf+Vsup/2)
                        Sinon
                                Si Réponse est « trop grand »
                                Alors
                                        Affecter à Vsup la valeur Vcentre
                                        Affecter à Vcentre la valeur de ENT(Vinf+Vsup/2)
                                Sinon
                                        Si Réponse est « bonne valeur »
                                        Alors
                                                Afficher « c'est le nombre entier secret »
                                        Sinon … (on continu la procédure « si »)
```

Figure 94 (Un algorithme du joueur B du type « arbre de probabilité »)

En effet, dans l'algorithme de gauche (Fig. 93), le binôme ne prend pas en compte qu'il manque une boucle de type « TantQue » ou « Répéter …Jusqu'à » qui permettrait de continuer la procédure de calcul tant que le nombre proposé ne correspondrait pas au nombre entier « secret ». En effet, nous observons que ce binôme ne propose qu'une lecture « verticale », orientée du « nord vers le sud », de la procédure de calcul. Les auteurs de cet algorithme ne semblent pas avoir conscience que la « machine » ne peut proposer une boucle de « continuité » dans la procédure jusqu'à ce que le nombre proposé, noté « Réponse », corresponde au nombre entier « secret » à trouver par le joueur B. On constate ainsi que les structures de type « TantQue » ou « Répéter … Jusqu'à » ne semblent pas avoir automatiquement de sens lors de l'écriture d'un algorithme et qu'elles renvoient plus à une automatisation « intuitive » qu'à une réalité de procédure chez des élèves débutants en programmation.

Dans l'algorithme de droite (Fig. 94), le binôme se lance, dans l'écriture d'un algorithme nécessitant une succession de structures conditionnelles du type « Si A répond : trop petit, alors donner B propose une valeur plus grande Sinon B une valeur plus petite » et ainsi de suite en prenant un format de type « arbre de probabilité ». Cette conception de l'algorithme

est apparue au début de la conception de l'algorithme A (phase 1), mais n'était pas présente dans les productions finales de la phase 1. De plus, nous pouvons observer une erreur « mathématique » dans la prise en compte des règles de priorités opératoires. En effet, ce binôme propose « Ent(Vinf+Vsup/2) », ce qui va introduire une erreur dans le résultat cherché.

d) Les variables

Elles donnent lieu à de nombreuses hésitations et erreurs. Les premières concernent l'identification des variables nécessaires, ce qui donne lieu aux échanges suivants:

L'élève qui avait proposé la « technique de valeur approchée » ré-intervient en demandant à voix haute à l'enseignant : *Combien de mémoires sont-elles nécessaires pour sauvegarder les résultats des nombres entiers obtenus et de la valeur centrale de l'intervalle ?* Un autre élève signale que pour chaque sous-étape du processus, on a trois valeurs à considérer : *borne inférieure, borne supérieur et centre de l'intervalle.* Le professeur envoie alors cet élève au tableau et lui propose de donner des noms à ces trois valeurs. A ce moment-là de la phase, l'enseignant introduit pour la première fois le terme de « variable » pouvant représenter ces trois valeurs. L'élève propose alors d'appeler VI_1 la valeur de la borne inférieure, VS_1 la valeur de la borne supérieure et VC_1 la valeur centrale au départ de la procédure, puis VI_2, VS_2 et VC_2 après la première proposition, VI_3, VS_3 et VC_3 après la troisième proposition, et ainsi de suite jusqu'à arriver au « nombre entier secret ».

L'enseignant rappelle alors qu'en informatique une « variable », n'a pas nécessairement la même « fonction » qu'en mathématique. En effet, celle-ci peut « changer » de valeur (numérique ou littérale) au fur et à mesure de l'évolution de la procédure choisie, mais garder toujours le même nom en donnant comme exemple que : […] *Le cerveau humain enregistre des « croyances » qui peuvent évoluer au cours de la vie dont certaines remplacent des « vielles » croyances désuètes ou fausses, mais que le « cerveau » est toujours le même*. Il complète sa remarque par l'explication suivante : *En mathématique, une « variable » est en général une inconnue, qui recouvre un nombre non précisé de valeurs. Par exemple, quand on pose : $y = 2x - 3$, les couples constitués des « variables » x et y satisfaisant cette expression existent en nombre infini, ils définissent l'ensemble des points qui représente une droite. De même, quand on écrit $a x^2 + b x + c = 0$, où a, b et c sont trois nombres réels fixés, la « variable »*

x désigne cette fois les solutions de cette équation, c'est-à-dire soit zéro, soit une, soit deux valeurs à la fois. Mais en informatique, une variable possède à un moment donné une valeur et une seule. A la limite, elle pourrait ne pas avoir de valeur du tout (une fois qu'elle a été déclarée, et tant qu'on ne l'a pas affectée).

Un autre élève constate qu'à chaque proposition une valeur des bornes reste identique. Un autre élève ayant fini de bâtir un algorithme du joueur B complet, prend alors la parole et propose de garder toujours les mêmes noms pour les trois variables sur le principe de VI (pour la « valeur inférieure »), VS (pour la « valeur supérieure ») et VC (pour la « valeur de la moyenne des bornes VI et VS ») et de partir de l'hypothèse que ces variables sont prises au sens de *stockage*.

Une fois que l'ensemble des élèves de la classe a compris et assimilé les différentes remarques et observations proposées par l'enseignant et au cours des échanges entre élèves, ils écrivent un algorithme du joueur B tenant compte des trois variables. Un élève est alors envoyé au tableau et énonce les alternatives qu'il vient de placer dans son algorithme « papier-crayon » :

> *Si la réponse de la valeur VC proposée par la machine est trop petite alors la machine stocke cette valeur VC dans VI et garde l'ancienne valeur de VS puis calcule une nouvelle valeur VC et recommence le test avec cette nouvelle valeur Sinon Si la réponse de la valeur VC proposée par la machine est trop grande alors la machine stocke cette valeur VC dans VS et garde l'ancienne valeur de VI puis calcule une nouvelle valeur VC et recommence le test avec cette nouvelle valeur Sinon la machine a trouvé la bonne valeur et l'algorithme s'arrête*.

D'autres erreurs concernent la position de l'affectation à la « valeur centrale ». Par exemple, dans l'algorithme ci-dessous (fig. 95), le binôme commet une erreur sur le fait que, bien que les valeurs initiales soient correctement affectées pour les variables d'initialisation, il faut aussi que dans la boucle « TantQue », le résultat du calcul de la moyenne des bornes pour chaque intervalle imbriquée soit à nouveau affecté à une variable dans le corps de la boucle. De plus, sa formulation mélange instruction de sortie et affectation.

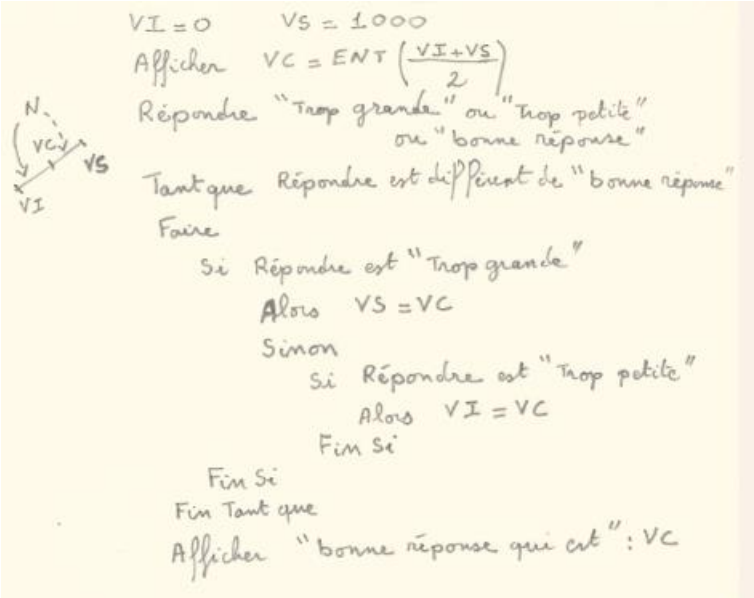


Figure 95 (Un algorithme avec erreur d'affectation dans le corps de boucle)

Après que les différents algorithmes « papier-crayon » du joueur B aient été relevés par l'enseignant, celui-ci indique aux élèves qu'une telle stratégie basée sur de calcul d'une moyenne des bornes de l'intervalle est nommée : […] *méthode de dichotomie. On utilise cette méthode en analyse pour la recherche d'un zéro d'une fonction en répétant des partages successifs d'un intervalle en deux parties, puis on sélectionne le sous-intervalle dans lequel existe un zéro de la fonction*.

4.3.1.2. En Première Scientifique

Dès le début de cette phase, les binômes de la classe (excepté deux binômes comme nous le verrons plus loin) ont bien intégré le fait que l'action du joueur B va être associée à la *dichotomie*. En effet, les élèves ayant pratiqué le jeu au début de la phase 1, ont rapidement pris conscience que la dichotomie pouvait être une stratégie « gagnante » et « rapide » pour le joueur B. L'enseignant, responsable de la classe, constatant que cet aspect de la phase 2 comme étant déjà résolu, prend alors l'initiative de donner comme consigne de limiter l'algorithme du joueur B à un nombre donné d'essais et d'introduire une variable de type « compteur », afin de valoriser la « stratégie de dichotomie ». Il précise aussi aux élèves que cette variable « compteur » a pour unique fonction de vérifier le nombre de coups utilisés pour trouver le nombre entier « secret » avec la mise en place d'une « stratégie de dichotomie ». Ces deux remarques faites à l'initiative de l'enseignant ont pu en partie parasiter le déroulement de cette phase dans cette classe.

Comme nous le disions au début de cette section, excepté pour les deux binômes sur les

quinze observés, l'ensemble des élèves se lance tout de suite dans l'élaboration d'algorithmes du joueur B avec une stratégie basée sur la dichotomie et une utilisation correcte de la partie entière. Il est à noter aussi qu'à l'exception d'un binôme, ils proposent tous des algorithmes « textuels » sur le principe décrit précédemment en Seconde. Le binôme qui propose un algorithme du joueur B complet sous la forme d'un organigramme, y ajoute un compteur donnant le nombre de coups utilisés pour déterminer le « nombre entier secret » (cf. fig. 96). Après avoir questionné ce binôme, ainsi que les binômes ayant écrit des algorithmes « textuels » avec un compteur (cf. fig. 97), nous observons que cet ajout est une conséquence de la consigne de l'enseignant précisée ci-dessus.

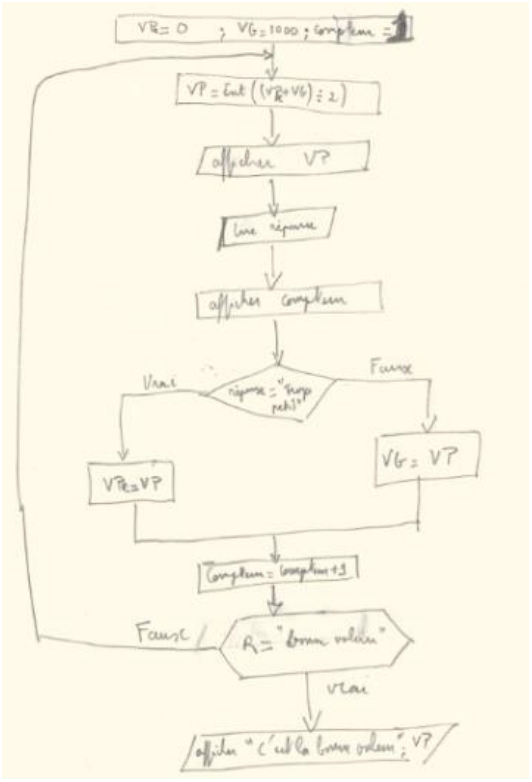


Figure 96 (Algorithme « spatial » avec compteur et stratégie « dichotomie »)

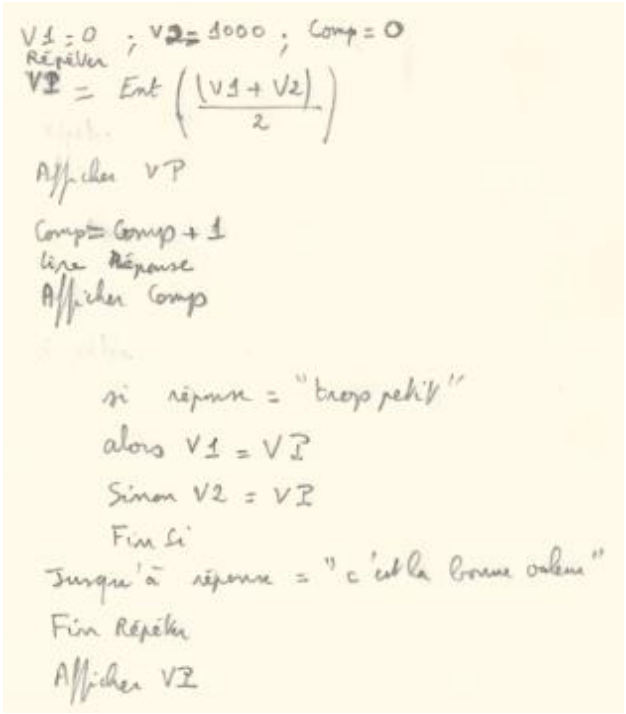


Figure 97 (Algorithme « textuel » du jouer B avec compteur et stratégie dichotomie)

Les élèves choisissant l'organigramme comme représentation de l'algorithme du joueur B respectent les contraires graphiques d'une telle représentation. Ils utilisent une structure de type « Répéter … Jusqu'à » et ont une bonne pratique des variables informatiques.

Les élèves choisissant une représentation textuelle de l'algorithme du joueur B utilisent aussi une structure de type « Répéter … Jusqu'à ». Ils ont une bonne pratique des variables informatiques.

Les deux binômes, qui ne prennent pas l'initiative d'utiliser dès le début de cette seconde phase une méthode en lien avec la dichotomie, proposent une méthode correcte qui consiste à ce que la machine renvoie un nombre entier aléatoire dans l'intervalle connu au lieu de la partie entière de la moyenne des bornes (Fig. 98). De plus, pour un de ces deux binômes, la remarque de l'enseignant sur l'utilisation d'un « compteur » est aussi prise en compte (Fig. 99). Les algorithmes qu'ils donnent sont écrits sous un format « textuel ». Ils utilisent une boucle « TantQue » et l'organisation des variables est aussi faite correctement. De plus, le fait d'utiliser un entier aléatoire compris entre 0 et 1 000 comme nombre proposé par le joueur B affectée à la variable « Valeur proposée » (ou « VP ») permet de faire que les entiers 0 et 1 000 puissent être « vérifiés » dès « le début de la procédure », contrairement à ce qui a été observé avec les algorithmes proposés par les élèves sur la stratégie avec la dichotomie.

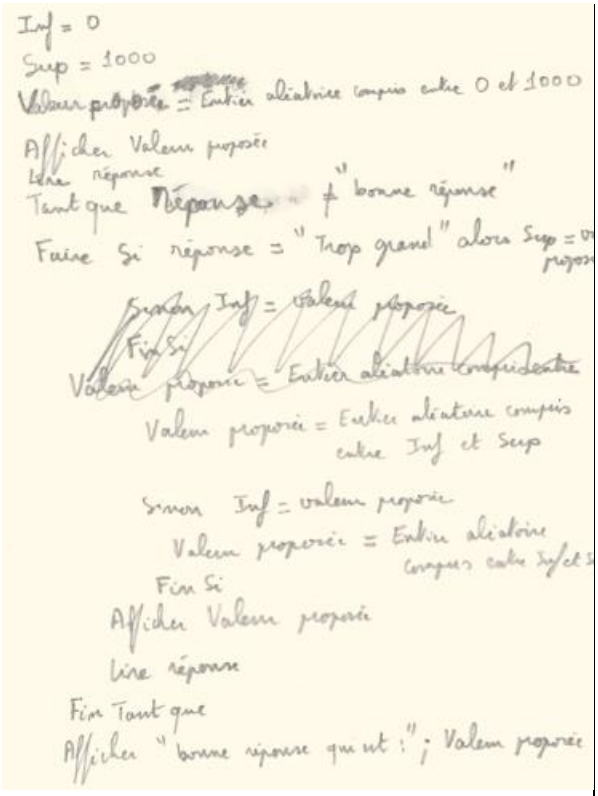


Figure 98 Algorithme « textuel » sans stratégie « dichotomie » et sans compteur

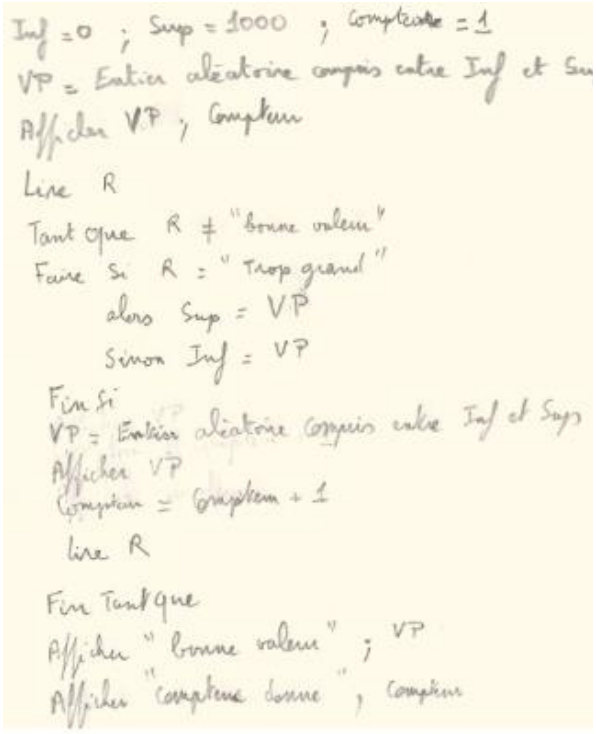


Figure 99 Algorithme « textuel » sans stratégie « dichotomie » avec compteur

Au cours du dernier quart d'heure de cette phase, par souci de gestion de temps, l'enseignant décide d'intervenir auprès des deux binômes qui s'étaient tournés vers une stratégie « aléatoire », afin de les orienter vers une stratégie « rapide » basée sur la

dichotomie.[76] Après cette intervention de l'enseignant, ces deux binômes proposèrent alors deux nouveaux algorithmes où l'utilisation de la fonction « partie entière » se fit sans problème comme pour les autres binômes.

4.3.1.3. En Terminale Scientifique

Pour cette classe, l'enseignant et le chercheur sont une seule et même personne. Une stratégie « gagnante » et « rapide » est tout de suite mise en lien avec la *dichotomie* par les élèves. L'utilisation de la fonction *partie entière* se fait aussi de façon « naturelle ». Familiarisés à diverses écritures d'algorithmes tant « textuelle » que « spatiale » depuis le début de l'année scolaire, les élèves proposent dès le début de cette deuxième phase diverses formulations correctes d'algorithmes du joueur B. Une proportion plus importante utilise une boucle « TantQue » (80% des binômes) plutôt qu'une boucle « Répéter …Jusqu'à » (20% des binômes). Ayant nous-même conduit la classe, nous avons observé que les élèves font référence à la structure de l'algorithme du joueur A construit à la phase 1. Par exemple, un élève fait remarquer qu'il a utilisé la même structure qu'à la phase 1, mais qu'il a dû introduire des variables nouvelles.

Certains binômes proposent d'eux-mêmes un compteur afin de dénombrer le nombre de coups nécessaires pour déterminer le nombre entier « secret » et de déterminer sur plusieurs essais qu'elle serait le nombre moyen de ces coups nécessaires pour déterminer ce nombre entier « secret ».

4.3.2 Analyse *a posteriori*

Cette seconde phase, quel que soit le niveau scolaire, dure beaucoup plus longtemps que ce qui est prévu lors de l'analyse *a priori*. Ceci vient en partie du fait que les enseignants, avec l'accord du chercheur, ont souhaité laissé le plus d'initiatives aux élèves. Et pour cela, ils ont encouragé des échanges oraux entre les différents binômes.

[76] Cette stratégie « aléatoire » est réalisable de façon automatique, elle est effective à condition que le choix soit fait dans l'intervalle strict et son efficacité « en moyenne » est la même que celle de la dichotomie. La dichotomie est plus efficace « au pire » (un maximum de 7 essais), alors que la stratégie « aléatoire » peut conduire à 99 essais. Nous ne souhaitons pas engager les élèves dans ces considérations, d'autant plus que dans le cas continu, la justification de l'effectivité met en jeu la convergence de suites de variables aléatoires et est donc hors de portée des élèves.

4.3.2.1. En Seconde

Nous analysons les difficultés rencontrées en référence à l'analyse *a priori* et au déroulement.

a) Le choix d'une stratégie

La dichotomie ne semble pas venir de façon « naturelle » à l'esprit des élèves de Seconde au moment où a lieu cette expérimentation. L'idée d'une suite d'intervalle et d'un nombre à proposer dans l'intervalle courant découle pour eux de la familiarisation avec le jeu à la phase 1. Mais ils ne conçoivent pas une règle systématique pour le choix d'un nombre à proposer dans le dernier intervalle. Curieusement un binôme propose de « couper en deux » l'intervalle, mais ensuite de choisir au hasard un des deux sous intervalles et un nombre dans ce sous-intervalle. Le choix du « milieu » ou de la « frontière » est long à venir. Les élèves n'ont pas une claire conscience des fonctionnalités supposées du dispositif pour lequel l'algorithme est écrit.

b) La partie entière

Conformément à ce qui est prévu au cours de l'analyse *a priori*, le problème d'une valeur entière lors d'un calcul de moyenne des bornes de l'intervalle, a pu être une réelle difficulté chez un certain nombre d'élèves. Nous observons particulièrement que la nécessité de valeurs entières peut rester implicite. En effet, deux binômes proposent un algorithme (l'un « spatial », l'autre « textuel ») complet du joueur B, en faisant l'hypothèse implicite (au sens de « Théorème en acte[77] ») que l'écriture $\frac{a+b}{2}$ donne automatiquement un nombre entier (Fig. 90), et ainsi cette « étape » de calcul n'est pas prise en compte dans l'algorithme qu'il propose.

Les autres élèves ont conscience de ce que cette moyenne peut ne pas être entière et cherchent des stratégies de compensation. Ce qui pose problème, c'est l'expression du calcul de la partie entière d'un nombre donné dans le langage algorithmique. L'analyse *a priori* avait prévu que ce point ne serait pas critique du fait de l'absence de contraintes au format « papier-crayon ». Cela n'a pas été le cas et au bout de 25 minutes l'intervention de l'enseignant a été nécessaire. Ceci rejoint des observations concernant la conception des

[77]Un *théorème en acte* est une règle d'action utilisée par les élèves et compatible avec la conception qu'ils se font d'une connaissance. Il est vérifié dans un certain domaine seulement.

fonctions par les élèves au début du lycée : un traitement (processus) n'est pas nécessairement compris comme établissant une dépendance fonctionnelle (objet) exprimable à l'aide du formalisme fonctionnel.

Quant au concept de « fonction » qui apparait avec la notion de « partie entière », l'élève le voit dans cette ingénierie comme « objet » modélisant une dépendance entre variables en décrivant une correspondance terme à terme entre les valeurs prises par ces variables.

c) La structure

Quelques élèves se dirigent initialement vers la construction d'une série d'alternatives, mais passent rapidement à une itération. Nous pensons que les élèves ont tiré parti du travail réalisé pour la construction de l'algorithme du joueur A lors de la phase 1, bien que nous n'ayons pas d'indice l'attestant directement.

d) Les variables

Comme le prévoit l'analyse *a priori*, la mise en œuvre de variables de boucles, nouvelle dans cette phase, fait l'objet de difficultés et de nombreux échanges entre les élèves. Ainsi, d'après Rogalski et Samurçay (1990), l'utilisation de structures itératives nécessite l'identification et la construction d'invariant de boucle (mise à jour), d'une condition de continuation (test) et d'une initialisation de variables. L'utilisation de « variables » et leurs « affectations » à l'initialisation et dans le corps de boucle sont nouveaux pour des élèves débutants en informatique. Dans un algorithme « papier-crayon », les nombres en jeu peuvent être identifiés par leur fonction (par exemple « le nombre choisi ») plutôt que par leur valeur. Cependant, ceci ne se traduit pas nécessairement directement par l'emploi de variables itératives des valeurs différentes à différents moments du traitement.

Ici, une majorité d'élèves ne prend pas conscience spontanément des variables qui vont être nécessaires pour mémoriser la suite de nombres entiers. La situation se débloque par l'intervention d'un élève qui met l'accent sur le fonctionnement de la machine. Cependant, la majorité des élèves conçoit cette mémorisation sous la forme d'une suite de variables indexées, plutôt que par des variables itératives. Ceci rejoint l'observation de Rogalski et Samurçay (1990) selon laquelle certains élèves ne conçoivent pas des variables « internes » à l'algorithme, c'est-à-dire ne correspondant pas à des objets donnés ou ont tendance à vouloir

utiliser des variables différentes à chaque passage dans la boucle. Ceci conduit l'enseignant à un long discours sur les spécificités de variables informatiques.

En se référant aux travaux de Lagrange (1992), on comprend qu'il s'agit en fait, de représentations installées en mathématiques sur la notion de « variable ». En effet, les expérimentations de Lagrange (Ibid.), comme coder la présence ou l'absence de personne lors d'un diner suite à une invitation, va dans le même sens que ce que nous observons lors de notre ingénierie, à savoir que les élèves débutants ont de réelles difficultés à assimiler l'affectation de « variables ».

4.3.2.2. En Première et Terminale Scientifiques

La majorité des élèves du cycle terminal semblent avoir une bonne habitude d'une stratégie « dichotomique » même si les applications de la méthode de dichotomie ont été essentiellement vues en Seconde dans des cas de « dichotomie continue » en lien avec la détermination de valeurs approchées d'une solution d'une équation à une inconnue. Seuls deux binômes se sont lancés dans une autre stratégie de type « aléatoire », bien que qu'ils aient connaissance de la *méthode de dichotomie*. Mais une brève intervention se référant à la « dichotomie » de la part de l'enseignant, aura suffi à les faire changer de stratégie. On peut supposer que le transfert du « continu » au « discret » n'a pas été fait chez les élèves de ces binômes. La prise en compte de ce que la moyenne des bornes de l'intervalle peut être non entière, n'a pas été un obstacle chez les élèves à ce niveau scolaire, même si l'étude de la fonction « partie entière » et l'utilisation du formalisme fonctionnel pour désigner cette fonction ne sont plus au programme du cycle terminal.

Nous avons pu observer diverses expressions des algorithmes par les élèves, toutes correctes. Certaines sont de type « textuel », bien qu'il fût explicitement demandé aux élèves des organigrammes. Les organigrammes comme les textes relèvent aussi bien de la structure « TantQue » que de la structure « Répéter … Jusqu'à ». La structure « TantQue » a été favorisée dans les travaux antérieurs des élèves en algorithmique, mais elle est peu élégante ici, obligeant à exprimer deux fois le calcul d'une moyenne d'abord à l'initialisation et ensuite dans la boucle.

4.3.2.3. Synthèse

En référence à l'analyse *a priori*, les difficultés prévues sont apparues quasi-exclusivement en Seconde. Dans cette classe, le *principe de dichotomie*, et la nécessité de considérer un arrondi ont été longs à émerger. Contrairement à ce qui a été prévu, il a été nécessaire de travailler sur l'expression de cet arrondi, les élèves ne comprenant pas le type de formalisation souhaité en « papier-crayon ».

Comme prévu, la conception d'une structure adaptée n'est pas un réel obstacle, même si il y a encore des hésitations en Seconde. Nous pensons que l'écriture de l'algorithme du joueur B s'appuie sur la structure construite pour le joueur A (cf. phase 1) et nous avons observé cela nous-même en Terminale. Les élèves de Seconde ont rencontré des difficultés pour la mise en œuvre des variables de boucle que nous avions prévues lors de l'analyse *a priori*. Cela n'a pas été le cas des élèves de Première et Terminale. Nous interprétons cela comme la conséquence d'une pratique régulière de l'algorithmique par les élèves du cycle terminal depuis la Seconde, et aussi en Première pour les élèves de Terminale. De plus, il s'agit d'élèves d'une filière scientifique. Une meilleure réussite dans la conception d'algorithmes a souvent été constatée chez les élèves des filières scientifiques, par comparaison avec d'autres filières où les difficultés tendent à persister. Comme le signalent Lagrange & Rogalski (2016), la question reste ouverte de précurseurs à l'activité de programmation dans les apprentissages en mathématiques, ou de précurseurs communs tels que des dispositions au raisonnement logique ou à la conception de l'espace.

Nous avons aussi observé qu'au cours de cette phase, quelques binômes ont éprouvé des difficultés à concevoir une condition d'arrêt dans la boucle « Répéter… Jusqu'à » car celle-ci s'exprime par une égalité. Ainsi, certains élèves ont pu dire : *Monsieur, il est plus naturel de procéder à une boucle faisant intervenir une « inégalité », car lors de la pratique du jeu, le joueur B continue à faire des propositions de nombres entiers tant que ces nombres sont différents de l'entier secret*. Voulant néanmoins utiliser une structure « Répéter… Jusqu'à », ces élèves sont confrontés à l'expression d'une condition d'arrêt. Ils font alors référence au cours de probabilités récent où ils ont appris à calculer la probabilité d'un événement E à l'aide de la probabilité de son événement contraire $\overline{\mathrm{E}}$. Nous observerons cela de nouveau dans la phase 3 (cf. 3.4.1.1)

4.4 Phase 3 (algorithme de joueur B – formalisation sous formes « textuelle » et « spatiale » – implémentation dans un environnement numérique de l'algorithme. Durée : 30 minutes en classe qui se termine à la maison pour la classe de Seconde – 40 minutes en Première et Terminale Scientifique)

4.4.1 Le déroulement

Dans cette phase, suivant l'environnement informatique choisi par le binôme, les élèves procèdent à : (1) une reformulation de l'algorithme avec un changement de structure de boucle si cela s'avère nécessaire ; (2) une utilisation de la *fonction partie entière* pour chaque calcul d'une moyenne ; (3) une gestion et une organisation des différentes variables ; (4) une prise en compte des contraintes du langage syntaxique pour l'implémentation de leur algorithme « papier-crayon » ; (5) une implémentation de l'algorithme sur la machine ; (6) une exécution de l'algorithme sur plusieurs parties de jeu afin de vérifier : – la validité de leur algorithme sur le plan syntaxique ; – le fait que l'algorithme s'exécute avec des nombres entiers pour le calcul des différentes valeurs des moyennes et stocke bien des valeurs entières pour ces moyennes ; – la pertinence du choix de la stratégie « rapide » définie au cours de la phase précédente.

4.4.1.1. L'environnement informatique et les choix des structures

Pour cette dernière phase de la sous-ingénierie sur la « dichotomie discrète », quelle que soit la classe, les binômes utilisent leurs ordinateurs, ainsi que les résultats de leurs travaux « papier-crayon » faits au cours des phases 1 et 2. Tous les ordinateurs sont équipés de deux environnements de programmation : *LARP* pour la conception et l'exécution d'algorithmes aux formats « organigramme » ou « pseudo-code » et *AlgoBox* pour la conception et l'exécution d'algorithmes au format « pseudo-code ».

Comme pour les deux premières phases, les élèves ont toujours leur polycopié sur les règles de construction d'un organigramme. A la demande de certains élèves, ils peuvent aussi garder leur calculatrice et utiliser le menu « PRGM » de celle-ci.

Une très grande majorité des productions des binômes à la fin de cette séance est constituée d'algorithmes construits et exécutés dans l'environnement *AlgoBox*. Cependant, l'ensemble des binômes, quel que soit la classe, a eu recours au moins de façon ponctuelle à l'environnement *LARP* pour obtenir une formalisation correcte de l'organigramme associé à

leur algorithme. En effet, cet environnement leur permet de construire, et éventuellement d'exécuter un algorithme écrit sous forme « spatiale », c'est-à-dire un organigramme. Cependant, les enseignants qui ont mis en œuvre cette ingénierie ont souhaité favoriser le choix du logiciel *AlgoBox* pour rester dans un cadre plus institutionnel. Même si officiellement aucun langage et aucun environnement informatique ne sont imposés par l'institution. Lors de l'entretien, les trois enseignants signalent en effet, que les manuels scolaires utilisés dans leur classe respective ne présentent pas de tâches sur les algorithmes mettant en jeu l'environnement *LARP*, et que celui-ci n'est pas présent dans les parties annexes des manuels concernant diverses descriptions d'environnements algorithmiques, et que, en revanche, de nombreux algorithmes sont écrits avec l'environnement *AlgoBox*. Ce fait est aussi visible dans la plupart des documents ressources institutionnels (cf. Partie 1).

En revanche, les élèves de Seconde n'ayant eu aucune préparation particulière sur les deux environnements retenus pour cette sous-ingénierie choisissent majoritairement l'environnement *LARP*. En effet, certains des binômes signalent que cet environnement leur permet de transcrire un algorithme représenté sous forme d'un organigramme en un algorithme écrit en « pseudo-code ». Lors de l'entretien qui suivra l'ingénierie, les élèves diront que cet atout de *LARP* a favorisé leur choix. Cependant, certains élèves, majoritairement des redoublants, préfèrent travailler avec l'environnement *AlgoBox*, qu'ils ont mis en pratique lors de leur première Seconde.

Malgré le fait que les élèves du cycle terminal scientifique des lycées aient une pratique « ancienne » de l'environnement *AlgoBox*, certains binômes travaillent seulement dans l'environnement *LARP*. Le reste des binômes travaille dans les deux environnements. Pour ces binômes, l'environnement *LARP* est utilisé comme tuteur pour la conception d'un organigramme vérifiant des règles de constructions graphiques précises.

Les binômes, qui utilisent un environnement informatique n'autorisant que des programmations au format « textuel » (AlgoBox ou calculatrice), se voient dans l'obligation d'adopter une structure « TantQue », tandis que ceux qui utilisent un environnement permettant des programmations de types « textuel » ou « spatial » (*LARP*) peuvent choisir cette structure ou « Répéter… jusqu'à ».

Comme pour la phase 2, nous constatons que quelques binômes éprouvent encore des

difficultés à concevoir une condition d'arrêt dans la boucle « Répéter… Jusqu'à ». En effet, le fait que cette condition d'arrêt s'exprime par une égalité leur semble être « paradoxale », car pour eux il est plus naturel de procéder à une boucle faisant intervenir une « inégalité » qui reflèterait mieux la pratique « manuelle » du jeu où, le joueur B continue à faire des propositions de nombres entiers tant que ces nombres sont différents de l'entier « secret » à déterminer. Voulant néanmoins utiliser une telle structure, ces élèves restent dans une transposition « erronée » du cours de probabilités où ils ont appris à calculer la probabilité d'un événement à l'aide de la probabilité de son événement contraire.

Mais finalement, nous constatons que le choix de l'environnement *AlgoBox* détermine chez l'élève la décision de prendre une structure « TantQue ». En effet, certains de ces élèves qui ont utilisé *AlgoBox*, on dit : *Mais, quoi qu'il en soit Monsieur, AlgoBox ne propose que deux structures de boucle : « TantQue » et « Pour… Faire ». Or pour la seconde, il serait nécessaire de connaitre le nombre d'itérations à faire avant de commencer le jeu ce qui n'est pas le cas, car vous ne nous proposez pas un nombre précis d'essais possibles dans le jeu. Donc nous pouvons pas*[78] *garder cette solution et par conséquent avec cet environnement, nous n'avons pas le choix !*

4.4.1.2. Les problèmes d'ordre syntaxique

Bien que cette phase n'ait pu être totalement terminée en classe par les élèves de Seconde, nous ne différencierons pas les niveaux Seconde et Première, car nous avons pu observer très peu de différence entre ces niveaux au cours de cette phase. Les difficultés constatées pour ces deux niveaux sont de quatre ordres : (a) pour ceux qui rendent un travail sur *AlgoBox* et qui utilisent une autre structure lors des phases précédentes, ils modifient leur algorithme afin d'avoir une structure « TantQue » écrite dans un langage d'algorithme qui respecte la syntaxe de l'environnement *AlgoBox* ; (b) pour ceux qui ont seulement dessiné un dessin « approximatif » représentant un organigramme, ils doivent reconsidérer la structure produite lors de la phase 2 afin que cette structure soit reconnue par l'environnement choisi ; (3) pour certains binômes, l'utilisation de la *fonction partie entière* n'est pas systématiquement prise en compte lors de l'implémentation dans l'environnement informatique et par conséquent, une fois qu'ils ont pris conscience de ce fait, ils cherchent la

[78] Dans l'enregistrement, nous n'attendons pas la forme correct « ne… pas ».

fonctionnalité de l'environnement qui permet de résoudre ce problème ; (4) tous les binômes doivent surmonter des problèmes en lien avec la syntaxe du langage de l'environnement informatique choisi afin de pouvoir implanter leur algorithme « papier-crayon ». Le cas échéant ceci peut nécessiter de tester l'algorithme à l'aide de l'environnement et d'interpréter correctement les messages d'erreur affichés par l'ordinateur.

Pour le niveau Terminale, nous n'observons des difficultés que pour ceux qui travaillent totalement avec *LARP*, relatives à l'appropriation d'un environnement nouveau.

4.4.1.3. La mise en place des variables

Contrairement à ce à quoi nous nous attendions lors de l'analyse *a priori*, l'exécution ne met pas en évidence des erreurs dans l'utilisation des variables de type nombre : initialisation de Binf (pour la borne inférieure) et Bsup (pour la borne supérieure), puis l'affectation à NbCh correspondant à la valeur moyenne des deux bornes. En revanche, nous observons des difficultés chez certains binômes en particulier en Seconde, résultant de ce que l'environnement *AlgoBox* nécessite de préciser le type de la variable lors de sa déclaration. En effet, il faut choisir parmi trois types : nombre, chaîne au sens de « chaîne de caractère » et liste au sens de « matrice-ligne ».

Ainsi, dans la classe de Seconde, nous avons deux algorithmes (fig. 100 et 101) écrits dans l'environnement *AlgoBox* avec une structure correcte et identique mais qui, cependant, ne fonctionnent pas lors de leurs exécutions. En effet, dans l'algorithme de la figure 100, le binôme ne prend pas en compte que toutes les variables doivent être déclarées en précisant leur type, et dans celui de la figure 102, le binôme ne considère pas qu'une affectation à une variable de type « chaîne de caractère » nécessite de mettre des guillemets à la chaîne concernée. Lors du codage de l'algorithme comme nous le voyons dans les deux cas aucun message d'erreur n'est indiqué par l'environnement *AlgoBox*, ce n'est que lorsque les utilisateurs testent leur algorithme qu'ils peuvent prendre conscience d'une erreur de syntaxe sans pour autant comprendre sa cause ou sa signification. En effet, lors de l'exécution de l'algorithme, *AlgoBox* affiche alors comme message : « ***Algorithme interrompu ligne 11 suite à une erreur dans son exécution*** » (fig. 101) pour l'algorithme où toutes la variable de type chaîne n'est pas déclarée, et « ***Algorithme interrompu ligne 12 suite à une erreur dans son exécution*** » (fig. 103) pour l'algorithme où toutes les variables ont bien été

déclarées. Cependant, dans la condition de boucle le fait que la variable « Réponse » corresponde à une chaîne de caractère nécessite des guillemets sur « Bonne_valeur ». Ainsi ce message n'est pas suffisamment explicite pour que l'élève débutant en programmation puisse en comprendre son sens. Ceci nécessite alors une intervention de l'enseignant.

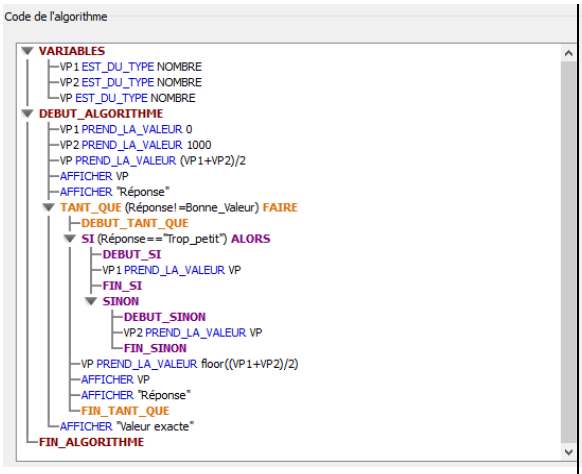


Figure 100 (Algorithme sous AlgoBox avec structure correcte et identique, mais qui ne fonctionne pas lors de son éxécution)

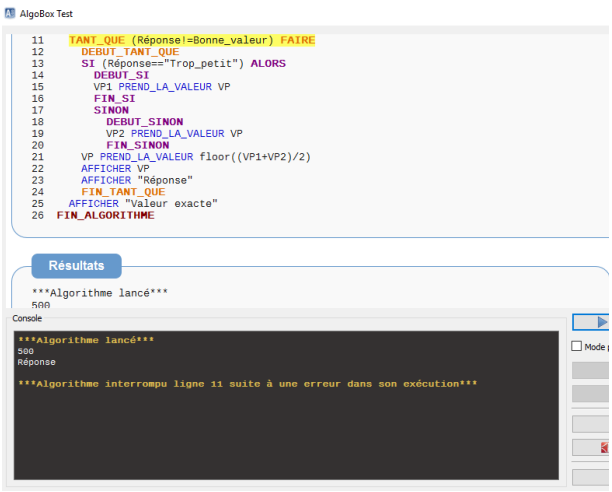


*Figure 101 (*Algorithme sous AlgoBox avec structure correcte et identique, mais qui ne fonctionne pas lors de son éxécution)

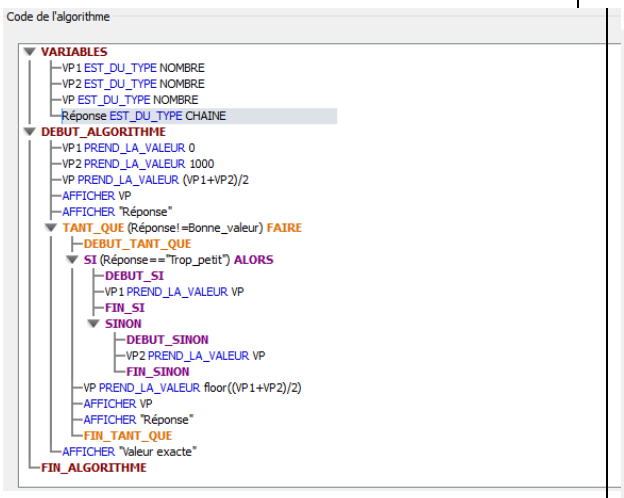


Figure 102 (Algorithme sous AlgoBox qui ne fonctionne pas lors de son éxécution : erreur sur la chaîne de caractère)

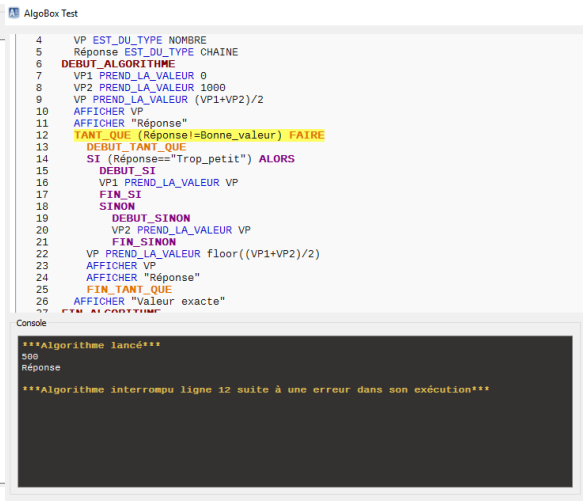


Figure 103 (Algorithme sous AlgoBox qui ne fonctionne pas lors de son éxécution)

Un troisième binôme de Seconde propose aussi un algorithme ayant une structure correcte mais montrant une erreur sur le type de la variable « Proposition » qui est définie comme étant une variable de type « nombre » (Fig. 104). Lors de l'exécution l'algorithme va tourner « indéfiniment » en boucle sans que les élèves ne puissent comprendre d'où vient leur erreur, d'autant plus qu'*AlgoBox* n'affiche pas un message d'erreur mais le fait que l'affichage est saturé par trop de données : « ***Affichage interrompu ligne 25 : affichage trop important de données*** » (Fig. 105).

Figure 104 (Algorithme sous AlgoBox qui ne fonctionne pas lors de son éxécution : erreur sur le type d'une variable)

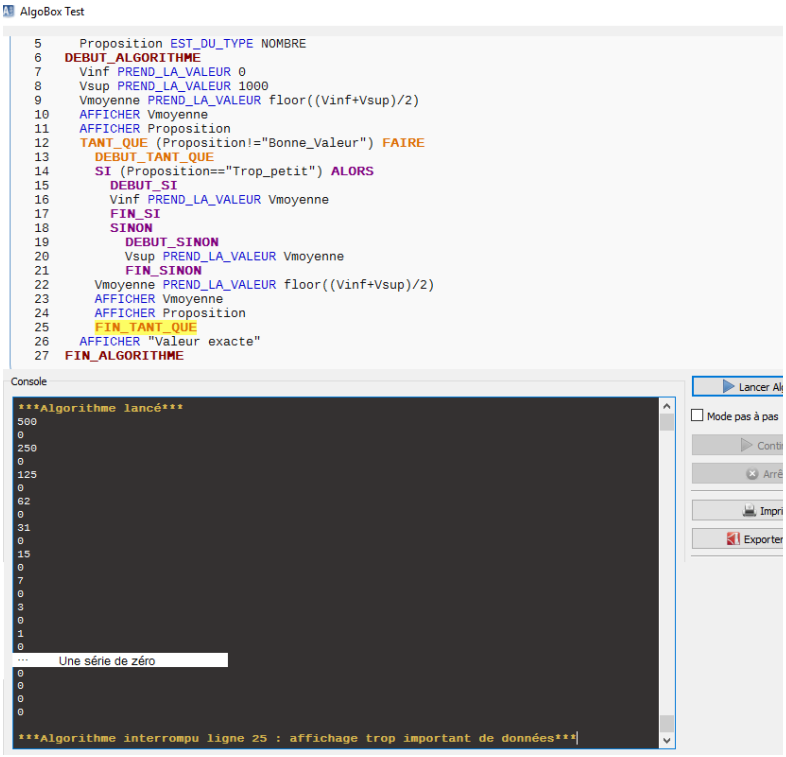


Figure 105 (Algorithme sous AlgoBox qui ne fonctionne pas lors de son éxécution : erreur sur le type d'une variable)

Ainsi, lors de cette phase, un certain nombre de binômes de la classe de Seconde éprouve des difficultés sur le type de variables.

La transcription suivant d'un enregistrement audio du binôme (Elève1 et Elève2) ayant proposé l'algorithme précédent (fig. 104 et 105) témoigne d'une incompréhension du langage informatique et des variables de la part de certains élèves débutants en programmation :

> Elève1[79] : *Je comprends pas, nous avons tout mis comme ce que nous avions trouvé pendant le travail sur le papier. Mais l'algorithme, il veut pas s'arrêter. Je crois qu'il faut que nous appelions le prof.*
> Elève2 : *Reviens sur la page « codage ».* [Temps de silence]*. On a déclaré toutes les variables, on a respecté la partie entière* [Temps de silence]*. Même la boucle est comme celle que nous avions utilisée tout à l'heure et le prof avait dit que c'était bon. Je ne comprends pas ! Appelle le prof.*
> [Commentaire : l'enseignant n'ayant pas vu la demande des élèves, ces derniers essayent à nouveau de « retester » leur programme, puis ils reviennent à l'écran « code de l'algorithme »]
> Elève2 : *M… regarde, il y a différentes possibilités pour les variables : « Nombre, Chaîne et Liste ». Ce doit être là que l'on doit se tromper. L'an dernier en techno, avec Excel on avait appris qu'écrire « 2 + 1 » ce n'était pas la même chose que « =2 + 1 ». Ça doit un problème comme ça ici.* [Temps de silence]
> Elève1 : *Peut-être… Que fait le prof ? Est-ce qu'on essaie avec l'autre logiciel ? Regarde E. et P.* [Commentaire : les élèves du binôme voisin]*, ils n'ont pas ce problème...*

[79] Les erreurs sur les formes négatives commises par l'élève ont été reproduites telles quelles.

La sonnerie annonçant la fin de la séance s'enclenche à cet instant et fais que nous ne savons pas ce qui aurait pu se passer si ces deux élèves avaient pu travailler dans l'environnement *LARP*, d'autant plus qu'ils ne poursuivirent pas leur travail à la maison.

Cependant, nous pouvons observer que l'un de ces deux élèves semble prendre conscience *du caractère calculable des objets du langage correspondant à ce type* de variables (Lagrange & Rogalski, 2015).

De même des erreurs sur la conception et la lecture de variables de type « chaîne de caractère » par les élèves sur des organigrammes (Fig. 106) ont pu être observées, malgré le fait que cet environnement ne demande pas de déclarer au début de l'algorithme le type des variables utilisées.

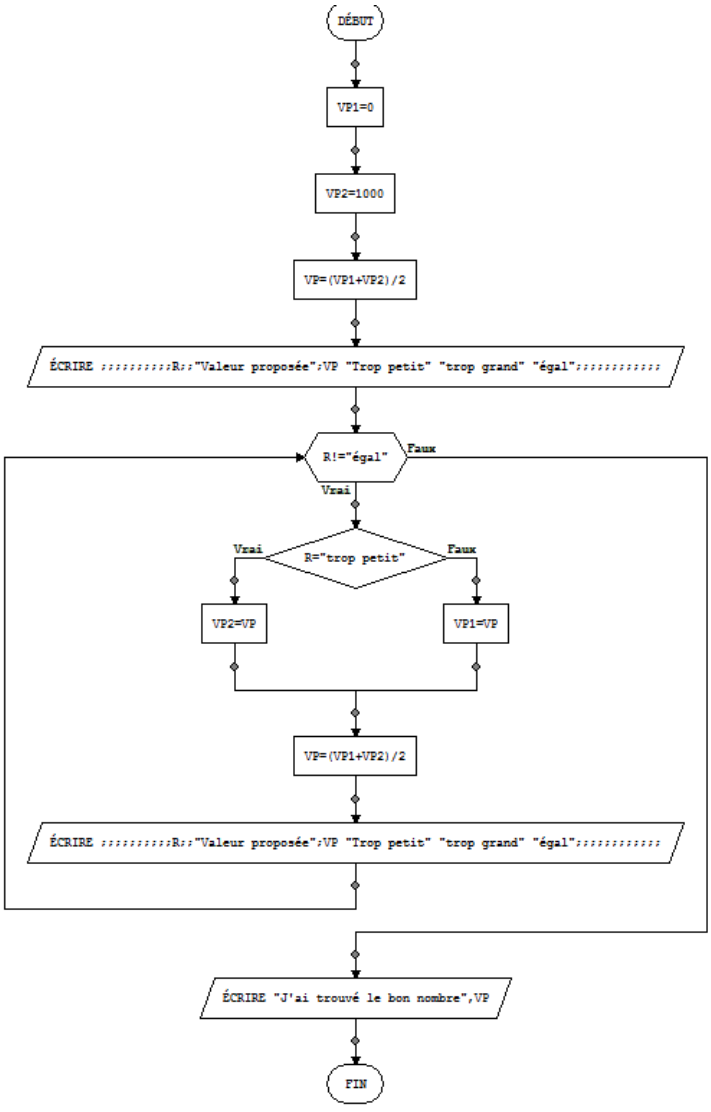


Figure 106 (Cas d'un organigramme comportant des erreurs sur la conception et la lecture de variables de type « chaîne de caractère »)

Nous observons que pour surmonter cette difficulté apparue sur les variables de type

« chaine de caractère », certains binômes proposent des organigrammes où les réponses proposées par le joueur B sont « – 1 » pour « trop petit », « 1 » pour « trop grand » et 0 pour « égal ». Ces binômes choisissent cette solution pour surmonter le problème de lecture des « chaines de caractères » (Fig. 107).

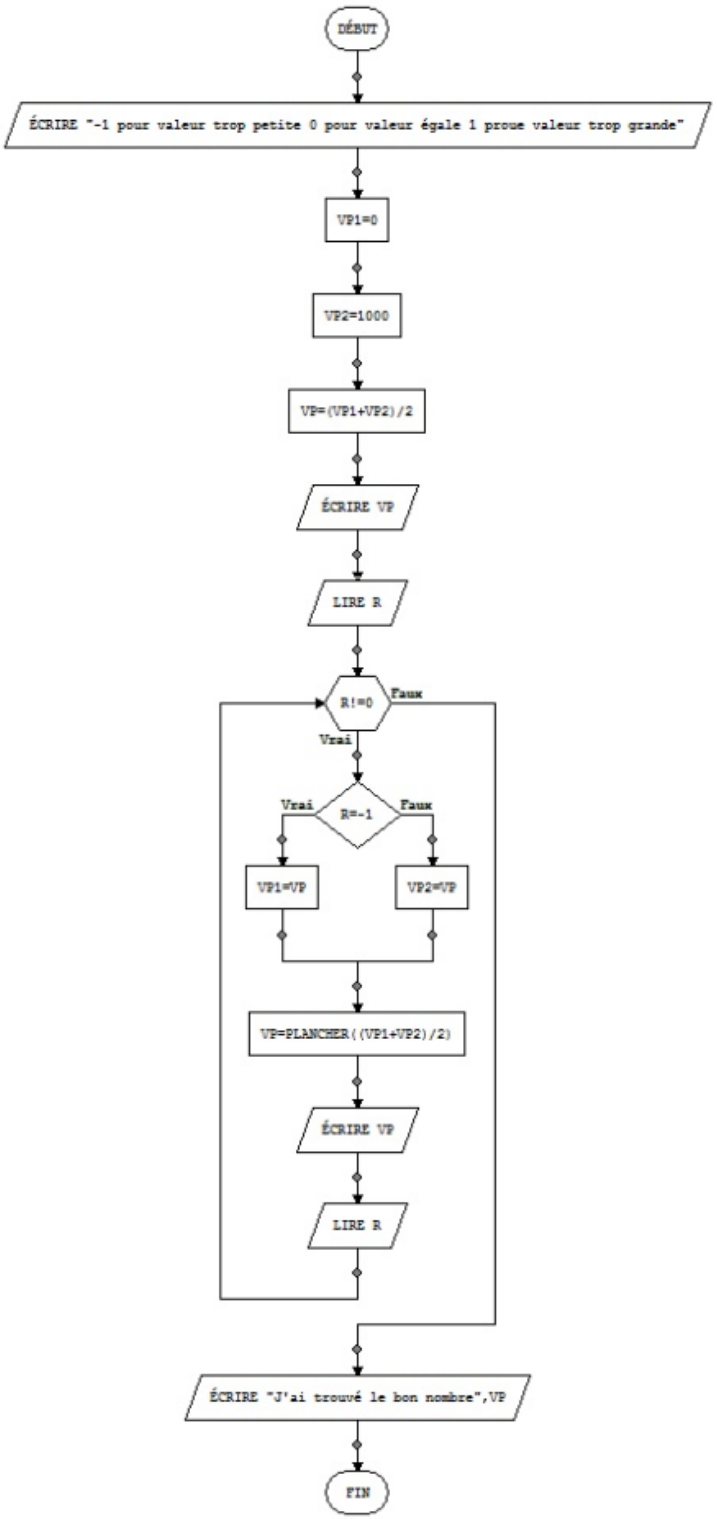
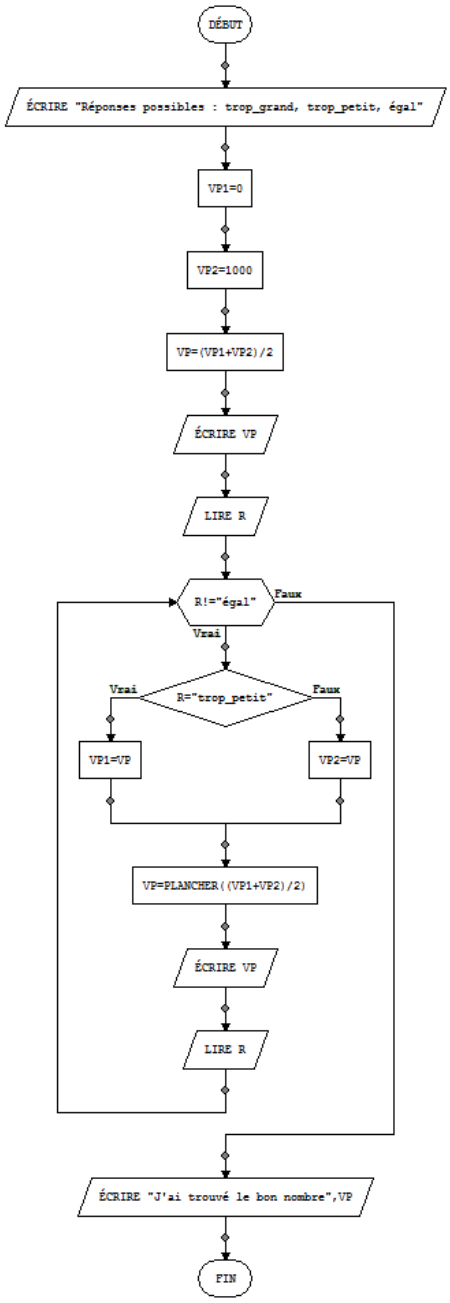


Figure 107 (Organigramme où les réponses proposées par le joueur B sont « – 1 » pour « trop petit », « 1 » pour « trop grand » et 0 pour « égal »)

D'autres suggèrent de mettre un commentaire pour décrire les « chaines de caractères » pouvant servir de réponses à l'ordinateur par l'utilisateur jouant le rôle du joueur B (fig. 108).



Les élèves profitant du fait que le logiciel utilisé permette de transformer l'organigramme en langage pseudo-code, obtiennent l'algorithme ci-dessous :

```
DÉBUT
  ÉCRIRE "Réponses possibles : trop_grand, trop_petit, égal"
  VP1=0
  VP2=1000
  VP=(VP1+VP2)/2
  ÉCRIRE VP
  LIRE R
  TANTQUE R!="égal" FAIRE[80]
    SI R="trop_petit" ALORS
      VP1=VP
    SINON
      VP2=VP
    FINSI
    VP=PLANCHER((VP1+VP2)/2)
    ÉCRIRE VP
    LIRE R
  FINTANTQUE
  ÉCRIRE "J'ai trouvé le bon nombre",VP
FIN
```

Figure 108 (Algorithmes avec commentaire pour décrire les « chaines de caractères » pouvant servir de réponses à l'ordinateur par l'utilisateur jouant le rôle du joueur B)

Le fait qu'AlgoBox impose pour une opération d'entrée une instruction « Afficher » précédant l'instruction « Lire » montre aussi des confusions chez certains élèves de Seconde qui semblent confondre les instructions « Lire » et « Afficher » (fig. 109).

[80] != signifie « différent de »

Figure 109 (Instruction « LIRE » et « AFFICHER » d'AlgoBox)

Nous constatons que ces difficultés sur les variables et les opérations d'entrée semblent ne plus concerner les élèves observés du cycle scientifique terminal. En effet, aucun des algorithmes qui nous ont été retournés par ces élèves ne révèlent des erreurs sur ce point.

4.4.1.4. Le problème de la *valeur entière*

Nous observons dans la classe de Seconde, une résistance de la part de certains binômes en ce qui concerne le calcul des moyennes de deux nombres entiers. En effet, ces binômes bien que sachant que le calcul d'une moyenne de deux entiers puisse ne pas donner un nombre entier, ne transfèrent pas systématiquement cet acquis algébrique travaillé au « papier-crayon » à un environnement informatique (fig. 110).

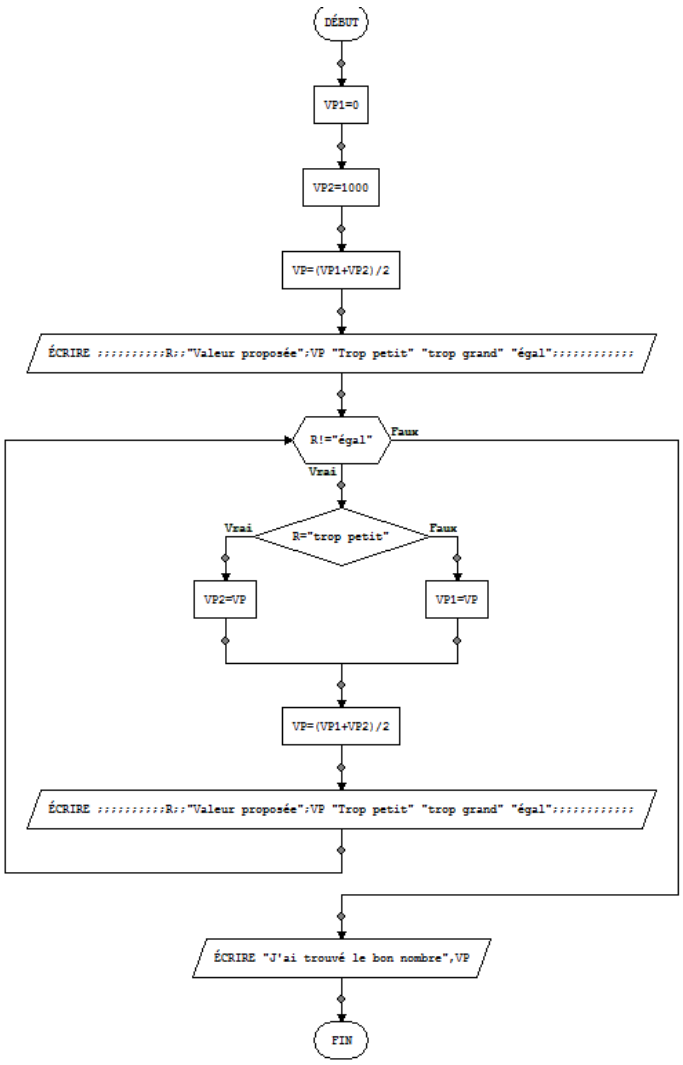


Figure 110 (Organigramme où l'élève oublie qu'une moyenne de deux nombres entiers peut ne pas être un nombre entier)

Pour certains de ces élèves, nous observons une conception erronée sur la nature des résultats d'opérations numériques dans un environnement informatique. En effet, nous voyons chez certains des « croyances » fortes venant entre autres du travail qu'ils ont pu mener au collège sur l'utilisation du tableur. En effet, nous rapportons ci-dessous un extrait illustré de l'échange entre les deux élèves (El1 et El2) ayant proposé l'organigramme (fig. 27) ci-dessus et l'enseignant (EnsSec) de la classe.

> EnsSec : […] *La structure de votre algorithme est totalement correcte. Les affections des données aux différentes variables sont aussi justes. Cependant, vous semblez oublier un point qui a été vu à la phase précédente concernant le calcul des moyennes. Voyez-vous ce que je veux dire ?*
> El1 : [Temps d'hésitation] *Non.*
> El2 : *Moi non plus.*
> EnsSec : *Que donne la moyenne de 50 et 70 ?*
> El2 : *60.*
> EnsSec : *Bien. Et celle de 50 et 60 ?*
> El2 : *55.*
> EnsSec. *D'accord. Et maintenant celle de 50 et 55 ?*
> El2 : *51,5. Euh… non. 52,5.*
> El1 : *Ce n'est pas un entier.*
> EnsSec : *Oui, ce n'est pas un entier. Alors, quel va être le problème avec la donnée de la variable VP située à l'intérieur de votre boucle ?*

L'enseignant quitte provisoirement ces deux élèves pour restaurer du silence parmi les autres binômes. Pendant ce temps qui dure environ 2 minutes, l'élève El2 prend l'initiative d'ouvrir une feuille du tableur « Excel » et commence à écrire dans les cellules A1 et B1 les entiers 125 et 84, puis dans la cellule C1 inscrit la formule =(A1+B1)/2 et regarde le résultat. Ensuite, il sélectionne la colonne C et avec le bouton droit de la souris cherche « Format de la cellule » et sélectionne « Nombre » puis dans la case à côté de « Nombre de décimales », inscrit le chiffre 0. Ensuite, il inscrit quelques nombres entiers choisis au hasard dans les cellules A2 à A6 et B2 à B6. Et à l'aide d'un copier-glisser complète la colonne C des calculs des moyennes, allant de C2 à C6 (Fig. 111). A ce moment, l'échange[81] entre les deux élèves reprend.

[81] Cet échange est une transcription telle quelle d'un extrait d'un enregistrement audio.

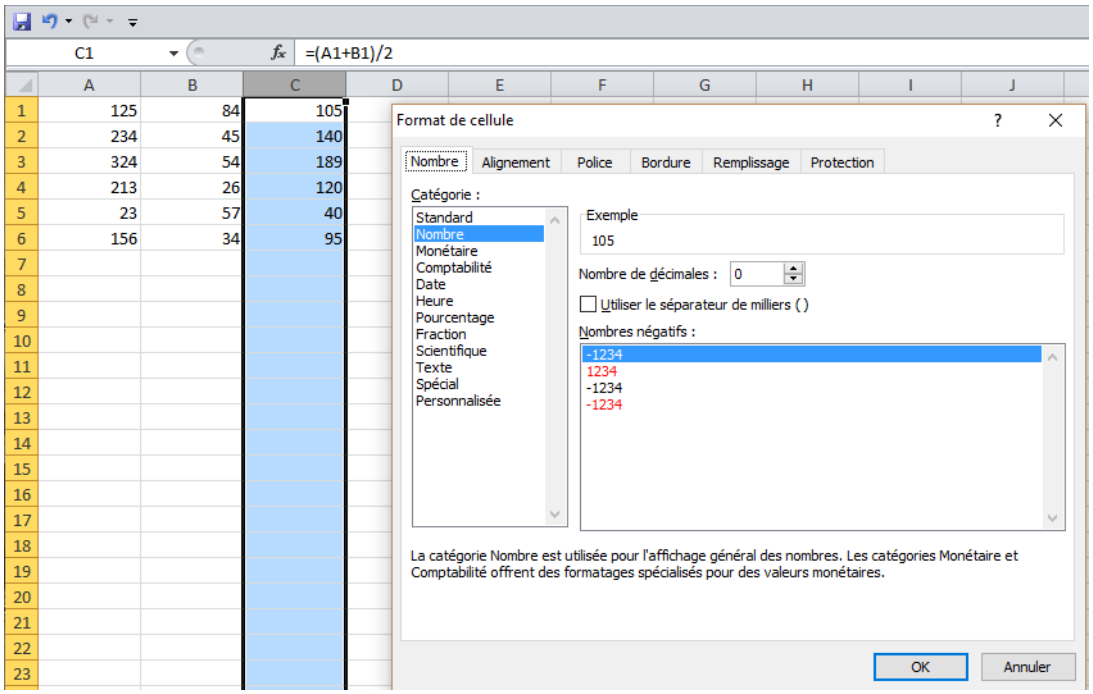


Figure 111 (Utilisation du tableur pour le calcul de le moyenne de deux nombres entiers)

El2 : *Regarde. Dans les cellules C2 à C6, toutes les moyennes sont entières. Or si on prend la moyenne de 125 et 84, on trouve 104,5.*
El1 : *Et tiens ?*
EL2 : *Il doit y avoir un truc semblable sur LARP. Il faut qu'on le trouve.*
[Retour de l'enseignant]
EnsSec : *Pourquoi cet écran Excel ?*
El2 : *Monsieur. Regardez sur Excel, les moyennes sont toutes entières car j'ai utilisé pour le format de la colonne des nombres à 0 décimale. Y a-t-il une possibilité de faire cela sur LARP ?*
EnsSec : *Je ne crois pas.* [Temps d'hésitation] *En fait, tu ne réponds pas au problème concernant la variable VP avec ta proposition. Tu utilises le fait que le tableur peut travailler d'une certaine manière avec des types de nombres particuliers. Et quand ils opèrent sur ces nombres, le résultat peut rester dans le même ensemble.*
El2 : *Je comprends pas.*
EnsSec : *Il existe des environnements informatiques uniquement faits pour la programmation et qui prennent en compte divers types de donnée standards : – les entiers ; – les réels ; – les chaînes de caractères ; et un quatrième que l'on nomme : les booléens. Mais dans les environnements LARP ou AlgoBox que vous utilisez le « type nombre » que vous utilisez ne prend pas en compte ces aspects « entiers » ou « réels ».*
[Commentaire : Cet enseignant enseigne entre autres en ISN[82]]
El2 : *C'est pour cela que vous avez parlé de partie entière dans la phase d'avant ?*
EnsSec : *Oui.* [Temps d'arrêt] *Revenons à notre problématique sur le VP dans la boucle.* [Temps d'arrêt] *Trouvez donc une solution permettant de tenir compte de ce problème concernant les données de VP. Elles doivent toutes être des nombres entiers.* [L'enseignant quitte alors ces deux élèves].

Nous observons donc que pour des élèves débutants les connaissances anciennes sur

[82] Informatique et sciences du numérique, nouvel enseignement de spécialité en Terminal Scientifique.

l'utilisation d'outils TICE peuvent être réactivées. Mais ceci n'autorise pas automatiquement une transposition d'un environnement numérique sur un autre. Cependant, ce binôme semble montrer que le travail fait lors des phases précédentes sur les variables itératives pour des algorithmes « papiers-crayons », peut être une source de questionnement en lien avec certains acquis anciens.

Nous constatons aussi que dans l'environnement *LARP*, les élèves de Seconde qui veulent utiliser le fait que la donnée numérique stockée dans la variable NbCh doit être un nombre entier éprouvent des difficultés quant à la reconnaissance de la fonction correspondante à la « fonction partie entière » proposée par cet environnement. En effet, *LARP* propose deux fonctions : « PLAFOND » qui retourne le plus petit entier supérieur ou égal à la valeur donnée et « PLANCHER » qui retourne le plus grand entier inférieur ou égal à la valeur donnée. A ce niveau scolaire, certains élèves n'arrivent pas nécessairement à faire la correspondance complète entre la « fonction partie entière » et celle qui lui correspond parmi ces deux fonctions proposées par l'environnement *LARP*.

Au niveau des classes scientifiques du cycle terminal, nous n'avons pas observé de difficultés sur le fait que la valeur moyenne devait être ramenée à un nombre entier. Les élèves ont bien conscience que la moyenne de deux entiers n'est pas nécessairement entière et que par conséquent il est obligatoire d'en tenir compte dans l'algorithme. Les seules difficultés observées sont des problèmes liés à la syntaxe de l'environnement, mais qui la plupart des cas sont surmontées par les élèves des classes scientifiques du cycle terminal contrairement aux élèves de Seconde (Fig. 112 et 113). Nous pouvons supposer que le fait que ce soit des élèves à vocation scientifique favorise cette meilleure maitrise des syntaxes des langages machines.

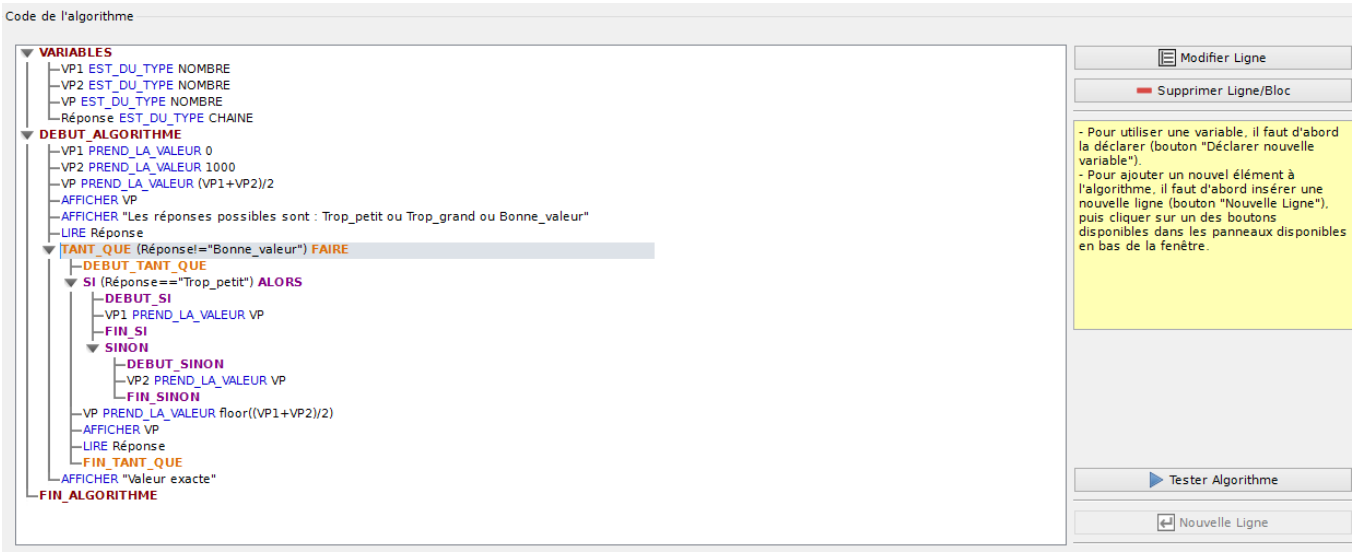
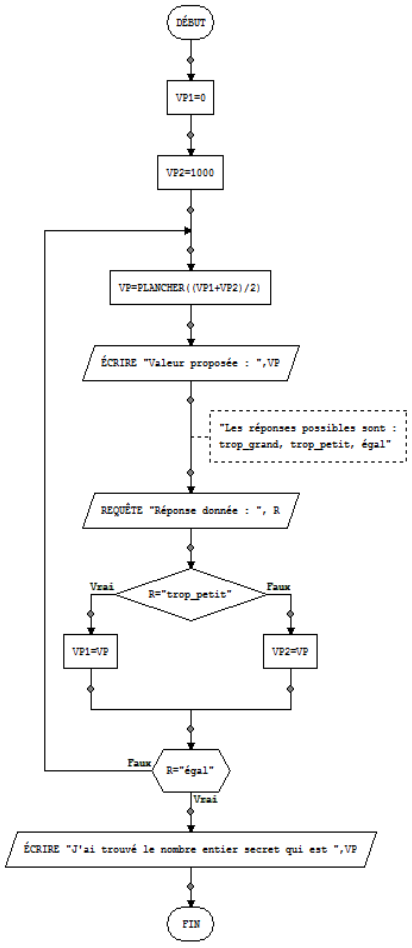


Figure 112 (Un algorithme du joueur A)



et sa transcription en pseudo-code :

```
DÉBUT
   VP1=0
   VP2=1000
   RÉPÉTER
      VP=PLANCHER((VP1+VP2)/2)
      ÉCRIRE "Valeur proposée : ",VP

      \\ "Les réponses possibles sont : trop_grand, trop_petit, égal"
      REQUÊTE "Réponse donnée : ", R
      SI R="trop_petit" ALORS
         VP1=VP
      SINON
         VP2=VP
      FINSI
   JUSQU'À R="égal"
   ÉCRIRE "J'ai trouvé le nombre entier secret qui est ",VP
FIN
```

Figure 113 (Organigramme de l'algorithme du joueur A et sa transcription en pseudo-code)

4.4.2 Analyse *a posteriori*

Comme pour la phase 2, cette dernière phase dure plus longtemps que ce qui est prévu lors de l'analyse *a priori*, quel que soit le niveau scolaire. En particulier dans la classe de

Seconde, pour des raisons de temps disponible, les élèves sont amenés à terminer le travail à la maison.

L'origine de ce décalage par rapport à l'horaire prévu, vient en partie du fait que les enseignants, avec l'accord du chercheur, comme pour la phase 2, laisse une réelle prise d'initiative aux élèves. Et pour cela, des échanges oraux entre les différents binômes sont encouragés.

4.4.2.1. L'environnement informatique et choix des structures

Nous analysons les difficultés rencontrées en référence à l'analyse *a priori* et au déroulement.

a) En Seconde

Lors de la phase 2, les élèves avaient à leur disposition un polycopié avec les modèles graphiques pour les organigrammes, mais ceux-ci étant établis au format « papier-crayon » ne respectaient pas toujours strictement les conventions relatives à ces modèles (exemple des représentations avec « modèles » rectangles pour toutes les instructions), Ceci n'empêche pas, chez les binômes qui utilisent le logiciel *LARP*, à la phase 3 de se conformer aux contraintes d'expression strictes dans ce logiciel. En effet, ces binômes utilisent les modèles standards proposés par *LARP* comme des briques de construction (fig. 114 et 115).

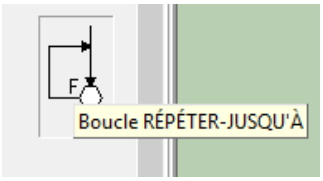


Figure 114 (Brique de construction d'une boucle « Répéter-Jusqu'à » avec LARP)

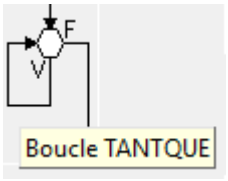


Figure 115 Brique de construction d'une boucle « TantQue » avec LARP)

Pour les binômes travaillant sur *LARP*, nous observons qu'ils utilisent une des fonctionnalités du logiciel leur permettant d'afficher les correspondances graphiques et « textuelles » comme nous le montre les figures ci-dessus (Fig. 114 et 115). Ainsi, ceci peut aider des élèves débutants en informatique à transcrire plus facilement tout ou une partie de l'organigramme au format « textuel », d'autant plus que cet environnement numérique leur donne aussi la possibilité de transformer un algorithme écrit sous forme d'un organigramme en un algorithme écrit en langage pseudo-code, le contraire n'étant pas possible. Les élèves utilisent donc majoritairement les modèles proposés par *LARP* et rendent un algorithme (« spatial » ou « textuel ») correct. Nous pensons que pour cela, ils ont tiré parti du travail

réalisé lors des phases 1 et 2, ainsi que des apports tutoriels du logiciel.

Lors de cette phase, quel que soit le travail fait pendant les phases précédentes, nous observons que le choix d'une structure de type « TantQue » est majoritairement retenue par les binômes. Lors de la phase 2, la majorité des organigrammes « papier-crayon » produits avait une structure du type « Boucle…FinBoucle ». Cette structure est proche du « Répéter...Jusqu'à » en ce sens que la sortie de boucle est placée en fin de boucle. Cependant, le test de sortie dans « Répéter…. Jusqu'à » n'est pas le même et donc le passage de la structure majoritairement adoptée en phase 2 à « Répéter...Jusqu'à » n'est pas immédiat. Nous pensons que cela a joué un rôle en faveur du « TantQue », d'autant plus que « TantQue » était la seule structure possible pour les élèves ayant choisi *Algobox*, et que donc elle a pu s'imposer par diffusion entre les binômes.

Parmi les binômes qui, au cours de la phase 2, ont dessiné des organigrammes présentant des structures insuffisamment précises pour reconnaitre de façon aisée si la structure était du type « TantQue » ou « Répéter… Jusqu'à », nous n'avons pas observé que ce problème était toujours d'actualité au cours de cette troisième phase, et nous supposons que cela vient de l'aide tutorielle qu'offre le logiciel *LARP* et des échanges entre les binômes, sachant que certains n'avaient pas eu de réelles difficultés dans le respect des modèles graphiques pour dessiner un organigramme.

b) En Première et en Terminale

Lors de cette phase, quel que soit le travail fait pendant les deux phases précédentes, nous observons que le choix d'une structure de type « TantQue » est très majoritairement retenu par les binômes. Nous pensons que cela est dû au fait que la majorité des binômes sont restés pour l'essentiel du travail sur ordinateur dans un environnement n'autorisant que la structure « TantQue ». Quant aux binômes ayant seulement utilisé l'environnement numérique *LARP*, nous avons eu la confirmation par certains de ces élèves lors d'un échange après l'ingénierie, que le fait que les binômes puissent échanger entre eux, a favorisé ce choix de structure lors de cette dernière phase et ceci quel que soit le travail fait lors de la phase précédente.

4.4.2.2. Les problèmes d'ordre syntaxique

Conformément à ce qui est prévu au cours de l'analyse *a priori*, des problèmes dus à la syntaxe du langage dans l'environnement informatique posent des difficultés aux élèves et

ceci quel que soit le niveau scolaire, même si des nuances sont à apporter.

a) En Seconde

Pour l'ensemble des binômes choisissant de travailler sur le logiciel *LARP*, le signe « = » pose des difficultés. En effet, celui-ci peut avoir deux significations : (1) affectation à une variable ou (2) opérateurs relationnels permettant de comparer deux valeurs. De plus, près de 90% de ces binômes ont beaucoup de difficulté à trouver la correspondance parmi les fonctions proposées par le logiciel celle qui correspond à la fonction *partie entière*. Quand ils pensent l'avoir trouvée, 10% éprouvent encore des difficultés car deux fonctions leur sont proposées : « PLAFOND » qui retourne le plus petit entier supérieur ou égal à la valeur donnée et « PLANCHER » qui retourne le plus grand entier inférieur ou égal à la valeur donnée, et ils n'arrivent pas à faire la correspondance complète avec la fonction *partie entière*. Lors d'un échange avec l'enseignant, nous apprenons que cela est dû à une confusion entre valeur entière et valeur approchée. En effet, ces élèves n'arrivent pas à intégrer que « 18,5 est égal à 19 à l'unité près par excès » ne signifie pas la même chose que « la valeur entière de 18,5 est 18 ». On retrouve ainsi une problématique qui s'était posée lors de la phase 2.

Nous avons la confirmation que la transcription d'un algorithme « papier-crayon » écrit en « langage naturel » en un algorithme écrit en « langage algorithmique » exécutable sur machine peut poser de nombreux problèmes chez des élèves débutants en informatique. Contrairement à ce qui est observé pendant les phases 1 et 2, les élèves ont plus recours à l'aide de l'enseignant pendant cette phase de transcription sur machine de l'algorithme « papier-crayon ».

b) En Première et en Terminale

Nous avons la confirmation que le travail fait en algorithmique depuis la Seconde pour la classe de Première, voire depuis la Seconde et la Première pour la classe Terminale, favorise une transcription d'un algorithme « papier-crayon » en un algorithme écrit en langage exécutable quand l'environnement informatique choisi est celui sur lequel les élèves travaillent depuis un an (ou deux ans). Cependant, quand cet environnement est nouveau, les élèves de ces niveaux scolaires éprouvent aussi des difficultés sur le plan syntaxique quant à la transcription de l'algorithme « papier-crayon » en un langage exécutable. Mais, nous n'avons pas de retour suffisamment précis, permettant d'attester totalement cette

observation.

4.4.2.3. La mise en place des variables

Pour toutes les classes observées et quel que soit le niveau scolaire, contrairement à ce que l'on pouvait s'attendre lors de l'analyse *a priori*, l'implémentation et l'exécution ne mettent pas en évidence de véritables faiblesses en ce qui concerne l'utilisation et l'organisation des variables itératives Binf (pour la borne inférieure) et Bsup (pour la borne supérieure), puis la valeur moyenne NbCh de ces deux bornes. Nous en déduisons que le travail fait par les élèves et les interventions ponctuelles des enseignants, quand cela s'avérait nécessaire, au cours des phases précédentes a permis aux élèves de surmonter les difficultés quand elles étaient présentes sur cet aspect informatique.

Une difficulté apparaît avec le typage de variables dans *Algobox*, en particulier chez les élèves de Seconde. Les variables dans cet environnement doivent être déclarées de type « Nombre », « Chaîne » ou « Liste ». Les opérations de lecture et d'écriture peuvent se faire sur des variables des trois types. Une comparaison entre des variables de types différents est syntaxiquement correcte et n'entraîne pas d'erreur à l'exécution. L'observation montre de nombreuses difficultés avec le type « Chaîne de caractère », bien que les variables de ce type n'interviennent que dans des opérations d'entrée et des comparaisons. Cette problématique n'avait été vue au cours des phases précédentes. Nous pensons qu'un environnement « papier-crayon » ne favorise pas la prise de conscience des différences qui existent entre les différents types de variables chez l'élève débutant en informatique. Cependant, la confrontation au dispositif lors de l'exécution n'est pas nécessairement productive : l'observation montre que les messages d'erreur de l'environnement de programmation aident peu l'élève à comprendre les raisons du non fonctionnement du programme.

De plus, *AlgoBox* impose pour une opération d'entrée une instruction « Afficher » afin de préciser le rôle de la variable qui peut suivre lors de l'instruction « Lire ». La conséquence est que certains binômes semblent confondre les instructions « Lire » et « Afficher », ce qui renvoie à la difficulté à concevoir les positions respectives de l'ordinateur et de l'utilisateur. Ici encore, l'intervention de l'enseignant peut s'avérer nécessaire afin de débloquer les élèves qui en ont besoin.

4.4.2.4. Une difficulté qui persiste sur la valeur entière

a) En Seconde

Contrairement à ce que nous prévoyions dans l'analyse *a priori*, une minorité d'élèves (10% des binômes) ne semblent pas avoir pris conscience de ce que le calcul de la moyenne de deux nombres entiers par une machine ne renvoie pas automatiquement un nombre entier. Cela peut être dû au fait que ces élèves ne trouvent pas l'expression de la fonction *partie entière* du logiciel ou bien encore qu'ils ne comprennent pas la signification de la partie entière ou plus simplement qu'ils imaginent que la machine renvoie automatiquement une valeur entière pour la moyenne. Lors d'un entretien avec deux de ces élèves concernés par cette difficulté, il en est ressorti que certains élèves ne pensent pas qu'il puisse y avoir une différence entre les actions menées par la machine et par l'être humain : dans le contexte de la dichotomie « discrète », seuls des entiers sont concernés, ce qui est implicite pour l'humain.

Une intervention de l'enseignant, mais aussi la confrontation au dispositif informatique lors de l'exécution s'avèrent nécessaires pour que l'élève puisse là-aussi surmonter ses représentations erronées de ce dispositif.

b) En première et en Terminale

A ces niveaux scolaires, aucune erreur concernant la valeur entière n'est observé, si ce n'est des questions d'ordre syntaxique comme nous avons pu le voir précédemment. En effet, l'élève a bien conscience que la moyenne de deux nombres entiers n'est pas nécessairement entière et que par conséquent il est obligatoire d'en tenir compte dans l'algorithme. De même au niveau du cycle terminal scientifique, les aspects mathématiques du problème ne posent plus de difficultés chez les élèves qui en avaient eu lors des phases précédentes. Seuls des problèmes concernant le langage syntaxique du logiciel utilisé peuvent nécessiter une intervention de l'enseignant.

4.4.2.5. Synthèse

En référence à l'analyse *a priori*, les difficultés en lien avec l'*alphabétisation informatique* (Rogalski, 1988) chez certains élèves débutants en informatique, en particulier au niveau de la Seconde, se sont produites lors de la confrontation à l'expression de l'algorithme sous une forme exécutable dans l'environnement informatique dédié. Nous observons que les interactions entre domaines mathématique et algorithmique peuvent être source d'obstacles

chez certains élèves, plus particulièrement de Seconde, en particulier avec le risque de confusion entre les concepts de variable informatique (Samurçay, 1985) et de variable mathématique.

Le typage de variables dans *AlgoBox* révèle une difficulté à concevoir le type « Chaîne de caractère ». Nous interprétons ceci comme la conséquence de ce que les élèves ont rencontré exclusivement des algorithmes traitant de nombres (entiers ou réels). Dans le cas de la dichotomie, des chaînes apparaissent lors des interactions entre la machine et l'utilisateur. Ce type de donnée est difficile à concevoir pour des élèves dont la représentation de l'ordinateur reste une « super calculatrice » et ne fait pas l'objet d'un enseignement.

Nous voyons aussi que, pour des élèves débutants, une difficulté est de concevoir les positions relatives de la machine et de l'utilisateur. Nous associons ces difficultés d'une part au fait qu'en Seconde les élèves sont des débutants en informatique et d'autre part qu'il s'agit d'un niveau indifférencié. En effet, dans les classes du cycle terminal scientifique, nous n'avons pratiquement plus observé de telles difficultés.

De plus, la difficulté pour des élèves de Seconde à considérer le résultat du calcul d'une moyenne de deux nombres entiers par une machine n'est pas systématiquement un nombre entier, peut montrer que certains n'ont pas conscience des différences entre les actions menées par la machine et par l'être humain. Ceci nous renvoie aux travaux de Rogalski (1988) sur *les représentations mentales du dispositif informatique* chez le débutant en informatique. Par ailleurs, ces élèves ne cherchent pas spontanément à utiliser une fonction « partie entière » existante dans l'environnement de programmation, alors que cette utilisation est discutée lors de la phase 2 quand le travail algorithmique se fait au format « papier-crayon ». Certains élèves pensent trouver une réponse en réglant un format d'affichage comme dans un tableur. Ils n'ont pas conscience de ce que le tableur peut afficher un entier dans une cellule alors que la valeur interne de la cellule n'est pas nécessairement entière. Ils cherchent à obtenir un entier par une action (régler un format) plutôt que comme résultat d'une fonction. Les difficultés renvoient ainsi à la conception des fonctions et du formalisme fonctionnel et donc à un domaine mathématique. Au niveau du cycle terminal, cette erreur semble ne plus se produire au cours de cette dernière phase. Nous pouvons supposer que ces élèves étant dans des classes scientifiques ont une plus grande maturité relativement aux fonctions qui leur permet de mieux intégrer le formalisme fonctionnel dans l'environnement de programmation.

Au cours de cette phase, nous observons qu'un travail en amont sur la conception d'une structure algorithmique au format « papier-crayon », permet à l'élève d'aborder la construction d'une structure algorithmique lors du travail sur un algorithme exécutable sur machine. Cependant, le passage d'une structure « Répéter… Jusqu'à » écrite au format « papier-crayon » semble provoquer des résistances chez certains élèves quel que soit leur niveau scolaire, lors de son expression dans les langages des environnements de programmation proposés. Nous pensons que le fait que pour sortir de cette boucle, la condition d'arrêt doit être une égalité peut poser des difficultés cognitives chez l'élève qui sera plutôt tenté de prendre une inégalité comme condition d'arrêt (cf. Chapitre 5). Le problème se pose ainsi de la négation d'une condition. Nous avons rapporté l'observation d'un binôme faisant référence pour cela aux événements complémentaires en probabilité.

Pour conclure, les élèves de Seconde ayant rencontré plus de difficultés que les élèves du cycle scientifique terminal pour la mise en œuvre d'une valeur entière lors du calcul d'une moyenne par une machine et pour la prise en compte des contraintes syntaxiques dues à l'environnement informatique utilisé, nous interprétons cela comme la conséquence d'une pratique régulière de l'algorithmique par les élèves du cycle terminal depuis la Seconde. De plus, s'agissant d'élèves d'une filière scientifique, une meilleure réussite dans la conception d'algorithmes et la compréhension de nouveaux concepts mathématiques est souvent constatée chez ces élèves, par comparaison avec d'autres filières où les difficultés tendent à persister. Comme nous le signalions déjà dans la synthèse de la phase 2, en accord avec Lagrange & Rogalski (2016), la question reste toujours ouverte de précurseurs à l'activité de programmation dans les apprentissages mathématiques, ou de précurseurs communs tels que des dispositions au raisonnement logique ou à la conception de l'espace.

5. Synthèse de la sous-ingénierie « algorithme de dichotomie discrète »

5.1 Les structures

Comme nous l'avions prévu dans l'analyse *a priori*, le travail sur les structures a été laborieux pour les élèves. En conséquence, nous avions choisi de demander aux élèves dans une première phase d'axer leurs efforts sur l'algorithme du joueur A (choix du nombre entier

secret et réponses aux propositions du joueur B), au lieu d'aborder directement la réalisation de l'algorithme du joueur B. Ce choix a été productif en permettant aux élèves de travailler d'abord sur la structure avant de se confronter aux difficultés des variables itératives. Nous pouvons penser que cela puisse être une conséquence chez certains élèves débutants, du fait que les représentations liées aux applications informatiques dans la sphère sociale, comme la console de jeu électronique, le téléphone portable,… pourraient privilégier l'idée de structure au détriment de l'idée de variable.

Ainsi, les élèves peuvent mettre en évidence une structure pour le joueur A (dans les deux premières phases) et la réutiliser pour le joueur B dans les deux autres phases. Pour cela, à partir d'essais en jeu réel, ils sont conduits à systématiser la proposition du joueur B en fonction d'un diagnostic de comparaison donné par le joueur A. Ils découvrent ainsi une stratégie « gagnante » et « rapide » apportant une solution à ce problème, sans que soit discutée la question de l'optimalité qui, comme nous l'avons déjà signalé, ne peut être traitée aux niveaux scolaires où ont eu lieu notre expérimentation (Seconde et cycle scientifique du lycée). Passant à l'algorithme du joueur B, ils explicitent une règle du choix d'un nombre à proposer (implicite lors du jeu réel) sous forme d'une alternative à l'intérieur du corps de l'itération.

Nous attendons que le travail fourni par les élèves sur ces structures (algorithmes du joueur A et du joueur B) soit réinvesti par les élèves lors de la sous-ingénierie sur la « *dichotomie continue* » (cf. Chapitre 5).

5.2 Les variables

La conception des variables itératives a été d'une grande difficulté pour les élèves lors de l'élaboration de l'algorithme du joueur B (celui qui doit trouver le nombre). Le temps alloué lors de l'analyse *a priori* a été sous-estimé, plus particulièrement en Seconde. Les élèves créent initialement des nouvelles variables à chaque itération au lieu de concevoir une variable itérative. Le passage d'une méthode de résolution du problème, exprimée en langage naturel ou sous forme d'organigrammes en « papier-crayon » à un environnement écrit sous forme exécutable les conduit ensuite à concevoir des variables itératives plutôt que d'utiliser une nouvelle variable à chaque pas de l'itération. Notre choix relatif à l'expression des algorithmes (cf. 4.3) s'est donc révélé pertinent.

Nous constatons aussi que malgré une première phase impliquant dans l'algorithme du joueur A, la proposition de nombres entiers par le joueur B, la nécessité de travailler avec des valeurs entières est restée, chez certains élèves de Seconde, au stade implicite lors du passage à l'algorithme du joueur B. En effet, pour ces élèves, l'écriture $\frac{a+b}{2}$ semble donner automatiquement un nombre entier. On peut penser qu'ils ne conçoivent pas la notion de « partie entière » comme pouvant être une fonction devant être appelée explicitement lors du passage à l'algorithmique. Nous pensons que cela peut être dû à l'absence de tout travail sur une telle fonction qui n'est plus officiellement dans les programmes de mathématiques du lycée.

5.3 Les modes d'expression

Nous avons indiqué dans la description de la sous-ingénierie que dans les deux premières phases les élèves étaient libres de choisir le mode d'expression de la méthode. Cependant, les algorithmes écrits en langage « naturel » sont privilégiés par une majorité des élèves. Comme nous avons pu le rapporter lors des analyses *a posteriori*, ce travail en langage « naturel » des algorithmes a été d'une grande aide aux élèves. En effet, par exemple un tel travail leur a permis de mieux expliciter une règle du choix d'un nombre, c'est-à-dire de concevoir un « traitement alternatif ».

Dans les phases suivantes, il était indiqué d'exprimer la méthode sous forme d'organigrammes en « papier-crayon » puis dans un environnement permettant leur expression sous forme exécutable : logiciels dédiés (*AlgoBox* et *LARP*) ou calculatrice (ce qui n'a pas été choisi par l'ensemble des élèves, excepté un binôme de Première). Nous avions fait le choix de leur demander l'expression de ce traitement sous forme d'organigramme pour deux raisons : la possibilité d'une boucle et la plus grande lisibilité d'une disposition « spatiale ». L'ensemble des élèves, quel que soit le niveau scolaire, rencontraient cette forme d'expression pour la première fois.

Comme prévu, l'organigramme en « papier-crayon » a permis à un grand nombre d'élèves, une expression de la structure, même pour des binômes n'ayant pas une structure totalement correcte dans le langage « naturel ». Le renvoi conditionnel de type « GOTO » permet, comme nous l'avions prévu, l'itération et le traitement alternatif sans que ceux-ci soient spécifiés par des marqueurs explicites.

Ensuite, une majorité d'élèves de Seconde choisit *LARP* qui permet d'implémenter l'organigramme « papier-crayon » au prix d'adaptations pour le rendre compatible avec ce logiciel qui, implicitement permet seulement l'itération avec une des structures du type « Répéter…Jusqu'à » ou « TantQue… Faire ».

Les élèves de Première et Terminale se sont dirigés plutôt sur l'environnement *AlgoBox* (sauf pour un binôme de Première qui a préféré travailler avec la calculatrice). Nous rappelons que cet environnement est familier pour ces élèves du cycle scientifique terminal. Pour ceux-là, il fut encore nécessaire de procéder à des adaptations supplémentaires : déclarer les variables, convertir la structure « Répéter… Jusqu'à » en « TantQue… Faire » qui implique une négation de la condition de sortie et l'inclusion d'un marqueur de fin de corps de boucle « FinTantQue ». Des médiations de l'enseignant ont été nécessaires en particulier sur le plan de la syntaxe. Ceci témoigne de difficultés qui se maintiennent pour des débutants même après les tous premiers apprentissages.

Notre choix de proposer un environnement permettant l'implémentation d'un organigramme est donc productif pour les élèves, notamment en Seconde, qui retiennent cette proposition : les adaptations nécessaires sont du domaine de la structure plutôt que de la syntaxe. Par exemple, comme nous avons pu le voir au début de ce chapitre (cf. les sections 1.2.3.1 et 1.2.3.2) la structure « Répéter » avec sortie à l'intérieur de la boucle est faisable sur un organigramme « papier-crayon ». Elle est utilisée par un binôme de Seconde lors de la première phase. Ces élèves adaptent sans difficultés cette structure à une structure « Répéter… Jusqu'à » pour passer dans l'environnement numérique. On peut penser que pour des élèves non conditionnés aux structures généralement imposées par les logiciels, la possibilité d'utiliser une sortie à l'intérieur de la boucle constitue une bonne entrée dans la tâche, en évitant de les bloquer dès le départ par une structure contraignante.

L'utilisation des logiciels permet aux élèves de quantifier le nombre d'essais (coût de l'algorithme) et donc d'avoir une conception plus précise de l'idée de « rapidité », notamment lorsque certains prennent l'initiative de faire calculer la moyenne du coût sur un grand nombre de parties. Les élèves prennent ainsi conscience de l'intérêt d'écrire un algorithme pour mettre en forme une stratégie imaginée d'abord de façon intuitive.

Le passage à une expression de l'algorithme dans un des logiciels proposés (*AlgoBox* ou

LARP ou calculatrice) permet une explicitation d'une méthode d'abord conçue de façon intuitive : systématisation de la méthode pour déterminer une structure, identification des données impliquées dans la méthode, détermination de variables itératives représentant les états successifs de ces données et calcul à faire sur ces variables.

6. Les perspectives pour la sous-ingénierie « algorithme de dichotomie continue » : articulation ETA – ETM ; Niveaux de paradigmes

6.1 Présentation

Afin de terminer la présentation de cette sous-ingénierie sur la « dichotomie discrète », nous proposons de présenter une étude des compétences mises en place en termes d'articulations d'ETA et d'ETM spécifiques, complétée d'une approche en fonction des niveaux de paradigmes algorithmiques utilisés. En effet, nous souhaitons ainsi présenter un bilan de « l'installation d'un ETA » et ce que sont maintenant les objets, les règles et les artefacts à disposition des élèves en cette fin de sous-ingénierie. En particulier, nous abordons les différentes visualisations, constructions, voire preuves du choix de prendre comme stratégie « gagnante » et « rapide » la méthode de dichotomie pour répondre à la problématique du jeu de la recherche d'un nombre entier « secret ».

Nous montrons aussi comment des éléments mathématiques peuvent intervenir, mais sans constituer un ETM en eux-mêmes : ainsi, la dichotomie est, après cette sous-ingénierie, un objet algorithmique, mais pas nécessairement un objet d'un Espace de Travail Mathématique spécifique au domaine de l'*analyse*.

Nous complétons cette approche sur les ETA, par un développement sur des ETA personnels qui diffèrent d'un niveau scolaire à l'autre permettant de mieux approcher les perspectives pour la sous-ingénierie « dichotomie continue ».

6.2 Les tableaux

6.2.1 Les paradigmes

Les niveaux de paradigmes algorithmiques	
Phase 1 : Conception générale d'un algorithme	→ Au cours de cette première phase, les élèves doivent se familiariser avec le jeu du « nombre entier secret » afin de concevoir la structure d'un algorithme écrit au « papier-crayon » sous une forme « textuelle » ou « spatiale » représentant le joueur qui choisit « secrètement » et au hasard ce nombre entier « secret » compris entre deux entiers connus. Ainsi, les élèves en majorité débutants en informatique, doivent

répondant à la tâche du joueur A, c'est-à-dire celui qui pense au nombre entier « secret ».	établir la structure algorithmique correspondant à la tâche du joueur A. Pour cela, ils ont la possibilité de proposer plusieurs « solutions » : • une utilisation d'alternatives successives plutôt qu'imbriquées et des expressions de boucle de type « Si *condition* sortir sinon *recommencer* », c'est-à-dire du type « GOTO » ; • une structure de type « Boucle…FinBoucle » directement déduite de la règle du jeu ; • une structure type « Répéter… Jusqu'à » qui nous semble moins s'éloigner de la règle du jeu que la structure « Tant Que…Faire ». La construction d'une structure algorithmique va situer le travail attendu des élèves au niveau 2 des paradigmes algorithmiques, en particulier pour l'étude du test de sortie de la boucle. → De plus, les élèves doivent mettre en place des variables informatique : • dans le cas du joueur A : le nombre « secret » est une constante ; • le choix du joueur B est lu à chaque pas de l'itération. Conformément aux travaux de Rogalski et Samurçay (1986), pour des débutants en programmation, le travail sur la variable constante concernant le joueur A va se situer au niveau 1 des paradigmes algorithmiques, tandis que celui sur les variables évoluant au cours de l'itération, ici représenté par les « choix » du joueur B, va se situer au niveau 2.
Phases 2 et 3 : Conception générale d'un algorithme répondant aux tâches du joueur B, c'est-à-dire celui qui propose des réponses.	Au cours de ces deux phases, les élèves conçoivent un algorithme associé à la tâche du joueur B. Nous ne distinguons pas ces deux phases car d'un point de vue type de tâches, la phase 3 diffère peu de la phase 2 si ce n'est au niveau de la maitrise de la syntaxe du langage machine utilisé pour l'implémentation de l'algorithme déterminé lors de la phase 2. Les élèves ont donc à décrire une stratégie gagnante en un minimum de coups où le joueur B va être joué par la machine. Ceci peut être interprété en termes d'optimalité par les élèves et par conséquent situé cette partie du travail au niveau 2 des paradigmes algorithmiques. Les élèves doivent mettre ainsi en évidence, la réduction de l'intervalle grâce à la moyenne des bornes (dichotomie), comme assurant une stratégie « gagnante » et « rapide ». Cependant, cette étape va situer le travail au niveau 1 des paradigmes lors de la phase 2, car le fait de rendre entière cette moyenne peut se faire de façon implicite par des élèves n'ayant pas la connaissance du concept de fonction partie entière. En revanche, pour la phase 3, cette « automatisation » implicite n'étant plus possible du fait que la machine ne peut réagir qu'à des instructions prédéfinies par l'utilisateur, ici l'utilisation de la fonction partie entière, va faire que pour cette tâche les élèves vont être obligés de situer leur travail au niveau 2 des paradigmes. Concernant les tâches sur les structures et les tests de sortie de boucles, nous n'avons pas de réelles difficultés supplémentaires par rapport au travail attendu lors de la phase 1. En revanche, bien que l'algorithme du jouer A possède la même structure que celle du joueur B, nous avons dans le cas du joueur B à organiser une gestion de variables de boucle supplémentaires. En effet, le type de variable dans le cas du joueur B va nécessiter de la part de l'élève, en particulier lors de la phase 3, de distinguer des variables de type « Nombre » de celles de type « Chaine de caractères ». Ainsi, ce travail nécessaire pour la phase 3, amène l'élève à se situer au niveau 2 des paradigmes algorithmiques.

Tableau 31

6.2.2 Les Espaces de Travail

Espaces de Travail spécifiques	***ETA***	***ETM***	**Commentaires afin de montrer que les deux précédentes sont porteuses d'idées intéressantes**
Exploiter le travail fait sur l'algorithme de la phase 1 pour ses apports à la construction de celui de la phase 2.	Pour chaque phase, l'élève doit établir la structure algorithmique rendant compte du déroulement des échanges qui ont lieu entre les deux joueurs : pour les actions du joueur A dans le cas de la phase 1 ; pour celles du joueur B pendant la phase 2 (ainsi que la phase 3) : Pour les deux phases : • Condition dans l'alternative ; • Branches « SI et SINON » ; • Choix d'une structure algorithmique de type « Boucle…FinBoucle » directement déduite de la règle du jeu ; • Choix d'une structure « Répéter… Jusqu'à » qui semble moins s'éloigner de la règle du jeu que la structure « Tant Que…Faire » ; • Organiser un test de sortie de boucle à l'aide d'une égalité ou d'une inégalité suivant la structure retenue. (Genèse discursive) Pour la phase 1 : • mise en place des variables de type	Analyser la stratégie mise en place dans le jeu en rapport avec la méthode retenue : • découpage de l'intervalle ; • calcul de moyenne ; • valeur entière ; • comparaison ; • Evolution des valeurs des bornes des intervalles emboités. (Genèse discursive)	• Certains élèves utilisent des alternatives successives plutôt qu'imbriquées. Nous pouvons interpréter cela comme une réduction à un « copier-coller » renvoyant à un ETA de type « tableur » (absence de travail dans le domaine de la *logique*) ou à une transcription d'un raisonnement issu d'un ETM de probabilité avec la représentation d'une situation donnée, à l'aide d'un arbre. • D'autres élèves éprouvent des difficultés sur la condition de sortie de boucle. Comme nous avons pu le voir, ceci est dû en partie à une mauvaise interprétation d'une compétence venant d'un ETM de probabilité avec le concept d'*événement contraire*. La validité de l'algorithme, question posée dans l'ETA résulte d'arguments relatifs à la méthode, qui relèvent donc de l'ETM.

	« Nombre » intervenant dans le cas du joueur A tenant compte que le nombre « secret » est une constante ; Pour la phase 2 : Au cours de cette phase axée sur l'action du joueur B, apparait assez rapidement l'idée qu'une « stratégie » doit s'imposer sinon, comme le dit un élève, « le jeu devient trop long ». Les élèves sont ainsi amenés à exploiter les résultats obtenus sur le choix des structures faites au cours de la phase précédente, mais ils doivent aussi gérer différents types de variables : « Nombre » et « Chaine de caractère » contrairement à la phase précédente où seule des variables de type « Nombre » interviennent. (Genèses sémiotique et discursive)		
Algorithme de la phase 3 (implémentation dans un environnement numérique de l'algorithme obtenu à la phase 2)	Lors de cette dernière phase, les élèves reprennent l'algorithme écrit à la phase précédente et procèdent aux adaptations syntaxiques nécessaires pour l'implémenter dans un environnement numérique afin de le tester. (Genèse instrumentale)	Utilisation de la fonction partie entière et générer un nombre aléatoire (Genèse discursive)	Le travail sur l'approximation d'un nombre décimal peut générer des difficultés lors du passage à un ETA logiciel, contrairement à ce qui est observé dans un ETA où l'algorithme est construit en langage « naturel ». L'aspect ETM autour de la fonction *partie entière* peut donner du sens à des élèves débutants en

	Se pose aussi le problème des variables de type « Nombre » : • nombre entier « secret » correspondant à un nombre entre aléatoire compris entre 0 et 1000 ; • Calcul de la moyenne des bornes des intervalles successifs. Cette moyenne doit « renvoyer » un nombre entier. Par conséquent, les élèves doivent déterminer comment faire que ce nombre soit entier : utilisation de la fonction partie entière. (Genèses discursive et instrumentale) Les élèves doivent aussi manipuler des variables type « Chaine de caractère » correspondant aux réponses : « plus petit », « plus grand », « bon nombre ». (Genèses sémiotique et instrumentale)		programmation sur la reconnaissance de la partie entière d'un nombre décimal, voire réel, en particulier au niveau de la Seconde.

Représentations « spatiale » ou « textuelle » des algorithmes	Construire des algorithmes soit sous forme d'un organigramme soit sous forme « textuelle » avec l'utilisation d'un langage « pseudo-code ». (Genèse instrumentale)	• Vision « spatiale » ou « textuelle » de l'algorithme. (Genèse sémiotique/visualisation) • Construire la suite des bornes des intervalles emboités (Genèse instrumentale) • Reconnaitre la nature de la fonction permettant d'obtenir une série de valeurs entières approximant les résultats de chaque calcul de moyenne de deux nombres entiers (Genèse discursive)	• Les gestes de construction et de visualisation d'un algorithme au format « textuel » à mettre en place sont familiers pour des élèves de cycle terminal scientifique. • En revanche, chez des élèves n'ayant pas encore une « culture » de tel ou tel environnement numérique, nous pouvons observer que des constructions « spatiales » ou « textuelles » peuvent être complémentaires afin de mieux comprendre les différentes étapes d'un algorithme. • L'environnement *LARP* permet à des élèves débutants de mieux voir la transcription d'une représentation spatiale sous forme d'organigramme à une représentation « textuelle » sous forme de langage pseudo-code. Ainsi, le travail personnel de l'élève peut être envisagé dans des ETA spécifiques logiciels.
Choix d'une stratégie « rapide » et « gagnante » : algorithme de dichotomie « discrète »	Analyser le jeu sous la forme d'une stratégie de recherche. Interpréter la condition : *le joueur B doit trouver le nombre entier « secret » en un minimum de coups* comme relevant de l'optimisation et de l'efficacité du choix de	Interpréter : • La condition « en un minimum de coups » ; • la valeur approchée de la moyenne de deux nombres entiers. (Genèse discursive)	• La condition « en un minimum de coups » relève d'une stratégie particulière. On reste dans l'ETA. Cependant, cette condition permet d'introduire la méthode de dichotomie et donner ainsi du sens à l'utilité de cette méthode, même si la vitesse de

	la stratégie. (Genèse discursive)		convergence ne peut pas faire l'objet d'un questionnement au niveau des classes des lycées français qui relèverait d'un ETM d'analyse.
Initiatives prises par les élèves	Ajouter : • La détermination d'un nombre entier aléatoire compris entre 0 et 1000 permettant de représenter le nombre entier « secret » choisi par le joueur A ; • la construction des intervalles emboités encadrant ce nombre entier « secret » • la nécessité d'introduire la fonction partie entière afin d'obtenir des approximations par un nombre entier lors du calcul des différentes moyennes des bornes des intervalles (Genèse instrumentale)	• Nombre aléatoire entre deux entiers • Partie entière d'un nombre décimal (Genèses sémiotique/visualisation et instrumentale) • Déterminer la partie entière d'un nombre décimal. (Genèse instrumentale)	• Initiative prise en fonction d'un objectif dans l'ETM, impliquant une construction dans l'ETA.
Interpréter le choix du sous-intervalle encadrant le nombre entier « secret » après chaque calcul des moyennes	Du point de vue des propriétés de l'algorithme : • supposer que l'algorithme dichotomie « discrète » est un algorithme correspondant à une stratégie « rapide » et « gagnante » • considérer que l'exécution d'un algorithme peut justifier ce choix de la dichotomie en proposant un	Du point de vue de la méthode : • stratégie « rapide » (Genèse discursive)	

	compteur (Genèse discursive)		

Tableau 32

7. Conclusion du chapitre 4

En conclusion, la sous-ingénierie « *algorithme de dichotomie discrète* » conduit globalement aux effets escomptés, néanmoins le temps prévu est très nettement dépassé, ce qui est l'indice de difficultés plus résistantes que prévues. Ceci va nous conduire à prendre plus de précaution pour la suite de cette ingénierie, avec l'aspect « *algorithme de dichotomie continue* », comme nous allons le voir dans le chapitre suivant.

Pour le cas continu, les élèves vont ainsi pouvoir réinvestir la structure de l'algorithme obtenu pour le joueur B, avec cependant, des adaptations nécessaires sur la condition d'arrêt et sur les conditions des alternatives. Les variables itératives vont pouvoir aussi être réinvesties.

Pour finir, bien que cela ne fasse pas explicitement partie de notre problématique, on peut se demander si un travail sur l'affectation des variables en algorithmique, ne peut pas permettre de modifier les représentations erronées des élèves débutants en informatique sur les variables mathématiques ?

Chapitre 5 : Une ingénierie articulant ETA et ETM spécifiques (partie2) : *Algorithme de dichotomie* « continu » – Vers une preuve du TVI[83]

1. Une sous-ingénierie

1.1 Présentation

Comme nous l'avons vu aux deux chapitres précédents, l'ingénierie sur la dichotomie se découpe en deux sous-ingénieries. Ainsi, après avoir conduit les élèves de Seconde et du cycle terminal scientifique à travailler la compréhension et la maitrise des structures de boucles et des variables informatiques dans le cadre d'une sous-ingénierie (a) portant sur la « *dichotomie discrète* », nous présentons maintenant la sous-ingénierie (b) traitant la « *dichotomie continue* ».

Cette deuxième sous-ingénierie est pour l'essentiel orientée vers l'étude de l'algorithme relativement à un problème de calcul approché d'un zéro d'une fonction et vers la prise de conscience de conditions suffisantes pour son effectivité, et pour l'unicité de solutions. Elle est menée dans les trois classes qui ont expérimenté la sous-ingénierie (a). Pour des raisons institutionnelles, la deuxième partie de cette sous-ingénierie, concernant les conditions de validité de l'algorithme et une ébauche d'une preuve algorithmique du « *Théorème des Valeurs Intermédiaires* » (TVI), va être conduite seulement dans la classe de Terminale.

1.2 Les premières problématiques de la sous-ingénierie

On attend d'un algorithme d'approximation d'un zéro d'une fonction qu'il rende deux valeurs aussi proches que l'on veut, encadrant un réel annulant la fonction (effectivité), et éventuellement qu'il soit garanti qu'il n'existe pas de zéros en dehors de cet encadrement (unicité). Nous nous intéressons particulièrement à la dichotomie qui, dans notre contexte, peut être mise en place dès la Seconde.

Soient a et b deux réels et f une fonction continue et définie sur l'intervalle $[a ; b]$, telle que $f(a)$ et $f(b)$ ne soient pas de même signe. Le processus de dichotomie va alors consister à

[83] Théorème des valeurs intermédiaires

découper l'intervalle [*a* ; *b*] en deux intervalles de « même » amplitude. Si la valeur médiane annule la fonction, l'algorithme termine, sinon on conserve l'intervalle tel que *f* n'a pas le même signe aux deux bornes. On recommence ensuite ce processus en découpant de même l'intervalle conservé, et ainsi de suite. On obtient alors soit un zéro, soit des intervalles emboîtés tels que *f* n'a pas le même signe aux deux bornes et dont les amplitudes forment une suite géométrique de raison 0,5. On considère que l'algorithme est effectif si les intervalles encadrent un zéro (fig. 116).

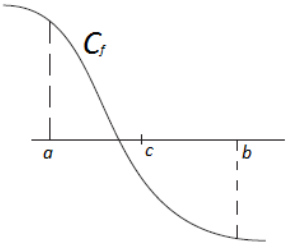


Figure 116 (Représentation graphique d'une fonction continue f sur [a ; b] avec f(a)×f(b)<0

Pour atteindre une précision ε, arbitrairement petite, il suffit d'itérer ce processus *n* fois, où *n* est le plus petit entier tel que $\frac{|a-b|}{2^n} \leq \varepsilon$. On peut démontrer que *n* est le plus petit entier naturel supérieur ou égal à $\log_2\left(\frac{|a-b|}{\varepsilon}\right)$. Cependant, ce résultat n'est pas exploité avec les élèves. Toutefois, il permet de prévoir qu'avec des données couramment utilisées comme une différence *a* – b égale à 1 ou 2 et ε choisi jusqu'à 10^{-10}, le nombre d'itération est au maximum de 34.

Notons que la dichotomie n'appartient pas à la famille dite des méthodes du point fixe. Cette famille est abordée en Terminale, mais impose des conditions supplémentaires sur la fonction. Nous n'abordons pas cet aspect dans cette sous-ingénierie (b).

La première partie de la sous-ingénierie (b), à travers des études de fonctions particulières, vise à ce que les élèves mettent en évidence la définition sur un intervalle et la « continuité » sur cet intervalle initial comme des conditions suffisantes d'effectivité de l'algorithme. Nous visons aussi à ce que les élèves mettent en évidence la « monotonie » sur l'intervalle initial comme une condition suffisante d'unicité du zéro cherché. Ces problématiques renvoient respectivement à une version restreinte du TVI et de son corollaire, le « *Théorème de la Bijection* » (TDB), au sens vu en Terminale Scientifique (cf. ci-après). La seconde partie de cette sous-ingénierie vise à ce que les élèves reconnaissent les propriétés des suites des bornes des

intervalles emboîtés, et utilisent ces propriétés pour construire une démonstration du TVI.

Afin d'aider à la compréhension de l'arrière-plan mathématique de cette sous-ingénierie, nous rappelons les contenus des deux théorèmes retenus : TVI et (TDB). Dans l'ingénierie, nous nous restreignons au cas $k = 0$ où le TVI est obtenu directement avec l'algorithme. La généralisation à k quelconque est un travail algébrique sans rapport avec notre problématique.

Théorème des Valeurs Intermédiaires (TVI)[84]

On considère une fonction f définie et continue sur un intervalle [a ; b].

Pour tout réel k compris entre f(a) et f(b), il existe au moins un réel c compris entre a et b tel que f(c) = k (Fig.. 117).

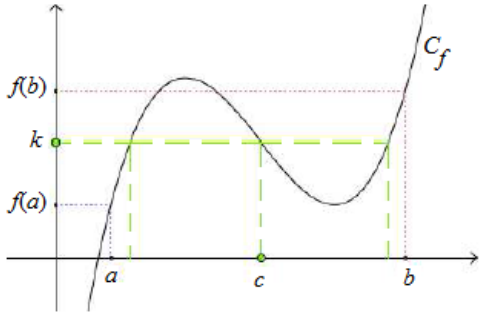


Figure 117 (Illustration du Théorème des Valeurs Intermédiaires)

Théorème de la Bijection (TDB)

On considère une fonction f définie, continue et strictement monotone sur un intervalle [a ; b].

Pour tout réel k compris entre f(a) et f(b), il existe un unique réel c compris entre a et b tel que f(c) = k (fig. 2b).

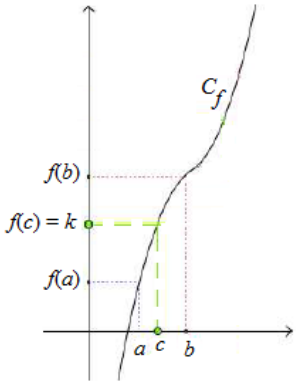


Figure 118 (Illustration du Théorème de la Bijection)

1.3 Arrière-plan mathématique et institutionnel

A ce stade de notre présentation des objectifs de la sous-ingénierie, nous tenons à rappeler quelques points théoriques autour du TVI et de la *méthode de dichotomie*. Nous considérons une fonction f définie d'une partie de **R** dans **R**. Le TVI fait intervenir des éléments issus de la Topologie : **(I)** la continuité de la fonction : si f est une fonction continue et définie à la limite l d'une suite convergente (U_n) (resp. (V_n)) dont les éléments appartiennent à son ensemble de définition et tels que $f(U_n) < 0$ pour tout n (resp. $f(V_n) > 0$) alors $f(l)=0$; **(II)** le fait que les

[84] Dans le cadre de notre ingénierie, nous faisons le choix de centrer notre travail sur le TVI.

connexes de **R** sont les intervalles, et que par conséquent l'intervalle de départ est un connexe (CN) ; en effet, si *f* est définie sur un connexe, cela garantit que si elle est définie en deux points *a* et *b*, elle est aussi définie en $\frac{a+b}{2}$ et si de plus *f* est continue cela garantit qu'elle est définie à la limite de toute suite convergente dont les éléments appartiennent à son ensemble de définition ; **(III)** la *complétude* de **R** qui peut s'exprimer avec les énoncés équivalents suivants : **(a)** « Toute partie majorée non vide admet une borne supérieure » ; **(b)** « Toutes les suites de Cauchy sont convergentes » ; **(c)** Le « Théorème de la convergence monotone » : « Toute suite croissante majorée est convergente » ; **(d)** « Toutes les paires de suites adjacentes convergent vers une même limite ». La propriété (d) permet de montrer que les suites adjacentes formées des bornes inférieures (resp. supérieures) des intervalles emboîtés, si elles existent, convergent vers une limite commune. Les conditions de continuité et de connexité garantissent l'existence des suites, et le fait que leur limite commune est un zéro de la fonction.

Nous considérons chacun de ces éléments du point de vue de l'enseignement au lycée. En effet, pour (I), la continuité s'appuie sur la prise de conscience de la continuité en un point ou sur un intervalle de fonctions de référence et sur des règles algébriques de conservation ; une définition rigoureuse, et utile pour la démonstration du TVI serait : « pour toute suite (U_n) d'éléments de l'ensemble de définition de *f*, convergeant vers un élément *l* de cet ensemble, la suite $f(U_n)$ converge vers $f(l)$ » ; pour (II), la connexité n'est pas abordée en classe scientifique, par contre la notion d'intervalle est supposée connue et la propriété de la connexité de l'intervalle est implicite ; le TVI n'est considéré que sur des intervalles fermés bornés ; pour (III), conformément au programme seul l'énoncé (c) « Théorème de la convergence monotone » est connue des élèves de Terminale Scientifique. En revanche, les élèves n'ont aucune connaissance particulière sur les suites adjacentes. Par conséquent, la démonstration du TVI ne peut pas être une application directe du cours. Les élèves doivent étudier individuellement chacune des deux suites produites par la *méthode de dichotomie* et montrer leur convergence et l'égalité des limites.

2. Première partie de la sous-ingénierie « dichotomie continue »

2.1 Les objectifs et les attentes

2.1.1 Les objectifs

Comme nous l'avons dit, la première partie de la sous-ingénierie vise à ce que les élèves prennent connaissance de l'algorithme et mettent en évidence la définition sur un intervalle et la continuité sur cet intervalle ainsi que la monotonie comme des conditions suffisantes d'effectivité de l'*algorithme de dichotomie*.

Cette première partie s'adresse au trois niveaux, incluant la Terminale scientifique. En Seconde et en Première, elle va mettre en jeu la compréhension de la *méthode de dichotomie* « continue » comme moyen d'obtenir une approximation d'un antécédent par une fonction donnée et de prendre conscience du « Théorème des Valeurs Intermédiaires » et des conditions nécessaires dans ce théorème. Le théorème n'est pas connu des élèves de Seconde et de Première Scientifique. En Terminale, le TVI est déjà connu des élèves et il s'agit plutôt de préparer une preuve de ce théorème (cf. la seconde partie de la sous-ingénierie).

Du fait que l'*algorithme de dichotomie* interagit fortement dans l'ingénierie avec des contenus mathématiques, elle va nous permettre d'étudier les interactions entre différents ETA et ETM relatifs au domaine de l'analyse.

2.1.2 Les attentes

Un travail sur l'*algorithme de dichotomie* va permettre aux élèves d'obtenir une suite d'intervalles emboités et à bornes rationnelles, avec des conditions de signe aux bornes des intervalles, jusqu'à une amplitude arbitrairement petite. L'étude du corps de boucle va permettre aux élèves de construire la suite des bornes et une condition d'arrêt faisant intervenir l'amplitude du dernier intervalle. En effet, afin de faire le lien avec la sous-ingénierie précédente sur le cas de la dichotomie « discrète », les élèves vont étudier les différences sur les conditions d'arrêt dans une structure de boucle. Nous attendons ainsi que les élèves prennent conscience que suivant que la situation soit « discrète » ou « continue », la condition d'arrêt va être ramenée à une égalité à un nombre donné ou à une comparaison à un nombre arbitraire.

Nous attendons aussi des élèves qu'ils puissent observer des cas où, bien que l'algorithme

termine, il n'encadre pas un zéro, ou il n'en encadre qu'un seul alors qu'il en existe plusieurs.

Cette première partie de la sous-ingénierie se place dans le prolongement de la sous-ingénierie « dichotomie discrète ». Au cours de cette partie sur la « dichotomie continue », les élèves sont à nouveau confrontés à des éléments de nature informatique comme la conception d'une structure pour l'algorithme et la mise en place de variables. Du fait que l'algorithme opère sur une fonction mathématique, il peut être intéressant que les élèves utilisent une fonctionnalité de l'environnement de programmation permettant le calcul de valeurs avec le formalisme fonctionnel (f(…)), faisant ainsi un lien entre fonctions mathématiques et fonctions informatiques.

2.2 Les choix pour chaque phase

Le travail fait lors de la sous-ingénierie « dichotomie discrète » a permis de favoriser la dévolution laissée aux élèves pour la conception d'algorithmes papiers-crayons. Nous choisissons aussi de favoriser cette dévolution chez les élèves tout au long de cette seconde sous-ingénierie.

Nous décidons de ne pas aborder la comparaison de différentes méthodes de recherche d'une approximation d'un zéro d'une fonction du point de vue de l'efficacité. En effet, cette question ne peut pas être réellement traitée au niveau du lycée.

Le découpage en phases vise à faire progresser les élèves en partant de la pratique habituelle où l'algorithme est vu comme un « outil » dans une perspective d'« application numérique » vers un questionnement sur l'algorithme comme « objet » dans une perspective d'« analyse numérique » (cf. chapitre 1 de cette seconde partie).

2.2.1 Phase 1 : L'algorithme comme « application numérique » autour des zéros d'une fonction

Au début de cette phase, nous faisons le choix de rappeler aux élèves la définition concernant le concept de zéro d'une fonction :

> « *Un zéro (ou un point d'annulation) d'une fonction est une valeur en laquelle cette fonction et définie et s'annule. Autrement dit, il s'agit d'un antécédent de la valeur zéro* »,

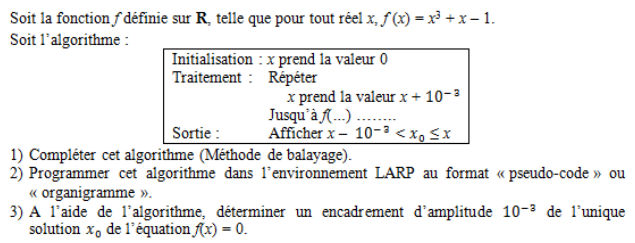
afin d'anticiper les difficultés des élèves sur le vocabulaire mathématique, en particulier en début de Seconde. Ce choix est aussi motivé par le fait qu'en Seconde les élèves savent bien rechercher un (ou des) antécédent(s) d'une valeur numérique par lecture graphique pour une fonction donnée, mais ils ne savent pas nécessairement faire le lien entre ce concept d'antécédent et la recherche des solutions d'une équation, car les registres mis en jeu ne sont pas exactement les mêmes. Nous faisons aussi le choix de leur rappeler que les zéros d'une fonction d'une variable réelle correspondent aux abscisses des points d'intersection de la courbe représentative de la fonction, dans un plan muni d'un repère, avec l'axe des abscisses. Nous leur rappelons aussi que déterminer les zéros d'une fonction f revient à résoudre l'équation $f(x) = 0$.

Lors de cette phase, nous choisissons, quel que soit le niveau scolaire, de tenir compte plus particulièrement des attentes du programme de Seconde. Ainsi, sur le plan de l'activité des élèves, nous choisissons de leur faire : (1) pratiquer une activité expérimentale et une analyse critique d'un résultat et d'une démarche, (2) utiliser un environnement informatique (ici *AlgoBox*) adapté à la résolution d'un problème donné dans un champ particulier des mathématiques. Sur le plan des tâches, comme nous l'avons vu, programmes et manuels se placent dans une perspective d'« application numérique » : il est demandé aux élèves de justifier avec les techniques connues à ce niveau (résolution graphique ou algébrique en Seconde, variations de la fonction en Première, TVI en Terminale) l'existence d'un zéro unique sur un intervalle donné ; ensuite l'algorithme est présenté comme un moyen d'obtenir un encadrement de ce zéro. Dans cette phase, nous nous conformons à cette demande.

De plus, cette 1ère phase s'inscrit dans une perspective des phases 3 et 4 (cf. page 8). En effet, au cours de ces deux phases, les élèves vont devoir étudier l'importance de connaitre quelles conditions nécessaire et suffisante devront être vérifiées par la fonction afin que la mise en place de la *méthode de dichotomie* ait du sens lors de son utilisation. Ainsi, les élèves vont se questionner sur l'interprétation des résultats obtenus quand on ne connait pas certaines propriétés de la fonction, comme être ou pas définie en tout point de l'intervalle I étudié, continue ou pas en tout point de I, (strictement) monotone ou pas sur I,…

2.2.2 Phase 2: Travail en algorithmique questionnant le choix d'une condition de sortie ; le balayage

Dans cette phase, nous souhaitons questionner les élèves sur ce qui permet, dans un algorithme, d'obtenir un encadrement de plus en plus « fin » d'un zéro d'une fonction sur un intervalle donné. Il s'agit pour eux d'identifier précisément les éléments de l'itération : condition d'arrêt (ou de continuation), ainsi que les variables itératives, leur initialisation et la mise à jour dans la boucle. Ainsi pour cette phase, nous faisons le choix de faire travailler les élèves, quel que soit le niveau scolaire, à partir d'un algorithme « à trou », écrit en langage « textuel » (énoncé de la tâche (Fig. 119) et son corrigé (Fig. 120)), sur une méthode de détermination d'un encadrement d'amplitude 10^{-p} d'une solution d'une équation de la forme $f(x) = 0$, où f est la fonction utilisée lors de la première phase, en leur demandant de rédiger une condition d'arrêt adéquate.



Soit la fonction f définie sur **R**, telle que pour tout réel x, $f(x) = x^3 + x - 1$.
Soit l'algorithme :

Initialisation :	x prend la valeur 0
Traitement :	Répéter
	x prend la valeur $x + 10^{-3}$
	Jusqu'à $f(...)$ .....
Sortie :	Afficher $x - 10^{-3} < x_0 \leq x$

1) Compléter cet algorithme (Méthode de balayage).
2) Programmer cet algorithme dans l'environnement LARP au format « pseudo-code » ou « organigramme ».
3) A l'aide de l'algorithme, déterminer un encadrement d'amplitude 10^{-3} de l'unique solution x_0 de l'équation $f(x) = 0$.

Figure 119 (Enoncé autour d'une fonction polynomiale de degré 3)

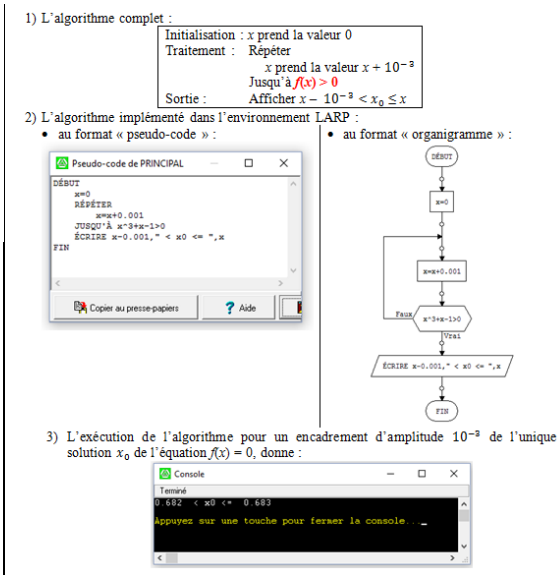


Figure 120 (Corrigé de l'exercice proposé à la figure 119)

Dans cet algorithme, du fait du choix de la structure « TantQue », la variable de boucle x va évoluer avant le test d'arrêt. Dans ce test, la condition doit donc porter sur $f(x)$ et la valeur de x après l'arrêt est la borne droite de l'encadrement ce qui justifie l'instruction de sortie. De plus, comme la solution de l'équation n'est pas un nombre rationnel, nous faisons le choix de prendre comme test d'arrêt une inégalité stricte.

Nous choisissons de ne pas conduire ce travail sur l'*algorithme de dichotomie*. D'une part, cet algorithme a été utilisé à la phase précédente et les élèves pourraient recopier la condition

sans nécessairement l'analyser et d'autre part, cette analyse serait difficile à conduire, du fait de la complexité du corps de boucle (deux alternatives) particulièrement par les élèves de Seconde débutants en informatique. Nous choisissons alors un algorithme faisant appel à une boucle dont le corps ne comporte pas d'alternative : la *méthode de balayage* avec une structure de type « Répéter … Jusqu'à » qui met l'accent sur une condition d'arrêt (plutôt que de continuation).

Nous faisons le choix que cet algorithme soit implémenté dans l'environnement *LARP* afin que les élèves puissent vérifier l'exécution correcte de leur algorithme. De ce point de vue, nous aurions pu laisser les élèves libres d'utiliser *AlgoBox* ou *LARP*. Dans ce cas, il aurait été probable qu'une grande majorité se serait dirigée vers *AlgoBox* utilisé à la phase précédente, ce qui aurait été une difficulté par rapport à un de nos objectifs de diversifier les environnements d'expression et d'exécution des algorithmes. Notons que ceci permet aux élèves de choisir entre une représentation « spatiale » avec l'organigramme et une « textuelle » avec le pseudo-code, mais aussi de passer de l'une à l'autre des représentations, et de choisir parmi les structures « Répéter… Jusqu'à » et « TantQue ». Notons aussi que l'appel à la fonction avec le formalisme fonctionnel passe dans *LARP* par une fonctionnalité de sous-programme non connue des élèves.

2.2.3 Phases 3 et 4 : condition suffisante pour l'unicité, condition suffisante pour l'effectivité ; dichotomie avec une fonction cachée

Pour chacune de ces phases, nous choisissons de donner aux élèves un algorithme de dichotomie « continu » écrit sous une forme « textuelle » dans l'environnement *AlgoBox*, de telle manière que l'algorithme soit exécutable avec une fonction appelée par un identificateur F1, sans que son expression soit visible aux élèves (fonction cachée) (cf. la présentation de l'environnement *AlgoBox*). En revanche, ceci nous conduit à ne faire travailler les élèves que sur des algorithmes bâtis avec une structure de type « Tant que … faire ».

Dans ces deux phases, nous choisissons de sortir du cadre institutionnel où, comme nous avons pu le voir dans la plupart des manuels scolaires et des documents ressources étudiés (cf. Chapitre 1 de cette seconde partie) sur des tâches en lien avec l'*algorithme de dichotomie* « continu », les exercices proposés sont de type « application numérique », entrant dans le cas « idéal » où deux conditions seraient remplies : la fonction utilisée dans l'algorithme

est bien définie sur l'intervalle choisi et elle n'admet qu'un seul zéro sur cet intervalle. En effet, notre choix ici, est de favoriser la prise de conscience par les élèves de ces conditions et par conséquent au moins une de ces deux conditions n'est pas vérifiée (sans que l'élève le sache au départ de l'activité), ce qui renvoie la tâche à une situation de type « analyse numérique ».

Dans la phase 3 (Fig. 121), les élèves ont à travailler sur une fonction « cachée » dont ils ignorent son expression ($x \mapsto x^3 - 3\,x^2 + 2$, avec $x \in [0\,;\,10]$) et sa représentation graphique. Les élèves ignorent ainsi que cette fonction admet plusieurs zéros sur l'intervalle étudié. Dans la phase 4 (Fig. 122), les élèves ont de nouveau à travailler sur une fonction « cachée » dont ils ignorent son expression ($x \mapsto \frac{1}{x^2-2}$, avec $x \in [1\,;\,2]$) et sa représentation graphique. Les propriétés de cette fonction étant inconnue des élèves, ils ne savent pas que cette fonction n'est pas définie $\sqrt{2}$, et qu'elle n'a pas le même signe à droite et à gauche de $\sqrt{2}$.

Exercice 1

Soit l'algorithme suivant, où F1 est une fonction dont on ne connait ni son expression, ni sa représentation graphique, ni les propriétés qu'elle vérifie.

```
1  VARIABLES
2    a EST_DU_TYPE NOMBRE
3    b EST_DU_TYPE NOMBRE
4    e EST_DU_TYPE NOMBRE
5  DEBUT_ALGORITHME
6    a PREND_LA_VALEUR 0
7    b PREND_LA_VALEUR 10
8    e PREND_LA_VALEUR 0.0001
9    TANT_QUE (b-a>e) FAIRE
10     DEBUT_TANT_QUE
11     SI (F1(b)*F1((a+b)/2)>0) ALORS
12       DEBUT_SI
13       b PREND_LA_VALEUR (a+b)/2
14       FIN_SI
15       SINON
16         DEBUT_SINON
17         a PREND_LA_VALEUR (a+b)/2
18         FIN_SINON
19     FIN_TANT_QUE
20   AFFICHER a
21   AFFICHER b
22 FIN_ALGORITHME
```

☛ **Ouvrir le fichier AlgoBox « question1_p2.alg » sans jamais ouvrir l'onglet « Utiliser une fonction numérique ».**

1) Décrire en quelques mots ce que fait l'algorithme donné ci-dessus.

2) a) Ouvrir le fichier « question1_p2.alg » et exécuter le programme.

b) Quelles sont les valeurs affichées lors de son exécution ?

c) Que pensez-vous des résultats affichés après l'exécution de l'algorithme ?

3) En utilisant l'onglet « Dessiner dans un repère » du logiciel AlgoBox, compléter l'algorithme de telle façon que le programme trace en plus la courbe de la fonction F1 sur l'intervalle de départ, dans un plan muni d'un repère orthogonale.

Que pouvez-vous en conclure ?

Figure 121 (Cas d'une fonction « cachée » admettant plusieurs zéros sur l'intervalle étudié)

Exercice 2

Soit l'algorithme suivant, où F1 est une fonction dont on ne connait ni son expressio représentation graphique, ni les propriétés qu'elle vérifie. :

```
1  VARIABLES
2    a EST_DU_TYPE NOMBRE
3    b EST_DU_TYPE NOMBRE
4    e EST_DU_TYPE NOMBRE
5  DEBUT_ALGORITHME
6    a PREND_LA_VALEUR 1
7    b PREND_LA_VALEUR 2
8    e PREND_LA_VALEUR 0.000000001
9    TANT_QUE (b-a>e) FAIRE
10     DEBUT_TANT_QUE
11     AFFICHERCALCUL F1(a)
12     AFFICHER " ; "
13     AFFICHERCALCUL F1(b)
14     SI (F1((a+b)/2)>0) ALORS
15       DEBUT_SI
16       b PREND_LA_VALEUR (a+b)/2
17       FIN_SI
18       SINON
19         DEBUT_SINON
20         a PREND_LA_VALEUR (a+b)/2
21         FIN_SINON
22     FIN_TANT_QUE
23   AFFICHER a
24   AFFICHER b
25 FIN_ALGORITHME
```

☛ **Ouvrir le fichier *AlgoBox* « dichoun_x2m2 » sans jamais ouvrir l'onglet « Utili fonction numérique ».**

1) Décrire en quelques mots ce que fait l'algorithme donné ci-dessus.

2) a) Ouvrir le fichier « dichoun_x2m2 » et exécuter le programme.

b) Quelles sont les valeurs affichées lors de son exécution ?

c) Que pensez-vous des résultats affichés après l'exécution de l'algorithme ?

3) En utilisant l'onglet « Dessiner dans un repère » du logiciel AlgoBox, co l'algorithme de telle façon que le programme trace en plus la courbe de la fonction l'intervalle [- 3 ; 3], dans un plan muni d'un repère orthonormé.

Que pouvez-vous en conclure ?

Figure 122 (Cas d'une fonction « cachée » qui n'est pas définie en une valeur de l'intervalle étudié)

- **Quelques remarques sur les tests de sortie des instructions conditionnelles mises en place dans les exemples proposés ci-dessus concernant les méthodes de balayage et de dichotomie.**

Nous rappelons d'abord que dans le cas de la *méthode de dichotomie*, nous prenons en général comme condition de test, mis en place dans l'instruction conditionnelle, le signe du produit « F1(b)*F((a+b)/2) > 0 » (avec $a < b$) (cf. figure 121). En effet, au niveau du lycée, cette méthode est principalement utilisée sur une fonction vérifiant des propriétés connues, sur la continuité et la monotonie dans l'intervalle [a ; b], où se trouve l'unique solution cherchée. En revanche, dans le cas de la *méthode balayage*, le produit n'est pas nécessairement pris en compte. En effet, par exemple, dans le cas de la fonction étudiée de la figure 119, nous savons que l'image de 0 par la fonction ($x \rightarrow x^3 + x - 1$) est une valeur négative et par conséquent comme la fonction est aussi strictement croissante sur **R**, il n'est pas utile de passer par une condition du type « produit ».

Afin de compléter cette présentation, nous souhaitons revenir sur certains choix faits d'un point de vue algorithmique pour chacun des deux exercices où les élèves travaillent sur des *algorithmes de dichotomie* « continue » associés à des fonctions « cachées » (fig. 121 et 122). En effet, nous faisons le choix de prendre comme condition conditionnelle le signe du produit « F1(b)*F1((a+b)/2) » dans le cas de la figure 121 et le signe de « F1((a+b)/2) » dans le cas de la figure 122. Dans le premier cas, la représentation graphique (Fig. 123) de la fonction, non connue des élèves, nous montre que sur l'intervalle étudié [0 ; 10], la fonction admet plusieurs zéros. Ainsi, les élèves sont confrontés à la problématique de la monotonie qui doit faire l'objet d'une étude autour du concept « condition suffisante » pour travailler sur l'unicité d'une solution. Dans le deuxième cas, la représentation graphique (Fig. 124) de la fonction, qui est aussi non connue des élèves, nous montre que sur l'intervalle étudié [1;2], la fonction n'admet pas un zéro, bien qu'il y ait un changement de signe. Ainsi, nous pensons qu les élèves peuvent prendre connaissance que la condition que la fonction soit « définie et continue » est une « condition nécessaire » pour qu'une utilisation de l'algorithme de dichotomie « continue » corresponde à une situation de résolution d'équation ou de recherche de zéro(s).

Figure 123 (Représentation graphique de la fonction « cachée » admettant plusieurs zéros sur l'intervalle étudié)

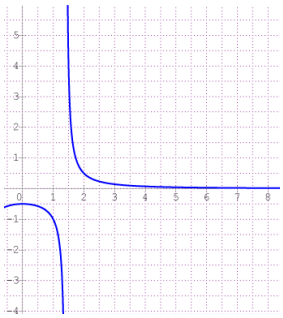


Figure 124 (Représentation graphique de la fonction « cachée » qui n'est pas définie en une valeur de l'intervalle étudié)

Ainsi, ce travail sur ces deux fonctions et une utilisation de l'algorithme de dichotomie « continue » autour de ces deux fonctions peuvent se situer dans une perspective d'étude des conditions de validité du TVI et de son corollaire qui font l'objet du travail de la deuxième partie de cette sous-ingénierie (cf. 3.).

2.3 Les tâches et le mode de travail

2.3.1 Les tâches

Dans un premier temps, il s'agit pour les élèves d'étendre le travail algorithmique fait dans le cas « discret » au cas « continu » tout en modifiant la condition d'arrêt. Pour cela, nous débutons cette approche dans une perspective d'« application numérique », où les élèves justifient l'existence d'un zéro unique pour une fonction dont l'expression est connue sur un intervalle donné, puis ils mettent en œuvre une méthode d'approximation de ce zéro à l'aide d'un algorithme (la *méthode de balayage*) ne faisant appel qu'à une boucle pour chaque décimale de ce zéro, sans modifier le comportement de cette boucle selon le signe de l'image d'un nombre.

Dans un second temps, après la prise de conscience par les élèves des concepts d'existence et d'unicité d'un zéro d'une fonction sur un intervalle particulier dans le cadre d'une « application numérique », ainsi que la conception et la mise en place d'une condition d'arrêt dans un corps de boucle, les élèves abordent l'aspect « analyse numérique » d'une méthode d'approximation, en travaillant à nouveau sur l'algorithme de dichotomie « continue » sur des questionnements concernant le domaine de validité de l'algorithme. Ainsi, les élèves ont pour tâches de se questionner sur des conditions nécessaires et suffisantes pour que l'algorithme détermine réellement un encadrement (ou une valeur approchée, variable didactique laissée à l'appréciation de l'enseignant responsable de la classe) d'un zéro d'une

fonction sur un intervalle.

Comme nous l'avons dit précédemment, cette première partie de la sous-ingénierie se découpe en phases visant à faire progresser les élèves sur leurs pratiques algorithmiques, en passant d'une perspective d'« application numérique », où l'algorithmique est vu comme un « outil » d'apprentissage, à une perspective d'« analyse numérique », où l'algorithmique est vu comme un « objet » d'apprentissage au sens de Douady (1986).

2.3.1.1. Pour la phase 1

Les élèves doivent justifier l'existence et l'unicité d'une solution d'une équation de la forme $f(x) = 0$, où f est une fonction dont l'expression est connue. Pour cela, une procédure de justification leur est demandée, tenant compte de leurs connaissances en analyse. Puis, à l'aide d'un algorithme représenté par l'enseignant sous forme d'un organigramme (Fig. 125) pour une fonction f générique, ils doivent le transposer en un langage pseudo-code, afin de pouvoir le programmer sur *AlgoBox* avec la fonction f étudiée et la tester pour obtenir un encadrement de la solution avec une amplitude donnée.

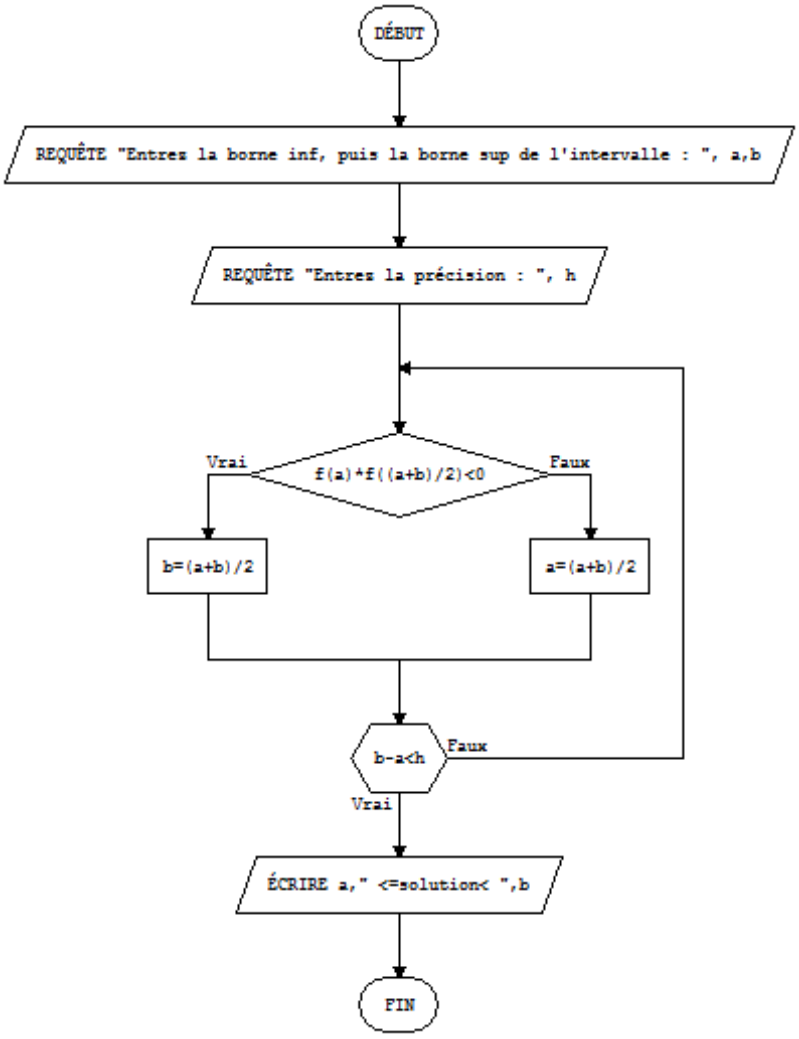


Figure 125 (organigramme pour une fonction f générique)

2.3.1.2. Pour la phase 2

Dans cette phase, l'enseignant distribue aux élèves un deuxième algorithme mettant en pratique la *méthode de balayage*. Cet algorithme comporte un « trou » volontairement laissé par l'enseignant. Les élèves doivent remplir ce « trou », puis programmer cet algorithme dans un environnement informatique (ici *LARP*) afin de le tester pour obtenir un encadrement d'amplitude 10^{-p} de la solution de l'équation $f(x) = 0$ sur un intervalle donné, où f est la fonction utilisée dans la phase précédente.

L'algorithme est écrit avec une structure « Répéter … Jusqu'à ». L'élément manquant dans cet algorithme est la condition de sortie dans la boucle. Les élèves vont donc devoir concevoir cette condition de sortie. Ainsi, la tâche met en jeu la notion d'amplitude d'un intervalle et permet d'aborder la recherche des bornes d'un intervalle encadrant une solution de l'équation $f(x) = 0$, pour une amplitude prédéfinie.

2.3.1.3. Pour les phases 3 et 4

Dans chacune de ces deux phases, un algorithme de dichotomie « continue » est distribué aux élèves en format *AlgoBox*, et ils ont à disposition sur l'ordinateur un fichier *AlgoBox* comprenant cet algorithme et la définition d'une fonction (fonction « cachée ») dont l'expression n'est pas visible aux élèves, mais qui est appelée par l'algorithme par un identificateur F1. La condition de continuation porte sur la distance entre Binf et Bsup (les deux bornes respectives de l'intervalle d'étude), les variables en jeu dans la dichotomie, qui doit être inférieure à une précision donnée. Dans les deux phases la question posée est : « *Est-ce que Binf et Bsup encadrent un zéro unique de la fonction, quelle que soit la précision choisie ?* ». Nous précisons que la consigne est donnée aux élèves de ne pas chercher à connaître l'expression de la fonction. En revanche, nous rappelons au lecteur que les élèves peuvent accéder dans *AlgoBox* à une représentation graphique de la fonction, mais cela ne leur est pas précisé ; nous attendons qu'ils prennent l'initiative de chercher dans le logiciel la fonctionnalité adéquate.

Pour la phase 3, cette fonction « cachée » est continue et admet plusieurs zéro sur l'intervalle du départ ; pour une précision suffisante un seul de ces zéros est en encadré par Binf et Bsup. Pour la phase 4, cette fonction « cachée » est monotone et change de signe sur l'intervalle du départ. De plus, il existe un point de cet intervalle qui le sépare en deux sous-intervalles où la fonction est de signes opposés et donc Binf et Bsup encadrent ce point. Cependant, ce point n'est pas un zéro de la fonction.

Les tâches proposées dans ces deux dernières phases sont directement en lien avec la dichotomie « continue » et les conditions suffisantes d'unicité (phase 3) et d'effectivité (phase 4), par l'étude et l'exécution d'algorithmes pour des fonctions données, dont l'expression, la représentation graphique et les propriétés ne sont pas connues des élèves.

2.3.2 Le mode de travail

Comme pour la sous-ingénierie « dichotomie discrète », les élèves des trois niveaux scolaires sont en salle informatique et travaillent en binôme, de façon qu'il y ait un ordinateur par binôme équipé d'environnements informatiques permettant d'utiliser soit des organigrammes, soit un langage pseudo-code. Quelle que soit la phase, chaque élève est libre d'utiliser les menus « Graphique » et « Tableur » de sa calculatrice. Pour les quatre phases de

cette première partie de cette sous-ingénierie, les binômes ont le choix de l'environnement informatique : *AlgoBox* ou *LARP*. Toutefois, il est demandé aux binômes de rendre leurs travaux sous *AlgoBox* pour les phases 1, 3 et 4, et sous *LARP* pour la phase 2.

Nous prévoyons un temps global de 2 heures pour le travail à faire au cours de cette première partie pour les élèves de Seconde et de Première Scientifique, et de 90 minutes pour les élèves de Terminale Scientifique.

2.4 Les analyses *a priori*

2.4.1 Phase 1 : L'algorithme comme « application numérique » autour des zéros d'une fonction

2.4.1.1. La justification de l'existence et de l'unicité de la solution à l'équation $f(x) = 0$

a) Niveau Seconde

Les élèves ont l'expression d'une fonction polynôme de degré 3 (ici, $f(x) = x^3 + x - 1$) définie et strictement monotone sur **R**. Ils représentent la courbe de cette fonction à l'aide de leur calculatrice. Nous n'attendons pas de difficultés particulières pour cette construction car elle fait partie des tâches classiques que les élèves sont habitués à mettre en place tant en Mathématiques, qu'en Sciences Physiques depuis la fin du collège.

A l'aide de la courbe qu'ils viennent de représenter sur leur calculatrice, les élèves « justifient » à l'aide du graphique que l'équation $f(x) = 0$ admet sur **R** une unique solution en utilisant le mode « Zoom » qu'offre leur calculatrice, si cela s'avère nécessaire. Nous pensons que certains élèves de Seconde vont pouvoir exprimer une certaine difficulté quant à la compréhension de cette consigne. En effet, nous pensons qu'en Seconde, les élèves peuvent ne pas interpréter la résolution d'une équation du type $f(x) = 0$ en termes de repérage sur l'écran du (ou des) point(s) d'intersection de la courbe représentative de la fonction f et de l'axe des abscisses, puis qu'énoncer la (ou les) solution(s) revient à déterminer l'(ou les) abscisse(s) de ce(s) point(s) commun(s), c'est-à-dire à interpréter la question comme la lecture de l'unique (ou des) antécédent(s) de 0 par la fonction f.

Nous pensons aussi qu'une difficulté concernant la lecture d'une approximation de la valeur numérique de la solution vue sur la courbe inscrite sur l'écran de la calculatrice peut

être observée. En effet, nous supposons qu'il est possible que les élèves n'aient pas conscience qu'il faille utiliser le mode « Zoom » qu'offre la calculatrice.

De plus, nous supposons que si les difficultés décrites ci-dessus sont surmontées, les élèves, en observant la courbe de la fonction, doivent pouvoir émettre des conjectures sur des propriétés vérifiées par la fonction *f*. En effet, nous supposons que les élèves vont pouvoir émettre des hypothèses sur la monotonie de la fonction et sur le fait que cette fonction ne s'annule qu'une seule fois sur **R**. Cependant, nous nous attendons, à ce que certains élèves puissent objecter que l'affichage de l'écran peut être trompeur car il ne donnerait qu'une représentation « partielle » de la fonction limitée à un intervalle de **R** et non à tout **R**. Ainsi, on peut s'attendre que parmi ces élèves certains utilisent le mode « Tableur » de leur calculatrice afin qu'ils puissent « confirmer » ce que semble leur renvoyer le graphique en ce qui concerne la monotonie de la fonction.

Nous nous attendons aussi à autre fausse interprétation de la question posée par l'enseignant sur l'étude de l'équation $f(x) = 0$. En effet, certains élèves peuvent interpréter cette question comme le fait de donner une valeur « approchée » de l'abscisse du point d'intersection plutôt que le fait d'émettre une « justification » de l'existence et de l'unicité de la solution à l'équation $f(x) = 0$.

Nous n'attendons pas qu'au niveau de la Seconde que les élèves se posent la question de la continuité ou pas de la fonction. Nous pensons que les élèves vont interpréter cela en terme graphique, c'est-à-dire qu'une fonction polynôme peut se tracer sur **R** sans jamais lever le crayon[85]. Pour cela, nous pensons qu'ils vont se référer au graphique d'une droite représentant une fonction affine, c'est-à-dire une fonction polynôme de degré 1.

Le calcul $f(0)$ et $f(1)$, peut être proposé par les élèves de plusieurs manières, soit par simple lecture graphique, après avoir interpréter ce calcul comme la détermination des images respectives de 0 et de 1, soit en utilisant le mode « Tableur » de leur calculatrice, soit en reprenant l'expression de la fonction et en remplaçant les *x* par les valeurs 0 puis 1. Nous nous attendons à ce que les élèves privilégient la dernière méthode par peur que la lecture graphique leur renvoie des valeurs approchées.

[85]Cette interprétation est proche de celle donnée en Terminale scientifique sur la continuité d'une fonction sur un intervalle (N.D.L.R.)

Enfin, pour la détermination d'un encadrement d'amplitude 1 à bornes entières de la solution, nous attendons que les élèves repèrent sur l'écran de leur calculatrice, le changement de signes de la fonction *f* sur l'intervalle [0 ; 1] et procèdent aux calculs des valeurs de *f*(0) et de *f*(1). Cependant, nous pensons que certains élèves vont mentionner la stricte monotonie de la fonction *f* et se contenter alors, pour justifier leur réponse, d'indiquer le changement de signes entre *f*(0) et *f*(1).

b) Niveaux Première et Terminale Scientifique

La différence par rapport au niveau Seconde va être dans le mode de raisonnement pour l'étude de la monotonie de la fonction. En effet, nous pensons qu'à ces niveaux scolaires, les élèves vont mettre en pratique un raisonnement de type analyse pour justifier la stricte monotonie de la fonction. Nous pensons alors que la difficulté pourra être dans le calcul de la fonction dérivée et dans l'étude du signe de la dérivée. Cependant, avec un calcul de la dérivée, il va s'agir de déterminer le signe de la somme de deux nombres positifs, dont l'un l'est strictement. Par conséquent, la détermination de ce signe va se faire par manipulation d'inégalités.

Nous ne pensons pas observer des difficultés particulières concernant le lien entre le signe de la dérivée et la stricte monotonie.

De plus, nous nous attendons aussi que des élèves de Terminale Scientifique ayant la connaissance des conditions du TVI, ne se contentent pas de justifier l'existence de la solution par une simple étude de changement de signes aux bornes de l'intervalle, mais énoncent aussi le fait que la fonction soit continue sur **R** et par conséquent sur tout intervalle de **R**, pour compléter leur justification de l'existence de la solution.

2.4.1.2. L'aspect algorithmique de la phase 1

Au cours de cette dernière partie de la phase 1 sur une « application numérique », nous ne distinguons pas les trois niveaux scolaires. En effet, les enseignants ayant donné un algorithme représenté sous forme d'un organigramme pour une fonction *f* générique, nous nous attendons à ce que les élèves s'appuient sur le travail fait lors la dernière phase de la sous-ingénierie « dichotomie discrète » pour transposer cet algorithme en un langage pseudo-code, afin de pouvoir le programmer sur *AlgoBox* avec la fonction *f* étudiée. Pour cela, nous pensons que la structure de la boucle proposée par l'organigramme sera reconnue par les

élèves. Cependant, nous nous attendons à ce que certains élèves éprouvent des difficultés sur l'implémentation de la fonction et son appel dans la boucle. En effet, dans le cas de l'algorithme « continu », les données des variables concernées ne vont plus être associées à un nombre ou à une opération élémentaire, mais à des calculs d'images par une fonction.

2.4.2 Phase 2 : Travail en algorithmique questionnant le choix d'une condition de sortie ; le balayage

Nous ne distinguons pas ici les différents niveaux scolaires. Cette phase reprend la même fonction que celle de la phase précédente en supposant acquises les propriétés de cette fonction vues en phase 1. Elle fait intervenir un algorithme de balayage de pas 10^{-p} donné que les élèves ont à compléter par la formulation d'une condition de sortie de boucle, et par l'affichage de l'intervalle d'amplitude 10^{-p} encadrant la solution.

Nous pensons que la formulation de la condition de sortie de boucle peut poser des difficultés chez des élèves débutants. En effet, cela suppose que les élèves prennent conscience de la signification de la variable de boucle, notée x, comme borne inférieure de l'intervalle courant et conçoivent la borne supérieur sous forme $x + 10^{-p}$.

Nous nous attendons aussi à des difficultés chez certains élèves, peut-être plus particulièrement en Seconde, dans le fait qu'ils doivent concevoir comme condition de sortie le changement de signe entre les deux images respectives de x et $x + 10^{-p}$. Ces prises de conscience sont certainement loin d'être triviales pour des élèves de lycée. Connaissant les propriétés de la fonction et de la structure itérative, une condition sur le signe de $f(x + 10^{-p})$ suffit. Cependant, on peut penser que les élèves vont faire intervenir aussi $f(x)$ soit en considérant le signe du produit $f(x) \times f(x + 10^{-p})$, soit sous forme d'une condition composée faisant intervenir les deux signes.

Les élèves peuvent ne pas dépasser la prise de conscience de la signification de la variable de boucle x et être incapable d'exprimer la condition de sortie.

Nous tenons aussi compte que l'on peut rencontrer chez des élèves débutants en algorithmique, une erreur dans la condition d'arrêt, celle-ci s'exprimant alors $|f(x)| < 10^{-p}$. En effet, des élèves débutants en analyse n'ont pas nécessairement pris conscience que, x_0 étant un zéro de la fonction, la condition $|f(c)| \leq \varepsilon$ n'implique pas $|x_0 - c| \leq \varepsilon$.

Les binômes devant travailler dans l'environnement *LARP*, nous nous attendons à ce qu'ils

utilisent plutôt le format organigramme que pseudo-code suite au travail fait pendant la sous-ingénierie « dichotomie discrète » dans cet environnement. Nous nous attendons aussi à des difficultés analogues à celles prévues pour la phase 1 chez certains élèves, en ce qui concerne l'implémentation de la fonction et son appel dans la boucle.

2.4.3 Phase 3 et 4 : Condition suffisante pour l'unicité, condition suffisante pour l'effectivité ; dichotomie avec une fonction « caché »

Dans la phase 3, nous pensons que les élèves vont reconnaitre, dans l'algorithme proposé en format *AlgoBox*, l'approximation d'une solution d'une équation de la forme F1(x) = 0. Nous avons choisi cette fonction avec les propriétés suivantes : polynomiale, définie continue sur un intervalle mais pas monotone.

Nous nous attendons alors à ce que les élèves interprètent la question posée sous l'influence de l'« application numérique » vue dans la phase 1, et que par conséquent ils considèrent que si l'on fait tourner l'algorithme, celui-ci va « résoudre » le problème dans sa totalité et ne pas laisser « échapper » de zéros sur l'intervalle donné. Nous pensons aussi que ceci va être renforcé dans leur esprit, par le contrat didactique en vigueur dans « l'application numérique » telle qu'elle est proposée dans les programmes et dans les manuels : mise en œuvre d'algorithmes d'approximation à la fin de la résolution d'un problème, après qu'il ait été déjà démontré que ce problème admet une solution unique.

Cependant, nous pensons que certains élèves, en particulier en Seconde, vont chercher à tracer le graphe de la fonction « cachée », en explorant les fonctionnalités offertes par *AlgoBox*, afin de vérifier ce qu'ils supposent sur l'unicité de la solution. En effet, nous pensons que le travail fait au cours de la phase 1 de cette sous-ingénierie, où les élèves de Seconde ont exploité la représentation graphique d'une fonction pour émettre des conjectures dans le cadre de la résolution approchée d'une équation, peut inciter certains à douter de la validité d'une « application numérique » de l'algorithme sans qu'aient été vérifiées au préalable des conditions sur la fonction étudiée. Cela peut les amener à une prise de conscience de conditions nécessaires assurant l'unicité des solutions et donc une application adéquate de l'algorithme (s'il existe plusieurs solutions, avoir des encadrements de l'une d'elle seulement n'a pas d'intérêt pratique).

En revanche, nous pensons que dans les deux classes du cycle terminal, cette remise en cause de l'unicité d'une solution sera moins systématique. En effet, nous nous attendons à ce

que les élèves de ces niveaux n'aient pas le réflexe de passer par le registre graphique pour confirmer ou infirmer ce que leur renvoie l'exécution de l'algorithme.

Dans la phase 4, nous pensons que les élèves se trouvant dans une situation proche de la précédente, vont avoir plus facilement un regard critique sur les résultats affichés lors de l'exécution de ce deuxième algorithme. Cependant, on s'attend à ce que les élèves, en particulier en Seconde, habitués par les usages de l'algorithmique en « application numérique » et tenant compte de la phase précédente, pensent que la fonction s'annule en au moins un point de l'intervalle et que l'algorithme donne par conséquent un encadrement d'un de « ces zéros ». Toutefois, nous pensons que des élèves du cycle terminal, peuvent remettre en cause cette conception en faisant afficher les valeurs successives prises par la fonction aux bornes de l'intervalle. Ces valeurs ne se rapprochent pas de zéro. Certains élèves peuvent aussi envisager de faire afficher la représentation graphique de la fonction « cachée » à l'aide d'une fonctionnalité de l'environnement informatique utilisé comme nous l'avons vu pour la phase 3. Ceci peut ainsi les aider à cette remise en cause. Cela doit les amener à une prise de conscience de conditions suffisantes à l'existence d'un zéro et à l'effectivité de l'algorithme. En particulier en Terminale, les élèves peuvent faire le lien avec les conditions suffisantes énoncées pour le TVI.

2.5 Mise en place de l'expérimentation, déroulements et analyses *a posteriori*

2.5.1 Les classes, les supports d'observations

Cette première partie de la sous-ingénierie « dichotomie continue » est expérimentée une semaine après la sous-ingénierie « dichotomie discrète » avec les mêmes élèves et enseignants de nos trois classes participant à notre expérimentation. Elle se fait en présence du chercheur. Nous rappelons que pour la classe de Terminale, l'enseignant et le chercheur sont une seule et unique personne.

Pour chaque classe, les élèves sont en salle informatique et travaillent en binôme, de façon qu'il y ait un ordinateur par binôme équipé d'environnements informatiques permettant d'utiliser soit des organigrammes, soit un langage pseudo-code.

Tout au long de cette première partie de la « dichotomie continue », chaque élève est libre d'utiliser les menus « Graphique » et « Tableur » de sa calculatrice. De plus, pendant

l'intégralité de cette première partie, les binômes ont le choix d'utiliser deux environnements informatiques : *AlgoBox* ou *LARP*. Cependant, il leur est bien précisé que suivant les phases, ils doivent rendre leurs travaux finaux sous tel ou tel environnement informatique. Ainsi, nous rappelons que pour les phases 1, 3 et 4, le retour des algorithmes doit se faire sous *AlgoBox*, pour la phase 2, sous *LARP*.

Pour les élèves de Seconde et de Première Scientifique, l'expérimentation est prévue durer 30 minutes par phase, soit deux heures en tout. L'ensemble des phases est fait sur deux séances de travaux dirigés d'une heure se succédant sur une même après-midi avec une pause de 10 minutes entre les deux séances.

Pour les élèves de Terminale Scientifique, l'expérimentation est prévue durer 15 minutes pour la phase 1 et 25 minutes pour chacune des autres phases, soit un total de 90 minutes. Avec l'accord des élèves, l'ensemble de ces phases doit se faire sur une seule séance de travaux dirigés sans qu'il y ait de pause.

L'analyse *a posteriori* est construite à partir de plusieurs supports : les enregistrements vidéo et audio des séances, les observations du travail des élèves et des réactions de leur enseignant. Ces divers supports favorisent l'étude et l'analyse des algorithmes produits lors des deux séances par les élèves. A la demande du chercheur, les élèves rendent à leur professeur un écrit avec un enregistrement sur clé USB de leurs algorithmes implémentés sur l'ordinateur.

2.5.2 Phase 1 : L'algorithme comme « application numérique » autour des zéros d'une fonction

2.5.2.1. Le déroulement

a) La justification de l'existence et de l'unicité de la solution à l'équation f(x) = 0

(i) Niveau seconde

Le temps prévu lors de la préparation de cette phase est globalement respecté. Les élèves utilisent leur calculatrice pour représenter la courbe de la fonction. Une fois la représentation graphique faite, les élèves utilisent le mode « Trace » pour suivre les valeurs des abscisses des points de la courbe. Quelques élèves ont des difficultés dues à un cadrage antérieur de leur machine, où le point d'intersection n'est pas visible, contrairement au cadrage par défaut de

la calculatrice. Après avoir obtenu un cadrage correct, les élèves déterminent alors par une simple lecture graphique l'encadrement du zéro par des entiers. Seulement 15% des binômes proposent un début de justification précisant la « monotonicité » de la fonction, les autres se contentent d'indiquer le changement de signes, en relation avec la position du graphique par rapport à l'axe des abscisses. Un binôme utilise aussi le mode « Solveur » de la calculatrice pour déterminer une valeur approchée de la solution et donne ensuite comme encadrement les entiers immédiatement inférieur et supérieur à cette valeur.

(ii) Niveaux Première et Terminale Scientifique

Le temps global prévu pour chacun des deux niveaux lors de la préparation de cette phase est respecté.

En Première

80% des binômes passent par le calcul de la dérivée et utilisent le signe de la dérivée pour déterminer le sens de variation de la fonction. Parmi ceux-là, 72% font un calcul exact de la dérivée, ainsi qu'une étude correcte du signe de la dérivée obtenue. Les autres, soit 8% des binômes ayant procédé à un calcul de dérivée, bien qu'ayant fait une erreur dans le calcul de la dérivée, obtiennent aussi un sens de variation correct de la fonction sur **R**. En général, les élèves dressent alors un tableau de variation classique où apparait le signe de la dérivée et le sens de variation de la fonction sur **R**. Ce tableau de variation n'est pas suffisant pour la question car, même dans le cas où ils portent 0 et 1 dans la ligne des x, ils ne mettent pas le signe de l'image.

15% des autres binômes mettent en place un raisonnement se basant sur la représentation graphique de la fonction faite à l'aide de la calculatrice. Cependant, l'ensemble de ces binômes ne se contente pas de cette représentation graphique pour répondre au problème posé. En effet, ils observent, par élimination des monômes en *x*, que l'image de 0 est négative et calculent alors la valeur de $f(1)$, pour constater que celle-ci est positive. Pour justifier l'unicité, ils partent du fait que la fonction semble être strictement monotone sur **R**, d'après le graphique.

Les 5% restant déterminent le sens de variation de celle-ci en utilisant le fait qu'elle est la somme de deux fonctions strictement croissantes sur **R**, donc elle-même doit être aussi

croissante sur **R**. Ils ne considèrent pas les signes en 0 et 1.

En Terminale

94% des binômes utilisent le calcul de la dérivée de la fonction pour répondre au problème. Tous ces binômes déterminent correctement le sens de variation de la fonction après étude du signe de cette dérivée.

Les autres 6% donnent le sens de variation en se référant à une connaissance ancienne vue en Première, utilisant une décomposition de la fonction comme somme de deux fonctions croissantes sur **R**.

85% des binômes représentent aussi la courbe de la fonction sur leur calculatrice en prenant le cadrage par défaut de leur machine. La détermination des bornes entières de l'intervalle donnant un encadrement d'amplitude 1 de la solution, se fait alors par une simple lecture graphique en utilisant le mode « Trace » de leur machine. Ensuite, ces élèves vérifient manuellement les valeurs des images de ces deux bornes et observent que ces calculs donnent bien un changement de signes pour les images, puis ils concluent en précisant bien le fait que l'unicité est due à la stricte monotonie de la fonction sur **R**.

Les 15% qui ne tracent pas la courbe, utilisent le mode « Tableur » de leur calculatrice avec un pas de 1, par référence à l'amplitude de l'intervalle cherché et en prenant comme valeur de départ – 5 et valeur finale 5, sans justification particulière du choix de ces deux valeurs. En prenant connaissance des résultats des valeurs de $f(x)$ fournis par la calculatrice pour chaque entier compris entre – 5 et 5, ils en déduisent ensuite que la solution cherchée est comprise entre 0 et 1. Dans leur conclusion, ils tiennent aussi compte de la stricte monotonie de la fonction pour préciser que cette solution est unique.

b) L'« application numérique » avec AlgoBox

(i) Niveau Seconde

L'enseignant ayant distribué un algorithme représenté sous forme d'organigramme « papier-crayon » pour une fonction f générique, l'ensemble des binômes s'appuie alors sur le travail fait lors la dernière phase de la sous-ingénierie « dichotomie discrète » et reconnait la structure de boucle (« Répéter…Jusqu'à ») de l'organigramme.

70% des binômes n'ont pas de difficultés particulières à transposer l'algorithme écrit sous forme « spatiale » en un algorithme écrit en langage pseudo-code respectant la syntaxe de l'environnement *AlgoBox* (Fig. 126). Parmi ceux-ci, 5% ont recours à l'expression de la fonction chaque fois qu'elle est appelée par le programme (Fig. 127), et les autres utilisent la possibilité de stocker l'expression de la fonction étudiée dans l'onglet « Utiliser une fonction numérique », de façon à y faire appel dans l'algorithme par l'identificateur F1. Près de 15% d'entre eux font ainsi appel à F1 sans toutefois préciser dans l'onglet correspondant l'expression de la fonction. De sa propre initiative, l'enseignant intervient alors auprès de ces binômes, avant qu'ils ne testent l'algorithme, pour leur indiquer qu'ils doivent aussi écrire l'expression de la fonction dans la partie correspondante.

Code de l'algorithme

```
1   VARIABLES
2     a EST_DU_TYPE NOMBRE
3     b EST_DU_TYPE NOMBRE
4     h EST_DU_TYPE NOMBRE
5   DEBUT_ALGORITHME
6     AFFICHER "Borne inf puis la borne sup de l'intervalle : "
7     LIRE a
8     LIRE b
9     AFFICHER "Entrez la précision : "
10    LIRE h
11    TANT_QUE (b-a>=h) FAIRE
12      DEBUT_TANT_QUE
13      SI (F1(a)*F1((a+b)/2)<0) ALORS
14        DEBUT_SI
15        b PREND_LA_VALEUR (a+b)/2
16        FIN_SI
17        SINON
18          DEBUT_SINON
19          a PREND_LA_VALEUR (a+b)/2
20          FIN_SINON
21      FIN_TANT_QUE
22    AFFICHER a
23    AFFICHER " <x0 < "
24    AFFICHER b
25  FIN_ALGORITHME
26
27 Fonction numérique utilisée :
28 F1(x)=pow(x,3)+x-1
```

Figure 126 (Algorithme sous AlgoBox avec l'utilisation d'un identicateur F1 pour la fonction)

```
VARIABLES
  a EST_DU_TYPE NOMBRE
  b EST_DU_TYPE NOMBRE
  h EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  AFFICHER "Entrer les bornes inf et sup de l'intervalle : "
  LIRE a
  LIRE b
  AFFICHER "Entrer la précision : "
  LIRE h
  TANT_QUE (b-a>=h) FAIRE
    DEBUT_TANT_QUE
    SI ((pow(a,3)+a-1)*(pow((a+b)/2,3)+(a+b)/2-1)<0) ALORS
      DEBUT_SI
      b PREND_LA_VALEUR (a+b)/2
      FIN_SI
      SINON
        DEBUT_SINON
        a PREND_LA_VALEUR (a+b)/2
        FIN_SINON
    FIN_TANT_QUE
  AFFICHER a
  AFFICHER " <= solution < "
  AFFICHER b
FIN_ALGORITHME
```

Figure 127 (Algorithme sous AlgoBox avec recours à l'expression de la fonction chaque fois qu'elle est appelée pat le programme)

10% des binômes rendent un algorithme incorrect. En effet, bien qu'ils reconnaissent la structure de boucle de l'organigramme et qu'ils sachent la transcrire en langage pseudo-code afin de l'implémenter dans l'environnement *AlgoBox*, ils éprouvent des difficultés sur l'implémentation de la fonction. En effet, ils présentent la fonction comme une variable de type « chaîne de caractère », et ainsi dans la liste des variables déclarées en début de programme, nous trouvons la variable : « f EST DU TYPE CHAINE » (Fig. 128).

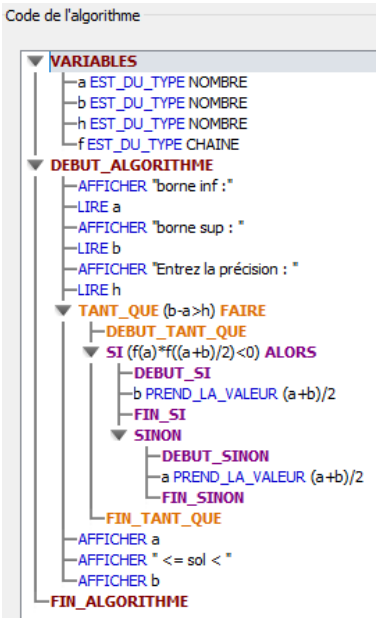


Figure 128 (Algorithme où la fonction est vue comme une variable de type « chaîne de caractère »)

20% des binômes n'écrivent pas d'algorithme. En fonction des résultats recueillis, nous ne pouvons pas réellement connaitre la cause de cette absence d'écriture et d'implémentation d'un algorithme dans l'environnement informatique. En effet, à l'écoute des enregistrements nous pouvons penser que, lors de cette séance, certains binômes n'ont pas été très motivés par le travail demandé.

(ii) Niveaux Première et Terminale Scientifique

A ces deux niveaux, les algorithmes rendus par les élèves ne montrent de difficultés sur la reconnaissance de la structure de l'algorithme décrit dans l'organigramme distribué par l'enseignant, ainsi que sur l'écriture de cet algorithme en langage pseudo-code implémentable sur *AlgoBox*. En particulier, l'implémentation de la fonction et son appel dans la boucle ne posent pas de difficultés auprès de ces élèves quel que soit le niveau scolaire.

2.5.2.2. Analyse *a posteriori*

a) La justification de l'unicité de la solution à l'équation f(x) = 0

(i) Niveau Seconde

Comme nous l'avions supposé lors de l'analyse *a priori*, nous observons une difficulté concernant l'utilisation de la calculatrice chez certains élèves. Une fois la représentation graphique faite, les élèves veulent utiliser le mode « Trace » pour suivre les valeurs des abscisses des points de la courbe, mais 20% d'entre eux n'arrivent pas à prendre en compte

que le cadrage de leur écran n'est pas correct pour pouvoir procéder à une lecture possible des abscisses de points de la courbe. Cela nécessite dans la plupart des cas, une intervention de l'enseignant. Ce fait représente une réelle difficulté pour un certain nombre d'élèves, d'autant plus qu'ils oublient que leur calculatrice garde en mémoire le cadrage de l'utilisation précédente de leur machine. Ce fait confirme bien ce qui est envisagé lors de l'analyse *a priori* sur les difficultés que peuvent éprouver des élèves quant à une utilisation correcte du cadrage, voire du « Zoom », qu'autorise les fonctionnalités de la calculatrice.

Une fois que la représentation graphique de la fonction est terminée avec un cadrage convenable pour la lecture des valeurs des abscisses des points de la courbe, nous observons une utilisation correcte de la part des élèves du mode « Trace » pour cette lecture. Les élèves déterminent alors l'encadrement demandé par une simple lecture graphique.

De même, comme nous l'avions envisagé lors de l'analyse *a priori*, une forte majorité des binômes se contentent d'indiquer le changement de signes comme début de justification. Nous constatons ainsi que la problématique de la « monotonie » de la fonction n'est pas un concept naturel chez des élèves de ce niveau scolaire. En effet, la plupart des binômes répondent à la question d'un encadrement du zéro en ne procédant qu'à une lecture d'abscisses de points du graphe correspondant à des pixels respectivement juste au-dessus et juste en-dessous de l'axe des abscisses.

Lors de l'analyse *a priori*, nous n'avions pas envisagé la possibilité d'utiliser le mode « Solveur » que permet la calculatrice pour déterminer une valeur approchée de la solution de l'équation $f(x) = 0$, car la question est sur un encadrement et non une valeur approchée. Cependant, comme nous l'avons rapporté plus haut, un binôme utilise cette fonctionnalité.

(ii) Niveaux Première et Terminale

En Première

Comme nous nous y attendions, une majorité des binômes utilise le signe de la dérivée pour déterminer le sens de variation de la fonction. Bien que cela ne soit pas demandé, les binômes proposent alors un tableau de variation de la fonction. De même, les élèves mettant en œuvre d'autre types de raisonnements pour justifier la « monotonie », ne donnent pas de raisons pour l'existence. Notre interprétation est que ces élèves, avec le calcul de la dérivée

ou une autre méthode, mettent en œuvre une technique standard d'étude de variation, sans questionner cette technique en rapport avec le problème posé.

En Terminale

Ici aussi, comme nous nous y attendions, les binômes utilisent majoritairement le calcul différentiel pour répondre à l'unicité de la solution. Nous constatons aussi qu'à ce niveau scolaire, les élèves ne se contentent pas de cette justification pour conclure sur la solution de l'équation. En effet, nous notons que quel que soit le raisonnement utilisé pour justifier le sens de variation de la fonction, soit à partir du signe de la dérivée, soit à partir de propriétés sur la somme de deux fonctions ayant la même monotonie, les binômes cherchent systématiquement à justifier que cette solution existe bien en procédant à une vérification du changement de signes pour les images des bornes de l'intervalle d'encadrement obtenu. Ce comportement, différent de celui des élèves de Première, nous semble résulter de l'enseignement reçu sur le TVI, qui met en avant la propriété de changement de signes dans la condition suffisante d'existence. En revanche, nous constatons qu'aucun binôme ne rappelle la condition de continuité de la fonction sur l'intervalle de départ, intervenant pourtant elle aussi dans la condition suffisante d'existence. Les élèves font comme si cette propriété était implicite[86]. Ainsi, dans l'enseignement reçu sur le TVI, il semble que la propriété de « continuité » soit mise en avant moins systématiquement que la propriété de « changement de signes ».

b) L'« application numérique » avec AlgoBox

(i) Niveau Seconde

Comme nous l'avions prévu lors de l'analyse *a priori*, les seules difficultés observées ont été sur l'implémentation de la fonction dans l'environnement *AlgoBox*, même si son appel dans la boucle a bien été reconnu par la majorité des binômes. En effet, malgré le travail fait lors de l'ingénierie « dichotomie discrète » sur les variables, le fait que dans cette seconde sous-ingénierie, il faille déterminer les valeurs de certaines images par la fonction et procéder à une étude de changement de signes concernant ces images, est l'objet d'erreurs ou de connaissances insuffisantes des possibilités qu'offrent l'environnement *AlgoBox* quant à l'implémentation d'une fonction. Certains élèves ont par exemple des difficultés à considérer

[86] Voir sur ce point aussi le déroulement de la phase 4 en Terminale.

que l'implémentation d'une fonction générique dans un algorithme/programme ne correspond pas à implémenter une chaine de caractère (cf. 7c). En effet, d'un point de vue mathématique, de tels élèves semblent ne pas prendre conscience que l'expression « $f(x)$ » pour un x donné correspond à une valeur numérique représentant l'image de x par la fonction f. Ainsi, nous observons que les connaissances mathématiques dans le domaine fonctionnel ne semblent pas encore totalement maitrisées par un certain nombre d'élèves à ce niveau scolaire malgré le fait que les concepts d'images et d'antécédents par une fonction sont des compétences vues depuis la Quatrième (Grade 8), soit depuis deux ans. D'un point de vue informatique, les élèves, qui considèrent l'expression d'une fonction comme une « chaîne de caractères », semblent conforter aussi nos hypothèses faites lors de la sous-ingénierie « dichotomie discrète » concernant les difficultés que peuvent éprouver un élève débutant en informatique à bien concevoir le concept de variable de type « chaîne de caractère ». En effet, pour certains élèves le fait que dans l'expression de la fonction il y ait une lettre *x* impliquerait que ce soit une « chaîne de caractère ». Ils ne conçoivent pas cette lettre *x* de l'expression de la fonction comme étant elle-même une variable de type nombre et par conséquent ils peuvent ne pas la prendre en compte dans la déclaration des variables.

Nous pouvons aussi supposer que certains élèves semblent montrer une certaine résistance à considérer que dans le cas d'une variable informatique, une affectation puisse être donnée sous forme d'une expression de type fonctionnel.

Parmi ceux qui pensent bien à utiliser l'identificateur F1, nous observons que certains font que cette utilisation soit implicitement associée à une fonction, sans que son expression soit pour autant stockée. Nous constatons alors qu'il est dommage que nous n'ayons pas pu observer ces binômes testant leur algorithme avec absence de stockage de l'expression de la fonction dans l'onglet « Utiliser une fonction numérique » qu'offre l'environnement *AlgoBox*, afin de voir leurs réactions aux résultats renvoyés par *AlgoBox* lors de l'exécution de l'algorithme. En effet, l'intervention de l'enseignant ne permet pas une telle observation.

(ii) Niveaux Première et Terminale Scientifique

Conformément à ce que nous avions prévu dans l'analyse *a priori*, les élèves du cycle scientifique, ne montrent pas de difficultés particulières tant sur l'aspect algorithmique que sur l'aspect « application numérique » pour le problème étudié.

2.5.3 Phase 2 : Travail en algorithmique questionnant le choix d'une condition de sortie ; le balayage

2.5.3.1. Le déroulement

Lors de cette deuxième phase, les binômes doivent travailler à compléter la condition d'arrêt d'un algorithme de résolution par balayage pour déterminer un encadrement d'amplitude 10^{-p} (ici, $p = 3$) de la solution de $f(x) = 0$, où f est la fonction polynôme de la phase (1).

Le temps prévu pour cette phase est globalement respecté quel que soit le niveau de la classe. Lors de la phase (1), les élèves se sont bien appropriés la fonction et sa représentation graphique et connaissent déjà un encadrement à bornes entières d'amplitude 1 de la solution, affiné au cours de l'« application numérique ».

Les binômes repartent ainsi de l'intervalle [0 ; 1], justifié au cours de la phase 1, pour débuter cette seconde phase. Cette connaissance sur un encadrement d'amplitude 1, leur permet de repartir de la borne inférieure de cet encadrement comme valeur de Binf pour déterminer un nouvel intervalle en fixant un nombre entier *p* afin d'obtenir le nouvel intervalle d'amplitude 10^{-p}.

Nous sommes obligés pour la suite de la description du déroulement de cette phase de distinguer le niveau de Seconde avec les niveaux du cycle terminal scientifique. En effet, nous pouvons observer de fortes différences entre ces deux niveaux.

a) Niveau Seconde

(i) Structure « TantQue »

Même si le choix de l'environnement *LARP* pour le retour du travail fait est clairement indiqué aux élèves, parmi les binômes observés de Seconde, très peu (environ 8% d'entre eux) ne travaillent que sur *AlgoBox*. Ils adaptent la structure de boucle donnée par l'algorithme distribué au début de la phase, afin qu'elle puisse être implémentable sur *AlgoBox*, et, rendent un algorithme avec une structure « TantQue » (Fig. 129), correct bien que non conforme à ce qui est attendu. Notamment, la condition de continuation est correcte, exprimée de façon économique (référence à la valeur à droite de l'intervalle et non au produit des deux valeurs aux bornes), et fait appel au formalisme fonctionnel. Le passage au « TantQue », où le corps

de boucle n'est pas exécuté après l'arrêt, les oblige aussi à modifier l'affichage de l'intervalle trouvé : ils savent reconnaitre les valeurs des bornes « x » et « $x + 10^{-3}$ » de l'encadrement final.

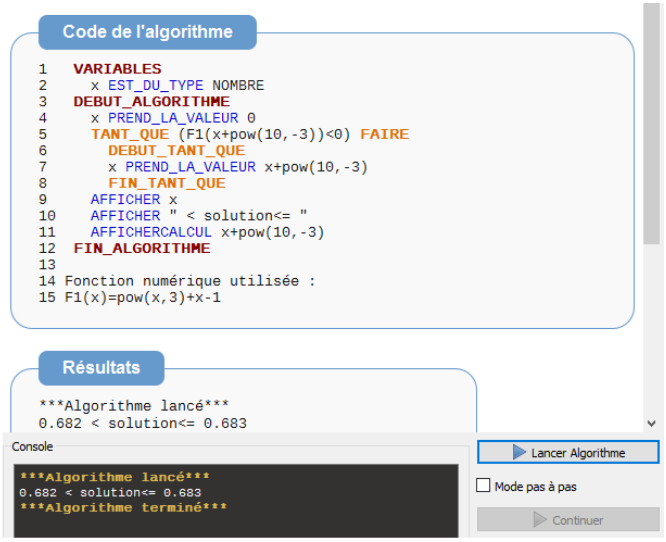


Figure 129 (Un algorithme sous AlgoBox avec une structure « TantQue »)

De plus, les élèves savent reconnaître la fonctionnalité qui permet le calcul d'une puissance d'un nombre (10^{-3} est implémenté « pow(10,-3) » dans *AlgoBox*). De même, ces élèves savent utiliser la fonctionnalité « Utiliser une fonction numérique » qui leur permet d'implémenter l'expression de la fonction en fonction de la variable x et de procéder à son appel dans le programme en utilisant l'identificateur F1 au lieu d'inscrire dans le programme l'expression complète de la fonction permettant les calculs des images de $x + 10^{-3}$. Le fait aussi d'utiliser la fonctionnalité d'*AlgoBox* : « AFFICHERCALCUL », nous montre que ces élèves ont une bonne connaissance des possibilités qu'offre *AlgoBox.* En effet, à la place de la ligne 11 « AFFICHERCALCUL x+pow(10,-3) » (Fig. 130), les élèves pourraient écrire les lignes 11 « x PREND LA VALEUR x+pow(10,-3) » et 12 « AFFICHER x », ce qui « alourdirait » l'algorithme.

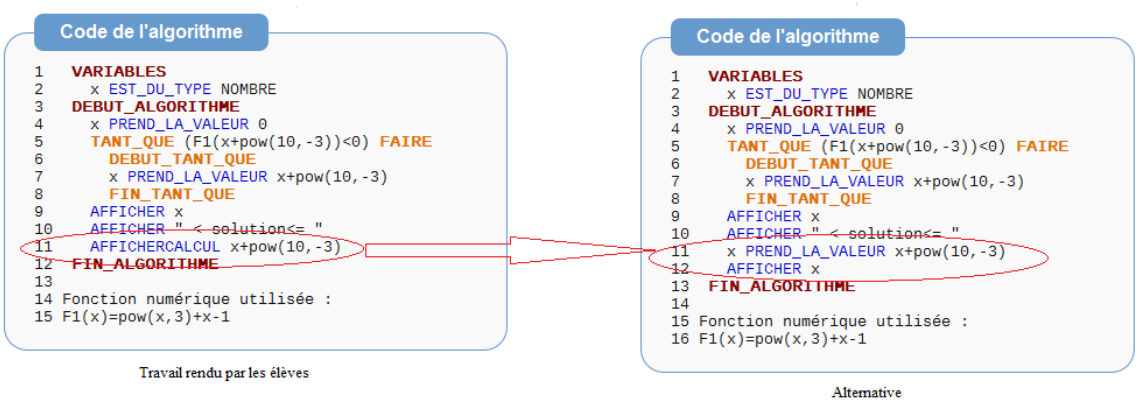


Figure 130 (Algorithmes proposant ou non d'utiliser la fonctionnalité d'AlgoBox : « AFFICHERCALCUL »)

Quant aux autres binômes qui travaillent aussi avec cette structure « TantQue », ils rendent tous un organigramme (Fig. 131) élaboré dans l'environnement *LARP* que certains convertissent aussi en « pseudo-code » en utilisant cette possibilité offerte par cet environnement (Fig. 132). En revanche, ces binômes travaillant sous *LARP* ne proposent pas une « fonction générique » comme nous le constatons ci-dessous (Fig. 131 et 132). Lors d'échanges avec ces binômes, les élèves nous répondent qu'ils n'ont pas vu la nécessité de le faire, d'autant plus que cela aurait impliqué la construction d'un « Module auxiliaire » (c'est-à-dire un sous-programme) qui aurait pour but de calculer les images des valeurs ($x + 10^{-3}$) par la fonction générique (ici, $x \rightarrow x^3 + x - 1$).

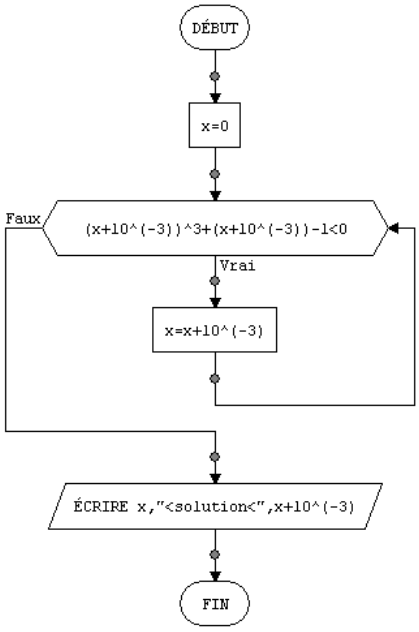


Figure 131 (Organigramme sous LARP avec utilisation d'une structure « TantQue »)

Pseudo-code de PRINCIPAL

```
DÉBUT
    x=0
    TANTQUE (x+10^(-3))^3+(x+10^(-3))-1<0 FAIRE
        x=x+10^(-3)
    FINTANTQUE
    ÉCRIRE x,"<solution<",x+10^(-3)
FIN
```

Figure 132 (Transcription en « pseudo-code » de l'organigramme obtenu à la figure 131 sous LARP)

(ii) Structure « Répéter… Jusqu'à »

Une très grande majorité des binômes travaillent sur cette structure et utilisent pour cela l'environnement *LARP*, ce qui correspond aux attentes de cette phase.

Au cours de cette phase, comme nous nous y attendions, nous observons des difficultés chez certains de ces binômes travaillant avec la structure « Répéter… Jusqu'à », même si pour certain le travail final rendu est en général correct. Nous proposons ici de présenter ces diverses difficultés et de les analyser.

Une première difficulté observée porte sur la condition de sortie de la boucle « Répéter …

Jusqu'à » dans l'algorithme « papier-crayon ». En effet, les différentes conditions de sortie erronées observées sont les suivantes : (E1) « $f(x + 10^{-3}) < 0$ » (Fig. 133) ; (E2) « $f(x + 10^{-3}) - f(x) > 0$ » (fig. 134) ; (E3) un ajout d'une ligne d'affectation « $f(x)$ prend la valeur $f(x) \times f(x + 10^{-3})$ » dans la boucle « Répéter… Jusqu'à », suivie de la condition de sortie « $f(x) > 0$ » (Fig. 135).

Initialisation :	x prend la valeur 0
Traitement :	Répéter
	x prend la valeur $x + 10^{-3}$
	Jusqu'à $f(x + 10^{-3}) < 0$
Sortie :	Afficher $x - 10^{-3} < x_0 \leq x$

Figure 133 (Cas où le test de sortie de boucle est $f(x + 10^{-3}) < 0$)

Initialisation :	x prend la valeur 0
Traitement :	Répéter
	x prend la valeur $x + 10^{-3}$
	Jusqu'à $f(x + 10^{-3}) - f(x) > 0$
Sortie :	Afficher $x - 10^{-3} < x_0 \leq x$

Figure 134 (Cas où le test de sortie de boucle est $f(x + 10^{-3}) - f(x) > 0$)

Initialisation :	x prend la valeur 0
Traitement :	Répéter
	x prend la valeur $x + 10^{-3}$
	$f(x)$ prend la valeur $f(x) \times f(x + 10^{-3})$
	Jusqu'à **$f(x) > 0$**
Sortie :	Afficher $x - 10^{-3} < x_0 \leq x$

Figure 135 (Cas d' un ajout d'une ligne d'affectation « f(x) prend la valeur $f(x) \times f(x + 10^{-3})$ » dans la boucle « Répéter… Jusqu'à », suivie de la condition de sortie « $f(x) > 0$ »)

Comme nous l'avions prévu, les implémentations dans *LARP* sont pour l'essentiel sous forme d'organigramme et gardent bien une structure de type « Répéter… Jusqu'à ». Cependant, comme le montrent les organigrammes (Fig. 136 à 140) et l'algorithme en pseudo-code (Fig. 141), une deuxième difficulté observée concerne l'implémentation de l'algorithme dans l'environnement *LARP* quels que soient les choix faits sur la condition de sortie de boucle.

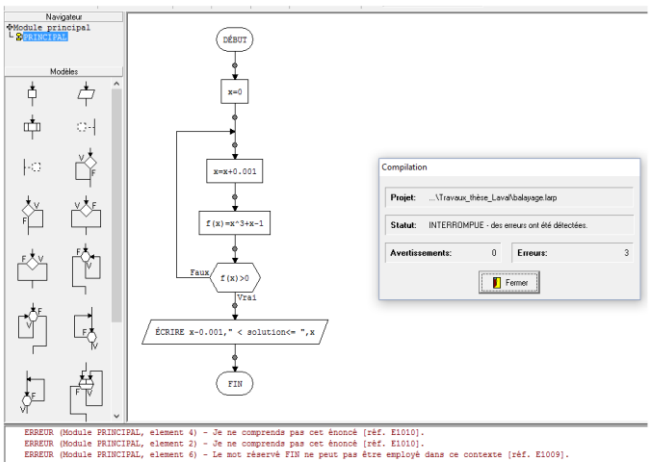


Figure 136 (Type d'erreur lors de l'implétemtation d'un algorithme dans l'environnement LARP)

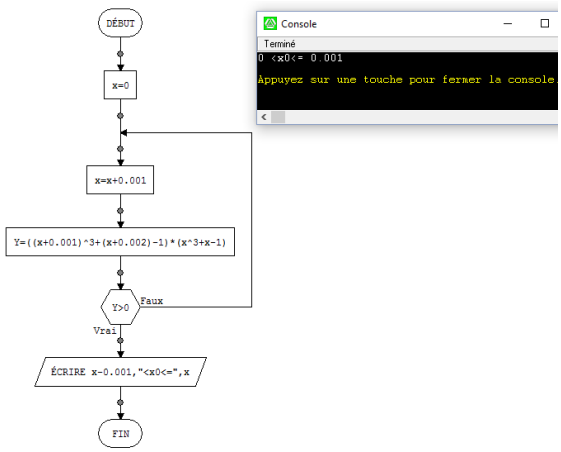


Figure 137 (Type d'erreur lors de l'implétemtation d'un algorithme dans l'environnement LARP)

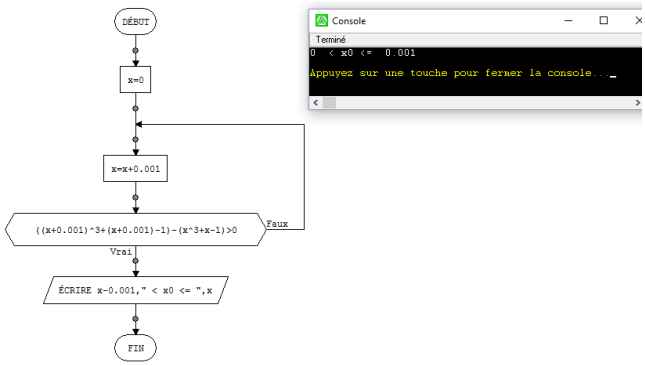


Figure 138 (Type d'erreur lors de l'implétemtation d'un algorithme dans l'environnement LARP)

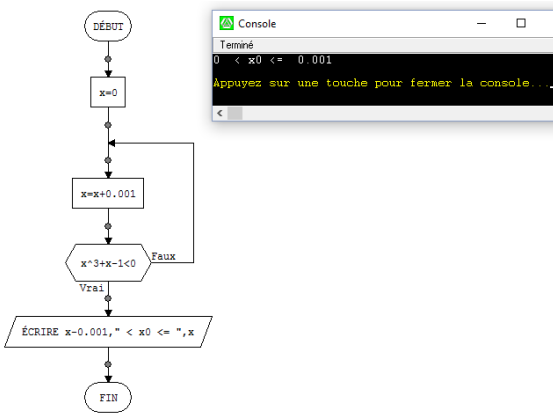


Figure 139 (Type d'erreur lors de l'implétemtation d'un algorithme dans l'environnement LARP)

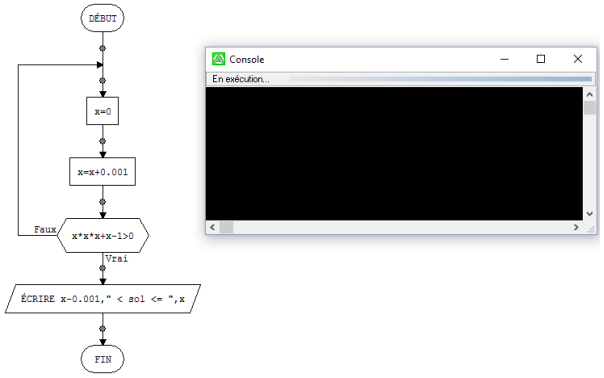


Figure 140 (Type d'erreur lors de l'implétemtation d'un algorithme dans l'environnement LARP)

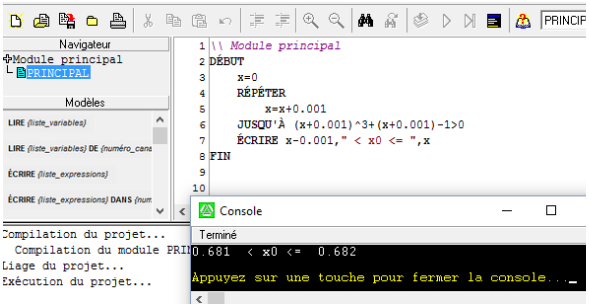


Figure 141 (Type d'erreur lors de l'implétemtation d'un algorithme dans l'environnement LARP)

Pour l'organigramme de la figure 136, le binôme utilise comme convenu une structure « Répéter… Jusqu'à » avec une condition d'arrêt correcte. L'initialisation de *x* est correctement faite en dehors de la boucle. Dans le corps de boucle, les élèves de ce binôme montrent bien qu'ils comprennent que l'amplitude de l'intervalle final d'encadrement de la solution fait que les données de la variable itérative *x* doivent évoluer avec un pas de 10^{-3}. Cependant, ils mettent en pratique une traduction d'une notation mathématique concernant l'image d'un nombre réel *x* par la fonction *f*, pour procéder une affectation à une valeur de fonction. Ainsi, lors de l'exécution de l'algorithme, un message d'erreur apparait indiquant une erreur de compilation. Les élèves du binôme obtenant les résultats donnés à la figure 12a, ne savent pas comment interpréter ces trois messages d'erreurs affichés par LARP : (1) « ERREUR (Module PRINCIPAL, élément 4) – Je ne comprends pas cet énoncé [réf. E1010]. » ; (2) « ERREUR (Module PRINCIPAL, élément 2) – Je ne comprends pas cet énoncé [réf. E1010]. » ; (3) « ERREUR (Module PRINCIPAL, élément 6) – Le mot réservé FIN ne peut pas être

employé dans ce contexte [réf. E1009]. ».

Pour l'organigramme de la figure 137, le binôme utilise aussi la structure « Répéter… Jusqu'à » et comme pour le cas précédent, ce binôme initialise correctement la variable *x* en dehors de la boucle. Puis dans le corps de boucle, les élèves de ce binôme montrent bien qu'ils ont aussi compris que l'amplitude de l'intervalle final d'encadrement de la solution correspond à un pas de 10^{-3} pour déterminer les données de la variable itérative *x*. Ensuite, toujours dans la boucle, ils affectent à une variable Y le produit de $f(x + 10^{-3})$ par $f(x)$, malgré une petite erreur d'« étourderie » sur l'image de $x + 10^{-3}$ par *f* dans le deuxième monôme $x \rightarrow 3x$. En effet, les élèves remplacent *x* par *x* + 0,002 au lieu de *x* + 0,001. Le test d'arrêt porte alors sur le produit $f(x + 10^{-3}) \times f(x)$. Cette condition d'arrêt étant « inversée » entraîne une exécution de l'algorithme sur un seul passage dans la boucle « Répéter… Jusqu'à ».

Pour l'organigramme de la figure 138, l'approche du problème par ce binôme est proche du précédent. En effet, les élèves utilisent aussi une structure « Répéter… Jusqu'à », initialisent correctement la variable *x* en dehors de la boucle et ensuite dans le corps de boucle, ils traduisent l'amplitude de l'intervalle cherché en pas pour déterminer les différentes valeurs de la variable itérative *x*. Cependant, cette fois le test d'arrêt porte sur la différence $f(x + 10^{-3}) - f(x)$. Là aussi, cette condition étant vérifiée dès le premier passage dans la boucle, l'algorithme renvoie un résultat incorrect par rapport au problème posé.

Pour l'organigramme de la figure 139, le binôme utilise aussi une structure « Répéter… Jusqu'à », avec une initialisation correcte de la variable itérative *x* en dehors de la boucle. Ensuite dans le corps de boucle, le binôme associe aussi correctement l'amplitude de l'intervalle cherché au pas pour déterminer les différentes valeurs de la variable itérative *x*. Le test d'arrêt porte bien sur le signe de $f(x + 10^{-3})$. Cependant, le test étant « inversé », il est vérifié dès le premier passage dans la boucle « Répéter… Jusqu'à ». Alors, l'algorithme s'arrête et renvoie les résultats correspondant à ce premier passage.

Pour les algorithmes (Fig. 137 à 139), les binômes ayant obtenu un résultat lors de l'exécution de l'algorithme, malgré une incitation de l'enseignant à analyser ce résultat, ils l'acceptent comme réponse au problème. Notons que des binômes voisins ont eu une réponse analogue avec un organigramme différent.

Pour l'organigramme (Fig. 140), le binôme utilise bien une structure du type « Répéter…

Jusqu'à » avec un test d'arrêt correct portant sur le signe de $f(x + 10^{-3})$. De plus, le binôme associe aussi l'amplitude de l'intervalle cherché en pas pour la détermination des différentes valeurs successives de la variable itérative *x* qui se trouve dans le corps de boucle. Cependant, les élèves de ce binôme commettent une erreur sur l'initialisation de cette variable *x,* qu'ils font faire dans le corps de boucle et non avant l'entrée dans ce corps. Ceci a pour conséquence que l'exécution de l'algorithme tourne en boucle infinie. Nous rapportons alors un extrait de l'échange entre les deux élèves (El1 et El2), ainsi que l'intervention de l'enseignant (EnsSec) observant l'exécution de cet algorithme. L'extrait débute juste avant l'exécution de l'algorithme proposé par les deux élèves.

El1 : *Regarde J…, l'algorithme correspond à celui proposé par le prof.*
(Commentaire : Il montre l'algorithme « papier-crayon » à son camarade)
El2 : *Oui je crois… Bon, on le fait tourner.*
(Commentaire : l'élève passe en mode « exécution ». Apparait l'écran de l'« exécution ».)
El2 : *L'écran est vide ! Eh ! Il y a écrit « En exécution ». Je ne comprends pas pourquoi on n'a pas de résultat.*
El1 : *Il calcule…*
El2 : *C'est long !! Les autres, ils ont tout de suite le résultat. On a dû faire une erreur.*
El1 : *Où ? Comment fait-on pour arrêter cela ?*
(Commentaire : L'enseignant vient d'arriver au niveau du binôme)
EnsSec : *Alors, votre algorithme fonctionne ?*
El1 : *Non. Quand on l'exécute, il tourne en rond.*
EnsSec : *Montrez-moi votre algorithme « papier-crayon ».*
(Commentaire : Les élèves montrent l'algorithme « papier-crayon » où la condition de sortie est correctement écrite (fig. 142)

Initialisation :	x prend la valeur 0
Traitement :	Répéter
	x prend la valeur $x + 10^{-3}$
	Jusqu'à $f(x) > 0$
Sortie :	Afficher $x - 10^{-3} < x_0 \leq x$

Figure 142 (Un algorithme « papie-crayon »)

EnsSec : *C'est bon. Regardons votre organigramme.*
(Temps de silence de quelques secondes)
EnsSec : *D'accord. Comparez vos deux algorithmes. N'observez-vous pas une différence ?*
(Temps de silence de 20 secondes)
EnsSec : *Alors ? Vous ne voyez pas de différence ?*
El1 : *Non.*
(Temps de silence de quelques secondes)
El2 : *Non Monsieur. Ce sont les mêmes. La seule différence est leur représentation.*
EnsSec : *A quoi correspond l'information « x prend la valeur 0 » dans l'algorithme papier ?*
(Commentaire : L'enseignant indique la ligne aux deux élèves)
El1 et El2 : *A l'initialisation !*
EnsSec : *A votre avis, où se trouve cette initialisation par rapport à la boucle ?*
(Temps de silence de quelques secondes)
El2 : *En dehors du traitement.*
EnsSec : *C'est-à-dire ?*
El2 : (Hésitation) *Euh… Avant la boucle ?*

EnsSec : *Oui. Bon, maintenant regardez votre organigramme. Que constatez-vous ?*
(Commentaires : (a) Suite à un problème de discipline dans la classe, l'enseigne laisse à ce moment les deux élèves sans attendre leurs réponses. (b) Après quelques secondes de réflexion de la part d'un des deux élèves, l'échange reprend entre les deux élèves.)
El1 : *Faut peut-être mettre la flèche de retour après l'instruction « x = 0 ».*
El2 : *Essayons.* […]

Les élèves procèdent alors à ce changement dans l'organigramme puis ils font exécuter l'algorithme ainsi obtenu et obtiennent les résultats attendus.

Pour l'algorithme (Fig. 141), les élèves du binôme ont utilisé un format « pseudo-code » avec une structure « Répéter… Jusqu'à ». L'initialisation de la variable *x* est correctement faite en dehors de la boucle. Dans le corps de boucle, les élèves font bien apparaître la succession des données de la variable *x* en ayant pris un pas de 10^{-3} correspondant à l'amplitude de l'intervalle cherché. Le test d'arrêt porte sur le signe de l'image d'une valeur d'un nombre réel par *f*. Cependant, les élèves le font porter sur l'image de $x + 10^{-3}$, et pas sur *x*. Ainsi, l'intervalle obtenu est incorrect car « décalé » de 0,001. En effet, pour les deux bornes affichées avec la valeur de *x* à la sortie de la boucle dans l'algorithme du binôme, les images sont encore de même signe. L'intervalle affiché après l'exécution de l'algorithme est celui qui précède celui encadrant le zéro.

(iii) Retour sur la structure « TantQue »

Nous souhaitons revenir sur une erreur observée (qui sera corrigée rapidement par les élèves après une courte intervention de l'enseignant) chez un des binômes ayant procédé à une modification de la structure de boucle et fait ainsi intervenir une boucle de type « TantQue ». En effet, ce binôme place, comme pour la figure 140 ci-dessus, l'initialisation de la variable *x* dans le corps de boucle. Ainsi, malgré une structure correcte présentant un test de continuation dépendant du signe de $f(x + 10^{-3})$ et une affection des données correctes sur la variable *x* à l'intérieur de la boucle autre que celle de l'initialisation, les élèves de ce binôme obtiennent un message d'erreur (fig. 143) lors de l'exécution de l'algorithme : comme *x* n'est pas initialisée et comme *LARP* n'affecte pas de type par défaut, ce logiciel repère une erreur de type au test à l'entrée de la boucle. Les élèves sont dans l'incapacité d'interpréter ce message.

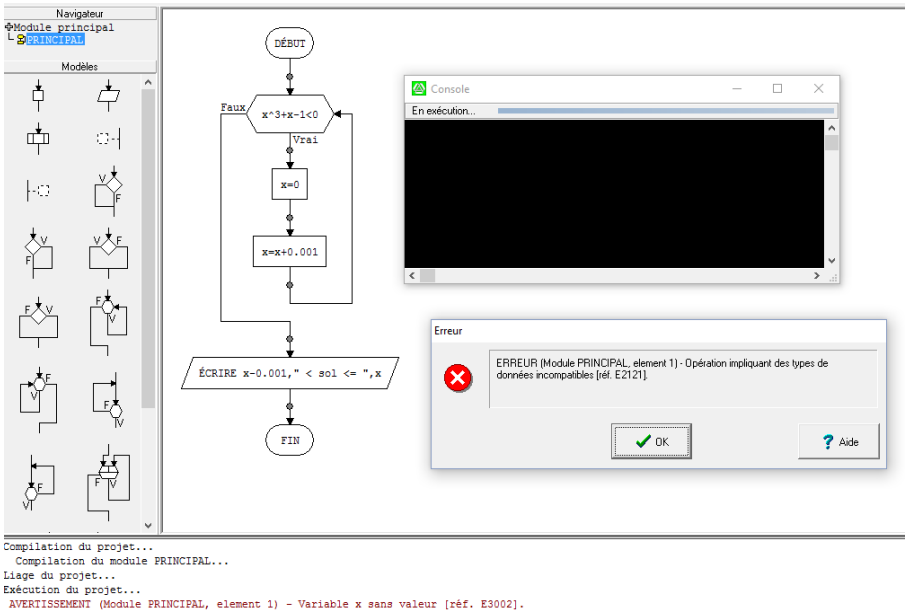


Figure 143 (Type d'erreur lors de l'implétemtation d'un algorithme dans l'environnement LARP)

(iv) Tableaux de synthèse des résultats observés lors la phase 2

Dans cette courte section, nous souhaitons résumer, sous forme de deux tableaux (Tableaux n° 33 et n° 34), les résultats observés lors du déroulement de cette seconde phase.

Sorties de boucle dans l'algorithme « à trou »				
Propositions erronées			Proposition juste	Absence de réponse
E1	E2	E3		
10%	20%	12%	50%	8%

Tableau 33

Algorithmes transcrits sous *LARP*				Algorithmes sous *AlgoBox*	Absence de transcription d'algorithme
84%				8%	8%
Organigramme		Pseudo-code		« TantQue »	
85,7%		14 ,3%		100%	
« Répéter… Jusqu'à »	« TantQue »	« Répéter… Jusqu'à »	« TantQue » [87]		
71,4%	28,6%	100%	0%		

Tableau 34

b) Niveaux Première et Terminale Scientifique

A ce niveau scolaire, nous n'observons pas de difficultés particulières de la part d'une très grande majorité des binômes, tant en Première qu'en Terminale.

L'algorithme « à trou » distribué étant écrit en langage « textuel » avec une structure

[87] 4% si nous tenons compte de ceux qui ont transcrit leur organigramme en pseudo-code avec LARP (cf. a) (i) Structure « TantQue ».

« Répéter … Jusqu'à », tous les binômes de Première et 90% de ceux de Terminale travaillent sur une implémentation de cet algorithme dans l'environnement *LARP* en choisissant le format « Organigramme ».

Bien qu'utilisant aussi l'environnement *LARP*, les 10% des binômes de Terminale qui ne font pas le choix de représenter un organigramme, utilisent directement le format « pseudo-code » pour implémenter leurs algorithmes.

Cependant, ceux qui font le choix du « pseudo-code » éprouvent des difficultés de type syntaxique pour l'implémentation de leurs algorithmes et font ainsi appel à l'enseignant pour trouver des réponses à leurs questions sur la syntaxe de l'environnement *LARP*, plutôt que de passer par une forme « organigramme ». En effet, l'environnement *LARP* ne permettant pas de passer du « pseudo-code » à l'organigramme, contrairement à l'inverse, ces élèves souhaitant rester dans le langage « textuel » ne peuvent pas travailler en totale autonomie pour l'implémentation de l'algorithme.

Concernant la condition de sortie de boucle qui correspond « au trou » de l'algorithme distribué, nous n'observons pas de difficulté de la part des binômes à la concevoir et à la formuler. En effet, les binômes ayant bien pris connaissance, lors de la phase précédente, des propriétés de la fonction et compris le travail fait lors de la sous-ingénierie « dichotomie discrète » sur les structures itératives, considèrent qu'il suffit de déterminer une condition sur le signe du nombre $f(x)$ pour sortir de la boucle, sachant que la variable x est affectée à l'étape précédente de la valeur $x + 10^{-3}$. La prise en compte et les affectations des données de la variable itérative x sont bien considérées par la quasi-totalité des binômes. En effet, nous observons qu'au début de l'implémentation des algorithmes, deux binômes de Première (fig. 144 et 145) « oublient » d'initialiser la variable x, mais lors de l'exécution de l'algorithme, *LARP* leur renvoie un message d'erreur à la ligne de mise à jour de la variable itérative x utilisée dans la boucle. Nous notons ci-dessus que contrairement *AlgoBox* qu'ils utilisent habituellement, *LARP* n'initialise pas par défaut les variables, et repère une erreur de type lors d'un test portant sur une variable « sans valeur ». Cependant, les élèves de ces deux binômes savent interpréter correctement cette erreur, et peuvent d'eux-mêmes corriger leurs algorithmes implémentés.

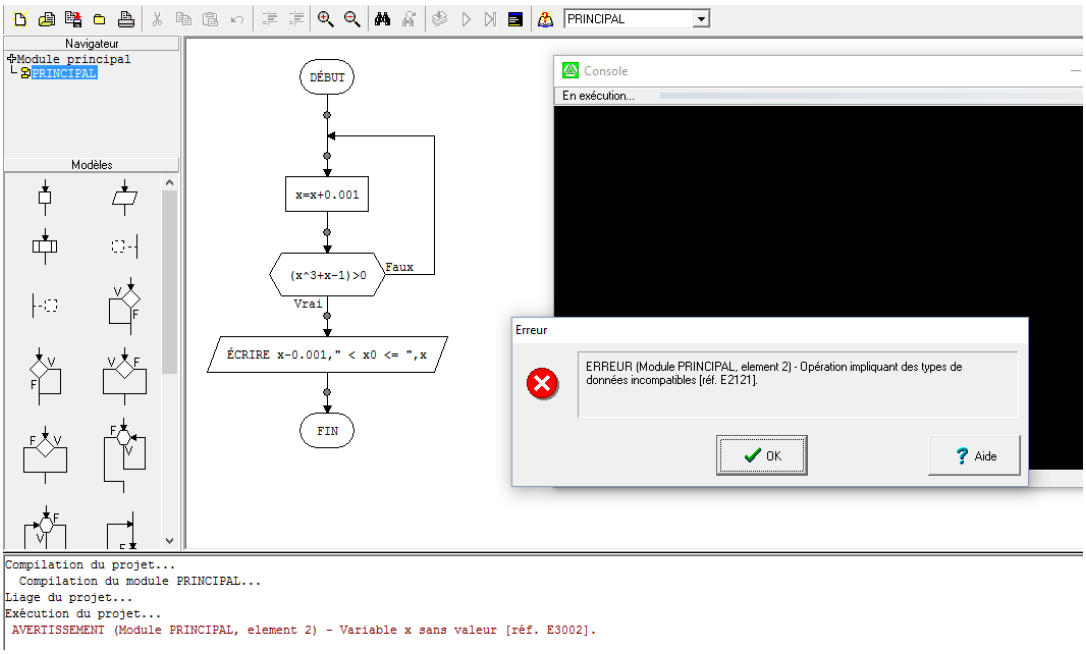


Figure 144 (Type d'erreur lors de l'implétemtation d'un algorithme dans l'environnement LARP)

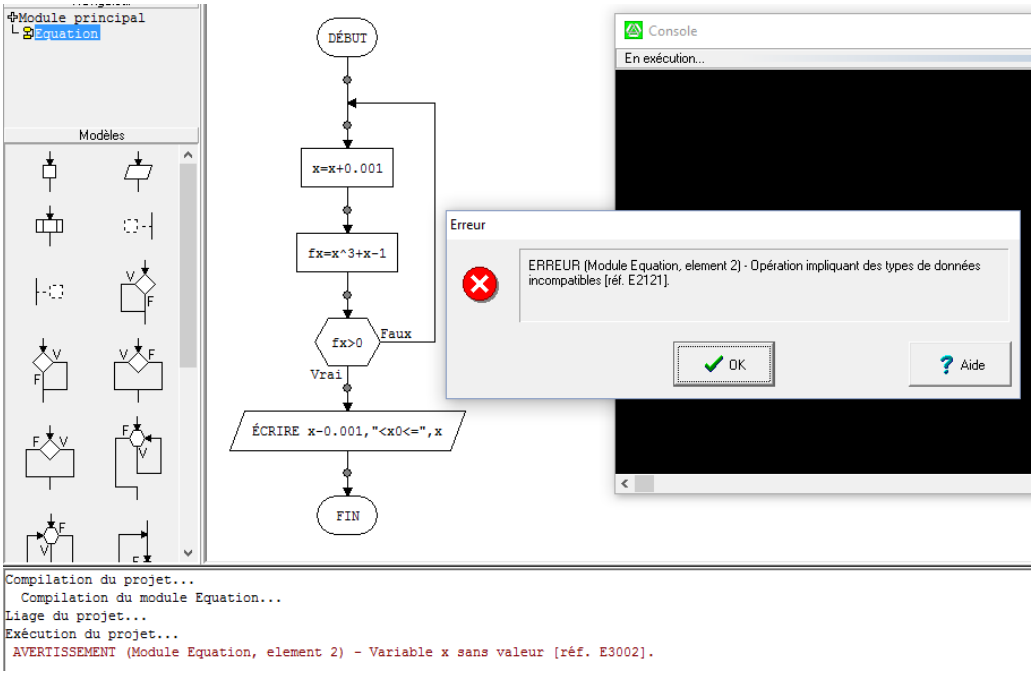


Figure 145 (Type d'erreur lors de l'implétemtation d'un algorithme dans l'environnement LARP)

2.5.3.2. Analyse *a posteriori*

Au vu des observations faites au cours du déroulement sur les difficultés et les types d'erreurs rencontrés par les binômes, nous choisissons pour cette analyse *a posteriori* de la phase 2 de ne pas distinguer les niveaux scolaires, mais de favoriser une analyse en fonction du choix fait par les binômes sur l'environnement informatique et des structures algorithmiques mises en en place, ainsi que sur les types d'erreurs.

a) L'environnement AlgoBox et la structure « TantQue »

Nous observons que certains binômes de Seconde (cf. le cas de la figure 129) ne respectent pas la consigne de travailler dans l'environnement *LARP* et préfèrent travailler dans *AlgoBox*. Nous pensons qu'à ce stade des expérimentions, cela vient du fait que la phase précédente est mise en œuvre dans cet environnement. De plus, l'enseignant de la classe ayant une pratique régulière d'*AlgoBox* propose, avant que ne soient mises en place les ingénieries, de nombreuses tâches algorithmiques à faire sur *AlgoBox*, en particulier lors d'évaluations.

Le travail fait sur *AlgoBox* oblige ces binômes à reprendre le travail qu'ils ont fait quand ils ont complété correctement l'algorithme « à trou », où la structure est de type « Répéter … Jusqu'à ». Nous observons ainsi que ces binômes font preuve d'une bonne maitrise des structures « Répéter… Jusqu'à » et « TantQue ». En effet, ils montrent qu'ils savent passer d'une structure à l'autre sans commettre d'erreur sur la transcription d'une condition d'arrêt à une condition de continuation qu'ils expriment en faisant référence à la valeur de droite de l'intervalle. De plus, nous observons que la gestion de la variable informatique *x* est bien traitée par ces mêmes élèves. En effet, ils font porter le test sur $f(x+10^{-3})$ plutôt que sur $f(x)$ pour tenir compte de l'arrêt de l'itération avant le corps de boucle et modifient en conséquence l'intervalle affiché. Ces mêmes élèves montrent aussi qu'ils connaissent la syntaxe de l'environnement informatique utilisé et savent aussi utiliser les possibilités fonctionnelles qu'offre cet environnement.

Le choix fait par les binômes des figures 129 et 143 de produire une structure « TantQue », bien que travaillant avec *LARP*, nous renvoie à ce que nous avions observé lors la phase 2 de la sous-ingénierie « dichotomie discrète », où un certain nombre d'élèves avaient préféré cette structure de type « TantQue » à tout autre structure, en signalant que la recherche d'un nombre vérifiant une condition se poursuivait tant que la condition n'était pas vérifiée.

b) LARP et la structure « Répéter… Jusqu'à »

(i) Erreur portant sur le test d'arrêt

Pour les binômes de Seconde ayant obtenus les algorithmes des figures 137 à 139, nous observons une erreur portant sur le test d'arrêt. Pour le binôme de la figure 138, nous constatons que les élèves mettent en place un raisonnement adapté de l'algèbre qui consiste à étudier le signe de la différence $b - a$ comme moyen de comparer deux nombres réels a et b, mais cette comparaison entre $f(x)$ et $f(x+10^{-3})$ n'a pas de sens pour le problème. En effet,

la fonction étant croissante, le test est vérifié pour toute valeur de *x*. Le binôme de la figure 137 propose un test d'arrêt sur le signe du produit de $f(x)$ et $f(x+10^{-3})$ et le binôme de la figure 139 sur le signe de $f(x+10^{-3})$. Ces deux binômes commettent une erreur d'inversion du signe de l'inégalité. Notre interprétation est que ces binômes ont des difficultés à concevoir un test d'arrêt, voyant le processus comme un « TantQue » conditionné par une condition de continuation.

(ii) L'initialisation

A ce stade de l'expérimentation, nous observons aussi que la place de l'initialisation d'une variable n'est pas encore acquise par certains élèves (cf. les figures 140 et 141 de certains binômes de Seconde), ce qui renvoie à une conception de l'itération qui n'identifie pas la fonction de chacune des composantes. De même, certains élèves de Première (cf. les figures 144 et 145) n'intègrent pas dans un premier temps la nécessité de l'initialisation. Cependant, nous pensons que des élèves de Première Scientifique pourraient faire le lien entre l'initialisation d'une variable informatique et la valeur initiale associée à une suite de réels *x* correspondant à une progression arithmétique de raison et de premier terme connus. De même, n'observant pas une telle erreur au niveau de la Terminale Scientifique, nous pouvons supposer que le travail fait en amont dans le champ de l'analyse (et de l'arithmétique pour ceux qui suivent l'enseignement de spécialité) autour du raisonnement par récurrence, peut permettre à ces élèves de considérer l'initialisation comme ne faisant pas partie du corps de boucle qu'ils peuvent interpréter comme correspondant à la phase d'« hérédité » pratiquée lors d'une démonstration par récurrence.

(iii) Une erreur concernant l'affectation

Pour le binôme de la figure 136, l'erreur porte sur une transposition d'une notation mathématique concernant l'image d'un nombre réel *x* par une fonction *f* à un environnement informatique sans procéder à l'adaptation que nécessiterait cet environnement afin de se conformer à un type « affectation » à une variable informatique. En effet, l'affectation telle que l'écrit ce binôme, présente en partie droite une expression algébrique comportant la variable *x* préalablement affectée, et en partie gauche, l'image de *x* par *f* en notation fonctionnelle ($f(x)$) plutôt qu'une variable numérique qui prendrait alors la valeur de l'image par la fonction de cette valeur de *x*. La conception de l'affectation semble ici très influencée par la « définition » d'objets mathématiques. Cette conception fonctionne dans les cas

habituels d'affectation d'une valeur à une variable et les élèves l'étendent aux fonctions.

c) L'exploitation des résultats de la compilation (analyse syntaxique par le logiciel) et de l'exécution.

Certaines erreurs que nous venons de relever, provoquent une erreur à la compilation ; nous avons montré dans l'observation que les messages renvoyés ne peuvent pas toujours être exploitées par les élèves même avec l'aide de l'enseignant. A l'exécution du programme, certaines erreurs se traduisent par une boucle infinie : nous avons montré que les binômes de Seconde ne prennent pas conscience du phénomène et donc ne reprennent pas leur algorithme.

D'autres erreurs conduisent à un encadrement qui n'est pas celui attendu (cf. les figures 137 à 139). Dans la majorité des cas, l'encadrement obtenu est [0 ; 0,001] ce qui correspond à un programme qui s'arrête au premier passage dans la boucle à cause notamment d'une inversion du test d'arrêt ou de continuation. Nous pensons que les élèves pourraient remettre en cause ce résultat manifestement inadapté. Une autre erreur conduit à un encadrement décalé d'un pas vers la droite (cf. figure 141). C'est plus difficile à détecter, mais les élèves pourraient comparer avec le résultat obtenu à la phase 1. Nous observons ainsi que les élèves n'ont pas de regard critique sur le résultat affiché après l'exécution de l'algorithme. Ils semblent ne pas ressentir la nécessité de vérifier si les valeurs numériques données par l'ordinateur répondent à la question posée sur l'encadrement de la solution.

d) La valeur prise en compte pour la sortie de l'itération

Comme nous l'avons fait remarquer précédemment, 40% des binômes ont fait porter leur test d'arrêt (ou de continuation) sur le signe du produit de $f(x)$ par $f(x+10^{-3})$, et les 60% autres plus simplement sur le signe de $f(x)$.

Voici certains éléments qui peuvent contribuer au choix des élèves :

(1) En faveur de $f(x)$:

- dans l'algorithme « papier-crayon » proposé, le test d'arrêt à compléter est « Jusqu'à f(…)……. » ;
- pour la sous-ingénierie « dichotomie discrète », quel que soit le type de structure de boucle mis en œuvre par les élèves, ils ont mis en place des tests portant plutôt sur une valeur et non sur un produit.

(2) En faveur du produit :

- l'*algorithme de dichotomie* de la phase 1 de cette première partie de la sous-ingénierie « dichotomie continue » met en œuvre une comparaison sur le produit $f(a) \times f\left(\frac{a+b}{2}\right)$, ne tenant pas compte par conséquent de la propriété de croissance de la fonction pourtant démontrée au préalable.

Les élèves qui font le choix (1), ont-ils réellement compris qu'une simplification serait possible compte tenu de la propriété de croissance vérifiée par la fonction ? Ceux qui font le choix (2), l'ont-ils réellement compris comme un test de changement de signe ? Ou ces élèves se laissent-ils guider par les formes proposées antérieurement ou dans l'énoncé ? Les éléments d'observation ne nous permettent pas réellement de répondre à ces interrogations. Il aurait été intéressant que les élèves puissent comparer les deux expressions. Plus généralement, dans cette phase, nous relevons une variété dans les productions des élèves qui aurait pu faire l'objet d'une discussion collective.

e) Le cas des élèves du cycle terminal Scientifique

D'une manière générale, nous pouvons observer qu'au niveau du cycle terminal Scientifique, les élèves semblent beaucoup plus autonomes que ceux de Seconde. Une très forte majorité d'entre eux montrent qu'ils ont bien pris conscience de l'aspect analyse du problème, en particulier par les actions à mettre en place avec la fonction, ainsi que sur la transposition du travail fait dans le domaine de l'analyse à celui à pratiquer dans celui de l'algorithmique « papier-crayon », lors de l'élaboration du test d'arrêt associé à la structure de type « Répéter… Jusqu'à ».

De plus, les binômes de cycle Scientifique semblent aussi comprendre pourquoi leurs enseignants leur demandent de rendre des travaux dans l'environnement *LARP* contrairement à ce que nous pouvons observer chez certains élèves de Seconde. En effet, la structure proposée par l'algorithme « à trou » se référant à un corps de boucle « Répéter… Jusqu'à » et cette structure ne faisant pas partie des choix possibles autorisés par l'environnement *AlgoBox*, les élèves du cycle Scientifique se situent tout de suite dans le seul environnement qui leur est accessible et qui leur permet d'utiliser cette structure de boucle.

f) Tableau synthétisant l'analyse sur le travail observé au cours de cette phase 2

Souhaitant synthétiser les différentes observations décrites ci-dessus, nous proposons

pour cela de les résumer dans un tableau (Tableau n° 35) en nous référant aux deux Espaces de Travail qui concernent les milieux dans lesquels les élèves travaillent tout au long de cette seconde phase.

	$ETA_{spécifique}$	$ETM_{Analyse}$	**Commentaires afin de montrer que ces ETM/ETA associés aux phases 1 er 2 sont porteurs d'idées intéressantes**
Comparer les algorithmes de la phase 1 et de la phase 2.	Dans le cas d'un $ETA_{papier/crayon}$, analyser l'algorithme textuel d'un point de vue algorithmique • Passer d'un organigramme (phase 1) à un algorithme « textuel » ; • Etudier la structure « Répéter… Jusqu'à » afin de définir le test d'arrêt (Genèses sémiotique et de visualisation, mais aussi genèse discursive)	Analyser le texte en rapport avec la méthode : • mettre en jeu la notion d'amplitude d'un intervalle ; • parcourir un intervalle de **R** ; • aborder la recherche des bornes d'un intervalle encadrant une solution de l'équation $f(x) = 0$, pour une amplitude prédéfinie ; • calculer la dérivée de la fonction *f* et déterminer le signe de cette dérivée pour en déduire le tableau de variation de la fonction ; • analyser que la fonction est strictement monotone sur **R** (Genèse discursive)	• Certains élèves ont comparé l'algorithme de la phase 1 avec celui proposé dans cette phase. Ils repèrent que la boucle est un « répéter… Jusqu'à » qui n'est pas autorisée par *AlgoBox*. • Ils proposent un test d'arrêt « erronée » en particulier chez les élèves de Seconde Ces élèves restent dans un *ETA* (On observe une absence de travail en analyse qui permettrait de vérifier la validité du test par exemple). • D'autres élèves, en particulier dans le cycle terminal Scientifique analysent l'algorithme de la phase 2 pour déterminer le test d'arrêt et conclure à sa validité. Leur analyse met en jeu leur compréhension de certaines propriétés de la fonction (comme le fait qu'elle soit strictement croissante sur **R**). On observe ainsi que la validité de l'algorithme, question posée dans l'*ETA* résulte d'arguments relatifs à la méthode, qui relèvent donc de l'*ETM*.

Programmer l'algorithme dans un environnement informatique	Dans le cas d'un $ETAI_{ogiciel}$: • Traitement des variables (affectation, initialisation, mise à jour) • Passage d'une structure « Répéter… Jusqu'à » à une structure « TantQue » • Contraintes d'expression. Analyser la condition de d'arrêt : • Suivant une condition d'inégalité de la forme A > 0, A<0 ou A – B>0 quelle peut être l'assertion vérifiée en sortie ? (Genèse instrumentale) (Genèse discursive)	• Mettre en place une suite arithmétique de pas 0,001 pour définir la suite des valeurs de *x*. • Notation fonctionnelle • Calcul d'images par une fonction. (Genèse discursive)	• Des difficultés d'un point de vue informatique sont observées chez les élèves en particulier sur l'initialisation qui peut se trouver de manière erronée dans le corps de boucle. • De même, l'affectation d'une variable peut poser aussi des difficultés chez certains élèves, en particulier au niveau de la Seconde. • Une transposition des notations fonctionnelles à l'environnement informatique peut poser aussi problème lors de son utilisation pour des affectations, comme nous pouvons le voir chez des élèves de Seconde. Ces difficultés supposent un travail sur une genèse instrumentale dans l'$ETA_{logiciel}$, s'appuyant sur une discursive dans l'$ETA_{logiciel}$ et dans l'$ETM_{Analyse}$. • La condition de sortie est correctement présentée chez les élèves du cycle terminal Scientifique. Les élèves de ce niveau scolaire peuvent mettre en place une étude de la fonction avant d'utiliser l'algorithme et de voir ainsi la tâche comme une « application numérique » d'une méthode d'approximation d'une solution d'une équation de la forme $f(x)=0$. En effet, ces élèves peuvent observer que la fonction

			est croissante et que la condition d'arrêt de boucle peut se faire que sur l'image d'une valeur et non un produit d'images de deux valeurs. Cela suppose une genèse discursive dans l'$ETA_{logiciel}$ s'appuyant aussi sur l'$ETM_{Analyse}$.
Tester l'algorithme	Dans le cas d'un $ETAI_{ogiciel}$: • Comprendre les messages d'erreurs qui sont affichées quand il y a des erreurs de syntaxe par exemple ; • Savoir interpréter les résultats affichés lors de l'exécution de l'algorithme par l'environnement informatique utilisé. (Genèse instrumentale) (Genèse discursive)	• Cohérence avec l'encadrement trouvé antérieurement • Plausibilité du résultat (Genèses instrumentale et discursive)	

Tableau 35

2.5.4 Phase 3 : condition suffisante pour l'unicité ; dichotomie avec une fonction cachée

2.5.4.1. Le déroulement

Nous rappelons que dans cette troisième phase, l'algorithme de « dichotomie continue » distribué aux élèves est écrit dans la syntaxe du format *AlgoBox*. Il est mis à leur disposition directement sur l'ordinateur dans un fichier *AlgoBox*. Les élèves ont pour consigne d'ouvrir ce fichier et de ne pas accéder à l'onglet « Utiliser une fonction numérique » pendant toute la phase, afin de ne pas connaitre l'expression de la fonction « cachée » appelée par son identificateur F1 dans l'algorithme.

a) Reconnaissance de l'algorithme par les binômes

Nous observons que quel que soit le niveau scolaire, à ce stade de l'expérimentation les élèves reconnaissent tous la nature de l'algorithme qu'ils associent immédiatement à une méthode d'approximation d'une solution d'une équation de la forme F1(x) = 0, où F1 serait

l'identificateur de la fonction « cachée ». La plupart des binômes justifient que ce soit la méthode de dichotomie en faisant remarquer que dans l'algorithme apparaît le calcul de la moyenne de *a* et de *b*, et que l'instruction conditionnelle « Si… Alors…Sinon » située dans la boucle « TantQue » porte sur le produit des valeurs de F1(*b*) par F1((*a*+*b*)/2). Pour cela, les élèves renvoient à l'algorithme qui a été distribué par les enseignants au cours de la phase 1 de cette sous-ingénierie, bien que les deux algorithmes ne soient pas identiques. En effet, celui de la phase 1 fait porter la condition de l'alternative à l'intérieur de la boucle sur le produit des valeurs de F1(*a*) par F1((*a*+*b*)/2, ce qui change aussi les affectations dans les branches SI et SINON.

Les élèves identifient aussi les variables *a* et *b* comme étant respectivement les bornes Binf et Bsup de l'intervalle de départ. En se référant à la condition de continuation associée à la boucle « TantQue ». De même, l'ensemble des binômes présente la variable *e* de l'algorithme de cette phase 3 comme étant l'amplitude de l'encadrement de « la » solution de l'équation F1(x) = 0.

b) Exécution de l'algorithme par AlgoBox : possibilité ou pas d'une erreur dans l'écriture de l'algorithme initial affiché.

Aucun binôme n'éprouve de difficulté à faire exécuter l'algorithme par l'ordinateur. Ainsi, après son exécution par *AlgoBox*, l'ensemble des binômes indiquent les valeurs affichées par celui-ci. Cependant, 30% des binômes de Seconde et 10% de Première considèrent, à tort, que l'algorithme présente une erreur dans sa conception et prennent l'initiative de permuter les affectations dans les branches SI et SINON. Ils obtiennent un encadrement erroné, proche de la borne droite de l'intervalle. Après l'exécution de l'algorithme modifié, nous observons alors chez les binômes concernés des réactions différentes devant les valeurs affichées.

En effet, parmi les binômes de Seconde ayant modifié l'algorithme, nous constatons que 90% se contentent des résultats affichés par l'algorithme modifié et les considèrent comme répondant à la problématique de la recherche d'un encadrement plus fin de « la » solution de l'équation de F1(x) = 0. Ils concluent alors sous cette forme : *La solution de l'équation de F1(x) = 0 est le nombre réel x_0 compris entre 9,9999237 et 10*, sans procéder à la moindre vérification. En revanche, nous remarquons que les 10% de binômes de Seconde et les binômes de Première concernés par cette modification, procèdent à des vérifications.

Nous observons que ces 10% de binômes de Seconde modifient à nouveau l'algorithme

initial. Ils ajoutent ainsi deux nouvelles lignes après la boucle « TantQue » leur permettant d'afficher les calculs de F1(*a*) et de F1(*b*) (fig. 146), où *a* et *b* seraient les valeurs affichées par l'algorithme.

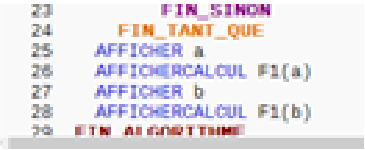


Figure 146 (Algorithme permettant d'afficher lors de son éxécution les calculs des images des bornes de l'intervalle par la fonction F1 dont les élèves ne connaissent pas l'expression)

Observant que les résultats renvoyés par *AlgoBox* pour ces F1(*a*) et F1(*b*) sont toutes les deux des valeurs strictement positives, ils en concluent que l'algorithme présenterait autre « chose » que la méthode de dichotomie, car sinon les deux valeurs devraient être de signes opposés d'après eux. A ce stade de la séance, nous observons un bref échange entre les binômes. Les binômes qui n'ont pas modifié l'algorithme initial font remarquer que dans l'algorithme de la phase 1, la condition était bien « $f(a)*f((a+b)/2)<0$ » et qu'ici la condition devient « $F1(b)*F1((a+b)/2)>0$ », et que par conséquent cela doit justifier qu'il est « logique » de ne pas modifier les affectations de *a* et *b* dans l'instruction conditionnelle. Cependant, nous constatons que les élèves n'arrivent pas à donner plus d'explication sur ce fait. Ainsi, faute de temps, l'enseignant se sent « obligé » d'intervenir auprès des binômes qui modifient l'algorithme initial. En effet, il leur confirme que l'observation faite par leurs camarades qui ne modifient pas l'algorithme est juste, sans donner pour autant d'explication supplémentaire. Il ajoute que leur première approche sur l'algorithme concernant une « application numérique » de la méthode de dichotomie est bien correcte et que par conséquent les résultats erronés affichés par *AlgoBox* ne viennent pas d'une erreur de l'algorithme initial mais de leur première modification. Malheureusement, ces binômes de Seconde n'ont pas le temps de reprendre leurs travaux et certains se contentent alors d'écrire sur leurs copies que l'algorithme est « faux » malgré les échanges entre les binômes et l'intervention de l'enseignant à la fin de la phase (cf. l'extrait de copie fig. 147). N'ayant pas eu plus de retour sur ce point, nous ne savons pas si la conclusion de ces binômes qui écrivent : *l'algorithme est bien faux*, signifie pour eux que ce serait l'algorithme initial qui serait faux ou l'algorithme modifié.

2nde 12 Lundi 6 octobre 2014

Binôme n° 12

Exercice 1

1. Nous avons l'algorithme de dichotomie car le test est proche de celui vu au début de la séance quand Monsieur [redacted] a distribué le premier algorithme. Nous voyons que le test porte sur F1(b) fois F1((a+b)/2). Dans le premier algorithme le test était F1(a) fois F1((a+b)/2). Nous pensons qu'il faudrait changer les valeurs de b et de a dans le tantque. Mais on fait intervenir la moyenne de a et b comme pour la dichotomie. Cet algorithme est fait pour calculer la solution d'une équation et les valeurs affichées sont les nombres qui encadrent cette solution. Le nouveau a sera la valeur petite et le nouveau b sera la valeur grande du nouveau intervalle.
2. a) et b) Les deux valeurs affichées sont 9,9999237 et 10. (Nous avons corrigé l'algorithme en remplaçant b par a et c par b dans le tantque).

c) D'après l'algorithme, la solution est comprise entre 9,9999237 et 10.

3. La courbe affichée après modification de l'algorithme permettant d'afficher la courbe de F1 est :

La courbe coupe deux fois la droite des x. Il n'y a pas qu'une solution.
L'algorithme est faux.
On a ajouté deux lignes au programme du départ permettant de calculer F1(9,9999237) et F1(10). Ces deux valeurs sont de même signe.
On constate qu'il y a bien une erreur car ces deux valeurs n'entourent pas 0. L'algorithme est bien faux.

Figure 147 (Une copie d'élève de 2nde sur la dichotomie avec la fonction « cachée » admettant plusieurs zéros sur l'intervalle d'étude)

En ce qui concerne les binômes de Première qui ont procédé à une modification de l'algorithme initial, nous observons bien que les résultats affichés par *AlgoBox* ne leur conviennent pas. Comme pour les binômes de Seconde se trouvant confrontés à cette problématique, ils font alors afficher les résultats de F1(*a*) et F1(*b*) pour les deux valeurs *a* et *b* trouvées. L'enseignant ayant pour consigne de ne pas intervenir, s'ensuit alors une discussion entre les différents binômes de la classe. Ainsi, parmi les binômes qui n'ont pas modifié l'algorithme initial, certains font remarquer que contrairement à l'algorithme de la phase 1, où la condition est « $f(a)*f((a+b)/2)<0$ », dans celui de cette phase 3 le calcul de

l'image de *a* par F1 est remplacé par celui de l'image de *b*, et que par conséquent il est « normal » de ne pas modifier les affectations de *a* et *b* dans l'instruction conditionnelle. En effet, certains de ces binômes, pour illustrer leurs propos sur le fait qu'il ne serait pas nécessaire de modifier les affectations des variables *a* et *b*, proposent une représentation graphique (Fig. 148) montrant une fonction F1 sur un intervalle [*a* ; b], avec changement de signe où ils font apparaître à l'aide de pointillés le point de coordonnée ((*a*+*b*)/2 ; *f*((*a*+*b*)/2)) (Fig. 148) pour illustrer leur raisonnement.

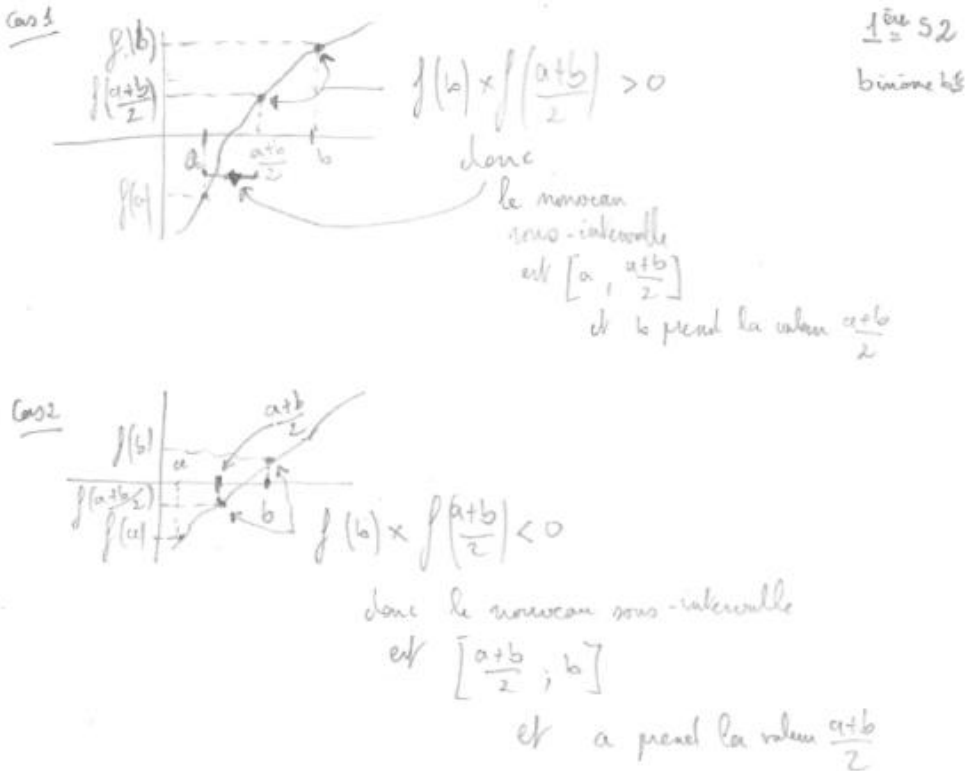


Figure 148 (Extrait de copie où tout est en jeu sur les registres et la sémiotique)

Suite à ces échanges, les binômes qui ont procédé à une modification de l'algorithme initial, acceptent les remarques apportées par leurs camarades et reprennent ainsi les affectations de *a* et *b* qui sont proposées au départ tout en laissant affiché les valeurs de F1(a) et de F1(b) pour les valeurs de a et b données à la fin de l'exécution de l'algorithme. Les valeurs étant de signes opposés les rassurent et ils passent à l'interprétation des résultats affichés.

c) Interprétation des valeurs affichées par l'exécution de l'algorithme

Quels que soient les résultats affichés dans un premier temps par l'exécution de l'algorithme, nous observons que la très grande majorité des binômes des différents niveaux (environ 80% de tous les binômes) considèrent que les valeurs affichées représenteraient bien

un encadrement de « la » solution de l'équation F1(*x*) = 0 sur l'intervalle du départ [0 ; 10]. Cependant, 30% des binômes de Terminale indiquent que cette réponse pourrait être remise en cause car ils signalent que l'algorithme ne permet pas de connaitre la monotonie de la fonction F1, sachant que cette fonction est « cachée ». Bien que donnant une réponse du type *l'affichage représente un encadrement de « la » solution de l'équation F1(x) = 0*, ils ajoutent comme commentaire sous réserve que la fonction F1 soit strictement monotone. 10% d'entre eux précisent même que cela renvoie au « corollaire du TVI » (c.-à-d. le *théorème de la bijection*) et non au TVI en lui-même. Aucun binôme ne prend l'initiative de faire afficher les valeurs successives des variables Binf et Bsup.

d) Quel retour sur l'interprétation des valeurs affichées après la représentation graphique de la fonction F1 ?

Conformément à la consigne, les élèves ajoutent à l'algorithme une boucle faisant afficher une représentation graphique de la fonction cachée F1 sur l'intervalle du départ. Nous observons que les binômes procèdent sans difficulté particulière à cette tâche. Les seules variations observées dans les algorithmes sont autour du choix fait sur les noms des variables supplémentaires à ajouter, ainsi que sur la valeur donnée au « pas » pour la suite des valeurs des abscisses des points de la représentation. En effet, suivant la valeur choisie pour ce pas, la représentation est plus ou moins précise, car la construction proposée par *Algobox* est faite à partir de segments (Extrait de représentations graphiques données par les élèves fig. 149).

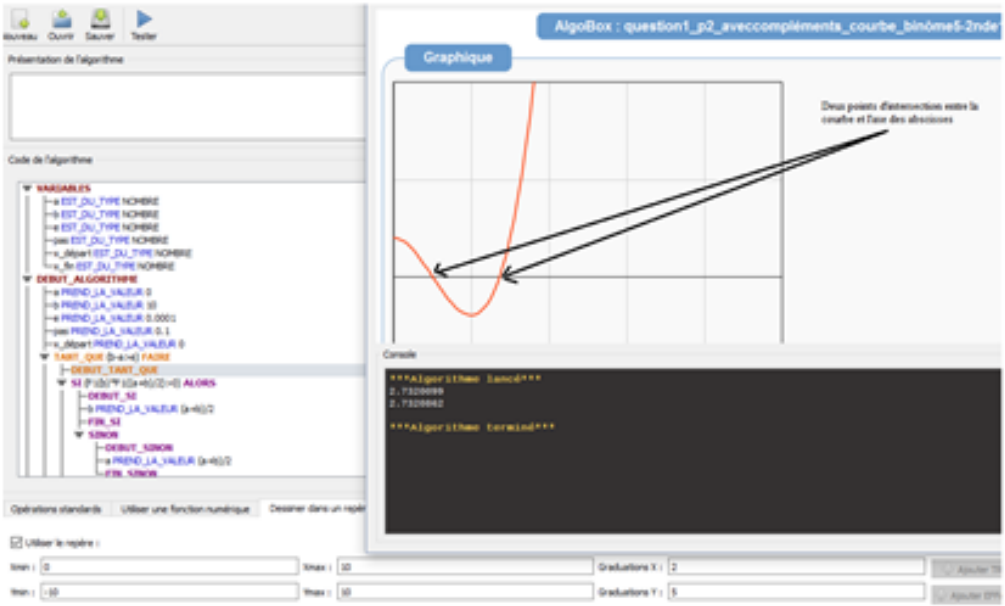
Figure 149 (Diverses représentations graphiques obtenues avec AlgoBox de la fonction « cachée » admettant plsuieurs zéros sur l'intervalle étudié)

Nous observons que quel que soit le niveau de scolarité, les binômes reconnaissent la possibilité d'un « problème » (dixit les élèves) entre le fait que l'encadrement affiché par l'exécution de l'algorithme supposerait une unique solution à l'équation F1(x) = 0 sur l'intervalle du départ et la représentation graphique de la fonction F1 sur ce même intervalle. Cependant, les réactions ne sont pas les mêmes suivant le niveau des connaissances scolaires

de l'élève en mathématique.

Ainsi, au niveau de la Seconde, nous observons que devant cette contradiction observable à partir de la représentation graphique et le fait que l'algorithme donnerait un encadrement de « la » solution de l'équation F1(x) = 0 sur l'intervalle [0 ; 10], certains binômes semblent être dans l'incapacité de trouver une réponse autre que celle d'un algorithme initial qui pourrait être faux (Fig. 150). Ce ressenti est assez majoritaire chez les binômes de ce niveau scolaire (près de 60% des binômes de Seconde). Par ailleurs, parmi ces binômes certains remarquent que l'encadrement proposé correspond à l'encadrement de l'abscisse du premier point commun entre la représentation graphique C_{F1} de F1 et l'axe des abscisses.

Nous avons la courbe suivante :



Il y a deux points d'intersection entre la courbe et l'axe des abscisses.
Nous pensons que l'algorithme est incorrect, car on devrait avoir qu'un seul point commun.

Figure 150 (Copie d'élèves de 2nde avec proposition d'une conclusion fausse dans le cas où la fonction « cachée » admet plusieurs zéros sur l'intervalle étudié)

Un binôme propose de faire tracer par le logiciel les points d'abscisses Binf et Bsup de C_{F1} (Fig. 151).

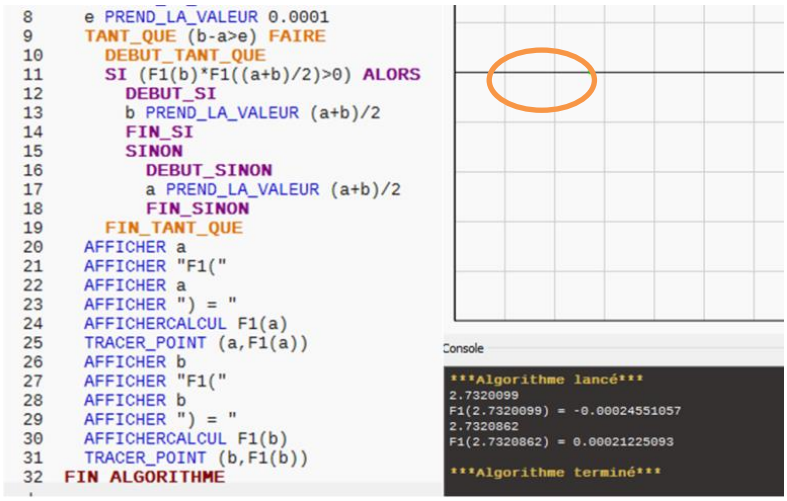


Figure 151 (Copie d'élèves de 2nde avec traçage des points de d'abscisses les bornes sup et inf de la courbe représentative de la fonction F1)

Les élèves de ce binôme indiquent alors : *La solution de la solution est donc bien comprise entre 2,7320099 et 2,7320862. Et l'amplitude de l'intervalle [2,7320099 ; 2,7320862] est 0,0000763 qui est plus petite que 0,0001, ce qui explique que la figure nous montre qu'un seul point. Les points sont presque confondus* (Extrait de la copie). Puis, nous observons qu'ils font tracer la représentation graphique de F1 (Fig. 152) et concluent : *Point encadré par les valeurs 2,7320099 et 2,7320862, correspondant à F1(x) = 0. Le problème est que nous avons deux points d'intersection entre la courbe et la droite des x. Donc, nous ne pouvons pas conclure sur l'algorithme. Il est peut-être faux* (Extrait de la copie).

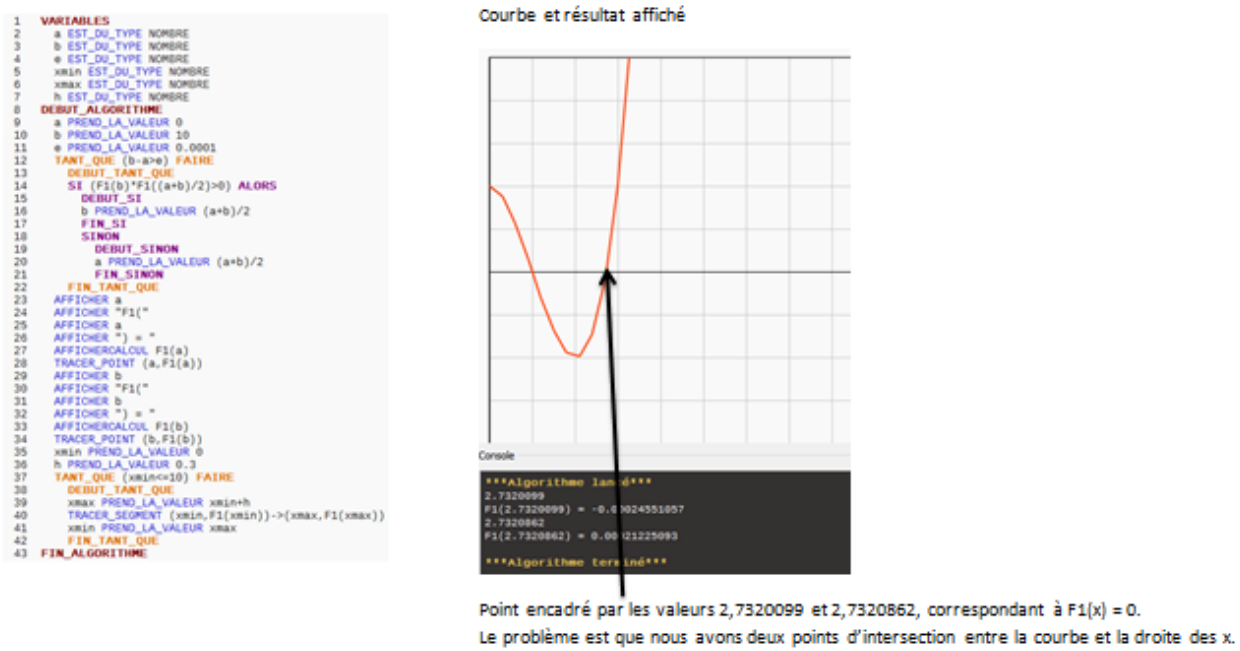


Figure 152 (Copie d'élèves dans le cas de la fonction « cachée » admettant plusieurs zéros sur l'intervalle d'étude)

Aux niveaux du cycle terminal Scientifique, les représentations graphiques affichées par l'environnement *AlgoBox* après que les binômes aient complété leurs algorithmes sont sensiblement les mêmes que celles observées en Seconde. Ce qui différencie les élèves du cycle scientifique de ceux de Seconde sont leurs interprétations de cet affichage donné par *AlgoBox*, et par conséquent la correspondance aux valeurs numériques affichées par *AlgoBox*. En effet, nous constatons que les binômes observent aussi les graphiques affichés par l'environnement *AlgoBox*, et considèrent alors que l'algorithme ne propose qu'un seul encadrement mais qu'en fait il y aurait dû avoir deux encadrements car la représentation de la fonction F1 présente deux points d'intersection avec l'axe des abscisses sur l'intervalle de départ, et que par conséquent l'équation F1(x) = 0 devrait avoir deux solutions sur l'intervalle de départ. De plus, aucun des binômes du cycle scientifique ne propose que l'algorithme puisse être faux.

Cependant, au niveau de la Première, nous observons que les élèves n'arrivent pas à aller plus loin dans leurs analyses, tandis qu'en Terminale, nous constatons que les élèves cherchent à expliquer cet état de chose. Ainsi, au niveau Terminale, conformément à ce qui est déjà observé lors de la première interprétation faite avant la représentation graphique de la fonction sur les résultats affichés, ces élèves font remarquer que cet algorithme ne peut être mis en application qu'une fois vérifiées certaines propriétés de la fonction F1. Nous remarquons alors que certains binômes rappellent sur leurs copies l'ensemble des propriétés (continuité, stricte monotonie et changement de signe) que devraient vérifier la fonction étudiée avant d'utiliser l'*algorithme de dichotomie*. Par ailleurs, à ce niveau scolaire, nous observons que les élèves se réfèrent aussi au travail fait lors de la phase 1 de cette première partie de la sous-ingénierie. En effet, ils signalent que la fonction de la phase 1 a été étudiée avant de procéder à l'« application numérique » qui a permis d'obtenir un encadrement de l'unique solution de l'équation $f(x) = 0$ sur l'intervalle du départ. De plus, 10% des binômes de Terminale indiquent que l'algorithme de « dichotomie continu » ne permet pas de justifier l'unicité d'une solution mais que ce serait seulement un outil de calcul d'une valeur approchée ou d'un encadrement d'une solution d'une équation de la forme $f(x) = 0$, dont l'exploitation serait pertinente seulement après justification de l'existence et de l'unicité de cette solution. Nous constatons alors que l'enseignant demande à ces élèves pourquoi ils précisent l'« existence » et pas seulement l'« unicité ». Les élèves répondent alors que dans le TVI, il y a

l'existence d'au moins une solution et que dans le *théorème de la bijection*, corollaire du TVI, il y a en plus l'unicité, mais que pour eux, ces deux théorèmes ne sont pas en lien direct avec l'*algorithme de dichotomie*.

Nous observons même qu'un binôme signale que le travail fait sur l'*algorithme de dichotomie* leur rappelle le travail fait en Troisième sur l'*algorithme d'Euclide*. En effet, pour expliquer cette comparaison, les élèves de ce binôme ajoutent deux commentaires concernant ce renvoi à l'*algorithme d'Euclide* : (1) on n'aurait pas besoin de travailler avec l'*algorithme d'Euclide* si on sait que les nombres du départ sont premiers entre eux ; (2) de plus, l'utilisation de l'*algorithme d'Euclide* ne se ferait que comme un « outil » de calcul de PGCD car cet algorithme ne fournirait aucune explication sur le fait que ce serait bien le résultat du PGCD des deux nombres entiers du départ que l'on obtiendrait après une utilisation de l'algorithme.

2.5.4.2. Analyse *a posteriori*

Comme nous l'avons vu lors du déroulement, nous avons observé différentes réactions de la part de certains binômes. Ainsi, l'ensemble des binômes, quel que soit le niveau scolaire montre à ce stade de l'expérimentation que le travail fait lors des phases précédentes leur donne une plus grande autonomie dans la reconnaissance de l'algorithme proposé. Par ailleurs, lorsqu'ils proposent une justification de cette reconnaissance de l'algorithme, ils se réfèrent en général à l'« application numérique » faite au cours de la phase 1 de cette première partie de la sous-ingénierie. De même, les fonctionnalités et la syntaxe de l'environnement *AlgoBox* sont bien maitrisées par l'ensemble des binômes à ce stade de cette ingénierie sur la dichotomie. Le travail fait au cours des phases précédentes leur permet d'avoir plus d'autonomie dans l'utilisation de l'environnement informatique et pour certains de prendre des initiatives sur les possibilités qu'offrent cet environnement, en particulier lors des observations et des critiques des résultats que donnent l'ordinateur. Les consignes et les tâches demandées sont comprises par les binômes même si certains commettent des erreurs sur la compréhension des résultats affichés par l'algorithme initial comme nous avons pu le constater lors du déroulement. En effet, les premières erreurs venant de certains binômes ont lieu après l'exécution de l'algorithme initial et non sur le rôle que peut jouer celui-ci, même si après son exécution des binômes supposent que l'algorithme pourrait être faux.

Nous proposons d'affiner notre analyse *a posteriori* de cette phase en nous référant aux

Espaces de Travail Mathématique et Algorithmique. Dans cette troisième phase, les élèves doivent de fait coordonner deux genèses discursives dans deux espaces de travail parallèles un $ETA_{AlgoBox}$ et un $ETM_{analyse}$, ainsi que deux genèses l'une instrumentale et l'autre sémiotique dans ces mêmes espaces de travail parallèles. Ils sont en effet amenés à réfléchir, à travers l'investigation de la recherche de l'existence d'une (ou des) solution(s) d'une équation de la forme $f(x) = 0$, à la détermination d'un encadrement d'une de ces solutions, et à faire le lien avec l'idéalité de l'application de l'algorithme de « dichotomie continu » pour déterminer « exactement » ces encadrements.

Une présentation sous forme d'un tableau (Tableau n° 36) va nous permettre de présenter de façon plus synthétique, les genèses de convergences entre les deux Espaces de Travail parallèles mis en jeu au cours de cette phase.

	***ETA**$_{AlgoBox}$*	***ETM**$_{Analyse}$*	**Commentaires afin de montrer que les deux Espaces de Travail sont porteurs d'idées intéressantes**
Comparer les algorithmes de la phase 1 et de la phase 3.	Analyser le texte d'un point de vue algorithmique : • Condition dans l'alternative • Branches « SI et SINON » (Genèse discursive)	Analyser le texte en rapport avec la méthode : • Changement de signe dans l'intervalle. • Evolution des valeurs des bornes. (Genèse discursive)	• Certains élèves comparant les algorithmes, repèrent une interversion de *a* et *b* dans la condition, et donc ils pensent qu'il y a besoin d'une nécessité erronée d'intervertir les branches. Ils ne considèrent pas le sens de l'inégalité. Nous interprétons cela comme une réduction à l'*ETA* (pas de travail en analyse). • D'autres élèves analysent l'algorithme de la phase 3 pour conclure à sa validité. Leur analyse met en jeu leur compréhension du principe de découpage de l'intervalle dans la méthode et de la signification de l'inégalité (comme changement ou non de signe sur l'intervalle). Parmi eux, certains l'appuient sur un travail graphique. La validité de l'algorithme, question posée dans l'*ETA* résulte d'arguments relatifs à la méthode, qui relèvent donc de l'*ETM*. • Notons qu'aucun binôme

			ne repère l'équivalence des conditions dans les deux algorithmes dans le cas d'une fonction monotone qui permettrait de valider directement l'algorithme de la phase 3. Cela supposerait une genèse discursive dans l'$ETA_{Algobox}$ (si les conditions sont équivalentes, il ne faut pas changer les branches) s'appuyant aussi sur l'$ETM_{Analyse}$ (« pas de changement de signe sur le sous-intervalle de droite » est équivalent « changement de signe sur le sous-intervalle de gauche).
Représentation graphique et interprétation	Construire des représentations graphiques : points, ensemble de points. (Genèse instrumentale)	• Repérer la non monotonie et la double intersection entre la représentation graphique de la fonction et l'axe des abscisses. (Genèses sémiotique et visualisation) • Construire la suite des *x* en choisissant le pas. (Genèse instrumentale) • Reconnaitre la nature arithmétique de cette suite des *x* en particulier au niveau du cycle terminal Scientifique. (Genèse discursive)	• Les gestes de construction et de visualisation à mettre en place sont familiers, chez des élèves de lycée qui ont l'habitude d'utiliser une calculatrice graphique. • Par rapport à une calculatrice graphique, la représentation doit être programmée par une itération. C'est aussi un geste familier. • Une utilisation du mode graphique de la calculatrice ou des possibilités de construction de représentations graphiques qu'offre l'environnement *AlgoBox*, • La variable « pas » n'est pas assez pertinente ici, car les zéros de la fonction sont assez éloignés. Il serait ainsi pertinent de reconsidérer le choix d'une fonction ayant des zéros très proches pour permettre aux élèves de mieux appréhender le choix d'une variable « pas » afin qu'ils puissent concevoir la nécessité d'affiner l'aspect « visualité » de la représentation graphique. Ceci amènerait l'enseignant à laisser plus de temps sur cette

			problématique de définir un « pas » adéquat, et par conséquent de permettre une discussion entre les élèves
Exploitation de l'intervalle rendu par l'algorithme	Analyser la condition de continuation : • avec une condition d'inégalité de la forme $A > 10^{-4}$, quelle est l'assertion vérifiée en sortie ? • avec une condition d'inégalité de la forme $A = 10^{-4}$, quelle est l'assertion vérifiée en sortie ? (Comme dans les entiers) • avec une condition d'inégalité de la forme $A \leq 10^{-4}$, quelle est l'assertion vérifiée en sortie ? (Négation) (Genèse discursive)	Interpréter : • la notion d'encadrement à 10^{-p} près ; • la valeur décimale approchée à 10^{-p} près. (Genèse discursive)	• L'intervalle obtenu est [2,7320099 ; 2,7320862]. • Certains considèrent qu'il convient et le donnent comme résultat, parfois en calculant l'amplitude. • D'autres considèrent ce résultat comme insuffisant. Ils le « corrigent » en élargissant l'intervalle à [2,7320 ; 2,7321] de façon à obtenir de nouvelles bornes, et un intervalle de longueur 10^{-4}. • Dans le premier comportement, une amplitude inférieure à 10^{-4} n'est pas remise en cause. On reste dans l'*ETA*. • Nous mettons en relation le second comportement avec une attitude de doute vis-à-vis de l'*ETA* que les élèves « contrôlent » dans l'*ETM*. • Nous ne pouvons en inférer davantage sur les genèses dans l'*ETA* (négation de la condition de continuation) et dans l'*ETM* (coordination intervalle d'amplitude 10^{-p} et valeur décimale approchée).
Initiatives prises par les élèves	Ajouter : • les images des bornes de l'encadrement ; • la construction de points particuliers de la représentation graphique de la fonction aux bornes de l'encadrement (Genèse instrumentale)	• Reconnaitre à l'aide de la représentation graphique de la fonction, les coordonnées d'un point de la représentation. (Genèses sémiotique, visualisation et instrumentale) • Calculer l'image d'un nombre réel par une fonction. (Genèse instrumentale)	• Initiative prise en fonction d'un objectif dans l'*ETM*, impliquant une construction dans l'*ETA*
Interpréter l'obtention d'un	Du point de vue des propriétés de	Du point de vue de la méthode :	La représentation graphique de la fonction « cachée » par

encadrement d'un seul zéro alors qu'il y en a deux sur l'intervalle du départ	l'algorithme : • supposer que l'algorithme proposé serait faux ; • considérer que l'exécution d'un algorithme ne justifie pas le résultat. (Cas de l'*algorithme d'Euclide* par exemple) (Genèse discursive)	• nécessité de conditions pour l'unicité. (Genèse discursive)	*AlgoBox* interpelle les élèves et leur permet de remettre en cause le résultat affiché sur l'encadrement, même si cette remise en cause peut être erronée (renvoi au fait que l'algorithme initial pourrait être faux). • Dans le cadre d'un *ETA*, les élèves questionnent l'algorithme. • Dans le cadre d'un *ETM*, les élèves questionnent la méthode. Les terminales s'appuient sur les conditions suffisantes d'unicité, comme prérequis à l'utilisation de l'algorithme (perspective d'une « application numérique » qui privilégie l'*ETM*). Les autres sont davantage dans l'interrogation et considèrent des propriétés générales de l'algorithme (privilégiant l'*ETA*). Les élèves, qui ont un algorithme faux et obtiennent l'intervalle [9,9999237 ; 10], ne remettent pas en cause ce résultat, mais considèrent seulement l'unicité. Pour ces élèves, il n'y a pas de retour sur l'*ETA*, leur visualisation est ainsi insuffisante. Aucun élève ne s'intéresse au comportement précis de l'algorithme dans le cas de plusieurs changements de signes sur l'intervalle (pourquoi ce zéro et pas l'autre ?) qui impliquerait de coordonner *ETA* et *ETM*. Il aurait été aussi possible de proposer une analyse de l'exécution montrant quel zéro va être encadré. Comme alternative, la possibilité de choisir d'autres fonctions avec des zéros répartis différemment.

Tableau 36

A ce stade de l'expérimentation, nous pouvons observer que le travail fait depuis le début de l'ingénierie sur la dichotomie est porteur de prises de conscience chez une majorité

d'élèves sur la nécessité d'articuler différents Espaces de Travail : $ETA_{AlgoBox}$ et $ETM_{analyse}$ en parallèle afin de répondre à un problème donné sur la résolution approchée d'une équation de la forme $f(x) = 0$. En effet, le travail suivi par les élèves semble être vecteur d'une meilleure compréhension de la mise en place d'un algorithme d'approximation comme « application numérique » mais aussi comme objet d'« analyse numérique ».

De plus, nous constatons que les outils issus du domaine de l'analyse nécessaires à mettre en place pour introduire une « application numérique » de recherche d'une solution approchée d'une équation semblent ne plus être une source d'incompréhension pour la quasi-totalité des élèves du cycle terminal Scientifique. Cependant, au niveau de la Seconde, nous observons encore de la part de certains élèves certaines résistances sur ce point. En effet, ces élèves semblent se contenter d'une simple remise en cause de la validité de l'algorithme si ce dernier est porteur de paradoxes.

Tout ceci va être confirmé et approfondi par le travail fait dans la phase 4 et qui fait l'objet d'une description et d'une analyse qui vont être présentées dans la section suivante.

2.5.5 Phase 4

2.5.5.1. Le déroulement

Le contexte de cette phase de la première partie de la sous-ingénierie « dichotomie continue » est identique à celle de la phase précédente. En effet, chaque binôme reçoit de la part de l'enseignant un énoncé papier où est écrit entre autres l'algorithme de « dichotomie continue » dans la syntaxe de l'environnement *AlgoBox* et les questions auxquelles le binôme doit répondre. De même, comme pour la phase 3, cet algorithme est mis à la disposition des élèves directement sur l'ordinateur dans un fichier *AlgoBox*. Les élèves ont toujours pour consigne d'ouvrir le fichier et de ne pas accéder à l'onglet « Utiliser une fonction numérique » pendant toute cette phase, afin de ne pas connaitre l'expression de la fonction « cachée » appelée par son identificateur F1 dans l'algorithme.

a) Reconnaissance de l'algorithme par les binômes

Tous les binômes devant répondre à des questions semblables à celles de la phase 3, nous observons que les élèves répondent très rapidement à la question concernant les objectifs de

l'algorithme. Ainsi, nous avons comme types de réponse[88] :

(B1_2nde) « C'est l'algorithme de dichotomie car nous retrouvons une structure proche de celles vues depuis le début du travail sur la dichotomie. » (Binôme de seconde)

(B2_2nde) « On divise l'intervalle en deux sous-intervalles de même longueur et on vérifie lequel répond à la condition « F1((*a*+*b*)/2)>0 » puis suivant que c'est vrai ou pas on garde tel ou tel sous-intervalle. » (Binôme de Seconde)

(B3_2nde) « On utilise le milieu de l'intervalle à chaque étape, et on réduit l'intervalle de moitié. Donc c'est l'algorithme de dichotomie comme pour l'exercice précédent. » (Binôme de Seconde)

(B1_1er) « Nous avons une troisième forme de condition dans le « SI », mais nous pensons que c'est toujours un travail sur la dichotomie, car cette condition fait aussi intervenir l'image du centre de l'intervalle [*a* ; *b*] et les affectations sont semblables à celles des cas précédents. Ces affectations permettent alors de réduire à chaque fois l'intervalle de moitié. Il faut cependant vérifier si l'encadrement que nous obtiendrons correspondra bien à un encadrement d'une solution de l'équation F1(*x*)=0 sur l'intervalle [1 ; 2]. » (Binôme de Première Scientifique)

(B2_1er) « Nous pensons que c'est probablement l'algorithme de dichotomie, mais qu'il faut connaitre la fonction pour savoir si c'est bien l'encadrement d'une solution de l'équation F1(*x*)=0, ou si il y aurait d'autres solutions sur l'intervalle du départ [1 ; 2] et par conséquent d'autres encadrements » (Binôme de Première Scientifique)

(B3_1er) « L'algorithme de dichotomie ne peut être utilisé qu'après avoir vérifié des conditions d'unicité et d'existence de la solution de l'équation sur l'intervalle [1 ; 2]. Ne connaissant pas si cela est vrai pour le F1, nous ne pouvons pas conclure directement sur le résultat que va donner la machine. Nous n'avons pas l'expression de la fonction. Nous devons la représenter pour savoir comment interpréter l'encadrement qui sera affiché. » (Binôme de Première Scientifique)

(B4_1er) « L'algorithme n'a pas la même condition dans le « SI… ALORS… SINON » mais les affectations dans cette instruction conditionnelle sont comme celles de l'algorithme de l'exercice précédent ainsi que dans celui distribué par le prof au début de la séance, donc nous pensons que c'est bien à nouveau l'algorithme de dichotomie. Mais nous ne savons pas si les valeurs finales de *a* et *b* qui seront affichés encadreront bien une solution unique de l'équation. Peut-être que l'équation F1(*x*)=0 admet d'autres solutions que l'algorithme

[88] Extraits non exhaustifs de réponses proposées par des binômes des niveaux Seconde et Première Scientifique.

ne donne pas, car nous n'avons pas de renseignements sur la fonction. » (Binôme de Première Scientifique)

Par ailleurs, nous observons qu'au niveau de la Terminale Scientifique, les réponses données par les binômes semblent indiquer qu'ils vont plus loin dans leurs raisonnements. En effet, les binômes de ce niveau scolaire proposent des réponses plus précises sur les conditions que devraient vérifier la fonction « cachée » pour conclure sur la nature des résultats qui seront affichés après l'exécution de l'algorithme dans l'environnement *AlgoBox*. Nous avons par exemple des réponses[89] comme :

(B1_TS) « Nous pouvons supposer que nous avons de nouveau un algorithme de dichotomie permettant de déterminer un encadrement d'une solution de l'équation F1(x)=0 sous réserve que cette solution existe bien. Il est possible que l'algorithme ne donne pas toutes les solutions dans l'intervalle du départ [1 ; 2] sous réserve qu'il y ait bien au moins une solution dans cet intervalle. Nous devons connaitre si la fonction F1 est définie, continue et connaitre sa monotonie, puis s'il y a changement de signe afin d'affirmer que le résultat qui sera affiché après l'exécution de l'algorithme correspond bien à un encadrement d'une solution de l'équation F1(x)=0. » (Binôme de Terminale S)

(B2_TS) « Comme pour les cas déjà vus depuis le début du travail sur la dichotomie, nous voyons que l'algorithme proposé correspond bien à la méthode de dichotomie. En nous référant au travail fait sur l'exercice 1[90], nous observons que la condition écriture dans la structure conditionnelle ne fait qu'intervenir l'image de (a+b)/2 par la fonction F1. Nous supposons que cela signifie que la fonction F1 est strictement croissante sur l'intervalle [1 ; 2]. Par conséquent, si c'est bien le cas, nous pensons que si en plus la fonction F1 est aussi continue sur l'intervalle [1 ; 2], l'encadrement qui sera affiché par l'ordinateur correspondra à celui de l'unique solution de l'équation F1(x)=0. Pour vérifier la continuité de F1 sur l'intervalle [1 ; 2], nous proposons de tracer la représentation graphique de la fonction sur cet intervalle. » (Binôme de Terminale S)

(B3_TS) « Nous avons un algorithme qui affiche les valeurs des images de F1(a) et de F1(b) pour chaque valeurs de a et b qui[91] des bornes des intervalles. Ces bornes a et b sont déterminées par la condition portant sur la conditionnelle « SI ». Nous avons aussi une condition qui porte sur le signe de l'image F1((a+b)/2) dans le test de la structure « SI ».

[89] Extraits non exhaustifs de copies de binômes de Terminale Scientifique.
[90] Exercice de la phase 3 (N.D.L.R.)
[91] Le texte reproduit ici correspond exactement à ce que le binôme a rendu à la fin de la séance (N.D.L.R.)

Nous pouvons supposer que ceci signifie que la fonction est strictement monotone sur l'intervalle étudié du départ. Ceci permet alors d'éviter de programmer une condition portant sur le calcul d'une image d'une des bornes de l'intervalle [*a* ; *b*], ainsi que le produit de ces images, comme cela avait été fait dans l'algorithme du début de la séance et de l'exercice précédent. Par conséquent, cet algorithme correspond bien à la dichotomie. Mais, d'après ce que nous avons vu précédemment, nous devons contrôler si la fonction F1 vérifie les propriétés du TVI pour l'existence d'au moins une solution sur l'intervalle [1 ; 2] et du TDB[92] pour l'unicité d'une solution sur cet intervalle. L'algorithme propose le calcul des F1(*a*) et F1(*b*) pour chaque nouvel intervalle. Nous supposons que cela permet de répondre aux questions de l'existence et de l'unicité avant même d'avoir tracé la courbe de la fonction F1. » (Binôme de Terminale S)

(B4_TS) : « C'est l'algorithme de dichotomie. Il propose de déterminer les images successives des bornes des différents sous-intervalles qui vont intervenir afin de vérifier à quoi correspond l'encadrement final. Nous pensons que cela permet de répondre au fait si l'encadrement final correspond à un encadrement d'une unique solution de l'équation F1(x)=0, ou si comme pour l'exercice précédent la solution n'est pas unique. Mais, comme le test dans l'instruction conditionnelle ne porte que sur le signe de F1((*a*+*b*)/2), nous supposons que cela signifie que la fonction F1 doit être strictement monotone et que par conséquent, il n'y aurait qu'une seule solution sur [1 ; 2]. Sans avoir fait tourner l'algorithme, nous ne pouvons toutefois dire totalement que cette solution est unique. Peut-être, que l'équation n'admet pas de solution car nous n'aurions pas d'encadrement affiché. Dans les conditions du TVI, il faut aussi vérifier la continuité de la fonction sur l'intervalle d'étude, et l'algorithme de dichotomie ne donne aucun renseignement sur la continuité. En conclusion, nous devons aussi représenter la fonction sur l'intervalle [1 ; 2] pour voir si la représentation graphique de la fonction se fait sans lever le crayon. L'algorithme de dichotomie devrait normalement terminer le problème et non le commencer, comme nous l'avons dans le premier exercice avec l'organigramme. » (Binôme de Terminale S)

A ce stade de l'expérimentation, nous observons que l'ensemble des binômes prend bien conscience qu'une utilisation de l'*algorithme de dichotomie* demande des vérifications sur certaines des propriétés de la fonction avant de conclure sur les résultats que pourraient donner l'exécution de cet algorithme.

[92] Théorème de la bijection

Nous observons aussi que les variables *a* et *b* sont considérées par les élèves comme étant les bornes Binf et Bsup des intervalles successifs. Cependant, aucun élève ne cite la condition de continuation associée à la structure « TantQue ». Les élèves font comme si c'était implicite. Ainsi, contrairement à la phase 3, les élèves ne parlent pas ici de l'amplitude de l'encadrement final. Tout reste dans l'implicite.

b) Exécution de l'algorithme par AlgoBox et interprétation des valeurs affichées par l'exécution de l'algorithme

Comme pour la phase 3, les élèves des différents binômes des trois niveaux scolaires n'éprouvent aucune difficulté à ouvrir le fichier *AlgoBox* et à faire exécuter l'algorithme par AlgoBox.

Tous les binômes rendent une image de l'écran d'ordinateur affichant les résultats des dernières exécutions de la boucle « TantQue » (fig. 153).

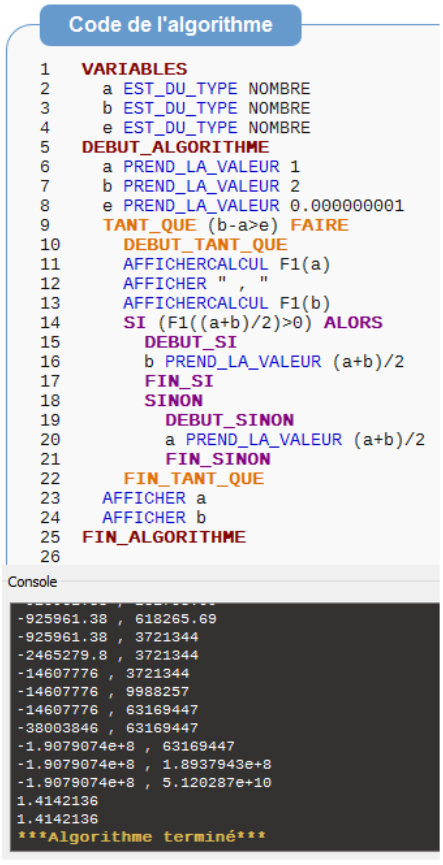


Figure 153 (Exemple d'implémentation dans AlgoBox de l'algorithme de la fonction « cachée » non définie en un point de l'intervalle d'étude)

Nous observons que tous les binômes indiquent que ce n'est pas exactement un encadrement qui est affiché, une fois que l'exécution de l'algorithme est terminée. Nous

constatons alors que les interprétations des résultats peuvent être très variées, en particulier suivant le niveau scolaire.

En Seconde, tous les binômes concluent que les valeurs finales affichées de a et b sont « 1,4142136 » et que par conséquent l'équation F1(x)=0 admet une unique solution qui serait « presque égale » (dixit les élèves) à 1,4142136. Nous pouvons observer qu'aucun binôme ne semble regarder l'affichage des valeurs de F1(*a*) et F1(b) pour les différentes valeurs successives des bornes *a* et *b*. En effet, aucune copie de Seconde ne fait allusion à ce fait. Il faut attendre la construction de la représentation graphique de la fonction F1 pour avoir des questionnements de la part de ces élèves comme nous le verrons dans la partie suivante, sur l'interprétation des valeurs affichées après la représentation graphique de la fonction F1.

Au niveau du cycle terminal Scientifique, nous avons des conclusions très différentes de celles observées en Seconde. En effet, à ce niveau une très grande majorité des binômes (95% en Première et 100% en Terminale) font références aux valeurs finales affichées par l'ordinateur mais aussi aux différentes valeurs de F1(*a*) et F1(*b*). Nous observons que les élèves de Première constatent que les valeurs qui précédent les deux dernières valeurs égales toutes les à 1,4142136, sont de « plus en plus petites » pour les F1(*a*) et de « plus en plus grandes » pour les F1(*b*). Les élèves de Terminale constatent aussi ce même fait mais ayant des connaissances sur des questions concernant les comportements asymptotiques en lien avec les fonctions, ils traduisent ces valeurs de F1(*a*) et F1(*b*) en termes de « limites ».

En effet, suivant les niveaux scolaires, nous obtenons les réponses suivantes[93] :

(B1_2nde) « L'exécution de l'algorithme semble donner qu'une seule solution pour l'équation F1(*x*)=0, dont la valeur approchée serait 1,4142316. » (Binôme de Seconde)

(B2_2nde) « La valeur de l'unique solution de l'équation est presque égale à 1,4142316. » (Binôme de Seconde)

(B3_2nde) « L'algorithme donne la valeur d'une solution de l'équation F1(x)=0. Cette valeur est presque égale à 1,4142316. Mais, tant que nous n'avons pas tracé la courbe de F1 nous ne pouvons pas savoir si cette solution est unique. » (Binôme de Seconde)

(B1_1ère) « L'algorithme de dichotomie propose une solution à l'équation sur l'intervalle [1 ; 2]. Mais, nous ne pouvons pas conclure directement sur la signification de ce résultat. Nous constatons que l'intervalle final est réduit à un point et que les valeurs de F1(*a*) et

[93] Extraits de copies

F1(*b*) sont très grandes. Nous devons représenter la fonction pour pouvoir vraiment conclure. » (Binôme de Première)

(B2_1ère) « L'algorithme donne un « encadrement » réduit à une valeur 1,4142316 (dû peut-être à un problème d'approximations de la machine) d'une solution de l'équation F1(x)=0. Nous devons tracer la courbe de F1 pour savoir à quoi correspond cette valeur. Les valeurs de F1(*a*) et F1(*b*) ne semblent pas aller vers 0. 1, 4142316 n'est peut-être pas une solution de cette équation. » (Binôme de Première)

(B3_1ère) « Les valeurs qui précédent les deux dernières valeurs égales à 1,4142136, sont de « plus en plus petites » pour les F1(*a*) et de « plus en plus grandes » pour les F1(*b*), donc la valeur 1,4142136 ne peut pas correspondre à une solution de l'équation F1(*x*)=0 sur l'intervalle [1 ; 2]. Peut-être que c'est une valeur interdite. La courbe de F1 doit avoir un trou en 1,4142136. Il faut donc afficher la courbe de F1 pour avoir plus d'indications pour répondre à la question. » (Binôme de Première)

(B4_1ère) « Les valeurs de F1(*a*) et F1(*b*) semblent de plus en grandes au fur et à mesure que l'algorithme tourne. Est-ce que les fonctions peuvent avoir une limite comme les suites ? Nous pensons que F1(*x*)=0 n'a peut-être pas de solution sur l'intervalle [1 ; 2]. Nous ne savons pas à quoi correspond la valeur 1,4142136 qui est affichée deux fois à la fin de l'exécution de l'algorithme. » (Binôme de Première)

(B1_TS) « D'après les valeurs affichées par l'exécution de l'algorithme, nous en déduisons que les deux valeurs finales affichées sont égales et qu'elles correspondent à « $x \rightarrow$ 1,4142136 avec $x < 1{,}412136$ », et « $x \rightarrow$ 1,4142136 avec $x > 1{,}4142136$ ». Pat conséquent, l'algorithme donne la valeur des limites de F1 en $x \rightarrow$ 1,4142136 avec $x < 1{,}4142136$, et $x \rightarrow$ 1,4142136 avec $x > 1{,}4142136$. Donc, l'algorithme de dichotomie peut aider au calcul de limites. » (Binôme de Terminale)

(B2_TS) « Les valeurs affichées pour F1(*a*) vont vers moins l'infini et vers plus l'infini pour F1(*b*). Donc l'algorithme de dichotomie a permis de calculer les limites infinies en une valeur. La courbe de F1 devra donc présenter une asymptote verticale en cette valeur. Il faut donc lever le crayon pour tracer la courbe de F1. Par conséquent, la fonction n'est pas continue sur l'intervalle [1 ; 2]. » (Binôme de Terminale)

(B3_TS) « Les deux valeurs affichées à la fin de l'exécution de l'algorithme sont toutes les deux égales à 1,4142136. Mais, comme les valeurs de F1(a) et de F1(b) vont vers des nombres très grands dans les négatifs et dans les positifs, nous pouvons en déduire que F1(x)=0 n'a pas de solution dans l'intervalle [1 ; 2]. La valeur 1,4142136 est une valeur

interdite pour la fonction F1. Nous avons une asymptote en 1,4142136. » (Binôme de Terminale)

c) Quel retour sur l'interprétation des valeurs affichées après la représentation graphique de la fonction F1 ?

Comme pour la phase 3, et conformément à la consigne, les élèves ajoutent à l'algorithme une boucle « TantQue », les variables « Binf » et « Bsup », ainsi que la variable « pas », le tout permettant d'afficher la représentation graphique de la fonction cachée F1 sur l'intervalle [-3 ; 3].

Suite au travail fait pendant la phase 3, nous observons que pour cette quatrième phase, tous les binômes du cycle Terminal Scientifique prennent un « pas » égal à 0,001 pour la suite des valeurs des abscisses des points de la représentation. Il y a ainsi très peu de variations entre les différents algorithmes rendus par les élèves de ces deux niveaux scolaires. Cependant, nous observons qu'à ces mêmes niveaux, une grande diversité d'interprétations des résultats obtenus lors de l'exécution de l'algorithme dans l'environnement *AlgoBox* une fois que celui-ci est complété, peut être proposée par les binômes, en particulier chez les élèves de Premières

En revanche, nous constatons qu'au niveau Seconde, il y a des variations très importantes dans le choix du « pas » pour la suite des *x*. Ainsi, nous observons que 80% des binômes de Seconde choisissent un pas de 0,001 comme ceux du cycle Terminal Scientifique, mais 5% prennent un pas de 1, 5% un pas de 0,1 et 10% un pas de 0,01. Il s'ensuit alors une grande variété dans les interprétations des résultats obtenus après l'exécution de l'algorithme complété par les élèves de Seconde.

Nous constatons que tous les binômes qui choisissent un pas de 0,001 obtiennent le même graphique après qu'ils aient complété leurs algorithmes (Fig. 154).

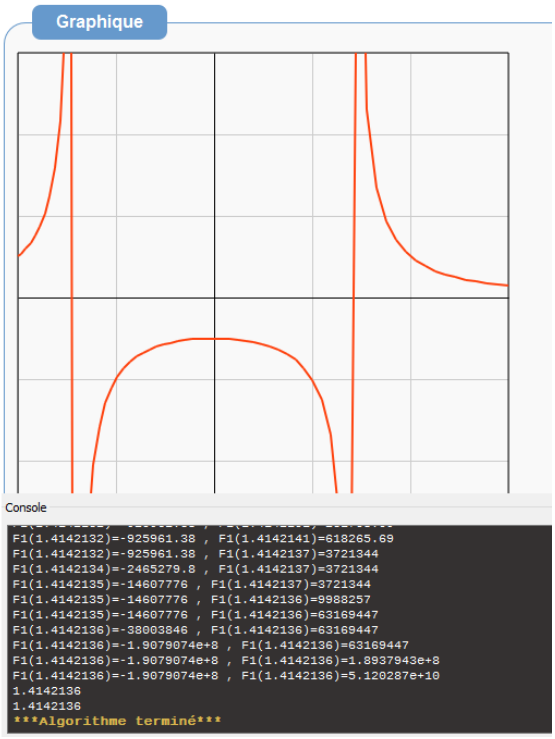


Figure 154 (Représentation graphique de la fonction « cachée » avec un pas de 0,001)

De même, les binômes de Seconde qui choisissent un pas de 1 ou de 0,1 ou de 0,01, obtiennent suivant le pas des graphiques analogues (Extraits de copies fig. 155 à 157).

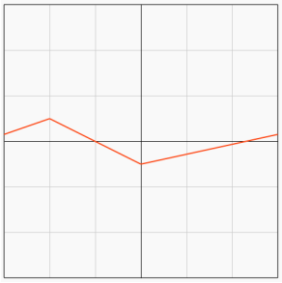


Figure 155 (Cas où le pas est de 1)

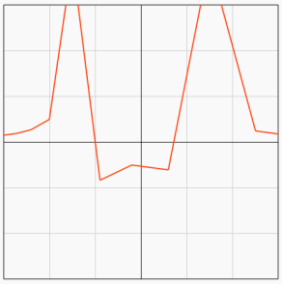


Figure 156 (Cas où le pas est de 0,1)

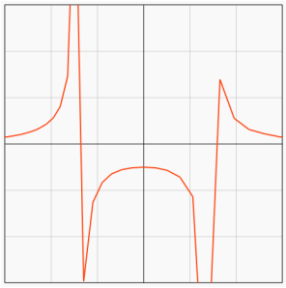


Figure 157 (Cas où le pas est de 0,01)

Comme nous le disions plus haute, les interprétations des résultats (graphiques et numériques) varient suivant les pas choisis et aussi suivant les niveaux scolaires. En effet, au niveau Seconde, nous observons que les élèves restent pour l'essentiel dans le descriptif de ce que donne l'exécution de l'algorithme dans l'environnement *AlgoBox* sans proposer une réelle justification mettant en jeu des connaissances issues du domaine de l'analyse. De plus, nous observons qu'à ce niveau, les élèves ne portent pas nécessairement un regard critique sur la représentation graphique affichée par l'ordinateur. En revanche, au niveau de cycle Terminal Scientifique, nous remarquons que les élèves ne se contentent pas d'affirmer des conjectures, mais essayent aussi d'argumenter leurs propos.

Avec l'étude d'un échantillon de copies représentatif de l'ensemble des binômes observés, nous pouvons lire différents commentaires proposés par les élèves sur l'observation des résultats de l'exécution de l'algorithme complété permettant d'afficher une représentation graphique de la fonction « cachée ».

- **En Seconde**

- **Pour le pas 1 (5% des binômes de Seconde)**

Les binômes faisant ce choix de pas, proposent comme réponse que « l'équation F1(x)=0 admet une seule solution sur l'intervalle [1 ; 2] qui est 2 » (Extrait d'une copie). Ils argumentent en disant que « […] le point commun de la représentation graphique de F1 et de l'axe des abscisses est le point (2 ; 0) » (Extrait de la même copie), sans revenir sur les valeurs finales affichées par l'exécution de l'algorithme.

- **Pour les pas 0,1 (5% des binômes de Seconde)**

Les binômes faisant ce choix de pas, proposent comme réponse que « l'équation F1(x)=0 n'admet pas de solution sur l'intervalle [1 ; 2] » (Extrait d'une copie). Ils argumentent en disant que « […] la courbe coupe l'axe des abscisses sur [0 ; 3] au point ayant une abscisse comprise entre 0,5 et 1, mais pas en un point dont le *x* serait compris entre 1 et 2 » (Extrait d'une copie), et complètent en ajoutant que leur « […] observation graphique ne correspond pas à la valeur finale 1,4142136 affichée par la machine » (extrait d'une copie).

- **Pour le pas 0,01 (10% des binômes de Seconde)**

Pour ceux faisant ce choix de pas, nous observons qu'ils proposent comme réponse que « l'équation F1(x)=0 admet une solution sur l'intervalle [1 ; 2] qui est 1,4142136 » (Extrait d'une copie). Ils argumentent en disant que « […] le point commun de la représentation graphique et de l'axe des *x* est le point ayant pour abscisse 1,4142136 » (Extrait de la même copie) et ajoutent « Nous retrouvons la valeur affichée par l'algorithme pour *a* et *b* » (Extrait de la même copie).

- **Pour le pas 0,001 (80% des binômes de Seconde)**

Nous observons que la majorité des binômes de Seconde choisit un pas suffisamment « fin » pour obtenir une représentation graphique correcte. Cependant, certains de ces binômes après avoir représenté la fonction sur l'intervalle [-3 ; 3] éprouvent des difficultés à interpréter la représentation graphique obtenue. En effet, au moment de l'expérimentation et parmi ces binômes, les élèves cherchent à se ramener à une des fonctions usuelles qu'ils connaissent (fonctions linéaires, affines, carrée, inverse, homographique, racine carrée, etc.). Ainsi, ils constatent tous que la fonction « cachée » n'est ni affine ni carrée. Ils écrivent par exemple :

- (B(a)_2nde) « La fonction F1 n'est pas une fonction carrée car la courbe n'est pas parabole. Elle ne correspond à aucune des courbes connues. » (Binôme de Seconde) ;

D'autres indiquent :

- (B(b)_2nde) « La fonction F1 est faite de cinq morceaux : deux branches d'hyperbole, deux droites verticales, une parabole. » (Binôme de Seconde)
- (B(c)_2nde) « La partie de la fonction F1 sur l'intervalle [1 ; 2] admet une droite qui est perpendiculaire à l'axe des abscisses et qui coupe la courbe de F1 en deux morceaux. » (Binôme de Seconde)
- (B(d)_2nde) « On peut voir que la courbe de F1 montre que pour un *x* compris entre 1 et 2, il y a une droite qui sépare les morceaux de la courbe comme pour une hyperbole. Les points de la courbe font en descendant du côté gauche de la droite frontière et en montant du côté droit de la droite frontière. Cela peut signifier qu'il y aurait une valeur interdite. » (Binôme de Seconde)

Nous observons ainsi que les élèves cherchent à expliquer les différents « morceaux » de la représentation graphique. Cependant, aucun binôme ne fait référence aux valeurs renvoyées par l'environnement sur les images de F1(a) et F1(b). Seuls 10% de ces binômes soulignent de manière graphique le caractère asymptotique de la représentation graphique sur l'intervalle [1 ; 2] (voir par exemple le commentaire de la copie B(d)_2nde).

Nous observons que certains de ces binômes (environ 30%) rendent en plus du graphique et des valeurs (fig. 154) un zoom (Fig. 158) sur l'intervalle [1 ; 2] de la représentation graphique

qu'ils commentent sur le principe des binômes (B(c)_2nde) et (B(d)_2nde) et font apparaitre l'abscisse du point commun à la droite « verticale » et à l'axe des abscisses.

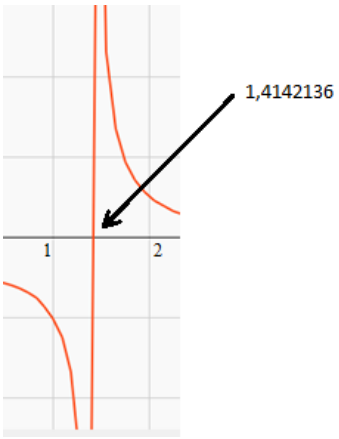


Figure 158 (Zoom de la représentation graphique de la fonction « cachée » sur [1 ;2])

Nous observons que 95% de ces binômes de Seconde ne présente plus le résultat « 1,4142136 » comme pouvant être une solution à l'équation F1(x)=0 sur l'intervalle [1 ; 2], mais comme étant « […] plutôt une valeur interdite pour la fonction F1 » (dixit un élève), après avoir procédé à la représentation graphique de la fonction F1 comme il leur est demandé. Par exemple, le binôme (B(a)_2nde) écrit : « La courbe de F1 montre que la fonction F1 ne s'annule pas sur [0 ; 3] et elle admet une valeur interdite. Cette valeur interdite correspond à la valeur finale « 1,4142136 » affichée par *AlgoBox*. »

En revanche, nous observons que les 5% des binômes de Seconde ayant considéré que la fonction F1 est représentée par cinq morceaux (cf. le binôme (B(b)_2nde), écrivent : « L'équation F1(*x*)=0 admet une unique solution presque égale 1,4142136 sur l'intervalle [1 ; 2]. C'est l'abscisse du point où un morceau de la courbe de F1 coupe l'axe des abscisses. » (Extrait de la copie du binôme (B(b)_2nde)).

- **En Première Scientifique**

Nous constatons qu'à ce niveau scolaire, les élèves choisissent tous comme pas 0,001. De plus, comme ils ont de plus grandes compétences dans le domaine des fonctions, ils ne se contentent pas de décrire la représentation graphique, mais ils essayent de porter une analyse explicative des résultats observés après l'exécution de l'algorithme dans l'environnement *AlgoBox*. Ainsi, nous obtenons comme retour de la part des élèves de ce niveau scolaire, les commentaires suivants :

- (B1_1ère) « D'après la représentation graphique et les valeurs de F1(*a*) et F1(*b*) sont très grandes, nous pouvons dire que l'algorithme de dichotomie ne donne pas la valeur approchée d'une solution à l'équation F1(x)=0 sur [1 ; 2], mais calcule une valeur interdite. Pour 1,4142316, on a donc que la fonction F1 n'est pas définie. »

- (B2_1ère) « Comme nous l'avions supposé dans la question 2), la valeur 1,4142316 n'est finalement pas une valeur approchée d'une solution à l'équation F1(*x*)=0. Sur cet intervalle, la courbe montre que la fonction n'est pas définie en cette valeur. Nous en déduisons que l'équation F1(*x*)=0 n'a pas de solution sur [1 ; 2]. »
- (B3_1ère) « Nous savons que les valeurs qui précédent les deux dernières valeurs égales à 1,4142136, sont de « plus en plus petites » pour les F1(*a*) et de « plus en plus grandes » pour les F1(*b*), donc la droite qui passe par le point (1,4142136 ; 0) montre que la fonction F1 n'est pas définie sur tout l'intervalle [1 ; 2] et que par conséquent l'équation F1(*x*)=0 n'a pas de solution sur l'intervalle [1 ; 2]. L'algorithme a donné le résultat de la valeur interdite car nous avions des changements de signe pour F1 avant et après cette valeur. »
- (B4_1ère) « La courbe de F1 montre bien que les valeurs de F1(*a*) et F1(*b*) sont de plus en grandes au fur et à mesure que l'algorithme tourne. Nous pouvons en déduire que lorsque la valeur de *x* se rapproche de la valeur affichée 1,4142136, la courbe de F1 admet une droite frontière. Nous voyons aussi que la fonction F1 est strictement décroissante sur les intervalles [0 ; 1, 4142136[et] 4142136 ; 3]. »

Nous observons donc que les binômes de Première ne considèrent pas la valeur « 1,4142136 » comme une valeur approchée solution d'une équation mais comme une « valeur interdite ».

- **Au niveau Terminale Scientifique**

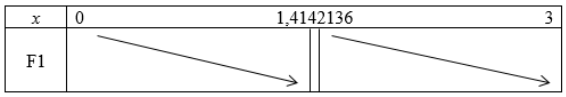
Comme pour le niveau Première, ici aussi le choix du pas se porte sur 0,001 par l'ensemble des binômes. De plus, nous observons de la part des binômes de ce niveau scolaire des explications en lien avec les conditions du TVI et de son corollaire, le TDB[94]. En effet, nous constatons que plus de 90% des binômes proposent un début de justification mettant en place des raisonnements basés sur les concepts de limites, d'« asymptotes verticales », de

[94] Théorème de la bijection

continuité et de monotonie de la fonction F1 sur l'intervalle de départ [1 ; 2] ou sur l'intervalle de la représentation graphique, c'est-à-dire [-3 ; 3].

Ainsi, au niveau Terminale, nous obtenons les commentaires suivants de la part de binômes représentatifs de la classe :

- (B1_TS) « La courbe représentative de la fonction F1, nous montre que la fonction admet des valeurs interdites sur l'intervalle [-3 ; 3] qui sont opposées, car la courbe semble admettre l'axe des ordonnées comme axe de symétrie et que nous avons deux asymptotes en ces valeurs. De plus sur l'intervalle [0 ; 3], la fonction F1 est strictement décroissante sur [0 ; 1,4142136[, ainsi que sur $]1{,}4142136\,;3]$. Nous avons donc le tableau de variation suivant sur [0 ; 3] :



  En conclusion, l'algorithme donne la valeur approchée de la valeur interdite de la fonction F1 sur [0 ; 3]. Comme cette valeur est interdite, alors il faut lever le crayon pour tracer la courbe et par conséquent la fonction F1 n'est pas continue sur [0 ; 3]. Les conditions du TVI ne sont pas vérifiées. »
- (B2_TS) « La représentation graphique montre bien que la fonction n'est pas continue sur [-3 ; 3] ni sur [1 ; 2]. Nous en déduisons que l'équation F1(x)=0 n'a pas de solution sur [1 ; 2]. D'après les valeurs de F1(*a*) et F1(*b*), nous avons $\lim_{x\to 1{,}4142136^-} \mathrm{F1}(x) = -\infty$ et $\lim_{x\to 1{,}4142136^+} \mathrm{F1}(x) = +\infty$. La valeur obtenue par l'algorithme de dichotomie correspond à une valeur interdite. »
- (B3_TS) « La représentation graphique de F1 confirme les observations précédentes. La courbe admet une asymptote « verticale » en 1,4142136. Par conséquent, la fonction n'est pas continue sur [1 ; 2]. Nous ne pouvons pas utiliser le Théorème des valeurs intermédiaires, même si nous avons le fait que la fonction est strictement décroissante sur les intervalles entre 0 et 3 et quelle change de signe sur l'intervalle [0 ; 3]. Comme nous l'avons vu dans l'exercice précédent[95], nous en concluons que l'algorithme de dichotomie ne peut être utilisé que si on connait les propriétés de la fonction sur l'intervalle du départ. »

[95] Voir phase 3 (N.D.L.R.)

2.5.5.2. Analyse *a posteriori*

A ce stade de l'expérimentation, nous observons donc que quasiment tous les binômes (près de 90% au niveau Seconde et 100% au niveau du cycle Terminal Scientifique) ont conscience que l'algorithme peut ne pas afficher un résultat qui correspondrait nécessairement à une solution d'une équation mais bien à une valeur interdite pour la fonction « cachée ». Cependant, une très grande variété d'explications concernant le fait que le résultat pourrait être une valeur interdite, est proposée par les binômes suivant leurs compétences acquises dans le domaine de l'analyse et non en algorithmique.

De plus, comme nous l'avions constaté déjà lors de l'analyse *a posteriori* de la phase précédente, les élèves en cette fin de première partie de la sous-ingénierie « dichotomie continue » qui a succédé à la sous-ingénierie « dichotomie discrète » sont totalement autonome pour l'utilisation de l'environnement informatique, tant sur le plan exécution que sur le fait de compléter l'algorithme proposé par l'enseignant, et ceci quel que soit le niveau scolaire. Nous constatons qu'une très forte majorité d'élèves semble sortir de l'aspect purement « application numérique » de l'*algorithme de dichotomie*. En effet, de nombreux binômes indiquent que l'utilisation de cet algorithme doit nécessiter un travail en amont sur l'étude de certaines propriétés de la fonction, en particulier le fait de vérifier si la fonction serait bien définie sur l'intervalle du départ pour l'utilisation de l'algorithme.

Au cours de cette phase 4 et à la vue du déroulement, nous pouvons aussi penser que les consignes et les tâches demandées sont bien comprises par l'ensemble des binômes. Cependant, nous observons que certains binômes de Seconde restent essentiellement dans un $ETA_{Algobox}$ pour expliquer leurs raisonnements. Nous proposons alors d'affiner notre analyse *a posteriori* de cette phase 4 en nous référant aux Espaces de Travail Mathématique er Algorithmique. En effet, comme pour la phase précédente, les élèves doivent de fait coordonner deux genèses discursives dans les deux Espaces de Travail parallèles que sont un $ETA_{AlgoBox}$ et un $ETM_{analyse}$, ainsi que deux genèses l'une instrumentale et l'autre sémiotique dans ces mêmes Espaces de Travail parallèles. En effet, ils sont amenés à réfléchir, à travers l'investigation de la recherche, sur la signification des résultats affichés par l'environnement *AlgoBox* lors de l'exécution de l'algorithme proposé dans un premier temps, puis une fois complété sur la pertinence de leurs premières explications.

En nous référant au travail fait lors des analyses *a posteriori* des phases 2 et 3 sur les

articulations possibles entre Espaces de Travail, nous proposons alors une présentation sous forme d'un tableau (Tableau n° 37) qui va nous permettre de présenter de façon plus synthétique, les genèses de convergences entre les deux Espaces de Travail parallèles mis en jeu au cours de cette quatrième phase.

	$ETA_{AlgoBox}$	**$ETM_{Analyse}$**	**Commentaires afin de montrer que les deux Espaces de Travail sont porteurs d'idées intéressantes**
Comparer les algorithmes de la phase 1 et des deux dernières phases de cette première partie de la sous-ingénierie « dichotomie continue ».	Analyser le texte de l'algorithme d'un point de vue algorithmique. Observer le fait que la condition dans l'alternative est différente et expliquer pourquoi cette condition ne modifie pas les branches « SI et SINON ». Interpréter la condition dans l'alternative en termes de monotonie pour la fonction F1 Observer et expliquer le fait que l'inégalité sur la condition dans l'alternative est stricte et simplifiée par rapport aux phases 1 et 3. (Genèse discursive)	Analyser le texte en rapport avec la méthode de dichotomie : • La fonction F1 est monotone • Changement de signe dans l'intervalle. • Evolution des valeurs des bornes. • Constater que ces valeurs des bornes tendent vers l'infini. (Genèse discursive)	• Les élèves de Seconde et de Première comparent les algorithmes des phases 3 et 4 et repèrent que la condition dans le « SI » est différente mais ne font pas de remarque sur le fait qu'il n'y ait pas d'interversion dans les branches d'affectation de *a* et de *b* contrairement à ce que nous avions rapporté lors de la phase 3. Nous interprétons cela comme une réduction à l'ETA (pas de travail particulier en analyse). • Les élèves de Terminale proposent des explications en lien avec le sens de variation de la fonction F1 sur l'intervalle du départ. Ils associent cette condition « allégée » au fait que la fonction serait croissante sur l'intervalle du départ. Nous interprétons cela comme un travail en analyse, ainsi ces élèves articulent un ETA et un $ETM_{analyse}$. • L'ensemble des binômes

			analyse l'algorithme de la phase 4 pour conclure à sa validité et sa correspondance à la méthode de dichotomie. Leur analyse met en jeu leur compréhension du principe de découpage de l'intervalle dans la méthode et de la signification de l'inégalité (comme changement ou non de signe sur l'intervalle). Cependant, les explications sur ce point sont moins détaillées qu'au cours de la phase 3. La validité de l'algorithme, question posée dans l'ETA$_{AlgoBox}$ résulte d'arguments relatifs à la méthode, qui relèvent donc de l'ETM$_{analyse}$.
Représentation graphique et interprétation	Construire des représentations graphiques : ensemble de points. Choisir un pas pour la suite des abscisses des points de la représentation graphique. (Genèse instrumentale)	• Repérer la monotonie de la fonction « cachée » et le fait que la fonction n'est pas définie en la valeur finale affichée par *AlgoBox* – Représentation graphique de la fonction sur l'intervalle donné et l'axe des abscisses. • Savoir interpréter les droites perpendiculaires à l'axe des abscisses. • Envisager de zoomer le graphique sur l'intervalle [1 ; 2]. • Reconnaitre à l'aide de la représentation graphique de la fonction les coordonnées d'un point de la représentation.	• Les gestes de construction et de visualisation à mettre en place sont familiers chez les élèves qui ont maintenant l'habitude de travailler avec des environnements informatiques comme *AlgoBox*. • Par rapport à une calculatrice graphique, la représentation doit être programmée par une itération. C'est aussi un geste familier. • Une utilisation des possibilités de construction de représentations graphiques qu'offre l'environnement *AlgoBox*. • La variable « pas » est assez pertinente lors de

		(Genèses sémiotique/visualisation et instrumentale) • Construire la suite des *x* en choisissant un pas adéquat afin d'affiner l'aspect « visuel » de la représentation graphique) (Genèse instrumentale) (Genèse visualisation) • Reconnaitre la nature arithmétique de cette suite des *x* en particulier au niveau du cycle Scientifique Terminal. (Genèse discursive)	cette quatrième phase. En effet, suivant le « pas » choisi par les binômes la représentation graphique donnée par l'environnement informatique peut aboutir à une conclusion erronée de la part du binôme ou à une remise en question de la signification des valeurs affichées. Contrairement à la phase précédente, le choix du « pas » dans cette phase peut permettre au binôme de reconsidérer le choix d'une fonction ayant ou pas une solution ou admettant une valeur interdite afin de mieux appréhender le choix d'une variable « pas » pour qu'ils puissent concevoir la nécessité d'affiner l'aspect « visualité » de la représentation graphique. Comme pour la phase 4, ceci amène l'enseignant à laisser plus de temps sur cette problématique de définir un « pas » adéquat, et par conséquent de permettre une discussion.
Exploitation des valeurs rendues par l'algorithme	Analyser la condition de continuation : Avec une condition d'inégalité de la forme $A > 10^{-8}$: • quelle est l'assertion vérifiée en sortie ? • pourquoi cette condition ne porte que sur l'image de	Interpréter : • les grandeurs des valeurs de F1(*a*) et F1(*b*) ; • l'égalité entre les valeurs finales de *a* et *b*. (Genèse discursive) (Genèse visualisation)	• La valeur obtenue est identique pour *a* et *b* : 1,4142136 • L'ensemble des binômes de Première et une majorité des élèves de Seconde considèrent ce nombre comme correspondant à une valeur interdite pour la

	(a+b)/2 et non sur le produit de deux images comme pour les phases 1 et 3 ? (Genèse instrumentale) (Genèse discursive)		fonction en se référant entre autres aux grandes valeurs affichées pour F1(a) et F1(b). Cependant, les élèves restent dans l'ETA. • Cependant, au niveau de la Terminale, les binômes considèrent aussi que 1,4142136 correspondrait à une valeur où la fonction ne serait pas définie et que par conséquent la fonction F1 ne serait pas continue sur l'intervalle [1 ; 2]. Les élèves sont dans l'ETM. • Très peu d'élèves (seulement en Seconde) considèrent cette valeur numérique comme pouvant être une solution approchée de l'équation F1(x)=0. • D'autres (toujours au niveau de la Seconde) considèrent qu'il y aurait un paradoxe entre les valeurs approchées et les lectures faites sur la représentation graphique de la fonction. Dans ces derniers cas, les élèves concernés restent dans l'ETA. • Nous mettons en relation le comportement des élèves de Terminale avec une attitude de doute vis-à-vis de l'ETA que les élèves « contrôlent » dans

			l'ETM.
Interpréter l'obtention d'une valeur interdite	Du point de vue des propriétés de l'algorithme : • La reconnaissance d'une asymptote au point correspondant à la dernière valeur affichée. • Interpréter les valeurs de F1(*a*) et F1(*b*) sous forme de limites à droite et à gauche en 1,4142136. • Associer la dernière valeur da *a* comme *x* → 1,4142136, avec *x*<1,4142136 et la dernière valeur de *b* comme *x* → 1,4142136 avec *x*>1,4142136 (Genèse discursive)	Du point de vue de la méthode : • Nécessité de conditions pour l'existence. (Genèse discursive)	• La représentation graphique de la fonction « cachée » par *AlgoBox* interpelle les élèves et leur permet de remettre en cause le résultat affiché, même si cette remise en cause peut être erronée. • Comme pour la phase précédente, dans le cadre d'un ETA, les élèves questionnent l'algorithme. Et, dans le cadre d'un ETM, les élèves questionnent la méthode. • Les élèves de Terminale s'appuient sur les conditions suffisantes d'existence, comme prérequis à l'utilisation de l'algorithme (perspective d'une « application numérique » qui privilégie l'ETM). Les autres sont davantage dans l'interrogation et considèrent des propriétés générales de l'algorithme (privilégiant l'ETA).

Tableau 37

Pour conclure sur cette quatrième phase, nous pouvons observer que le travail fait depuis le début de l'ingénierie sur la dichotomie est porteur de prises de conscience chez une majorité d'élèves sur la nécessité d'articuler différents Espaces de Travail $ETA_{AlgoBox}$ et $ETM_{analyse}$ en parallèle afin de répondre à un problème donné lors d'une situation de type « analyse numérique » autour de l'*algorithme de dichotomie*. En effet, à la fin de cette première partie de la sous-ingénierie « dichotomie continue », nous observons que le travail suivi par les élèves semble être vecteur d'une meilleure compréhension de la mise en place d'un algorithme d'approximation comme outil d'une « application numérique » mais aussi comme objet d'« analyse numérique ».

2.6 Synthèse de la première partie de la sous-ingénierie « dichotomie continue »

Comme nous l'avions considéré, les élèves ont pu réinvestir au cours de cette première partie de la sous-ingénierie « dichotomie continue » les structures des algorithmes obtenus et étudiés au cours de la sous-ingénierie « dichotomie discrète », en particulier celle du joueur B (celui qui recherche le nombre secret). Cependant, les élèves ont dû procéder à des adaptations nécessaires sur les conditions d'arrêt ou de continuation, ainsi que sur les conditions des alternatives.

Le travail fait au cours de ces quatre premières phases nous montre une évolution dans l'installation d'un *ETA* qui complète bien le premier bilan observé lors de la sous-ingénierie « dichotomie discrète » sur une première mise en place de cet *ETA*. De plus, nous avions constaté dans la première sous-ingénierie, que la « dichotomie discrète » pouvait être un objet algorithmique mais pas un objet d'un *ETM* spécifique. En revanche, au cours de cette deuxième sous-ingénierie, nous observons des articulations possibles entre *ETA* et *ETM* en particulier au niveau du cycle Terminal Scientifique.

Comme pour la sous-ingénierie « dichotomie discrète », nous constatons que le travail fait par les élèves au niveau de la Seconde reste assez laborieux en particulier lors des phases 2 (*algorithme de balayage*) et 3 (*algorithme de dichotomie* – cas où la fonction n'admet pas qu'un seul zéro sur l'intervalle du départ).

Cependant, au fur et à mesure de l'avancé du travail des élèves, quel que soit le niveau scolaire, nous observons que le choix de l'organisation de ces quatre phases est très productif. En effet, outre le travail fait sur les structures algorithmiques de contrôle et de données, cette organisation permet aux élèves de changer leur posture vis-à-vis de la place de la dichotomie en analyse et en particulier d'aller d'une utilisation de l'*algorithme de dichotomie* (ou *de balayage* (voir la phase 2)) comme « application numérique » à une utilisation où la dichotomie serait un objet[96] de l'« analyse numérique ».

Ainsi, les élèves peuvent mettre en évidence des conditions d'existence (phase 4) et d'unicité (phase 3) sur la recherche d'une valeur approchée (ou d'un encadrement) d'une solution d'une équation de la forme $f(x) = 0$, où f doit vérifier certaines propriétés permettant

[96] Au sens de Douady (1984)

de faire que la dichotomie devienne un objet d'un $ETM_{analyse}$. Cependant, suivant les niveaux scolaires, nous remarquons chez l'élève des interactions plus ou moins fortes entre *ETA* personnels et *ETM* personnels.

3. Deuxième partie de la sous-ingénierie « dichotomie continue »

Afin de donner du sens à l'approche « analyse numérique » de la « dichotomie continue », nous proposons, en accord avec les élèves[97], d'aller vers une preuve du TVI qui serait bâtie sur le principe de la dichotomie et l'utilisation de deux suites adjacentes.

Pour cela, nous présentons auprès des élèves une brève introduction de la notion de suites adjacentes et des propriétés qui leur sont liées. En effet, ces notions et propriétés ne sont plus au programme de mathématiques des classes de lycée depuis l'introduction officielle (2010 à 2012, suivant le niveau scolaire) d'un enseignement de l'algorithmique.

Par conséquent, cette seconde partie ne peut se faire qu'au niveau de la classe de Terminale Scientifique. En effet, seuls les élèves de ce niveau scolaire peuvent « approcher » une telle preuve qui ne fait plus partie des attentes institutionnelles, sachant que les compétences mathématiques sur les suites adjacentes ne sont plus du ressort des élèves du lycée. La conséquence de cet aspect institutionnel fait que les enseignants des classes de Seconde et de Première Scientifique participant à l'ingénierie ne souhaitent pas participer à cette dernière partie.

Afin de ne pas alourdir cette présentation, nous proposons de découper en deux phases cette deuxième partie terminant l'expérimentation sur la « dichotomie continue ». Ainsi, nous avons une première phase (notée phase 5, dans la continuité des phases précédentes), où les élèves prennent connaissance de la définition et des propriétés des suites adjacentes avec une application de ces nouveaux savoirs sur deux exercices (N° 1 et 2 de la figure 159) suivie d'une deuxième phase, notée phase 6, où les élèves proposent une preuve du TVI basée sur le principe de la dichotomie (Exercice n° 3 de la figure 159).

Exercice 1 (Extrait du sujet de Métropole, série S, Juin 2005)
On suppose connus les résultats suivants :
(1) deux suites (u_n) et (v_n) sont adjacentes lorsque : l'une est croissante, l'autre est décroissante et $u_n - v_n$ tend vers 0 quand n tend vers $+\infty$;

[97] Pour cette dernière partie de l'expérimentation sur la dichotomie, l'enseignant et le chercheur sont une seule et unique personne.

(2) si (u_n) et (v_n) sont deux suites adjacentes telles que (u_n) est croissante et (v_n) est décroissante, alors pour tout n appartenant à **N**, on a $u_n \leq v_n$; (3) toute suite croissante et majorée est convergente ; toute suite décroissante et minorée est convergente. Démontrer alors la proposition suivante : « *Deux suites adjacentes sont convergentes et elles ont la même limite* ».
Exercice 2 : Soient deux suites (x_n) et (y_n) définies pour $n > 0$ par les relations : $x_n = 1/n + 1/(n+1) + \cdots + 1/(2n)$ et $y_n = 1/(n+1) + 1/(n+2) + \cdots + 1/(2n)$. Démontrer que les suites (x_n) et (y_n) convergent vers une même limite. (On ne demande pas de déterminer la valeur de cette limite.)
Exercice 3 : Construire une preuve du Théorème des valeurs intermédiaires à l'aide de la méthode de dichotomie.

Figure 159 (Série d'exercices sur les suites adjacentes)

3.1 Les objectifs et les attentes

3.1.1 Les objectifs

Comme nous l'indiquions en introduction de cette deuxième partie, celle-ci ne s'adresse qu'aux élèves de la classe de Terminale scientifique. Les élèves de ce niveau ayant eu l'occasion d'étudier les différentes conditions permettant de prendre conscience des conditions nécessaires du « Théorème des Valeurs Intermédiaires » au cours des quatre phases précédentes, peuvent désirer proposer une preuve de ce théorème qui serait construite à partir de la construction de deux suites adjacentes dont les termes seraient obtenus à l'aide de la méthode de dichotomie.

Ceci va ainsi nous aider à compléter l'étude des interactions entre différents *ETM* et *ETA*, suite à l'importance des contenus mathématiques issus du domaine de l'analyse sur des compétences (les suites adjacentes) que les élèves ne sont plus censés avoir selon les attentes de l'institution.

3.1.2 Les attentes

Le travail sur l'*algorithme de dichotomie* fait au cours des phases précédentes a permis aux élèves d'obtenir une suite d'intervalles emboités et à bornes rationnelles, avec des conditions de signe aux bornes des intervalles, jusqu'à une amplitude arbitrairement petite. De plus, l'étude du corps de boucle leur a aussi permis de construire la suite des bornes et une condition d'arrêt faisant intervenir l'amplitude du dernier intervalle. Ils ont pu observer aussi des situations où, bien que l'algorithme termine, il n'encadre pas un zéro, ou il n'en encadre qu'un seul alors qu'il en existe plusieurs. L'aspect informatique d'une utilisation de l'*algorithme de dichotomie* comme objet d'« analyse numérique » ayant été compris et

analysé par les élèves aux cours des phases 3 et 4, nous pensons que ces élèves vont vouloir aller plus loin dans le raisonnement mathématique et ainsi construire une preuve du TVI. Cette preuve doit les amener à étudier des questions de convergence de deux suites dont les termes sont respectivement les bornes des intervalles emboîtés obtenus lors du déroulement de l'*algorithme de dichotomie*.

3.2 Les choix pour chacune des deux phases

3.2.1 Phase 5 : Une introduction à la notion des suites adjacentes

Au début de cette phase, nous faisons le choix d'indiquer aux élèves qu'ils vont travailler avec une nouvelle famille de suites : les *suites adjacentes*. Ainsi, nous leur fournissons la définition suivante :

> Déf. : « *Deux suites réelles (u_n) et (v_n) sont dites adjacentes si l'une est croissante, l'autre décroissante et si leur différence converge vers 0.* »

Puis, nous leur précisons l'existence de deux propriétés vérifiées par cette famille de suites sans pour autant leur en donner des démonstrations :

> Prop. 1 : *« Soient deux suites adjacentes (u_n) et (v_n), telles que la suite (u_n) soit croissante et la suite (v_n) soit décroissante. Alors, pour tout entier naturel n, $u_n \leq v_n$.* »
> Prop. 2 : « *Deux suites adjacentes sont convergentes et convergent vers une même limite.* »

Afin d'anticiper des difficultés possibles de la part des élèves sur la convergence d'une suite, en particulier en début de Terminale, où les élèves approfondissent les compétences acquises en Première sur la convergence d'une suite, nous leur rappelons le *Théorème de convergence monotone* (TCM) :

> TCM : *« Toute suite croissante (resp. décroissante) et majorée (resp. minorée) converge. »*

Nous leur précisons aussi que ce théorème est admis au niveau de la Terminale. Nous faisons aussi le choix de leur rappeler que de savoir qu'une suite converge ne donne en rien la valeur de sa limite mais qu'elle permet dans certains cas d'appliquer des théorèmes qui permettent de la calculer.

Ces choix sont aussi motivés par le fait que depuis la rentrée 2012, le concept de *suite adjacente* n'est plus au programme et que par conséquent ils nous semblent opportun de donner un complément de connaissance sur ces suites et les propriétés qui leur sont associées. Ainsi, nous faisons le choix de « sortir » du cadre institutionnel pour cette phase et

la suivante.

Une fois ces compléments théoriques faits, nous donnons aux élèves deux énoncés qui doivent leur permettre de proposer une démonstration de la propriété de convergence de deux suites adjacentes. Pour cela nous faisons le choix de présenter l'exercice sous la forme d'un R.O.C. (Restitution et organisation des connaissances) :

> « On suppose connus les résultats suivants :
> (1) deux suites (u_n) et (v_n) sont adjacentes lorsque : l'une est croissante, l'autre est décroissante et $u_n - v_n$ tend vers 0 quand n tend vers $+\infty$;
> (2) si (u_n) et (v_n) sont deux suites adjacentes telles que (u_n) est croissante et (v_n) est décroissante, alors pour tout n appartenant à **N**, on a $u_n \leq v_n$;
> (3) toute suite croissante et majorée est convergente ; toute suite décroissante et minorée est convergente.
> Démontrer alors la proposition suivante : « *Deux suites adjacentes sont convergentes et elles ont la même limite* ».

Ensuite, nous faisons le choix de proposer aux élèves un exercice d'application sur deux suites dont on connaitrait les expressions respectives :

> « Soient deux suites (x_n) et (y_n) définies pour $n > 0$ par les relations :
> $x_n = 1/n + 1/(n+1) + \cdots + 1/(2n)$ et $y_n = 1/(n+1) + 1/(n+2) + \cdots + 1/(2n)$.
> Démontrer que les suites (x_n) et (y_n) convergent vers une même limite.
> (On ne demande pas de déterminer la valeur de cette limite.) »

De plus, cette phase s'inscrit dans une perspective de la phase suivante. En effet, au cours de la dernière phase, les élèves vont devoir élaborer une preuve du *Théorème des valeurs intermédiaires* qui serait construite à l'aide de deux suites obtenues par la méthode de dichotomie. Ainsi, les élèves vont pouvoir se questionner sur l'interprétation des termes des bornes respectives des intervalles emboîtés obtenus dans le cadre de l'*algorithme de dichotomie*.

3.2.2 Phase 6 : Construction d'une preuve dichotomique du TVI

Pour cette dernière phase de l'ingénierie sur la dichotomie, nous faisons le choix de donner un problème relativement « ouvert » qui se limite à une unique question :

> « *Construire une preuve du Théorème des valeurs intermédiaires à l'aide de la méthode de dichotomie.* »

Nous partons de l'hypothèse que des élèves de ce niveau scolaire peuvent prendre l'initiative de chercher à prouver ce théorème, compte tenu des différentes tâches proposées en amont de cette dernière problématique (cf. la première partie de la sous-ingénierie

« dichotomie continue »). Afin de guider dans un premier temps les élèves, en particulier pour des raisons de temporalité, nous faisons le choix de leur préciser que la preuve doit être bâtie sur la méthode de dichotomie.

3.3 Les tâches et le mode de travail

3.3.1 Les tâches

3.3.3.1. Pour la phase 5 (Une introduction à la notion des suites adjacentes)

Les élèves doivent résoudre deux exercices qui sont en lien avec les suites adjacentes. Le premier de ces exercices va leur demander une exploitation d'hypothèses et de savoirs donnés afin de démontrer que *deux suites adjacentes sont convergentes vers une même limite*. Puisque ces suites sont adjacentes, nous nous attendons à ce que les élèves pensent à utiliser le *Théorème de la convergence monotone* qui leur est rappelé en début de phase. En effet, les élèves vont avoir pour première tâche de rappeler que la suite (u_n) est majorée par v_0 et que de même la suite (v_n) est minorée par u_0. Pour cela, nous attendons à ce que les élèves justifient que pour des raisons de monotonies (croissante pour (u_n) et décroissante pour (v_n)), on ait : pour tout entier naturel n, $u_0 \leq u_n \leq v_n \leq v_0$. Puis, une fois cette justification faite, nous attendons que les élèves répondent à la deuxième tâche sur les convergences des suites (u_n) et (v_n) vers les réels respectifs L et L'. Enfin, la troisième tâche va consister à créer une suite (w_n) définie par $w_n = v_n - u_n$ et de justifier que cette suite (w_n) converge bien vers le réel L' – L, puis de conclure sur l'égalité de L' et L en considérant le rappel de la définition de deux suites adjacentes et une propriété sur l'unicité de la limite qui n'est pas précisée dans l'énoncé de l'exercice.

Pour le deuxième exercice, nous attendons que les élèves, ayant la culture des exercices de type R.O.C., pensent à mettre en lien le travail fait au cours de l'exercice précédent et de celui-ci qui consiste à mettre en application la propriété de convergence des suites adjacentes. Nous attendons ainsi que les élèves justifient que les suites (x_n) et (y_n) sont deux suites adjacentes en prouvant qu'elles vérifient bien les conditions : l'une est croissante (la suite (x_n)), l'autre est décroissante (la suite (y_n)) et leur différence tend vers 0 quand n tend vers $+\infty$. Puis, nous attendons qu'ils concluent sur la convergence de ces deux suites en se référant à la propriété démontrée à l'exercice précédent.

3.3.3.2. Phase 6 (Construction d'une preuve dichotomique du TVI)

Nous attendons que les élèves posent comme propriété de départ (alternative du TVI) que : « *Si f est une fonction définie et continue sur un intervalle [a, b], telle que f(a)×f(b)≤0, alors f admet au moins une racine dans cet intervalle [a, b].* ». Puis une fois cette propriété énoncée, nous attendons qu'ils utilisent le principe de dichotomie, sachant que l'ingénierie a pour thématique principale la dichotomie. Pour cela, nous attendons qu'ils construisent deux suites (u_n) et (v_n) définies par : $u_0 = a$ et $v_0 = b$, et pour tout entier naturel n,

$$\begin{cases} u_{n+1} = u_n \text{ et } v_{n+1} = \frac{u_n+v_n}{2}, \text{ si } f(u_n) \times f\left(\frac{u_n+v_n}{2}\right) \leq 0 \\ u_{n+1} = \frac{u_n+v_n}{2} \text{ et } v_{n+1} = v_n, \text{ si } f(u_n) \times f\left(\frac{u_n+v_n}{2}\right) \geq 0 \end{cases}.$$

Puis, nous attendons qu'ils justifient que ces deux suites sont adjacentes, en démontrant que l'une est croissante et que l'autre est décroissante, et que pour tout entier naturel n,

$$|u_n - v_n| \leq \frac{b-a}{2^n} \to 0, \text{ quand } n \text{ tend vers } + \infty.$$

Nous attendons ensuite que les élèves pensent à utiliser la continuité de la fonction f pour justifier que les limites de $f(u_n)$ et $f(v_n)$ sont les mêmes et égalent à f(L), où L est la limite des suites (u_n) et (v_n) située dans l'intervalle [a ; b].

Puis, afin de terminer une preuve de la propriété, nous nous attendons à ce que les élèves pensent à passer à la limite dans la relation $f(u_n) \times f(v_n) \leq 0$, qui leur permet ainsi de justifier que f(L) = 0, et de conclure sur la validité de la propriété.

Pour cette dernière phase, nous n'attendons pas à ce que les élèves aient nécessairement recours à l'outil informatique. En effet, le travail demandé ici se situe pour l'essentiel dans un *Espace de Travail Mathématique* spécifique à l'analyse.

3.3.2 Le mode de travail

Comme pour la première partie de la sous-ingénierie « dichotomie continue », les élèves restent en salle informatique et travaillent en binôme, de façon qu'il y ait un ordinateur par binôme équipé des environnements informatiques utilisés depuis le début de l'ingénierie : *AlgoBox* et *LARP*. Les élèves peuvent aussi utiliser les menus « Graphique » et « Tableur » de leurs calculatrices.

Il est demandé aux binômes de rendre leurs travaux au format numérique (traitement de texte) et s'ils ont construit un (ou des) algorithme(s) permettant de les aider à répondre aux

exercices, en particulier dans le cas de l'exercice d'application des suites adjacentes, ils doivent le (ou les) fournir avec leur copie.

Nous prévoyons un temps global de 90 minutes pour le travail à faire au cours de cette deuxième partie.

3.4 Les analyses *a priori*

3.4.1 Phase 5 : Une introduction à la notion des suites adjacentes

3.4.1.1. Résolution de l'exercice 1 : Démonstration de la propriété « Deux suites adjacentes sont convergentes et ont la même limite L. »

Nous rappelons que les élèves ont à leur disposition l'énoncé du *Théorème de la convergence monotone* et la définition de deux *suites adjacentes*. De plus, dans l'énoncé est précisé qu'il est admis la propriété suivante : « *Si* (u_n) *et* (v_n) *sont deux suites adjacentes telles que* (u_n) *est croissante et* (v_n) *est décroissante, alors pour tout n appartenant à* $\mathbf{N}$*, on a* $u_n \leq v_n$. ».

Une première difficulté qui peut transparaitre chez certains élèves fragiles en mathématiques va être due à l'absence d'expressions de deux suites. Nous nous attendons alors à ce que ces élèves ne cherchent pas à répondre à l'unique question posée qui peut par conséquent leur sembler trop abstraite et trop générale.

Nous pensons que les élèves qui vont surmonter ce souci de généralisation, vont chercher à démontrer que les suites adjacentes (u_n) et (v_n) sont bornées, puis déterminer un majorant de la suite croissante (u_n) et un minorant de la suite décroissante (v_n) afin de pouvoir utiliser le *théorème de la convergence monotone*.

Ainsi, une première erreur qui peut apparaitre chez ces élèves va être de considérer que l'inégalité $u_n \leq v_n$ vérifiée pour tout entier naturel n, signifierait que v_n serait un majorant de la suite (u_n) et que u_n serait un minorant de la suite (v_n). Cependant, nous attendons qu'à ce niveau scolaire, une majorité des élèves ait bien pris conscience qu'un majorant et un minorant ne peuvent être des valeurs dépendant de n. Par conséquent, nous supposons que le fait d'avoir demandé aux élèves de travailler en binôme doit aussi les aider à surmonter cette difficulté lors des échanges entre eux. Nous pensons aussi que certains élèves vont

proposer une première succession d'inégalités de la forme : $u_0 \leq u_1 \leq u_2 \leq \ldots \leq u_n$ car la suite (u_n) est croissante, puis une seconde analogue concernant la décroissance de la suite (v_n), c'est-à-dire : $v_n \leq v_{n-1} \leq v_{n-2} \leq \ldots \leq v_0$. Nous supposons alors que ces deux séries d'inégalités peuvent s'avérer être une impasse chez certains élèves qui n'auraient pas pris conscience de l'importance de l'hypothèse supplémentaire vérifiée par les deux suites, c'est-à-dire $u_n \leq v_n$, pour tout entier naturel n.

Nous pensons aussi que des élèves de ce niveau scolaire peuvent chercher à justifier la validité de l'inégalité $u_n \leq v_n$, Pour cela, certains de ces élèves vont peut-être proposer une tentative erronée de démonstration par récurrence. Nous supposons aussi que d'autres vont chercher à déterminer la nature de la suite $(v_n - u_n)$ en se référant au fait que démontrer que $u_n \leq v_n$ équivaut à démontrer que $v_n - u_n \geq 0$. Pour cela, nous pensons qu'ils vont partir des monotonies respectives des suites (u_n) et (v_n) afin de déterminer le signe de la différence de $(v_{n+1} - u_{n+1}) - (v_n - u_n)$ pour en déduire la monotonie de la suite $(v_n - u_n)$, puis conclure à l'aide du *théorème de convergence monotone* après avoir justifié que cette suite était minorée par 0.

Nous pensons qu'une fois que les élèves auront obtenu un majorant de la suite (u_n) et un minorant de la suite (v_n), ils concluront dans un premier temps que ces deux suites sont convergentes. Cependant, nous supposons que certains élèves ne vont pas envisager qu'à ce stade de la démonstration, on n'a justifié que la convergence mais pas le fait que la limite des deux suites était la même. Et par conséquent, chez ces élèves nous nous attendons à ce qu'ils arrêtent là leurs raisonnements.

Cependant, nous pensons qu'une majorité d'élève va avoir conscience de ce fait concernant la limite, et que par conséquent qu'ils vont chercher à justifier l'égalité des deux limites. Pour cela, nous supposons que les élèves vont soit partir des propriétés opératoires sur les limites, ou proposer de partir de l'égalité $u_n = (u_n - v_n) + v_n$ valable pour tout entier naturel n et conclure en utilisant l'hypothèse que $u_n - v_n$ tend vers 0 quand n tend vers $+\infty$.

Nous ne nous attendons pas à ce que les élèves essayent d'utiliser un environnement numérique pour résoudre cet exercice.

3.4.1.2. Résolution de l'exercice 2 : Application de la propriété de la convergence de deux suites adjacentes

Nous supposons que chez des élèves de Terminale Scientifique, cet exercice va être mis en lien avec l'exercice précédent comme « application numérique » de la propriété sur deux suites particulières dont on connait les expressions en fonction de *n*. En effet, nous pensons que les élèves vont interpréter la question : « *Démontrer que les suites* (x_n) *et* (y_n) *convergent vers une même limite* » comme revenant à démontrer que ces deux suites sont adjacentes et par conséquent qu'elles vérifient la propriété démontrée dans l'exercice précédent.

Nous nous attendons à ce que les élèves éprouvent des difficultés à déterminer $x_{n+1} - x_n$ et $y_{n+1} - y_n$ en fonction de *n* afin de prouver que l'une est croissante et l'autre est décroissante. Nous pensons que certains élèves vont se contenter de calculer quelques valeurs de ces deux suites et de déterminer le signe d'un nombre fini de différence de quelques termes consécutifs pour chacune de ces deux suites et conclure sur les monotonies respectives. Nous pensons aussi que certains élèves vont proposer des algorithmes permettant de calculer un certain nombre de termes, voire de calculer un nombre fini de différences entre deux termes consécutifs, pour chacune de ces deux suites. Et ensuite, suivant les résultats observés lors de tests des algorithmes dans un environnement numérique, ils peuvent penser que ceci peut être suffisant pour justifier la nature de la monotonie de chaque suite, en particulier si le nombre de termes calculés, voire de différences est important, la machine pouvant « tourner » pour des valeurs de *n* assez grandes.

Une fois que les élèves auront déterminé les monotonies respectives des deux suites, nous pensons qu'ils vont vouloir déterminer l'expression de la différence $x_n - y_n$ en fonction de *n* afin de vérifier la dernière condition si la limite de $x_n - y_n$ permettant de conclure que les deux suites sont adjacentes. Nous pensons que certains élèves vont éprouver de nouveau des difficultés à obtenir l'égalité attendue $x_n - y_n = \frac{1}{n}$ pour conclure que la limite de $x_n - y_n$ est 0 quand *n* tend vers $+\infty$. Nous pensons qu'ici aussi les élèves peuvent proposer un algorithme permettant de calculer un nombre fini de valeurs de $x_n - y_n$ pour un *n* donné et conclure à partir des résultats obtenus une fois l'algorithme implémenté dans un environnement numérique.

Nous pensons qu'une fois que les élèves auront terminé une preuve (correcte ou partiellement correcte) que les deux suites sont adjacentes, ils vont appliquer la propriété

démontrée à l'exercice précédent et conclure sur la convergence des deux suites vers une même limite. Cette dernière étape ne doit pas poser de difficulté particulière aux élèves.

Nous prévoyons 30 minutes pour cette première phase.

3.4.2 Phase 6 : Construction d'une preuve dichotomique du TVI

Nous rappelons que pour cette dernière phase sur la dichotomie, les élèves élaborent une preuve du TVI basée sur le principe de dichotomie. Pour cela, nous pensons que les élèves vont s'inspirer de l'*algorithme de dichotomie* vue dans les phases 1, 3 et 4 de la première partie sur la « dichotomie continue » afin de construire deux suites adjacentes nécessaires à la démonstration. En effet, lors de la phase 1 de la première partie sur la dichotomie « continue », les élèves ont travaillé sur la résolution de l'équation « $f(x) = 0$ », telle que la fonction f définie sur **R** ait pour expression $f(x) = x^3 + x - 1$. Après avoir justifié que cette équation admet bien une unique solution dans **R**, les élèves ont dû utiliser la méthode de dichotomie afin d'obtenir un algorithme permettant d'afficher un encadrement à e près de la solution de l'équation $f(x)=0$ dans l'intervalle $[a ; b]$, où a, b et e ont été saisis par l'utilisateur, la fonction f étant entrée dans la variable F1, de l'onglet « Utiliser une fonction numérique » de l'environnement numérique *AlgoBox*. L'algorithme de cette phase 1 est donné selon la syntaxe d'*AlgoBox* (fig. 160).

Code de l'algorithme

```
1   VARIABLES
2     a EST_DU_TYPE NOMBRE
3     b EST_DU_TYPE NOMBRE
4     e EST_DU_TYPE NOMBRE
5     m EST_DU_TYPE NOMBRE
6   DEBUT_ALGORITHME
7     AFFICHER "Donner les valeurs des deux bornes de l'intervalle d'étude où se situe la
solution de l'équation"
8     LIRE a
9     LIRE b
10    AFFICHER "Donner l'amplitude de l'encadrement cherché"
11    LIRE e
12    TANT_QUE (b-a>=e) FAIRE
13      DEBUT_TANT_QUE
14      m PREND_LA_VALEUR (a+b)/2
15      SI (F1(a)*F1(m)<=0) ALORS
16        DEBUT_SI
17        b PREND_LA_VALEUR m
18        FIN_SI
19        SINON
20          DEBUT_SINON
21          a PREND_LA_VALEUR m
22          FIN_SINON
23      FIN_TANT_QUE
24    AFFICHER "Les valeurs des bornes cherchées sont : "
25    AFFICHER a
26    AFFICHER " et "
27    AFFICHER b
28  FIN_ALGORITHME
29
30 Fonction numérique utilisée :
31 F1(x)=pow(x,3)+x-1
```

Figure 160 (Algorithme sous AlgoBox sur la résolution de l'équation $x^3 + x - 1 = 0$ par la méthode de dichotomie)

Nous supposons ainsi que les élèves vont détailler le « parcours » de l'algorithme afin d'obtenir les deux suites permettant d'obtenir les valeurs des bornes de l'encadrement cherché. Ainsi, nous pensons qu'ils vont présenter un écrit proche de cette forme : « On part de deux valeurs entières (ou réelles) a et b telles que $a < b$. Puis, tant que $b - a \geq e$ (e > 0), on écrit : $a \leftarrow a$ et $b \leftarrow (a+b)/2$ si $f(a) \times f((a+b)/2) \leq 0$ sinon $a \leftarrow (a+b)/2$ et $b \leftarrow b$, avec f représentant la fonction associée à l'équation $f(x)=0$ ». Nous supposons ensuite que les élèves vont chercher à transcrire ces écritures en termes de suites. En effet, nous pensons que les élèves vont créer deux suites (a_n) et (b_n) correspondant respectivement aux valeurs successives des bornes inférieures et supérieures des intervalles emboités encadrant la solution de l'équation. En observant le déroulement de l'algorithme, nous pensons que les élèves vont obtenir les expressions suivantes pour les deux suites :

- $a_0 = a$ et $b_0 = b$;
- $a_{n+1} = a_n$ et $b_{n+1} = (a_n + b_n)/2$, si $f(a_n) \times f((a_n + b_n)/2) \leq 0$;
- $a_{n+1} = (a_n + b_n)/2$ et $b_{n+1} = b_n$, si $f(a_n) \times f((a_n + b_n)/2) > 0$.

Nous pensons que les élèves ayant acquis un savoir-faire au cours des phases 1 à 5, ne vont pas éprouver de difficulté particulière à obtenir les expressions de ces deux suites. En revanche nous pensons que la justification concernant le fait que ces deux suites sont adjacentes peut poser des difficultés chez certains élèves. En effet, le fait d'obtenir par exemple pour la suite (a_n) : $a_{n+1} - a_n = 0$ si $f(a_n) \times f((a_n + b_n)/2) \leq 0$, et $a_{n+1} - a_n = (b_n - a_n)/2$ si $f(a_n) \times f((a_n + b_n)/2) > 0$ peut poser des difficultés d'interprétation sur la monotonie de la suite pour certains élèves. Cette interprétation nécessite que les élèves aient conscience que le signe de $(b_n - a_n)$ est strictement positif. En effet, nous pensons que des élèves peuvent ne pas voir lien entre ce « problème » de signe et la condition de sortie de la boucle « TantQue ». Nous supposons que pour ces élèves va être de même ordre pour l'étude de la monotonie de la suite (b_n).

Nous pensons aussi que la détermination de la limite de l'expression $a_n - b_n$ peut poser des difficultés pour certains élèves. En effet, les élèves vont devoir conjecture une égalité (voire une inégalité suivant l'interprétation) de la forme $|a_n - b_n| = \frac{b-a}{2^n}$ (voire $|a_n - b_n| \leq \frac{b-a}{2^n}$), puis justifier ce résultat à l'aide d'un raisonnement par récurrence. Nous nous attendons que

les élèves est des difficultés à obtenir cette conjecture puis à énoncer une preuve de cette conjecture.

Nous pensons qu'une fois ce résultat trouvé sur l'expression $a_n - b_n$, nous pensons que les élèves vont se référer à un théorème de comparaison pour conclure sur la limite de l'expression $a_n - b_n$. Nous ne pensons que des élèves de ce niveau scolaire vont éprouver des difficultés particulières sur ce dernier point.

Nous pensons qu'ensuite, les élèves vont interpréter ces résultats en termes de suites adjacentes et conclure sur le fait que ces deux suites convergent vers une même limite L.

Nous pensons qu'une fois cette justification faite, les élèves ne vont pas nécessairement penser à exploiter la continuité de la fonction f afin de justifier que les suites $(f(a_n))$ et $(f(b_n))$ tendent vers une même limite $f(L)$ pour conclure sur la validité du TVI. En effet, nous que les élèves ne vont pas faire référence à la continuité pour l'obtention des limites des suites $(f(a_n))$ et $(f(b_n))$ et vont proposer le résultat sans aucune justification particulière.

3.5 Mise en place de l'expérimentation, déroulement et analyses *a posteriori*

3.5.1 La classe, les supports d'observations

Cette deuxième partie de la sous-ingénierie « dichotomie continue » est expérimentée une semaine après la première partie de la sous-ingénierie « dichotomie continue » seulement avec les élèves et l'enseignant de la classe de Terminale Scientifique en présence du chercheur qui en fait est aussi l'enseignant.

Les élèves sont en salle informatique et restent en binôme. Ces binômes sont identiques à ceux mis en place lors de la première partie sur la « dichotomie continue ». Chaque binôme a à sa disposition un ordinateur équipé des mêmes environnements informatiques utilisés depuis le début de l'ingénierie.

Tout au long de cette deuxième partie de la « dichotomie continue », chaque élève reste libre d'utiliser les menus « Graphique » et « Tableur » de sa calculatrice.

L'expérimentation est découpée en deux phases prévues durer une demi-heure chacune, soit une heure en tout. Les deux phases se déroulent sur une seule après-midi, au retour du temps de pause de l'après-midi et terminent la journée de présence en classe des élèves. Cette heure d'expérimentation se trouve être la première heure d'une séance de travaux dirigés

(TD) durant deux fois 55 minutes. Il est ainsi possible que pour des élèves qui n'auraient pas terminé pendant l'heure prévue initialement, ils puissent « déborder » d'une dizaine de minutes sur la deuxième heure de la séance de TD.

L'analyse *a posteriori* est construite à partir d'enregistrements audio des deux phases de l'expérimentation, des écrits numérisés des élèves et des algorithmes implémentés par ces derniers dans les environnements numériques (*LARP* et *AlgoBox*) mis à leur disposition. Les écrits numérisés et les algorithmes sont enregistrés sur une clé USB. Les enregistrements audio permettent de présenter les points forts observés, tant au niveau des binômes que de l'enseignant. L'enseignant et le chercheur étant la même personne, de nombreux points d'enregistrements audio sont mis en place permettant d'avoir un retour de l'ensemble des binômes et de « libérer » l'enseignant de sa tâche de chercheur lors du déroulement des deux phases.

3.5.2 Phase 5 : Une introduction à la notion des suites adjacentes

3.5.2.1. Le déroulement (30 minutes)

Au début de cette phase, l'enseignant interroge les élèves d'un binôme qui rappellent la définition de deux *suites adjacentes* ainsi que les énoncés de deux propriétés qui leur sont associées. Celles-ci ont été données la veille sans aucun commentaire de la part de l'enseignant à la fin d'un cours qui clôturait le travail demandé par l'institution sur le chapitre intitulé « Suites numériques ». Ce temps consacré au rappel de la définition et des deux propriétés ne dure que deux minutes.

Une fois ce temps de rappel fait, les élèves ont tous devant eux les énoncés suivants (Fig. 161) :

Enoncé 1	Enoncé 2
Les définitions et propriétés suivantes sont supposées admises : 1) deux suites (u_n) et (v_n) sont adjacentes lorsque : l'une est croissante, l'autre est décroissante et $u_n - v_n$ tend vers 0 quand n tend vers $+\infty$; 2) si (u_n) et (v_n) sont deux suites adjacentes telles que (u_n) est croissante et (v_n) est décroissante, alors pour tout n appartenant à **N**, on a $u_n \leq v_n$; 3) toute suite croissante et majorée est convergente ; toute suite décroissante et minorée est convergente. Démontrer alors la propriété suivante : Deux suites adjacentes sont convergentes et elles ont la même limite.	Soient deux suites (x_n) et (y_n) définies pour $n>0$ par les relations : $x_n = 1/n + 1/(n+1) + \cdots + 1/(2n)$ et $y_n = 1/(n+1) + 1/(n+2) + \cdots + 1/(2n)$. Démontrer que les suites (x_n) et (y_n) convergent vers une même limite. (On ne demande pas de déterminer la valeur de cette limite.)

Figure 161 (Deux énoncés sur les suites adjacentes)

Nous observons que 40% des binômes se lancent dans la résolution successive des deux exercices en respectant l'ordre de ceux-ci. De plus, 55% des binômes proposent d'admettre la proposition à démontrer de l'énoncé 1, afin de résoudre d'abord l'énoncé 2. Les 5% qui restent se contentent de « pianoter » sur leur clavier d'ordinateur malgré les rappels de l'enseignant.

Nous observons que la résolution de l'énoncé 1 demande environ une vingtaine de minutes pour les binômes qui cherchent à le résoudre. Pour l'énoncé 2, nous observons que ces élèves restent une dizaine de minutes pour essayer de résoudre cet exercice.

Quant aux 60% des binômes qui débutent avec l'énoncé 2, nous n'observons que seulement 8% d'entre eux vont essayer de résoudre l'énoncé 1, une fois qu'ils ont eu considéré que la résolution de l'exercice 2 est terminée. Les autres vont se contenter de « vérifier » le travail fait sur l'énoncé 2 ou patienter jusqu'à la fin de la phase.

Nous remarquons qu'aucun binôme n'a cherché à travailler avec l'ordinateur en particulier lors de la résolution de l'exercice 2.

3.5.2.2. Analyse *a posteriori*

Nous observons que l'ensemble des binômes semblent maitriser sans difficulté apparente les compétences de base sur les suites adjacentes données la veille par l'enseignent. En effet, pendant ce très cours temps de rappel et lors de l'interrogation des deux élèves d'un binôme, nous constatons que d'autres élèves demandent aussi à être interrogés afin de montrer qu'ils ont « retenu » la leçon. Un élève propose même un cas particulier de suites adjacentes et de suites non adjacentes (Fig. 162) pour illustrer sa compréhension de ce nouveau savoir.

Les suites (u_n) et (v_n) définies par : $u_n = -\frac{1}{n+2}$ et $v_n = \frac{1}{n+1}$ sont adjacentes, car la suite (u_n) est croissante et la suite (v_n) est décroissante. Et, la limite de v_n - u_n quand n tend vers $+\infty$ est zéro.	Les suites (u_n) et (v_n) définies par : $u_n = 4 - \frac{1}{n+1}$ et $v_n = 1 + \frac{1}{n^2+1}$ sont respectivement croissante et décroissante. Mais, la limite de v_n - u_n quand n tend vers $+\infty$ n'est pas égale zéro. Donc ces suites ne sont pas convergentes.

Figure 162 (Extrait d'un enregistrement audio concernant les réponses d'un élève sur les suites adjacentes)

a) Cas de l'énoncé : démonstration d'une propriété

Nous observons que parmi les 40% de binômes qui débutent avec l'énoncé 1, 67% d'entre eux éprouvent des difficultés à justifier le fait que la suite (u_n) est majorée et que la suite (v_n) est minorée. En effet, un tiers de ces binômes écrivent que le fait que les suites vérifient la condition « $u_n \leq v_n$, pour tout entier naturel n » permet d'obtenir la conclusion erronée

suivante : « […], la suite (u_n) est majorée par v_n et la suite (v_n) est minorée par u_n ». On observe ainsi une mauvaise interprétation des notions de majoration et de minoration chez ces élèves. Ils semblent ne prendre en compte que l'inégalité mais pas le fait que la borne (majorant ou minorant) doit être un nombre ne dépendant pas de l'entier n.

Nous observons aussi qu'un deuxième tiers des 67% des élèves ayant des difficultés sur la majoration et la minoration écrivent correctement une succession d'inégalités de la forme « pour tout entier n, on a $u_0 \leq u_1 \leq u_2 \leq \ \ldots \leq u_{n-1} \leq u_n \leq v_n \leq v_{n-1} \leq \ldots \leq v_2 \leq v_1 \leq v_0$ » mais concluent que « le premier terme v_n qui ne dépend pas de n » (par exemple, certains citent v_2) « est un majorant de la suite (u_n) ». Ainsi, ici aussi, il semble que les élèves éprouvent des difficultés à avoir une interprétation correcte de la notion d'indépendance de n pour obtenir un majorant (resp. un minorant). Cependant, nous observons que ce groupe d'élèves concernés par ce type de conclusion se questionnent sur l'exactitude ou non que v_2 soit un majorant. En effet, certains demandent à l'enseignant s'ils peuvent conclure cela en faisant la remarque sur les conditions que devraient vérifier ou non les deux derniers termes v_1 et v_0. Nous observons que sur les copies numérisées, ces élèves proposent en fait comme majorant v_0. Seuls les enregistrements audio permettent de voir ce type de réponse sur v_2. Nous avons bien entendu, le même type de progression dans leurs raisonnements par ces élèves sur le minorant de la suite (v_n).

Le dernier tiers des 67% des élèves ayant éprouvés des difficultés à justifier l'existence et la valeur d'un majorant et d'un minorant, se contente d'écrire un résultat correct mais non argumenté. En effet, nous trouvons comme type de réponse proposée par les élèves : « La suite (u_n) est majorée par le plus grand terme de la suite décroissante (v_n) et la suite (v_n) est minorée par le plus petit terme de la suite croissante (u_n) ». Nous pensons que ces élèves ont le sens de la question mais éprouvent encore des difficultés à organiser une démonstration mathématique dans le cas où aucune valeur numérique ou expression en fonction de n ou par récurrence de la suite ne serait donnée. Nous pouvons supposer que ceci nous renvoie aux difficultés observées chez certains élèves lors des phases précédentes où les travaux concernaient la « dichotomie continue » appliquée à des fonctions dont les expressions n'étaient pas connues.

Nous observons aussi que les autres 33% des binômes qui débutent la phase avec l'énoncé 1, écrivent aussi que « les suites (u_n) et (v_n) sont deux suites [adjacentes[98]] telles que (u_n) est croissante et (v_n) décroissante. On a alors respectivement $u_0 \leq u_1 \leq u_2 \leq \cdots \leq u_{n-1} \leq u_n$ et $v_n \leq v_{n-1} \leq \cdots \leq v_2 \leq v_1 \leq v_0$ ». Puis, ils complètent leur raisonnement en écrivant que l'hypothèse « […] pour tout *n* appartenant à **N**, on a $u_n \leq v_n$ », en ajoutant « car les deux suites sont aussi adjacentes », leur permet de conclure que « la suite (u_n) (resp. (v_n)) admet un majorant qui vaut v_0 (resp. u_0) ».

Cependant, nous observons que la conclusion sur la convergence des deux suites ne pose pas de difficulté particulière une fois que les élèves ont comme résultat (justifié correctement ou pas) que l'une des suites est croissante majorée et l'autre est décroissante minorée. En effet, une fois ce résultat obtenu, nous constatons qu'ils se réfèrent tous au *théorème de la convergence monotone* pour conclure sur le fait que les deux suites convergent. En revanche, nous observons que 10% font une mauvaise interprétation de ce théorème. En effet, ils concluent que « ces deux suites convergent vers une même limite » sans aucun autre complément de justification. Quant aux autres qui interprètent le théorème comme justifiant la convergence sans pour autant donner la valeur de la limite finie de la suite, ils proposent d'écrire que les deux suites convergent respectivement vers deux réels L et L'. Puis, ils cherchent à démontrer que « L = L'».

Nous observons deux variantes de justification de l'égalité des deux limites chez les élèves qui proposent une preuve. En effet, parmi les 90% des binômes qui proposent une démonstration, environ 75% introduisent une suite (w_n) qui est définie par $w_n = v_n - u_n$ et justifient en se référant à une propriété opératoire sur les limites que la suite (w_n) converge vers le réel L' – L. Ensuite, ils concluent à l'aide de l'unicité de la limite lorsqu'elle existe. En effet, nous constatons qu'ils posent que la limite de w_n, quand *n* tend vers + ∞, est égale à la limite de $v_n - u_n$, quand *n* tend vers + ∞. Puis, ils observent alors que L' – L = 0 et concluent. Parmi les autres 15%, nous observons que 10% des binômes utilisent une égalité entre v_n et u_n de la forme $v_n = (v_n - u_n) + u_n$et concluent sur l'égalité des limites en utilisant une propriété opératoire des limites et l'hypothèse que la limite de $v_n - u_n$ est égale à zéro quand *n* tend vers + ∞. Parmi les 5% qui restent, nous avons une proposition de *raisonnement par*

[98] Ce mot n'est pas nécessairement écrit dans les copies rendues par les binômes. Cependant, cela n'affecte pas la validité de la démonstration.

l'absurde qui n'avait pas été envisagé lors de l'analyse *a priori*. En effet, un binôme propose le raisonnement suivant : « *Supposons que l'une des deux suites diverge vers + ∝ ou − ∝, alors la limite de l'expression* $|v_n - u_n|$ *est égale à + ∝ quand n tend vers + ∝. Or, ceci n'est pas possible car les deux suites sont adjacentes. Par conséquent, les deux suites convergent respectivement vers des limites finies L et L'. Or, les deux suites sont adjacentes, alors la limite de* $|v_n - u_n|$ *est égale à 0 quand n tend vers + ∝. Et par unicité de la limite,* $L - L' = 0$*. D'où la conclusion.* » (Extrait de la copie de ce binôme).

Ainsi, nous observons que des élèves de ce niveau scolaire, ont conscience que les problématiques de convergence et de calcul de la valeur d'une limite à l'infini ne sont pas équivalentes.

b) Cas de l'énoncé sur une application numérique de la propriété démontrée

Parmi les 95% des binômes qui travaillent l'énoncé 2, nous observons qu'aucun binôme ne cherche à s'aider de l'ordinateur. En effet, nous pensions que certains pourraient chercher à construire un (ou des) algorithme(s) permettant de calculer un certain nombre de termes à l'aide des expressions des deux suites numériques, ou encore étudier un certain nombre de différences de termes de même rang des deux suites. Nous observons ainsi que ces élèves n'ont pas nécessairement le réflexe de penser ou de transcrire une question mathématique sous forme d'une question qui serait liée à la construction d'algorithmes et à leurs implémentations dans un environnement numérique afin de les tester. Cet aspect, nous semble intéressant, sachant que l'ensemble des élèves de cette classe a participé aux trois ingénieries décrites dans cette seconde partie de la thèse, et que cette ingénierie sur la dichotomie arrive après le travail fait sur l'*algorithme de Kaprekar*. En revanche, nous observons que 20% de ces élèves utilisent leur calculatrice afin de déterminer quelques valeurs des deux suites pour conjecturer sur la monotonie de chacune des deux suites, sans pour autant créer un programme. Comme pour l'*algorithme de Kaprekar*, nous observons cependant, que les élèves ne se contentent pas de conjecturer mais cherchent aussi à justifier par des raisonnements mathématiques la conjecture obtenue.

Bien que l'énoncé ne donne aucune précision quant à la démarche mathématique à suivre pour démontrer que les deux suites convergent, nous observons que l'ensemble des binômes adoptent une procédure analogue pour cet aspect du problème. En effet, tous les binômes

mettent en lien cet exercice avec le précédent où le travail consistait à démontrer une propriété de convergence sur les suites adjacentes. Ainsi, mêmes les binômes qui n'ont pas su ou voulu résoudre le premier énoncé sur cette propriété, ont admis la propriété afin de répondre à la problématique du deuxième énoncé. Lors d'échanges avec le chercheur après la fin de cette phase, certains élèves ont présenté cette façon de faire en citant une remarque antérieure à l'expérimentation de leur enseignant : « Dans le cas d'une difficulté, une réponse (ici la démonstration d'une propriété) à une question peut être admise afin de poursuivre la résolution de l'exercice ».

Nous observons que 95% des binômes qui travaillent cet énoncé, proposent pour l'étude de la monotonie de la suite (x_n) le raisonnement suivant (Fig. 163) :

> Pour tout entier n non nul, $x_{n+1} = 1/(n+1) + 1/(n+2) + \cdots + 1/(2(n+1))$ et $x_n = 1/n + 1/(n+1) + \cdots + 1/(2n)$, d'où la différence membre à membre des deux termes des égalités donne : $x_{n+1} - x_n = \frac{1}{2n+2} + \frac{1}{2n+1} - \frac{1}{n} = -\frac{3n+2}{n(2n+1)(2n+2)} < 0$.
> Alors la suite (x_n) est décroissante.

Figure 163 (Extrait d'une copie d'un binôme A sur l'étude de la monotonie d'une suite)

Nous observons que pour 5% des binômes qui cherchent à résoudre l'énoncé 2, ils calculent quelques termes (par exemple x_1, x_2 et x_3) de la suite (x_n) avec la calculatrice, puis les différences $x_2 - x_1$ et $x_3 - x_2$ et concluent sur la monotonie de la suite (x_n) en fonction du signe du nombre qu'ils obtiennent pour ces deux différences. Nous constatons que ces binômes qui se « contentent » de travailler avec quelques calculs de différences, sont des élèves qui au cours des ingénieries ont demandé à avoir leurs calculatrices avec eux, mais aussi un accès à l'ordinateur, même si la phase ne l'autorisait pas. De plus, après échanges entre le chercheur et les élèves concernés, nous apprenons que ces élèves ont proposé au cours de leur classe de Première Scientifique des TPE[99] sur l'utilisation de l'informatique en particulier avec une utilisation conséquente de l'algorithmique et de la programmation sur des problématiques en lien avec la géométrie, la trigonométrie et la simulation du mouvement d'un coude.

85% des mêmes binômes qui proposent une démonstration valable pour tout entier n non nul, pratiquent un raisonnement analogue pour la suite (y_n) pour conclure sur la croissance de celle-ci. Cependant, pour 15% des binômes qui ont proposé une démonstration correcte pour la monotonie de la suite (x_n), nous observons qu'ils constatent que la suite (y_n) vérifie

[99] Travaux personnels encadrés

l'égalité suivante $y_n = x_n - 1/n$. Nous observons alors, que ces binômes proposent ensuite le raisonnement suivant (Fig. 164) pour justifier la croissance de la suite (y_n).

Nous avons $y_n = x_n - 1/n$, avec $n > 0$.
Par conséquent $y_{n+1} - y_n = (x_{n+1} - 1/(n+1)) - (x_n - 1/n) = (x_{n+1} - x_n) + 1/n - 1/(n+1)$.
Alors $y_{n+1} - y_n = -\frac{3n+2}{n(2n+1)(2n+2)} + \frac{1}{n(n+1)} = \frac{1}{2(n+1)(2n+1)} > 0$.
Donc (y_n) est croissante.

Figure 164 (Extrait d'une copie d'un binôme B sur l'étude de la monotonie d'une suite)

Quant aux élèves qui ont raisonné sur des termes particuliers de la suite (x_n), nous observons sans surprise une procédure analogue pour l'étude de la monotonie de la suite (y_n).

Ensuite, nous observons qu'à ce stade de la résolution de l'exercice, les binômes se citent la définition des suites adjacentes. Par exemple, un binôme écrit : « *Nous avons justifié que nous avions une suite croissante et une suite décroissante. Alors, d'après la définition écrite au tableau, il suffit de déterminer la limite de la différence des deux expressions des suites* ». Nous constatons que les élèves proposent diverses approches pour justifier que $x_n - y_n = 1/n$. En effet, environ 72% des binômes ayant répondu à cette question partent des égalités données dans l'énoncé et procèdent à une soustraction membre à membre entre les deux égalités. Puis, ils éliminent les termes du second membre de l'égalité qui peuvent l'être afin d'obtenir une écriture de la forme $x_n - y_n = 1/n$ ou $y_n - x_n = -1/n$. 20% des binômes partent alors de l'égalité $y_n = x_n - 1/n$ (cf. ci-dessus) pour conclure. En revanche, les binômes qui ont proposé une étude des monotonies basée sur les calculs de trois ou quatre premiers termes respectifs des deux suites, restent dans cette perspective pour justifier que la différence des termes de même rang des deux suites tend vers zéro quand n tend vers $+\infty$. Cependant, pour justifier cela, ils proposent le calcul d'un nombre conséquent de termes (environ 7 à 8 termes pour chaque suite) qu'ils font à l'aide de la calculatrice sans créer la capacité de programmation que leur offre la calculatrice. Puis, ils concluent, qu'« il semble que $x_n - _{yn}$ tend vers 0 quand n devient de plus en plus grand ».

Nous observons que seulement 3% des binômes proposent comme réponse au problème : « $y_n = x_n - 1/n$. *Alors d'après les règles opératoires,* y_n *et* x_n *ont même limite car la limite de 1/n est 0 quand n tend vers* $+\infty$ ». Nous observons ainsi que ces élèves n'ont pas conscience que pour justifier ce qu'ils écrivent, ils doivent d'abord prouver que les suites (x_n) et (y_n) admettent une limite. En effet, ces élèves ne proposent pas une démonstration complète sur le fait que les deux suites pourraient être adjacentes. Ils se « contentent » d'écrire une démonstration correcte sur l'étude des monotonies, puis ils cherchent à justifier que la différence des termes

de même rang des deux suites tend bien vers 0, mais le fait d'écrire que $y_n = x_n - 1/n$ semble arrêter leur démonstration que les suites seraient adjacentes pour conclure sur l'égalité des limites. Lors d'un échange après cette expérimentation, un élève a répondu au chercheur qu'il ne voyait plus la nécessité d'aller au bout du raisonnement sur les suites adjacentes, car l'égalité $y_n = x_n - 1/n$ permettait de répondre directement au problème d'après lui. Pour compléter ces propos, ce même élève ajoute que l'auteur de l'exercice ne demandait pas de justifier que les suites étaient adjacentes, ni de calculer la valeur de la limite. Pour lui, seule la justification de l'égalité des limites était à prendre en compte.

Parmi les binômes qui ont désiré mener une démonstration complète : existence de la convergence pour chacune des suites puis qu'elles convergent vers une même limite, nous observons que ceux-ci cherchent à se ramener aux conditions du *théorème de la convergence monotone*. Ainsi, nous remarquons que le problème de majoration (resp. de minoration) ne semble pas poser de difficulté majeure pour l'ensemble de ces binômes. En effet, sur de nombreuses copies rendues par les binômes, nous voyons écrit le résultat suivant (Fig. 165).

Les deux suites sont adjacentes, alors $y_n \leq x_n$.
La suite (x_n) est décroissante, alors pour tout entier n, $x_n \leq x_0$.
Donc la suite croissante (y_n) est majorée par x_0. Et elle alors convergente.
De même, comme la suite (y_n) est croissante, alors la suite croissante (x_n) est minorée par y_0. Elle converge aussi.
Conclusion : Les suites (x_n) et (y_n) admettent donc comme limites L_1 et L_2.

Figure 165 (Extrait d'une copie du binôme B sur l'étude de deux suites adjacentes)

Nous observons aussi qu'un élève d'un binôme propose oralement, le résultat suivant pour la suite (x_n) : « *la suite décroissante (x_n), est minorée par 0 car elle est constituée de termes positifs* ».

Pour conclure sur cette phase 5, nous observons aussi que la durée prévue initialement pour l'ensemble de la phase est respectée par les élèves. En effet, au bout des 30 minutes prévues, les binômes rendent leurs travaux numérisés et se concentrent rapidement sur les objectifs de la phase suivante.

3.5.3 Phase 6 : Construction d'une preuve dichotomique du TVI

3.5.3.1. Le déroulement (30 minutes)

Nous rappelons que lors de cette dernière phase, les élèves doivent répondre à la question « *Construire une preuve du Théorème des valeurs intermédiaires à l'aide de la méthode de dichotomie* ». Les élèves sont toujours dans la salle informatique. Les binômes ne changent

pas et les ordinateurs sont allumés avec la possibilité d'accéder aux fichiers enregistrés dans l'environnement *AlgoBox* lors des phases précédentes. L'enseignant/chercheur n'intervient pas, il reste un observateur passif pendant toute cette phase. Les élèves travaillent pendant cette phase seulement 30 minutes comme il était convenu lors de l'élaboration de l'expérimentation.

3.5.3.2. Analyse *a posteriori*

Nous observons au début de cette dernière phase un moment de flottement chez les élèves qui dure environ 2 à 3 minutes sur la démarche qu'ils doivent mettre en place afin de répondre au problème posé. En effet, nous constatons que le fait d'avoir choisi une question « partiellement ouverte » ne favorise pas la pris d'initiative d'un grand nombre d'élèves. Cependant, rapidement au cours de cette courte période d'incertitude, les deux tiers des binômes se questionnent sur la condition « à l'aide de la méthode de dichotomie ». En effet, nous avons par exemple cet échange entre quatre élèves de deux binômes voisins :

El1B1 : […] *pourquoi méthode dichotomie ?*
El1B2 : *Sais pas !*
El1B1 : *Peut-être que ce théorème est comme celui sur les gendarmes…*
El2B1 : *Quoi ? le TVI serait à la fois pour les fonctions et pour les suites ?*
El1B1 : *Probable.*

Commentaire : Un élève du binôme 2 (le seul qui n'a pas pris la parole jusque-là) ouvre un des fichiers *AlgoBox* sur la dichotomie utilisé lors des phases 3 et 4 (cf. fig. 166). Les quatre élèves demandent à l'enseignant s'ils peuvent imprimer les algorithmes des phases 1, 3 et 4 sur la « dichotomie continue ». L'enseignant donne son accord. Au bout d'une minute l'échange reprend entre ces quatre élèves.

El1B1 : (Il prend un des algorithmes imprimé) *Euh ! Idée. Faisons tourner l'algo à la main*

Commentaire : cet aspect du travail n'est pas rendu par ces élèves. Ils ont pris une feuille de brouillon et on commencer à tester l'algorithme manuellement pendant 2 à 3 minutes.

El1B1 : […] *Bof les calculs… Modifions l'algorithme pour faire afficher les valeurs de* a_n *et* b_n (cf. fig. 167).

Binôme n° 5 TS 5

Noms : [redacted]

Prénoms : [redacted]

Objectif : Démontrer le TVI avec la méthode de dichotomie

Algorithme de dichotomie	Interprétation
FONCTIONS_UTILISEES VARIABLES a EST_DU_TYPE NOMBRE b EST_DU_TYPE NOMBRE h EST_DU_TYPE NOMBRE DEBUT_ALGORITHME AFFICHER "Entrer les bornes inf et sup de l'intervalle : " LIRE a LIRE b AFFICHER "Entrer la précision : " LIRE h TANT_QUE (b-a>=h) FAIRE DEBUT_TANT_QUE SI (F1(a)*F1((a+b)/2)<0) ALORS DEBUT_SI b PREND_LA_VALEUR (a+b)/2 FIN_SI SINON DEBUT_SINON a PREND_LA_VALEUR (a+b)/2 FIN_SINON FIN_TANT_QUE AFFICHER a AFFICHER " <= solution < " AFFICHER b FIN_ALGORITHME	On a une fonction F1 continue sur un intervalle [a ; b]. On suppose que F1(a) et F1(b) ne sont pas de même signe. Il existe alors au moins un réel c tel que F1(c)= 0 (TVI). On prend deux valeurs a et b que l'on stocke dans a0 et b0. a0= a et b0= b (les bornes de l'intervalle du départ [a ; b]. On fixe une précision h. On suppose que la différence b0 – a0 est plus grande que h. Alors on fait : a1=a0=a et b1=(a0+b0)/2=(a+b)/2 si F1(a0)*F1((a0+b0)/2)<0, sinon a1=(a0+b0)/2=(a+b)/2 et b1=b0=b. Puis, on suppose que b1 – a1 est encore plus grande que h. a2=a1 et b2=(a1+b1)/2 si F1(a1)*F1((a1+b1)/2)<0, sinon a2=(a1+b1)/2 et b2=b1. Et encore, on suppose que b2 – a2 est encore plus grande que h. a3=a2 et b3=(a2+b2)/2 si F1(a2)*F1((a2+b2)/2)<0, sinon a3=(a2+b2)/2 et b3=b2. Et ainsi de suite, tant que bn – an ≥ 0. On obtient deux suites (an) et (bn) : an=an-1 et bn=(an-1+bn-1)/2 si F1(an-1)*F1((an-1+bn-1)/2)<0 an=(an-1+bn-1)/2 et bn=bn-1 si F1(an-1)*F1((an-1+bn-1)/2)>0 On a posé que a0=a et b0=b.

Figure 166 (Copie de deux élèves sur la démonstration du TVI par la dichotomie)

Binôme n° 5

On étudie les suites (an) et (bn) pour voir si elles convergent vers une même limite L.

En prenant la fonction F1(x)=x^3+x-1 sur l'intervalle [0 ; 1] car F1(o)=-1 et F1(1)=1 donc pas le même signe, on prend une précision h=0,0001et on calcule des valeurs de a1, b1, a2, b2, a3, b3, a4, b4, … et on voit que la suite (an) est normalement croissante. On voit aussi que la (bn) est normalement décroissante.

On modifie l'algorithme précédent.

Code de l'algorithme

```
1  VARIABLES
2    a EST_DU_TYPE NOMBRE
3    b EST_DU_TYPE NOMBRE
4    h EST_DU_TYPE NOMBRE
5    n EST_DU_TYPE NOMBRE
6  DEBUT_ALGORITHME
7    AFFICHER "La précision est "
8    h PREND_LA_VALEUR 0.0001
9    AFFICHER h
10   AFFICHER "Les bornes inf et sup de l'intervalledu départ sont 0 et 1. "
11   AFFICHER "Pour n=0, on a a0=0 et b0=1"
12   a PREND_LA_VALEUR 0
13   b PREND_LA_VALEUR 1
14   n PREND_LA_VALEUR 0
15   TANT_QUE (b-a>=h) FAIRE
16     DEBUT_TANT_QUE
17     SI (F1(a)*F1((a+b)/2)<0) ALORS
18       DEBUT_SI
19       b PREND_LA_VALEUR (a+b)/2
20       FIN_SI
21       SINON
22         DEBUT_SINON
23         a PREND_LA_VALEUR (a+b)/2
24         FIN_SINON
25     n PREND_LA_VALEUR n+1
26     AFFICHER "Pour n="
27     AFFICHER n
28     AFFICHER ", on a an= "
29     AFFICHER a
30     AFFICHER " et bn= "
31     AFFICHER b
32     AFFICHER " et bn-an="
33     AFFICHERCALCUL b-a
34     FIN_TANT_QUE
35   AFFICHER a
36   AFFICHER " <= solution < "
37   AFFICHER b
38 FIN_ALGORITHME
39
40 Fonction numérique utilisée :
41 F1(x)=pow(x,3)+x-1
```

Console

```
La précision est 0.0001
Les bornes inf et sup de l'intervalledu départ sont 0 et 1.
Pour n=0, on a a0=0 et b0=1
Pour n=1, on a an= 0.5 et bn= 1 et bn-an=0.5
Pour n=2, on a an= 0.5 et bn= 0.75 et bn-an=0.25
Pour n=3, on a an= 0.625 et bn= 0.75 et bn-an=0.125
Pour n=4, on a an= 0.625 et bn= 0.6875 et bn-an=0.0625
Pour n=5, on a an= 0.65625 et bn= 0.6875 et bn-an=0.03125
Pour n=6, on a an= 0.671875 et bn= 0.6875 et bn-an=0.015625
Pour n=7, on a an= 0.6796875 et bn= 0.6875 et bn-an=0.0078125
Pour n=8, on a an= 0.6796875 et bn= 0.68359375 et bn-an=0.00390625
Pour n=9, on a an= 0.68164063 et bn= 0.68359375 et bn-an=0.001953125
Pour n=10, on a an= 0.68164063 et bn= 0.68261719 et bn-an=0.0009765625
Pour n=11, on a an= 0.68212891 et bn= 0.68261719 et bn-an=0.00048828125
Pour n=12, on a an= 0.68212891 et bn= 0.68237305 et bn-an=0.00024414063
Pour n=13, on a an= 0.68225098 et bn= 0.68237305 et bn-an=0.00012207031
Pour n=14, on a an= 0.68231201 et bn= 0.68237305 et bn-an=0.000061035156
0.68231201 <= solution < 0.68237305
```

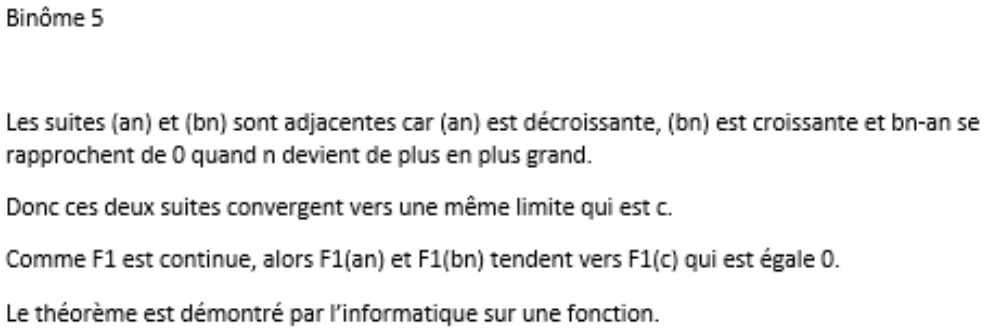
Les valeurs an et bn se rapprochent de façon que l'intervalle obtenu encadre la valeur c telle que F1(c)=0.

Figure 167 (Copie d'un binôme montrant une utilisation d'AlgoBox pour l'implémentation d'un algorithme de dichotomie appliqué à la fonction f définie sur [0 ; 1], par $f(x) = x^3 + x - 1$)

Nous pouvons observer sur cet exemple assez représentatif du travail fait en début de cette phase, que les élèves associent la démarche de preuve du théorème à l'action menée lors de l'exécution de l'algorithme de dichotomie sans toutefois en avoir saisi le sens de « transposition ». En effet, nous remarquons que la première approche de cette preuve à mettre en place autour de la *méthode de dichotomie* amène un certain nombre de binômes

(environ 40%) à faire une analogie erronée avec la place d'un théorème particulier (le *théorème des gendarmes*[100]) qui se trouve être à la fois utilisable dans le champ des fonctions et le champ des suites numériques.

Nous constatons aussi que les élèves étudient les différentes étapes du processus de calcul mis en place lors de l'exécution de l'algorithme de « dichotomie continue » (fig. 166, 167 et 168).


Binôme 5

Les suites (an) et (bn) sont adjacentes car (an) est décroissante, (bn) est croissante et bn-an se rapprochent de 0 quand n devient de plus en plus grand.

Donc ces deux suites convergent vers une même limite qui est c.

Comme F1 est continue, alors F1(an) et F1(bn) tendent vers F1(c) qui est égale 0.

Le théorème est démontré par l'informatique sur une fonction.

Figure 168 (Copie d'un binôme sur les suites adjances et une démonstration du TVI)

Ainsi, nous observons sur cette copie (représentative d'une grande majorité des travaux rendus par les différents binômes (environ 90% des binômes, les autres binômes restent passifs pendant cette phase) que les élèves ont modifié l'algorithme d'une des phases précédentes afin de faire afficher les valeurs successives des suite (a_n) et (b_n) pour une précision fixée et une fonction particulière vérifiant les conditions du théorème. Nous observons aussi qu'ils mettent en lien ces deux suites avec le concept de suites adjacentes et qu'ils font référence à la continuité de la fonction sur l'intervalle initial pour conclure quant à la validité du théorème. En revanche, nous observons que leur démonstration se base sur une application numérique de l'algorithme de « dichotomie continue » pour une fonction particulière. Nous pouvons penser que des élèves de ce niveau se contentent d'une telle « preuve » car ils peuvent éprouver des difficultés pour travailler sur une fonction qui vérifierait les conditions du théorème mais dont on ne connaitrait pas son expression. Cet aspect semble confirmer les difficultés observées aux phases 3 et 4, où les élèves étaient amenées à travailler sur l'algorithme de « dichotomie continue » sans connaitre l'expression de la fonction.

[100] Soit (u_n), (v_n) et (w_n) trois suites. Si pour tout entier naturel n supérieur à un certain n_0, $v_n \leq u_n \leq w_n$, et si les suites (v_n) et (w_n) convergent vers le même réel L, alors la suite (u_n) est convergente et sa limite est L.

De plus, nous observons qu'aucun binôme ne cherche à déterminer une expression de la différence $b_n - a_n$ en fonction de n, a_0 et b_0 pour calculer la limite de cette différence quand n tend vers 0. L'ensemble des binômes qui rendent un travail reste dans un *Espace de Travail Algorithmique* pour répondre à une preuve qui demanderait aussi un travail dans un *Espace de Travail Mathématique Spécifique* à l'analyse. Nous pouvons supposer que cela vient entre autres du fait que l'introduction de la notion des suites adjacentes n'est pas officiellement au programme et que le temps de recherche et d'écriture laissé aux élèves pour cette dernière phase ne soit que de 30 minutes.

Nous observons toutefois qu'à la fin de la phase, lors du retour des copies des binômes, un élève d'un binôme exprime oralement l'idée qu'un raisonnement par récurrence pourrait être utilisé afin de travailler sur des suites dont on ne connaitrait pas la succession des valeurs respectives des deux suites utilisées. Mais cet élève ne poursuit pas plus loin sur cette hypothèse qu'il émet.

3.6 Synthèse en termes d'analyse en lien avec les Espaces de Travail Mathématique et Algorithmique des phases 5 et 6 de la sous-ingénierie « dichotomie continue »

Afin de terminer l'analyse de ces deux dernières phases sur la dichotomie, nous souhaitons revenir sur une approche de l'analyse *a posteriori* en termes de travail mener par les élèves dans des Espaces de Travail Algorithmique et Mathématique afin de mettre en évidence les apports que pourraient avoir l'algorithmique lors d'une démarche de preuve d'un théorème classique (ici le TVI) au niveau de la Terminale Scientifique dans un domaine précis des mathématiques.

3.6.1 Cas de la phase 5 : Preuve de la propriété sur la convergence de deux suites adjacentes et application numérique

Lors de la phase 5, les élèves devant démontrer une propriété sur la convergence de deux suites adjacentes, montrent qu'ils restent dans un *ETM* d'analyse sans proposer de mettre en lien avec une démarche possible dans un *ETA*. De même, nous observons que les élèves pour l'application numérique de cette propriété, ne proposent toujours pas un travail se situant dans un *ETA* et restent encore dans une démarche associée à un *ETM*. Nous pensons que le fait de faire travailler les élèves sur une preuve d'une propriété mettant en jeu des objets mathématiques que seraient les suites adjacentes et dont on ne connaitrait pas leurs expressions respectives afin d'avoir une propriété valable quel que soit les suites adjacentes,

ne favorisent pas chez des élèves de ce niveau scolaire relativement débutant en informatique, de procéder en premier lieu par un travail de type algorithmique afin d'émettre une conjecture qui serait obtenue dans un *ETA* et validée dans un *ETM*. Cependant, nous voyons lors de la phase suivante que des élèves confrontés à de nouveaux concepts mathématiques peuvent considérer qu'une « preuve » même incomplète suite à son cas « factuel », peut être validée de manière erronée comme une « preuve générale ».

Nous constatons aussi que les élèves de ce niveau n'ont pas nécessairement conscience qu'une application numérique d'une propriété peut être l'objet d'un début de travail mathématique associé à un *ETA*. En effet, comme nous l'avons rapporté dans l'analyse *a posteriori* sur le travail fait par les binômes lors de la résolution de l'énoncé 2 (Fig. 169), les élèves n'ont pas proposé de mener un travail dans un *ETA* qui aurait pu aider à émettre des conjectures sur les résultats demandés.

Enoncé 2
Soient deux suites (x_n) et (y_n) définies pour $n > 0$ par les relations :
$x_n = 1/n + 1/(n+1) + \cdots + 1/(2n)$ et
$y_n = 1/(n+1) + 1/(n+2) + \cdots + 1/(2n)$.
Démontrer que les suites (x_n) et (y_n) convergent vers une même limite. (On ne demande pas de déterminer la valeur de cette limite.)

Figure 169 (Enoncé sur l'étude de la convergence de deux suites dont on connait leurs expressions en fonction de n)

3.6.2 Cas de la phase 6 : Une preuve dichotomique du TVI

Lors de cette dernière phase sur l'ingénierie « dichotomie », nous observons que les élèves travaillent pour l'essentiel dans un *ETA*. En effet, pour l'ensemble des binômes la question semble être interprétée comme relevant pour l'essentiel du domaine de l'algorithmique. Cependant, les élèves montrent aussi qu'ils ont conscience que des savoirs mathématiques sont mis en jeu pour répondre à la question posée. Toutefois au cours de cette phase, le travail rendu par les élèves ne se situe pas pour autant dans un *ETM*. Nous pouvons penser que le fait de les amener à travailler avec des notions nouvelles qui n'ont pu être approfondies en amont car ne relevant pas du programme officiel, a favorisé cette absence de « théorisation » de la part d'élèves qui pourtant savent que pour prouver qu'une affirmation est toujours vraie, ils ne peuvent se limiter à l'étude d'un cas particulier. Nous voyons que seul un élève fait référence au raisonnement par récurrence qui pourrait être nécessaire pour faire une « preuve générale » et par cela montre que des interactions peuvent ici aussi se mettre en

place entre *ETA* et *ETM*. Toutefois l'absence de retours de travaux diversifiés sur cette phase 6 ne nous permet pas d'aller plus loin dans une analyse sur des interactions probables entre *ETA* et *ETM*.

4. Conclusion du chapitre 5 – Synthèse de l'ingénierie complète « *Algorithme de dichotomie* »

Dans le cas de la « dichotomie discrète », l'objectif d'amener les élèves à s'approprier le problème pour identifier un nombre « entier secret » dans un intervalle donné, par des essais successifs donnant lieu à un diagnostic de comparaison de l'essai et du nombre « entier secret », aide les élèves à prendre conscience que la *méthode de dichotomie* peut être une stratégie « gagnante » et « rapide » répondant à ce problème. Cette hypothèse sur la dichotomie semble rapidement perçue par les élèves comme nous avons pu le constater lors de l'expérimentation. La consigne donnée aux élèves dans le processus du jeu, que le nombre entier « secret » doit être découvert « *en faisant le minimum de propositions* » est interprété par une majorité des élèves comme un objectif de performance comparée. Celle-ci ne s'exprime pas pour autant chez les élèves en termes d'optimalité, d'autant plus que ces derniers étant des débutants tant sur le plan formalisation mathématique qu'utilisation d'outils de programmation, ils ne cherchent pas à interpréter cela en termes de vitesse de convergence. Nous pouvons observer que les élèves conçoivent rapidement la pertinence de l'utilisation d'outils informatiques, même si dans un premier temps, ceci se traduit par une demande de leur part d'utiliser un tableur, artefact dont ils ont la culture tout au long des années de collège, tant en cours de mathématiques qu'en cours de technologie. Cependant, cet artefact ne faisant pas partie des choix autorisés par le chercheur, nous pouvons penser que la possibilité d'utiliser des algorithmes se fait ainsi jour dans l'esprit des élèves, afin d'aborder intuitivement le problème d'optimalité de la stratégie « rapide » conjecturée. En effet, la transposition du travail, fait à l'aide du « papier-crayon » lors du jeu en binôme, à un algorithme associé à cette stratégie « rapide », nous permet, comme l'avions supposé, d'observer que la programmation d'un algorithme peut lors de la sous-ingénierie « dichotomie discrète », aider les élèves à améliorer leurs capacités de mener des raisonnements abstraits dans le domaine des mathématiques mais aussi de changer leur conception des variables informatiques. Le fait, de distinguer deux algorithmes « discrets »,

l'un concernant le rôle du joueur A, c'est-à-dire celui qui choisit le nombre « entier secret » et qui répond aux différentes propositions du joueur B et un second traduisant les tâches du joueur B s'avère judicieux. En effet, les élèves peuvent ainsi progresser dans la découverte et la maitrise des différents nouveaux concepts qui sont introduits tant sur le plan mathématique avec les notions de *partie entière*, de *centre d'un intervalle* et de *nombre entier aléatoire*, que sur le plan algorithmique avec le choix des variables itératives et la détermination des structures algorithmiques à mettre en place. L'intérêt de ce choix vient du fait que l'algorithme du jouer B a une structure analogue à celle du joueur A, mais n'implique pas de gestion de variables de boucle[101]. Ainsi, nous séparons la difficulté pour les élèves.

De plus, en accord avec les travaux de Samurçay[102] (1985), nous pouvons nous pencher sur les différentes conceptions de la notion de variables que se font des élèves débutants. Ainsi, dans le cadre d'une analyse épistémologique et cognitive s'appuyant notamment sur des travaux de Vergnaud (1983), nous pouvons observer lors des analyses *a posteriori* des deux sous-ingénieries « dichotomie discrète » et « dichotomie continue » que chez l'élève débutant le choix de l'amener à travailler la notion de variable comme objet d'étude, peut être pertinent par son caractère central en programmation. En particulier, l'étude des variables dans une structure de boucle plutôt que lors de simple déclaration ou d'alternative, vient du fait que dans une boucle, le concept de variable informatique apparait différent de celui de variable mathématique. Nous pouvons ainsi identifier trois opérations sur les variables dans une boucle, renvoyant à trois aspects cognitifs : *la mise à jour*, *le test d'arrêt* et l'*initialisation*. Comme le souligne Samurçay (1985), nous avons également deux types de variables relatives à des problèmes de programmation, celles qui sont des données explicites du problème et celles qui sont rendues nécessaire par le travail dans le domaine informatique et le langage syntaxique du logiciel algorithmique utilisé.

Nous pouvons aussi confirmer l'hypothèse que plus le traitement informatique des variables va s'éloigner de l'exécution « *à la main* » de l'algorithme (que ce soit la « dichotomie discrète » ou la « dichotomie continue »), plus les élèves vont être susceptibles de rencontrer des difficultés d'apprentissage et de compréhension, comme lors de la gestion d'un invariant

[101] Travaux de Samurçay et de Rogalski
[102] Psychologue cognitive française

de boucle par exemple.

Concernant la construction d'un test d'arrêt, nous pouvons observer que la variable intervenant dans l'élaboration du test peut être un objet de difficultés chez certains élèves. En effet, dans le cas de la « dichotomie discrète », bien qu'il y ait la nécessité de procéder à une égalité à un nombre fixé, les élèves doivent mettre en place une technique de calcul afin d'obtenir des nombres entiers lors des calculs des moyennes pour les différents intervalles emboités obtenus. Ceci fait l'objet d'une première difficulté mais pas directement en lien avec le test. Dans le cas de la « dichotomie continue », le fait de procéder à une inégalité à un nombre arbitraire est l'objet de certaines difficultés chez certains élèves mais moindre que ce qui est prévu lors de l'analyse *a priori*.

De même, suivant la sous-ingénierie, nous observons que lors de la construction d'un test d'arrêt, la variable intervenant dans son élaboration peut être à la fois une donnée explicite et un compteur dégressif, ou plus simplement un compteur explicite.

Nous pouvons donc, nous poser plusieurs questions concernant le perçu des élèves sur l'initialisation des variables, sur les variables d'accumulation ainsi que sur les compteurs.

Les variables d'accumulation sont moins bien traitées par les élèves que les autres. En effet, il semble que les variables avec lesquelles les élèves éprouvent le moins de difficultés sont vraisemblablement les compteurs, et en moindre mesure l'initialisation des variables, en particulier chez les élèves de Terminale, qui ont aussi dans leur boite à outils le raisonnement par récurrence. Ceci est probablement dû au fait que ce type de variable est plus institutionnalisé, comme l'indique Samurçay (1985). Lors de l'analyse de la séance, nous observons aussi qu'il en est de même pour la structure de boucle « Répéter… Jusqu'à ».

D'autre part, l'hypothèse, que les élèves de Seconde et de Première Scientifique aillent privilégier la lecture à l'affectation pour l'initialisation, est vérifiée et validée comme nous l'avons rapporté dans les analyses *a posteriori*. Comme le souligne Samurçay (1985), ce fait est probablement dû à une meilleure compréhension de l'instruction de lecture de la part des élèves débutants ainsi que par un besoin d'instruction de communication.

Pour conclure, au cours de cette ingénierie, nous observons aussi que de nombreuses

interactions entre *ETM* et *ETA* peuvent être mis en place chez les élèves, en particulier quand les compétences maitrisées par ceux-ci dans le domaine mathématique, ici l'analyse, sont importants. Le fait d'avoir procédé à une ingénierie sur les trois années de lycée : Seconde générale et classes du cycle Terminal Scientifique nous a permis ainsi de mieux apprécier ces interactions entre *ETM* et *ETA*. Cependant, l'analyse des deux dernières phases de l'ingénierie portant sur l'introduction d'un nouveau savoir que sont les suites adjacentes, mais aussi la construction d'une preuve du *théorème des valeurs intermédiaires* basée sur la dichotomie et les suites adjacentes, nous montrent que les élèves sont encore dans une approche incomplète sur les liens qui pourraient y avoir entre raisonnements de nature algorithmique et raisonnements de nature mathématique en particulier quand ils se trouvent libre dans leur choix du raisonnement.

Chapitre 6 : Une ingénierie articulant *ETA* et *ETM* spécifique : *Une politique des naissances*

Comme nous l'évoquions dans les chapitres précédents de nombreux liens entre les mathématiques et l'algorithmique sont reconnues par l'institution. Ainsi, les auteurs du programme de mathématiques de Seconde écrivent que *l'algorithmique a une place naturelle dans tous les champs des mathématiques et les problèmes posés doivent être en relation avec les autres parties du programme (fonctions, géométrie, statistiques et probabilités, logique) mais aussi avec les autres disciplines ou la vie courante* et que *la démarche algorithmique est, depuis les origines, une composante essentielle des mathématiques.* Tenant compte de ce constat, notre motivation est de terminer notre analyse sur les contributions de l'algorithmique *aux apprentissages de nouveaux concepts mathématiques chez des élèves de Seconde et de Première Scientifique dans le domaine des probabilités, afin d'étudier quels pourraient être les développements de savoirs spécifiques dans ce domaine et lors de l'élaboration d'une modèle associé à une expérience aléatoire, et de* compléter notre travail sur les articulations possibles entre *ETA* et *ETM* spécifiques.

A la différence de l'analyse et de l'arithmétique, domaines considérés dans les chapitres précédents, les probabilités dans le secondaire, associées aux statistiques, constituent un domaine où des situations extra-mathématiques jouent un rôle fondamental pour l'introduction de l'aléatoire. Les programmes récents insistent sur une approche « fréquentiste » à travers l'observation de fréquences relatives de caractères représentatifs de ces phénomènes, préparant la compréhension de la notion de probabilité théorique d'un événement, et l'observation de moyennes statistiques préparant la compréhension de la notion d'espérance d'une variable aléatoire (Parzysz, 2007, 2009 et 2011).

La simulation sur ordinateur de ces phénomènes est d'abord le moyen pratique d'obtenir des séries statistiques de taille suffisante pour observer la convergence des fréquences relatives et des moyennes statistiques. Par ailleurs, la simulation d'un phénomène aléatoire issu de situations extra-mathématiques suppose une modélisation de cette situation. Le modèle ainsi obtenu peut alors être mis à profit pour fonder le calcul de probabilités théoriques (Kiet, 2015).

Nous nous intéressons comme Kiet (2015) à la construction par les élèves de tels modèles. Nous situons pour notre part cette construction d'un modèle et son exploitation dans le cadre des *ETM* et des *ETA*. En effet, la question spécifique que nous considérons dans ce chapitre est le caractère « algorithmique » des *Espaces de Travail* intervenant dans le cycle lors de la construction d'une simulation. Nous souhaitons utiliser les ETM et les ETA comme outil d'analyse du travail tant mathématique qu'algorithmique lors des phases de *cycle de modélisation* (Blum & Leiss, 2005) qui peuvent apparaitre au cours de l'ingénierie, en particulier pour souligner la spécificité du chapitre où les élèves doivent envisager la problématique d'une situation donnée comme une approche relative à un monde « réel » plutôt qu'un monde « imaginaire ». Ainsi, certaines étapes de modélisation vont pouvoir s'avérer nécessaire. Pour cela, nous renvoyons le lecteur à la partie 1 sur le chapitre décrivant notre cadre théorique.

Lors de cette ingénierie, nous souhaitons étudier les réactions des élèves devant une situation issue du monde « réel ». Pour cela, ils doivent mener une double approche, mathématique et algorithmique, afin d'étudier les conséquences prévisibles qu'entraineraient la mise en place de cette situation dans le monde « réel ». En effet, les élèves doivent élaborer une approche « probabiliste » à travers une mathématisation de la situation et une approche « fréquentiste » à travers un travail algorithmique permettant de créer un modèle du processus aléatoire issue de cette situation afin d'émettre des conjectures sur les conséquences que pourraient avoir ce processus s'il était mis en place dans le monde « réel ».

1. Introduction

Comme pour l'ingénierie sur la dichotomie (cf. les chapitres 3, 4 et 5), nous faisons le choix d'une tâche où les connaissances mathématiques théoriques (ici la *loi géométrique tronquée*) pour interpréter une situation donnée ne sont pas connues des élèves au moment de l'expérimentation (sauf cas de redoublement). En effet, pour les élèves de Seconde, le programme indique qu'une étude théorique de cette loi n'est pas envisagée, mais qu'elle peut être approchée en utilisant des considérations algorithmiques. Quant à la Première Scientifique, bien que cette loi soit inscrite dans le programme, au moment de l'expérimentation les élèves n'en ont pas encore la connaissance. Dans la partie sur nos

motivations, nous expliquerons ce choix.

Cette ingénierie reprend une problématique autour d'une « politique des naissances » qui est présentée dans un document d'accompagnement sur l'enseignement des probabilités en Première Scientifique.

> *Afin de limiter le nombre de garçons, les dirigeants d'un pays imaginaire décident de la politique nataliste suivante :*
> - *Chaque famille a au maximum 4 enfants ;*
> - *Chaque famille arrête d'avoir des enfants après la naissance d'un garçon.*
>
> *On considère que chaque enfant a une chance sur deux d'être une fille ou un garçon et que, pour chaque couple de parents, le sexe d'un enfant est indépendant du sexe des précédents.*
> *On se demande si ce dispositif a une incidence sur la proportion de garçons dans la population.*

D'un point de vue mathématique, cette « politique de naissance » peut être modélisée par une loi de probabilité discrète : la *loi géométrique tronquée*. Elle mobilise certains savoirs issus des probabilités : arbre de probabilités pondérées, variables aléatoires discrètes, calcul d'espérances mathématiques et de proportion. Elle met en jeu aussi des savoirs issus du domaine des statistiques : fréquences, simulations, moyennes. Les simulations du processus aléatoire sont associées à une approche algorithmique de modèles mathématiques de la situation.

Plus généralement, l'enjeu de ce chapitre est l'observation et l'analyse des niveaux de compétences en probabilités, en algorithmique mais aussi en modélisation chez des élèves n'ayant pas nécessairement de connaissances particulières sur les *lois de probabilité discrète* au moment de l'expérimentation.

Les élèves se trouvent confronter à un problème issu d'une situation du monde « réel ». Ils doivent d'abord organiser et traiter les informations leur permettant de définir qu'elle serait les conséquences possibles sur l'évolution possible de la population en filles (voire en garçons) et le nombre d'enfants qu'il y aurait en moyenne par famille avec la mise en place d'une telle « politique ». Cette situation amène par conséquent les élèves à s'engager dans une démarche scientifique, à émettre des conjectures qui peuvent être en partie validées lors d'expérimentations dans un environnement numérique. Pour cela, nous faisons le choix que les élèves élaborent un modèle probabiliste permettant une approche algorithmique du phénomène aléatoire donné afin de procéder à une simulation dans un environnement

numérique pour en dégager des prévisions quant à l'évolution de la population féminine et du nombre moyen d'enfants par famille si cette « politique » est mise en place.

De plus, amener les élèves à construire des algorithmes en lien avec les *lois de probabilité discrète* est très fortement conseillé par l'institution. En effet, le programme de Seconde insiste sur le fait qu'un travail sur – l'échantillonnage ; – la réalisation d'une simulation aléatoire ; – la conception et la mise en œuvre, et l'exploitation de simulations d'une situation concrète à l'aide du tableur ou d'un logiciel algorithmique, peut-être source d'une meilleure compréhension chez l'élève sur l'exploitation et la construction d'une analyse critique d'un résultat d'échantillonnage. De plus, l'élève peut utiliser lors de la mise en place d'une simulation aléatoire, les fonctions logiques du tableur ou d'un logiciel algorithmique, en particulier la construction d'instructions conditionnelles dans un algorithme. Dans le cas de la Première Scientifique, le programme insiste sur le fait que les élèves peuvent aussi simuler une *loi géométrique tronquée* avec un algorithme lors d'une problématique autour d'une dynamique de population. Les documents d'accompagnement, tant en Seconde qu'en Première insistent aussi sur cette utilisation de l'algorithmique dans les domaines de probabilités et des simulations aléatoires (voir les paragraphes 2.1 et 2.2).

2. Approche institutionnelle

2.1 En Seconde

2.1.1 Le programme « côté probabilités et statistiques »

Dans le programme[103], nous observons que les cadres relatifs à l'enseignement des probabilités et des statistiques sont présentés séparément, mais à la suite l'un de l'autre. Cependant, comme le soulignent les concepteurs du programme, ces deux domaines mathématiques sont en relation étroite l'un avec l'autre et par conséquent doivent faire l'objet d'allers et retours. Les objectifs visés par ces deux enseignements sont en étroite relation avec des résolutions de problèmes dans le cadre de l'analyse des données. En effet, dans le cadre de l'analyse des données, cela doit aider les élèves à être capables[104] :

- *de déterminer et interpréter des résumés d'une série statistique* ;
- *de réaliser la comparaison de deux séries statistiques à l'aide d'indicateurs de position*

[103] http://eduscol.education.fr/pid26017/programmes-du-lycee.html (Consulté le 12 octobre 2017)

[104] http://cache.media.education.gouv.fr/file/30/52/3/programme_mathematiques_seconde_65523.pdf (Consulté le 12 octobre 2017)

et de dispersion, ou de la courbe des fréquences cumulées ;

Par rapport à l'enseignement des probabilités et des statistiques abordés en classe de Troisième, l'essentiel des nouveaux savoirs dans le domaine des statistiques en classe de Seconde va se porter sur une introduction aux *statistiques inférentielles*, sans que cela soit dit explicitement. Ainsi, le programme insiste sur le fait qu'un tel enseignement va permettre, dans le cadre de l'échantillonnage[105] :

- de *faire réfléchir les élèves à la conception et la mise en œuvre d'une simulation* ;
- de *sensibiliser les élèves à la fluctuation d'échantillonnage, aux notions d'intervalle de fluctuation et d'intervalle de confiance et à l'utilisation qui peut en être faite*.

Les compétences à acquérir par les élèves de Seconde sur l'échantillonnage sont celles du tableau donné ci-dessous.

Extraits du programme sur l'échantillonnage	
Contenus	**Capacités**
Notion d'échantillon. […] *Réalisation d'une simulation.*	*Concevoir, mettre en œuvre et exploiter des simulations de situations concrètes à l'aide du tableur ou d'une calculatrice.* *Exploiter et faire une analyse critique d'un résultat d'échantillonnage*.

Tableau 38

De même dans le domaine des probabilités, on attend des élèves qu'ils soient capables de[106] :

- *modéliser afin d'étudier des expériences leur permettant de s'approprier le concept d'équiprobabilité ;*
- *de proposer un modèle probabiliste à partir de l'observation de fréquences dans des situations simples ;*
- *d'interpréter des événements de manière ensembliste ;*
- *de mener à bien des calculs de probabilité.*

Les situations proposées par l'institution (documents d'accompagnement et manuels) concernent des expériences à une ou plusieurs épreuves. De même, la répétition d'expériences aléatoires doit pouvoir donner lieu à l'écriture d'algorithmes, en particulier lors de *marches aléatoires*.

Les compétences à acquérir par les élèves de Seconde sur les probabilités sont celles rappelées dans le tableau ci-dessous.

[105] Ibid.

[106] Ibid.

Extraits du programme	
Probabilités	**Capacités**
Probabilité sur un ensemble fini. *Probabilité d'un événement.* […]	*Déterminer la probabilité d'événements dans des situations d'équiprobabilité.* *Utiliser des modèles définis à partir de fréquences observées*. […]

Tableau 39

Quelques commentaires

Pour compléter les extraits donnés ci-dessus, dans les contenus sur l'échantillonnage, il est proposé la notion d'intervalle de fluctuation au seuil de 95 %, relatif aux échantillons de taille *n*. Cet intervalle peut être obtenu, de façon approchée, par simulation[107]. Pour cela, l'enseignant a la possibilité de faire percevoir expérimentalement la validité des propriétés en lien avec un tel intervalle, sachant que leurs validités ne sont pas exigibles au niveau de la Seconde.

Le programme précise qu'*un échantillon de taille n est constitué des résultats de n répétitions indépendantes de la même expérience*.

En ce qui concerne l'analyse de données, le programme indique que les élèves doivent utiliser un environnement informatique de type tableur ou logiciel algorithmique, ou encore une calculatrice scientifique afin d'étudier les différentes propriétés d'une série statistique.

À l'occasion de la mise en place d'une simulation d'une expérience aléatoire, l'élève peut[108] :

- *utiliser les fonctions logiques d'un tableur ou d'une calculatrice* ;
- *mettre en place des instructions conditionnelles dans un algorithme*.

L'apport de l'environnement informatique (tableur, algorithmes et programmation) mis en pratique au cours de cet enseignement doit permettre d'amener les élèves à un questionnement lors d'activités comme[109] :

- *l'estimation d'une proportion inconnue à partir d'un échantillon* ;
- *la prise de décision à partir d'un échantillon*.

Pour les calculs de probabilités, il est conseillé aux élèves de construire et d'interpréter des arbres pondérés de probabilités ainsi que des diagrammes ou des tableaux.

Il est à souligner qu'aucune connaissance n'est exigible sur la notion de variable aléatoire

[107] Ibid.
[108] Ibid.
[109] Ibid.

et par conséquent sur les lois discrètes au niveau de la classe de Seconde.

2.1.2 Un document d'accompagnement

Le document d'accompagnement[110] sur « *Les statistiques et les probabilités* » de Seconde propose dans le paragraphe intitulé « *Calculs de probabilités* » divers exemples d'algorithmes autour de situations aléatoires. Par exemple, Les auteurs du document écrivent :

> « *Une marche aléatoire est une trajectoire constituée de pas successifs, aléatoires et indépendants. Les applications des marches aléatoires sont multiples tant en botanique* […]*, en sciences physiques* […]*, en économie* […] *ou en sciences informatiques* […]*, etc.* »

Cette observation est accompagnée d'exemples comme celui concernant des sauts d'une puce se déplaçant sur un axe :

> *« Une puce se déplace sur un axe gradué : à chaque saut, elle se déplace d'une unité, de manière aléatoire et équiprobable vers la droite ou vers la gauche. Elle part de l'origine et effectue une marche de 30 sauts. Proposer un algorithme donnant la position d'arrivée de la puce, c'est-à-dire la position de trentième saut. Enrichir l'algorithme pour donner la liste des positions d'arrivée de N marches aléatoires. »*

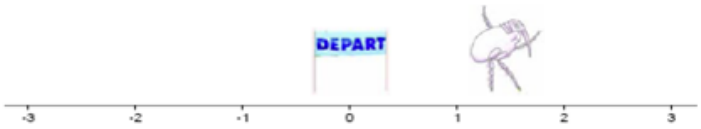


Figure 170 (Une marche aléatoire sur un axe gradué)

Ainsi, dans cet exemple nous constatons que les auteurs attendent que l'élève mette en place une démarche scientifique nécessitant la construction d'un algorithme qui requière l'emploi de la fonction RANDOM() (ou ALEA suivant l'environnement informatique choisi) pour générer un nombre aléatoire compris dans l'intervalle [0 ; 1[. De plus, l'élève doit aussi déterminer le type de la variable informatique (ici, NOMBRE) permettant de désigner la position de la puce. Ainsi, dans le cas d'un algorithme implémentable dans l'environnement numérique AlgoBox, l'élève peut proposer l'algorithme donné ci-dessous :

[110] http://cache.media.eduscol.education.fr/file/Programmes/17/9/Doc_ressource_proba-stats_109179.pdf (Consulté le 12 octobre 2017)

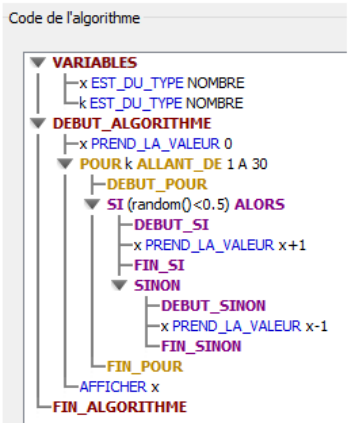


Figure 171 (Un algorithme modélisant une marche aléatoire sur un axe gradué)

Les auteurs proposent ensuite que l'élève s'interroge sur la fréquence des différentes positions d'arrivée. Pour cela, il est attendu que l'élève modifie son algorithme initial afin de simuler *N* marches aléatoires de sauts donnés (ici 30) pour représenter les résultats obtenus à l'aide d'un graphique, comme un diagramme à bâtons.

Ainsi, les travaux sur des marches aléatoires proposés aux élèves restent dans une perception intuitive, lorsque les épreuves successives sont indépendantes (au sens où la première épreuve n'a aucune influence sur la seconde, et ainsi de suite).

Le document d'accompagnement propose des situations simples à deux épreuves qui ont pu être travaillées au collège à l'aide de petits arbres pondérés. Il s'agit ainsi d'entretenir, sans aucun nouveau développement ni aucune complexification, le type de présentation et son mode opératoire, qu'est un arbre de probabilité. En Seconde toute connaissance sur le conditionnement reste hors programme.

2.1.3 Un manuel scolaire : Maths-repères (Edition 2010)

Nous faisons le choix de ne retenir qu'un seul manuel scolaire représentatif d'une majorité des manuels de ce niveau scolaire se trouvant sur le marché. Ainsi, pour notre étude dans les domaines des statistiques et des probabilités, nous choisissons le manuel *Maths-repères* (Edition 2010) chez *Hachette Education*.

Notre faisons le choix de limiter notre présentation à deux exercices se situant respectivement dans les domaines des probabilités et des statistiques et dont les thématiques ont un lien avec les objets mathématique et algorithmique qui vont être nécessaire pour la mise en place de notre ingénierie. Nous renvoyons le lecteur à la lecture du chapitre 3 de la partie 2 où une présentation rapide et plus complète du manuel est proposée.

2.1.3.1. Dans le domaine des statistiques descriptives

Dans la partie « Exercices d'approfondissement » du chapitre « Statistique descriptive. Analyse de données », les auteurs proposent par exemple un exercice intitulé *Algorithme et paramètre statistique* dont nous donnons l'énoncé ci-dessous.

27 **Algorithme et paramètre statistique**

Une série statistique est résumée dans le tableau suivant.

Valeur	x_1	x_2	…	x_p
Effectif	n_1	n_2	…	n_p

Entrées
Saisir p : entier naturel non nul
Saisir $x_1, x_2, \ldots, x_p$
Saisir $n_1, n_2, \ldots, n_p$
Initialisations
N prend la valeur 0
S prend la valeur 0
Traitement
Pour i de 1 jusqu'à p
  N prend la valeur $N + n_i$
  S prend la valeur $S + n_i x_i$
FinPour
Sortie
Afficher S/N

a) À la sortie de l'algorithme, quelles sont les valeurs des variables N et S ?
b) Quel est le rôle de cet algorithme ?

Figure 172 (Extrait d'un exercice du manuel Maths-repères de Seconde : Algorithme et paramètre statistique)

a) Présentation des tâches attendues dans l'exercice proposé

Dans cet exercice, nous observons que les auteurs proposent dans un tableau résumant une série statistique suivi d'un algorithme complet écrit en langage naturel. Les élèves doivent travailler sur le rôle de l'algorithme proposé et indiquer l'affichage obtenu après exécution de l'algorithme sans préciser s'ils doivent ou pas implémenter cet algorithme dans un environnement numérique afin de le tester sur des valeurs particulières. Nous observons aussi que le tableau donné est un tableau abstrait étant donné qu'aucune valeur numérique n'est donnée tant pour la ligne des « valeurs » que pour celle des « effectifs ».

Les élèves doivent interpréter les valeurs initiales respectives de N et S, puis comprendre le but de la boucle « Pour …Faire » qui permet de calculer la somme des effectifs n_i, pour i allant de 1 à p (où p correspond au nombre de valeurs), cette somme étant stockée dans la variable N et la somme des produits de chacune des valeurs x_i par son effectif n_i correspondant, cette somme étant stockée dans la variable S. De plus, une fois cette

compréhension du traitement est faite par les élèves, ils doivent interpréter l'affichage obtenu en fonction des variables S et N comme étant le calcul de la moyenne pondérée de la série statistique.

b) Analyse de l'exercice proposé à l'aide du cadre théorique sur les ETA/ETM

Nous souhaitons compléter l'analyse des tâches faites à la section précédente par une approche en termes de paradigmes algorithmiques et d'articulations entre *ETA/ETM* que nous présentons dans le tableau ci-dessous.

<table>
<tr><td>Thème de l'exercice</td><td colspan="2">Calcul d'une moyenne pondérée d'une série statistique</td></tr>
<tr><td>Paradigmes</td><td colspan="2">Le questionnement sur les variables N et S se situe au niveau 2 de l'algorithmique. En effet, le fait que les élèves doivent travailler dans un cadre général, c'est-à-dire sans avoir de valeurs numériques précises tant pour la ligne des « valeurs » que pour celles des « effectifs » correspondants, amènent ces derniers à mener une étude au niveau d'une axiomatique naturelle de l'algorithme.
En revanche, le questionnement sur l'affichage obtenu après exécution de l'algorithme se situe au niveau 1 de l'algorithmique. En effet, le résultat obtenu renvoie l'élève à interpréter ce résultat comme relevant de la définition vue en cours sur la moyenne pondérée d'une série statistique.</td></tr>
<tr><td rowspan="2">Espaces de Travail</td><td>ETA</td><td>ETM</td></tr>
<tr><td>La question portant sur les valeurs des variables N et S demande que l'élève exécute manuellement l'algorithme. Cependant, celui-ci peut proposer d'implémenter l'algorithme donné dans un environnement numérique afin de le teste. Toutefois, l'implémentation peut s'avérer difficile car faisant intervenir deux listes de nombres comme type de variable. Or ce type de variable n'étant pas institutionnalisée au niveau de la Seconde, il se peut alors que les élèves ne connaissent cette possibilité que peut offrir un environnement numérique comme AlgoBox par exemple.
(Genèse instrumentale)

En conséquence, l'élève va probablement voir la question comme une application papier-crayon de l'algorithme.
(Genèses sémiotique et instrumentale)

La question portant sur l'affichage S/N demande à l'élève de faire le lien entre le résultat obtenu et une définition nouvelle vue dans le cours sur les statistiques descriptives. La difficulté algorithmique va</td><td>L'essentiel du travail va consister à mettre en place des connaissances anciennes sur le calcul de produits et de sommes, ainsi que celui d'un quotient. Le tout relève donc de l'algèbre élémentaire, c'est-à-dire un ETM d'algèbre.
(Genèse discursive)

De plus, les élèves vont devoir se référer à une formule de calcul de la moyenne pondérée d'une série statistique, ce qui va les renvoyer à un ETM de statistiques.
(Genèse discursive)</td></tr>
</table>

	consister ici à comprendre que ce résultat est obtenu une fois que l'on est sorti de la boucle « Pour ». (Genèses instrumentales et discursives)	

Tableau 40

2.1.3.2. Dans le domaine des probabilités

Dans la partie « Problèmes ouverts » du chapitre « Probabilités et problèmes », les auteurs proposent par exemple un exercice intitulé *Le lièvre et la tortue* (Fig. 173).

Figure 173 (Extrait d'un exercice du manuel Maths-repères de 2nde : Le lièvre et la tortue)

a) Présentation des tâches attendues dans l'exercice proposé

Dans cet exercice, nous observons que les auteurs donnent les contraintes d'une situation aléatoire portant sur le déplacement de deux animaux en fonction des résultats obtenus lors du lancer d'un dé à six et équilibré. Cependant, ils ne proposent aucune indication particulière en ce qui concerne la démarche à suivre pour comprendre le sens de cette situation aléatoire, seul un dessin est proposé afin d'aider l'élève à interpréter le type de « déplacement » de chacun des animaux suivant le résultat du lancer du dé.

Nous observons aussi qu'il est demandé aux élèves de construire un algorithme pour simuler l'expérience sans qu'il soit précisé s'ils doivent ou pas l'implémenter dans un environnement numérique afin de le tester sur un nombre fini de lancers.

D'un point de vue algorithmique, les élèves ont pour tâches à mettre en place. En premier, ils doivent déterminer les variables de type NOMBRE qui vont être en jeu : résultat d'un lancer de dé – numéro de la case où se situe la tortue (sachant qu'il y a six cases qui peuvent être numérotées de 1 à 6). Puis, ils doivent déterminer les valeurs correspondantes aux affectations initiales de ces deux variables. Ensuite, ils doivent construire une boucle « TantQue » avec une double condition de sortie de boucle, portant sur le numéro de la case où se situe la tortue et le numéro du dé obtenu lors d'un lancer du dé. Cette boucle doit aussi

faire intervenir une instruction conditionnelle du type « Si… Alors » permettant de connaitre le numéro de la case où se trouve la tortue après un lancer du dé. De plus, une fois sortie de la boucle « TantQue », l'algorithme doit afficher le vainqueur et pour cela une deuxième instruction conditionnelle du type « Si…Alors…Sinon » doit être implémenter (Fig. 174).

```
VARIABLES
    Face_obtenue_du_dé EST_DU_TYPE NOMBRE
    case_où_passe_tortue EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
    AFFICHER "Nous avons six cases numérotées de 1 à 6. Le premier animal qui arrive à la case 6 a gagné."
    AFFICHER " "
    Face_obtenue_du_dé PREND_LA_VALEUR 0
    case_où_passe_tortue PREND_LA_VALEUR 0
    TANT_QUE (Face_obtenue_du_dé<6 ET case_où_passe_tortue<6) FAIRE
        DEBUT_TANT_QUE
        Face_obtenue_du_dé PREND_LA_VALEUR ALGOBOX_ALEA_ENT(1,6)
        SI (Face_obtenue_du_dé<6) ALORS
            DEBUT_SI
            case_où_passe_tortue PREND_LA_VALEUR case_où_passe_tortue+1
            AFFICHER "La tortue se trouve sur la case : "
            AFFICHER case_où_passe_tortue
            FIN_SI
        FIN_TANT_QUE
    SI (case_où_passe_tortue==6) ALORS
        DEBUT_SI
        AFFICHER "La tortue a gagné."
        FIN_SI
        SINON
            DEBUT_SINON
            AFFICHER "Le lièvre se trouve sur la case 6. Il a gagné."
            FIN_SINON
FIN_ALGORITHME
```

Figure 174 (Algorithme permettant de simuler une partie du « lièvre et la tortue »)

Nous observons que les auteurs n'attendent pas que l'élève détermine la probabilité de gagner pour le lièvre (ou pour la tortue) lors d'une partie. De même, une extension possible sur le calcul et la représentation d'un intervalle de fluctuation n'est pas envisagée par les auteurs de l'exercice. En effet, les savoirs mathématiques des élèves au niveau de la classe de Seconde ne leur permettraient pas de répondre de façon formelle à ces questions.

De plus dans le cas des parties répétées, si l'on se cantonne à un travail purement algorithmique, les élèves vont être confrontés aux limitations de l'environnement *AlgoBox* qui ne permet pas de découper l'algorithme en fonctions ou procédures réutilisables (cf. le chapitre sur l'*Algorithme de Kaprekar*).

b) Analyse de l'exercice proposé à l'aide du cadre théorique sur les ETA/ETM

Comme pour l'exercice précédent, nous souhaitons compléter l'analyse des tâches faites à la section précédente par une approche en termes de paradigmes algorithmiques et d'articulations entre *ETA/ETM* que nous présentons ci-dessous.

<table>
<tr><td>Thème de l'exercice</td><td colspan="2">Le jeu du lièvre et de la tortue</td></tr>
<tr><td>Paradigmes</td><td colspan="2">Dans un premier temps, le problème posé aux élèves consiste à émettre une hypothèse quant aux « chances » de gagner la plus favorable pour l'un des deux animaux. Cette première approche peut être synonyme d'une première tentative de modélisation voire de justification. Ici, l'élève se situe au niveau 1 de l'algorithmique. Il a une approche intuitive de l'algorithme qui va être issu d'une situation imaginaire. Les élèves vont entrer dans le problème par des manipulations « papier-crayon ». Cette phase située au niveau de l'algorithmique va leur permettre de modéliser le problème.
La construction de l'algorithme demandée dans la deuxième question pour simuler l'expérience va leur permettre de mettre en place l'idée qu'il faut faire de nombreuses parties pour avoir une idée de la réponse. Lors de ce travail de construction, l'élève reste encore à la « frontière » des niveaux 1 et 2 de l'algorithmique.</td></tr>
<tr><td rowspan="2">Espaces de Travail</td><td>ETA</td><td>ETM</td></tr>
<tr><td>Comme nous l'avons décrit plus haut, le choix des variables à mettre en place va être l'occasion pour l'élève de mener un travail de réflexion dans un ETA. En effet, devra-t-il travailler sur les « avancées » de la tortue au celles du lièvre ?
(Genèses instrumentale et discursive)

Le travail sur le choix et la gestion des structures de boucles et des instructions conditionnelles lors de la construction de l'algorithme en langage naturel (ou pseudo-code) vont permettre à l'élève de mieux saisir l'aspect aléatoire de la situation, en particulier avec la dépendance de l'avancée de l'animal en fonction du résultat obtenu lors de chaque lancer du dé.
(Genèse instrumentale et discursive)

La construction d'un nombre aléatoire compris entre 1 et 6 permettant de simuler les lancers du dé vont devoir prendre en compte de la syntaxe du langage algorithmique si l'élève souhaite implémenter son algorithme dans un environnement numérique.
(Genèses instrumentale et sémiotique)</td><td>Le travail mathématique que doit mener l'élève peut être vu d'un point modélisation du problème.
(Genèses discursive et sémiotique)

L'élève doit aussi déterminer le choix de l'animal à étudier pour simuler le jeu.
(Genèse discursive)

Cet aspect du problème demande de prendre en compte que l'avancée de la tortue correspond à une variable aléatoire associée au numéro de la case où se trouve la tortue à la fin de la partie. Mais à ce niveau scolaire, l'élève ne peut avoir connaissance de ce fait.</td></tr>
</table>

Tableau 41

2.2 En Première Scientifique[111]

2.2.1 Les programmes « côté probabilités et statistiques »

Le travail fourni par les élèves en classe de Seconde, sur les probabilités et les statistiques se poursuit et s'approfondit tout au long du cycle Terminal Scientifique avec la mise en place de nouveaux outils dans l'analyse de données, ainsi que dans l'étude de lois de probabilités (discrètes en Première et continues en Terminale). L'objectif est de faire réfléchir les élèves sur des données réelles, riches et variées.

> *La notion de loi de probabilité d'une variable aléatoire permet de modéliser des situations aléatoires, d'en proposer un traitement probabiliste et de justifier certains faits observés expérimentalement en classe de seconde*[112].

L'utilisation d'arbres pondérés est développée pour modéliser la répétition d'expériences identiques et indépendantes. Cependant, cette utilisation *est restreinte à ce cadre afin d'éviter toute confusion avec des situations relevant des probabilités conditionnelles*[113]. Ce dernier point est étudié seulement en Terminale.

> *Dans le cas particulier d'expériences identiques et indépendantes à deux issues, on introduit la loi binomiale. En s'appuyant sur cette loi, on poursuit la formation des élèves dans le domaine de l'échantillonnage*[114].

Extraits du programme des probabilités	Capacités attendues
• Variable aléatoire discrète et loi de probabilité. Espérance, variance et écart-type. • Modèle de la répétition d'expériences identiques et indépendantes à deux ou trois issues. • Epreuve de Bernoulli, loi de Bernoulli. • Schéma de Bernoulli, loi binomiale. • Espérance, variance et écart-type de la loi binomiale.	*Déterminer et exploiter la loi d'une variable aléatoire*. […] Représenter la répétition d'expériences identiques et indépendantes par un arbre pondéré. Utiliser cette représentation pour déterminer la loi d'une variable aléatoire associée à une telle situation. Reconnaître des situations relevant de la loi binomiale. Calculer une probabilité dans le cadre de la loi binomiale. […] Utiliser l'espérance d'une loi binomiale dans des contextes variés.

Tableau 42

[111] B.O. n° 9 du 30 septembre 2010
[112] Ibid.
[113] Ibid.
[114] Ibid.

Extraits du programme sur l'échantillonnage	Capacités attendues
• Utilisation de la loi binomiale pour une prise de décision à partir d'une fréquence	[…]

Tableau 43

Quelques commentaires

Le programme de Première Scientifique donne les commentaires suivants :

- *à l'aide de simulations et d'une approche heuristique de la loi des grands nombres*, on peut faire *le lien avec la moyenne et la variance d'une série de données* ;
- *on doit exploiter les fonctionnalités de la calculatrice ou d'un logiciel* (algorithmique ou tableur) *pour déterminer l'espérance, la variance et l'écart-type d'une variable aléatoire* ;

Les élèves apprennent à ce niveau de leur scolarité que lorsque l'on a la répétition d'expériences identiques et indépendantes, alors *la probabilité d'une liste de résultats est le produit des probabilités de chaque résultat*.

Pour comprendre et assimiler cette propriété sur le calcul d'une probabilité d'une liste de résultats, le programme insiste sur une représentation graphique définie sous forme d'un arbre pondéré de probabilité, qui doit permettre ainsi d'installer une représentation mentale efficace chez le jeune élève.

La loi géométrique tronquée

Les auteurs du programme signalent aussi que les élèves doivent traiter quelques expériences aléatoires autour de la loi géométrique tronquée (Tableau 44). Ils précisent aussi que cette loi peut être simulée à l'aide d'un algorithme.

Modèle de la répétition d'expériences identiques et indépendantes à deux ou trois issues.	• Représenter la répétition d'expériences identiques et indépendantes par un arbre pondéré. • Utiliser cette représentation pour déterminer la loi d'une variable aléatoire associée à une telle situation.	Pour la répétition d'expériences identiques et indépendantes, la probabilité d'une liste de résultats est le produit des probabilités de chaque résultat. La notion de probabilité conditionnelle est hors programme. On peut aussi traiter quelques situations autour de la loi géométrique tronquée. ◇ On peut simuler la loi géométrique tronquée avec un algorithme.

Tableau 44

De même, dans le document d'accompagnement de Première, les auteurs écrivent :

> *Les situations de répétition d'une même expérience aléatoire, reproduite dans des conditions identiques constituent un élément fort du programme de Première.*
> *L'introduction de la loi géométrique tronquée présente de nombreux avantages :*
> - *travailler des répétitions d'une expérience de Bernoulli ;*
> - *envisager ces répétitions sous l'angle algorithmique ;*
> - *présenter une situation d'arbre pour lequel tous les chemins n'ont pas la même longueur ;*
> - *exploiter hors de l'analyse les propriétés des suites géométriques ;*
> - *exploiter hors du cadre habituel des résultats relatifs à la dérivation ;*
> - *travailler les variables aléatoires.*

Ils ajoutent que d'un point de vue algorithmique, *le processus lié à la loi géométrique tronquée est aisé à mettre en œuvre avec un algorithme. Il suffit de remarquer que l'instruction* ***ent(NbrAléat + p)*** *génère un nombre aléatoire entier qui vaut 1 avec la probabilité p, et 0 avec la probabilité 1 – p*.

Toujours selon ces auteurs, dans le domaine de l'analyse, *au niveau de la classe de Première, la détermination de l'espérance de la loi géométrique tronquée de paramètres n et p mobilise à la fois les suites géométriques et la dérivation*.

2.2.2 Présentation de quelques définitions clés sur certaines notions de probabilités vues en Première

2.2.2.1. Variable aléatoire discrète

On considère un ensemble fini U et une loi de probabilité *p* sur U. Une variable aléatoire X sur U est une fonction définie sur U à valeurs dans R. Si x_1, x_2, …, x_r désignent les valeurs prises par la variable aléatoire X, alors on note « X = x_i » l'événement « X prend la valeur x_i ».

2.2.2.2. Loi de probabilité

On définit une loi de probabilité associée à la variable aléatoire X, par le donnée des réels x_i et des probabilités p_i = P(X = x_i) avec $i \in [\![1 ; r]\!]$. On la présente souvent sous la forme d'un tableau.

x_i	x_1	x_2	…	x_r
p_i = P(X = x_i)	p_1	p_2		p_r

2.2.2.3. Espérance, variance, écart type

Soit X une variable aléatoire de loi de probabilité (x_i ; p_i), où $i \in [\![1 ; r]\!]$. On rappelle les définitions suivantes telles qu'elles sont présentées en classe de Première Scientifique.

- L'espérance de X est le nombre noté E(X), tel que : $E(X) = p_1 x_1 + p_2 x_2 + \ldots + p_r x_r$.
- La variance de X est le nombre noté V(X), tel que $V(X) = p_1 (x_1 - E(X))^2 + p_2 (x_2 - E(X))^2 + \ldots + p_r (x_r - E(X))^2$.
- L'écart type de X est le nombre noté $\sigma(X)$ tel que $\sigma(X) = \sqrt{V(X)}$.

2.2.2.4. Représentation par un arbre pondéré d'une répétition d'expériences aléatoires identiques et indépendantes

Dans le cas d'une répétition d'expériences identiques et indépendantes, l'institution propose de choisir comme modèle de représentation de cette répétition un arbre pondéré où sur chaque branche est indiqué la probabilité de l'issue correspondante. Ainsi, l'élève apprend que dans le cas d'une répétition d'expériences identiques et indépendantes, représentée par un arbre pondéré :

- La probabilité d'un événement correspondant à un chemin sur l'arbre est obtenue en multipliant les probabilités portées par ses branches.
- La probabilité d'un événement correspondant à plusieurs chemins est alors obtenue en ajoutant les probabilités des événements correspondants à chaque chemin, puisque ceux-ci sont incompatibles.

2.2.2.5. La loi géométrique tronquée

Dans le cadre institutionnel, les élèves doivent connaitre les définitions d'une épreuve de Bernoulli, de la loi de Bernoulli, du schéma de Bernoulli, ainsi que la propriété d'une loi binomiale.

Le document d'accompagnement *Ressources pour la classe de première générale et technologique* (juin 2011) propose entre autres comme complément d'activités différentes tâches en lien avec la *loi géométrique tronquée*. Les auteurs du document signalent qu'une situation de répétition d'une expérience aléatoire, reproduite dans des conditions identiques, constitue un élément fort du programme. En effet, ils supposent qu'une introduction de la loi géométrique tronquée peut présenter plusieurs avantages, comme :

- *travailler des répétitions d'une expérience de Bernoulli ;*
- *envisager ces répétions sous un angle algorithmique ;*
- *présenter une situation d'arbre pondéré de probabilités où les chemins n'ont pas tous*

la même longueur ;
- *exploiter hors de l'analyse des propriétés issues de d'autres champs des mathématiques : les suites géométriques, la dérivation ;*
- *travailler avec des variables aléatoires discrètes.*

De nombreux exemples sont ainsi proposés, dont celui que nous allons étudier lors de notre ingénierie didactique sur deux niveaux scolaires : Seconde et Première Scientifique (cf. section 3), où une *politique des naissances* (C. Robert, 2000) est envisagée.

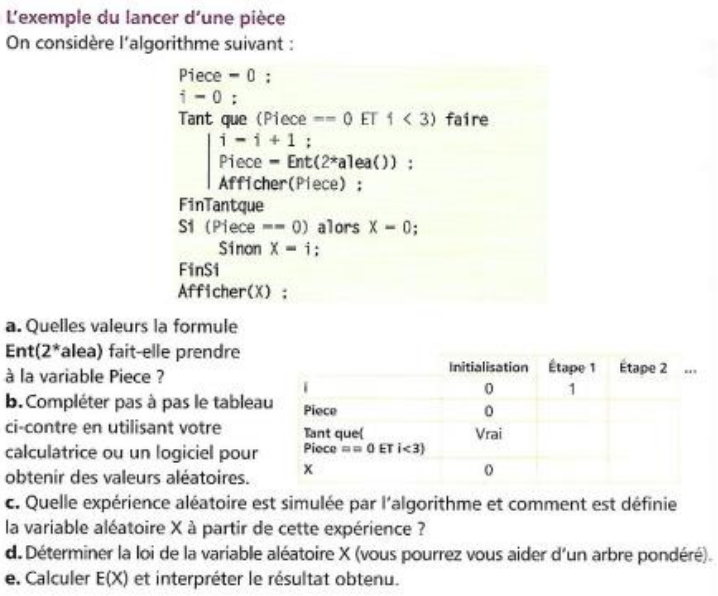
> *Pour limiter le nombre de filles dans un pays (imaginaire ?), on décide que :*
> - *Chaque famille aura au maximum 4 enfants ;*
> - *Chaque famille arrêtera de procréer après la naissance d'un garçon.*
>
> *On considère que chaque enfant a une chance sur deux d'être un garçon ou une fille et que, pour chaque couple de parents, le sexe d'un enfant est indépendant du sexe des précédents.*

Dans cet exemple, nous pensons qu'il est attendu que les élèves se posent la question suivante, *ce choix a-t-il la conséquence attendue, à savoir de diminuer le nombre de filles dans la population ?* puis qu'ils argument leurs réponses en proposant un modèle de cette situation et qu'ils l'expérimentent dans un environnement informatique après avoir construit un ou plusieurs algorithmes correspondants à ce modèle.

2.2.3 Un manuel scolaire : Symbole (Edition 2011)

Comme pour la Seconde, nous faisons le choix de ne retenir qu'un seul manuel scolaire, le manuel *Symbole* chez *Belin* dont nous fait une présentation rapide dans le chapitre 3 de la partie 2 (cf. page 226).

Nous faisons le choix de ne présenter qu'un seul exercice du manuel où un travail de nature algorithmique est proposé dans le cadre d'une introduction à la loi géométrique tronquée. Ainsi, dans le chapitre *Loi binomiale*, nous trouvons de nombreux exercices mettant en place divers travaux autour de l'étude de lois discrètes avec une approche algorithmique. Parmi ces exercices, nous en trouvons un qui introduit la loi géométrique tronquée. Cet exercice (Fig. 175) se trouve dans la partie Travaux pratiques du manuel.



L'exemple du lancer d'une pièce

On considère l'algorithme suivant :

```
Piece ← 0 ;
i ← 0 ;
Tant que (Piece == 0 ET i < 3) faire
    i ← i + 1 ;
    Piece ← Ent(2*alea()) ;
    Afficher(Piece) ;
FinTantque
Si (Piece == 0) alors X ← 0;
    Sinon X ← i;
FinSi
Afficher(X) ;
```

a. Quelles valeurs la formule **Ent(2*alea)** fait-elle prendre à la variable Piece ?

b. Compléter pas à pas le tableau ci-contre en utilisant votre calculatrice ou un logiciel pour obtenir des valeurs aléatoires.

	Initialisation	Étape 1	Étape 2	...
i	0	1		
Piece	0			
Tant que(Piece == 0 ET i<3)	Vrai			
X	0			

c. Quelle expérience aléatoire est simulée par l'algorithme et comment est définie la variable aléatoire X à partir de cette expérience ?

d. Déterminer la loi de la variable aléatoire X (vous pourrez vous aider d'un arbre pondéré).

e. Calculer E(X) et interpréter le résultat obtenu.

Figure 175 (Extrait d'un exercice du manuel Symbole en Première Scientifique : Exemple d'un lancer d'une pièce)

2.2.3.1. Présentation des tâches attendues dans l'extrait de l'exercice

Dans cet extrait, les élèves ont un algorithme associé à une situation aléatoire particulière. Ils doivent à l'aide de leur calculatrice ou d'un logiciel algorithmique compléter le tableau des valeurs prises par une variable aléatoire X. Les auteurs attendent que les élèves donnent la loi de probabilité de la variable aléatoire afin de calculer l'espérance mathématique de la variable X, puis qu'ils interprètent le résultat obtenu.

Pour cela, les élèves doivent d'abord expliquer le rôle d'une formule mathématique associée aux deux côtés d'une pièce de monnaie. Il ne leur est pas précisé la façon d'obtenir les valeurs de cette formule.

Les élèves doivent aussi interpréter une double condition de sortie sur une boucle « TantQue ». Les auteurs donnent un tableau à compléter afin de déterminer les différentes valeurs d'une variable aléatoire.

Les compétences informatiques mises en jeu dans cet extrait d'exercice se situent dans une exploitation des structures classiques que les élèves ont l'habitude d'utiliser depuis la Seconde. Les compétences mathématiques nécessaires pour la résolution de l'exercice sont anciennes pour l'aspect nombre aléatoire, en revanche nouvelles pour l'utilisation de la fonction partie entière, même si les élèves de ce niveau scolaire sont sensés connaitre le concept de partie entière d'un nombre réel. De plus, dans le domaine des probabilités, les élèves doivent savoir définir une variable aléatoire discrète et déterminer la loi de probabilité

qui lui est associée, ainsi que calculer l'espérance mathématique d'une variable aléatoire. Ces compétences dans le domaine des probabilités sont nouvelles pour des élèves de ce niveau scolaire.

2.2.3.2. Analyse de l'exercice à l'aide du cadre théorique sur les *ETA/ETM*

Pour compléter l'analyse des tâches faites à la section précédente, nous souhaitons poursuivre cette analyse par une approche en termes de paradigmes algorithmiques et d'articulations entre *ETA/ETM* que nous présentons dans le tableau ci-dessous.

<table>
<tr><td>Thème de l'exercice</td><td colspan="2">Etude d'une expérience suivant une loi géométrique tronquée</td></tr>
<tr><td>Paradigmes</td><td colspan="2">Cet exercice se situe au niveau 1 de l'algorithmique. En effet, il n'est pas demandé aux élèves de valider l'algorithme ni de justifier les résultats obtenus pour la loi de probabilité à l'aide de considération algorithmique.
En revanche, le fait que pour la détermination de cette loi, les auteurs proposent de construire un arbre pondéré, nous montre que les élèves peuvent aborder le travail mathématique attendu dans cet exercice au niveau 2 d'un ETM spécifique à l'analyse puisque l'arbre est considéré au lycée comme étant une justification « formelle » pour la détermination d'une loi de probabilité.</td></tr>
<tr><td rowspan="2">Espaces de Travail</td><td>ETA</td><td>ETM</td></tr>
<tr><td>Les élèves doivent avoir connaissance du rôle des fonctions ENT et ALEA.
(Genèses instrumentale et sémiotique)

Les élèves doivent comprendre le rôle de chacune des trois variables de type NOMBRE que sont Pièce, i et X. En particulier, ils doivent comprendre le modèle choisi pour simuler les lancers d'une pièce et les résultats en découlant pour une variable aléatoire dont ils doivent interpréter sa signification.
(Genèses instrumentale et discursive)

Les élèves doivent comprendre la double condition de sortie se trouvant dans la boucle « TantQue ».
(Genèses instrumentale, sémiotique et discursive)

Les élèves doivent interpréter les conditions initiales que sont « Pièce ←0 » et « i ←0 ».
(Genèses sémiotique et discursive)

Pour finir, les élèves doivent comprendre la signification de la structure de boucle « TantQue » et de l'instruction</td><td>Les élèves interprètent le modèle représenté par un algorithme permettant de simuler une expérience aléatoire.
(Genèse instrumentale et discursive)

Les élèves ont à connaitre ou à déterminer certaines définitions et propriétés mathématiques autour de la fonction partie entière et des nombres aléatoires.
(Genèses instrumentale et discursive)

Les élèves doivent déterminer à l'aide d'un algorithme donné et d'une représentation sous forme d'un tableau les valeurs d'une variable aléatoire X.
(Genèses instrumentales et discursives)

Les élèves doivent aussi interpréter la signification en termes de logique du mot « ET » dans la condition de sortie de la boucle « TantQue ».
(Genèses sémiotique et discursive)

Les élèves peuvent construire un arbre pondéré de probabilité pour déterminer la loi de probabilités associée à la variable aléatoire X puis en déduire l'espérance</td></tr>
</table>

	conditionnelle qui sont mises en place dans l'algorithme donné (Genèses instrumentale et sémiotique)	mathématique de cette variable. (Genèses instrumentale, sémiotique et discursive) Pour conclure, les élèves doivent interpréter en termes probabilistes la valeur de l'espérance mathématique obtenue. (Genèse discursive)

Tableau 45

2.3 Conclusion

Nous venons d'observer à travers cette étude, des attentes institutionnelles dans le domaine des probabilités, en particulier lors de l'introduction de certaines lois discrètes qu'une approche algorithmique peut aider à donner du sens à de nouvelles compétences mathématiques chez des élèves de Seconde et de Première Scientifique, et qu'en tant que chercheur, une analyse des tâches demandées sur une loi discrète, ici la loi géométrique tronquée peut nous permettre de mieux entrevoir les articulations entre *ETA* et $ETM_{probabilité}$ idoines.

Pour cela, nous souhaitons compléter cette première approche en organisant une ingénierie didactique dans le domaine des probabilités/statistiques avec des élèves ayant déjà acquis une certaine maitrise de l'outil informatique en lien avec l'algorithmique, comme l'organisation des variables algorithmiques, les structures de boucles, … afin d'étudier les conséquences que cela peuvent avoir sur l'apprentissage de nouvelles compétences mathématiques dans le domaine des probabilités enseignées dans le secondaire second cycle.

3. Une ingénierie didactique dans le champ des probabilités

3.1 Nos motivations

Comme nous l'évoquions précédemment, de nombreux liens entre les mathématiques et l'algorithmique sont reconnues par l'institution, en particulier dans le domaine des probabilités et des statistiques. Pour cela, nous choisissons une situation aléatoire ayant pu être issu du monde « réel » afin d'étudier divers objets mathématiques que sont la simulation d'un phénomène aléatoire et la modélisation dans le cadre des programmes de Seconde et de Première Scientifique.

L'intérêt de faire construire par des élèves des simulations de situations aléatoires dans le cadre d'un enseignement des probabilités et des statistiques est souligné par certains chercheurs en didactique des mathématiques (Kiet, 2015). Nous nous intéressons à ces aspects tout en nous situant dans un cadre plus général des niveaux d'acquisition des structures algorithmiques dans un domaine précis des mathématiques (Nijimbere, 2015).

Nous souhaitons mener un travail conjoint en probabilités et en algorithmique autour d'une simulation d'une situation aléatoire qui va s'effectuer dans le cadre d'une activité de modélisation.

3.1.1 L'aspect modélisation d'un processus aléatoire

Comme nous le signalions cette ingénierie à cette particularité qu'elle s'effectue dans le cadre d'une activité de modélisation. Ainsi, pour aider à une meilleure compréhension de l'analyse de notre ingénierie, nous souhaitons rappeler quelques points didactiques sur la notion de modèle mathématique ainsi que sa place dans une approche conjointe en mathématique et en algorithmique d'une situation aléatoire en lien avec le monde « réel ».

3.1.2 Pourquoi utiliser un modèle mathématique ?

Nous rappelons qu'un *modèle mathématique* est fondé sur des observations et des hypothèses. Il est constitué d'objets mathématiques, contenant essentiellement des *variables* concernant le phénomène observable, dépendant de *paramètres* fixes, généralement inconnus.

Lors de cette ingénierie, conformément aux directives formulées par l'institution, les élèves doivent *distinguer ce qui est empirique (du domaine de l'expérience) de ce qui est théorique*. La modélisation occupe une part importante des activités scientifiques (par exemple en *Sciences de l'Ingénieur*) pour comprendre ou prévoir. Les mathématiques fournissent un langage de modélisation. C'est au travers de modèles mathématiques que l'élève affine sa compréhension du monde. Selon Bouleau[115], *la modélisation […] tente de s'habiller des habits de la science, demande à être critiquée […] d'où l'importance de l'introduction de la modélisation dans l'enseignement*. Dans le cas de la modélisation d'une expérience aléatoire, l'élève doit lui associer une loi de probabilité, comme le propose le document

[115] Commission Kahane (Mars 2001)

d'accompagnement du programme de Première.

Pour une tâche de simulation d'un processus aléatoire, la démarche choisie peut être décrite par le graphique ci-dessous (Fig. 176) :

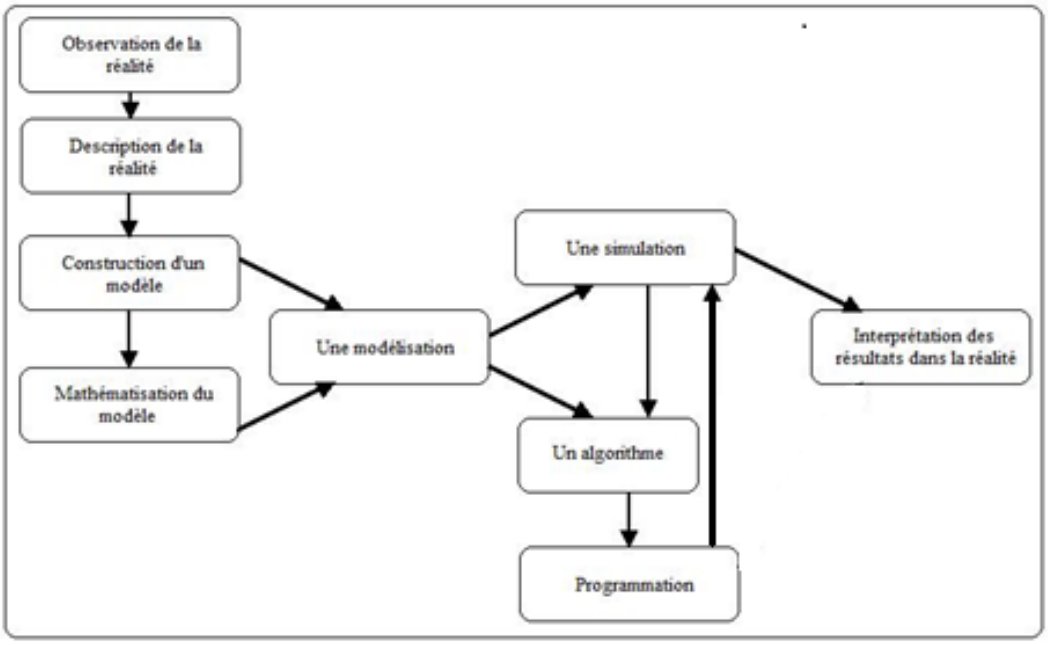


Figure 176 (Représentation graphique de la démarche à mettre en place lors d'un travail sur le simulation d'un processus aléatoire)

Une des premières difficultés que l'élève doit surmonter lors d'une tâche de simulation d'un processus aléatoire, est de construire les étapes intermédiaires entre l'observation de la réalité et la construction élaborée du modèle mathématique. L'étape suivante est alors le passage à la simulation elle-même, puis à l'écriture d'un (ou plusieurs) algorithme(s) en langage naturel du modèle obtenu, ainsi que la simulation définie. Ensuite, l'élève se confronte à une autre difficulté d'ordre syntaxique, lors du passage en langage pseudo-code de l'algorithme pour une utilisation dans un environnement informatique. L'élève doit aussi savoir interpréter les résultats obtenus à l'aide du modèle mathématique, pour en tirer des conjectures dans le cadre de la réalité. On peut faire l'observation suivante que réaliser « *l'expérience, utiliser un modèle mathématique, simuler l'expérience* » (Bordier, 1991), avec l'aide entre autres de l'algorithmique et des outils informatiques, sont des « *approches possibles d'un même problème* » (Bordier, 1991). Nous pouvons interpréter cela à l'aide du graphique présenté ci-dessous (Fig. 177).

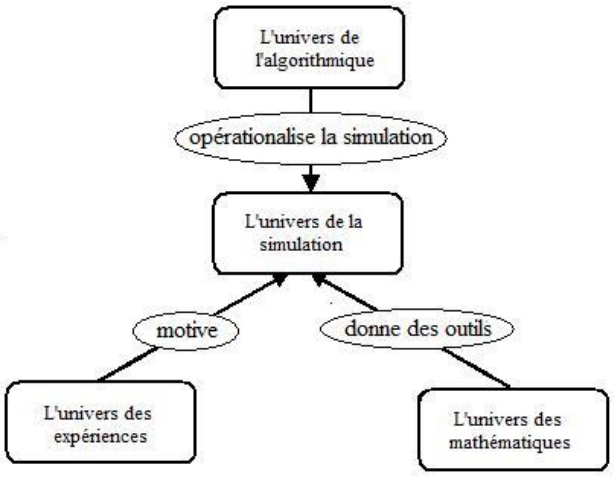


Figure 177 (Les objectifs des univers des expériences, de l'algorithmique et des mathématiques en direction de l'univers de la simulation)

3.1.3 Pourquoi choisir une simulation d'une expérience aléatoire ?

Selon le programme[116] de Première Scientifique, la simulation d'une expérience aléatoire joue un rôle important. Ainsi, *en permettant d'observer des phénomènes variés, elle amène les élèves à enrichir considérablement leur expérience de l'aléatoire et favorise l'émergence d'un mode de pensée propre* aux statistiques et aux probabilités. *Elle rend de plus nécessaire la mise en place de fondements théoriques*.

En Troisième et en Seconde, les élèves acquièrent une première pratique de l'aléatoire en pratiquant eux-mêmes des expériences de référence, comme des lancers de dés, de pièces, etc. et en simulant d'autres expériences à l'aide de listes de chiffres (ou de nombres) au hasard produit par un environnement numérique (calculatrice ou ordinateur).

Dans les classes du cycle Terminal Scientifique, une partie importante du programme concerne *la modélisation d'expériences de référence, modélisation permettant d'expliquer des résultats observés ou d'en prévoir d'autres* (extrait du B.O. n° 7). Ainsi, en Première Scientifique, les élèves approfondissent la notion *de simulation d'une expérience, qui consiste à choisir un modèle et à le simuler ; la simulation permet, d'une part, d'avoir des estimations de résultats impossibles à calculer explicitement et, d'autre part, par la comparaison de telles estimations avec des résultats expérimentaux, de valider le modèle choisi* (Ibid.).

Selon Gaydier (2011*), il y a un renversement dans l'utilisation de l'expression simulation, un glissement de « simulation d'une expérience aléatoire », qui renvoie à une certaine réalité, à « simulation d'une loi de probabilité », qui renvoie à la théorie*.

[116] B.O. hors-série n°7 du 31 août 2000, Volume 5, p. 29 et 30 de l'édition papier.

3.1.4 Pourquoi utiliser l'algorithmique ?

Nous pouvons supposer qu'une situation aléatoire donnée peut être rendue isomorphe à un modèle probabiliste, par exemple à un modèle de *tirage de boules d'une urne de composition bien choisie* (Gaydier, 2011). En effet, c'est l'*expérience aléatoire de tirage de boules d'une urne qui est la simulation du modèle* (Ibid.). Sa *réalisation effective peut être « effectuée à la main » ou remplacée* (Ibid.) par un (ou des) algorithme(s) implémentable(s) dans un environnement informatique afin de le tester, *en ayant mis en place un dispositif expérimental qui rende plausible l'hypothèse de tirage au hasard* (Ibid.) (Par exemple, le fait que les boules soient indiscernables au toucher).

Notre ingénierie sur la « *politique des naissances* » peut par exemple être simulée par le lancer d'une pièce quatre fois tant que l'on obtient « FACE », ou on s'arrête dès qu'on a « PILE ». Ainsi, notre ingénierie doit permettre à l'élève de réaliser ce genre de conditions expérimentales. Selon le BO, *la simulation permet d'une part d'avoir des estimations de résultats impossibles à calculer explicitement et d'autre part, par la comparaison de telles estimations avec des résultats expérimentaux, de valider le modèle choisi*.

3.2 Notre ingénierie didactique

3.2.1 Une politique des naissances

3.2.1.1. La problématique d'une situation aléatoire autour d'une politique des naissances

Pour aider à la compréhension de ce qui suit, nous rappelons la problématique de la situation aléatoire que nous mettons en place. Celle-ci met en jeu une *politique des naissances* (Fig. 178) que les élèves doivent modéliser et simuler afin d'émettre des hypothèses sur les conséquences qu'entraineraient une telle politique sur l'évolution du nombre d'enfants par famille ainsi que sur le nombre de total de garçons.

Pour limiter le nombre de filles dans un pays, il est décidé que :
- *Chaque famille aura au maximum n enfants ;*
- *Chaque famille arrêtera de procréer après la naissance d'un garçon.*

On considère que la probabilité de naissance d'un garçon est p, et pour chaque couple de parents, le sexe d'un enfant est indépendant du sexe des précédents.

Figure 178 (Une politique des naissances)

Ainsi, l'ingénierie didactique construite sur cette situation aléatoire permet de déterminer un modèle où la simulation va précéder la détermination de loi de probabilité associée à deux

variables aléatoires : nombre total d'enfants par famille (variable aléatoire discrète notée N) et nombre total de garçons (variable aléatoire discrète notée G).

3.2.1.2. Une première approche de nos choix généraux

a) La simulation, les calculs, la loi de probabilité

Nous faisons le choix dans un premier temps de supposer que la probabilité de naissance d'un garçon soit égale à celle d'une fille afin d'aider les élèves, en particulier de Seconde, à gérer moins de variables informatiques et faciliter ainsi la compréhension de cette politique proposée. Ainsi, lors d'une série de phases axées sur la simulation du processus aléatoire, les élèves sont conduits à étudier les moyennes du nombre d'enfants par famille et du nombre total de garçons, afin de conjecturer les conséquences d'une telle politique. Puis au cours d'une dernière phase, les élèves étudient l'aspect probabiliste de la politique. En effet, ils vont devoir entre autres déterminer les espérances mathématiques E(N) et E(G) des variables N et G, et ensuite déterminer le rapport E(N)/E(G) pour conclure quant à la validité « mathématique » d'une telle politique.

Ensuite, pour tous les élèves de Première Scientifique et les élèves volontaires de Seconde, nous faisons le choix de terminer la compréhension d'une telle politique en généralisant au cas où la probabilité de naissance d'un garçon serait égale à p.

b) Les environnements numériques

Nous faisons le choix de limiter les environnements numériques : le tableur, et les logiciels algorithmiques LARP et ALGOBOX. En effet, le tableur fait partie des environnements que les élèves de Seconde et de Première côtoient en classe depuis les deux dernières années du collège, que ce soit pendant les cours de mathématiques ou ceux de technologie. Quant aux environnements numériques LARP et ALGOBOX, les classes participant à cette dernière ingénierie ont eu la possibilité de se familiariser avec ces deux environnements tout au long du travail fait lors de l'ingénierie portant sur la dichotomie.

c) Les niveaux scolaires où se font l'expérimentation

Comme nous le signalions dans la section précédente, nous faisons le choix de poursuivre cette dernière expérimentation dans deux des classes qui ont participé à l'ingénierie sur la

dichotomie. En effet, ce choix permet de travailler avec des élèves qui ont la maitrise des environnements numériques utilisés lors de cette dernière expérimentation.

3.2.2 Etude du problème

Nous faisons le choix d'aborder la problématique de la situation aléatoire associée à la politique des naissances sous deux points de vue : **(1)** une approche « fréquentiste » afin de calculer des moyennes statistiques et des proportions sur un nombre important de familles ; **(2)** une approche « probabiliste théorique » avec le calcul d'espérances mathématique de variables aléatoire discrètes.

Ces deux approches supposent aussi deux modélisations. L'intérêt est alors que le travail sur deux modèles peut être une aide au travail sur la compréhension de la situation aléatoire. En effet, le modèle associé aux probabilités est du côté des mathématiques, tandis que celui associée à la simulation est du côté de l'algorithmique.

3.2.2.1. Le type d'algorithme en jeu pour la simulation

Comme nous le signalions plus haut, lors de cette ingénierie les élèves travaillent avec le tableur et des environnements numériques leur permettant d'implémenter des algorithmes écrits en langage « textuel » au sens de « pseudo-code » ou en langage « spatial » au sens d'organigramme.

Nous choisissons pour cette section de présenter l'aspect simulation à l'aide du tableur, qui est entre autres décrit par les auteurs du document d'accompagnement. Cette première approche à l'aide de simulations sur tableur permet aux élèves de faire des conjectures sur le nombre d'enfants, de garçons et sur la proportion de garçons dans une population donnée de familles.

Les naissances d'une famille peuvent être simulées sur une ligne. Si nous prenons une population constituée 1 000 familles, on obtient le tableau suivant :

A4 : =ENT(ALEA()+$K$1)

	A	B	C	D	E	F	G	H	I	J	K
1	Un colonne représente une naissance. Le nombre indiqué est le nombre de garçons.							Probabilité de naissance d'un garçon p =			0,5
2											
3	Naissances				Nombre d'enfants	Nombre de garçons					
4	1				1	1		Nombre total d'enfants N =		1847	
5	1				1	1					
6	1				1	1		Nombre total de garçons G =		951	
7	0	0	1		3	1					
8	0	1			2	1		Proportion de garçons p =		51,49%	
9	0	1			2	1					
10	0	0	0	0	4	0					
11	1				1	1					
12	1				1	1					
13	0	0	0	1	4	1					
14	0	1			2	1					

Tableau 46

Selon les cellules, nous entrons les formules suivantes :

- dans K1, la valeur de la probabilité de naissance d'un garçon (ici 0,5) ;
- dans A4, la formule =ENT(ALEA()+$K$1) qui donne le sexe de l'enfant qui nait (1 pour garçon et 0 pour fille) ;
- dans B4, la formule =SI(OU(A4=1;A4="");"";A4+ENT(ALEA()+$K$1)).

Cette formule est ensuite recopiée jusqu'à la cellule D4 afin d'avoir les quatre enfants au maximum pour une famille ;

- dans E4, la formule =NB(A4:D4) afin d'avoir le nombre d'enfants dans une famille ;
- dans F4, la formule =NB.SI(A4:D4;1) afin d'obtenir le nombre de garçons dans une famille ;

Ensuite, nous sélectionnons les cellules de la ligne 4 jusqu'à la ligne 1 003 afin d'avoir les résultats pour les 1 000 familles étudiées. Le calcul du nombre d'enfants est obtenu dans la cellule J4 en y inscrivant la formule =SOMME(E4:E1003) et le calcul du nombre de garçons est donné dans la cellule J6 par la formule =SOMME(F4:F1003). Quant à la proportion de garçons pour les 1 000 familles, elle est déterminée dans la cellule J7 par la formule =J6/J4. Si l'on souhaite obtenir plus d'enfants au maximum par famille, il suffit de recopier la formule de la cellule B4 jusqu'à la colonne correspondant à ce maximum d'enfants.

3.2.2.2. Le type de calcul probabiliste en jeu

Le traitement mathématique de cette politique peut aussi se faire à l'aide d'un arbre pondéré (Fig. 179) permettant de valider ou invalider les conjectures faites lors de simulations dans un environnement informatique.

a) Supposons dans un premier temps qu'il n'y ait au maximum que 4 enfants.

Afin d'aider à la compréhension du problème, nous proposons de représenter par un arbre pondéré les différents types de familles à 4 enfants possibles, sachant qu'à partir du premier

garçon, la famille arrête d'avoir des enfants. Nous supposons aussi que la probabilité de naissance d'un garçon lors d'une grossesse est *p*.

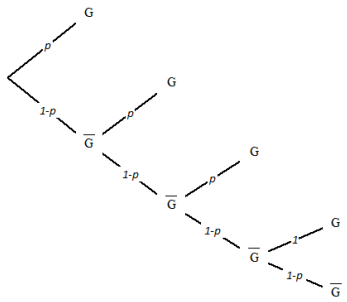


Figure 179 (Arbre pondéré représentant les différents types possibles de familles à 4 enfants)

Nous en déduisons ainsi les lois de probabilités respectives des variables N et G présentées dans le tableau ci-dessous :

Nombre total d'enfants (N)	Nombre total de garçons (G)	Probabilités
4 (quatre filles)	0	$(1-p)^4$
4 (trois filles et un garçon)	1	$(1-p)^3\, p$
3 (deux filles et un garçon)	1	$(1-p)^2\, p$
2 (une fille et un garçon)	1	$(1-p)\, p$
1 (un garçon)	1	p

Tableau 47

On en déduit alors que les espérances mathématiques respectives du nombre total d'enfants et du nombre total de garçons sont données par $E(N) = (2-p)(p^2 - 2p + 2)$ et $E(G) = p(2-p)(p^2 - 2p + 2)$. D'où la proportion de garçon est $E(G)/E(N) = p$.

b) Généralisons le problème

Supposons maintenant qu'une famille peut avoir *n* enfants au plus avec une probabilité de naissance d'un garçon à chaque grossesse de *p*. Nous avons alors une expérience aléatoire qui consiste à répéter dans des conditions identiques une *expérience de Bernoulli* de paramètre *p* avec au maximum *n* répétitions et nous arrêtons le processus au premier succès.

Soit X la variable aléatoire discrète qui représente le rang du 1er succès et qui vaut 0 si aucun succès n'a été obtenu. La variable aléatoire discrète X suit alors une *loi géométrique tronquée* de paramètres *n* et *p*.
Nous nous intéressons à la pertinence de cette politique sur l'évolution de la population du pays souhaitant la mettre en place.

Soit S la variable aléatoire associée au « nombre de succès » et T la variable aléatoire associée au « nombre d'étapes ».

La loi de probabilité (Tableau 48) que suit la variable aléatoire S ne prend que deux valeurs 0 et 1, avec :

k	0	1
P(S = k)	P(X = 0) = $(1-p)^n$	$\sum_{i=1}^{n} \mathrm{P}(\mathrm{X}=i) = 1-(1-p)^n$

Tableau 48

Alors, l'espérance mathématique de la variable aléatoire S est $E(S) = 1-(1-p)^n$.

De même, la loi de probabilité que suit la variable aléatoire T prend des valeurs entre 1 et n, avec :

- $P(T = k) = P(X = k) = (1-p)^{k-1}\,p$, où $k \in [\![1\,;n-1]\!]$;
- $P(T = n) = P(X = 0) + P(X = n) = (1-p)^n + (1-p)^{n-1}\,p$, où $k = n$.

Alors, l'espérance mathématique de la variable aléatoire T est :

$$E(T) = \sum_{k=1}^{n-1} k\,p\,(1-p)^{k-1} + n\,[p\,(1-p)^{n-1} + (1-p)^n].$$

Ce qui donne après simplification : $E(T) = \frac{1-(1-p)^n}{p}$. D'où : $E(S)/E(T) = p$.

En conclusion, si nous répétons un grand nombre de fois ce processus de n étapes au maximum (quelle que soit la valeur de l'entier n), nous obtenons en moyenne un nombre de succès égale à $1-(1-p)^n$ pour un nombre moyen d'étapes égale à $\frac{1-(1-p)^n}{p}$. Ainsi, en moyenne, nous avons encore une proportion de succès égale à p et ceci quel que soit le nombre maximal d'étapes du processus.

3.2.3 Les objectifs

Pour cette ingénierie, nous avons plusieurs objectifs. Un premier objectif qui s'inscrit dans une démarche de modélisation. En effet, la tâche va consister à interpréter une problématique donnée sous forme d'une question : « la mise en place de cette politique a-t-elle la conséquence attendue, à savoir de diminuer le nombre de filles dans la population ? ». Pour cela, les élèves vont pouvoir mener deux types d'approche : (1) une approche « fréquentiste » du processus aléatoire, où les élèves vont calculer des moyennes en lien avec le nombre d'enfants et le nombre de garçon, à l'aide du tableur dans un premier temps, puis en utilisant un modèle représenté par un (ou des) algorithme(s) implémentable(s) dans un environnement numérique (autre que le tableur) afin de le (ou les) tester pour déterminer une valeur approchée de la proportion de garçons parmi le nombre d'enfants pouvant naître, cet algorithme pouvant dépendre de plusieurs paramètres comme la probabilité de naissance d'un garçon, le nombre d'enfants maximum que peut avoir une famille et le nombre de

familles étudiées ; (2) une approche « probabiliste » du problème, en mettant en place un modèle mathématique permettant de déterminer les lois de probabilité de deux variables aléatoires (N pour nombre total d'enfants et G pour nombre de garçons) afin de calculer leurs espérances mathématiques respectives pour obtenir la proportion de garçons par rapport au nombre total de naissances.

Quel que soit le niveau scolaire, nous avons aussi l'objectif que les élèves présentent un arbre de probabilité pour lequel tous les chemins ne sont pas de même longueur.

Dans le cadre de notre recherche, nous souhaitons analyser le travail et les résultats obtenus des élèves en procédant à des analyses basées sur notre cadre théorique, en particulier les articulations entre *ETM* et *ETA* spécifiques, d'un point des genèses et l'étude des paradigmes algorithmiques en jeu dans cette ingénierie.

3.2.4 Les attentes

Notre ingénierie se découpe en quatre phases réparties sur une unique séance de deux heures, avec une pause de dix minutes entre les deux heures correspondant au temps de « récréation » de l'après-midi.

Au cours de la première phase (20 minutes), nous attendons que les élèves vivent un temps d'échange collectif où ils émettent des hypothèses sur l'évolution démographique de la population si une telle politique était mise en place. Ainsi, nous attendons qu'ils se questionnent sur la conséquence « espérée » sur la population, à savoir la diminution du nombre de filles dans la population.

Cette première phase est suivie d'une seconde d'une durée de 30 minutes, où les élèves utilisent un tableur pour simuler la politique des naissances sur une population de 1 000 familles, puis de 10 000 familles, afin de travailler sur des calculs de moyennes permettant d'évaluer la proportion du nombre total de garçons par rapport au nombre total d'enfants.

Puis, lors d'une troisième phase durant 50 minutes, nous attendons qu'ils élaborent un modèle basé sur des lancers d'une pièce simulant cette politique et qu'ils construisent un algorithme correspondant soit à ce modèle, soit à la politique telle qu'elle est définie au départ afin de l'implémenter dans un environnement numérique autre que le tableur, pour le tester sur un nombre de familles donné, une probabilité de naissance d'un garçon donnée et le choix

du nombre maximal d'enfants pour une famille, le tout permettant de déterminer une estimation de la proportion moyenne du nombre total de garçons par rapport au nombre total d'enfants par famille.

Progressivement, pendant les trois phases précédentes, les élèves ont pris conscience de l'importance de prendre le temps de raisonner avant d'émettre et de valider des conjectures. Ainsi, au cours d'une dernière phase, nous attendons que les élèves ayant travaillé sur une approche « fréquentiste » de la problématique où ils ont pu proposer des conjectures à travers une série de calculs de moyennes, cherchent à valider ces conjectures en mettant en place une approche « probabiliste » de la problématique leur permettant de travailler sur des calculs d'espérances.

Pour conclure sur les attentes, nous comptons analyser ces différentes phases dans le cadre d'une étude des articulations possibles entre *ETA* et *ETM* spécifiques.

3.2.5 Les choix de la mise en place de l'expérimentation

3.2.5.1. Choix des classes et des conditions de travail

Nous choisissons de mener cette ingénierie dans deux classes : une Seconde et une Première Scientifique. La classe de Seconde est une classe d'un lycée international de la région parisienne et la classe de Première est située dans un établissement ordinaire de banlieue. Les enseignants accompagnant ces classes n'ont pas de connaissances spécifiques en informatique et en algorithmique. En revanche, ils ont chacun participé quelques semaines auparavant à l'ingénierie globale sur la dichotomie avec leur classe respective.

Nous faisons le choix que les deux classes travaillent pendant toute l'ingénierie dans une salle informatique. Dans chacune des classes, les élèves sont répartis en binôme afin que chaque binôme ait un ordinateur allumé à sa disposition.

Pour chaque classe, du matériel audio et vidéo est installé dans la salle, ce qui permet d'enregistrer le travail fourni par les élèves lors de la séance.

Nous faisons aussi le choix d'organiser une ingénierie locale durant deux heures autour d'une activité dans le domaine des probabilités mettant en jeu des compétences en algorithmique (simulations, modélisation, structures de boucles, nombre aléatoire) et en

mathématiques (variables aléatoires discrètes, probabilités, arbre pondéré de probabilité, espérance mathématiques, proportions, fréquences, moyennes, modèle de répétition d'expériences identiques et indépendantes).

Nous référant aux propos données par les auteurs du document d'accompagnement de Première sur le fait que […] *la détermination de l'espérance de la loi géométrique tronquée de paramètres n et p mobilise à la fois les suites géométriques et la dérivation* […], nous choisissons de mettre en place cette ingénierie dans la classe de Première après les cours sur les suites géométriques et la dérivation, même s'il n'est pas attendu lors de cette expérimentation une démonstration de la formule donnant l'espérance de la loi géométrique tronquée. Nous supposons ainsi que le contrat didactique en place au niveau de la classe de Première peut permettre aux élèves de penser qu'ils peuvent ne pas se contenter de quelques simulations, mais aussi chercher à déterminer une « preuve » mathématique (au sens probabiliste) de la proportion de garçons par rapport au nombre d'enfants.

Pour faciliter, l'organisation des expérimentations, celle de la classe de Seconde a lieu en parallèle de celle de Première et sans la présence du chercheur. En effet, le chercheur est présent dans la salle où les élèves de Première assurent cette expérimentation.

De plus, cette expérimentation se fait environ un mois après que celle sur la dichotomie ai été faite, afin de faire travailler des élèves qui ont déjà la connaissance des environnements numériques choisis pour cette nouvelle ingénierie.

3.2.5.2. Choix des prérequis attendus pour les élèves

a) Dans le domaine des probabilités

L'enseignant de Première introduit en amont de notre expérimentation la notion de variable aléatoire discrète, de loi de probabilité associée à une variable aléatoire discrète, du calcul de l'espérance d'une variable aléatoire, puis fait travailler les élèves sur des situations aléatoires se ramenant à une *épreuve de Bernoulli*.

En revanche, au niveau de la Seconde, ces notions n'étant pas inscrites au programme, nous nous demandons à l'enseignante d'introduire à partir d'exemples simples sans présenter aucune théorie particulière les mêmes notions en lien avec les variables aléatoires et de

travailler quelques situations aléatoires se ramenant à une *épreuve de Bernoulli*. Nous faisons ce choix à ce niveau en tenant compte du très bon niveau de la classe en mathématique.

Nous faisons aussi le choix de faire travailler des élèves qui n'ont aucune connaissance sur la loi binomiale et la loi géométrique tronquée.

b) Dans les domaines de l'informatique et de l'algorithmique

L'ensemble des élèves ont la connaissance et la pratique des formules mathématiques du tableur. En algorithmique, les élèves ont aussi les savoir-faire sur les principes d'élaboration de listes d'instructions à suivre qui, à partir des données, permettent d'obtenir des résultats clairement définis en un nombre fini d'étapes (Fig. 180) :

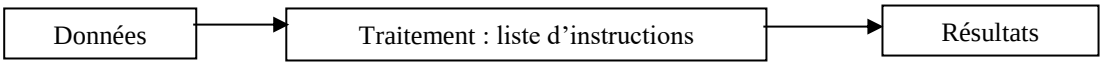


Figure 180 (Principe d'élaboration d'une liste d'instructions)

Ils ont aussi une pratique régulière des différentes fonctions utiles à la conception d'algorithmes autour des variables (Nombre, liste de nombres, chaine de caractères), des affectations, des structures alternatives « Si … Alors » et « Si … Alors … Sinon », des structures itératives « POUR », quand on connaît le nombre de répétitions et « TANTQUE », quand on connaît un test d'arrêt mais pas le nombre de répétitions.

Nous faisons le choix de demander aux élèves de relire (deux semaines avant la séance) le polycopié sur la conception d'organigrammes (cf. Annexe) distribué avant le début de l'ingénierie sur la dichotomie. En effet, nous faisons ce choix car les élèves n'ont pas une grande habitude d'exploiter cette technique de représentation d'un algorithme. Par ce polycopié, nous choisissons que les élèves acquièrent d'eux-mêmes les savoirs nécessaires sur les symboles usuels dans l'élaboration d'organigrammes simples. Cependant, afin de compléter cet apprentissage, lors de l'heure qui précède la séance, nous faisons le choix que les élèves avec leur enseignant consacrent une trentaine de minutes pour retravailler une série de deux petits exercices sur la construction d'organigrammes : (1) organigramme permettant d'obtenir la valeur absolue d'un nombre réel donné et (2) organigramme permettant d'obtenir la table de multiplication d'un nombre entier donné.

3.2.5.3. Choix de l'environnement informatique

En accord avec le chercheur, les enseignants des deux classes demandent aux élèves de n'utiliser qu'un nombre restreint d'environnements numériques. En effet, nous faisons le choix de faire que les élèves n'utilisent que les deux environnements numériques : *LARP* et *ALGOBOX* qu'ils ont déjà pratiqués lors de l'ingénierie sur la dichotomie, et qu'ils puissent aussi travailler dans le cadre d'une des phases avec un tableur. Ainsi, les deux salles informatiques où ont lieu les expérimentations sont équipées d'ordinateurs où les environnements numériques choisis sont installés. Comme pour l'ingénierie sur la dichotomie, le choix des deux environnements algorithmiques va permettre aux élèves d'approfondir l'écriture d'algorithmes en langage pseudo-code avec *ALGOBOX* et *LARP*, mais aussi d'écrire des algorithmes sous forme d'organigramme avec *LARP*. Ainsi, nous rappelons que ce choix permet à l'élève de voir qu'un « texte » ou un organigramme écrit dans ces deux environnements numériques peut être vu soit comme un algorithme si on s'intéresse à sa logique et à ses performances, soit comme un programme si on s'intéresse à son implémentation.

Le choix d'utiliser un tableur pour une des phases (la deuxième) vient de la demande de l'enseignante de Seconde qui se réfère aux acquis des élèves sur l'utilisation du tableur lors des années de collège et l'existence de divers exercices proposés dans le manuel de la classe mettant en place des applications numériques avec le tableur, en particulier dans les domaines des statistiques et des probabilités. Initialement, ce choix n'avait pas été envisagé par le chercheur. Cependant, après un échange sur l'utilisation ou pas du tableur entre les deux enseignants et le chercheur en amont de l'expérimentation, il a été d'avis que ce choix du tableur pouvait être judicieux au regard des compétences mathématiques en jeu dans la résolution de certaines étapes du problème et des fonctionnalités qu'offraient le tableur qui, potentiellement, soutiennent ces compétences. (Tableau 49).

Compétences mathématiques	**Fonctionnalités du tableur**
(a) Utilisation d'une représentation graphique – Une entrée dans l'algorithmique	
Un arbre de probabilité	Une succession de structures conditionnelles « Si…. Alors… Sinon »
(b) Utilisation de formules	
Introduction de lettres, mise en place de symboles • Découvrir le concept de nombres aléatoires, de partie entière d'un nombre réel, …	Jouer sur les relations dynamiques entre les cellules Les objets du tableur : « variable – cellule » et « formule – cellules »

• Identifier des relations entre des nombres comme pouvant être une formule	
(c) Faire travailler sur les formules et les variables	
Transition du numérique ou du verbal ou du graphisme vers le symbolique Transition du spécifique vers le général : • Généraliser des calculs numériques dans une même formule • Déterminer, écrire une formule correspondant aux calculs numériques voulus	Jouer sur les relations entre plusieurs colonnes Utiliser la fonctionnalité de « copier-glisser » Objets du tableur : « formule – colonne » et « variable – colonne »
(d) Approcher la résolution de la problématique : conséquence sur le nombre de filles, de garçons, d'enfants	
Calculs d'espérances mathématiques	Calculs de moyennes

Tableau 49

Lors de la mise en place de notre expérimentation, cette gradation va nous permettre d'étudier pendant l'observation du travail des binômes, l'intégration du tableur (qui intervient dès la phase 2) comme objet de calculs lors de l'approche « fréquentiste » du problème de la *politique des naissances*, mais aussi comme transition vers des stratégies de choix d'une structure algorithmique adaptée (travail fourni dans la phase 3).

Nous pouvons aussi supposer qu'une entrée dans la simulation par une utilisation du tableur peut aider les élèves débutant en informatique à aller vers une approche algorithmique de la simulation de cette politique.

L'ordinateur de chacun des deux enseignants est connecté au tableau numérique pour les moments collectifs.

3.2.6 Les tâches pour chacune des phases de notre ingénierie et le mode de travail

L'expérimentation sur la politique des naissances démarre environ un mois après l'ingénierie sur la dichotomie (cf. les chapitres 3 à 5). La veille de l'expérimentation, les élèves retravaillent la construction d'organigrammes sur deux exemples simples : (1) détermination de la valeur absolue d'un nombre réel donné ; (2) construction de la table de multiplication d'un nombre entier donné en proposant deux types d'organigramme : un avec une boucle « TantQue », l'autre avec une boucle « Pour ». Ces exercices se font en présence de l'enseignant sans qu'il y ait un retour auprès du chercheur, mais seulement une confirmation que ces exercices ont pu être traités en classe.

Nous rappelons que cette expérimentation sur la politique des naissances se déroule au cours d'une unique séance de deux heures découpée en quatre phases. Nous proposons ainsi de présenter ci-dessous les différentes tâches attendues pour chacune des phases ainsi que le mode de travail.

3.2.6.1. Phase 1 : Temps de réflexion au cours d'un échange collectif sur l'évolution démographique de la population que pourrait entrainer cette politique (20 minutes)

Au début de cette phase qui commence la séance de deux heures, l'enseignant doit distribuer à tous les élèves un texte présentant la politique des naissances (Fig. 181). Les élèves ont environ cinq minutes pour le lire et comprendre la « portée » de cette politique.

Cette histoire est basée sur des faits réels, qui ont par exemple été instaurés dans certains pays à forte population. Voulant augmenter la proportion des garçons au sein de sa population naissante, un gouvernement d'un pays imaginaire décide d'appliquer une nouvelle politique nataliste à partir de la règle suivante :
Les naissances au sein d'une famille s'arrêtent :

- *soit à la naissance du premier garçon ;*
- *soit lorsque la famille comporte 4 enfants.*

Les personnes de ce gouvernement ayant voté cette loi affirment qu'il y aura plus de garçons puisque 50 % des familles auront 1 garçon et pas de fille. Les opposants rétorquent qu'il y aura plus de filles à cause des familles de plus de 2 enfants.
L'idée est d'utiliser la simulation afin de répondre à cette question.

Figure 181 (Texte distribué aux élèves sur la politique des naissances 1ère partie)

Après, l'enseignant a pour tâche de distribuer la suite de la présentation de la problématique (Fig. 182). Il a pour consigne de préciser que ce complément doit permettre de simplifier l'étude du problème sans pour autant en dénaturer ses objectifs.

Pour simplifier, on admet que tout couple est capable de procréer 4 fois et on peut négliger l'impact des familles sans enfant sur la natalité.
On considère qu'à chaque naissance, on a une chance sur deux d'avoir un garçon et que, pour chaque couple de parents, le sexe d'un enfant est indépendant du sexe des précédents.

Figure 182 (Texte distribué aux élèves sur la politique des naissances 2ème partie)

Puis, il doit terminer en projetant les questions suivantes (Fig. 183) au tableau numérique. Il a pour consigne de commenter oralement ces questions en signalant aux élèves qu'ils ont droit à quinze minutes d'échange collectif pour proposer leurs réflexions sur la remarque formulée par les membres du gouvernement fictif. Il doit aussi préciser qu'il n'est qu'un observateur et un modérateur.

On procède tout d'abord à un sondage d'opinion. Ainsi, avant de procéder à une simulation, donnez votre impression sur les effets de cette politique nataliste (on cochera une seule case par questionnaire) :
Concernant l'équilibre des sexes

- ❑ *il y aura plutôt plus de garçons que de filles ;*
- ❑ *il y aura plutôt plus de filles que de garçons ;*
- ❑ *il y aura environ autant de filles que de garçons ;*
- ❑ *ne se prononce pas.*

Concernant l'évolution démographique

- ❑ *la population va plutôt augmenter ;*
- ❑ *la population va plutôt diminuer ;*
- ❑ *la population va rester plutôt stable ;*
- ❑ *ne se prononce pas.*

Figure 183 (Sondage d'opinion distribué aux élèves sur la politique des naissances 3ème partie)

Les binômes ont pour tâche de répondre à cette série de questions en s'aidant du « papier-crayon » afin de de mener une première réflexion au sein du binôme sur des questions soulevées par la problématique de la politique proposée. Une fois cette réflexion faite, les élèves ont pour tâche d'engager un bref débat sur les hypothèses qu'ils ont émises au cours de ce premier temps de réflexion. Les élèves travaillent en totale autonomie pendant cette tâche qui est prévue durer une quinzaine de minutes. L'enseignant se contente d'une tâche de modérateur au sein de la classe.

3.2.6.2. Phase 2 : Approche « fréquentiste » de la proportion de garçons. Simulation à l'aide d'un tableur sur une population de 1 000 familles, puis de 10 000 familles (30 minutes)

Pour cette deuxième phase, les élèves ont pour tâche de mettre en pratique leurs connaissances sur le tableur pour simuler un grand nombre de familles et observer la fréquence statistique de garçons avec une hypothèse d'équiprobabilité que pour chaque naissance un enfant soit un garçon ou une fille. Les élèves ont pour consigne de ne pas demander d'aide à leur enseignant. Cette première approche « fréquentiste » est cohérente avec les pratiques en simulation au lycée.

Au début de cette phase, l'enseignant a pour consigne de distribuer à chaque binôme le document suivant :

- *A l'aide d'un tableur faire une simulation de cette politique des naissances pour 1 000 familles, en simulant les naissances d'une famille sur une ligne.*
- *Exécuter sur un tableur la simulation des naissances de 10 000 familles, en indiquant le nombre total d'enfants (noté : NE), le nombre total de garçons (noté : NG) et la proportion de garçons (noté PG)*

Figure 184 (Document fourni aux élèves en lien avec une utilisation du tableur pour un nombre de familles fixé)

3.2.6.3. Phase 3 : Organigramme – Modélisation – Approche « fréquentiste » de la proportion de garçons – Simulation en langage pseudo-code (50 minutes)

A la phase précédente, le tableur a été retenu parce que c'est l'outil de simulation généralement utilisé à ces niveaux scolaires. Il s'agit ici pour les élèves de considérer la simulation comme un traitement algorithmique, en mettant en œuvre des structures qu'ils ont mises en œuvre dans d'autres domaines mathématiques, en particulier au cours de l'ingénierie sur la dichotomie.

L'enseignant a pour consigne de distribuer au début de cette phase le document suivant :

- *Construire un organigramme associé à la politique des naissances proposée par le gouvernement fictif.*
- *Avec AlgoBox faire une simulation de cette politique des naissances pour un nombre quelconque de familles, et observer la proportion de garçons parmi les naissances.*
- *Modifier la simulation en supposant que la probabilité d'avoir un garçon lors d'une naissance est un nombre p compris entre 0 et 1.*
- *Modifier l'algorithme précédent afin de conjecturer la proportion de naissance de garçons rapporté au nombre total pour un nombre donné de familles.*

Figure 185 (Tâches algorithmiques que les élèves doivent faire autour de la politique des naissances)

Les élèves ont donc pour tâche de construire des modèles représentatifs de la politique des naissances et de construire des simulations de ces modèle à partir d'algorithmes (aux format « spatial » et « textuel ») mettant en place des variables informatiques qu'ils doivent déterminer et interpréter, et utiliser différentes structures comme « TantQue » pour le nombre maximal d'enfants dans une famille et « Pour » pour un nombre donné de familles, ainsi que l'utilisation d'une instruction conditionnelle correspondant au sexe de l'enfant.

3.2.6.4. Phase 4 : Approche « probabiliste » du problème : détermination de deux lois de probabilité et calculs de leurs espérances mathématiques pour obtenir la proportion de garçons parmi le nombre total d'enfants (20 minutes)

Bien que cela soit implicite, le travail fait lors des phases 2 et 3 doit se terminer par un retour sur les hypothèses concernant l'évolution de la population émises lors de la phase 1. En effet, les élèves doivent prendre l'initiative de mettre en lien les résultats obtenus lors des simulations (dans un environnement numérique) avec les hypothèses faites au début de la séance. Pour cela, nous attendons qu'ils justifient les conjectures émises lors de l'approche « fréquentiste » faite pendant les deux phases précédentes en déterminant, dans le cadre d'une approche « probabiliste » de la problématique, les lois de probabilités associées

respectivement aux nombres de garçons et au nombre total d'enfants suivant les différents types de familles possibles, afin d'en déduire des espérances mathématiques et la proportion de garçons dans le nombre total d'enfants.

3.2.7 Les analyses *a priori*

Pour aider à la compréhension des analyses qui suivent, nous rappelons la politique des naissances (Fig. 186) sur laquelle les élèves doivent travailler au début de l'expérimentation. Nous rappelons aussi que pour les phases 1 et 2, nous faisons l'hypothèse d'équiprobabilité que pour chaque naissance un enfant soit un garçon ou une fille.

Les naissances au sein d'une famille s'arrêtent :
- *soit à la naissance du premier garçon ;*
- *soit lorsque la famille comporte 4 enfants.*

Figure 186 (Rappel de la politique des naissances)

3.2.7.1. Phase 1 : Temps de réflexion au cours d'un échange collectif sur l'évolution démographique de la population que pourrait entrainer cette politique (20 minutes)

Pendant les cinq minutes prévues pour la lecture et l'interprétation du texte présentant la politique des naissances, nous prévoyons que les élèves interprètent ce texte sans difficultés et que même certains élèves prennent l'initiative de représenter les formats des familles possibles sous la forme d'un arbre pondéré en prenant comme issues à chaque naissance « Garçon » ou « Fille ». Nous nous attendons cependant, qu'au cours de cette première phase les élèves n'inscrivent pas nécessairement les probabilités sur les différentes branches de l'arbre. En revanche, lors de l'échange qui va suivre entre élèves, nous pensons que ces derniers vont compléter leurs arbres en inscrivant les valeurs des probabilités de chaque branche. En effet, à ce stade de la phase, l'enseignant leur aura fait parvenir des compléments sur les conditions d'équiprobabilité d'avoir un garçon ou une fille lors d'une naissance.

Concernant le choix du binôme parmi la série d'affirmations (cf. 3.2.6.1) concernant l'évolution de la population, nous nous attendons à ce que les binômes considèrent que l'hypothèse émise par les membres du gouvernement fictif puisse être exacte ou qu'ils ne se prononcent pas en particulier chez les élèves qui se contenteront de répondre sans prendre le temps de dessiner un arbre par exemple, voire de commencer un début d'approche « probabiliste » du problème. Nous rappelons que celle-ci n'est pas attendue dans cette phase.

En effet, nous n'attendons pas à ce que les élèves puissent avoir une idée précise sur ces questionnements d'équilibre des sexes et de l'évolution démographique. Nous supposons que le fait de ne pas tout de suite proposer aux élèves de recourir à un modèle comme la construction d'un arbre peut favoriser une réponse du type « ne se prononce pas » en particulier chez des élèves de Première Scientifique. En effet, ces derniers sont habitués à expliquer, voire justifier leurs réponses à une question demandée par l'institution, en particulier en mathématique. Nous pensons alors qu'ils vont vouloir justifier leurs réponses et que l'absence d'indication à ce stade de l'ingénierie va les inciter à ne proposer qu'une réponse évasive. En revanche, chez des élèves de Seconde, nous prévoyons une plus grande diversité de réponses. En effet, ces derniers, sortant du collège n'ont pas nécessairement pris conscience de l'importance de proposer une argumentation à une affirmation donnée, en particulier quand celle-ci ne leur semble pas relever d'un domaine précis des mathématiques, en l'occurrence les probabilités, qui fait partie de leurs connaissances récentes (son introduction est souvent faite en fin de Troisième).

Cependant, nous supposons que le fait de prévoir une vingtaine de minutes pour répondre à cette série d'affirmations peut amener certains élèves à essayer de proposer un début de justification de leurs réponses en particulier avec la construction d'un arbre et peut-être un début de tentatives de calculs de probabilités. En effet, les élèves étant familiarisés avec la construction d'arbres de probabilité pour représenter certaines situations aléatoires, nous supposons que la reconnaissance du phénomène aléatoire dans la politique proposée, peut les inciter à construire un arbre de probabilité. De plus, nous pouvons penser que le nombre maximum limité à 4 enfants dans une famille peut aussi favoriser ce choix.

3.2.7.2. Phase 2 : Approche « fréquentiste » de la proportion de garçons. Simulation à l'aide d'un tableur sur une population de 1 000 familles, puis de 10 000 familles (30 minutes)

Dans cette phase, comme nous l'avions déjà évoqué plus haut, nous pensons qu'une entrée dans la simulation par une utilisation du tableur va aider les élèves à aller vers une approche algorithmique de la simulation de cette politique qui fera l'objet du travail de la phase suivante. En effet, nous supposons que des élèves, ayant une pratique de cet environnement numérique depuis les deux dernières années de collège, en particulier lors des enseignements

suivis en cours de technologie et en mathématiques, auront moins de difficultés avec l'utilisation des fonctions permises par cet environnement. En effet, nous pensons que les élèves vont prendre l'initiative d'utiliser les fonctions suivantes =ENT(ALEA()+0,5) ou =ALEA.ENTRE.BORNES(0;1) pour déterminer le sexe de l'enfant lors de la naissance (0 : pour une fille et 1 : pour un garçon), =NB(A4:D4) pour compter le nombre d'enfants dans une famille, puis =NB.SI(A4:D4;1) pour déterminer le nombre de garçons dans une famille.

Nous supposons que l'utilisation du tableur va aussi permettre aux élèves de prendre conscience que la mise en place d'un algorithme construit sur une succession de structures conditionnelles « SI…ALORS… » peut être lourd à gérer. En effet, les élèves savent qu'il est possible d'imbriquer plusieurs niveaux : chaque « SINON » peut ouvrir sur un nouveau « SI ». Ainsi, avec un nombre réduit d'enfants par famille (ici maximum 4 enfants pour une famille), nous attendons que les élèves restent dans ce mode de raisonnement. En ayant choisi de faire travailler les élèves sur des familles ayant au maximum 4 enfants, nous pensons qu'ils peuvent ne pas comprendre dans un premier temps qu'il est préférable de se limiter à un nombre réduit de niveaux, car plus ceux-ci vont devenir importants plus il va être difficile d'analyser le code. Nous pensons ainsi que l'utilisation d'un tableur va aider les élèves à prendre conscience de cette « lourdeur ». Cependant, nous supposons aussi que l'utilisation du tableur ne peut pas permettre aux élèves de penser à une autre stratégie basée sur une utilisation d'une structure de type « TantQue ».

Pour obtenir la « proportion de garçons » demandée dans l'énoncé, les élèves vont devoir simuler dans un premier temps 1 000 familles. La famille créée étant organisée sur 4 cellules d'une colonne, on s'attend à ce que les élèves utilisent spontanément la recopie vers la droite de ces quatre cellules, de façon à remplir 1 000 colonnes. On s'attend aussi à ce qu'ils utilisent la fonction NB.SI, respectivement avec l'argument 0 et l'argument 1 pour obtenir le nombre total de filles, respectivement de garçons, puis, par calcul, la fréquence de garçons. Bien que ce ne soit pas explicitement demandé dans l'énoncé, on s'attend à ce que les élèves observent cette fréquence, et éventuellement ses variations en répétant l'expérience avec la touche F9.

Pour conclure sur cette phase, nous pensons qu'après le travail fait sur 1 000 familles, les élèves ne vont pas éprouver de difficultés particulières pour la simulation à 10 000 familles. Nous pensons aussi que le fait de leur demander de simuler pour 10 000 familles, après l'avoir

fait pour 1 000 familles, peut les conduire à se poser la question sur l'importance d'avoir un nombre de familles conséquent pour affiner les calculs de moyennes et par conséquent les préparer à l'approche « probabiliste » de la phase 4 avec le travail sur les espérances mathématiques, sans pour autant avoir une approche théorique à travers la loi des grands nombres qui ne fait pas partie des compétences mathématiques à ces niveaux de la scolarité.

3.2.7.3. Phase 3 : Organigramme – Modélisation – Approche « fréquentiste » de la proportion de garçons – Simulation en langage pseudo-code (50 minutes)

Afin de faciliter la lecture, nous rappelons que pour cette phase, les binômes doivent d'abord construire un organigramme permettant de représenter la politique des naissances proposée par le gouvernement fictif et ensuite construire un algorithme au format « textuel » pour représenter un modèle de la simulation de la politique avec un nombre quelconque de familles, afin d'observer la proportion de garçons parmi les naissances. Cet algorithme doit être alors implémenté dans l'environnement numérique *AlgoBox* afin d'être testé. Puis, les élèves doivent modifier la simulation en supposant que la probabilité d'avoir un garçon lors d'une naissance soit un nombre *p* compris entre 0 et 1. Ensuite, les élèves adaptent l'algorithme précédent afin de conjecturer la proportion de naissance de garçons rapporté au nombre total pour un nombre donné de familles.

a) L'organigramme

Nous rappelons que la construction d'organigrammes n'est pas conseillée par l'institution. Nous pouvons penser alors que cette construction peut être encore une source de difficultés pour des élèves débutants, malgré le travail fait un mois auparavant lors de l'ingénierie sur la dichotomie. Dans le cas où les élèves ont choisi de dresser un arbre de probabilité pour déterminer les types de familles possibles lors de la phase 1, nous pouvons supposer que ceux-ci vont se contenter de transcrire l'arbre en un organigramme constitué d'une succession de structure conditionnelle « Si … alors ». Nous pensons que l'utilisation d'une boucle « TantQue » au cours de cette construction de l'organigramme peut ne pas être utilisée par les élèves. Nous pensons que l'aspect « dessin » de l'organigramme va favoriser cette transcription d'un arbre de probabilité à quatre niveaux de branches, qui est lui-même un « dessin ». Nous supposons aussi que le choix d'imposer dans un premier temps la construction d'un organigramme au lieu d'un algorithme « textuel » ne va pas inciter l'élève

débutant à bâtir un algorithmique faisant référence à une structure de type « TantQue ». Nous pensons ainsi que sur cet aspect de l'algorithme « spatial », nous pouvons observer des différences entre des élèves de Seconde qui sont toujours dans la découverte pour une grande majorité d'entre eux de la notion d'algorithme et des élèves de Première Scientifique qui ont déjà une année de pratique des algorithmes écrit sous forme « textuelle ».

b) Modélisation – Approche « fréquentiste »

Nous pensons que les élèves vont interpréter, dans un premier temps, la question « *Avec AlgoBox faire une simulation de cette politique des naissances pour un nombre quelconque de familles, et observer la proportion de garçons parmi les naissances* » comme nécessitant une première approche de type modélisation probabiliste avant d'aborder l'aspect algorithmique de la simulation. Ainsi, certains élèves vont souhaiter bâtir un modèle mathématique permettant une simulation de la politique des naissances. Nous pensons que ces élèves vont vouloir choisir un modèle dont ils ont l'habitude d'utiliser : des lancers successifs d'une pièce de monnaie équilibrée.

Ainsi, nous pensons que le choix d'un tel modèle va être interpréter comme le fait que les deux issues possibles pour chaque lancer sont : PILE ou FACE, chacune étant associée à un des deux sexes. Nous pensons aussi que ces deux issues vont être interprétées par des élèves de ces niveaux scolaires comme représentatifs de deux événements contraires : G : « avoir un garçon » et F : « avoir une fille ». Le fait que la fréquence théorique d'obtenir un garçon à chaque naissance soit de 0,5 dans la politique donnée, nous supposons que les élèves vont considérer cette condition en termes d'équiprobabilité, c'est-à-dire le lancer d'une pièce équilibrée. Nous pensons aussi que le fait de répéter l'action de lancer la pièce de monnaie va permettre aux élèves de concevoir chaque naissance comme une épreuve de Bernoulli, où l'issue représentative du succès serait S : « l'enfant qui nait est un garçon » avec une probabilité *p* connue. Pour finir, nous pensons aussi que le fait d'avoir 4 enfants au maximum va être interprété par les élèves comme une répétition successive de 4 lancers.

Suivant le modèle choisi par le binôme, nous nous attendons à ce que les élèves éprouvent des difficultés à gérer le type de variable informatique utilisée. En effet, certains élèves peuvent par exemple penser que pour stocker le résultat de chaque lancer, il faille introduire une variable de type LISTE et qu'il faille aussi compter les enfants afin de stopper à quatre

lancers au maximum, représentatifs de quatre enfants au maximum. Cette dernière variable peut avoir une conséquence dans la gestion des variables lors de la construction de l'algorithme associé au modèle. En effet, le fait de compter les lancers n'est pas pris en compte comme une variable de type NOMBRE dans le cas du tableur. Il suffit d'arrêter le « copier-coller » fait à l'aide de la main au bout de la 4ème colonne représentant le 4ème lancer (ou le 4ème enfant) (cf. phase 2). De plus, nous pensons que les élèves peuvent négliger la nécessité d'initialiser le nombre de PILE (associé au nombre de garçons dans une famille) et le nombre de lancers pratiqués (associé au nombre d'enfants dans une famille) dans l'algorithme.

Lors de la construction d'un algorithme permettant de simuler le modèle choisi, nous pensons que malgré le travail mis en place pendant la phase précédente avec le tableur (cf. la description du choix de l'environnement informatique, paragraphe 3.2.5.3), une majorité d'élèves, en particulier au niveau de la Seconde, vont proposer un algorithme constitué d'une succession de « SI… ALORS » ou de « SI … ALORS …SINON ». En effet, nous supposons qu'un tel algorithme va être considéré par des débutants en informatique comme représentatif de l'arbre de probabilité, en particulier quand le nombre de répétition est réduit.

Nous nous attendons aussi à ce que certains élèves éprouvent des difficultés lors de la transcription de l'algorithme écrit au « papier-crayon » à l'algorithme écrit en langage machine afin d'être implémenté dans un environnement informatique pour le tester. Ces difficultés peuvent être dues à une maitrise insuffisante de la syntaxe de l'environnement choisi, bien que les élèves aient déjà l'expérience de ces environnements.

En ayant demandé aux élèves de ne plus réutiliser le tableur au cours de cette phase, nous pensons qu'ils ne vont pas se contenter d'une simple transposition de l'arbre de probabilité pour l'écriture de l'algorithme associé au modèle. Ainsi, nous supposons qu'une intervention de l'enseignant sur ce point peut s'avérer nécessaire. En effet, ce dernier peut montrer que si le nombre maximum de lancers (ou d'enfants dans le cas des familles) devient important, il s'avère alors nécessaire de trouver une autre stratégie pour déterminer les résultats possibles. Pour cela, nous supposons que les élèves ayant une connaissance des différentes structures de boucles possibles, vont interpréter la condition « nombre maximum » en termes de boucle de type « TantQue ».

De plus, une fois que l'algorithme est implémenté dans l'environnement numérique, nous pensons que les élèves vont modifier et exécuter celui-ci afin de prendre en compte que la probabilité d'avoir un garçon lors d'une naissance est *p* et non 0,5.

Pour conclure, nous pensons que les élèves de ce niveau scolaire vont penser à interpréter le travail accompli au cours de cette phase comme permettant d'avoir un retour critique sur les affirmations et les échanges faits lors de la phase 1, c'est-à-dire sur le fait que cette politique puisse ou pas réduire le nombre de filles dans les familles. Ainsi, nous supposons qu'une dernière phase va être mise en place de façon « naturelle » par des élèves de ce niveau scolaire pour valider les conjectures obtenues lors de l'approche « fréquentiste » de la problématique. Nous supposons aussi que le fait d'avoir choisi d'expérimenter cette dernière ingénierie dans deux classes ayant participé à l'ingénierie sur la dichotomie (cf. les chapitres 3, 4 et 5), les élèves vont souhaiter justifier les conjectures obtenues en proposant une approche « probabiliste » de la problématique.

3.2.7.4. Phase 4 : Approche « probabiliste » du problème : détermination de deux lois de probabilité et calculs de leurs espérances mathématiques pour obtenir la proportion de garçons parmi le nombre total d'enfants (20 minutes)

Bien que les tâches attendues dans cette phase ne soient pas explicitement demandées aux élèves, nous nous attendons à une prise d'initiative de leur part, en particulier au niveau de la Première, et qu'ils cherchent à répondre à la question implicite : « déterminer les lois de probabilité associées respectivement aux nombres de garçons et au nombre total d'enfants suivant les différents types de famille possible afin de calculer la proportion théorique de garçon par rapport au nombre total d'enfants ». Cependant, nous pouvons supposer qu'au niveau de la classe de Seconde, l'enseignante peut être amenée à guider les élèves vers ce type de questionnement.

Nous supposons que les élèves vont se référer à l'arbre de probabilité en inscrivant les probabilités de chaque branche sur le principe de construction d'un arbre pondéré, pour voir sous forme de « chemins » les parcours permettant de donner un type de famille. Les élèves ayant la connaissance du calcul de la probabilité d'un « chemin », nous pensons qu'ils ne vont montrer aucune difficulté pour déterminer les lois de probabilité correspondant à cette situation sans avoir la connaissance du calcul de probabilités conditionnelles. Nous pensons

que les élèves vont présenter ces lois sous forme d'un (ou de deux) tableau(x). En effet, une telle représentation est favorisée par l'institution, en particulier au niveau de la Première.

Nous pensons que le calcul des espérances mathématiques des deux variables aléatoires dont ils auront déterminé les lois de probabilités à l'étape précédente ne va pas poser de réelles difficultés, tant au niveau de la nécessité de les calculer afin de répondre à la question sur la proportion de garçons parmi le nombre total d'enfants, qu'au niveau du calcul de ces espérances en particulier au niveau de la Première. En effet, les élèves de Seconde, contrairement aux élèves de Première n'ont pas officiellement la notion de variable aléatoire et d'espérance mathématique au programme, l'enseignant peut être alors amené à intervenir malgré le fait que ces notions aient été abordées sur des exemples simples lors d'un TD, quelques jours avant l'expérimentation, à la demande du chercheur. Nous attendons donc à voir plus de difficultés chez les élèves de Seconde pour répondre à cette dernière partie de la phase que chez les élèves de Première.

Nous pensons aussi que suite au travail fait lors de la phase précédente avec la possibilité que la probabilité de naissance d'un garçon soit *p* au lieu de 0,5, certains élèves, en particulier de Première, vont plutôt traiter la problématique de la proportion de garçons parmi le nombre d'enfants, en prenant comme hypothèse qu'il n'y aurait pas équiprobabilité de sexe lors d'une naissance.

3.2.8 Mise en place de l'expérimentation, déroulement et analyse *a posteriori*

3.2.8.1. La classe, les supports d'observations

Nous rappelons que cette ingénierie est expérimentée dans deux classes : une Seconde et une Première Scientifique ayant un mois auparavant participé à l'ingénierie sur la dichotomie.

Comme pour l'ingénierie sur la dichotomie, les élèves sont regroupés en binôme dans une salle informatique où les ordinateurs sont équipés d'un tableur, et des logiciels *LARP* et *ALGOBOX*. Ces binômes sont identiques à ceux mis en place lors de l'ingénierie sur la dichotomie. Pour chaque binôme, nous avons un ordinateur qui reste allumé pendant les deux heures de la séance. Les élèves peuvent aussi pour de simples calculs utiliser leur calculatrice.

L'expérimentation est découpée en quatre phases durant respectivement 20, 30, 50 et 20 minutes. L'enseignant est dans la classe et a pour consigne de ne pas intervenir pendant les 2

heures où son rôle se limite à la gestion de la classe et à aider les élèves pour des problèmes de matériel si cela s'avère nécessaire. Cependant, en accord avec le chercheur, l'enseignante de Seconde peut intervenir au cours des travaux sur l'utilisation des variables aléatoires discrètes et le calcul d'espérance d'une variable aléatoire discrète. En effet, ces deux notions n'étant pas institutionnalisées au niveau de la Seconde, seul une première approche est mise en place en amont de l'expérimentation autour de quelques exemples simples sans aucune étude théorique de ces notions.

L'ensemble des quatre phases a lieu en parallèle dans les deux classes, sur un créneau horaire de 2 heures avec une pause de 10 minutes au bout de la première heure. Le chercheur est présent dans la classe de Première, où il se contente d'observer sans intervenir.

L'analyse *a posteriori* est construite à partir des enregistrements audios des quatre phases de l'expérimentation, des écrits numérisés, des documents tableurs et des algorithmes implémentés par les élèves dans les environnements numériques (*LARP* et *AlgoBox*) mis à leur disposition. L'ensemble des écrits numérisés, des travaux sur tableur et des algorithmes implémentés sont enregistrés sur une clé USB. L'apport des enregistrements audio permet de présenter les points forts observés chez les élèves.

3.2.8.2. Phase 1 : Temps de réflexion au cours d'un échange collectif sur l'évolution démographique de la population que pourrait entrainer cette politique

a) Le déroulement (20 minutes)

Au début de cette phase, l'enseignant distribue progressivement les documents aux différents binômes.

Dans un premier temps, les élèves ont le texte présentant la politique des naissances proposée par un gouvernement fictif. Les élèves lisent ce texte pendant quelques minutes à voix basse.

Puis, l'enseignant distribue la suite de la présentation de la problématique (tout couple est capable de procréer 4 fois, à chaque naissance il y a équiprobabilité entre la naissance d'un garçon et une fille, et le sexe d'un enfant est indépendant du sexe des enfants nés avant) en précisant que ce complément permet de simplifier l'étude du problème sans pour autant en dénaturer ses objectifs.

Puis, il termine en projetant les questions du sondage d'opinion au tableau numérique. Il accompagne cette projection d'un commentaire oral : « *vous avez droit à une quinzaine de minutes d'échange collectif pour proposer des réflexions quant à la remarque des membres du gouvernement fictif. Je précise que je ne suis qu'un observateur* ».

Nous observons que l'ensemble des binômes répond à cette série de questions en s'aidant du « papier-crayon ». En effet, nous constatons que tous les binômes dessinent des arbres de probabilités quel que soit le niveau scolaire observé. Cela leur prend en moyenne cinq minutes. Nous observons que l'enseignant reste passif pendant ces premières minutes de réflexion de la part des élèves. Nous constatons qu'au niveau de la Première les binômes ne prennent pas le temps à mettre en place un réel débat sur les conséquences qu'il pourrait y avoir sur l'évolution de la population si une telle politique était mise en place. En effet, nous observons que les élèves se contentent de cocher certaines cases des différentes affirmations après un échange dans le binôme. En revanche, nous constatons qu'au niveau de la Seconde un véritable débat a lieu autour de ces affirmations dans la classe. Ce début dure une dizaine de minutes.

Nous observons qu'une fois que les arbres sont dessinés, seulement 70% des binômes de Seconde pensent à inscrire les probabilités sur l'ensemble des branches de l'arbre tandis que l'ensemble des binômes de Première ont le réflexe de le faire. Nous observons aussi que les élèves utilisent l'arbre pour donner les types de famille possibles ayant au maximum quatre enfants. Nous constatons que le travail fait pour obtenir les différents types de famille ne prend seulement que quatre à cinq minutes dans les deux classes.

b) Analyse a posteriori

N'ayant pas observé de grandes différences entre la classe de Seconde et celle de Première au niveau des rendus, nous choisissons de ne pas distinguer ces deux classes pour la description de l'analyse *a posteriori* de cette phase.

Quel que soit la classe, nous observons que l'ensemble des binômes répond de façon réfléchie aux différentes affirmations données. Nous observons que dans chaque binôme les élèves modélisent les conditions sur les familles à partir d'un arbre de probabilités, ce qui leur permet d'obtenir les différents types de famille. En revanche à ce stade de la phase, les arbres ne sont pas nécessairement accompagnés des valeurs des probabilités de chaque branche, en

particulier au niveau de la Seconde. De plus, nous observons que les élèves choisissent de nommer les événements « Garçon » et « Fille » par les lettres F et G. Cependant, au niveau de la Première, nous observons que 10% des binômes définissent l'événement « Fille » comme événement contraire de celui des garçons et écrivent $\overline{G}$ pour nommer cet événement. Nous observons qu'une majorité d'élèves (90% des binômes de Seconde et 75% de ceux de Première) répondent pour chacune des affirmations : « ne se prononce ». En revanche, 5% des binômes de Seconde et 20% de ceux de Première considèrent que cette politique a pour conséquence qu'il y aura plutôt plus de filles que de garçons et que quoi qu'il arrive la population va augmenter. Une très forte majorité des élèves de Seconde qui répondent cela affirme que ceci est dû au fait qu'il a plus de types de famille à plus d'un enfant que de famille à un enfant. En revanche, les élèves de Première qui ont donné ces affirmations se contentent de dire que celles-ci peuvent être fausses et que par conséquent, il faut procéder à une étude des probabilités des différents types de famille.

95% des binômes constatent à l'aide de l'arbre (Fig. 187), qu'il y a cinq types de familles : G, FG, FFG, FFFG et FFFF. Seul un binôme de Seconde propose comme réponse : G, FG, FFG et FFFG. En effet, ce binôme rend comme arbre de probabilité, un arbre présentant pour le cas où il y aurait un quatrième enfant, une seule branche à ce niveau de l'arbre (Fig. 188).

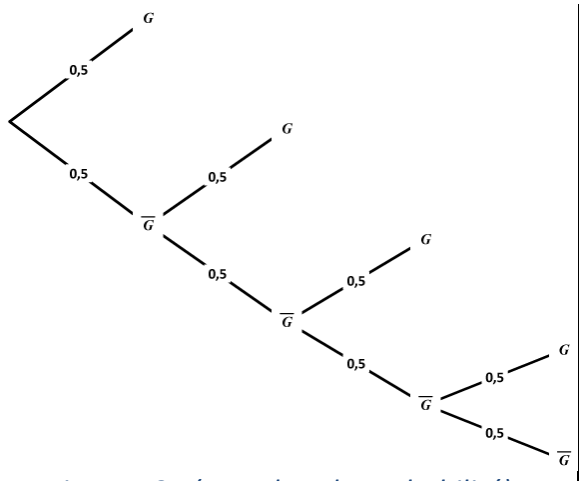


Figure 187 (Un arbre de probabilité)

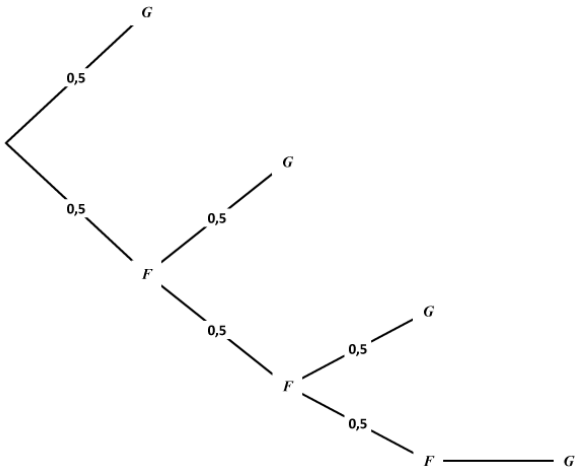


Figure 188 (Un arbre de probabilité)

Nous constatons aussi que ce binôme inscrit les valeurs des probabilités des branches, exceptées sur dernière branche correspondant à la naissance d'un quatrième enfant. Nous verrons lors de l'analyse de la détermination des lois de probabilité (cf. la phase 4), que ce binôme considère que la probabilité de cette branche est 1.

Pour finir, nous observons que le temps prévu pour cette phase est légèrement surestimé. En effet, au bout d'une quinzaine de minutes les deux classes abordent la deuxième phase, même si un groupe de Seconde semble éprouver quelques difficultés dans le choix de ces réponses pour les affirmations. Mais ce cas est dû en partie au manque d'investissement dans cette phase de l'expérimentation des deux élèves et par conséquent ne peut être pris en compte pour notre analyse.

Nous pouvons retenir que quel que soit le niveau scolaire, les élèves se sont mis dans une démarche de type « probabiliste », traduite par la représentation d'un arbre, pour répondre au sondage même si aucun calcul de probabilités et a fortiori d'espérances n'ont été proposés par les élèves.

3.2.8.3. Phase 2 : Approche « fréquentiste » de la proportion de garçons. Simulation à l'aide d'un tableur sur une population de 1 000 familles, puis de 10 000 familles

a) Le déroulement (30 minutes)

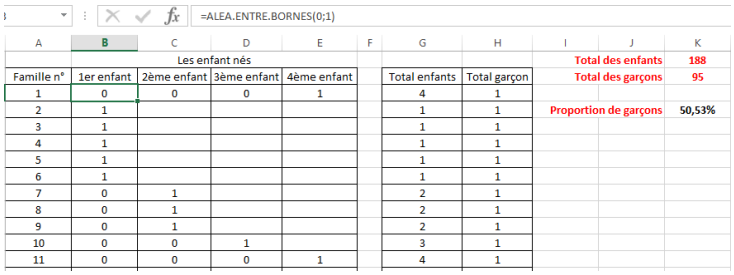
Pendant cette deuxième phase, nous observons une totale uniformité dans le travail accompli et rendu par les différents groupes tant au niveau Seconde, qu'au niveau Première. Dès le début de cette seconde phase, les binômes ouvrent tous un fichier tableur. Nous n'avons comme retour qu'un seul type de présentation du tableau correspondant à ce qui avait prévu lors de l'analyse *a priori*. Nous observons qu'aucun binôme ne fait appel aux enseignants. Nous observons aussi que le temps prévu pour cette deuxième phase est respecté par les deux classes.

b) Analyse a posteriori

Nous observons que les élèves n'éprouvent aucune difficulté pour manipuler les formules du tableur qu'ils doivent utiliser au cours de cette simulation. Au regard du travail accompli, le fait aussi d'utiliser le principe du « copier-coller » semble faciliter le travail des élèves qui n'ont pas besoin d'utiliser une boucle « POUR » contrairement à la phase suivante.

Ainsi, nous avons pour la plupart des travaux rendus par les binômes, un tableau où les élèves présentent sous forme de colonnes les naissances des enfants, le nombre d'enfants par famille et le nombre de garçon par famille (Fig. 189 et 190). Pour simuler une naissance, les binômes utilisent en grande majorité la formule =ALEA.ENTRE.BORNES(0 ;1), où le 0

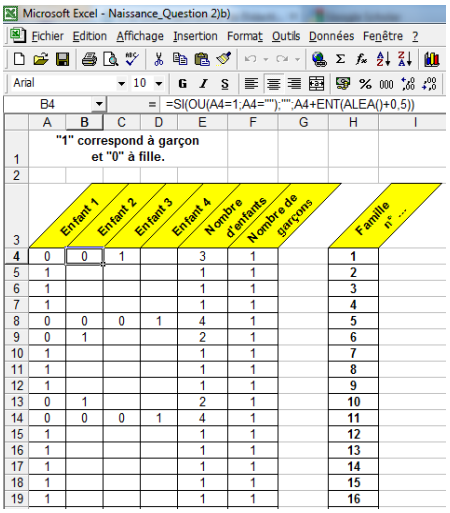
représente une fille et le 1 un garçon (Fig. 189) et seul 10% des binômes utilisent la formule =ENT(ALEA()+0,5). On constate aussi qu'un tiers des binômes ajoutent une colonne leur permettant de vérifier le nombre de familles étudiées. Nous observons que pour le calcul du nombre total d'enfants dans une famille, ils utilisent la formule =NB(Bnuméro de ligne ;Enuméro de ligne). De même, nous observons que pour compléter la colonne du nombre total de garçons, les binômes exploitent encore une fonctionnalité offerte par le tableur qu'est la formule =NB.SI(Bnuméro de ligne ;Enuméro de ligne ;1), où 1 représente la naissance d'un garçon. Pour finir, nous observons que les élèves déterminent le nombre total d'enfants et le nombre total de garçons en utilisant la fonction =SOMME(nombre1 ;nombre100). Ces deux résultats sont utilisés pour le calcul de la proportion de garçons. Le résultat de cette proportion est donné pour un quart des binômes de Première en pourcentage. En revanche, nous ne pouvons pas savoir au vu des travaux rendus et des enregistrements audios, si les binômes ont utilisé la touche F9 afin d'obtenir plusieurs échantillons de 1 000 familles. Nous n'observons pas non plus de difficultés particulières à générer des échantillons de 10 000 familles. Les élèves se contenteront de signaler que c'était « *fastidieux* » de faire des « copier-coller » correspondant aux 10 000 familles.



Famille n°	1er enfant	2ème enfant	3ème enfant	4ème enfant	Total enfants	Total garçon
	Les enfant nés					
1	0	0	0	1	4	1
2	1				1	1
3	1				1	1
4	1				1	1
5	1				1	1
6	1				1	1
7	0	1			2	1
8	0	1			2	1
9	0	1			2	1
10	0	0	1		3	1
11	0	0	0	1	4	1

Total des enfants 188
Total des garçons 95
Proportion de garçons 50,53%

Figure 189 (Le tableur pour représenter les naissances des enfants, le nombre d'enfants par famille et le nombre de garçon par famille)



"1" correspond à garçon et "0" à fille.

Enfant 1	Enfant 2	Enfant 3	Enfant 4	Nombre d'enfants	Nombre de garçons	Famille n° ...
0	0	1		3	1	1
1				1	1	2
1				1	1	3
1				1	1	4
0	0	0	1	4	1	5
0	1			2	1	6
1				1	1	7
1				1	1	8
1				1	1	9
0	1			2	1	10
0	0	0	1	4	1	11
1				1	1	12
1				1	1	13
1				1	1	14
1				1	1	15
1				1	1	16

Figure 190 (Le tableur pour représenter les naissances des enfants, le nombre d'enfants par famille et le nombre de garçon par famille)

Lors d'échanges après l'expérimentation, certains élèves ont signalé que le travail fait avec le tableur ressemblait à celui fait lors de la construction de l'arbre pondéré. Et, nous observons que certains font allusion aux limites que pourraient avoir cet environnement numérique pour des nombres maximums d'enfants par famille plus élevés ainsi que pour une étude sur une population de plusieurs milliers de familles. Ces mêmes élèves compléteront ces propos à la fin de la phase 3 après le travail dans les environnements *LARP* ou *ALGOBOX*.

3.2.8.4. Phase 3 : Organigramme – Modélisation – Approche « fréquentiste » de la proportion de garçons – Simulation en langage pseudo-code

a) Le déroulement (50 minutes)

Au cours de cette troisième phase, nous observons au cours de la deuxième partie de cette phase beaucoup de plus de diversité et de difficultés chez les binômes que dans les phases précédentes. Cependant, les déroulements sont très proches quel que soit la classe observée.

Dans la première partie de cette phase, la construction des organigrammes est semblable chez presque tous les binômes que ce soit en Seconde ou en Première, et nous observons qu'ils transposent le modèle de l'arbre à l'organigramme. Cette construction se fait dans l'environnement LARP. En effet, le fait d'avoir utilisé cet environnement dans l'ingénierie sur la dichotomie permet de faire que les élèves aient le réflexe de « réexploiter » les fonctionnalités de cet environnement pour la construction d'un organigramme. Cette construction demande une dizaine de minutes aux élèves.

Lors du travail de la deuxième partie de cette phase qui se situe dans l'environnement ALGOBOX, nous observons ici aussi que les élèves n'éprouvent pas de réelles difficultés à gérer les variables informatiques. Nous supposons encore que cela vient du fait que les deux classes ont participé au cours du mois précédent à l'ingénierie sur la dichotomie. En revanche les démarches mises en place pour répondre à la problématique de modélisation montrent des différences majeures entre les élèves.

En effet, chez 80% des binômes de Première, une partie du temps est consacrée à la construction d'un modèle en lien avec les habitudes pratiquées lors des travaux faits dans le domaine des probabilités, c'est-à-dire des lancers de pièces ou des tirages de boules dans une urne. Ici, les élèves n'ayant que deux issues possibles à chaque naissance, nous observons

qu'ils choisissent de construire un modèle basé sur le lancement d'une pièce de monnaie. Ce choix de modèle et la construction de ce modèle prend une vingtaine de minutes chez les élèves qui le mettent en place. Ce qui a pour conséquence de leur laisser peu de temps pour répondre à la totalité des questions de la phase. En revanche, ceux qui restent sur le modèle initial autour des naissances sans se référer aux lancers d'une pièce (la totalité des élèves de Seconde et 20% des élèves de Première), arrivent à mieux gérer le temps pour répondre aux attentes de cette deuxième partie de la phase.

Nous observons qu'une minorité des binômes de Seconde (environ 15%) a le réflexe de faire un retour sur le débat vécu au début de l'expérimentation au vu des résultats obtenus au cours de cette troisième phase. Ceux qui le font, ne prennent que cinq minutes pour le faire à la fin de cette phase. Mais à ce niveau scolaire, cela ne débouche pas de façon « naturelle » sur la phase suivante (Cf. phase 4). En revanche, les élèves de Première prennent l'initiative de ce retour qui dure aussi cinq minutes à la fin de cette phase et celui-ci se poursuit pendant la phase suivante sans l'intervention de l'enseignant.

b) Analyse a posteriori

Pour l'analyse de cette dernière phase, nous souhaitons présenter deux sections, une première pour la construction de l'organigramme et une deuxième pour le travail fait autour de la simulation d'une politique des naissances dans l'environnement AlgoBox.

- **Construction de l'organigramme**

La construction des organigrammes est semblable chez presque tous les binômes que ce soit en Seconde ou en Première, et nous observons qu'ils transposent le modèle de l'arbre à l'organigramme.

Ainsi, nous constatons que les élèves proposent des organigrammes mettant en place une série de « SI … ALORS … SINON ». Nous observons aussi qu'ils utilisent une fonctionnalité qu'offre l'environnement LARP, à savoir que le sexe de l'enfant est donné par le choix aléatoire entre les nombres 0 et 1, qu'ils représentent par la fonction « ALÉATOIRE(0,1) » et qu'ils stockent dans une variable notée « SEXE » (Fig. 191 – Extrait d'une copie).

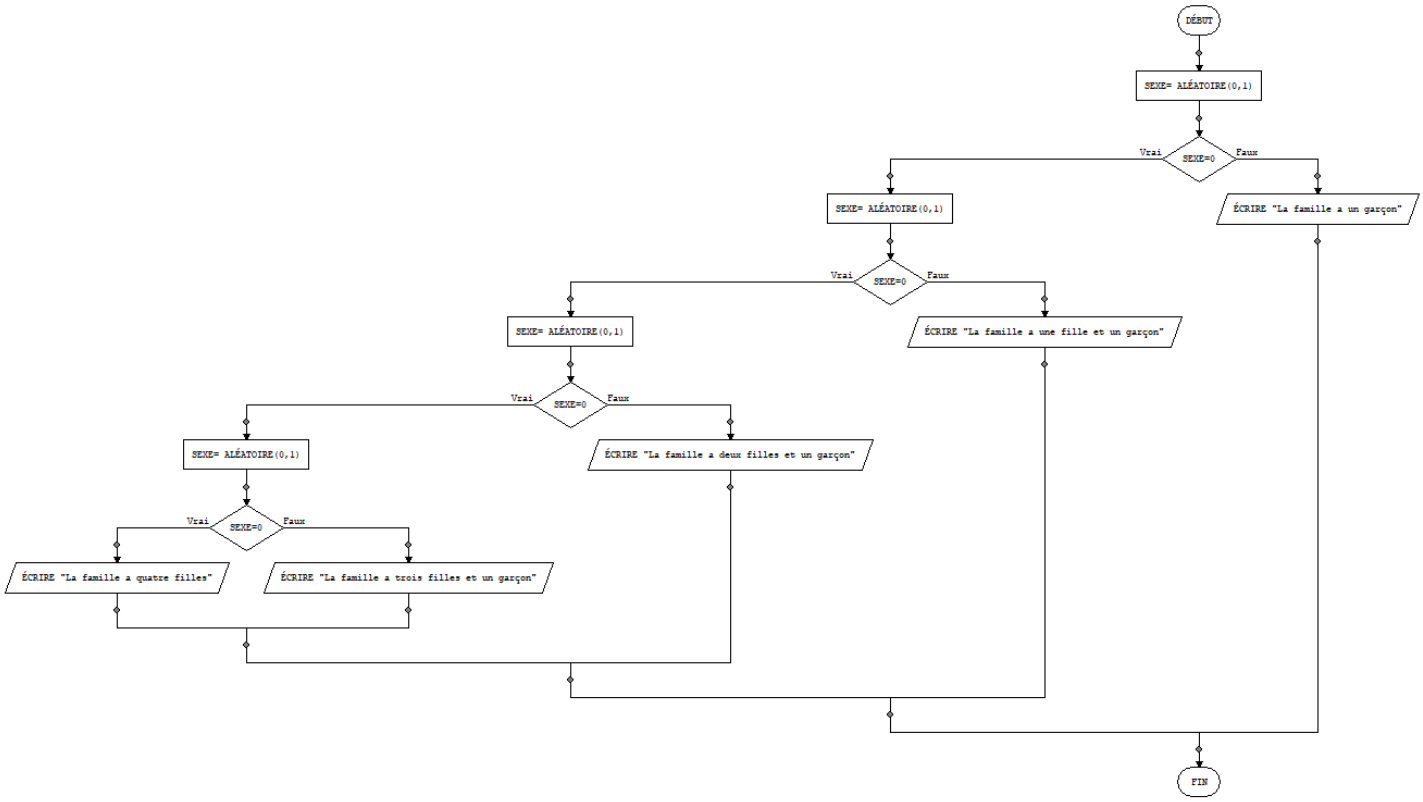


Figure 191 (Extrait d'une copie : Un organigramme de la politique des naissances où est mis en place une série de « SI … ALORS … SINON »)

Cependant, après échange entre trois binômes de Première, lors de cette construction des organigrammes, nous constatons que ces derniers émettent l'hypothèse qu'un tel organigramme serait « *difficile à construire si les familles avaient par exemple au maximum 10 enfants* » (Transcription d'une remarque d'un élève). Ainsi, un des trois binômes proposent de construire un organigramme où le nombre maximal d'enfants serait représenter par une boucle « TantQue » (Fig. 192).

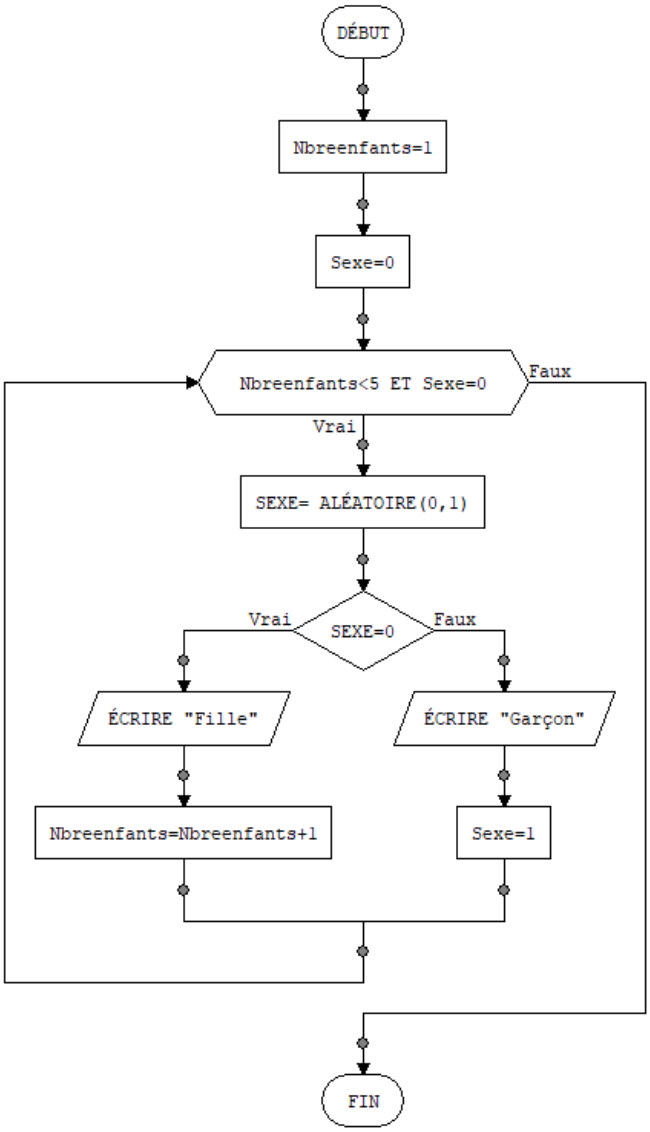


Figure 192 (Extrait d'une copie : Un organigramme de la politique des naissances utilisant une boucle « TantQue »)

Nous avons aussi la confirmation lors de ces constructions d'algorithmes que ces élèves qui ont suivi l'ingénierie précédente, ne montrent plus de difficultés sur l'organisation et la gestion des variables informatiques ainsi que sur l'utilisation des structures de boucles et des instructions conditionnelles.

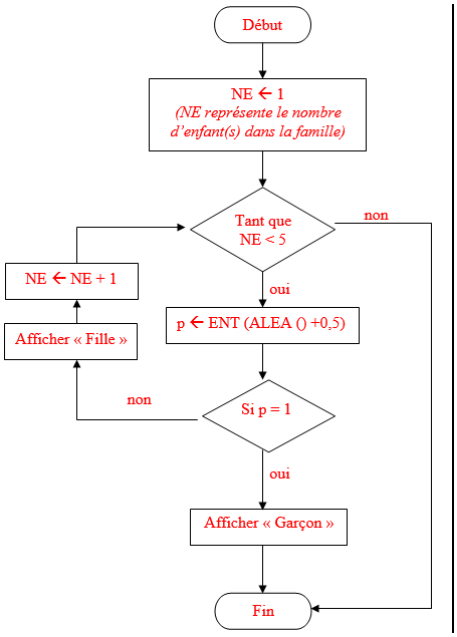
- **Travail dans l'environnement AlgoBox**

Pour l'analyse de cette deuxième partie de la phase 3, nous souhaitons différencier les deux niveaux scolaires où l'expérimentation a eu lieu. En effet, nous observons de réelles différences d'approche sur la deuxième partie de cette phase entre les élèves de la classe de Seconde et ceux de la classe de Première que nous allons présenter ci-dessous.

La classe de Seconde

Nous observons que les élèves de Seconde travaillent sur un modèle représentant la politique des naissances, où pour chaque naissance les issues sont déterminées par les sexes des enfants et non des lancers d'une pièce comme nous le verrons dans la section suivante avec les élèves de Première. De plus, nous observons que l'ensemble de la classe propose dans un premier temps un algorithme correspondant à l'évolution d'une famille avant de proposer un algorithme pouvant permettre l'étude de plusieurs familles.

Les algorithmes proposés par les binômes montrent une évolution dans l'apprentissage et un détachement des élèves par rapport au choix fait pour certains au cours de la première partie de cette phase de transposer la technique de l'arbre pondéré à la conception d'un algorithme construit sur une succession de « SI…ALORS…SINON ». En effet, au cours de cette deuxième partie de la phase, 85% des algorithmes proposés sont construits sur une structure « TANTQUE » où la condition de sortie correspond au nombre maximum d'enfants dans une famille (ici quatre), que les élèves traduisent par exemple par « NE<5 » ou « nombre_enfants<5 » (Fig. 193), ce qui ne correspond pas à ce qui était envisageable après l'observation des constructions d'organigrammes, où un nombre important de binômes avait travaillé sur une succession de « SI…ALORS…SINON ».




```
1 VARIABLES
2       enfants_famille EST_DU_TYPE NOMBRE
3       p EST_DU_TYPE NOMBRE
4 DEBUT_ALGORITHME
5       enfants_famille PREND_LA_VALEUR 1
6       TANT_QUE (enfants_famille<5) FAIRE
7               DEBUT_TANT_QUE
8               p PREND_LA_VALEUR ALGOBOX_ALEA_ENT(0,1)
9               SI (p==1) ALORS
10                      DEBUT_SI
11                      AFFICHER "Garçon"
12                      enfants_famille PREND_LA_VALEUR 5
13                      FIN_SI
14                      SINON
15                              DEBUT_SINON
16                              AFFICHER "Fille"
17                              enfants_famille PREND_LA_VALEUR enfants_famille+1
18                              FIN_SINON
19      FIN_TANT_QUE
20 FIN_ALGORITHME
```

Figure 193 (Extraits de copies présentant des algorithmes de la politique des naissances)

Nous observons aussi que parmi ces 85% des binômes, nous en avons 15% qui proposent une approche « spatiale » de l'algorithme (figure de gauche ci-dessus) correspondant à la situation décrite par la politique.

Toujours parmi ces 85% de binômes, nous observons que 70% proposent une approche « textuelle » de l'algorithme qu'ils implémentent dans l'environnement ALGOBOX afin de le tester sur une famille (figure de droite ci-dessus).

Nous observons qu'ensuite, ces 85% de binômes reprennent leur algorithme obtenu pour une famille (ceux qui ont donné un premier algorithme « spatial » le transpose en un algorithme « textuel ») et proposent une modification permettant de répondre à une simulation de la politique pour un nombre quelconque de familles et donnant la fréquence des garçons par rapport au nombre total de naissances pour l'ensemble des familles. Nous avons ainsi, pour le binôme qui rend l'algorithme « textuel » vu ci-dessus, le nouvel algorithme ci-dessous qui montre que ce binôme a bien pris en compte des attentes des auteurs de l'expérimentation. Nous observons que le binôme marque en jaune (fig. 194) les apports et changements qu'il est nécessaire de faire pour procéder à cette « généralisation ».

```
1 VARIABLES
2       enfants_famille EST_DU_TYPE NOMBRE
3       p EST_DU_TYPE NOMBRE
4       nombre_familles EST_DU_TYPE NOMBRE
5       nombre_garçons EST_DU_TYPE NOMBRE
6       nombre_total_enfants EST_DU_TYPE NOMBRE
7       f EST_DU_TYPE NOMBRE
8       k EST_DU_TYPE NOMBRE
9 DEBUT_ALGORITHME
10      AFFICHER "Donner le nombre de familles concernées. "
11      LIRE nombre_familles
12      nombre_garçons PREND_LA_VALEUR 0
13      nombre_total_enfants PREND_LA_VALEUR 0
14      POUR k ALLANT_DE 1 A nombre_familles
15              DEBUT_POUR
16              enfants_famille PREND_LA_VALEUR 1
17              TANT_QUE (enfants_famille<5) FAIRE
18                      DEBUT_TANT_QUE
19                      nombre_total_enfants PREND_LA_VALEUR nombre_total_enfants+1
20                      p PREND_LA_VALEUR ALGOBOX_ALEA_ENT(0,1)
21                      SI (p==1) ALORS
22                              DEBUT_SI
23                              nombre_garçons PREND_LA_VALEUR nombre_garçons+1
24                              enfants_famille PREND_LA_VALEUR 5
25                              FIN_SI
26                              SINON
27                                      DEBUT_SINON
28                                      enfants_famille PREND_LA_VALEUR enfants_famille+1
29                                      FIN_SINON
30                      FIN_TANT_QUE
31              FIN_POUR
32      f PREND_LA_VALEUR nombre_garçons/nombre_total_enfants
33      AFFICHER "La fréquence des garçons dans les "
34      AFFICHER nombre_familles
35      AFFICHER " familles est de : "
36      AFFICHER f
37 FIN_ALGORITHME
```

Figure 194 (Extrait d'une copie sur la politique des naissances)

Nous observons qu'ici les élèves restent avec un nombre maximal d'enfants fixé à quatre et qu'ils tiennent compte de l'initialisation du nombre total de garçons (nombre_garçons) et du nombre total d'enfants (nombre_total_enfants) qu'ils initialisent en dehors des boucles. De même, nous observons que l'initialisation du nombre d'enfants (enfants_famille) est aussi correctement faite et se situe dans la boucle « POUR », avant que ne débute la boucle « TANTQUE ».

Nous observons que le calcul de la fréquence des garçons par rapport au nombre total d'enfants nés se trouvent en dehors de la boucle « Pour » qui permet de déterminer la typologie de l'ensemble des familles étudiées. Le calcul de cette fréquence ne semble pas poser de difficulté et elle est donnée sous forme d'un décimal et non d'un pourcentage.

Concernant les 15% de binômes qui sont restés dans le schéma d'une succession de « SI…ALORS…QUE », nous obtenons des algorithmes du type de ceux que nous pouvons voir dans les figures qui suivent. Malgré la « lourdeur » de ces algorithmes, nous observons ici encore une assez bonne maitrise du langage pseudo-code, en particulier pour la construction des instructions conditionnelles. En revanche, nous observons que la gestion des variables peut poser des difficultés chez certains élèves (5% de ces binômes) qui ne proposent qu'une seule fois d'affecter à la variable représentant le sexe de l'enfant le fait que celle-ci est un nombre aléatoire pouvant valoir 0 ou 1 (figure 195 de gauche). Les autres binômes tiennent comptent de cette problématique et pour chaque instruction conditionnelle, ils génèrent un nouveau nombre aléatoire 0 ou 1 permettant d'avoir le sexe du nouvel enfant si celui existe (figure 195 de droite).

Nous observons aussi que le binôme de la figure de droite rend un algorithme complet affichant aussi la proportion de garçons parmi la totalité des enfants nés pour un nombre fixé de familles, dans le cas où il y a au maximum quatre enfants par famille.

En revanche, nous n'avons pas de retour dans le cas où il n'y aurait pas d'équiprobabilité entre les naissances d'un garçon et d'une fille. Nous pensons que cela est dû entre autres au fait que les élèves de Seconde n'ont pas la connaissance du principe d'une épreuve de Bernoulli, qui ne fait pas partie du programme, même si à la demande du chercheur, l'enseignante a préparé ses élèves en amont de la séance autour de quelques exercices portant sur ce concept.

Nous constatons aussi que les élèves de Seconde, bien qu'ils n'aient pas traité le cas où le nombre maximal d'enfants pourraient être supérieur à quatre enfants, ont pour une très grande majorité (environ 95% de l'ensemble des binômes) conscience du choix des variables informatiques et du rôle de chacune de ces variables ainsi que de l'initialisation de ces variables quand cela s'avère nécessaire.

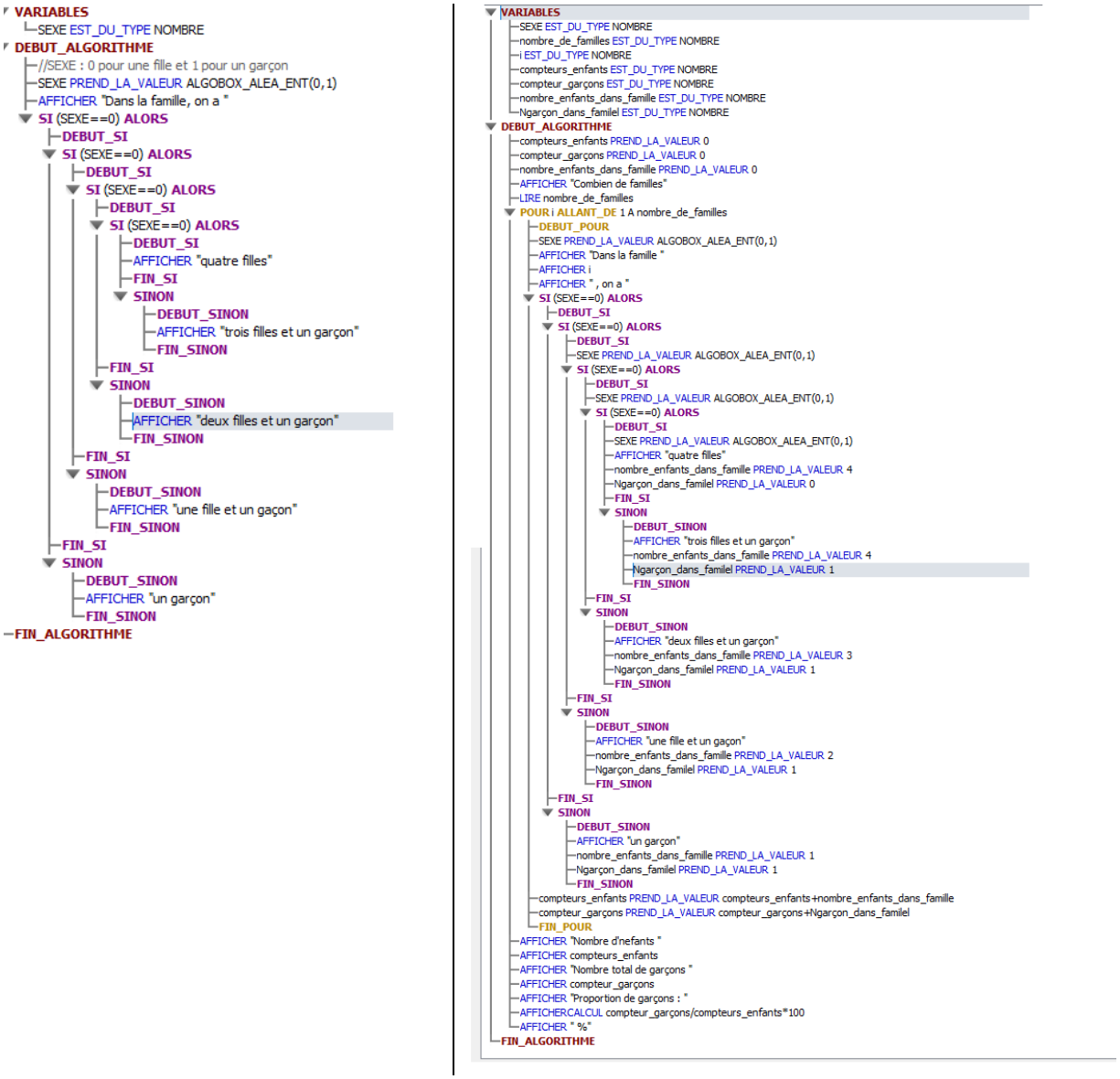


Figure 195 (Deux extraits de copies sur la politique des naissances)

Nous observons aussi que le temps total prévu initialement de 50 minutes pour cette phase n'a probablement pas permis que les élèves aillent jusqu'au bout des attentes de la phase. En effet, la problématique sur la non équiprobabilité de naissance d'un garçon ou d'une fille n'a pu être réellement abordée par les élèves à la fin de cette dernière phase.

La classe de Première

Tout d'abord, nous observons que pour cette deuxième partie de la phase 3, les élèves de Première ne propose plus une série de « SI…ALORS…SINON » pour répondre à la problématique de la situation aléatoire. Nous supposons que l'échange qui a terminé la fin de la première partie de la phase est en partie responsable de cette évolution chez les élèves. En effet, au cours de cette deuxième partie de la phase, les élèves ne proposent que des structures de type « TANTQUE » pour répondre à la problématique du nombre maximal d'enfants dans une famille.

Conformément à ce que nous avions supposé lors de l'analyse *a priori*, une majorité des binômes (80%) de Première déterminent un modèle qu'ils pensent représentatif de la politique des naissances construit sur un jeu de lancers d'une pièce. Lors de l'entretien avec l'enseignant après la séance, nous apprenons un fait qui n'a pas été prévu lors de la préparation de l'expérimentation. En effet, la veille de l'expérimentation, celui-ci a demandé à ces élèves de traiter la phase 3 de l'expérimentation en commençant par construire un modèle représentatif de la situation de la politique des naissances à partir d'un jeu de lancers successifs d'une pièce de monnaie équilibrée ou pas (fig. 196). Lors de cet entretien, se déroulant juste après la séance, l'enseignant ne souhaite pas justifier auprès du chercheur ce choix de construire un modèle basé sur des lancers d'une pièce. Cependant, cette possibilité a été envisagée par le chercheur lors de l'analyse *a priori* comme nous avons pu le voir plus haut. Ainsi, nous observons que les élèves de Première proposent des modèles comme ceux qui suivent dans les figures ci-dessous sans pour autant donner de compléments d'explications justifiant la construction de tel ou tel modèle, en particulier pour celui de la figure de droite. Le chercheur étant présent dans la salle peut observer ce travail fait par les élèves et l'absence de justifications tant orale qu'écrite.

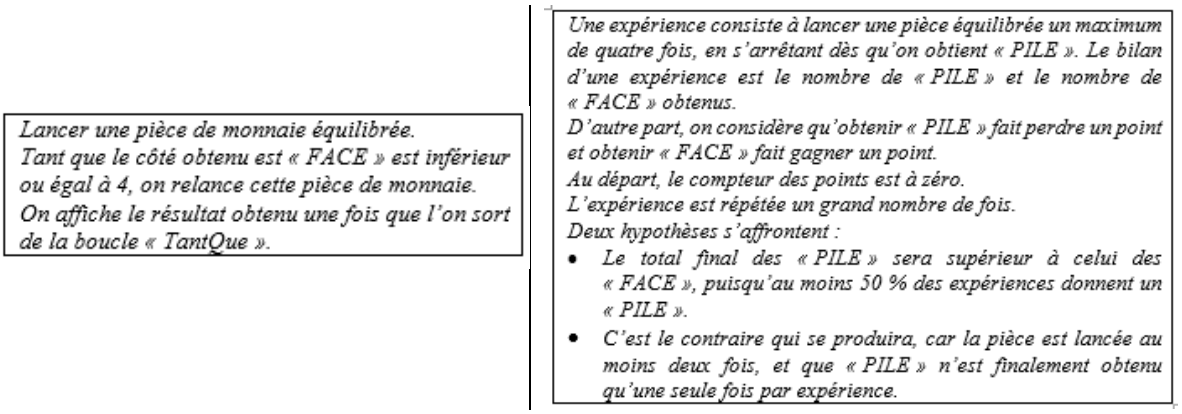


Figure 196 (Un modèle représentatif de la situation de la politique des naissances à partir d'un jeu de lancers successifs d'une pièce de monnaie équilibrée ou pas)

Nous observons que les binômes qui choisissent le modèle de la figure de droite donnée ci-dessus proposent un algorithme (Fig. 197) construit avec une boucle « TantQue » qu'ils écrivent en langage pseudo-code et qu'ils implémentent ensuite dans l'environnement ALGOBOX. Ces élèves montrent une maitrise de l'initialisation, de la structure itérative « TANTQUE » ainsi que de l'instruction conditionnelle. De plus, nous observons que ces élèves (60% des binômes de Première) distinguent la variable représentant le nombre de FACE possible, c'est-à-dire 4 au maximum, de la variable représentant le résultat du lancer du dé.

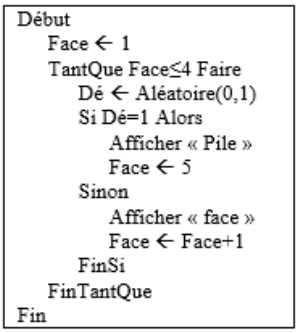


Figure 197 (Extrait d'une copie où est proposé un algorithme sur le jeu de lancers successifs d'une pièce de monnaie équilibrée avec l'utilisation d'une boucle « TantQue »)

Les 20% de binômes qui proposent un modèle du type celui décrit par la figure 196 (celle de droite), montrent des approches différentes pour construire un algorithme représentant ce modèle. En effet, certains rendent des algorithmes écrits sous forme « textuelle » (figure de gauche ci-dessous) et reprenant le modèle de la figure 196 (celle de droite), d'autres proposent une représentation « spatiale » (figure de droite ci-dessous) de l'algorithme proche du modèle proposé à la figure 196 (celle de droite).

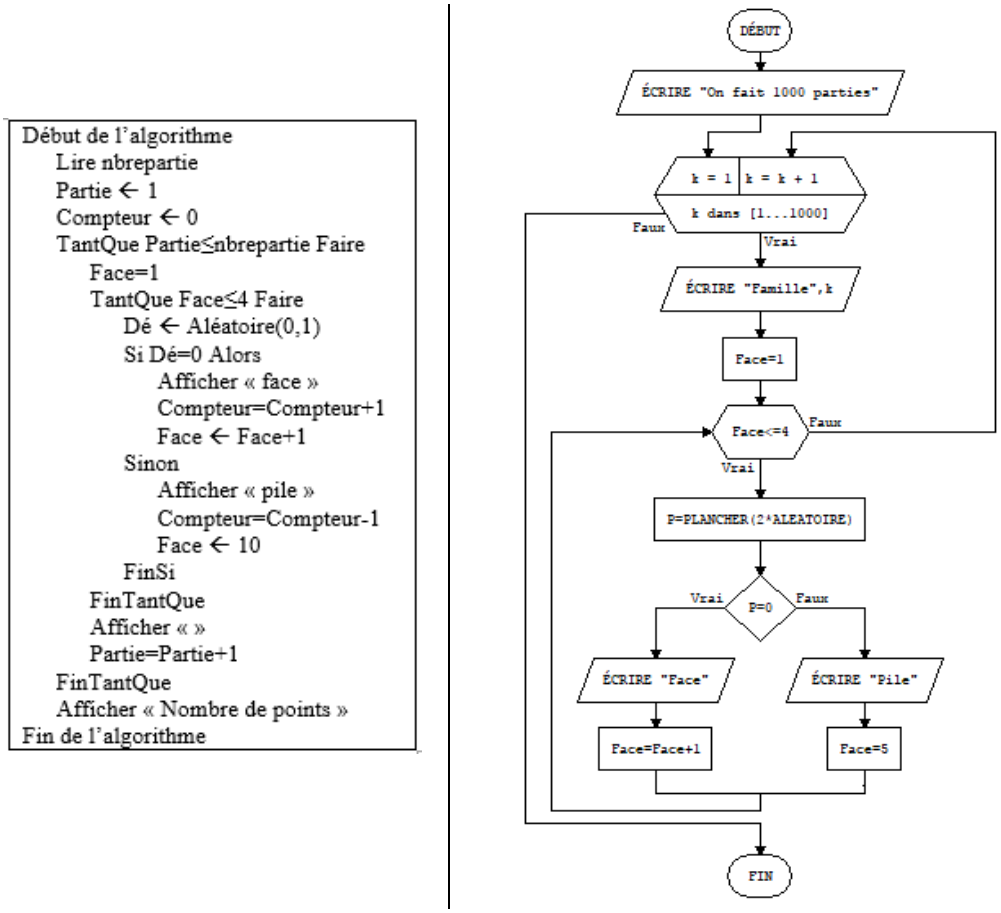


Figure 198 (Extraits de copies représentant des algorithmes aux formats « textuel » ou « spatial » du jeu du lancers successifs d'une pièce et de la politique des naissances)

Nous observons chez ces élèves de Première une bonne maitrise des représentations « textuelle » et « spatiale » des algorithmes. Nous observons aussi que le choix de la structure « POUR » (Figure de droite) est favorisé quand le nombre de fois que l'algorithme doit « tourner » est fixé (par exemple pour 100 parties). En revanche, quand ce nombre peut être un paramètre qui peut varier lors d'une utilisation de l'algorithme, nous constatons que les élèves favorisent le choix d'une boucle « TANTQUE » (figure de gauche). Bien que l'ensemble des binômes travaillant sur le modèle de la figure 196 (celle de droite) représentant un échantillon restreint, nous constatons une certaine régularité quant à ces choix faits sur les boucles « POUR » et « TANTQUE ». Nous constatons aussi que très peu de binômes proposent de tenir compte de la variable « compteur de points ».

Nous pensons que le fait que cette ingénierie étant faite dans une classe de Première où les élèves ont participé à l'ingénierie sur la dichotomie explique le fait que nous n'avons pas de retour d'algorithme(s) où les initialisations de variables seraient incorrectes ainsi qu'une utilisation des boucles erronées.

Parmi les 20% des binômes de Première qui proposent un algorithme représentatif du modèle initial, c'est-à-dire travaillant directement sur les familles, nous avons de nombreux binômes (environ 12% des binômes de ces binômes) qui construisent à partir de l'environnement ALGOBOX un algorithme, contrairement à ce que nous observons chez les élèves qui passent par le jeu des lancers d'une pièce. En effet, bien que ces derniers rendent aussi des algorithmes construits sous ALGOBOX, un certain nombre de binômes (1/4 de ces binômes) travaillent dans un premier temps la conception de l'algorithme soit sous une forme « textuelle », soit sous une forme « spatiale ».

Que ce soit dans le cas des lancers d'une pièce ou la politique elle-même, nous observons que la modification de l'algorithme où on suppose que la probabilité d'avoir un garçon lors d'une naissance soit un nombre *p* compris entre 0 et 1 ne semble pas poser de difficultés particulières pour les élèves. En effet, dans l'environnement ALGOBOX, nous observons que 90% des binômes rendent des algorithmes du type de la figure de gauche ci-après ou un algorithme écrit en langage naturel équivalent (figure 199 de droite ci-après).

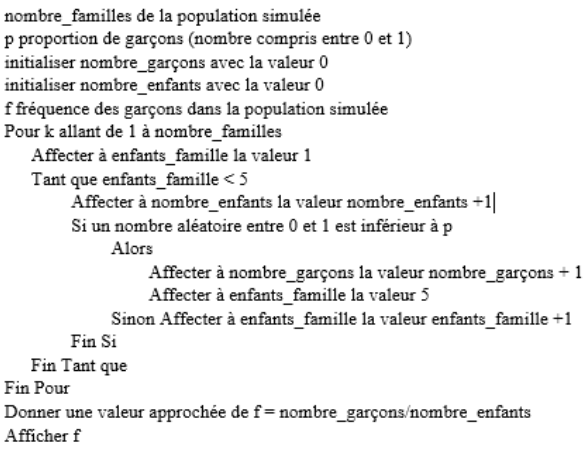


Figure 199 (Extraits de copies sur la politique des naissances)

De même, nous observons que le concept de nombre aléatoire est compris tant d'un point de vue mathématique qu'algorithmique.

Pour finir, nous observons que 55% des binômes (les autres ne répondent pas à cette question, faute de temps) proposent un dernier algorithme complet permettant d'envisager

la prise en compte du nombre maximum d'enfants par famille, du nombre de familles étudiées et de la probabilité de naissance d'un garçon (Fig. 200).

```
VARIABLES
  Nombrefamille EST_DU_TYPE NOMBRE
  k EST_DU_TYPE NOMBRE
  enfantsfamille EST_DU_TYPE NOMBRE
  Nombregarçons EST_DU_TYPE NOMBRE
  f EST_DU_TYPE NOMBRE
  Nbreenfants EST_DU_TYPE NOMBRE
  progarçon EST_DU_TYPE NOMBRE
  maxenfantsparfamille EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
  AFFICHER "Maximum d'enfants pour une famille"
  LIRE maxenfantsparfamille
  AFFICHER "Nombre total de familles étudiées"
  LIRE Nombrefamille
  AFFICHER "Probabilité de naissance d'un garçon (réel dans [0;1])"
  LIRE progarçon
  Nbreenfants PREND_LA_VALEUR 0
  Nombregarçons PREND_LA_VALEUR 0
  POUR k ALLANT_DE 1 A Nombrefamille
    DEBUT_POUR
    enfantsfamille PREND_LA_VALEUR 1
    TANT_QUE (Nbreenfants<=maxenfantsparfamille) FAIRE
      DEBUT_TANT_QUE
      Nbreenfants PREND_LA_VALEUR Nbreenfants+1
      SI (floor(random()+progarçon)<progarçon) ALORS
        DEBUT_SI
        Nombregarçons PREND_LA_VALEUR Nombregarçons+1
        enfantsfamille PREND_LA_VALEUR Nbreenfants+1
        FIN_SI
        SINON
          DEBUT_SINON
          enfantsfamille PREND_LA_VALEUR enfantsfamille+1
          FIN_SINON
      FIN_TANT_QUE
    FIN_POUR
  f PREND_LA_VALEUR Nombregarçons/Nbreenfants
  AFFICHER "La fréquence des garçons est : "
  AFFICHER f
FIN_ALGORITHME
```

Figure 200 (Extrait d'une copie où l'algorithme « complet » permet d'envisager la prise en compte du nombre maximum d'enfants par famille, du nombre de familles étudiées et de la probabilité de naissance d'un garçon)

Contrairement aux élèves de Seconde, nous observons que le temps imparti pour cette dernière phase est suffisant pour des élèves de Première, en particulier si l'expérimentation suit une autre expérimentation sur l'algorithmique, même si celle-ci se trouve faite dans un autre domaine des mathématiques (par exemple, dans notre cas, la dichotomie).

3.2.8.5. Phase 4 : Approche « probabiliste » du problème : détermination de deux lois de probabilité et calculs de leurs espérances mathématiques pour obtenir la proportion de garçons parmi le nombre total d'enfants

a) Le déroulement (20 minutes)

Cette phase débute par un temps d'échanges entre les élèves au niveau des binômes afin de déterminer si l'approche « fréquentiste » pourrait suffire pour confirmer ou pas les réponses données lors du sondage de la phase 1. Ce temps d'échanges dure environ 5 minutes.

Puis, une fois que l'ensemble des élèves a accepté la nécessité de passer à une approche « probabiliste » de la problématique, nous observons que le travail se poursuit autour de la

construction d'un arbre pondéré de probabilités afin de déterminer à l'aide des calculs de probabilités des différents « chemins » décrits par l'arbre les lois de probabilités associées aux variables aléatoires représentant le nombre total d'enfants et le nombre de garçons. Ce temps de construction ou de reprise de l'arbre obtenu à la phase 1 demande environ 5 minutes quel que soit le niveau scolaire observé.

Un nouveau temps d'échanges entre les élèves sur la nécessité de présenter un tableau ou deux tableaux pour représenter les lois de probabilités dure aussi environ 5 minutes au niveau de la Première. En revanche, au niveau de la Seconde cette étape de la phase pose des difficultés pour certains élèves et peut nécessiter une intervention de l'enseignante au niveau de certains binômes. Une conséquence due à cette difficulté va être que plus de 60% des binômes de Seconde a besoin de la totalité du temps restant, soit 10 minutes, pour donner les lois de probabilités complètes.

Pendant les cinq dernières minutes restantes pour cette phase et par conséquent pour la totalité de la séance, parmi les élèves qui ont obtenu sans difficulté majeure les lois de probabilités, nous observons qu'ils poursuivent la démarche attendue (calcul d'espérances) pour déterminer la proportion de garçons par rapport au nombre d'enfants, sans pour autant avoir suffisamment de temps pour revenir sur le sondage faute de temps.

d) Analyse a posteriori

Environ 40% des binômes de Seconde et 90% des binômes de Première ont conscience que le travail fait au cours des phases 2 et 3 autour d'une approche « fréquentiste » de la problématique ne permet pas nécessairement de porter une analyse critique sur les réponses données au sondage lors de la phase 1. En revanche, nous observons qu'au niveau de la Seconde, l'enseignante doit intervenir pour rappeler aux élèves que le travail fait dans le cadre d'une approche « fréquentiste » peut s'avérer insuffisante pour valider ou invalider les réponses faites au sondage lors de la première phase. Ainsi, ce temps d'échange et d'intervention de l'enseignante au niveau de la Seconde sur la nécessité ou pas de passer à une approche de la problématique pour valider ou pas les réponses données au sondage dure environ 5 minutes. Il en est de même pour les 10% de binômes de Première qui considèrent que le travail demandé est terminé.

Une fois que l'ensemble des élèves a accepté la nécessité de passer à une approche « probabiliste » de la problématique, nous observons que les élèves qui ont construit un arbre

de probabilité lors de la phase 1, pour formuler des réponses « réfléchies » au sondage distribué, prennent l'initiative de reprendre cet arbre au début cette dernière phase. Pour les autres, nous constatons aussi qu'ils mettent en place une démarche analogue en construisant en ce début de phase 4 un arbre de probabilité. Dans tous les cas et quel que soit le niveau scolaire, nous observons que les élèves complètent leurs arbres en indiquant les probabilités sur les différentes branches de l'arbre.

Cette construction d'arbres ne demandant que quelques minutes à l'ensemble des binômes et ceci quel que soit le niveau scolaire des élèves, il s'ensuit alors des échanges pouvant aller jusqu'à dix minutes suivant les différents binômes sur la nécessité de présenter un ou deux tableau(x) permettant d'avoir les lois de probabilités associées respectivement aux variables aléatoires représentant le nombre total d'enfants et le nombre de garçons.

Ainsi, nous avons peu de diversité dans le retour des binômes sur ces lois de probabilités. En effet, 85% des binômes des deux classes confondues, nous renvoient un seul tableau montrant les deux lois dans le cas où il y a équiprobabilité de sexe sur une naissance. De plus, nous observons que 10% de l'ensemble des binômes nous renvoient deux tableaux associés respectivement au nombre total d'enfants et au nombre de garçons. Le binôme qui avait rendu un arbre incomplet au niveau du quatrième enfant (cf. l'analyse *a posteriori* de la phase 1), rend un tableau erroné (Fig. 202) mais conforme à son arbre (Fig. 188).

Nombre total d'enfants	**Nombre total de garçon**	**Probabilités**
4 (quatre filles)	0	$0,5^4$
4 (trois filles et un garçon)	1	$0,5^3 \times 0,5 = 0,5^4$
3 (deux filles et un garçon)	1	$0,5^2 \times 0,5 = 0,5^3$
2 (une fille et un garçon)	1	$0,5 \times 0,5 = 0,5^2$
1 (un garçon)	1	0,5
Total		**1**

Nombre total d'enfants	**Probabilités**
4 (4 filles ou 3 filles et 1 garçon)	$0,5^4 + 0,5^4 = 2 \times 0,5^4 = 0,125$
3 (deux filles et un garçon)	$0,5^2 \times 0,5 = 0,5^3 = 0,125$
2 (une fille et un garçon)	$0,5 \times 0,5 = 0,5^2 = 0.25$
1 (un garçon)	0,5
Total	**1**

Nombre total de garçons	**Probabilités**
0	$0,5^4 = 0,0625$
1	$0,5^4 + 0,5^3 + 0,5^2 + 0,5 = 0,9375$
Total	**1**

Figure 201 (Extraits de copies donnant les probabilités des différentes types de famille)

Nombre total d'enfants	Nombre total de garçon	Probabilités
4 (3 filles et 1 garçon)	1	$0,5^3 \times 1 = 0,125$
3 (2 filles et 1 garçon)	1	$0,5^2 \times 0,5 = 0,125$
2 (1 fille et 1 garçon)	1	$0,5 \times 0,5 = 0,25$
1 (1 garçon)	1	0,5
Total		**1**

Figure 202 (Extrait d'une copie donnant les probabilités des différentes types de famille)

5% de l'ensemble des binômes ne cherchent pas à répondre à ces questions en lien avec les lois de probabilités et interrompent leur travail au niveau de la construction de l'arbre. Nous ne pouvons pas savoir si ce fait est dû à des difficultés pour déterminer ces lois et par conséquent les espérances sachant que ces binômes sont tous des élèves de seconde ou bien si c'est dû à un désintérêt pour la tâche. En effet, nous n'avons pas pu avoir un temps d'échange avec les élèves concernés.

De plus, contrairement à ce que nous avions envisagé lors de l'analyse *a priori*, nous n'observons pas de binômes proposant de travailler ces lois de probabilités dans le cas où la naissance d'un garçon serait *p* au lieu de 0,5, malgré le fait que lors de la phase précédente cette situation ait été demandée. Nous pouvons penser que ceci soit dû à un problème de gestion du temps pour les élèves, mais aussi par le fait que la problématique initiale était sur une situation d'équiprobabilité de sexe lors d'une naissance et que les élèves n'ont pas senti la nécessité d'étudier le cas où il n'y aurait pas cette équiprobabilité.

Nous observons aussi que 90% élèves de Première et un binôme de Seconde (soit 6% des binômes de la classe) prennent l'initiative de calculer les espérances mathématiques concernant le nombre d'enfants et le nombre de garçons. Nous observons ensuite qu'ils calculent le quotient de ces deux résultats pour conclure sur la proportion de garçon. Cependant, l'unique binôme de Seconde ayant fait ces calculs et 22% des binômes de Première souhaitent commenter ces résultats en comparant avec ce qui avait été émis comme hypothèses lors du travail sur le sondage de la phase 1.

3.2.9 Synthèse de l'analyse *a posteriori* autour des articulations entre Espaces de Travail Mathématique et Algorithmique des différentes phases

3.2.9.1. Présentation

Afin de terminer l'analyse des trois phases décrites dans la section précédente, nous souhaitons revenir sur l'analyse *a posteriori* afin de répertorier dans le travail fait par les élèves les articulations possibles entre Espaces de Travail Algorithmique et Mathématique, pour mettre en évidence les apports que pourraient avoir l'algorithmique dans le domaine des probabilités, en particulier lors d'un travail de modélisation d'un processus aléatoire en lien avec le monde « réel » au niveau de la Seconde et de la Première Scientifique.

3.2.9.2. Les tableaux résultant de l'analyse *a priori* dans le cas des paradigmes et de l'expérience observée pour les articulations entre Espaces de Travail

a) Les paradigmes

Les niveaux de paradigmes algorithmiques	
Phase 1 • Réponse à un sondage autour d'une réflexion sur la politique proposée et les conséquences quant à l'évolution envisagée par le gouvernement fictif. • Détermination des fratries possibles dans une famille	→ Au cours de cette première phase, les élèves doivent se positionner sur l'hypothèse émise par le gouvernement fictif quant à la conséquence d'une telle politique sur l'évolution démographique de la population et en particulier la fréquence des garçons sur l'ensemble des naissances. Cette première approche de la politique des naissances se fait au papier-crayon et se situe au niveau 1 des paradigmes de la modélisation d'une simulation d'un processus aléatoire. Le monde « fictif » peut-être ici assimilé à un monde « réel » dont il faudra au cours des phases suivantes déterminer un modèle mathématique ou algorithmique. → Les élèves doivent déterminer les fratries possibles et pour cela, nous pensons qu'ils vont utiliser la construction d'un arbre de probabilités qui est fortement encouragé par l'institution. Ainsi, cette tâche se situe au niveau 2 des paradigmes des probabilités.
Phase 2 • Approche « fréquentiste » du processus aléatoire • Simulation sur tableur dans le cadre de l'approche « fréquentiste »	→ Au cours de cette phase, dans le cadre d'une approche « fréquentiste », les élèves conçoivent un modèle d'une simulation de la politique pour 1 000 familles puis pour 10 000 familles à l'aide d'un tableur. Cette tâche nécessite que les élèves maitrisent la syntaxe du langage machine et les fonctionnalités de l'environnement numérique permettant d'obtenir la fratrie d'une famille que sont : =NB.SI, =NB, SI(…),…, mais aussi qu'ils interprètent les répétitions sur 1 000 (ou 10 000) familles comme le fait de procéder à des copier-coller afin d'obtenir les différentes fratries. Nous nous situons ainsi au niveau 2 des paradigmes algorithmiques. → Nous pouvons penser que les élèves peuvent interpréter le fait de construire

	cette simulation à l'aide du tableur comme une première approche de type « statistique » du processus aléatoire permettant d'avoir un premier modèle de type algorithmique sans pour autant s'intéresser au problème des tests de sortie de boucles, qui fait l'objet du travail de la phase suivante. Cet aspect se situe donc au niveau 1 de l'algorithmique.
Phase 3 • Construction d'un organigramme permettant d'obtenir une modélisation de cette politique • Approche statistique du processus aléatoire • Simulation dans un environnement algorithmique et permettant une étude fréquentiste	→ Au cours de cette phase, comme pour la précédente, les élèves conçoivent de nouveaux modèles d'une simulation de la politique à l'aide d'algorithmes au format « spatial » avec l'organigramme, puis « textuel » afin d'être implémentés dans l'environnement numérique ALGOBOX, en tenant compte que le nombre de familles peut être quelconque, que les issues (filles ou garçons) possibles lors d'une naissance peuvent ne pas être équiprobables. → La construction d'un algorithme modélisant ce processus aléatoire doit permettre aux élèves de sortir de la représentation qu'offre l'arbre de probabilités. En effet, les élèves ont la possibilité de proposer plusieurs « solutions » : • une utilisation d'alternatives successives plutôt qu'imbriquées et des expressions de boucle de type « Si *condition* sortir sinon *recommencer* », c'est-à-dire du type « GOTO » qui fait que l'élève transpose le principe mis en place pour la construction de l'arbre de probabilité à la construction de l'organigramme ; • une structure de type « TantQue» directement déduite du fait que le nombre d'enfants par famille est de quatre au maximum. La construction d'une structure algorithmique va situer le travail attendu des élèves au niveau 2 des paradigmes algorithmiques, en particulier pour l'étude du test de sortie de la boucle. → De plus, les élèves doivent mettre en place des variables informatique : • le nombre d'enfants ; • le sexe de l'enfant. Conformément aux travaux de Rogalski et Samurçay (1986), pour des débutants en programmation, le travail sur des variables informatiques évoluant au cours de l'itération (nombre d'enfants évolue et le sexe de l'enfant fait appel au concept de nombre aléatoire) va se situer au niveau 2 des paradigmes algorithmiques. → Lors de l'approche « fréquentiste » avec l'utilisation de l'environnement numérique AlgoBox, les constructions des structures algorithmiques vont en effet, situer le travail des élèves au niveau 2 des paradigmes algorithmiques, en particulier pour déterminer les choix des structures algorithmiques : série imbriquée de « Si…Alors…Sinon » (comme dans le cas de l'organigramme (cf. ci-dessus)) ou utilisation d'une boucle « TantQue » concernant la variable associée au nombre maximal d'enfants dans une famille ; utilisation d'une boucle « Pour » permettant de simuler le nombre de familles étudiés. → De plus, les élèves doivent mettre en place différentes variables informatique : • le nombre maximal d'enfants ; • le sexe de l'enfant ; • la proportion de garçon parmi les naissances ; • le nombre total d'enfants ;

	• le nombre total de garçons ; • la proportion de garçons par rapport au nombre total d'enfants. → Cette phase, comme la précédente, conduit les élèves à un retour sur les hypothèses émises au début de la première phase sur les conséquences démographiques prévisibles si une telle politique est mise en place. Ceci fait l'objet du travail implicite à mettre en place dans la phase 4. Ainsi, l'ensemble du travail que doit mener l'élève tout au long de cette phase se situe au niveau 2 de l'algorithmique
Phase 4 • Approche « probabiliste » du processus aléatoire	→ Prise de conscience que l'« approche fréquentiste » peut ne pas être suffisante pour analyser totalement les réponses faites au sondage lors de la phase 1. Et par conséquent, l'élève doit avoir conscience de la nécessité de passer par une approche « probabiliste ». Ainsi, nous nous situons au niveau 2 des paradigmes des probabilités. → La détermination des lois de probabilités et des espérances se situent au niveau 2 des paradigmes des probabilités. → Retour sur les hypothèses émises au début de la première phase sur les conséquences démographiques prévisibles si une telle politique est mise en place.

Tableau 50

b) Les articulations entre Espaces de Travail vues lors de l'expérimentation

Espaces de Travail spécifiques	***ETA***	***ETM***	**Commentaires afin de montrer que les deux Espaces de Travail sont porteurs d'idées intéressantes**
Phase 1 • Répondre à un sondage sur les conséquences possibles avec la mise en place de la politique envisagée par le gouvernement fictif • Possibilité de construire un arbre de probabilités		Analyser la politique et proposer un modèle construit à partir d'un arbre pondéré de probabilités permettant d'obtenir les types de familles possibles (Genèses instrumentale et discursive)	Mener un raisonnement dans le cadre d'un ETM de probabilité avec la représentation d'une situation donnée, à l'aide d'un arbre de probabilités permettant de déterminer le type de fratries possibles sans pour autant aller jusqu'au bout de l'approche « probabiliste » qui fait l'objet du travail de la dernière phase.
Phases 2 et 3 • Approche « fréquentiste » du problème	• Construire un modèle de la simulation à l'aide du tableur • Connaissance des	• Générer un modèle mathématique représentatif du processus aléatoire.	• Les gestes de construction et de visualisation d'un algorithme au format

• Modèles de la simulation du processus aléatoire à l'aide du tableur (phase 2) et de l'écriture d'algorithmes au format « spatial » avec l'organigramme ou « textuel » afin d'être implémentable dans l'environnement numérique ALGOBOX pour y être testé (phase 3).	fonctionnalités qu'offre le tableur • Utilisation du copier-coller pour simuler plusieurs familles (Genèses instrumentale et sémiotique) • Construire des algorithmes soit sous forme « spatiale » avec un organigramme soit sous forme « textuelle » avec l'utilisation d'un langage « pseudo-code ». (Genèses instrumentale et sémiotique/visualisation) • Suivant le choix de la structure algorithmique retenue, l'algorithme (quel que soit son format) nécessite de mener un travail sur : - Choix de Branches « SI et SINON » - Condition dans l'alternative ; - Choix d'une structure algorithmique de type « TantQue » directement déduite de l'impossibilité de garder le choix précédent si le nombre maximal d'enfants devient plus grand ; - Organiser un test de sortie de boucle à l'aide d'une inégalité suivant la structure retenue ; - Considérer que l'inégalité peut-être strict et alors prendre	Par exemple avec les lancers successifs d'une pièce de monnaie • Générer un nombre aléatoire • Calcul d'effectifs • Calcul d'une fréquence • Possibilité d'utiliser des pourcentages pour cette fréquence • Envisager un retour sur les hypothèses faites au début de la phase 1 (cf. phase 4) (Genèse discursive)	« textuel » à mettre en place sont familiers pour des élèves de Seconde et de Première. • Nous pouvons observer que des constructions « spatiales » ou « textuelles » peuvent être complémentaires afin de mieux comprendre les différentes étapes d'un algorithme. • L'environnement *LARP* permet à des élèves débutants de mieux voir la transcription d'une représentation spatiale sous forme d'organigramme à une représentation « textuelle » sous forme de langage pseudo-code. Le travail personnel de l'élève peut être envisagé dans des ETA spécifiques logiciels. • Le recours à l'arbre de probabilité lors de la phase précédente peut générer des difficultés lors du passage à un ETA logiciel. En effet, Certains élèves utilisent des alternatives imbriquées se ramenant à transposer l'arbre de probabilité à l'organigramme. Nous pouvons interpréter cela comme une réduction à un « copier-coller » renvoyant à un ETA de type « tableur »

	un nombre maximal d'enfants supérieur au nombre maximal d'enfants autorisé, ou que l'inégalité est large et alors prendre le nombre maximal d'enfants autorisé comme condition de sortie de la boucle (Genèses instrumentale, sémiotique et discursive) • mise en place de variables de type « Nombre » avec initialisation de certaines ; (Genèses instrumentale et discursive) • Utilisation d'un nombre aléatoire afin de déterminer le sexe de l'enfant qui nait : cas où il y a équiprobabilité dans le sexe des naissances – cas où il n'y a pas équiprobabilité (Genèses instrumentale et sémiotique) • Construire une structure de boucle « POUR » permettant de générer plusieurs familles lors du travail à mener dans l'environnement ALGOBOX (Genèses instrumentale) • Calcul d'une fréquence (Genèses instrumentale, sémiotique et discursive)		(comme nous l'avons observé dans la phase 2) ou à une transcription d'un raisonnement issu d'un ETM de probabilité avec la représentation d'une situation donnée, à l'aide d'un arbre. • Des élèves ont conscience qu'une structure « TantQue ». est plus favorable pour l'organigramme en particulier dans le cas où le nombre maximal d'enfants dans une famille est important. Comme nous avons pu le voir, ceci est dû en partie à une mauvaise interprétation d'une compétence venant d'un ETM de probabilité avec le concept d'*événement contraire*. L'aspect ETA autour de l'approche « fréquentiste » peut donner du sens à des élèves qui n'ont pas la connaissance de loi géométrique tronquée et par conséquent donner du sens à l'approche probabiliste du problème. De plus, les ETA peuvent permettre de proposer plusieurs modèles permettant de simuler le processus aléatoire.
Phase 4 • Approche « probabiliste »	Prendre conscience que le travail fait dans l'ETA renvoi à une approche	Analyser la politique et proposer un modèle construit à partir d'un	Cette dernière phase se situe dans un ETM de probabilités.

du processus aléatoire • Retour sur les réponses données dans le sondage de la phase 1	« fréquentiste » de la problématique et que celle-ci doit amener une articulation avec un ETM de probabilité pour valider à travers le calcul d'espérances, les observations faites sur des moyennes calculées dans le cadre d'un ETA sur des échantillons de taille plus ou moins grande de la population. (Genèse discursive)	arbre pondéré de probabilités permettant d'obtenir les types de familles possibles, ainsi que les lois de probabilités que suivent deux variables aléatoire. Calcul d'espérances mathématiques et d'une proportion. (Genèses instrumentale et discursive)	Cependant, nous observons que le travail fait lors des deux phases précédentes dans des ETA montre que les élèves peuvent travailler en autonomie et prendre d'eux même l'initiative de mener un travail dans le cadre d'une approche « probabiliste » de la problématique. En effet, ces élèves ne se contentent pas de calculs de moyennes sur un échantillon même si la taille de celui-ci est grande pour valider ou pas les réponses données au sondage lors de la phase 1. Plus généralement, cette dernière phase permet d'observer l'évolution du travail mis en place dans des classes ayant eu deux ingénieries didactiques articulant ETA et ETM. En effet, les élèves montrant une plus grande aisance dans l'ETA suite au travail accompli lors de l'ingénierie sur la dichotomie, peuvent plus se concentrer sur l'ETM et en particulier sentir la nécessité de justifier les résultats obtenus dans un ETA.

Tableau 51

4. Conclusion du chapitre 6

Nous rappelons que l'enjeu de cette ingénierie est d'observer et d'analyser les niveaux de compétences en probabilités, en algorithmique mais aussi en modélisation chez des élèves n'ayant pas de compétences sur la loi géométrique tronquée au moment de l'expérimentation

sur une politique des naissances formulée par un gouvernement fictif.

4.1 L'aspect modélisation

Lors de la description de notre cadre théorique (cf. Chapitre 2, partie 1), nous avons présenté la possibilité que dans le cadre des *ETM* et des *ETA*, nous pouvions être amenés lors de phases de *cycle de modélisation* (Blum & Leiss, 2005), à observer que certaines étapes de modélisation peuvent être nécessaires. Pour cela, nous sommes conduits à référencer les différents stades entre le passage du monde « réel » au monde « mathématique » (Fig. 203), dans le cadre d'un *ETM* spécifique aux probabilités et d'*ETA*.

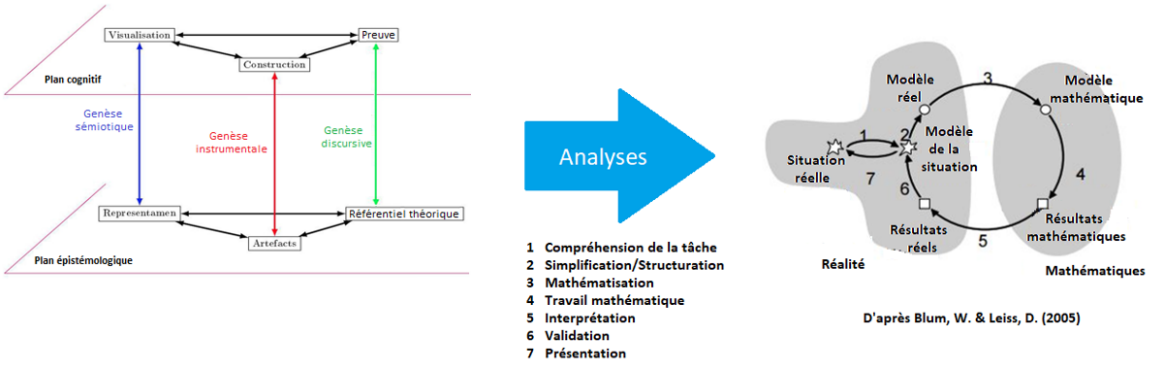
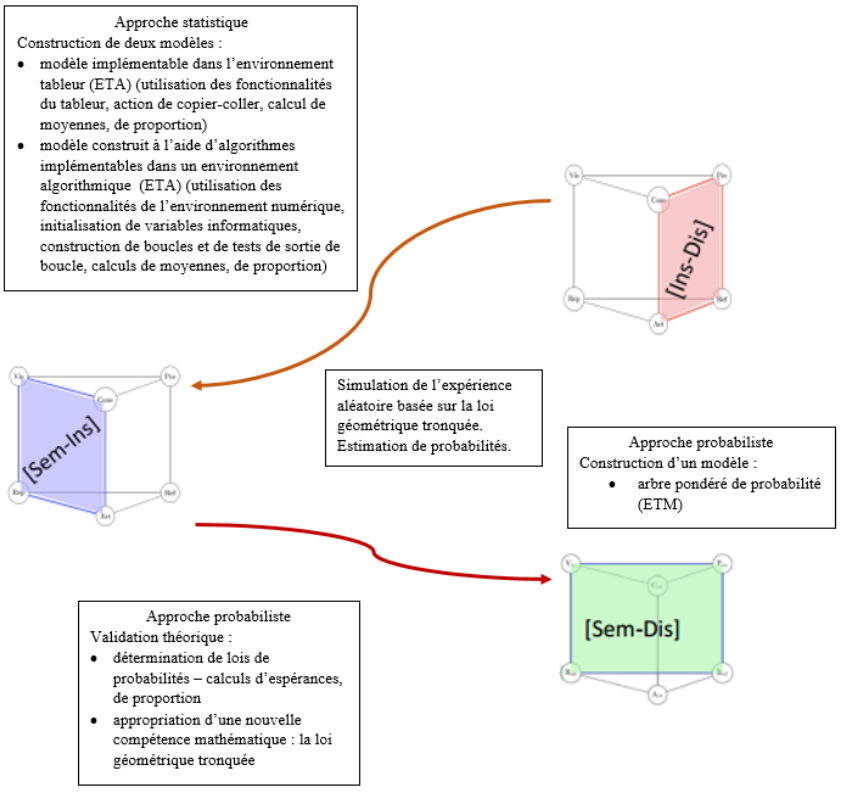


Figure 203 (ETM/ETA associés au cycle de modélisation de Blum et Leiss (2005))

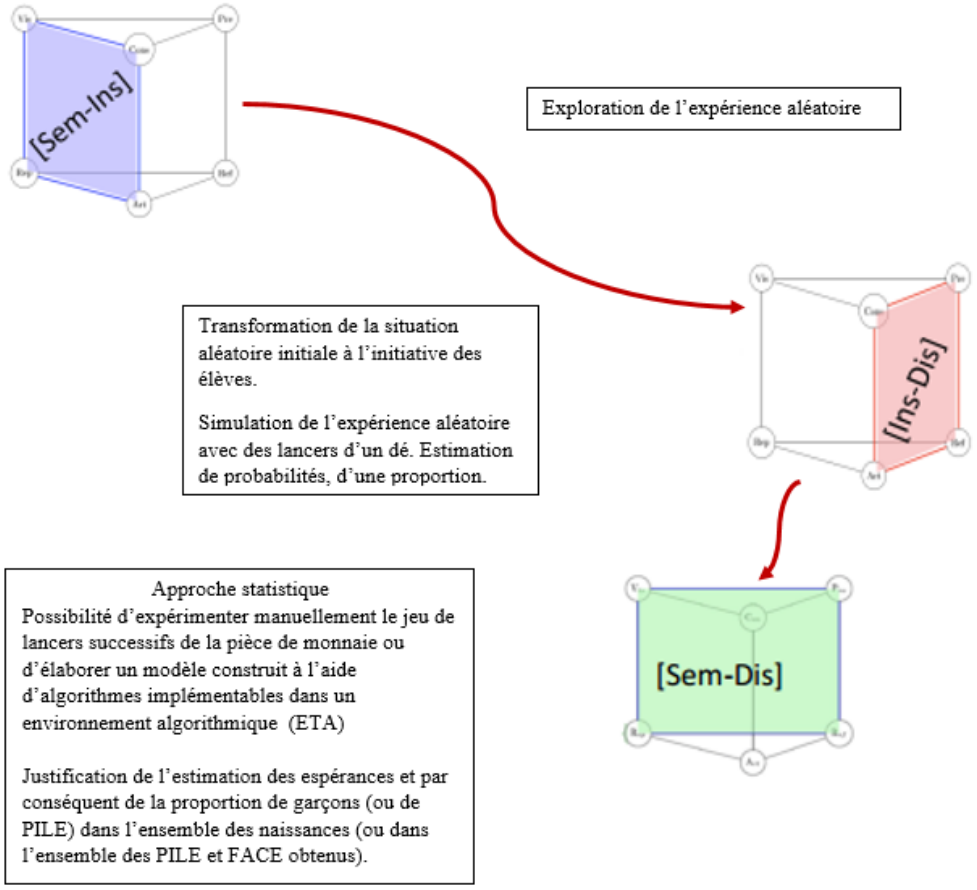
Nous avons donc vu que lors des quatre phases, les références aux Espaces de Travail mis en place peuvent être schématisées suivant le graphique suivant (Fig. 204) :



Références aux ETM et ETA (phases 2, 3 et 4)

Figure 204 (Références aux ETM et ETA (phases 2, 3 et 4))

De même, nous observons chez un certain nombre d'élèves de Première que ceux-ci souhaitent partir d'un modèle représentant la simulation du processus aléatoire basé sur des lancements successifs d'un dé qui permettent aussi d'expérimenter manuellement la situation aléatoire sans nécessairement passer par l'algorithmique (Fig. 205).



Adaptation des ETM et ETA (phases 2, 3 et 4)

Figure 205 (Adaptation des ETM et ETA (phases 2, 3 et 4)

Ainsi, l'analyse à travers une série d'images (Fig. 206) nous montre une circulation complète à travers les trois plans verticaux des modèles des *ETM* et *ETA* conduisant à ce que Kuzniak et Nechache (2017, dans le cas géométrique) identifient comme un travail complet dans un domaine des mathématiques (ici, les probabilités/statistiques) mais aussi dans le domaine de l'algorithmique pour notre expérimentation.

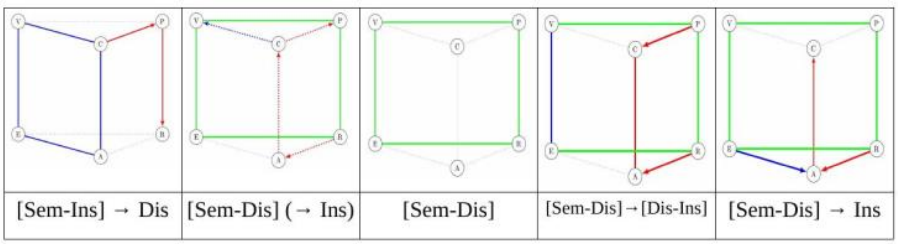


Figure 206 (La dynamique d'évolution d'un travail mathématique et algorithmique durant les différentes sessions)

Selon Kuzniak, Nechache et Drouhard (2016), le travail mathématique, voire algorithmique, est considéré comme complet lorsque les conditions suivantes sont satisfaites :

(1) une véritable relation entre les plans épistémologique et cognitif. Cet aspect signifie que les élèves, qu'ils soient génériques ou non, sont en mesure de sélectionner les outils utiles pour résoudre le problème posé, puis de les utiliser de manière appropriée comme instruments pour résoudre la tâche demandée ;
(2) une articulation d'une diversité riche entre les différentes genèses et les plans verticaux du modèle. Cet aspect signifie que diverses dimensions du travail liées aux outils, aux techniques et aux propriétés vont être prises en compte. Ce point reflète la manière dont différents contextes de travail sont impliqués pendant l'activité de l'élève.

4.2 Donner du sens aux objets algorithmiques et informatiques mis en jeu lors de cette expérimentation

Lors de cette expérimentation, nous observons plusieurs points intéressants. Un premier qui concerne le peu d'erreurs faites par les élèves lors de la construction des algorithmes, tant au niveau de la Première que de la Seconde. En effet, nous pouvons supposer que le fait que cette expérimentation ait eu lieu dans des classes ayant participé à l'expérimentation sur la dichotomie a vraisemblablement favorisé le fait que les élèves aient une bonne connaissance des objets algorithmiques et informatiques utilisés : choix et maitrise des structure algorithmiques, gestion des variables informatiques, connaissance des environnements numériques, ainsi que du langage pseudo-code et d'un langage machine nécessaire pour implémenter l'algorithme construit par l'élève dans un environnement numérique.

Nous pouvons ainsi constater que pour des élèves de ces niveaux scolaires et motivés par un travail de nature algorithmique en lien avec un domaine mathématique spécifique (probabilités, analyse,…), les apprentissages de nouvelles compétences mathématiques à travers l'algorithmique se font de façon plus aisés que si ces compétences attendues restent cloisonnées au domaine mathématique spécifique. Nous constatons aussi que des élèves du lycée semblent donner plus facilement du sens aux objets algorithmiques que mathématiques.

4.3 Les différentes approches de la politique des naissances

Nous constatons qu'une approche « fréquentiste » avec la détermination de fréquences relatives aux caractères représentatifs d'un processus aléatoire aide l'élève à une meilleure compréhension de la notion de « probabilité théorique » d'un événement. De même, le calcul de moyennes statistiques le prépare à donner du sens au concept d'espérance d'une variable aléatoire conformément à ce que relève Parzysz (2007) lors de l'analyse de travaux d'élèves sur les expériences aléatoires et les simulations, ainsi que le passage d'une expérience aléatoire au modèle, via la simulation (Parzysz, 2009).

Le fait aussi d'utiliser, des algorithmes permet aux élèves de pouvoir mieux comprendre cette approche « fréquentiste » en pouvant augmenter par exemple, de façon conséquente la taille de l'échantillon de la population étudiée.

4.4 Une suite possible à cette ingénierie

Une cinquième phase peut être envisagée où serait mis en place variable aléatoire X qui vaudrait :

- i si les $i - 1$ « preuves tirages » sont des échecs et le $i^{ème}$ est un succès avec i parcourant l'ensemble E = {1 ; 2 ; 3 ; 4} ;
- 0 si les 4 tirages sont des échecs.

Les élèves seraient alors amenés à expliquer dans un premier temps comment la politique des naissances consiste en une expérience aléatoire de ce type, puis à adapter l'algorithme obtenu au début de la phase 3, pour déterminer la fréquence du nombre de succès lors de la répétition d'une expérience de ce type. Ceci pourrait être complété en considérant la même expérience avec les paramètres m et q, où m correspondrait au nombre de répétitions de l'expérience et q le nombre de la proportion du succès lors d'un tirage. Les élèves seraient à nouveau amenés à modifier l'algorithme pour la simulation. Une telle variable aléatoire permettrait de donner du sens à la loi géométrique tronquée chez l'élève découvrant cette loi.

Conclusion de la partie 2

1. Deux ingénieries sur algorithmes et apprentissage de la preuve

Dans la continuité des travaux menés par Modeste (2012), nous nous sommes intéressés à la problématique de l'enseignement de l'algorithme dans les classes de Seconde et du cycle scientifique Terminal, avec l'introduction de l'algorithmique, mais avant tout pour son lien étroit au concept de preuve dans deux domaines spécifiques des mathématiques : théorie élémentaire des nombres (algorithme de Kaprekar) et analyse (algorithme de « dichotomie continue »). Ceci nous a amené à soulever plusieurs questions qui sont à l'origine de nos travaux de recherche. En effet, ces questions s'inscrivent dans la suite du travail sur l'algorithmique mené par Ouvrier-Buffet, Modeste et Gravier (2010) autour de l'apprentissage de la preuve ainsi que *l'épistémologie de l'algorithme, la transposition de l'algorithmique au lycée, et la construction de situations* (Modeste, 2012). Pour cela, nous avons construit deux ingénieries dans les domaines mathématiques cités ci-dessus en tenant compte que les élèves observés travaillent en autonomie et peuvent réagir en fonction du contrat didactique mis dans les classes concernées où le concept de preuve d'une conjecture ou d'un théorème peut être la clé de l'apprentissage en mathématique.

1.1 L'algorithme de Kaprekar

Au cours de l'ingénierie « Algorithme de Kaprekar » menée dans une de classe de Terminale Scientifique et plus précisément en enseignement de spécialité, les élèves ont poursuivi le travail accompli lors d'une première ingénierie (dite « ingénierie-Guy ») qui était basée sur la planification à mettre en place lors de la construction d'un algorithme complexe (Guy, Lagrange, 2013) permettant de modéliser un processus de calcul sur des nombres entiers de 000 à 999 (cf. Chapitre 2). Ainsi, lors de notre ingénierie sur Kaprekar les élèves de cette classe ont centré leur travail algorithmique sur une *preuve* de la conjecture émise lors de l'« ingénierie-Guy » mettant en jeu des outils mathématiques et algorithmique.

Ce travail permet dans le cadre de notre recherche de témoigner des différents niveaux de paradigmes algorithmiques, ainsi que des différentes genèses pouvant apparaître tant dans un *ETM* que dans un *ETA*. Pour cela, après avoir construit un premier algorithme du processus de calcul leur permettant d'émettre une conjecture sur un résultat numérique observé dans

un premier temps manuellement, les élèves ont pu, suite à la proposition faite par le chercheur de la lecture d'un article sur la résolution du *Rubik's Cube*, prendre conscience de l'importance de construire une preuve « explicative » de la conjecture. En effet, nous nous sommes appuyés sur le fait que le niveau de connaissances de la part des élèves observés au niveau de l'abstraction et du raisonnement en mathématique pouvait donner du sens à cette lecture de l'article en transposant l'idée centrale de la problématique de « réduction des cas » à l'idée d'une preuve « explicative » sur un ensemble de nombre à tester. Nous avons ainsi élaboré une telle approche afin de favoriser les articulations possibles entre *ETM* et *ETA*. En effet, ces articulations *Espaces de Travail Algorithmique* et *Espaces de Travail Mathématiques spécifiques* participent à l'élaboration de la « pensée mathématique » chez l'élève car bien que le travail autour de la preuve se fasse pour l'essentiel dans un *ETM*, l'*ETA* reste continuellement « en toile de fond » sur les deux aspects que nous avons observés au cours de l'expérimentation. L'élève fait en quelques sortes des allers-retours entre les deux Espaces de Travail.

Ainsi, dans le cadre d'articulations entre *ETM* et *ETA*, nous constatons que l'élève comprend qu'il mène un travail en mathématiques, et que l'*ETA* lui sert de référence en ce qui concerne les représentations sémiotiques (au sens de Duval) du nombre entier. En effet, pour l'élève le nombre entier n'est pas un objet réel ou physique. Pour le manipuler, il doit passer par des représentations mentales et sémiotiques.

Nous avons pu mettre en valeur les représentations mentales d'un objet mathématique comme le « nombre entier » chez l'élève en fin de scolarité obligatoire. Ces constructions et représentations mentales sont faites dans un contexte particulier et à des fins spécifiques. Elles correspondent aux conceptions de l'élève sur l'objet « nombre entier », mais aussi sur une situation et ce qui lui est associé. Ainsi, les représentations sémiotiques sont définies comme des productions constituées par l'emploi de signes, d'écritures appartenant à un système de représentations qui a ses propres contraintes de signifiance et de fonctionnement que ce soit dans le domaine des mathématiques ou le domaine de l'algorithmique. De cette façon l'élève renforce la prise de conscience que l'objet mathématique « nombre entier » peut avoir plusieurs représentations sémiotiques.

1.2 L'algorithme de dichotomie

Au cours de la seconde ingénierie sur l'« algorithme de dichotomie », tenant compte de la dichotomie *outil-objet* sur les algorithmes définie par Ouvrier-Buffet, Modeste et Gravier (2010), nous souhaitons étudier les différents aspects de l'algorithme, en particulier le fait que celui-ci puisse être enseigné en tant qu'objet mathématique. En effet, ces derniers considèrent que *pour mieux appréhender les questions de l'algorithme, il faut se pencher sur son rôle dans les mathématiques et sur les différents aspects qu'il revêt* (Ouvrier-Buffet, Modeste et Gravier, 2010). Pour cela, ils s'appuient sur la dialectique *outil-objet* développée par Régine Douady (1986), où l'on peut classer les algorithmes *suivant qu'ils relèvent de l'aspect « outil » du concept ou de l'aspect « objet »* (Ouvrier-Buffet, Modeste et Gravier 2010). Selon eux, *l'algorithme n'est pas uniquement un outil pour la résolution de problèmes mais c'est aussi un objet mathématique à part entière, pour lequel il existe un cadre d'étude précis*. […] *regarder l'algorithme en tant qu'objet, c'est s'intéresser aux questions de bon fonctionnement, de domaine de validité, de complexité et de description des algorithmes. Ce sont des problématiques de l'algorithmique. Les aspects preuve, complexité, machine de Turing et constructif réfèrent à l'objet algorithme. Regarder l'algorithme en tant qu'outil, c'est s'intéresser à l'utilisation que l'on en fait pour résoudre des problèmes. Les aspects résolution de problème, effectivité et formule réfèrent à l'outil algorithme.*

Tenant compte de compte de cette approche d'Ouvier-Buffet, Modeste et Gravier, nous avons souhaité construire une ingénierie « globale » sur les trois années de lycée : Seconde et cycle Terminal scientifique. Pour cela, nous sommes partis d'un algorithme dont l'institution reconnait sa particularité pour être enseigné ou utilisé dans ces différents niveaux scolaires. Nous avons aussi choisi un algorithme qui permet d'avoir une double approche : « discrète » et « continue » dans le domaine de l'analyse afin de mener un travail qui se conclue sur son utilisation dans le cadre d'une preuve mathématique d'un théorème fondamental pour les élèves de Terminale Scientifique : le « théorème des valeurs intermédiaires ».

L'importance du choix d'une première approche « discrète » de l'« algorithme de dichotomie » permet à l'élève de construire un algorithme présentant une stratégie « rapide » et « gagnante » de recherche d'un nombre entier « secret » se situant entre deux nombres entiers connus (cf. chapitre 4) sans que cela ne s'exprime pas pour autant chez l'élève en termes d'optimalité, conception qui n'est pas institutionnalisée au niveau du lycée. La

structure et les variables, ainsi que la construction d'un test d'arrêt comme égalité à un nombre fixé de cet algorithme dit de « dichotomie discrète » permet à l'élève d'approcher celle de l'algorithme de « dichotomie continue » objectif principal de cette seconde ingénierie.

En effet, la structure algorithmique dans les cas « discret » et « continue » est la même, mais ils n'opèrent pas sur les mêmes données et ne répondent pas au même problème. Cette double approche algorithmique de la méthode de dichotomie permet de montrer aux élèves l'importance du choix des structures et des conditions d'alternatives, mais aussi des difficultés que peuvent entrainer la construction de tests d'arrêt différents (condition d'égalité à nombre fixé dans le cas « discret » ou condition d'inégalité à un nombre arbitraire dans le cas « continu ») suivant que le travail mené soit dans une situation « discrète » ou « continue ».

De plus, les variables itératives vues dans le cas « discret » peuvent aussi être réinvesties dans le cas « continu » et un travail sur l'affectation des variables en algorithmique peut permettre de modifier les représentations erronées sur les variables informatiques et mathématiques qu'ont certains élèves débutants en informatique.

Ainsi, en accord avec les travaux de Samurçay et Rogalski (1986), nous pouvons nous pencher sur les problèmes cognitifs rencontrés par les élèves débutants dans l'apprentissage de l'informatique, *d'une part sur les représentations que les élèves construisent sur le fonctionnement du système et sur la structuration de l'information (représentation des données) et, d'autre part, sur l'acquisition des concepts clés de variable et d'itération* (Samurçay & Rogalski, 1986). Nous pouvons ainsi observer, lors des analyses *a posteriori* dans les cas « discret » et « continu » des travaux rendus par des élèves débutants en informatique, que le choix de le faire travailler la notion de variable comme objet d'étude, peut être pertinent par son caractère central en programmation. En particulier, l'étude des variables dans une structure de boucle plutôt que lors de simple déclaration ou d'alternative, vient du fait que dans une boucle, le concept de variable informatique apparait différent de celui de variable mathématique.

Nous pouvons ainsi identifier trois opérations sur les variables dans une boucle, renvoyant à trois aspects cognitifs : *la mise à jour*, *le test d'arrêt* et l'*initialisation*. Comme le souligne Samurçay (1985), l'élève peut donner du sens aux deux types de variables relatives à des problèmes de programmation, celles qui sont des données explicites du problème et celles qui sont rendues nécessaire par le travail dans le domaine informatique et le langage syntaxique de l'environnement numérique utilisé. De plus, l'hypothèse que des élèves

débutants en informatique privilégient la lecture à l'affectation pour l'initialisation, va être aussi vérifiée et validée. Samurçay (1985) souligne que ce fait est probablement dû à une meilleure compréhension de l'instruction de lecture de la part des élèves débutants ainsi que par un besoin d'instruction de communication.

Pour finir sur cette ingénierie « dichotomie », lors du travail mis en place sur l'algorithme de « dichotomie continue », les élèves peuvent aussi avoir une approche de type « analyse numérique » de cet algorithme leur permettant de le voir comme objet d'apprentissage mathématique contrairement à ce qui est présenté dans les manuels (cf. chapitre 5) où l'utilisation de cet algorithme est du type « application numérique ».

En effet, nous souhaitons que les élèves dans le cas « continu » mettent aussi en jeu la compréhension de la méthode de dichotomie comme moyen d'obtenir une approximation d'un antécédent par une fonction donnée et prennent ainsi conscience de la propriété des valeurs intermédiaires.

De plus, l'élève prend conscience que dans le cadre d'une approche « analyse numérique » du cas « continu », il est nécessaire de se questionner sur le domaine de validité de l'algorithme et d'identifier des conditions suffisantes pour la validité des résultats renvoyés par la machine après l'implémentation de l'algorithme et donc par conséquent pour la propriété des valeurs intermédiaires. Nous observons ainsi que de nombreuses articulations entre *ETM* et *ETA* peuvent être mises en place chez les élèves, en particulier quand les compétences maitrisées par ces derniers dans le domaine mathématique, ici l'analyse, sont importantes. Le fait de procéder à une ingénierie sur les trois années de lycée permet de mieux comprendre et affiner ces articulations entre *ETM* et *ETA*. Cependant, l'analyse des deux dernières phases de l'ingénierie portant sur l'introduction d'un nouveau savoir que sont les suites adjacentes, mais aussi la construction d'une preuve du *théorème des valeurs intermédiaires* basée sur la dichotomie et les suites adjacentes, nous montrent que les élèves sont encore dans une approche incomplète sur les liens qui peuvent y avoir entre raisonnements de nature algorithmique et raisonnements de nature mathématique en particulier quand ils se trouvent libre dans leur choix du raisonnement à mettre en place.

2. Une ingénierie sur la modélisation et une double approche : « fréquentiste » et « probabiliste », dans le cadre d'une *politique des naissances*

Suite à la lecture des travaux menés par Briant (2013) sur la nouvelle place de l'algèbre se situant désormais dans le domaine fonctionnel, et l'introduction d'un enseignement de l'algorithmique, permettant de lier ces deux sujets autour d'une étude didactique de la reprise de l'algèbre élémentaire en classe de Seconde, en particulier l'étude d'*objets gravitant autour du concept d'équation* (Briant, 2015), objets dont Briant cherche *à affiner le sens par le détour de l'algorithmique* (Ibid.), nous nous sommes intéressés à la place de l'algorithmique dans l'enseignement des probabilités autour de la construction de modèles permettant de simuler un processus aléatoire, implémentable dans des environnements numériques.

Pour cela, nous nous appuyons sur le fait que les statistiques et les probabilités enseignées dans les classes de fin du cycle 4 du collège et des années de lycée constituent un domaine où les phénomènes aléatoires issus de situations extra-mathématiques jouent un rôle fondamental. Les programmes depuis l'introduction de ces domaines mathématiques insistent sur une approche « fréquentiste » à travers une observation de fréquences relatives de caractères représentatifs des phénomènes aléatoires permettant le passage à la notion de probabilité théorique d'événements, ainsi que la détermination de moyennes statistiques prépare au concept d'espérance mathématique d'une variable aléatoire.

Favoriser une approche « fréquentiste » de processus aléatoire dans les classes du secondaire français aide aussi à mieux comprendre le passage du monde « réel » au monde « mathématique ». En effet, une telle approche favorise la construction de modèles de simulations sur ordinateurs. C'est un moyen pratique d'obtenir des séries statistiques de taille suffisante pour observer la convergence des fréquences relatives et des moyennes statistiques. Ces simulations issues de situations extra-mathématiques amènent l'élève à élaborer une modélisation du processus aléatoire étudié. Selon Kiet (2015), le modèle ainsi obtenu permet à l'élève de mieux donner du sens aux calculs de probabilités théoriques. De plus, en lien avec notre cadre théorique sur les articulations possibles entre *ETM* et *ETA*, ainsi que sur les niveaux de paradigmes algorithmiques mis en place par les élèves, l'élaboration d'une ingénierie sur un processus aléatoire nous permet d'observer chez l'élève les étapes

nécessaires pour la construction du modèle représenté par un algorithme lors des phases de *cycle de modélisation* (Blum & Leiss, 2005).

Ainsi, lors de l'analyse du travail des élèves (cf. chapitre 6), nous observons une diversité riche entre les différentes genèses et les plans verticaux des modèles des *ETM* et *ETA*. Cet aspect signifie que diverses dimensions du travail liées aux outils, aux techniques et aux propriétés sont prises en compte. Ce point reflète la manière dont différents contextes de travail sont impliqués pendant l'activité de l'élève.

Nous observons aussi au cours de la dernière phase de type implicite (en effet, les élèves devaient avoir l'initiative de la mettre en place), une majorité des élèves ayant acquis une certaine « aisance » dans l'*ETA* ont conscience que le travail mené dans cet *Espace de Travail* peut ne pas être suffisant pour valider les prévisions envisagées par les auteurs de la *politique des naissances*. Ainsi, nous constatons qu'une grande majorité des élèves vont compléter d'eux-mêmes le travail mené sur des fréquences et des moyennes avec des échantillons plus ou moins grands dans le cadre d'un *ETA* associé à une approche « fréquentiste », par une approche « probabiliste » de la problématique dans le cadre d'un *ETM*. Ainsi, sans avoir des connaissances plus théoriques sur les statistiques et les probabilités, comme la loi des grands nombres, nous observons que des élèves ayant une maitrise des outils informatiques (en effet, ces élèves ont participé à l'ingénierie sur la dichotomie), ont conscience des articulations nécessaires entre *ETA* et *ETM* pour valider ou non des hypothèses faites dans le cadre d'un *ETA*. Nous revenons sur ce point dans la section suivante.

3. L'introduction de l'*ETA* : un réel bénéfice pour favoriser l'accès à de nouvelles compétences mathématiques.

Nous souhaitons conclure cette seconde partie de nos travaux de recherche par un fait qui concerne l'élaboration des deux dernières ingénieries didactiques. En effet, nous rappelons que la dernière ingénierie dans le domaine des probabilités/statistiques est expérimentée par deux des classes ayant participé à l'ingénierie précédente sur la dichotomie.

Bien que les domaines mathématiques soient différents, nous rappelons que nous avons fait le choix que ces deux ingénieries sur la dichotomie et la *politique des naissances* mettent en place des expérimentations qui doivent utiliser les mêmes environnements numériques.

Ainsi, nous constatons que les difficultés que nous avions supposées possibles lors de l'analyse *a priori* de la dernière ingénierie en lien avec les langages pseudo-code et machine, ne se révèlent pas exactes lors de la construction de modèles algorithmiques pour la simulation du processus aléatoire donné par des élèves, pourtant relativement débutant en informatique. Nous supposons que ce choix fait sur les classes où sont expérimentées nos deux ingénieries est en partie à l'origine des résultats observés dans la dernière ingénierie (cf. Chapitre 6).

En effet, comme nous l'avons mentionné dans le chapitre 6, très peu d'algorithmes rendus par les élèves ont témoigné de réelles difficultés dans l'écriture et la syntaxe de ces algorithmes, même si le choix des structures informatiques retenues par les élèves n'était pas toujours pertinent, en particulier pour les élèves ayant choisi de transposer le travail fait lors de la construction d'un arbre de probabilité à un algorithme permettant de simuler la constitution de la fratrie d'une famille, que cet algorithme soit donné sous forme « spatial » avec la construction d'un organigramme ou « textuel » avec l'écriture en langage pseudo-code implémentable dans une machine. Ainsi, au fur et à mesure, de l'avancé du travail de la part d'élèves initialement débutants en informatique, nous observons que la maitrise de l'outil informatique à travers l'algorithmique et la programmation, permet aux élèves de mieux définir leur travail dans le domaine mathématique étudié et favorise chez ces élèves la possibilité de mettre en place de façon plus « naturelle » la nécessité de justifier une conjecture, voire une preuve des résultats observés dans un *ETA*. Pour conclure, nous observons ainsi l'importance de développer les articulations et les interactions entre *ETA* et *ETM* spécifiques aux différents niveaux scolaires du secondaire second cycle français.

Conclusion : Quels enseignements tirons-nous de ce travail de recherche ?

- Pourquoi un nouveau cadre théorique ?
- Un travail spécifique sur la *théorie élémentaire des nombres*
- Des travaux spécifiques au domaine d'analyse
- Un travail spécifique à une simulation aléatoire : une *politique des naissances*
- Les ingénieries sur la dichotomie et la simulation sont pratiquées dans les mêmes classes. Quelles sont les conséquences de ce choix ? Quels enseignements en tirons-nous de ce choix ? Pour les élèves ? Pour les pratiques enseignantes ?
- Pensée mathématique / Pensée algorithmique
- Les limites observées au cours de notre recherche – Quelles perspectives pouvons-nous envisager à ce travail de recherche ?
- Conclusion

Dans le cadre de l'introduction d'un enseignement de l'algorithmique au niveau des classes du lycée depuis le début des années 2010, la problématique qui est à l'origine de nos travaux de recherche, a été la contribution de l'algorithmique chez l'élève débutant en informatique aux apprentissages dans les différents domaines mathématiques enseignés et au développement de savoirs spécifiques. En effet, les *nouveaux programmes de lycée ont fixé des objectifs précis en matière d'algorithmique. L'algorithmique* est un champ transversal du lycée, permettant aux élèves d'apprendre à construire une démarche scientifique. Ainsi, d'après les auteurs de ces programmes autour de l'enseignement algorithmique, trois objectifs fondamentaux émergent :

- L'approfondissement des bases de la logique et du raisonnement ;
- L'illustration des concepts enseignés par l'utilisation d'outils informatiques ;
- Le développement chez les élèves d'un esprit de créativité et d'initiative au travers de l'expérimentation.

Nous sommes alors partis du constat qu'à travers cette lecture des programmes des trois années du lycée, l'enseignement de l'algorithmique apparaît comme un outil permettant de donner du sens à un certain nombre de notions étudiées. Cependant, dans la continuité des travaux de Modeste (2012) et de Briant (2013), nous nous demandons comment dépasser ce stade pour que l'algorithmique, dans le cadre d'une dialectique « *outil-objet* » au sens de Douady, devienne un objet d'apprentissage dans des domaines mathématiques spécifiques.

1. *Pourquoi un nouveau cadre théorique ?*

Pour cela, le passage de l'algorithme *outil* d'apprentissage à l'algorithme *objet* d'apprentissage nécessite d'introduire un nouveau cadre théorique. Nous analysons nos travaux à travers celui-ci. Les différentes lectures sur les cadres théoriques de la didactique existant à ce jour, nous ont permis de nous pencher plus précisément dans une première phase sur les ETG[117] et les niveaux de paradigmes de géométrie qui sont introduits en 2006 par Houdement et Kuzniak. En effet, dans le cadre théorique développé pour l'étude de la géométrie enseignée et dont la finalité sur le long terme est de construire une didactique de la géométrie pour la formation des enseignants (PE et PLC), Kuzniak propose la genèse et des éléments de ce nouveau cadre théorique, partant du fait que l'approche de la didactique de

[117] Espace de Travail Géométrique

la géométrie élémentaire est basée *sur une explicitation et un jeu entre différents paradigmes géométriques : la Géométrie I (géométrie naturelle), la Géométrie II (géométrie axiomatique naturelle) et la Géométrie III (géométrie axiomatique formaliste)* (Kuzniak, 2003). Selon Kuzniak ce qui importe, *c'est l'existence de plusieurs approches cohérentes de la géométrie élémentaire vue comme une théorisation de l'espace. L'activité géométrique se déploie dans un espace particulier : l'espace de travail de la géométrie. Cet espace s'organise autour de trois composantes : l'espace support, le modèle théorique et les artefacts. Cette organisation dépend de la géométrie de référence mais aussi de l'utilisateur* (Ibid.). L'Espace de Travail ainsi conçu désigne un environnement pensé et organisé pour permettre le travail d'élèves qui ne sont pas des experts en géométrie mais devant résoudre des problèmes scolaires de géométrie.

En 2011, Kuzniak propose de généraliser l'*Espace de Travail Géométrique* à la notion d'*Espace de Travail Mathématique*. Il l'introduit à partir de certaines caractéristiques que les études sur le travail géométrique ont permis de dégager.

Ainsi, nous avons deux niveaux fondamentaux qui *structurent l'Espace de Travail Mathématique : un niveau épistémologique qui s'attache au contenu mathématique et un niveau cognitif relié aux processus de visualisation, de construction et de preuve. Pour articuler ces deux niveaux et permettre la réalisation du travail mathématique, trois genèses principales sont retenues : une genèse sémiotique, une genèse instrumentale et enfin une genèse discursive supportant le raisonnement* (Kuzniak, 2011).

En 2014, Kuzniak et Richard approfondissent ce modèle théorique et en explorent les utilisations comme outil d'analyse dans des domaines mathématiques spécifiques. Pour cela ce modèle suppose une étude approfondie et précise des domaines mathématiques spécifiques et nous conduit à examiner des *ETM*[118] pouvant s'appliquer à des spécificités mathématiques comme l'arithmétique, l'analyse, les probabilités… Nous les notons ETM_s.

Selon Kuzniak et Richard (2014), l'*ETM* peut aussi être vu *comme une mise en réseau des diverses fibres que constituent les* ETM_s. […] *Ces interactions entre les domaines* spécifiques *sont essentielles pour comprendre le fonctionnement global du travail mathématique et elles*

[118] Espaces de Travail mathématique

forcent en outre la considération des processus de modélisation dans le cadre des ETM, au-delà des seules questions sémiotiques.

Ainsi, tenant compte du fait qu'un programme de construction d'une figure géométrique peut être considérée comme un algorithme où la succession des différentes étapes de construction de la figure permettent d'aboutir à la figure voulue, et ceci autant de fois que nous appliquons ce programme de construction, nous sommes partis de l'hypothèse que le cadre des *Espaces de Travail Géométrique* et leurs genèses, ainsi que les paradigmes géométriques peuvent être adaptés à des *Espaces de Travail Algorithmique* (ETA) et à des niveaux de paradigmes algorithmiques (Partie 1, chapitre 2).

Nous partons aussi de l'hypothèse que la mise en place d'*ETA* aide à analyser les positions respectives des élèves et des enseignants, sous l'influence des programmes. En effet, comme pour les *ETM*, l'*ETA* « paradigmatique » tel que le défini Kuzniak et Richard (2014) est appelé *ETA* de référence. En revanche, dans l'institution scolaire qu'est le lycée, la résolution d'un problème dans un domaine mathématique spécifique va supposer qu'un *ETA* idoine puisse être organisé afin de permettre à l'élève débutant en informatique de s'engager dans la résolution de ce problème. Comme pour les *ETM*, cet *ETA* idoine doit satisfaire deux conditions : *d'une part permettre de travailler dans le paradigme correspondant à la problématique visée, d'autre part être « bien construit », dans le sens où ses différentes composantes sont organisées de manière valide* (Kuzniak & Richard, 2011). Le concepteur joue ici aussi un rôle semblable à celui de l'informaticien qui conçoit un *Espace de Travail* pour des utilisateurs potentiels n'ayant pas nécessairement une maitrise des outils fournis par l'environnement numérique. Ainsi dans la classe, la conception de l'*ETA* va dépendre de l'*ETA* personnel de l'enseignant. Nous référant toujours aux travaux de Kuzniak et Richard, lorsqu'un problème dans un domaine mathématique spécifique est proposé à un élève, son traitement mathématique par une approche algorithmique par l'élève est conduit dans l'ETM_s et l'*ETA*, ainsi que dans les *ETM* et *ETA* personnels de l'élève. De ce fait, les *ETM* et *ETA* idoine ne sont pas figés. Ils doivent sans cesse être modifiés pour s'ajuster aux contraintes locales. De plus, *l'ETA idoine prévu par l'enseignant n'est pas nécessairement celui de l'élève* (Laval, 2015).

Ainsi, les travaux mathématiques et algorithmiques peuvent, dans le cadre du lycée, être décrits grâce à trois niveaux d'*ETM/ETA* : le domaine mathématique spécifique et l'utilisation de l'algorithmique visés par l'institution doit être décrite dans l'*ETM* de référence. Ceci explique notre choix de présenter pour chaque domaine mathématique où ont lieu nos ingénieries didactiques, des analyses des parties de manuels et de documents d'accompagnement en lien avec les tâches demandées dans l'ingénierie. Celui-ci est aménagé par l'enseignant en *ETM/ETA* idoines pour permettre une mise en place effective dans les classes, où chaque élève travaille dans son *ETM/ETA* personnels.

Le projet de recherche de cette thèse se situe donc dans le cadre d'apprentissages de compétences sur l'utilisation et la construction d'algorithmes dans l'enseignement des mathématiques des classes du lycée. Pour cela nous construisons trois ingénieries didactiques expérimentées dans des classes de Seconde et du cycle Terminal Scientifique. Trois domaines mathématiques spécifiques sont retenus : la *théorie élémentaire des nombres* au niveau de la Terminale Scientifique, l'analyse aux niveaux de la Seconde et des classes du cycle Terminal Scientifique et les probabilités/statistiques au niveau de la Seconde et de la Première Scientifique.

Nous souhaitons aussi à travers l'analyse des activités des élèves lors de chacune de ces ingénieries, aborder l'hypothèse que l'algorithmique et la programmation font parties de l'activité humaine (Lagrange, 2016). En effet, quand nous mettons en place des activités nécessitant le recours à la programmation, il semble utile que l'élève ait une réflexion de nature algorithmique. De même, quand nous nous intéressons aux questions centrales de l'algorithmique que sont la *terminaison*, l'*effectivité* et la *complexité,* la programmation peut constituer un champ expérimental utile, justifiant ainsi le fait que les élèves soient amenés à expérimenter des travaux sur des algorithmes « papier-crayon » dans des environnements numériques permettant un travail plus approfondi sur des algorithmes écrits au format « spatial » avec les organigrammes, ou « textuel » avec un langage pseudo-code.

*Lors des deux premières ingénieries (« algorithme de Kaprekar » et « algorithme de dichotomie »), nous désirons étudier dans le cadre des articulations possibles entre ETM$_s$ et ETA, le passage d'un enseignement de l'algorithmique outil vers un enseignement de l'algorithmique objet au sens de Douady (1986). Ainsi, d*ans la continuité des travaux menés par Modeste (2012), nous nous sommes intéressés à la problématique de l'enseignement de

l'algorithme dans les classes de Seconde et du cycle scientifique Terminal, avec l'introduction de l'algorithmique, mais avant tout pour son lien étroit au concept de « preuve ». Pour cela, nous avons mené deux recherches, une première sur l'élaboration d'une « preuve explicative » d'une *conjecture dans le domaine de la théorie élémentaire des nombres avec l'algorithme de Kaprekar, suivie d'une seconde sur la construction d'une « preuve formelle » du Théorème des valeurs intermédiaires (TVI) basée sur la méthode de dichotomie et les suites adjacentes.*

Lors de la troisième et dernière ingénierie didactique (une *politique des naissances*), nous étudions les démarches mises en place pour l'étude et la construction d'algorithmes par les élèves dans un cadre plus général d'une démarche de modélisation d'un processus aléatoire. Ainsi, nous utilisons ici aussi les *ETM* et les *ETA* comme outil d'analyse du travail tant mathématique qu'algorithmique lors des différentes phases du *cycle de modélisation* (Blum & Leiss, 2005). En effet, certaines étapes de modélisation sont nécessaires lors de la construction d'un modèle permettant de simuler le processus aléatoire défini par une politique des naissances. Pour cela, nous utilisons comme outil d'analyse des travaux mathématique et algorithmique pratiqués par les élèves lors des différentes phases du cycle de modélisation, le modèle suivant :

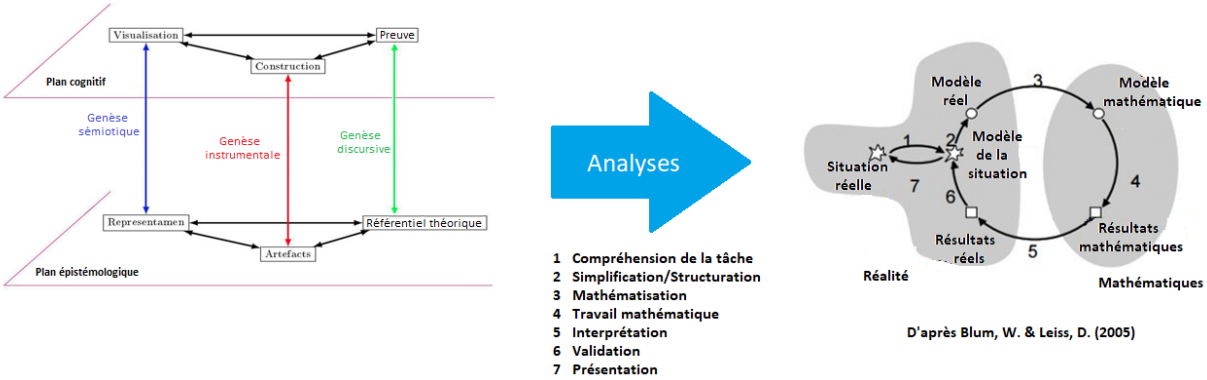


Figure 207 (ETM/ETA associés au cycle de modélisation de Blum et Leiss (2005))

A travers les analyses faites autour de ces trois ingénieries, nous étudions l'effectivité et l'écologie d'un tel enseignement de l'algorithmique dans le cadre institutionnel français du point de vue des apprentissages réalisés par les élèves, pour inférer des résultats plus généraux sur l'algorithmique et en moindre mesure sur son apport à la modélisation dans l'enseignement des mathématiques pour les classes du lycée.

La méthodologie choisie au départ permet d'obtenir des données globales et d'observer finement les activités des élèves en classe et les pratiques des enseignants, mêmes si ces dernières ne font pas partie intégrante de notre dynamique de recherche.

2. Un travail spécifique sur la *théorie élémentaire des nombres*

L'expérimentation portant sur la *théorie élémentaire des nombres* lors d'une séquence en enseignement de spécialité, permet de montrer que des élèves de Terminale Scientifique qui ont su donner du sens, lors d'une première ingénierie, sur un travail de planification d'un algorithme complexe (Guy, Lagrange, 2013) permettant de modéliser un processus de calcul sur des nombres entiers de 000 à 999 (cf. Partie 2 – Chapitre 2), émettent une conjecture sur les résultats numériques dus à l'exécution de ce processus. Dans la continuité de ce travail, nous observons alors que des élèves de ce niveau ne se contentent pas d'émettre une conjecture, mais conformément au contrat didactique mis en place à ce niveau scolaire, désirent prouver cette conjecture qui fait l'objet de notre ingénierie (Cf. Chapitre 2).

Pour cela, les élèves construisent une première preuve de type heuristique ou encore « non explicative » en modifiant l'algorithme complexe par l'ajout d'une structure répétitive « POUR » permettant d'exprimer la répétition du processus de calcul un nombre fini de fois, représentant l'ensemble des nombres entiers compris entre 000 et 999.

Une fois que cette première preuve est obtenue, le chercheur propose aux élèves la lecture d'un article sur la résolution du *Rubik's Cube* où il est question du nombre minimal de mouvements pour résoudre n'importe quelle configuration du célèbre casse-tête. Le chercheur attend de cette lecture que les élèves prennent conscience que des chercheurs issus du monde des mathématiques et de l'informatique ont pu montrer qu'un travail de réflexion de type mathématique pouvait permettre de réduire le nombre réel d'essais nécessaires à faire à l'aide d'un environnement numérique pour justifier de la validité de ce nombre minimal de mouvements.

Ainsi, les élèves peuvent prendre conscience qu'une preuve « non explicative » peut s'avérer difficile d'un point de vue temporel à mettre en place à l'aide de la seule utilisation d'un ordinateur. En effet, nous nous sommes appuyés sur le fait que le niveau de connaissances de la part des élèves, observés au niveau de l'abstraction et du raisonnement en mathématique, peut donner du sens à la lecture de l'article en transposant l'idée centrale

de la problématique de « réduction des cas » à l'idée d'une preuve « explicative » sur un ensemble de nombres entiers à tester.

Dans le cadre de notre recherche, cette ingénierie nous permet aussi de témoigner des différents niveaux de paradigmes algorithmiques, ainsi que des différentes genèses pouvant apparaître tant dans un ETM_s que dans un *ETA*, et ainsi d'étudier les articulations possibles entre l'ETM_s et l'*ETA*. En effet, ces articulations *ETA*/ETM_s participent à l'élaboration de la « pensée mathématique » chez l'élève car bien que le travail autour de la preuve se fasse pour l'essentiel dans un *ETM*, l'*ETA* reste continuellement « en toile de fond ». Nous constatons que l'élève comprend qu'il mène un travail en mathématiques, et que l'*ETA* lui sert de référence en ce qui concerne les représentations sémiotiques (au sens de Duval) du nombre entier. Ainsi, les représentations sémiotiques sont définies comme des productions constituées par l'emploi de signes, d'écritures appartenant à un système de représentations qui a ses propres contraintes de signifiance et de fonctionnement que ce soit dans le domaine de la *théorie élémentaires des nombres* ou le domaine de l'algorithmique. Par cela, l'élève renforce sa prise de conscience que l'objet mathématique « nombre entier » peut avoir plusieurs représentations sémiotiques.

3. Des travaux spécifiques au domaine d'analyse

Dans cette seconde ingénierie expérimentée sur les trois niveaux du lycée : Seconde et cycle terminal Scientifique, les élèves ont pour objectif principal de construire une preuve formelle du TVI à l'aide de la dichotomie et des suites adjacentes. Cependant, afin de faciliter le travail en autonomie de l'élève, nous montons une première expérimentation sur la dichotomie mettant en place un jeu auquel les élèves jouent en binômes. Lors du travail en autonomie complète sur le jeu « *Trouver une stratégie rapide et gagnante pour déterminer un nombre entier secret compris entre deux nombres entiers connus* », les élèves prennent conscience que la méthode de dichotomie répond à cette problématique concernant la « rapidité » sans toutefois aborder la notion d'optimalité qui, elle, n'est pas institutionnalisée au niveau des classes du lycée.

Une fois que l'ensemble de la classe accepte de prendre la dichotomie comme stratégie « gagnante » et « rapide », les élèves étudient les algorithmes correspondants aux deux joueurs : le joueur qui pense au nombre entier secret, puis le joueur qui propose des nombres entiers afin de trouver l'entier « secret » en un minimum de coups. Pour cela, nous voyons

que les élèves doivent étudier les structures des algorithmes correspondant à chacun des joueurs, ainsi que l'organisation des variables informatiques, en particulier l'initialisation de ces variables quand cela est nécessaire.

De plus, ce travail dans le cas « discret » permet aux élèves de construire un test de sortie de boucle autour d'une égalité à un nombre entier (le nombre entier « secret »). Cette approche leur permet aussi d'acquérir des compétences sur la partie entière d'un nombre réel, en particulier en Seconde et en Première. En effet, des élèves de ces niveaux scolaires peuvent être amenés, comme nous avons pu l'observer, à se référer à des résultats sur le concept d'approximation vue en mathématiques mais aussi dans d'autres disciplines comme la physique. Ainsi, nous observons que des élèves de ce niveau scolaire vont d'abord penser en termes d'approximation par « estimation ». Il suppose que pour chaque calcul où le résultat n'est pas un nombre entier, il est possible de prendre une valeur entière qu'on peut juger suffisamment près d'une valeur entière proposée par la machine lors de l'action par celle-ci du joueur qui doit proposer des nombres entiers. Les élèves ont conscience que cette valeur peut-être difficile à mesurer sans des conditions plus appropriées et en particulier sans l'utilisation d'un objet mathématique permettant d'obtenir ces valeurs entières. Pour cela, nous observons que l'enseignant peut être amené à intervenir afin d'utiliser le savoir des élèves pour les conduire à une nouvelle compétence qu'est ici la fonction partie entière.

Lors du passage à l'algorithme de « dichotomie continu », les élèves ont la possibilité de réutiliser le travail fait sur l'algorithme du joueur proposant des nombres entiers dans le cadre d'une série d'essais qui se termine lors de l'égalité entre le nombre entier proposé et le nombre entier « secret ». En effet, le fait de partir de la structure mise en place pour ce joueur en tenant compte des adaptations nécessaires à mettre en place, aide l'élève à construire un algorithme de dichotomie permettant d'obtenir une valeur approchée d'une solution d'une équation de la forme $f(x) = 0$. Mais, nous référant aux exercices proposés par les manuels, nous constatons que cet algorithme de dichotomie ne trouve qu'une place limitée dans les utilisations possibles. Ceci nous renvoie à la problématique autour de la dialectique *outil/objet* au sens de Douady (1986) qui a été l'objet des travaux de recherche de Modeste (2012) et Briant (2013).

En effet, dans un premier temps nous observons que des élèves de Seconde, et en moindre mesure de Première Scientifique, éprouvent des difficultés à interpréter des résultats renvoyés par la machine dans le cas de l'utilisation d'un algorithme de dichotomie « continue » implémenté dans un environnement numérique sans qu'il y ait la connaissance par l'utilisateur des propriétés vérifiées par la fonction sur laquelle on teste l'algorithme dans un intervalle donnée. En effet, cet aspect permet de donner du sens à l'importance des conditions de validité du *théorème des valeurs intermédiaires* (TVI) et ainsi de la mise en pratique de l'algorithme de dichotomie comme objet d'« analyse numérique ».

Nous observons par exemple qu'au niveau de la Seconde, les élèves n'ont pas nécessairement un esprit critique sur les résultats obtenus quand on teste l'algorithme. En effet, si celui semble renvoyer une valeur numérique finie, les élèves de Seconde et de Première interprètent de façon erronée ce résultat comme étant l'unique solution de l'équation de la forme $f(x) = 0$ dans l'intervalle, sans savoir s'il y a bien unicité. De plus, nous voyons que dans le cas où les valeurs obtenues par la machine peuvent être interprétées comme un problème asymptotique, nous observons que, suivant les compétences mathématiques, les élèves vont montrer des réactions différentes.

Ainsi, en Seconde nous pouvons voir des élèves qui interprètent ces résultats comme des encadrements d'une solution, tandis qu'au niveau de la Première, bien que les élèves n'aient pas encore les compétences mathématiques permettant d'approfondir ce questionnement, nous observons qu'un certain nombre parmi eux s'interroge sur la validité de ces valeurs renvoyées par la machine.

En revanche, au niveau de la Terminale Scientifique, nous observons que les élèves ont conscience que l'utilisation de la dichotomie doit se faire après justification de la validité des conditions permettant une utilisation du TVI. De plus, ces élèves de Terminale ayant des compétences sur des problèmes d'analyse de nature asymptotique, nous observons qu'ils proposent une interprétation différente de celle des élèves des classes antérieurs. En effet, ils interprètent ces valeurs comme signifiant que la fonction peut ne pas être définie sur tout l'intervalle étudié. Par exemple, la fonction pourrait admettre une valeur interdite qui justifierait des résultats de type asymptotique que la machine renverrait. Ainsi, ils font le lien

entre la vérification des conditions de validité du TVI et la possibilité d'utiliser à bon escient l'algorithme de dichotomie.

Nous observons aussi lors de ce travail fait sur la dichotomie, que les élèves de Seconde et de Première sont restés plus dans des genèses instrumentales et discursives en lien avec un *ETA* que les élèves de Terminale qui se sont plus positionnés sur des articulations possibles entre $ETM_{analyse}$ et *ETA*.

Lors de l'élaboration d'une preuve du TVI (phase qui n'a été expérimentée que par les élèves de Terminale), nous avons constaté que les élèves de Terminale ont pu faire le lien entre le concept de suites adjacentes et la construction des suites des bornes inférieures et supérieures des intervalles emboités. Ainsi, à nouveau les élèves ont su profiter du travail fait dans un *ETA* en particulier dans le plan instrumental-discursif pour travailler une genèse discursive dans l'*ETM* permettant d'élaborer une preuve du TVI à partir de la dichotomie et des suites adjacentes.

4. Un travail spécifique à une simulation aléatoire : *une politique des naissances*

Cette troisième et dernière ingénierie, mise en place aux niveaux des classes de Seconde et de Première Scientifique, nous permet d'étudier l'introduction d'une loi discrète par une première approche d'une situation du « monde réel » à travers le ressenti des élèves autour d'une politique nataliste proposée par un gouvernement fictif.

Nous constatons que les élèves des deux niveaux scolaires pour interpréter cette politique proposent un arbre de probabilité sans que pour autant ils aient marqué les valeurs des probabilités de naissance de chaque sexe. Un tel arbre est souvent le point de départ d'une approche « fréquentiste » du problème.

En effet, une majorité d'élèves, en particulier au niveau de la Seconde propose de construire un algorithme permettant de simuler les fratries possibles en proposant une succession de « Si…Alors…Sinon » qui est finalement une transposition de l'arbre de probabilité à l'algorithme. Nous pouvons supposer que le fait d'avoir limité le nombre maximal d'enfants dans une famille à 4, n'amène pas les élèves à prendre du recul quant à la conception de l'algorithme. Pour cela, prendre un nombre plus important pour ce nombre maximal aurait

probablement permis à ces élèves qui ont déjà participé à l'ingénierie sur la dichotomie de prendre l'initiative d'envisager une autre structure informatique du type « TantQue » permettant d'éviter la problématique de la gestion de complexité de représenter une succession de « Si… Alors … Sinon » qui devient vite « trop lourd » si le nombre maximal d'enfants est assez important.

Ainsi, nous observons que certains élèves de Première dans le cadre de cette approche « fréquentiste » envisage la possibilité de construire un algorithme qui ne serait pas basé sur cette succession d'instructions conditionnelles.

De plus, nous observons que le travail fait lors de l'ingénierie sur la dichotomie permet de faire que les élèves n'éprouvent pas de difficulté particulière pour générer des nombres aléatoires de type 0 ou 1 permettant de donner le sexe de l'enfant qui nait. En revanche, nous constatons que des difficultés sur ce point peuvent apparaitre s'il n'y a pas une situation d'équiprobabilité sur le sexe des enfants lors des naissances. Cette difficulté ne concerne que la construction d'un algorithme en langage pseudo-code implémentable dans un environnement numérique. Elle ne semble pas apparaitre lors du travail papier-crayon.

Nous constatons aussi que la phase où les élèves doivent aborder l'approche « fréquentiste » à l'aide du tableur ne pose pas de difficultés majeures auprès des élèves. Nous pouvons supposer que le fait que cet environnement numérique est pratiqué régulièrement dans un certain nombre de disciplines, favorise une maitrise correcte des fonctionnalités offertes par le tableur et en particulier une présentation sous forme d'un tableau. En revanche, nous constatons que l'utilisation du tableur ne favorise pas la construction d'un algorithme en langage pseudo-code, en particulier la construction des tests d'arrêts représentant le nombre maximal d'enfants dans une famille et le nombre total de familles à simuler. En effet, dans le cadre du tableur, ces tests se font manuellement par des « copier-coller ».

De plus, nous constatons que l'approche « fréquentiste » du processus aléatoire permet aux élèves de mieux déterminer un modèle mathématique ou algorithmique lors du passage du monde « réel » au monde de la simulation. Conformément, aux travaux de Kiet (2015) nous constatons aussi que des simulations de processus aléatoires issues de situations extra-mathématiques peuvent conduire l'élève des classes de lycée à mieux observer les

convergences entre les moyennes statistiques et les fréquences relatives, et ainsi donner du sens aux calculs de probabilités théoriques.

5. Les ingénieries sur la dichotomie et la simulation sont pratiquées dans les mêmes classes. Quelles sont les conséquences de ce choix ? Quels enseignements en tirons-nous de ce choix ? Pour les élèves ? Pour les pratiques enseignantes ?

Lors de l'ingénierie sur la simulation aléatoire, nous avons observé un fait que nous avons déjà abordé dans la conclusion de la partie 2 et sur lequel nous souhaitons revenir ici en particulier pour étudier les conséquences de mener des apprentissages de nouvelles compétences mathématiques avec des élèves débutants en informatique, à travers des interactions possibles entre *ETA* et *ETM*, et leurs genèses mais aussi sur les pratiques enseignantes lors de travaux algorithmiques quel que soit le domaine mathématique enseigné au lycée. Ceci peut s'inscrire dans l'approche faite par Briant (2013) sur les apports de l'algorithmique à l'algèbre élémentaire, en particulier la résolution d'équations algébriques. En effet, dans la conclusion de sa thèse, Briant signale que l'expérimentation qu'elle a menée sur un travail spécifique à l'algèbre à travers une approche algorithmique, lui a permis de remarquer que les élèves du lycée peuvent *améliorer ou du moins reprendre leur conceptualisation des objets de l'algèbre gravitant autour du concept d'équation, même si des écarts importants subsistent dans les évolutions de chacun des élèves* (Briant, 2013) et ceci grâce à l'utilisation d'algorithmes. Le fait que son expérimentation ait consisté à réaliser des tâches non routinières lui a permis d'aborder un enseignement de l'algèbre « différemment » et permettre ainsi aux enseignants qu'ils puissent *sortir de leurs pratiques ordinaires* (Ibid.). Par exemple dans une de ses situations, les élèves n'ont pas pour consigne de résoudre des équations mais de les classer. De même dans d'autres situations, les élèves ne résolvent toujours pas des équations particulières comme ils ont l'habitude de le faire. En effet, ils ont pour consigne de déterminer *un modèle de résolution qui* s'appliquerait *à toute une famille d'équations* (Ibid.). Ainsi, les élèves et leurs enseignants *« regardent » les équations avant de chercher à les résoudre. Ils s'attachent à considérer chaque composant d'une équation, les coefficients en présence, la place de l'inconnue, le signe d'égalité, les transformations possibles*

à effectuer (Ibid.). Briant constate aussi que *deux aspects d'une équation sont mis en relation lors de la tâche de conception d'un algorithme de résolution d'une catégorie d'équations : l'aspect structural, lorsque qu'est déterminée la forme générique de l'équation et l'aspect procédural lorsqu'est déterminée sa technique de résolution. Ces allers-retours entre les deux aspects sont un point-clef de* ses *expérimentations*. Elle considère en tant que chercheuse que *la considération d'une catégorie d'équations sous leur forme générique, avec l'introduction de paramètres, est un type de tâches d'un niveau supérieur de conceptualisation*. Elle parle ainsi de *réification d'un concept* (au sens de Sfard (1991)) où celle-ci deviendrait possible quand sont proposées des tâches de niveaux supérieures. Briant conclut sur le fait que faire travailler des élèves sur *les objets d'une équation* les aide à mieux *comprendre les techniques de résolution* d'une équation.

Partant de cette conclusion, nous essayons de voir si ce constat fait en algèbre par Briant, peut s'appliquer aussi dans d'autres domaines spécifiques des mathématiques enseignés dans les classes de Seconde et du cycle Terminal Scientifique. Pour cela, les conditions dans lesquelles s'est déroulée la dernière ingénierie, permettent d'aborder ce point sur la réification de concepts mathématiques. Nous rappelons que la dernière ingénierie se fait dans certaines des classes ayant participé à l'ingénierie « globale » sur la dichotomie. Le fait de garder des classes sur plusieurs ingénieries nous permet de mieux voir les progrès possibles chez l'élève de Seconde et de Première pour aborder des tâches de niveaux supérieures afin de mieux comprendre les enjeux d'une loi discrète par exemple. En effet, ces classes ayant acquis des connaissances suffisantes dans le domaine de l'algorithmique sur les langages informatiques et les représentations « textuelle » et « spatiale » d'un algorithme, peuvent centrer leurs efforts sur le modèle de la simulation du processus aléatoire lors de l'ingénierie sur la *politique des naissances*. Ceci vient du fait que les élèves n'ont plus à éprouver des difficultés dans la construction d'un algorithme avec la mise en place de variables informatiques, de leurs initialisations, du choix des structures informatiques et de leurs constructions. Cela permet à l'élève de ne plus générer de difficultés autres que celles inhérentes aux compétences utilisées dans le nouveau domaine de travail que sont les statistiques et les probabilités. Ainsi, nous pouvons penser qu'un travail mené dans un *ET*A, dans le cadre d'un *ETM* spécifique, permet d'aider les élèves à aborder de nouvelles

articulations possibles entre *ET*A et nouveau *ETM* spécifique pour proposer un travail sur des tâches mathématiques d'un niveau supérieur quel que soit le domaine étudié.

L'expérimentation sur la *politique des naissances* propose à l'élève de construire un type de modèles mathématiques défini sur la construction d'algorithmes permettant ainsi de générer une résolution algorithmique de l'approche « fréquentiste » du processus aléatoire et par conséquent de répondre à toutes les instances du problème posé.

Comme pour les travaux de Briant, la résolution algorithmique d'un problème mathématique dans un domaine spécifique (par exemple les statistiques et les probabilités) et son écriture dans un environnement numérique nécessitent de la part de l'élève de mener une *transposition*, dont la complexité peut venir non seulement de la *non-congruence* entre les environnements papier-crayon et numérique, comme les analyses l'ont montrée avec l'arbre de probabilité et de ses effets sur la construction de l'algorithme, *mais également d'une pensée algorithmique qui n'est pas contenue entièrement dans la pensée mathématique* (Briant, 2013).

6. Pensée mathématique / Pensée algorithmique

Pour rebondir sur le dernier point de la section précédente, revenons d'abord sur les travaux de Briant. En effet, elle rappelle que l'idée de *pensée algorithmique* est développée par Modeste (2012) qui précise qu'une *activité mathématique est centrée sur la résolution de problèmes* et que lorsque la pensée algorithmique est considérée *en tant que pensée mathématique parmi d'autres,* elle est alors *une approche particulière de certains problèmes mathématiques* (p. 47). Cependant, Briant observe que Modeste pense qu'il est nécessaire de considérer une pensée algorithmique qui ne serait pas totalement incluse dans la pensée mathématique. Pour cela, il s'appuie sur des travaux de Knuth (1985). En effet, Knuth signale que l'algorithmique fait appel à des raisonnements communs aux mathématiques mais aussi à un mode de pensée spécifique.

Ainsi, Knuth s'interroge sur les différences qui peuvent exister entre la pensée mathématique et la pensée algorithmique (Pour Knuth, l'algorithmique est étendu au sens large, comme la science informatique). Il repère dans l'activité mathématique neuf grands modes de pensée et remarque que six d'entre elles sont communes à la pensée algorithmique : – la manipulation des formules ; – la représentation d'une réalité ; – la

réduction à des problèmes plus simples ; – le raisonnement abstrait ; – les structures d'informations ; – les algorithmes. Cependant, il ajoute deux catégories spécifiques à la pensée algorithmique qui lui semblent ne pas être nécessairement présentes dans la pensée mathématique : – la notion de complexité ; – la notion d'affectation symbolisé par un symbole particulier comme := ou ←. Ainsi, il fait référence à une notion d'affectation dynamique : l'affectation successive de différentes valeurs à une même variable au cours des différentes étapes du processus (cf. Chapitre 4 sur l'ingénierie « dichotomie discrète »).

La notion de complexité, qui semble propre à l'algorithmique, apparait dans le cadre de notre première ingénierie où les élèves travaillent sur deux types de preuves : une « heuristique » et une « explicative ». Cette seconde peut être interprétée par des élèves débutants en informatique comme synonyme d'une approche de la « complexité » de l'algorithme au sens large. En effet, la notion de complexité ne peut être abordée explicitement dans des classes du lycée.

La notion d'affectation n'est pas sans rappeler la distinction entre *variable mathématique* et *variable informatique*. Ce dernier point trouve son importance dans les trois ingénieries proposées comme nous avons pu le rapporter dans la partie 2 pour chacune d'elles. En effet, les élèves du cycle Scientifique Terminal, mais aussi les élèves de Seconde ont une assez bonne représentation mentale d'une variable mathématique qu'ils interprètent comme étant un symbole représentant un élément non spécifié ou inconnu d'un ensemble jouant un rôle de marque place (en logique par exemple). En revanche, nous observons, en particulier au cours de l'ingénierie sur la dichotomie qui débute le travail accompli pendant les ingénieries en analyse et en statistiques/probabilités, que les élèves peuvent avoir une représentation erronée du concept de variable informatique. En effet, nous constatons dans les premières phases sur la dichotomie (plus précisément dans le cas « discret ») que certains élèves ne considèrent pas une variable informatique comme pouvant désigner un emplacement dans la mémoire et que son contenu pourrait évoluer. Ainsi, nous observons que ces élèves génèrent une nouvelle série de variables informatiques chaque fois que celles-ci doivent avoir des contenus qui évoluent au fur et à mesure que les étapes de l'algorithme avancent. Ils ne semblent pas aussi avoir conscience au début des expérimentations que l'opération d'affectation (relation antisymétrique) diffère de la relation d'égalité (évidemment symétrique). Ce retour sur les opérations d'affectations est d'autant plus visible si la syntaxe

de l'environnement numérique choisi propose des symbolismes pouvant porter à confusion chez des élèves débutants en informatique. En effet, dans le cas de l'utilisation de l'environnement LARP, pour des algorithmes représentés sous forme d'organigrammes, l'affectation est représentée par le symbole =.

L'algorithmique fait donc appel à une typologie de raisonnements communs aux mathématiques mais aussi à un mode de pensée spécifique. Pour Modeste, *la différence essentielle réside dans le fait que la pensée algorithmique, vue comme pensée mathématique, ne questionne pas l'efficacité des algorithmes qu'elle produit, et qu'elle n'utilise pas la notion de variable informatique, ce qui renvoie à la notion de complexité d'un algorithme et à l'opération d'affectation* (Briant, 2013). Ainsi, pour comprendre la signification de la pensée algorithmique, il est *indispensable d'appréhender simultanément les deux points de vue, intra-mathématique mais aussi extra-mathématique* (Modeste, 2012, p.53).

Briant (2013) relève que prendre en considération une dimension qui n'existerait pas dans un *environnement mathématique usuel en papier-crayon,* […] *induit un mode de pensée différent.* Ainsi, l'*existence de cette pensée spécifique* […] *semble être un élément didactique important qui permet de mieux comprendre la transposition qui se produit lors de la recherche d'algorithmes pour résoudre un problème.* Reprenant cette hypothèse qu'il existe un mode de pensée spécifique à l'algorithmique, nous avons proposé au cours de notre recherche de conceptualiser cette pensée algorithmique. Ainsi, au cours de nos expérimentations sur la preuve d'une conjecture dans le domaine de la *théorie élémentaire des nombres* (Partie 2 - Chapitre 2), d'un théorème dans le domaine de l'analyse (Partie 2 – Chapitre 5), mais aussi sur la conception d'un modèle basé sur la construction d'un algorithme dans le domaine des probabilités/statistiques (Partie 1 – Chapitre 6), nous avons pu observer chez l'élève débutant en informatique, une progression sur la compréhension des objets mathématiques au fur et à mesure des acquisitions des savoirs et savoir-faire dans l'ETA associé à l'ETM spécifique. Ainsi, cette observation permet de voir le développement d'une dialectique « pensée mathématique/pensée algorithmique » chez les élèves des lycées français. En effet, suite aux analyses *a posteriori* de nos ingénieries faites à travers l'étude des interactions et des articulations entre *ETA* et *ETM* idoines et personnels, il nous apparaît qu'articuler des raisonnements de type algorithmique relevant de l'extra-mathématique avec des raisonnements mathématiques « classiques » relevant de l'intra-mathématique lors de

l'apprentissage de nouvelles compétences mathématiques comme la preuve, la modélisation,… quel que soit le domaine mathématique spécifique étudié, peut aider l'élève à considérer une nouvelle dimension que le seul environnement mathématique « classique » ne lui aurait pas permis d'aborder. De plus, le fait d'avoir choisi des classes qui ont pu travailler plusieurs de ces ingénieries, mêmes si celles-ci étaient associées à divers domaines mathématiques, nous a permis d'observer l'évolution de la pensée algorithmique chez l'élève et des conséquences sur le développement de sa pensée mathématique. Ainsi, nous observons que cette évolution permet de faciliter chez l'élève l'acquisition et la compréhension d'objets mathématiques (au sens de Douady, 1986) relavant initialement de l'intra-mathématiques et de donner du sens à certains concepts mathématiques comme la preuve ou la modélisation.

7. Les limites observées au cours de notre recherche – Quelles perspectives pouvons-nous envisager à ce travail de recherche ?

7.1 Les limites observées au cours de notre recherche

7.1.1 Des ingénieries « locales » - Une ingénierie « globale »

Le fait d'avoir mené deux ingénieries « locales » (*Algorithme de Kaprekar* et *politique des naissances*) ne nous a pas permis d'avoir autant de recul qu'au cours de l'ingénierie « globale » faite dans les trois niveaux scolaires du lycée et sur plusieurs séances expérimentées en classe. Ainsi, cette ingénierie a pu s'inscrire de façon « naturelle » dans la progression de l'enseignant quel que soit le niveau scolaire. En particulier au niveau de la Terminale, cela a permis à l'enseignant de compléter son enseignement sur les fonctions et les théorèmes des valeurs intermédiaires et de la bijection. De plus, toujours à ce niveau scolaire, l'enseignant a pu profiter des séances allouées à cette ingénierie sur la dichotomie pour faire travailler les élèves en salle informatique sur des environnements numériques divers. En revanche, aux niveaux des classes de Seconde et de Première, bien que les enseignants aient reconnu le fait que les séances proposées peuvent s'inscrire dans la progression prévue sur l'année scolaire, ces derniers ont rapporté une certaine lassitude des élèves à mesure que les séances avançaient.

Concernant les ingénieries « locales », nous avons été davantage confrontés au problème de l'importance temporelle que pouvaient prendre les différentes séances dans une progression réglée en fonction des contraintes de l'établissement : - progression commune entre les différentes classes pour chaque niveau scolaire permettant des évaluations sommatives régulières communes ; - organisation des disponibilités des salles informatiques sur l'ensemble de l'établissement.

Dans le cas de l'ingénierie « algorithme de Kaprekar », nous avons eu comme problème que cette ingénierie était découpée en deux sous-ingénieries : l'« ingénierie-Guy » avec la planification d'un algorithme complexe (présentée lors d'un mémoire de Master par M.N. Guy en 2013) et la nôtre avec l'élaboration d'une preuve de la conjecture observée lors l'« ingénierie-Guy ». En effet, dans les deux cas les adultes présents dans la salle étaient l'étudiante Guy et l'enseignant de la classe qui assurait pour la première sous-ingénierie son rôle d'enseignant et pour la seconde sous-ingénierie la charge du chercheur, étant donné que le chercheur et l'enseignant étaient la même personne. Quant à l'étudiante lors de la seconde sous-ingénierie sur la preuve, elle assura la tâche de l'enseignante. En effet, cette étudiante en tant qu'agrégée de mathématiques accepta de gérer la classe lors de la sous-ingénierie sur la preuve. Ceci fut perturbateur pour un certains nombres d'élèves de la classe, en particulier, au cours d'une des séances qui se déroula lors du carnaval du lycée.

De plus, le fait que la dernière ingénierie sur la *politique des naissances* s'est faite sur deux classes ayant déjà participé à l'ingénierie « globale » sur la dichotomie a probablement biaisé les obstacles qu'auraient pu rencontrer des élèves débutants en informatique lors de travaux sur les algorithmes et de leurs implémentations dans des environnements numériques qui furent en l'occurrence identiques pour les deux ingénieries. En revanche, cela a permis de mieux observer les retombées des avancées dans l'*ETA* sur l'*ETM*. En effet, le fait qu'au cours de l'ingénierie sur la dichotomie les élèves aient acquis des savoirs et savoir-faire dans un *ETA*, leur a probablement permis de donner plus aisément du sens aux enjeux des cycles de modélisation nécessaire à la conception d'un modèle de type algorithmique permettant de simuler le processus aléatoire et de progresser plus rapidement dans la compréhension des approches « fréquentiste » et « probabilistes » de la problématique. Au niveau des ETA, la « nouveauté » dans la dernière ingénierie fut l'introduction du tableur dans le cadre d'une élaboration d'un modèle de la simulation du processus aléatoire. Mais les élèves de ces

niveaux scolaires et le fait que ce soient des classes ayant un bon niveau en mathématiques n'a pas permis de voir des difficultés majeures chez les élèves dans cet environnement qu'ils côtoient depuis le collège tant en mathématiques qu'en enseignement de technologie au cours des deux dernières années de collège.

7.1.2 Le choix des environnements numériques

Concernant le choix fait dès le début des ingénieries sur les environnements numériques autorisés, cela a pu s'avérer au final être réducteur auprès des apprentissages des élèves en particulier pour ceux souhaitent poursuivre des études scientifiques après le baccalauréat. En effet, ces deux environnements ont comme inconvénient qu'ils ne sont plus utilisés dans l'enseignement supérieur.

Cependant, ce choix a été validé par l'ensemble des enseignants participants aux expérimentations. En effet, une majorité d'entre eux ont indiqué qu'ils avaient l'habitude d'utiliser le logiciel AlgoBox dans leurs classes les années antérieures. En revanche, l'ensemble de ces enseignants a dû découvrir les techniques de construction des organigrammes et par conséquent l'environnement LARP. Mais, le tutoriel proposé par les auteurs de ce logiciel et le polycopié (voir annexe) distribué par le chercheur les ont bien aidés à s'approprier cet environnement et à (re)découvrir les représentations « spatiales » d'un algorithme.

En plus, les instructions données par les concepteurs des environnements utilisés sont écrites en français et non en anglais ce qui peut être un gain de temps pour les élèves.

Pour conclure, nous avons aussi constaté que les structures de boucles possibles n'étant pas forcément les mêmes d'un logiciel à l'autre, cela peut rendre les constructions des algorithmes plus difficiles pour certains élèves en particulier quand ils doivent passer d'un environnement à l'autre.

Une difficulté qui est apparue aussi avec AlgoBox : c'est le fait qu'il n'est pas possible de construire des sous-programmes. C'est pour cela que cet environnement a été aussi choisi pour la première ingénierie sur l'algorithme de Kaprekar. Car elle a permis aux élèves d'organiser une planification de l'algorithme surmontant cette impossibilité.

7.2 Quelles perspectives pouvons-nous envisager à ce travail de recherche ?

Pour conclure nos travaux, nous souhaitons donner quelques pistes de réflexion relatives à un prolongement possible du travail de recherche mené pendant ces années de thèse, en particulier pour aider à favoriser les conditions nécessaires à un enseignement de l'algorithmique qui trouverait sa place au lycée mais aussi au collège, voire à l'école élémentaire comme le propose les nouveaux programmes des cycles 2, 3 et 4 en application depuis la rentrée 2016.

7.2.1 Pour une suite à ce travail de recherche

Pour tester les effets et la robustesse des résultats de notre recherche, nous devrions proposer de reprendre une partie des activités mises en place en tenant compte que les élèves du lycée vont avoir dans les années à venir entre 6 et 7 ans de pratiques sur les algorithmes et les déplacements de robots en fonction du codage donné.

Il serait aussi pertinent de voir ces activités pratiquées dans des établissements autres que ceux de l'Ile de France. En particulier un projet de recherche autour de la modélisation et de l'algorithmique pourrait être construit avec d'autres systèmes d'enseignements qu'ils soient européens, américains, africains ou asiatiques. En effet, suivant les diversités socio-culturelles, linguistiques des populations observées et des institutions des pays, nous pourrions avoir une plus grande diversité dans les approches des élèves que ce soit sur le plan mathématique, modélisation, algorithmique. Ainsi, une utilisation de l'algorithmique comme objet mathématique pour bâtir une typologie de preuves d'une conjecture ou d'un théorème pourrait être envisagée sur plusieurs classes réparties dans plusieurs pays au moins au niveau de l'Union Européenne.

Au cours de ce travail de recherche, nous n'avons pas particulièrement abordé les pratiques enseignantes. En effet, le travail d'observation s'est essentiellement fait sur les pratiques des élèves en classe dans le cadre d'un enseignement mathématique avec l'utilisation de l'algorithmique comme objet d'apprentissage dans différents domaines mathématiques enseignés et le développement de savoirs spécifiques. Ainsi, lors d'une suite à ces travaux de recherche, une possibilité serait d'observer le travail de l'enseignant tant dans la préparation des séances que dans la gestion de la classe en particulier en salle informatique.

7.2.2 Favoriser la place de l'algorithmique dans l'enseignement des mathématiques

Lors des échanges avec les enseignants qui ont participé aux expérimentations mais aussi avec des collègues enseignants les mathématiques dans les établissements où ont eu lieu ces expérimentations, nous avons pu noter qu'une majorité d'enseignants ont eu dans un premier temps un ressenti de recul vis-à-vis de l'introduction de l'algorithmique dans l'enseignement des mathématiques. En effet, les premières années où l'enseignement de l'algorithmique au lycée s'est institutionnalisé, les enseignants ont considéré qu'ils n'étaient pas formés pour cela et que cet enseignement allait conduire à des réductions sur les compétences à acquérir en mathématiques « classiques ». En revanche, les enseignants qui avaient la responsabilité des enseignements de spécialité en mathématiques des classes du cycle Terminal en Sciences Economiques eurent moins de difficultés à accepter cette introduction. Nous pouvons supposer que cela venait du fait des enseignements pratiqués dans ces classes : calcul matriciel et introduction à la théorie des graphes. De plus, les enseignants interrogés ont indiqué qu'ils n'avaient pas nécessairement conscience des complémentarités qui pouvaient exister entre l'algorithmique et la programmation. Des outils comme le calcul formel qu'offrent certaines calculatrices n'ont pas favorisé l'acceptation chez un certain nombre d'enseignants d'un enseignement de l'algorithmique. En revanche, nombreux sont les enseignants qui font un parallèle entre l'introduction d'un enseignement de l'algorithmique et un enseignement du raisonnement logique aux trois niveaux scolaires du lycée.

Pour revenir sur les observations faites par Briant (2013) dans sa thèse, nous observons aussi que l'introduction de l'algorithmique dans la scolarité de l'élève permet *de faire abstraction de la surenchère de toutes les fonctions complexes que l'on trouve* […] *dans une calculatrice de calcul formel* afin de le guider à concevoir des algorithmes implémentables dans des environnements numériques et de les tester en mettant en place des situations où l'élève doit analyser le problème initial, décomposer chacune des actions en des *actions élémentaires* (au sens de Briant) et planifier la réunion de ces actions élémentaires (au sens de Guy, 2013) afin de construire des algorithmes complexes. L'algorithmique offre ainsi un nouveau registre sémiotique au sens de Duval pour la représentation de modèle mathématique dans les différents domaines mathématiques enseignés au lycée.

De fait, nous pensons que l'introduction d'*ETA* dans des tâches issues de différents domaines mathématiques se doit d'intégrer cette dimension sémiotique, mais aussi les deux autres dimensions : discursives et instrumentales comme nous avons pu l'observer lors des analyses des différentes ingénieries. En effet, cette prise de conscience sur ces trois dimensions dans le cadre d'*ETA* et des articulations possibles avec des *ETM* afin d'aider les élèves à s'approprier des savoirs spécifiques permettraient de mieux définir la pensée algorithmique en relation avec le travail algorithmique nécessaire afin de permettre à l'élève de mieux accéder aux savoirs mathématiques enseignés.

8. Conclusion

Au terme de cette recherche menée sur plusieurs années dans différentes classes des années du lycée, nous avons conscience de n'avoir pu que donner que quelques éléments de réponses sur les contributions de l'algorithmique *aux apprentissages dans différents domaines mathématiques enseignés et des possibilités qu'offrent l'algorithmique au développement de savoirs spécifiques.* En particulier, il nous semble que beaucoup d'études pourraient être mené sur l'algorithmique dans les domaines des statistiques et des probabilités autour de marches aléatoires et de déplacements d'un robot sur des surfaces planes définies par des quadrillages. Nous pensons aussi qu'une approche du codage comme le préconise l'introduction de l'algorithmique dans les nouveaux programmes de 2016 pour les cycles 2 et 3 de l'école élémentaire et du début du collège peut donner des pistes sur l'enseignement de l'algorithmique du CE2 à la Terminale, voire aux deux premières années du supérieur, en particulier dans les sections à visée professionnelles courtes comme les classes de STS ou d'IUT qu'elles soient tertiaires ou industriels.

Bibliographie

ARSAC, J. (1980). Premières leçons de programmation. CEDIC-Nathan, Paris.

ARTIGUE M. (1992). Ingénierie didactique. *Recherches en Didactique des Mathématique*s, vol. 9-3, pp. 281-308.

BALACHEFF, N. (1994). Didactique et intelligence artificielle. *Recherches en didactique des mathématiques, 14*(1), 9-42.

BESSOT A., & NGUYEN C.T. (2003). La prise en compte des notions de boucle et de variable. *Petit x*. n°62, pp. 7-32.

BLUM, W. et LEISS, D. (2005). « Filling Up » - The problem of independence-preserving teacher interventions in lessons with demanding modeling tasks. *In Proceedings for the CERME4, WG 13 Modeling and Applications* (pp. 1623-1633).

BORDIER, J. (1991). *Un modèle didactique, utilisant la simulation sur ordinateur, pour l'enseignement de la probabilité* – Thèse. Paris, Université Paris VII.

BRIANT, N. (2013). *Étude didactique de la reprise de l'algèbre par l'introduction de l'algorithmique au niveau de la classe de seconde du lycée français*. Thèse Université Montpellier II - Sciences et Techniques du Languedoc.

BRIANT, N. & BRONNER A. (2015). Étude d'une transposition didactique de l'algorithmique au lycée : une pensée algorithmique comme un versant de la pensée mathématique, *Actes du Colloque EMF2015 – GT3*, pp. 231-246.

BROUSSEAU, G. (1998). *Théorie des situations didactiques.* Grenoble : La Pensée Sauvage.

CHEVALLARD, Y. (1982). Pourquoi la transposition didactique ? Dans *Actes du Séminaire de didactique et de pédagogie des mathématiques* de l'IMAG, Université scientifique et médicale de Grenoble. (p. 167-194).

CHEVALLARD, Y. (1985). *La transposition didactique*. Grenoble : La pensée sauvage.

COUTAT, S., LABORDE, C. & RICHARD, P. R. (2013). L'apprentissage instrumenté de propriétés en géométrie : propédeutique à l'acquisition d'une compétence de démonstration. *Educational Studies in Mathematics*.

COUTAT, S. & RICHARD, P. R. (2011). Les figures dynamiques dans un espace de travail mathématique pour l'apprentissage des propriétés géométriques. *Annales de didactique et de sciences cognitives. 16,* pp. 97-126.

DJISKRA, E.D. (1979). *A discipline of programming*. Prentice-Hall, Englewood Cliff.

DOUADY, R. (1986). Jeux de cadres et dialectique outil-objet. Recherches en didactique des Mathématiques, Vol 7.2, pp. 7-31. Editions La Pensée Sauvage.

DUBINSKI, E. (1999). One theoretical perspective in undergraduate mathematics education research, In O. Zaslavsky (Ed.), *Proceedings of the 23rd Conference of PME*, Haifa, Israel, 4, pp. 65-73

DUVAL, R. (2005). Les conditions cognitives de l'apprentissage de la géométrie. *Annales de Didactique et de sciences cognitives, 10,* pp. 5-54.

DUVAL, R. (2006). Quelle sémiotique pour l'analyse de l'activité et des productions mathématiques ? *Relime, Numero Especial*, pp. 45-81.

ENGEL, A. (1979). *Mathématique élémentaire d'un point de vue algorithmiqu*e, (adapté par Daniel Reisz), Paris : CEDIC.

GAYDIER, F. (2011). *Simulation informatique d'expérience aléatoire et acquisition de notions de probabilité au lycée*. Thèse. Paris, Université Paris Descartes.

GUY, M.-N. (2013). *Utilisation du cadre théorique de la planification pour la conception d'algorithmes complexes par des élèves de lycée*. Mémoire de master de Didactique des mathématiques. Université Paris Diderot.

HOC, J.-M. (1987a). *Psychologie cognitive de la planification.* Grenoble, Presses Universitaires de Grenoble.

HOC, J.M. (1987b). L'apprentissage de l'utilisation des dispositifs informatiques par analogie à des situations familières. Psychologie Française n° 4.

HOUDEMENT C., KUZNIAK A. (2006) Paradigmes géométriques et enseignement de la géométrie. *Annales de Didactique et de Sciences Cognitives*, **11**, pp. 175-195.

KAHANE, J.-P. (2002). *Enseignement des sciences mathématiques : Commission de réflexion sur l'enseignement des mathématiques : Rapport au ministre de l'éducation nationale* (O. Jacob, Ed.). Paris : CNDP. Disponible sur https://www.pedagogie.ac-aix-marseille.fr/jcms/c_75043/fr/commission-kahane.

KIET, B.A. (2015) Apports de la simulation et de l'utilisation de logiciels pour l'enseignement /apprentissage des probabilités et des statistiques en première année d'Université au Viêtnam dans un cursus non mathématique. *Thèse de doctorat*. Université Paris-Diderot.

KNUTH, D.E. (1985). Algorithmic thinking and mathematical thinking. *The American Mathematical Monthly*, *92*(1), 170-181. (Réédition avec corrections (1996) Algorithms in Modern Mathematics and Computer Science. In *Selected Papers on Computer Science*, *87-114.*

KNUTH, D.E. (2000) Selected papers on analysis of algorithms. Center for the Study of Language and Information, Stanford

KNUTH, D.E. (2010) Selected papers on design of algorithms. Center for the Study of language and Information, Stanford.

KUZNIAK, A. (2006). Paradigmes et espaces de travail géométriques. Éléments d'un cadre théorique pour l'enseignement et la formation des enseignants en géométrie. *Canadian Journal of Science and Mathematics Education, vol 6.2. pp 167-188.*

KUZNIAK, A. (2011). L'espace de Travail Mathématique et ses genèses. *Annales de didactique et de sciences cognitives, 16,* 9-24. El espacio de trabajo matemático y sus génesis, traducción J. Lezama, Cicata.

KUZNIAK, A., & NECHACHE, A. (2014). Penser une progression en géométrie en formation des enseignants. *In Acte du* 41*e Colloque COPIRELEM*, Mont de Marsan.

KUZNIAK, A., NECHACHE, A. & DROUHARD, J. P. (2016). Understanding the development of mathematical work in the context of the classroom. *ZDM*, pp. 1-14.

KUZNIAK, A., & RICHARD, R. P. (2014). Spaces for mathematical work: viewpoints and perspectives. *Relime*, 17(4.1), pp. 17-26.

LAGRANGE, J.B. (1990). Des situations connues aux traitements sur des données codifiées. *Actes du Second Colloque Francophone de Didactique de l'Informatique* Presses Universitaires de Namur

LAGRANGE, J-B. (1990). *Des situations connues aux traitements sur des données codifiées : représentations mentales et processus d'acquisition dans les premiers apprentissages en informatique*. Thèse de Doctorat. Université Paris 7.

LAGRANGE J.-B. (1992), Représentations mentales des données informatiques et difficultés d'acquisition chez des débutants en programmation, *EPI n° 67-68*.

LAGRANGE, J.B. (1995). Bridging a GAP from Computer Science to Algebra. In *Technology in Mathematics Teaching* L. Burton & B. Jaworski (eds.) Chartwell-Bratt, Bromley.

LAGRANGE, J.B. (2014). Algotithmics. In S. Lerman (ed.), *Encyclopedia of Mathematics Education*, DOI 10.1007/978-94-007-4978-8, Springer Science+Businees Media Dordrecht 2014, pp. 32-35.

LAGRANGE, J.-B., & GUY, M.-N. (2015). Planification et connaissances mathématiques dans une situation d'apprentissage au lycée : l'algorithme de Kaprekar. *Petit x*, 97, pp. 45-70.

LAGRANGE, J.-B. & KIET, B.A. (2016). Une approche fréquentiste des probabilités et statistiques en première année d'Université au Vietnam dans un cursus non mathématique. In *First conference of International Network for Didactic Research in University Mathematics*. Montpellier. Disponible sur https://hal.archives-ouvertes.fr/hal-01337930/document

LAGRANGE, J.B., & ROGALSKI, J. (2017) Savoirs, concepts et situations dans les premiers apprentissages en programmation et en algorithmique, in Annales de Didactique et de Sciences Cognitives, 22, pp. 119-158.

LAVAL, D. (2015). L'algorithmique comme objet d'apprentissage de la démarche de preuve en théorie élémentaire des nombres : l'algorithme de Kaprekar. In Gómez-Chacón, Escribano, Kuzniak & Richard (Eds.), *Mathematical Working Space, Proceedings Fourth ETM Symposium,* pp. 103-116. Madrid : Publicaciones del Instituto de Matemática Interdisciplinar, Universidad Complutense de Madrid.

LAVAL, D. (2017). L'algorithme de dichotomie « discret » : une stratégie « rapide » et « gagnant ». . In Gómez-Chacón, Kuzniak, Nikolantonakis, Richard & Vivier (Eds.), *Mathematical Working Space, Proceedings Fifth ETM Symposium*, pp. 267-279. University off Western Macedonia, Florina, Greece.

MODESTE, S. (2012) Enseigner l'algorithme pour quoi ? Quelles nouvelles questions pour les mathématiques ? Quels apports pour l'apprentissage de la preuve ? Thèse de didactique des mathématiques. Université de Grenoble.

MONTOYA-DELGADILLO, E., & VIVIER, L. (2014). Les changements de domaine dans le cadre des espaces de travail mathématique. Vol 19, pp. 73-101.

NEUKIRCH, J. (2006). *Algebraische Zahlentheorie.* Springer-Verlag Berlin Heildeberg 1992

NGUYEN, C.T. (2005) *Etude didactique de l'introduction d'éléments d'algorithmique et de programmation dans l'enseignement mathématique secondaire à l'aide de la calculatrice*. Thèse en cotutelle France – Viêtnam de didactique des mathématiques. Université Joseph Fourier (Grenoble) et Ecole Normale Supérieur n° 1 de Hanoi.

NGUYEN, C.T., & BESSOT, A. (2010) Introduire des éléments d'algorithmique et de programmation dans l'enseignement secondaire ? Une étude didactique *Petit x*. n° 83 (pp. 27-49)

NIJIMBERE, C. (2015) *L'enseignement de savoirs informatiques pour débutants, du second cycle de la scolarité secondaire scientifique à l'université en France. Une étude comparative*. Thèse Université Paris Descartes. Paris.

OUVRIER-BUFFET, C., MODESTE, S., & GRAVIER, S. (2010). Algorithmique et apprentissage de la preuve. *Repères IREM, 70*, pp. 51-71. Tropiques.

PAPERT, S., & HAREL, I. (1991). *Constructionism.* Ablex Publishing Corporation, Norwood.

PARZYSZ, B. (1993). Des statistiques aux probabilités : exploitons les arbres. *RepèresIREM*, 10, pp. 91–104.

PARZYSZ, B. (2007). Expérience aléatoire et simulation : le jeu de croix ou pile. *Repères-IREM*, 66, pp. 27–44.

PARZYSZ, B. (2009). Des expériences au modèle, via la simulation (From experiences to a model, via simulation). *Repères-IREM, 74*, pp. 91-103.

PARZYSZ, B. (2011). Quelques questions didactiques de la statistique et des probabilités. *In Annales de didactique et de sciences cognitives*, volume 16, pages 127–147.

PETRE, M., & BLACKWELL, A. F. (1997). A glimpse of expert programmers' mental imagery. In *Papers presented at the seventh workshop on Empirical studies of programmers* (pp. 109-123). ACM. Retrieved from https://www.cs.duke.edu/courses/fall00/cps189s/readings/petre-expert.pdf

RAUSCHER, J.-C (2007). Eléments de formation initiale des professeurs d'école à partir de l'algorithme de Kaprekar et de questions de pavage du plan. *XXXIV^e Colloque COPIRELEM, Expérimentation et modélisation dans l'enseignement scientifique : quelles mathématiques à l'école?* - Troyes 2007, p. 158-171. Disponible sur http://www.arpeme.fr/documents/589C3D622D7F9FD0616D.pdf

ROGALSKI J. (1988) Les représentations mentales du dispositif informatique dans l'alphabétisation. *Actes du premier colloque franco-allemand de didactique*, pp. 235-245.

ROGALSKI, J., & SAMURCAY, R. (1986), Les Problèmes Cognitifs Rencontrés par des Elèves de l'Enseignement Secondaire dans l'Apprentissage de l'Informatique, *European Journal of Psychology of Education*, Vol. I, n° 2, pp 97-110.

ROGALSKI, J., & SAMURCAY, R. (1990). Acquisition of programming knowledge and skills. In Hoc J.-M., Green T. G. R., Samurçay R., & Gilmore D. (Eds.), *Psychology of Programming*. Academic Press, Londres, pp. 157-174.

ROGALSKI, J., & VERGNAUD, G. (1987). *Didactique de l'informatique et acquisitions cognitives en programmation*, Psychologie Française, 4, pp. 267-273.

ROLAND H. (2010). *L'algorithme de dichotomie*. IREM. Disponible sur http://www.irem.univ-mrs.fr/IMG/pdf/dichotomie.pdf

SFARD, A. (1991). On the dual nature of mathematical conceptions: Reflections on processes and objects as different sides of the same coin. *Educational Studies in Mathematics, 22,* pp. 1-36.

SAMURCAY, R. (1985). *Signification et fonctionnement du concept de variable informatique chez des élèves débutants*. Educationnal Studies in Mathematic, n° 16-2: pp. 143-161.

SAMURCAY, R., & ROUCHIER, A. (1990). Apprentissage de l'écriture et de l'interprétation des procédures récursives. Recherches en didactique des Mathématiques, Vol 10 2.3, pp. 287-327. Editions la Pensée Sauvage.

TURING, A.M. (1937). *On Computable Numbers, with an Application to the Entscheidungsproblem*, Proc. London Math. Soc., 2e série, 42, pp. 230-265.

VERGNAUD, G. (1983). Didactique et Acquisition du Concept de Volume. N° spécial de Recherches en Didactique des Mathématiques, 4.

VERGNAUD, G. (1987) Réflexions sur les finalités de l'Enseignement des Mathématiques, *Gazette des mathématiciens,* 32, pp. 54-61.

VERGNAUD, G. (1990). La théorie des champs conceptuels. *Recherches en Didactique des Mathématiques*, *10*(2-3), pp. 133-170.

WILENSKY, U., & RESNICK, M. (1999). Thinking in levels: a dynamic systems approach to making sense of the world. *Journal of Science Education and Technology.* 8(1) pp. 3-19.

Programmes scolaires

MEN (2009). Ministère de l'Éducation nationale. Programme de mathématiques, enseignement commun, seconde générale et technologique, arrêté du 23 juin 2009, BO numéro 30, du 23 juillet 2009. Disponible sur http://cache.media.education.gouv.fr/file/30/52/3/programme_mathematiques_seconde_65523.pdf.

MEN (2010). Ministère de l'Éducation nationale. Programme d'enseignement spécifique de mathématiques en classe de première de la série scientifique, BO spécial numéro 9, du 30 septembre 2010. Disponible sur http://cache.media.education.gouv.fr/file/special_9/21/1/mathsS_155211.pdf.

MEN (2011). Ministère de l'Éducation nationale. Programme d'enseignement spécifique de mathématiques en classe de terminale de la série scientifique, BO spécial numéro 8, du 30 septembre 2011. Disponible sur http://cache.media.education.gouv.fr/file/special_8_men/98/4/mathematiques_S_195984.pdf.

Documents ressources

MEN (2009a). Ministère de l'Éducation nationale. Ressources pour la classe de Seconde : Algorithmique. EDUSCOL, DGESCO. Disponible sur http://mathematiques.ac-bordeaux.fr/txtoff/prog/lycee/2de_09/doc_ress_algo_v25.pdf.

MEN (2009b). Ministère de l'Éducation nationale. Ressources pour la classe de Seconde : Fonctions. EDUSCOL, DGESCO. Disponible sur http://cache.media.eduscol.education.fr/file/Programmes/18/1/Doc_ressource_fonctions_109181.pdf

MEN (2012). Ministère de l'Éducation nationale. Ressources pour la classe de Première générale et technologique : Statistiques et Probabilités. EDUSCOL. Disponible sur http://cache.media.eduscol.education.fr/file/Mathematiques/59/6/Ressource_Statistiques_Probabilites_1eres_208596.pdf.

Manuels scolaires

Déclic (2012). Terminale S. Enseignement spécifique. Hachette.

Hyperbole (2010). Seconde. Nathan.

Hyperbole (2011). Terminale S. Enseignement de spécialité. Nathan.

Maths-repères (2010). Seconde. Hachette.

Math'*x* (2012a). Terminale S. Enseignement de spécialité. Didier.

Math'*x* (2012b). Terminale S. Enseignement spécifique. Didier.

Odyssée (2011). Première S. Hatier.

Radial (2006). Terminale S. Enseignement obligatoire. Belin.

Symbole (2011). Première S. Belin.

Annexes

- **Annexe 1** – Ecrire un algorithme au format spatial : les organigrammes (Document distribué aux élèves des classes ayant participé au moins à une des ingénieries)
- **Annexe 2** – Article sur le Rubik's Cube (Extrait de la revue « Pour la Science n° 400 – Février 2011 »)
- **Annexe 3** – Article « Eléments pour la formation initiale des professeurs d'école à partir de l'algorithme de Kaprekar » de J.-C. Rauscher (Extrait des actes du XXXIVème colloque : *Expérimentation et modélisation dans l'enseignement scientifique : quelles mathématiques à l'école ?* - Troyes 2007)

Annexe 1 – Ecrire un algorithme au format spatial : les organigrammes (Document distribué aux élèves des classes ayant participé au moins à une des ingénieries)

Quand on ne travaille pas sur tableur, mais on pratique la programmation sur calculatrice ou un environnement numérique comme *AlgoBox*, les structures algorithmiques de base utiles pour les compétences mises en jeu lors des exercices ou des cours dans les classes de Seconde, Première et Terminale sont en général données en langage « naturel » (c-à-d quasiment en français courant) ou en langage pseudo-code du type celui utilisé dans l'environnement *AlgoBox*.

Nous allons voir maintenant une façon de les représenter graphiquement, ce qui peut être plus parlant et aider l'élève à mieux comprendre le fonctionnement d'un algorithme décrivant un programme sur un simple « coup d'œil ».

Ainsi, une des façons de représenter un algorithme graphiquement (*format spatial*) s'appelle un organigramme. Les organigrammes sont composés de figures représentant un type d'action à effectuer et dans lequel est écrit l'action elle-même, ainsi que les relations entre ces différentes figures représentées par des lignes ou des flèches, et qui indiquent le chemin à suivre pour accéder d'une figure à la suivante. Un organigramme se « parcourt » en partant de la case marquée « début » et en suivant les lignes ou flèches jusqu'à la case marquée « fin ». Et cela tout en effectuant les actions qui sont décrites au cours des différents étapes décrites par l'algorithme écrit sous une « forme spatiale ».

Les figures usuelles utilisées dans la construction d'organigrammes

- Le **rectangle** (Fig. 208) qui représente une action, traitement de donnée ou de calcul. Cette action peut être simple ou complexe (c.-à-d. un groupe d'actions). C'est ce qui correspond à la séquence. On entre par une seule ligne ou flèche et on en sort par une seule ligne ou flèche de ce rectangle.

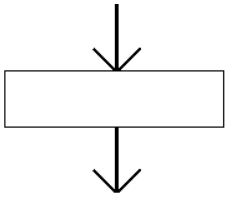


Figure 208 (Organigramme : représentation d'une séquence)

- Le **losange** (Fig. 209) représente un choix binaire. Dans le losange, on met une condition. On entre dans un losange par un seul chemin, mais on peut en sortir par deux chemins différents : celui noté VRAI et celui noté FAUX. On continue la « lecture » de l'algorithme en suivant le chemin qui correspond à la valeur de la condition, c.-à-d. : VRAI ou FAUX.

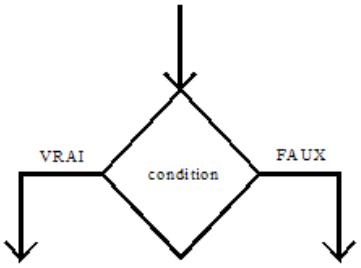


Figure 209 (Organigramme : représentation d'un choix binaire)

- Le **parallélogramme** (Fig. 210) représente une action d'échange entre la machine et l'utilisateur, qui peut être soit une introduction de données, appelée *Entrée*, soit une impression de résultats, appelée *Sortie*.

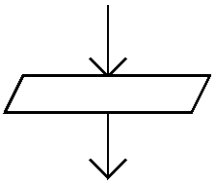


Figure 210 (Organigramme : représentation d'une action d'échange entre la machine et l'utilisateur (Entrée ou Sortie))

- La figure représentée ci-contre (Fig. 211) est utilisée pour **la sélection**. Il n'y a qu'un seul chemin entrant et chacun des chemins sortants correspond à une valeur particulière du sélecteur.

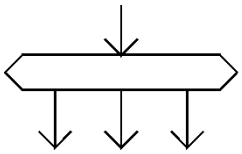


Figure 211 (Organigramme : Représentation pour la sélection)

- Les **ovales** sont les figures dans lesquelles se trouvent soit le mot DEBUT (Fig. 212), soit le mot FIN (Fig. 213). Ils représentent respectivement le début (ou point d'entrée) et la fin de l'algorithme

Figure 212 (Organigramme : DEBUT de l'algorithme)

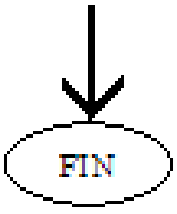


Figure 213 *(Organigramme : FIN de l'algorithme)*

Remarque : Dans un algorithme, il ne peut y avoir qu'un seul point d'entrée, par contre il peut y avoir plusieurs sorties suivant les chemins pris.

Types d'organigrammes utilisables pour représenter les structures étudiées dans les classes de lycée

- La séquence (Fig. 214)

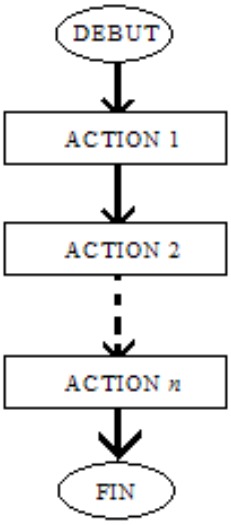


Figure 214 (Organigramme : la séquence)

- L'instruction conditionnelle (Fig. 215)

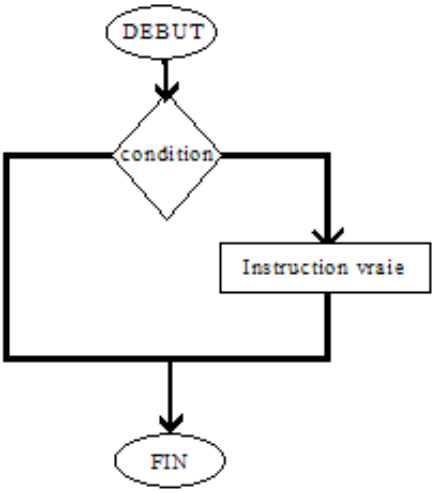


Figure 215 (Organigramme : l'instruction conditionnelle)

- L'alternative (Fig. 216)

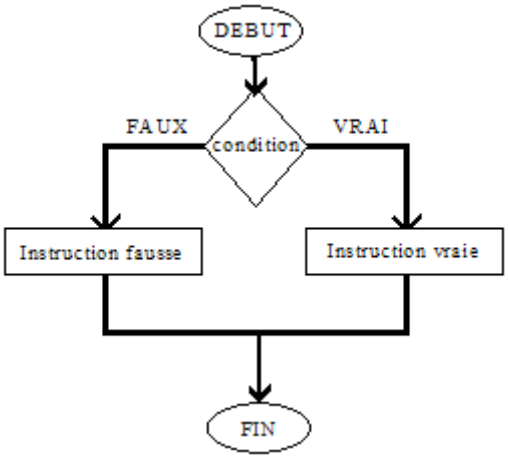


Figure 216 (Organigramme : l'alternative)

- Les structures de répétitions

  ➢ « TANTQUE…FAIRE »

  On procède ainsi : test puis action.

  L'algorithme de répétition est alors le suivant :

TANTQUE « la condition est vraie » FAIRE « exécuter l'action de répétition »

que l'on représente par le schéma[119] (fig. 217) ci-dessous :

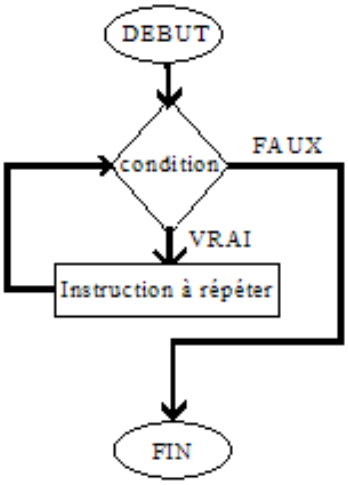


Figure 217 (Organigramme : boucle « TantQue … Faire »)

- « JUSQU'A CE QUE…FAIRE »

  Si l'on procède dans l'ordre inverse : action puis test, on obtient la structure de l'algorithme suivant :

« Exécuter l'action de répétition » JUSQU'A CE QUE « la condition soit vraie »

que l'on représente par le schéma (Fig. 218) ci-dessous :

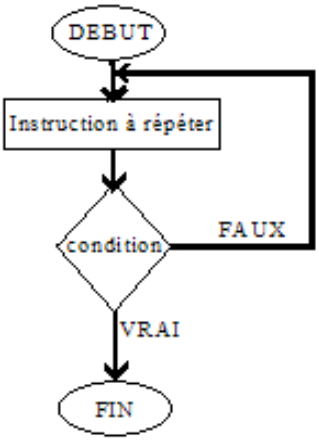


Figure 218 (Organigramme : Boucle « Jusqu'à ce que … Faire »)

- La répétition à compteur : « POUR… ALLANT DE … JUSQU'A »

  Dans le cas où l'on connaît d'avance – au plus tard au moment où l'on va commencer l'action répétitive – le nombre de fois que l'action doit être refaite, on peut contrôler la répétition par une variable simple jouant le rôle de compteur. Trois valeurs sont

[119] On parle d'*ordinogramme*

indispensables pour utiliser un compteur : Une valeur initiale (bien commencer), une valeur finale (bien finir) et une valeur qui fera varier le compteur (bien continuer).

La structure obtenue se présente de la manière suivante :

POUR « compteur » ALLANT DE « valeur initiale » JUSQU'A « valeur finale » REPETER « action de répétition »

Cette structure fonctionne de la façon suivante. Avant toute action, le compteur qui est une variable reçoit la valeur initiale ; après chaque exécution de l'action R, la valeur courante du compteur est augmentée de 1 ; la nouvelle valeur est comparée à la valeur finale ; tant que cette nouvelle valeur est inférieure (ou égale) à la valeur finale, il y a reprise de l'action de répétition.

Cela se représente sous la forme d'un des deux organigrammes vus dans les cas « TANTQUE » et « JUSQU'A ». En effet, la « répétition compteur » n'est par une nouvelle structure de répétition. Elle est en fait soit une structure de type TANT QUE, soit une structure de type JUSQU'A.

On peut utiliser comme type de schéma (avec un « pas » qui n'est pas nécessairement de 1), la figure ci-dessous (Fig. 219) :

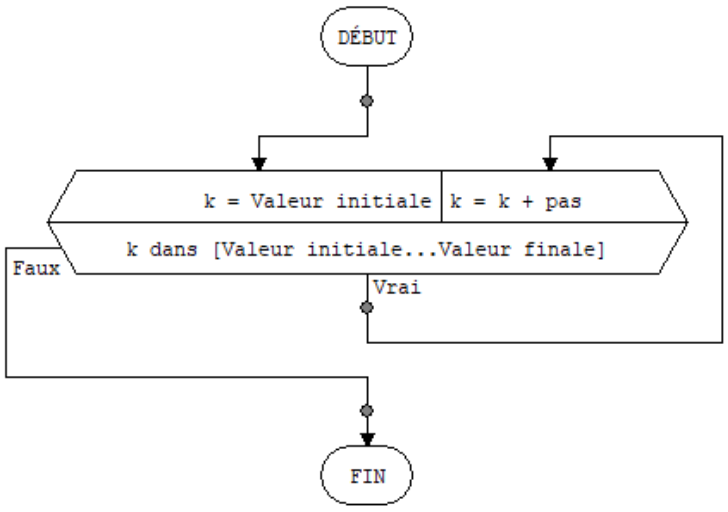


Figure 219 (Organigramme : boucle « « Pour… Allant de … Jusqu'à »)

Annexe 2 – Article sur le Rubik's Cube (Extrait de la revue « Pour la Science n° 400 – Février 2011 »)

Le *Rubik's Cube* : pas plus de 20 mouvements !

Le cube de Rubik est le numéro un de tous les casse-tête. Trente ans après son invention, on continue à y jouer et à résoudre les problèmes qu'il pose.

Jean-Paul DELAHAYE

Le cube de Rubik (ou *Rubik's Cube*) est le plus étonnant des casse-tête jamais inventés. D'abord, aucun jeu n'a suscité autant d'intérêt et conduit à la publication d'autant d'articles et de livres. Ensuite, il est difficile et intéresse pourtant des millions de gens, en donnant lieu à toutes sortes de compétitions et d'exploits... plus ou moins étranges et de plus en plus fous. De plus, il est à l'origine de centaines d'autres casse-tête mécaniques souvent aussi intéressants que lui, et qu'on peut aujourd'hui pratiquer virtuellement avec tout ordinateur. Enfin, le problème des configurations les plus difficiles a résisté pendant 30 ans et a exigé l'emploi d'un puissant réseau informatique avant qu'on en vienne à bout il y a quelques mois.

Nous allons reprendre ces quatre points et en particulier donner quelques informations sur la preuve récemment obtenue que 20 mouvements suffisent toujours pour reconstituer les faces colorées du cube.

Ingénieux mécanisme

En août 1980, dans le n° 34 de *Pour la Science*, Emmanuel Haiberstadt publiait un article intitulé *Le cube hongrois et la théorie des groupes* où il décrivait le cube de Rubik et indiquait un moyen pratique de le remettre en ordre, fondé sur l'analyse de la structure mathématique associée au cube.

Cet article contribua au succès du cube en France, succès qui, comme partout dans le monde, fut d'une ampleur considérable et d'une rapidité sidérante.

Le cube inventé six ans plus tôt par le sculpteur et professeur d'architecture Ernő Rubik est un bloc de 26 petits cubes rendus solidaires par un ingénieux mécanisme permettant à chaque face de 9 cubes (3 × 3) de pivoter autour d'un axe parallèle aux arêtes et passant par le centre de chaque face. L'emplacement du cube central, qui compléterait le tout et donnerait un empilement de 3 × 3 × 3 cubes, est occupé par le système de pivots qui donne sa cohésion à l'ensemble et le met en mouvement. La configuration initiale est telle que chaque face est colorée uniformément avec six couleurs : bleu, rouge, orange, vert, jaune et blanc. En opérant quelques rotations des faces, on mélange les couleurs et le problème posé est alors de reconstituer la configuration initiale.

Quelques instants de manipulation conduisent à comprendre les premiers éléments de la structure du casse-tête.

– Les cubes centraux des faces tournent sur eux-mêmes, mais restent toujours en place ; ils déterminent donc la couleur de chaque face et permettent d'orienter le cube d'une manière fixe.

– Un cube de coin (il y en a huit) sera toujours un cube de coin. Chacun d'eux a trois faces visibles de couleurs différentes qui, en prenant en compte les centres des faces, indiquent précisément la position qu'il doit occuper dans le cube remis en ordre.

– Les 12 cubes restants sont ceux des milieux des arêtes. Ils ont chacun deux faces visibles de couleurs différentes qui, comme pour les cubes de coin, déterminent précisément leur position dans la configuration à atteindre.

Un succès planétaire

D'apparence simple, la remise en ordre se révèle particulièrement coriace, et une fois mélangé, sans aide ou sans étude très approfondie du problème mathématique que pose le cube, vous ne réussirez jamais à le remettre en ordre. Le casse-tête est réellement difficile et son succès reste mystérieux.

On évalue que 350 millions de cubes de Rubik ont été fabriqués et vendus depuis que les premiers exemplaires ont été proposés au public hongrois en 1977. Cela fait de lui le casse-tête le plus vendu de tous les temps, à l'exception peut-être du Taquin (dénommé aussi 15-puzzle) dont il est impossible de comptabiliser les exemplaires (fabriqués selon des milliers de modèles différents) depuis son invention en 1879. Bien que le cube de Rubik ait été copié sans vergogne par l'industrie de la contrefaçon, Ernő Rubik, l'inventeur du cube, a fait fortune grâce à son casse-tête qui lui a valu la gloire et... lui a permis de consacrer le reste de sa vie à la mise au point d'autres casse-tête.

Une multitude de livres et d'articles ont été publiés au sujet du cube, de sa théorie et des méthodes pour le résoudre. Sur Internet, un grand nombre de pages lui sont consacrées. Le nombre de documents différents

parus chaque année permet de suivre l'évolution de la cubomania des années 1980. Une bibliographie réunie par George Helms recense 719 textes en 27 langues. Il y a 14 publications en 1979, 52 en 1980, 174 en 1981, 70 en 1982, 15 en 1983. L'ouvrage de James Nourse, *The Simple Solution to Rubik's Cube*, a été le livre le plus vendu au monde pour le mois de janvier 1982. Il est resté présent au palmarès des livres les plus vendus pendant trois ans et au total plus de six millions d'exemplaires de l'ouvrage ont été commercialisés. Plusieurs autres livres sur le cube de Rubik ont dépassé le million d'exemplaires.

Des compétitions de cubes de Rubik sont organisées partout dans le monde. La *World Cube Association* (WCA) patronne une multitude de concours, 200 en 2010.

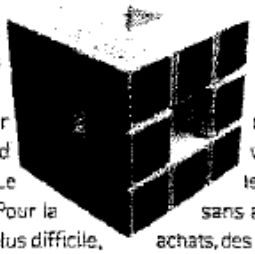
Les meilleurs cubistes résolvent un cube 3 × 3 × 3 en moins de 10 secondes. Précisons que les règles de la WCA permettent une inspection du cube pendant 15 secondes avant que la résolution ne commence. Le champion australien Feliks Zemdegs, qui a 15 ans, rétablit un cube en 8,5 secondes en moyenne. Il sait aussi résoudre le cube 4 × 4 × 4, beaucoup plus difficile, qui lui demande 42 secondes en moyenne, et le cube 5 × 5 × 5, extrêmement difficile, qui l'occupe 68 secondes en moyenne.

D'autres exploits moins sérieux sont devenus classiques et les résultats progressent comme des records sportifs. La résolution du cube d'une seule main a demandé 14,7 secondes au champion de 2010. Avec le pied, un autre champion range un cube en 42 secondes. Le 16 novembre 2008, Milan Batick a reconstruit 4 786 cubes en moins de 24 heures. Il battait le précédent record qui était de 3 505. Notez que M. Batick réussit donc, pendant 24 heures, l'extraordinaire moyenne de 18,05 secondes par cube !

La résolution en aveugle est fascinante. Après que le joueur a observé le cube, on lui bande les yeux et il résout le cube. Le total des opérations (incluant cette fois la phase d'observation) n'exige que 31 secondes à

1. VOICI LA CONFIGURATION LA PLUS DIFFICILE rencontrée lors du calcul qui a conduit à démontrer que 20 mouvements suffisent toujours pour remettre un cube en ordre. Elle a été indiquée mouvement par mouvement, pour remettre en ordre la configuration en 20 étapes. Vous pouvez la lire et l'effectuer à l'envers pour concocter un problème difficile à votre cousin qui se croit champion du cube de Rubik.

Hiayan Zhuang, le champion de 2010. Le record en nombre de cubes pour le réarrangement en aveugle est dû à l'Indonésien Muhammad Irfi Khairul Anam, qui a réussi en 2010 à rétablir 16 cubes en moins d'une heure (temps maximum alloué) : après une période d'observation des 16 cubes, on lui bande les yeux et il refait alors les cubes les uns après les autres.



Le plus jeune manipulateur de cube avait quatre ans quand son exploit fut homologué. Le plus âgé avait, lui, 88 ans. Pour la résolution les yeux bandés, plus difficile, les âges records sont de 10 et 60 ans.

Le succès du cube de Rubik est dû au plaisir qu'on éprouve à le manipuler. Un autre cube, composé seulement de huit petits cubes aimantés, avait été inventé quelques années avant celui de Rubik. Il constituait un casse-tête plus facile, mais du même type. Il n'eut que très peu de succès, car il était trop tentant de tricher en séparant les cubes aimantés au lieu de les faire glisser. Le génie de l'invention de Rubik est plus celui d'une trouvaille mécanique que mathématique.

Parmi les plus étonnantes des variantes, le *Void Cube* est une merveille : il se présente exactement comme le cube de Rubik et permet les mêmes mouvements, sauf que tout ce qui dans le cube de Rubik sert au fonctionnement a disparu : il n'y a rien ni au centre du cube ni au centre des faces. Seuls persistent les 8 cubes de coin et les 12 cubes des milieux des arêtes qui se déplacent vraiment dans un cube de Rubik *(ci-contre)*. L'ingéniosité du mécanisme, le succès du cube, est surpassée ! Si vous voulez essayer et étudier les variantes du cube de Rubik sans avoir à effectuer de nombreux achats, des programmes informatiques gratuits en ligne (applet) permettent de jouer virtuellement (voir en particulier : www.randelshofer.ch/rubik/index.html).

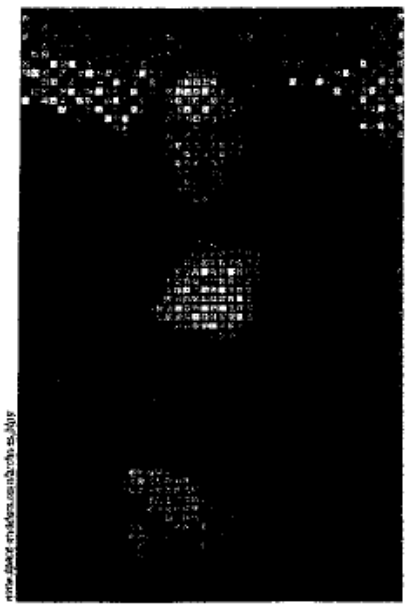
Des décomptes

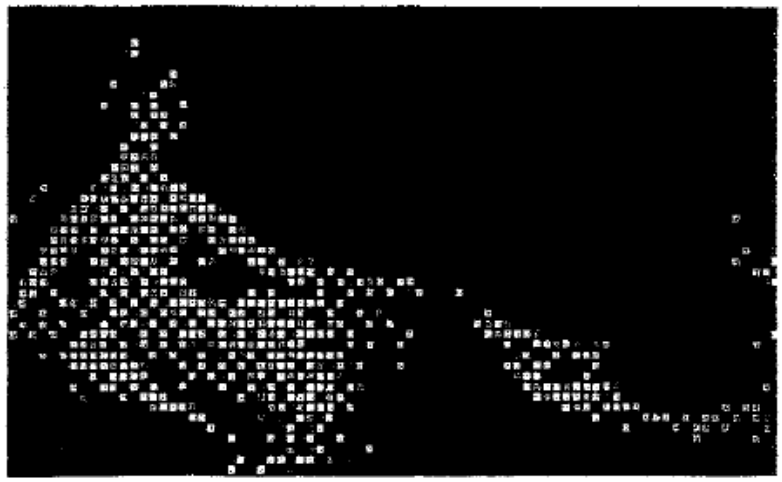
Il est facile de voir que 15 mouvements ne suffisent pas à remettre en ordre n'importe quelle configuration du cube de Rubik. À chaque fois que vous faites un mouvement (rotation d'une des faces du cube) vous avez 18 choix possibles, car chacune des six faces peut être tournée de 90, 180 ou 270 degrés. En faisant un mouvement, on peut donc atteindre 18 configurations. En réalisant deux mouvements, on peut atteindre au plus $18 \times 18 = 18^2$ configurations différentes. Etc. En effectuant 15 mouvements ou moins, vous atteindrez donc un nombre de configurations différentes inférieur ou egal à :
$18 + 18^2 + 18^3 + 18^4 + ... + 18^{15} = 7{,}1435 \times 10^{18}$.

Or ce nombre est inférieur au nombre de configurations du cube de Rubik, égal à $4{,}3 \times 10^{19}$. En faisant 15 mouvements ou moins, on ne peut donc pas atteindre toutes les configurations possibles. Si on place le cube dans l'une de ces configurations non atteintes, il faudra donc au moins 16 mouvements pour remettre le cube en ordre. Un raisonnement similaire, mais un peu plus subtil, montre que certaines configurations ne pourront être remises en ordre qu'avec 17 mouvements ou plus.

Refaire le cube, c'est bien, mais le refaire en opérant un minimum de mouvements c'est mieux... et plus délicat. La WCA a prévu une épreuve qui mesure cette aptitude des concurrents à l'économie dans l'action. Les règles de l'épreuve sont les suivantes :

(a) Le joueur examine le cube mélangé qu'on lui soumet et l'étudie soigneusement pendant une heure en s'aidant de crayons, de papier, de trois cubes auxiliaires et de gommettes colorées s'il le souhaite.





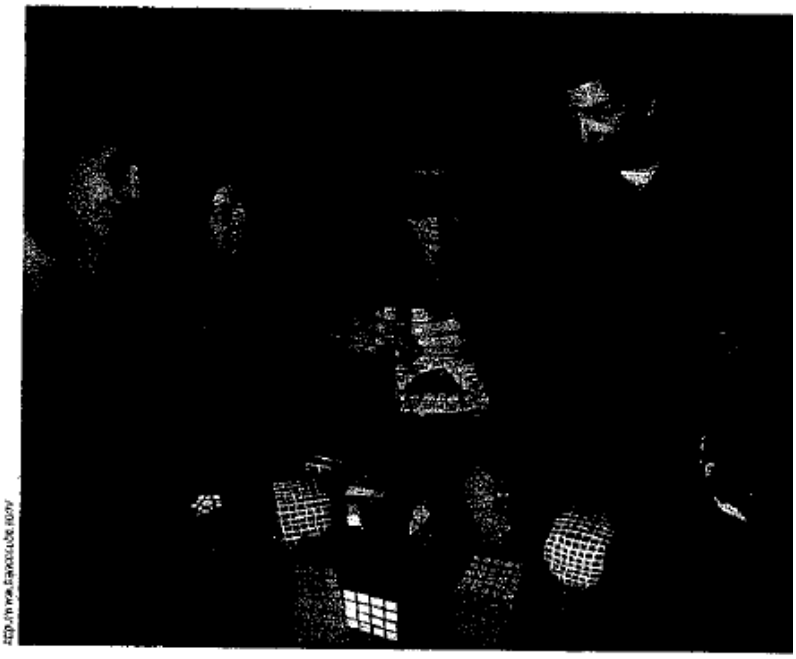
2. « RUBIKCUBISME » : ces images sont décomposées en pavés de neuf pixels, que l'artiste réalise alors en prenant autant de cubes de Rubik qu'il faut et en les manipulant pour faire apparaître sur une des faces les différents pavés colorés de neuf pixels. Ils composent ainsi patiemment le tableau avec six couleurs. D'autres œuvres de ce type sont visibles sur www.space-invaders.com/archives.html.

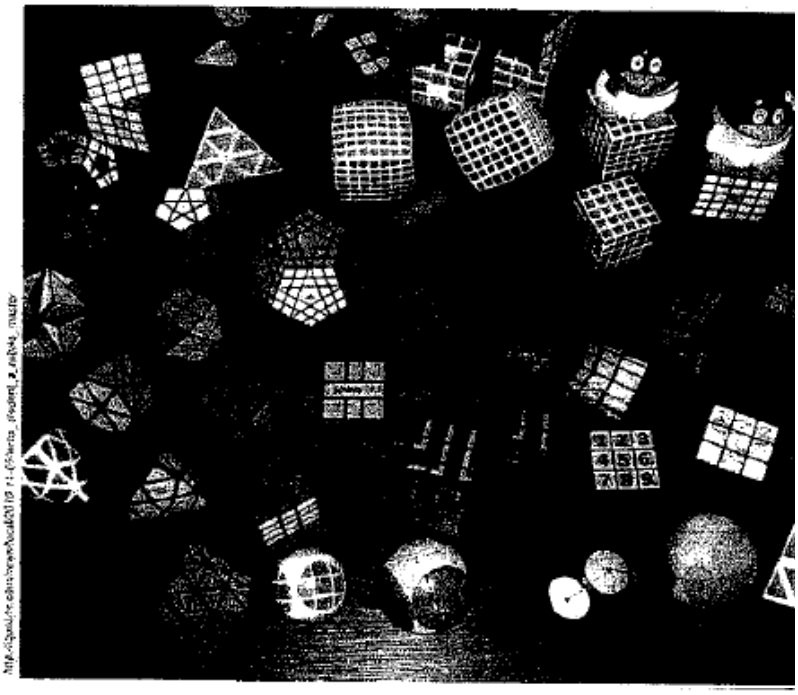
(b) L'heure écoulée, il remet sa solu
sous forme écrite en utilisant un cod
standardisé pour indiquer la meilleure s
de mouvements qu'il a trouvée.

La longueur d'une solution est le nom
de rotations de face, une rotation pouv
être d'un quart de tour ou d'un demi-tou
champion 2010 est le Hongrois (tiens, tie
 Istvan Kozza, qui a proposé une solutio
22 mouvements pour le cube mélangé q
lui avait soumis. Il faut noter que ce nom
est particulièrement faible, car les métho
de reconstitution de cubes expliquées d
les livres ou les multiples pages Internet
en décrivent conduisent à des reconst
tions en 60 mouvements environ, ou 30 p
les meilleures... qui sont plus difficile
apprendre. Ce score de 22 n'est pas dû
hasard qui aurait fait que le problème sou
en 2010 était particulièrement simple
preuve en est qu'en 2009, le champion
cette épreuve avait réussi lui aussi en 22 m
vements. En 2008, le record était de 27,
2007 de 35, en 2006 de 28, en 2005 de
Les joueurs progressent et une question vi
à l'esprit : peut-on toujours faire 22, ou moi
Plus précisément, quel est le nombre de m
vements qu'un joueur parfait aura à faire d
le pire cas ?

Quand la théorie pure bute

Depuis longtemps, on soupçonne que
réponse est 20. En juillet dernier, la preu
du résultat en a été donnée. Puisque l'ét
des mouvements du cube de Rubik
ramène à l'étude d'un certain groupe mat
matique, on pouvait penser que la soluti
proviendrait de progrès mathématiques.
structure étudiée ne contient aucun ar
traire, contrairement par exemple au j
o échecs dont le graphe des parties p
sibles est compliqué, au fait des règles co
plexes. Ici, la structure du problème,
graphe aussi (*voir la figure 4*), est défi
à partir d'éléments géométriques tr
simples et la situation paraît donc idé
pour que les mathématiciens exerce
leur talent et produisent une répon
s'appuyant sur tout ce qu'ils savent d
groupes, de leur classification, de leu





3. LES VARIANTES LES PLUS CLASSIQUES du cube de Rubik sont le cube 2 × 2 × 2, le cul
(Rubik's Revenge), le cube 5 × 5 × 5 *(Professor's Cube)*. Tous donnent lieu à des épreuv
compétitions de la WCA. Les plus gros, commercialisés depuis 2008, sont le cube 6 × 6 × 6
et le cube 7 × 7 × 7 *(V-cube 7)*. Le nombre de configurations du 2 × 2 × 2 est 3 674 160.
de configurations du 3 × 3 × 3 est $4,3 \times 10^{19}$. Celui du 4 × 4 × 4 est $7,4 \times 10^{45}$, celui du 5
$2,8 \times 10^{74}$, celui du 6 × 6 × 6 est $1,57 \times 10^{116}$, celui du 7 × 7 × 7 est $1,95 \times 10^{160}$. On dé
le nombre de bits d'information contenus dans l'Univers visible ! Le succès du cube a ete
dizaines de variantes ont été inventées, dont des versions en braille pour les aveugles.

4. Le graphe du cube de Rubik

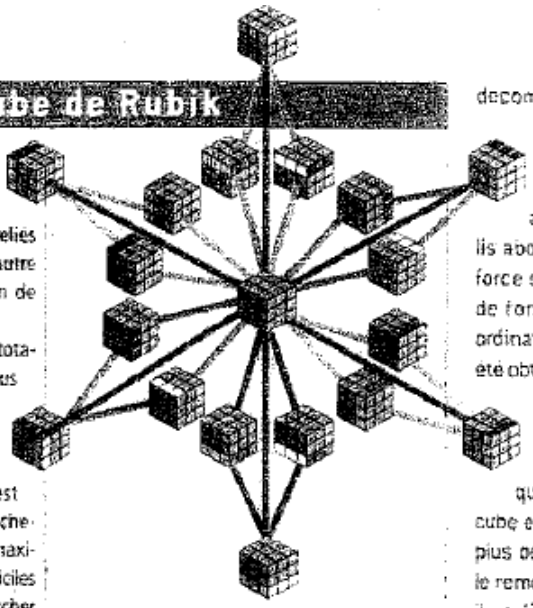
On associe au cube de Rubik le graphe de toutes ses configurations. Les nœuds du graphe sont les $4,3 \times 10^{19}$ configurations possibles et deux nœuds sont reliés par un arc si l'on peut passer de l'un à l'autre en opérant un mouvement de rotation de l'une des faces du cube.

Ce graphe, impossible à dessiner en totalité, est extrêmement symétrique, car tous les nœuds sont équivalents comme c'est le cas par exemple du graphe des sommets d'un cube. Trouver les meilleures suites de mouvements pour résoudre le cube est équivalent à rechercher les plus courts chemins dans ce graphe. Trouver le nombre maximum de mouvements qu'exigent les plus difficiles des configurations est équivalent à rechercher les configurations les plus éloignées de la configuration initiale, ce qui, à cause de la symétrie du graphe, revient à trouver le diamètre du graphe (la distance maximale séparant deux nœuds du graphe).

Les algorithmes généraux de calcul dans un graphe (calcul de plus court chemin, calcul de diamètre, etc.) ne peuvent pas s'appliquer directement à cause de la taille du graphe, et cela même en utilisant de puissants réseaux d'ordinateurs. Le diamètre du graphe a été calculé en adaptant ces méthodes et en exploitant au mieux les propriétés particulières du graphe.

L'une des idées utilisées par les chercheurs est la suivante. Pour connaître un court chemin entre deux positions A et B, on peut choisir une configuration C pas trop loin de A, puis rechercher le plus court chemin entre A et C et le plus court chemin entre C et B. La mise bout à bout des deux plus courts chemins ne conduit pas toujours au plus court chemin entre A et B, mais elle en donne un assez bon. De plus, si on fait varier C, cela permet presque à coup sûr de découvrir le plus court chemin entre A et B.

Le problème du diamètre du graphe du cube de Rubik est étudié depuis 30 ans et les progrès ont été lents jusqu'à la démonstration en juillet 2010 que le diamètre est 20. Pour mesurer les progrès, voyons la date, le nom du prouveur et le diamètre entre parenthèses : juillet 1981, Morwen Thistlethwaite (52) ; avril 1992, Hans Kloosterman (42) ; mai 1992, Michael Reid (39) ; mai 1992, Dik Winter (37) ; janvier 1995, Michael Reid (inférieur à 29 et supérieur à 20) ; décembre 1995, Silviu Radu (28) ; avril 2006, Silviu Radu (27) ; mai 2007, Dan Kunkle (26) ; mars 2008, Tomas Rokicki (23) ; août 2008, Tomas Rokicki (22) ; juillet 2010, Tomas Rokicki, Herbert Kociemba, Morley Davidson et John Dethridge prouvent enfin que le diamètre est 20.



Début de l'immense graphe des configurations du cube de Rubik.

Le tableau suivant (obtenu à l'occasion du dernier résultat) indique le nombre de nœuds du graphe ayant une distance donnée à la configuration initiale. Les dernières lignes du tableau sont des valeurs approchées en cours de calcul.

0	1
1	18
2	243
3	3 240
4	43 239
5	574 908
6	7 618 438
7	100 803 036
8	1 332 343 288
9	17 596 479 795
10	232 248 063 316
11	3 063 288 809 012
12	40 374 425 656 248
13	531 653 418 284 628
14	6 989 320 578 825 358
15	91 365 146 187 124 313
16 environ	1 100 000 000 000 000 000
17 environ	12 000 000 000 000 000
18 environ	29 000 000 000 000 000
19 environ	1 500 000 000 000 000
20 environ	300 000 000

décompositions, etc. Il n'en a rien été, la théorie pure s'est révélée inopérante !

La réponse, 20, n'a été obtenue qu'à l'issue d'une série de progrès algorithmiques s'étalant sur 20 ans. Ils aboutissent grâce à l'utilisation d'une force de calcul équivalente à 35 années de fonctionnement ininterrompu d'un ordinateur de bureau. Ce temps de calcul a été obtenu en mobilisant un ensemble d'ordinateurs prêtés par le géant Google pendant quelques semaines.

Le nombre de configurations qu'on peut obtenir en mélangeant le cube est de 43 milliards de milliards. Si en plus de le manipuler, vous le démontez et le remontez sans prendre de précautions, il y a 12 fois plus de configurations et vous n'aurez donc qu'une chance sur 12 de pouvoir le remettre en ordre. Cette propriété du cube est analogue à celle du Taquin : en démontant un Taquin et en replaçant les pièces sans précaution, vous n'avez qu'une chance sur deux de pouvoir revenir à la position initiale.

On peut s'occuper une à une des $4,3 \times 10^{19}$ configurations possibles, en recherchant à chaque fois la meilleure suite de mouvements pour l'atteindre. Trouver la suite la plus courte de mouvements pour ranger une position donnée est aujourd'hui possible grâce aux bons algorithmes mis au point par Herbert Kociemba, qui travaille sur la question depuis 1992. Pour une position donnée, il faut souvent plusieurs secondes, même avec une machine puissante, pour identifier la suite optimale de mouvements. La recherche de la pire configuration du cube de Rubik que nous obtiendrions en calculant, pour chaque configuration, la séquence optimale de mouvements avec un algorithme traitant une configuration par seconde, exigerait donc de mobiliser pendant plus de 1 300 ans le milliard d'ordinateurs qu'il y a sur Terre aujourd'hui. La méthode brutale n'est pas en mesure de donner la réponse.

Une autre idée consiste à procéder progressivement en mémorisant ce qu'on obtient et en le réutilisant. On regarde toutes les configurations qu'on atteint en un mouvement (il y en a 18) et on mémorise leur liste avec

l'information qu'elles sont atteignables en un mouvement. Partant de ces configurations, on opère un mouvement de plus de toutes les façons possibles. On mémorise les configurations nouvelles qu'on vient d'atteindre avec l'information qu'elles sont atteignables en deux mouvements.

On poursuit de la même façon et, de proche en proche, toutes les configurations possibles se trouvent mémorisées avec l'information de la longueur du plus court chemin qui permet d'y arriver (car quand on engendre une configuration pour la première fois, c'est qu'il n'y a pas de suites de mouvements plus courtes y conduisant). Quand une étape nouvelle de calcul n'ajoute plus de configuration, on arrête l'algorithme : on est certain d'avoir toutes les longueurs des plus courts chemins et on connaît donc les configurations du cube de Rubik les plus longues à ranger.

En théorie, cette approche est bien plus rapide que la précédente, car elle exploite au mieux les résultats des calculs déjà faits qu'elle engrange petit à petit. Mais elle est toute aussi impossible à cause de la quantité de mémoire bien trop importante qu'elle exige. C'est la totalité de la mémoire de tous les disques durs des ordinateurs sur Terre, de l'ordre de 10^{22} octets, qu'il faudrait utiliser pour exécuter l'algorithme de calcul progressif de tous les plus courts chemins que nous venons de décrire !

Un exploit algorithmique

Entre trop de calculs et trop de mémoires, il fallait trouver une solution médiane. Tomas Rokicki, H. Kociemba, Morley Davidson et John Dethridge ont su faire ce travail et démontrer que 20 est le nombre de mouvements nécessaires pour ranger le cube de Rubik dans les cas les plus difficiles. La méthode résulte d'une longue recherche et de la mise au point d'une série de perfectionnements pour apprivoiser la complexité du problème et exploiter à la fois la mémoire disponible sur une machine contemporaine et la capacité de calcul qu'elle donne, en s'arrangeant pour ne jamais dépasser les bornes que la technologie fixe aujourd'hui.

Outre l'utilisation des symétries du problème qui permet d'en réduire un peu la taille, l'idée a été de séparer le problème en un grand nombre de sous-problèmes que la mémoire disponible sur un ordinateur permet de traiter globalement par une méthode apparentée à celle progressive évoquée. Le problème a ainsi été découpé en 2 217 093 120 ensembles de 19 508 428 800 configurations chacun. L'utilisation des symétries a ramené le nombre d'ensembles à traiter à 55 882 296.

Une variante de l'algorithme de H. Kociemba exploite l'information que le pire cas ne demande pas moins de 20 mouvements. On sait depuis 1995 que certaines configurations ne peuvent être résolues en moins de 20 mouvements, et donc, dès que pour une configuration donnée, on connaît une solution en moins de 20 mouvements, et même si ce n'est pas la meilleure, on peut se dispenser d'en rechercher une plus courte.

Le long calcul mené pour arriver au résultat que 20 mouvements, même dans le pire cas, suffisent à remettre en ordre un cube de Rubik, a produit d'autres informations intéressantes. En moyenne, le nombre de mouvements nécessaires est 17,7. Les configurations exigeant 20 mouvements sont assez rares : il y en a environ 300 millions, ce qui signifie qu'une configuration prise au hasard a moins d'une chance sur 100 milliards de nécessiter 20 mouvements.

Ces informations font maintenant comprendre que les joueurs humains qui réussissent en 22 mouvements réalisent un exploit pas très lointain de l'optimum, chose remarquable. Cependant, malgré tous leurs efforts et tous les entraînements auxquels les joueurs humains se soumettent, ils ne réussissent pas à découvrir les meilleures séquences de mouvements.

Ainsi, ce jeu inventé il y a plus de 30 ans a défié le monde des informaticiens qui n'ont pu que difficilement résoudre le problème de la pire configuration. Le jeu défie aussi la communauté des mathématiciens qui, sur une question simple purement algébrique et abstraite, a dû, pour l'instant, laisser la machine l'emporter en proposant un résultat qu'aucun théoricien ne peut contrôler ni valider à la main. ■

L'AUTEUR

Jean-Paul DELAHAYE
est professeur à l'Université de Lille et chercheur au Laboratoire d'informatique fondamentale de Lille (LIFL)

BIBLIOGRAPHIE

J. Slocum, **The Cube. The Ultimate Guide to the World's Bestselling Puzzle**, Black Dog & Leventhal Publisher Inc., 2009.

T. Rokicki, **Twenty-two moves suffices for the Rubik's cube**, *The Mathematical Intelligencer*, vol.32(1), pp.33-40, 2010 http://www.springerlink.com/content/q088143tn805k124/fulltext.pdf

D. Joyner, **Adventures in Group Theory : Rubik's Cube, Merlin's Magic and Other Mathematical Toys**, The Johns Hopkins University Press, 2008.

SUR LE WEB

T. Rokicki et al., **God's Number is 20** (consulté le 11-11-2010), http://www.cube20.org/

Cubeland : http://cubeland.free.fr/index.html

Francocube : http://www.francocube.com

Site officiel du cube de Rubik : http://www.rubiks.com/

World Cube Association : http://www.worldcubeassociation.org/index.php

Wikipedia : Optimal solutions for Rubik's Cube http://en.wikipedia.org/wiki/Optimal_solutions_for_Rubik's_Cube

Annexe 3 – Article « Eléments pour la formation initiale des professeurs d'école à partir de l'algorithme de Kaprekar » de J.-C. Rauscher (Extrait des actes du XXXIVème colloque : *Expérimentation et modélisation dans l'enseignement scientifique : quelles mathématiques à l'école ?* - Troyes 2007)

ÉLEMENTS POUR LA FORMATION INITIALE DES PROFESSEURS D'ECOLE À PARTIR DE L'ALGORITHME DE KAPREKAR

Jean-Claude Rauscher
Maître de conférences, IUFM d'Alsace
DIDIREM Paris 7
Jc.Rauscher@wanadoo.fr

Résumé
L'algorithme de D.R. Kaprekar, mathématicien indien de Devlali, publié en 1949 peut être le support d'une aventure, riche d'étonnements et de réflexions mathématiques. Dans le cadre de cet atelier, les participants ont vécu cette aventure qui donne lieu à des démarches d'observations, d'expérimentations, de communications et de validations tant au niveau de formateurs que d'étudiants et d'élèves de cycle trois, de collège et de lycée. Parallèlement, les apports de ce support dans le cadre de la formation initiale de futurs professeurs ont été discutés et mis en lumière : présentation et mise en œuvre de démarches d'enseignement des mathématiques (Brousseau 1972), repérage des finalités de l'enseignement des mathématiques (G. Vergnaud, 1987), réactualisation des connaissances de la numération, repérage de l'importance de la prise en compte de différents registres d'écriture en mathématique (R. Duval, 1995), prise de conscience de modalités de pensée chez les élèves (C. De Block-Docq, 1994).

I – L'ALGORITHME DE KAPREKAR ET LA QUESTION DE SON UTILISATION

Les travaux du mathématicien autodidacte D.R. Kaprekar sur les nombres entiers, relayés par Martin Gardner, continuent d'une part à questionner mathématiciens et informaticiens et d'autre part donnent des supports d'activités pour les enseignants en mathématique. Il en est ainsi de l'algorithme dit de Kaprekar publié en 1949 qui consiste à choisir un nombre entier n écrit généralement en base 10 et de former le plus grand et le plus petit nombre à l'aide des chiffres de n. Au nombre n on associe ensuite la différence $K(n)$ de ces deux nombres et on recommence le processus avec les chiffres de $K(n)$. Cet algorithme génère des phénomènes remarquables. Ils sont parfois présentés dans le cadre de mathématiques dites récréatives. Les questions qu'ils suscitent servent aussi de supports pour des activités d'apprentissages en classe. Comme exemple, citons la proposition de Gérard Chauvat (1980), qui s'appuie sur l'algorithme pour développer des activités en « Arithmétique élémentaire en classe de $5^{ème}$ » utilisant la notion de classes d'équivalence. L'arithmétique ayant retrouvé une place explicite dans les classes de lycée, on les trouve aussi évoqué dans des manuels scolaires de terminale S. Pour ma part j'ai rencontré cet algorithme à partir d'une idée « d'activité » du manuel de $6^{ème}$ de l'IREM de Strasbourg (1986). C'est à partir de là que j'ai élaboré et mis à l'épreuve des séquences d'enseignement comportant pour les élèves de collège des phases

d'observations, de conjectures d'expérimentations et de preuves. Par la suite, j'ai utilisé ce travail comme support de formation en PE et PLC.

La motivation des participants à cet atelier était double. Pour certains, qui avaient déjà entendu parlé de cet algorithme sans le connaître précisément, il s'agissait de l'appliquer effectivement. Pour tous ensuite, il s'agissait de savoir ce qui est possible de faire avec cet algorithme dans le cadre de classes de cycle 3 ou de collège et dans le cadre de la formation PE ou PLC. Bien entendu, l'intérêt est de dépasser le côté simplement spectaculaire ou récréatif des phénomènes qu'il génère. C'est dans cette perspective que j'ai conçu un déroulement pour l'atelier qui permettait dans un premier temps de découvrir ce support puis de l'interroger comme support de formation, avec l'éclairage aussi de mes expériences dans ce domaine. Le déroulement de l'atelier sera décrit dans la partie II.

Voici les dimensions qui seront abordés ensuite dans le compte rendu.

À un premier degré, les activités qui s'appuient sur l'algorithme de Kaprekar provoquent chez les élèves et les étudiants des démarches d'observations, d'expérimentations, de communications et de validations (G. Brousseau, 1972) dans les domaines de la numération. J'ai eu maintes fois l'occasion de le vivre avec des élèves de début de collège et par la suite avec des étudiants se destinant à l'enseignement (professorat d'école ou de lycée collège). Cette fois ci j'ai partagé cette aventure avec les formateurs de mathématiques en IUFM qui ont participé à cet atelier. Il sera rendu compte de leurs observations, de leurs surprises, de leurs questions.

A un deuxième degré, ces activités peuvent être analysées en termes d'apports dans les apprentissages pour les élèves de fin de cycle 3 et en collège. Habituellement, je mène cette analyse avec les futurs professeurs en formation initiale. Ici, nous l'avons menée entre collègues formateurs : que peuvent apprendre les élèves à partir de cette activité et quels scénarios d'enseignement en classe sont les plus susceptibles de favoriser ces apprentissages ? L'analyse se base sur ma propre expérience et a été enrichie par les apports des uns et des autres.

Enfin on peut analyser ces activités en termes d'apports pour la formation des futurs professeurs tant au point de vue des contenus mathématiques sous-jacents, question importante pour les étudiants préparant le concours qu'au point de vue, tout aussi important pour eux, de questions d'enseignement des mathématiques à l'école et au collège.

II – LE DEROULEMENT DE L'ATELIER

Pour commencer, les participants à l'atelier ont découvert et exploré le fonctionnement de l'algorithme de Kaprekar pour trois chiffres à partir du programme de calcul suivant :

1) *Choisir un nombre entier de trois chiffres.*
2) *A l'aide de ces trois chiffres, fabriquer le nombre entier le plus grand possible.*
3) *A l'aide de ces trois chiffres, fabriquer le nombre entier le plus petit possible.*
4) *Calculer la différence des nombres obtenus en 2) et 3)*

5) Recommencer (à partir de l'instruction 2) avec le résultat obtenu. etc.
Après cette exploration, l'analyse du support et de sa richesse a été amorcée individuellement par écrit autour des points suivants :

- *Réactions et questions par rapport au support et par rapport au comportement du groupe de participants*
- *Connaissances et compétences qu'il peut développer chez les élèves*
- *Connaissances et compétences qu'il peut développer chez les étudiants et PE*
- *Modalités de formation envisageables pour exploiter cette situation.*

Au cours de cet atelier, j'ai eu l'occasion de rapporter et de soumettre à analyse la façon dont pour ma part j'exploitais cette situation avec des élèves de collège puis ensuite avec des étudiants se destinant à l'enseignement. L'écrit individuel initial a permis à tout un chacun de se faire une idée a priori et de nourrir les présentations et discussions qui ont suivi.

III – QUE FAIRE AVEC CET ALGORITHME DANS LE CADRE DE L'ENSEIGNEMENT EN CYCLE 3 ET AU COLLEGE ?

III – 1 La découverte de l'algorithme et de ses effets par les participants

Au cours de l'atelier, dans cette phase de découverte de l'algorithme, la surprise a joué à plein, tout comme elle joue habituellement dans un groupe de cycle 3 ou de collège : l'étonnement se lit sur les visages lorsque l'on s'aperçoit que les suites se stabilisent sur 495, on consulte ses voisins pour confirmer cette observation, etc... Quelques réactions écrites à ce sujet :

- *« C'est étonnant ! Peut on généraliser à 2, 4, ..., n chiffres ? Démonstration ? »*
- *« Etonnant ! Le résultat se stabilise. Pourquoi ? Est-ce qu'on peut généraliser ? »*
- *« C'est magique ! Ca marche toujours ? Comment ça marche ? »*

Mais on remarque dans ces réactions écrites, une attitude qui est spécifique de personnes qui ont pratiqué des mathématiques : il y a d'emblée un soucis de savoir si cette observation peut se généraliser et de chercher des explications ou de démontrer. On rencontre évidemment souvent ces réactions chez les étudiants ou en lycée. Les forums sur internet, les activités présentées dans des manuels de lycée, ou même de collège (Chauvat; 1980) à ce sujet, se focalisent exclusivement sur la résolution de ces questions mathématiques.

III – 2 Une piste sans issue ?

À partir de la constatation initiale, il s'agit pour un professeur de décider comment il va poursuivre l'activité. En l'occurrence, la question est de savoir si cette préoccupation de prouver cette stabilisation peut être partagée d'emblée avec les élèves. C'est ce que semblent penser a priori plusieurs participants de l'atelier qui trouvent globalement ce support « *très motivant pour les élèves* et *intéressant pour les inciter à poursuivre une recherche* ». Un autre participant précise : « *Il s'agit de trouver des règles, les vérifier, puis les prouver ; s'interroger sur le pourquoi ? comment ? toujours ?* ». Néanmoins cette perspective est relativisée par certains qui évoquent bien « *l'apparition d'un fait expérimental pour les élèves* » mais qui soulignent que « *le désir de preuve peut être suscité mais que tout dépend du rapport des élèves à l'expérience* ». Pour ma part, mes observations en classe me font adhérer complètement à cette prudence au niveau d'un cycle 3 ou du collège. En effet l'étonnement des élèves ne se transforme pas immédiatement en désir manifeste de comprendre le pourquoi de ce phénomène. Il y a un étonnement, certes, mais qui se rapporte plutôt à une certaine vision des mathématiques à cet âge : « *c'est magique !* » ; une magie très vite intégrée : « *C'est comme çà ! »*. Van Hiele à propos d'expériences en géométrie relève que « *les enfants s'étonnent plus de ce qui ne va pas que de ce qui va* » (rapporté par DE BLOCK-DOCK C. ; 1994). En l'occurrence beaucoup d'élèves sont plus sensibles aux ruptures et contres exemples de la règle observée qu'à son accomplissement. Ces ruptures et contres exemples suscitent en effet des questionnements spontanés : « *Pourquoi ça ne marche pas dans ce cas là ?* ». Il s'agit là d'une attitude qui, prise en compte par l'enseignant, permet de faire avancer les élèves dans leurs pensées plus sûrement qu'une injonction à « démonter » la propriété. De plus, la situation proposée comporte un potentiel d'apprentissage concernant le calcul et la numération décimale qui risque d'être négligé si on focalise immédiatement et uniquement l'attention des élèves sur la démonstration. Ce potentiel fut d'emblée repéré par les participants à l'atelier : « *Cette situation est intéressante pour les élèves au niveau de la numération et du calcul (soustractions avec retenues) et pour comprendre ce qu'est un algorithme »;* ou encore : « *Elle met en jeu la comparaison de nombres entiers, la connaissance du système décimal de position, la désignation écrites d'entiers* ».

Mais comment alors, à la fois faire progresser les élèves dans le domaine mathématiques pour les faire sortir de la pensée magique du « *c'est comme çà !* », et, parallèlement réactiver leurs connaissances dans le domaine du calcul et de la numération décimale ? Au cours de l'atelier, cette question étant installée, j'ai eu l'occasion de présenter ma façon de faire dans les classes de sixième, où je commençais pratiquement l'année avec cette activité en poursuivant ces deux objectifs.

Le fait de faire vivre cette activité aux participants de l'atelier comme je la faisais vivre avec des élèves et comme je la fais vivre avec des étudiants, nous a permis par la suite de continuer l'analyse des apports de l'activité pour les élèves et pour les futurs professeurs.

III – 3 Une façon d'exploiter la richesse potentielle de la situation avec des élèves

III – 3.1 Le travail sur les nombres à trois chiffres

Dans une première phase de travail qui peut s'étaler sur plusieurs séances, je demande aux élèves de m'annoncer les suites obtenues et les copie telles quelles au tableau. De cette façon, on obtient un corpus que l'on peut soumettre à l'observation de tous les élèves de la classe. En effet, outre la surprise initiale, il y a bien d'autres régularités à remarquer.

Remarque : la particularité des cas où on prend des nombres à trois chiffres identiques avait été évoquée préalablement par les élèves (444-444 =0) et ces cas, en connaissance de cause, étaient écartés des corpus à explorer.

Voici un exemple de corpus (le nombre choisi au départ est encadré) :

547 ; 297 ; 693 ; 594 ;495
962 ; 693 ; 594 ; 495
393 ; 614 ; 515 ; 396 ;594 ; 495
666 ; 0
765 ; 198 ; 792 ; 693 ; 594 ; 495
409 ; 891 ; 693 ; 594 ; 495
836 ; 495
432 ; 198 ; 782 ; 594 ; 495

Afin que chaque élève puisse avoir l'occasion et le temps de faire cette exploration, cette observation est sollicitée par écrit individuel : « Ecrivez toutes les remarques que vous faites en observant les suites obtenues ? ». Quelques remarques sont ensuite transcrites au tableau. Le travail de la classe porte ensuite sur l'analyse de ces remarques.

L'attention est en particulier portée sur leurs formulations. Les nuances à propos de remarques a priori identiques permettent en effet d'expliciter quelques savoirs en jeu. Par exemple, la comparaison des trois remarques suivantes permettent de revenir sur le vocabulaire utilisé pour signaler la position des chiffres dans l'écriture d'un nombre et aussi sur la différence entre chiffre et nombre :

- *« Au milieu, il y a toujours un 9 »*
- *« Le chiffre du milieu est toujours un 9 »*
- *« Le chiffre des dizaines est 9 »*
- *« Le nombre des dizaines est 9 »*
- *« Dans la suite des nombres, le chiffre des centaines baisse d'une unité chaque fois »*

De même :

- *« Quand on additionne le premier et le dernier chiffre on obtient toujours 9 »*
- *« Quand on additionne le chiffre des centaines et le chiffre des unités on obtient 9 »*

Le mot *« toujours »* qui apparaît fréquemment dans les observations est intéressant, tout comme son absence. Des élèves mettront ce mot ou la généralité des affirmations en

question en relevant des contres exemples. Ces contres exemples peuvent être de différentes natures :
- Certains de ces contres exemples, inciteront les élèves à vérifier spontanément les résultats des calculs. Par exemple dans les lignes 3 et 8 de l'exemple de corpus, on trouve des contres exemples qui correspondent à des erreurs de calculs (pour la ligne 3 : erreur dans la soustraction pour le rang des dizaines dans les deux premières étapes ; pour la ligne 8 : erreur dans l'élaboration du plus petit nombre possible à partir des chiffres utilisés).
- D'autres contres exemples donneront l'occasion aux élèves de prendre conscience que les propriétés qu'ils ont remarquées ne sont pas vraies sur tout le corpus (ligne 4) ou ne sont même pas vraies sur l'ensemble des nombres d'une suite.

Dans un cas comme dans l'autre, ces contres exemples font passer les observations exprimées par les élèves du statut d'observation locale au statut d'observation dont il faut étudier le domaine de validité. Le pas qui consiste ensuite à donner le statut de conjecture à vérifier aux observations émises est alors possible et est franchi par bon nombre d'élèves. En particulier, la remarque qui consiste à dire que *« le dernier nombre de la suite est toujours 495, sauf pour les nombres de trois chiffres identiques »* est alors mise en question, laissant la place à une étude exhaustive des cas, soit systématique (le travail est partagé dans la classe), soit par classe d'équivalence de nombres (nombres ayant les mêmes chiffres en considérant le plus grand et le plus petit).

Un tel travail peut clore la première phase de travail avec les élèves. Mais en fonction de l'avancement de la réflexion de la classe, on peut demander aux élèves de chercher une explication au fait que le chiffre des dizaines des différences calculées est 9 par examen attentif des soustractions posées, explication que certains formulent plus ou moins complètement. C'est une occasion de revisiter l'algorithme de la soustraction posée : quand on fabrique le plus grand nombre et le plus petit, on a toujours le même chiffre pour la colonne des dizaines, pour la colonne des unités il y a toujours une retenue à reporter sur la colonne des dizaines puisqu'on enlève un nombre à un nombre plus petit. Il s'agit là d'une première preuve qui ne consiste pas à faire une étude exhaustive des cas. Un coin du voile de la magie de l'algorithme commence à se lever…

III – 3.2 « Un drôle de truc » et le travail avec les nombres à deux chiffres

La deuxième phase de travail qui pourra occuper plusieurs séances aussi, est amorcée à la fin de la première phase par l'installation d'une question que s'approprie avec curiosité les élèves. En effet lorsque je demande à un élève de m'annoncer le chiffre choisi, je lui annonce quasi immédiatement la suite des nombres qu'il a obtenu par application de l'algorithme de Kaprekar. Les élèves sont étonnés : *« Mais comment vous faites si vite, Monsieur ? Vous devez avoir un truc ! »* Je leur dis qu'à la place du programme de calcul initial, j'utilise un autre programme, plus facile que je leur demande de trouver. Ils entament alors une recherche et les idées, souvent astucieuses, ne manquent pas. Mais en général leurs propositions ne marchent que sur une suite ou parfois même sur une partie seulement de la suite. Par exemple ils proposent de mettre 9 au milieu et d'ajouter 1 au chiffre des centaines et de retrancher 1 au chiffre des unités. Ces échecs leur permettent de prendre conscience qu'il s'agit de trouver *« un programme qui marche à tous les coups ! »* Mais ce n'est pas chose aisée ! C'est pourquoi la suite du travail consistera à reprendre cette question pour l'algorithme de

Kaprekar avec des nombres à deux chiffres. Le travail à ce sujet passe d'abord par les mêmes étapes que pour l'algorithme appliqué aux nombres à trois chiffres.

1ère étape : programme de calcul appliqué à plusieurs exemples.

Il s'agit d'une phase de travail individuel. Très vite les élèves remarquent que le nombre clé qui apparaît ici est le nombre 9 auquel on aboutit si l'on ne choisit pas un nombre de deux chiffres identiques. Mais ce résultat pose problème et laisse les élèves perplexes : la question est de savoir que faire lorsqu'un résultat de soustraction ne comporte qu'un seul chiffre, en l'occurrence 9. Certains élèves demandent ce qu'il faut faire dans ce cas. Il y a une décision à prendre ! Je ne réponds pas mais renvoie à différentes décisions possibles adoptées implicitement par d'autres élèves. Certains considèrent en fait que c'est fini puisqu'on n'est pas dans les conditions de réaliser l'étape suivante. D'autres calculent 9-9 et leur suite finit par 0. Certains considèrent qu'ils ont deux chiffres 0 et 9 et continuent l'algorithme en calculant 90-9=81. Ce sont les plus nombreux car ils sont focalisés sur le résultat de la soustraction posée qui leur donne comme résultat 09. Ils sont attentifs aux chiffres obtenus et non pas à l'aspect « nombre » du résultat. A la fin ils retombent sur 09 et comprennent qu'ils sont sur une boucle. Mais en fait, au-delà de ces différentes positions possibles, aucun élève n'adopte spontanément l'attitude d'explorer ce qui se passe suite à chacune des décisions possibles. En fait une telle attitude qui serait celle d'un chercheur en mathématique qui explore les limites de la formulation initiale de l'algorithme n'apparaît pas spontanément chez les élèves. C'est une synthèse collective explicitant toutes ces possibilités qui permet en fait de leur montrer qu'une telle liberté d'exploration est possible en mathématiques. Cette synthèse permet aussi de clore le débat et de prendre une décision collective qui est de considérer qu'on notera les suites jusqu'à ce qu'on obtienne 9.

2ème étape : récolte d'un corpus et analyse.

Ici aussi ne figurent pas de cas où on prend des nombres constitués de deux chiffres identiques.

Voici un exemple de corpus et les remarques qu'il suscite chez les élèves :

49 ; 45 ; 9
67 ; 9
90 ; 81 ; 63 ; 27 ; 45 ; 9
28 ; 54 ; 9
62 ; 36 ; 27 ; 45 ; 9
86 ; 18 ; 63 ; 27 ; 45 ; 9
19 ; 72 ; 45 ; 9
73 ; 36 ; 27 ; 45 ; 9

Remarques à propos de ces suites :

- *« Elles finissent toutes par 9 »*
- *« Ce sont des nombres de la table de 9 »*
- *« La somme des chiffres des nombres est 9 »*
- *« Ce sont des multiples de 9 »*

On peut remarquer des enjeux d'apprentissages semblables à ceux qui apparaissent lorsqu'on applique l'algorithme aux nombres à trois chiffres. En particulier la sensibilisation au problème de la formulation des remarques et à la généralisation des observations réalisées est à nouveau envisageable et une étude exhaustive des cas peut

confirmer les observations. Néanmoins les élèves ont ici l'occasion de travailler et d'approfondir plus particulièrement leurs connaissances sur des nombres plus familiers, nombres à deux chiffres, multiples de 9 etc.

3ème étape : recherche d'un programme plus facile pour calculer la suite des nombres.

La situation avec les nombres à deux chiffres est encore mise à profit pour instiller à nouveau l'idée qu'un programme plus facile que le programme initial existe. Pour cela je montre à nouveau que je peux réciter la suite des nombres obtenus à partir du nombre initial choisi par les élèves et ceci quasi instantanément. Dans cette phase, j'invite les élèves à chercher le procédé qui permet de trouver la suite très rapidement sans recourir au programme initial. Les élèves tâtonnent et proposent des procédés qui sont discutés par la classe. Très souvent, ils donnent à nouveau des procédés qui ne sont valables que sur une partie d'une suite ou seulement sur certaines suites. Ils trouvent plus difficilement un procédé qui s'applique de façon générale. En fait, ils cherchent surtout des procédés basés sur l'obtention séparée de chaque chiffre des nombres de la suite. Certains de ces procédés sont valables. Comme par exemple :

- *pour trouver le premier chiffre (celui des dizaines), je calcule la différence des deux chiffres du nombre initial et je diminue le résultat de 1 ;*
- *pour trouver le deuxième chiffre (celui des unités), je calcule la différence de 9 et du premier chiffre.*

Plus rarement, ils cherchent des procédés qui permettent de trouver globalement le nombre de la suite. En fait, il arrive souvent que la recherche piétine. Pour débloquer la situation, on peut, sans donner la solution, rappeler que les nombres obtenus dans les suites sont des multiples de 9 et demander de repérer quels sont les multiples de 9 correspondants aux différents éléments d'une suite. Par exemple pour la suite issue de 53, à savoir 18, 63, 27, 45 et 9, on repère : 18=**2**x9, 63=**7**x9, 27=**3**x9, 45=**5**x9 et 9=**1**x9. Une fois que ce repérage figure au tableau, les élèves trouvent assez aisément le procédé. On leur demande de le rédiger sous forme de programme de calcul. En voici un exemple :

- *calculer la différence des deux chiffres du nombre initial ;*
- *multiplier cette différence par 9.*

Se pose alors la question de la validation de ces procédés. Les élèves auraient tendance à continuer à les entériner sans discussion. On peut alors leur rappeler que certains des procédés proposés précédemment ne se sont révélés que partiellement valables. Pour continuer le travail avec les élèves, la nécessité d'une validation étant partagée, le professeur peut alors envisager trois possibilités.

La première, non satisfaisante, est de dire qu'ils sont validés par des outils mathématiques qui ne sont pas encore à leur portée.

La deuxième, en revanche est à la portée de tous les élèves. Comme il y a un nombre raisonnable de cas à envisager (il y en a 82), une vérification exhaustive est à nouveau possible. La détermination des cas à envisager est déjà une tâche intéressante pour les élèves. Le travail peut ensuite être partagé dans la classe.

La troisième possibilité est de faire chercher aux élèves des preuves autres que des vérifications exhaustives. Nous entrons ici dans des démarches explicatives qui continuent à dévoiler les mystères de l'algorithme de Kaprekar. Mais ces démarches ne sont pas à la portée de l'ensemble des élèves et demande prudence de la part du

professeur. Mais très tôt néanmoins, on peut observer quelques propositions pertinentes, comme celle que nous allons maintenant évoquer à propos d'un algorithme qu'on trouve et qu'on justifie en se centrant sur l'algorithme de la soustraction posée à partir d'un exemple numérique précis comme ici à propos de 72-27 :

$$\begin{array}{rr} 7 & {}^{1}2 \\ -\ {}_{1}2 & 7 \\ \hline 7\text{-}1\text{-}2 & 2\text{+}10\text{-}7 \end{array}$$

Tout comme on a pu montrer que dans le cas de l'algorithme appliqué aux nombres à trois chiffres, le chiffre des dizaines est toujours 9, on peut accéder ici à une explication en prenant conscience d'invariants en jeu dans la soustraction posée. En effet, quels que soient les exemples envisagés, la nécessité d'augmenter le chiffre des unités du nombre le plus grand d'une dizaine et de répercuter ce fait sur le chiffre des dizaines du nombre le plus petit en l'augmentant de 1 (qui représente une dizaine) se retrouve dans tous les cas à cause des contraintes fixées par l'algorithme de Kaprekar initial. Chaque fois en effet, puisque pour obtenir le nombre le plus grand, on relègue nécessairement le chiffre le plus petit au rang des unités et pour avoir le nombre le plus petit on relègue nécessairement le chiffre le plus grand au rang des unités. Il s'agit alors de traduire les effets de ces constatations sur l'algorithme de la soustraction posée en un programme de calcul qui est ici :

- *pour trouver le chiffre des unités, j'ajoute 10 au plus petit des chiffres du nombre initial et je calcule la différence du résultat obtenu avec le plus grand des deux chiffres du nombre initial ;*
- *pour trouver le chiffre des dizaines, je calcule la différence des deux chiffres du nombre initial et je diminue le résultat de 1.*

L'aventure avec des élèves de cycle 3 ou de début de collège peut s'arrêter ici, après être néanmoins revenu au cas de l'application de l'algorithme de Kaprekar aux nombres à trois chiffres et trouver alors de nouvelles remarques à faire ou de nouveaux programmes remplaçant l'algorithme initial (laissons au soin du lecteur cette petite recherche).

III – 3.3 Prolongements dans les classes ultérieures.

Le lecteur aura constaté que nous avons en fin de compte basculé dans une démarche de généralisation qui est au seuil de l'entrée dans le monde algébrique. De quoi reprendre et prolonger l'aventure avec des élèves plus avancés dans leur scolarité ! En effet, on peut aussi avoir l'idée de formaliser la situation en remplaçant les variables de la situation, à savoir les chiffres des nombres par des lettres, a et b par exemple. Cette formalisation permet de trouver la valeur de chacun des chiffres du nombre obtenu comme l'indique le schéma suivant :

$$\begin{array}{rr} a & {}^{1}b \\ -\ {}_{1}b & a \\ \hline a\text{-}1\text{-}b & b\text{+}10\text{-}a \end{array}$$

Remarque : dans cette formalisation nous supposons évidemment que $a>b$ et donc que ab est le nombre le plus grand possible obtenu à partir des chiffres a et b. Ceci ne se fait pas d'emblée par les élèves, et même beaucoup d'étudiants se demandent s'il ne

faut pas considérer les deux cas $a>b$ et $b>a$ à partir du nombre initial ab choisi avant de comprendre qu'on aura considéré l'ensemble des cas en supposant a priori dans la formalisation algébrique de l'algorithme que $a>b$.

Malgré cette difficulté au départ, l'entrée dans ce monde algébrique fournit évidemment des outils puissants pour valider les programmes remplaçant l'algorithme initial ou pour trouver de nouveaux programmes. On se focalise ici sur l'algorithme de l'obtention séparée de chaque chiffre du nombre obtenu. Mais l'hétérogénéité des écritures utilisées pour formaliser la situation et la tentative de l'homogénéiser permettent d'envisager des issues intéressantes. En effet lorsqu'on écrit les nombres sous la forme ab et ba et le résultat obtenu sous la forme $\boxed{a-1-b}$ $\boxed{b+10-a}$, les lettres a et b ont des statuts différents. Dans le premier cas (ab et ba), elles désignent les chiffres des dizaines et des unités : il ne s'agit pas à proprement parler d'une écriture algébrique classique qui désignerait le produit des nombres a et b. Dans $(a-1-b)$ et $(b+10-a)$, en revanche, elles désignent les nombres a et b. Lorsqu'on essaye de donner la valeur du nombre dont le chiffre des dizaines est $\boxed{a-1-b}$ et le chiffre des dizaines est $\boxed{b+10-a}$ en fonction des nombres a et b, on est amené à écrire qu'il s'agit de $10(a-1-b)+(b+10-a)$. Une réduction, puis une factorisation de cette expression algébrique aboutit alors à $9(a-b)$ qui est une confirmation du programme qui se focalise non pas sur l'obtention de chaque chiffre du résultat de la soustraction mais sur le résultat comme nombre considéré globalement et non comme un assemblage de deux chiffres :

- *calculer la différence des deux chiffres du nombre initial ;*
- *multiplier cette différence par 9.*

De façon encore plus directe on a également une validation du procédé en transformant l'écriture ab désignant le nombre dont le chiffre des dizaines est a et le chiffre des unités b, en $10a+b$ et de ba en $10b+a$. En effet, la différence s'écrit alors $(10a+b)-(10b+a)$ qui donne $9(a-b)$.

Mais rappelons que nous sommes ici entrés dans un domaine où les démarches ne sont pas en général à la portée de jeunes élèves. L'expérience montre que dans les années ultérieures du collège et du lycée, de plus en plus d'élèves peuvent s'y engager. Il reste aux professeurs de déterminer les moments dans la scolarité de leurs élèves où cette aventure peut être tentée, aventure qui favorise et justifie l'entrée dans le monde algébrique.

Je finirai cette présentation des prolongements possibles en évoquant une autre dimension qui a été pensée par certains participants à l'atelier. Il s'agit d'une prise en compte des outils de calculs (calculettes) ou de programmations qui pourrait accompagner, étayer, développer, enrichir les activités présentées. Ce point bien prometteur n'a pas été développé davantage dans l'atelier mais pourrait faire l'objet d'une nouvelle réflexion à propos de tâches à proposer aux élèves.

IV APPORTS DANS LE CADRE DE LA FORMATION DE FUTURS PROFESSEURS

IV – 1 Une situation d'homologie

Les participants de l'atelier et moi-même ayant eu l'occasion de vivre les développements que nous venons de décrire, en se projetant à la fois comme élèves, comme futurs professeurs et étant formateurs, nous avons pu élaborer ensemble progressivement un bilan des apports de cette situation dans le cadre de la formation des futurs professeurs. En fait il s'agit d'une situation d'homologie (au sens d'Houdement-Kuzniak, 1993) qui permet d'initier les futurs professeurs :

- à certains enjeux d'apprentissages pour les élèves de cycle 3 ou de collège ou chez eux-mêmes (en particulier pour les étudiants n'ayant pas un cursus d'études scientifiques)
- aux finalités de l'enseignement des mathématiques dans le cadre de la scolarité obligatoire
- à la théorie des situations et à certains phénomènes didactiques qu'elle décrit.

Passons en revue ces différentes dimensions.

IV – 2 Prise de conscience d'enjeux et d'obstacles dans les apprentissages

Nous soulignons ici principalement ce qui surprend en général les étudiants se destinant au professorat.

Il s'agit d'abord de savoirs concernant le domaine numérique et aussi le domaine algébrique :

- les principes et les sens des algorithmes utilisés pour poser une soustraction. C'est l'occasion de rappeler et distinguer les différentes techniques opératoires : la technique par emprunt, la technique des écarts constants, l'addition à trou.
- le fait que le même chiffre peut revêtir des significations différentes. Ainsi dans la technique de l'écart constant le *1* et le *3* figurants dans la colonne des unités forment treize alors que la même combinaison de signes dans la colonne des dizaines signifie *1+3* (que les maîtres demandent d'ailleurs parfois d'écrire non pas $_{1}3$ mais 3_{1}). Les étudiants mesurent ainsi la complexité des algorithmes des calculs posés :

$$\begin{array}{r} 7\ {}^{1}3 \\ -\,{}_{1}3\ \ 7 \\ \hline 3\ \ 6 \end{array}$$

- les appréhensions différentes possibles par les élèves (et par eux-mêmes) des nombres écrits, soit comme des assemblages de chiffres (*36* = un *3* et un *6*), soit comme des nombres *(trente-six)*
- la différence entre nombre et écriture des nombres à l'aide de chiffres
- les principes de la numération décimale : *36 = trente-six = 3 dizaines et 6 unités = 3×10 + 6*. Il s'agit souvent là d'une redécouverte
- de façon plus générale la coexistence en mathématiques de différents cadres (numériques, algébriques) et de différents registres d'écriture et de traitements et leurs puissances. C'est particulièrement à l'occasion des prolongements de l'activité vers des horizons algébriques que s'opèrent ces prises de conscience : par exemple lorsqu'on se permet d'écrire un nombre à deux chiffres sous la forme générale *ab* puis *10a+b*. Une question qui trouvera difficilement réponse dans un des registres d'écriture se traitera plus facilement lorsqu'on aura traduit le problème dans un

autre. Les étudiants sont ainsi initiés au geste qui consiste à choisir le registre pertinent ou aisé pour résoudre une question (Duval, 1995).

Il s'agit d'autre part d'une activité qui permet d'installer chez les étudiants des questions et aussi, comme on a pu le constater, des éléments de réponses, à propos du raisonnement en mathématique :

- qu'est ce qu'une preuve ? Y a-t-il différents niveaux acceptables de preuves ?
- par quoi peut-on être convaincu ? Différence entre conviction d'ordre probabiliste et certitude ?
- est-ce que « convaincre » est différent de « prouver » ?
- quel est le statut d'un exemple ? Peut-il être « générique » ? (repérage d'invariants dans les exemples numériques)
- y a-t-il différentes façons de prouver en mathématiques ? Y a-t-il des manières de prouver qui ne nécessitent pas forcément l'application de théorèmes établis (preuve par l'étude exhaustive des cas)
- quelle est la place des contres exemples dans les processus de raisonnement et dans le chemin vers la prise de conscience de la nécessité de dépasser une constatation pour lui donner un statut de conjecture ?

Cette activité permet d'autre part, de développer des compétences dépassant le cadre des mathématiques. Les étudiants peuvent prendre conscience que les disciplines ne sont pas isolées mais sont en relation. Dans les activités décrites, les élèves sont en effet très vite amenés à développer leur sens de l'observation par l'organisation des résultats, le repérage et le tri des informations en vue d'émettre des remarques. Nous sommes là dans le domaine du développement de compétences générales bien utiles au-delà du domaine mathématique. La mise au point de la formulation des remarques, la lecture et la compréhension du programme de calcul initial et par la suite la rédaction d'un nouveau programme remplaçant le précédant confortent la maîtrise de la langue et de ses différentes fonctions : description, programmation....

IV – 3 Prise de conscience des finalités de l'enseignement des mathématiques

La prise de conscience par les étudiants des enjeux et des obstacles dans les apprentissages est pour moi l'occasion de faire avec eux un point sur la question essentielle des finalités de l'enseignement des mathématiques dans le cadre de la scolarité obligatoire. Préalablement à toutes les activités autour de l'algorithme de Kaprekar, je demande aux étudiants d'écrire pourquoi enseigner les mathématiques. Dans une synthèse menée collectivement les propositions sont classées et je propose alors aussi comme outil de classement la proposition de G. Vergnaud (1987). Il distingue :

- la transmission d'un patrimoine scientifique (les connaissances en mathématiques en l'occurrence)
- le développement de compétences au service de la vie quotidienne, d'autres disciplines et de professions
- le développement de la conceptualisation du réel (pour ma part je parle de la pensée des individus).

On a là un repérage possible des missions de l'école : mission de transmission de l'héritage de l'humanité, mission d'insertion sociale, mission de développement des

individus. Le fait que les étudiants aient pratiqué les activités liées à l'algorithme de Kaprekar me permet de leur demander de renseigner cette classification en se rapportant à ces activités. Ils ont ainsi l'occasion de formuler plus précisément des compétences et des savoirs en jeu dans cette situation et de les relier aux principales finalités. Voici quelques exemples : la notion d'écriture décimale des nombres apparaît comme un lègue des humains à transmettre ; l'accès à la notion de conjecture à prouver fait partie du développement de la pensée ; la capacité de concevoir et de formuler des programmes se rapporte aussi au développement de la maîtrise de la langue.

Avec ce travail sur les finalités, c'est aussi l'occasion de prendre conscience de la difficulté d'enseigner les mathématiques qui est analysée par Vergnaud, à savoir, comment relier les trois pôles ? Par exemple, à quel moment du développement de la pensée des élèves peut-on les initier à la numération, aux outils algébriques, à la géométrie déductive ?

IV – 4 Initiation à la théorie des situations didactiques.

Que ce soit pour des élèves ou des étudiants, l'algorithme de Kaprekar crée très rapidement un corpus très riche de résultats qui permet d'émettre des observations qui sont formulées, discutées et améliorées au sein de la classe ou du groupe. Ces observations peuvent se transformer en conjectures. Ces conjectures donnent lieu à des activités de validation. Les étudiants ont ainsi l'occasion de vivre et de pointer des situations d'actions, de formulations et de validations proposées par Guy Brousseau (1972) pour décrire les processus d'acquisition de connaissance en mathématique.

Dans ce cadre, une première sensibilisation à la notion de contrat didactique peut être réalisée à partir de l'analyse de certains évènements vécus dans le groupe ou rapportés par le formateur. Ainsi à partir de la tâche initiale qui est d'effectuer le programme de calcul proposé les attitudes des étudiants sont très différentes. Certains arrêtent leur activité dès le premier exemple lorsqu'ils tombent sur 495. D'autres continuent à prospecter et essayent pour d'autres nombres. Il en est de même pour les élèves dont certains s'arrêtent immédiatement, très satisfaits, parce qu'ils ont choisi au départ un nombre à trois chiffres identiques et s'écrient *: « j'ai fini le travail ! »*. L'algorithme de Kaprekar est effectivement un support intéressant pour inciter à poursuivre une recherche. Mais cela ne va pas de soi pour tous les individus. On peut aussi attirer l'attention sur des attitudes très différentes lorsque les élèves ou les étudiants tombent sur la question qui se pose à propos de l'algorithme appliqué aux nombres à deux chiffres : *« Que faire lorsqu'on tombe sur 9 ? »*. Certains en font uniquement une question de règle à préciser et d'autres un champ de prospection. A travers ces différents exemples vécus ou rapportés, il est ainsi possible de sensibiliser les étudiants à la différence entre les enjeux qu'attribuent les apprenants aux tâches qui leur sont proposées.

Il est aussi possible d'orienter à partir de là les étudiants qui en auraient le goût et qui plus tard pourraient rejoindre la cohorte des chercheurs en didactique des mathématiques, vers un approfondissement de leurs connaissances des travaux de G. Brousseau et en particulier vers son introduction à la théorie des situations didactiques où il s'appuie sur la situation dite de la *« Course à 20 »* pour analyser et modéliser les situations d'enseignement à partir des échanges entre les élèves, le professeur et le milieu (Brousseau, 1998, pp25-43).

V – CONCLUSION

Entre les extrêmes d'une utilisation assez réduite qui consisterait uniquement à en exhiber la curiosité et qui ne serait que l'occasion de s'exercer à la technique de la soustraction posée d'une part et la question non triviale du pourquoi de la stabilisation des suites d'autre part, l'algorithme de Kaprekar se révèle bien riche de potentialités d'apprentissages et de formation pour les futurs professeurs d'école. L'analyse de ces potentialités dans le cadre de l'atelier a permis d'en tracer les grandes lignes de force et aussi d'entrevoir quelques prolongements ou variantes qui nécessiteraient des réflexions et expérimentations supplémentaires : utilisation des outils de programmations ou de calculs et extension des observations aux bases autres que la base 10 par exemple.

De façon plus distanciée et pour se situer dans le thème du colloque, on peut constater qu'on a là un matériau qui permet de développer des démarches d'observations, d'expérimentations puis de validations mathématiques tant pour des élèves pour développer leurs apprentissages que pour la formation des professeurs. On peut remarquer qu'il ne s'agit pas ici d'une situation qui s'appuie sur « le réel » ou qui part d'un environnement quotidien extra scolaire des élèves : on est d'emblée sur le terrain mathématique, tout comme par exemple pour la « Course à 20 » (Brousseau, 1998). Mais on peut aussi remarquer que ce support permet de développer des compétences utiles en dehors de la discipline mathématique. Ces remarques sont donc une invitation à poursuivre et à compléter la réflexion initiée par le colloque : « Quelles mathématiques à l'Ecole ? ».

RÉFÉRENCES BIBLIOGRAPHIQUES

BROUSSEAU G. (1972) Processus de mathématisation, *Bulletin APMEP* , **282**, 418-429.

BROUSSEAU G. (1998) *Théorie des situations didactiques*, La Pensée Sauvage, éditions, Grenoble.

CHAUVAT G. (1980) Arithmétique élémentaire en classe de cinquième, *Plot*, **12**, 10-13.

DE BLOCK-DOCK C. (1994) Modalités de la pensée mathématique d'élèves de douze ans devant des problèmes de pavage, *Educational Studies in Mathematics*, **27**, 165-189.

DUVAL R. (1995) *Sémiosis et pensée humaine*, Peter Lang, Bern.

HOUDEMENT C., KUZNIAK A. (1996) Autour des stratégies utilisées pour former les maîtres du premier degré en mathématiques, *Recherches en Didactique des Mathématiques*, **16/3**, 289-322.

IREM DE STRASBOURG (1986) *Mathématiques 6ème*, Collection collèges dirigée par F.Mollet-Petit, Istra, Strasbourg.

VERGNAUD G. (1987) Réflexions sur les finalités de l'Enseignement des Mathématiques, *Gazette des mathématiciens* , **32**, 54-61.