Analyse et étude des processus markoviens décisionnels

Christophe Nivot

université de BORDEAUX

Inria INVENTEURS DU MONDE NUMÉRIQUE

N° d'ordre :

THÈSE

présentée pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE BORDEAUX

par

Christophe NIVOT

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET D'INFORMATIQUE

Spécialité : Mathématiques appliquées et calcul scientifique

Analyse et étude des processus markoviens décisionnels

Sous la direction de **François DUFOUR** et de **Benoîte DE SAPORTA**

Thèse soutenue le jeudi 19 mai 2016

JURY

Jérôme SARACCO	Professeur	INP Bordeaux	Président
Damien BÉRARD-BERGERY	Ingénieur	Airbus DS	Invité
Benoîte DE SAPORTA	Professeuse	U. de Montpellier	Directrice de thèse
François DUFOUR	Professeur	INP Bordeaux	Directeur de thèse
Charles ELEGBEDE	Docteur	Airbus DS	Examinateur
Antoine GRALL	Professeur	U.T. de Troyes	Rapporteur
Nikolaos LIMNIOS	Professeur	U.T. de Compiègne	Rapporteur

Titre : analyse et étude des processus markoviens décisionnels

Résumé : nous explorons l'étendue du champ applicatif des processus markoviens décisionnels au travers de deux problématiques. La première, de nature industrielle, propose l'étude numérique de l'optimisation d'un processus d'intégration lanceur en collaboration avec Airbus DS. Il s'agit d'un cas particulier des problèmes de gestion d'inventaire dans lequel un calendrier de tirs joue un rôle central. La modélisation adoptée entraîne l'impossibilité d'appliquer les procédures d'optimisation classiques liées au formalisme des processus markoviens décisionnels. Nous étudions alors des algorithmes basés sur des simulations qui rendent des stratégies optimales non triviales et qui sont utilisables dans la pratique. La deuxième problématique, de nature théorique, se concentre sur les questions d'arrêt optimal partiellement observables. Nous proposons une méthode d'approximation par quantification de ces problèmes lorsque les espaces d'états sont quelconques. Nous étudions la convergence de la valeur optimale approchée vers la valeur optimale réelle ainsi que sa vitesse. Nous appliquons notre méthode à un exemple numérique.

Mots-clés : Processus markoviens décisionnels, optimisation, modélisation, méthodes probabilistes, simulation, arrêt optimal, quantification

---

Title : a study of Markov decision processes

Abstract : we investigate the potential of the Markov decision processes theory through two applications. The first part of this work is dedicated to the numerical study of an industrial launcher integration process in co-operation with Airbus DS. It is a particular case of inventory-control problems where a launch calendar has a key role. The model we propose implies that standard optimization techniques cannot be used. We then investigate two simulation-based algorithms. They return non trivial optimal policies which can be applied in actual practice. The second part of this work deals with the study of partially observable optimal stopping problems. We propose an approximation method using optimal quantization for problems with general state space. We study the convergence of the approximated optimal value towards the real optimal value. The convergence rate is also under study. We apply our method to a numerical example.

Keywords : Markov decision processes, optimization, modeling, probabilistic methods, simulation, optimal stopping, quantization

---

Unité de recherche :
Inria Bordeaux Sud-Ouest - 200818243Z
200, avenue de la Vieille Tour,
33405 Talence

Remerciements

En tout premier lieu, je souhaite remercier les rapporteurs de ma thèse, Antoine Grall et Nikolaos Limnios, pour avoir gentiment accepté de prendre le temps de lire cette thèse. Leurs précieuses remarques et observations m'ont été d'une grande utilité. En second lieu, je remercie Jérôme Saracco d'avoir accepté de présider mon jury de thèse. Enfin, je remercie tous les membres du jury pour avoir consenti à écouter mon exposé de soutenance et à évaluer mon travail de trois années.

J'adresse bien sûr des remerciements tout particuliers à mes directeurs de thèse, François Dufour et Benoîte de Saporta. Votre accueil chaleureux au début de mon stage de Master 2 a conditionné ma motivation et mon envie de donner le meilleur de moi-même pour cette thèse. Vous avez su m'encourager, me motiver, me faire me surpasser. Votre oreille attentive dans mes moments de doute a été essentielle. Je vous remercie pour votre présence, votre gentillesse, votre encadrement et votre soutien.

Je voudrais aussi remercier Huilong Zhang pour son aide précieuse sur les questions informatiques et numériques diverses et variées que nous avons rencontrées.

Je remercie également mes deux collaborateurs chez Airbus, Charles Elegbede et Damien Bérard-Bergery pour leur gentillesse et leur professionnalisme.

Ces trois années de thèse auraient été bien tristes sans vous, mes chers collègues de bureau. Alizée, nos discussions, nos fous-rires sur tout et sur rien ont apporté du soleil dans mes journées ordinaires ! Adrien, je te remercie vraiment de m'avoir donné l'astuce géniale qui a fait marcher mon simulateur en C !

À ce propos, je veux adresser un énorme merci à mon frère Stéphane, d'une part pour m'avoir dit que je codais comme un pied (*sic*), d'autre part pour avoir mis tes compétences informatiques à contribution pour m'apprendre le langage C.

J'ai une pensée très particulière pour ma famille et mes amis, et plus particulièrement encore

pour mes parents. Ces trois années passées loin d'eux n'ont fait que confirmer l'importance qu'ils ont pour moi. Sans leur aide et leur soutien, je ne serais jamais arrivé jusqu'ici. Je ne pourrai jamais assez les remercier de tout ce qu'ils ont fait afin que je sois la personne que je suis aujourd'hui. Rémy, ta bonne humeur, ton soutien inconditionnel et ton engagement m'ont été vitaux durant ces trois ans. Je n'aurais jamais trouvé la force de partir si tu n'avais pas été là.

Enfin, je remercie les infusions Éléphant et les chocolats Lindt, compagnons d'infortune dans les heures les plus difficiles et amis fidèles dans les moments plus légers.

« *Never lock yourself.* »

Table des matières

Table des figures

Liste des tableaux

Notations et symboles

$\#E$	Cardinal de l'ensemble E
$\mathbb{1}_E$	Fonction indicatrice de l'ensemble E
$\mathcal{B}(E)$	Tribu des boréliens sur E
δ_x	Masse de Dirac en x
ℓ_d	Mesure de Lebesgue sur $\mathbb{R}^d$
$[\![a;b]\!]_2$	Ensemble $\left\{a+\frac{k}{2} \;\middle|\; k\in\mathbb{N} \text{ et } a+\frac{k}{2}\leq b\right\}$
$[\![m;n]\!]$	Ensemble $\{m;m+1;\ldots;n\}$
$\lceil x\rceil$	Partie entière supérieure de x
$\lfloor x\rfloor$	Partie entière de x
$[\![n;+\infty[\![$	Ensemble des entiers supérieurs ou égaux à n
$\mathrm{L}^p(\Omega,\mathcal{F},\mathbb{P})$	Ensemble des variables aléatoires X définies sur $(\Omega,\mathcal{F},\mathbb{P})$ telles que $\mathbb{E}[\lvert X\rvert^p]<+\infty$
$\mathbb{L}(E)$	Ensemble des fonctions lipschitziennes bornées sur E
$\mathcal{N}(\mu,\sigma^2)$	Loi normale d'espérance μ et de variance σ^2
$\mathcal{P}(E)$	Ensemble des mesures de probabilité sur $(E,\mathcal{B}(E))$
$\mathcal{P}(X\mid Y)$	Ensemble des noyaux stochastiques sur X sachant Y
$\mathcal{U}(E)$	Loi uniforme sur E
$\mathrm{adh}(A)$	Adhérence de A
$\mathrm{conv}(E)$	Enveloppe convexe fermée de E
$\mathrm{int}(A)$	Intérieur de A

$\mathrm{supp}(\mu)$	Support de la mesure μ
$\mathbb{N}$	Ensemble des entiers naturels
$\mathbb{N}^*$	Ensemble des entiers naturels strictement positifs
$\perp$	Orthogonalité
$\mathbb{R}$	Ensemble des nombres réels
$\mathbb{R}_+$	Intervalle $[0; +\infty[$
$\square$	Fin de preuve
$\lvert x\rvert$	Norme euclidienne usuelle d'un élément $x \in \mathbb{R}^d$
${}^tM$	Transposée de la matrice M
$a \wedge b$	Minimum de a et de b
x_+	Partie positive du réel x

Introduction

Quel itinéraire dois-je prendre pour parvenir à ma destination ? Dois-je garer ma voiture à cette place ou tenter d'en trouver une plus près du magasin ? Dois-je faire réparer mon lave-vaisselle maintenant ou attendre encore ? Quelle somme dois-je investir dans mon placement risqué ? Autant de questions auxquelles chacun d'entre nous a pu être confronté. En effet, dans notre vie quotidienne, nous devons souvent faire face à des situations où plusieurs options s'offrent à nous, et où il nous incombe d'en choisir une et une seule. Nous nous retrouvons alors face au dilemme de la « bonne » décision : quelles sont les options convenables, et éventuellement, quelle est la meilleure d'entre elles ? La réponse à cette question ne va pas de soi.

Nous nous fixons certains critères pour décider (longueur du trajet ou coût financier par exemple), mais chaque décision ayant potentiellement des conséquences à court et à long terme, les paramètres à prendre en compte peuvent être très nombreux. Ainsi, on peut ne pas mesurer toute l'importance de notre choix dans l'immédiateté de l'instant. L'information sur laquelle se baser pour décider peut être complètement ou partiellement accessible, en particulier lorsque plusieurs personnes doivent décider indépendamment les unes des autres. Choisir fait alors appel à notre intelligence, à notre rationnalité, à notre intuition, à notre expérience et finalement à notre personnalité toute entière. On touche du doigt l'immense complexité de l'étude du choix, qui relève tout à la fois des sciences fondamentales, des sciences humaines, et du plus vieux des rêves de l'être humain, celui de connaître l'avenir.

Dans la démarche de formaliser et de modéliser ces situations dont l'issue dépend d'une décision humaine sont nées la recherche opérationnelle et les sciences de la décision. Disciplines aux facettes multiples, ce sont dans les mathématiques qu'elles puisent leur source. Dès le XVII^{e} siècle, les problèmes de décision dans l'incertain occupent des mathématiciens comme Pascal. Mais ce n'est qu'au début du XX^{e} siècle qu'on commence à étudier l'archétype de ce genre de problème : la gestion de stocks. Les mathématiques, notamment grâce au développement des Probabilités et de la Théorie des Jeux, se sont emparées de la question et ont essayé d'élaborer un formalisme pour résoudre le problème de la « bonne » décision. Comment représenter l'évolution, éventuellement soumise au hasard, d'un système influencé par une décision extérieure ?

Dès les années 1950 émerge une classe de processus stochastiques contrôlés : les processus markoviens décisionnels (MDP en abrégé, pour *Markov Decision Processes*). Il s'agit de suites de variables aléatoires (les états du système) dont la transition d'un état à un autre dépend de l'état de départ et d'une décision extérieure. Une quantité, coût ou récompense selon l'interprétation, mesure l'optimalité du choix opéré. Des critères doivent alors évaluer la qualité de toutes les décisions prises, dont l'ensemble constitue ce qu'on appelle une stratégie. Il existe de nombreux types de MDP : complètement ou partiellement observés, à temps discret ou à temps continu, à horizon fini ou infini... Un grand nombre de situations d'optimisation décisionnelle peuvent trouver un MDP adapté. Ainsi, on retrouve ce formalisme mathématique dans de nombreux domaines, notamment la finance, l'industrie ou la gestion. Même des situations de la vie quotidienne peuvent être modélisées par un MDP : par exemple, sous réserve d'une étude statistique préalable, on peut appliquer les MDP à la recherche d'une place dans un parking.

Naturellement, lorsqu'on utilise un tel formalisme, la question de la modélisation point toujours. Qu'est-ce qu'un état, c'est-à-dire quelles seront les variables qui le doivent le composer ? Seront-elles continues, discrètes, ou même booléennes ? Comment ces variables vont interagir entre elles ? Comment écrire la dynamique du système ? Quel coût va-t-on utiliser ? Ces questions, d'une importance capitale, peuvent se trouver parmi les plus délicates, en particulier dans les applications issues de problèmes réels. Toutefois, une fois la modélisation choisie, la phase d'optimisation dont elle dépend peut également se révéler ardue. Les applications des MDP ont donné l'impulsion pour le développement de procédures d'optimisation et ont permis une littérature théorique et appliquée abondante sur le sujet. Ainsi, la réponse mathématique apportée à des problèmes concrets toujours plus nombreux et plus complexes implique d'élaborer des modèles de plus en plus précis.

Le travail de thèse que nous allons présenter ici s'inscrit dans un dialogue permanent entre industrie et mathématiques. Il est issu d'une étroite collaboration avec le groupe européen Airbus, présent dans le secteur aérodynamique et spatial civil et militaire. Fondé en 2000 sous le nom d'EADS (European Aeronautic Defence and Space company), ses activités les plus importantes sont la construction d'avions de ligne, d'hélicoptères et d'avions militaires. Mais le groupe est également engagé dans les lanceurs spatiaux et les satellites artificiels.

Dans ce contexte se place le problème industriel qui a motivé cette thèse. L'intégration de lanceurs spatiaux nécessite une organisation rigoureuse de la chaîne de production pour parvenir à assurer leur qualité, à garantir la sécurité des équipements et à honorer la demande des clients à la date prévue. Notamment, il faut décider des infrastructures à mettre en place, de leur position dans la chaîne mais aussi de leur situation géographique. Il faut savoir quand effectuer des maintenances pour ne pas perturber le calendrier établi. Il faut imprimer à la chaîne une cadence de production adaptée. Tous ces sujets sont au centre des préoccupations d'Airbus.

Les discussions ont commencé avec la présentation d'une chaîne de montage, celle décrite dans ce travail de thèse. L'objectif d'Airbus était de trouver des taux de production d'entrée optimaux afin que les coûts générés soient les plus bas possibles et que le calendrier soit respecté. En effet, il s'agissait de trouver le juste équilibre entre une cadence rapide qui permet d'effectuer

les tirs en temps voulu mais qui engendre des composantes inutilisées et une cadence trop lente qui limite la formation de stocks mais qui peut générer des retards. Le problème d'optimisation a été étendu à une partie de l'architecture toujours en cours d'étude. En l'occurrence, la capacité d'un des bâtiments de stockage n'est pas fixée. Donc, un travail bibliographique préliminaire pour s'assurer que les MDP permettaient de remplir les objectifs demandés a été nécessaire.

Toutefois, Airbus avait également en vue l'industrialisation du procédé. Les méthodes de simulation et d'optimisation devaient donc être suffisamment malléables pour être utilisables par des industriels et adaptables à d'autres situations d'optimisation décisionnelle séquentielle. Une importante partie du travail a consisté à interagir avec les industriels pour à la fois fixer le modèle, en comprendre toutes les subtilités, expliciter notre démarche et notre méthode, et préparer le terrain pour des modèles plus précis et donc plus pointus.

Le problème posé par Airbus, moteur majeur de cette thèse, est prétexte à l'analyse des processus décisionnels markoviens. Plus précisément, le but de notre travail est d'étudier deux types d'applications des MDP. D'une part, nous nous penchons sur le problème industriel d'Airbus, pour lequel nous nous proposons de développer un modèle mathématique et une méthode d'optimisation qui rend une stratégie optimale de cadencement de la production des sous-assemblages sous une forme utilisable en pratique.

D'autre part, nous travaillons sur une application plus théorique : les problèmes d'arrêt optimal. Il s'agit d'une classe de problèmes qui consistent à arrêter (ou plus généralement effectuer une action particulière) un système au moment opportun de sorte à optimiser une fonction d'utilité. La portée de l'arrêt optimal est vaste, notamment en finance et dans l'industrie. Dans cette thèse, nous en étudions un cas particulier : l'arrêt optimal partiellement observé. Pour décider de la décision à prendre, l'agent extérieur n'a accès qu'à une partie seulement des informations sur le système. Les techniques de résolution dans ce cas nécessitent un effort supplémentaire par rapport aux méthodes classiques. Généralement étudié lorsque les espaces d'états sont finis, nous nous proposons de développer une méthode d'approximation par quantification de problèmes d'arrêt optimal partiellement observables lorsqu'ils sont quelconques. Nous allons l'appliquer à un petit exemple.

L'architecture de cette thèse est donc organisée autour de trois grands axes. Le premier, contenu dans la partie I, propose une revue introductive des techniques classiques d'optimisation des MDP développées dans la littérature à partir des années 1970. Le chapitre 1 reprend le formalisme des MDP, précise les notations et le vocabulaire. Il est suivi par le chapitre 2 qui présente les algorithmes et méthodes standard d'optimisation ainsi que les résultats théoriques qui les justifient. Il y sera question de programmation dynamique, de programmation linéaire, et de MDP partiellement observables. Le deuxième axe, correspondant à la partie II, est consacré au travail fourni dans le cadre de la collaboration avec Airbus. Nous y étudions le problème industriel posé et expliqué dans le chapitre 3. Nous discutons de sa modélisation, de sa simulation et des difficultés induites dans le chapitre 4. S'ensuit le chapitre 5 qui décrit les algorithmes particuliers d'optimisation sur lesquels nous nous sommes focalisés pour répondre au problème. Nous y éclaircissons certains points concernant leur implémentation, notamment du point de vue

numérique. Nos résultats d'optimisation sont repris dans le chapitre 6, dans lequel nous comparons les performances des algorithmes sélectionnés, tentons de répondre au problème d'Airbus et discutons une utilisation concrète de notre solution d'optimisation dans le cadre d'un problème de robustesse. Le troisième axe, auquel la partie III fait écho, concentre notre étude théorique des problèmes d'arrêt optimal partiellement observables généraux. Nous commençons par une revue de la notion de quantification dans le chapitre 7 où nous expliquons son principe, où nous donnons les principaux résulats qui soutiennent cette théorie et où nous détaillons les procédures usuelles. Dans le chapitre 8, nous utilisons la quantification pour développer une méthode d'approximation de problèmes d'arrêt optimal partiellement observables. Très détaillé, ce chapitre expose les principaux résultats obtenus. Enfin, nous reprendrons les principales idées contenues dans cette thèse dans une conclusion générale. Nous y donnerons également quelques pistes pour un travail ultérieur.

Première partie

Processus markoviens décisionnels à temps discret

1 Formalisme des processus markoviens décisionnels

Les processus markoviens décisionnels sont une classe de processus stochastiques contrôlés qui modélisent des situations de prise de décision séquentielle dans l'incertain. Ils sont nés dans le giron de la recherche opérationnelle dans les années 1950. Leur étude a connu un essor formidable à partir des années 1970 et a produit une littérature abondante à la fois théorique et appliquée. Au fil des années, ils ont prouvé leur utilité dans des domaines très variés, comme la finance, l'ingénérie, l'écologie, la gestion de ressources matérielles ou humaines par exemple. Leurs applications ont conduit à des progrès notables sur ce formalisme. À ses débuts, la recherche sur les MDP était concentrée sur des modèles où

— soit l'espace des états est fini ou dénombrable,

— soit la fonction de coût est bornée,

— soit l'ensemble des décisions est compact.

L'évolution de la puissance de calcul a permis récemment d'avancer sur l'étude de cas généraux et sur la création de nouveaux modèles, comme les processus markoviens décisionnels partiellement observables (POMDP en abrégé, pour *Partially Observable Markov Decision Processes*) qui concernent des situations où un contrôleur n'a pas accès à toutes les informations sur l'état courant du système étudié.

Dans ce chapitre, nous proposons de reprendre la base du formalisme des MDP et des POMDP dans le cas général en se basant sur [HLL96, HLL99] et [BR11]. Sommairement, considérant un processus, les MDP modélisent une suite d'états dont la transition de l'un à l'autre est aléatoire et dépend de deux éléments : l'état de départ et la décision prise par un

agent extérieur. Ces deux éléments génèrent un coût, appelé coût par transition. Au cours de l'évolution du système, l'agent suit ce qu'on appelle une stratégie, c'est-à-dire une règle qui donne l'action à faire selon l'état dans lequel le système se trouve. Les problèmes associés aux MDP consistent alors à trouver la meilleure stratégie, c'est-à-dire celle qui minimise le coût en moyenne. Il existe différents critères de coût, qui correspondent à différents types de problème. Nous exposons ici les deux plus courants.

Dans un premier temps, nous suivrons le cadre développé dans [HLL96]. La section 1.1 regroupe toutes les notions nécessaires à la définition d'un MDP. La section 1.2 précise la notion de stratégie et la met en lien avec la propriété markovienne du processus. Puis nous décrirons les critères de coûts usuels dans la section 1.3. La dernière section, dont le contenu reprendra le cheminement de [BR11], développe le formalisme des POMDP.

1.1 Modèles markoviens décisionnels

1.1.1 Définitions préliminaires

Dans le cas général, l'espace des états ne doit pas être tout à fait quelconque. En l'occurrence, il doit appartenir à la classe des espaces de Borel dont nous donnons la définition.

Définition 1.1.1.1. On appelle espace de Borel un borélien d'un espace métrique complet et séparable.

Étant donné un espace de Borel X, on notera $\mathcal{B}(X)$ la tribu borélienne associée. Par la suite, lorsque l'on parlera d'ensembles ou de fonctions mesurables, on sous-entendra « mesurables par rapport à la tribu borélienne ».

La notion suivante de noyau stochastique est centrale dans le formalisme des MDP. On la retrouvera souvent, notamment pour la loi de transition.

Définition 1.1.1.2. Soient X et Y deux espaces de Borel. Un noyau stochastique sur X sachant Y est une fonction $Q(\cdot \mid \cdot)$ telle que :

— pour tout $y \in Y$ fixé, $B \mapsto Q(B \mid y)$ est une mesure de probabilité sur X ;

— pour tout $B \in \mathcal{B}(X)$ fixé, $y \mapsto Q(B \mid y)$ est une fonction mesurable sur Y.

L'ensemble des noyaux stochastiques sur X sachant Y est noté $\mathcal{P}(X \mid Y)$.

Maintenant que nous avons défini ces deux objets, nous pouvons expliquer ce qu'est un modèle markovien décisionnel.

1.1.2 Modèles et processus markoviens décisionnels.

Introduisons la définition essentielle pour notre propos.

Définition 1.1.2.1. Un modèle markovien décisionnel est la donnée d'un sextuplet

$$(\mathbb{X}, \mathbb{A}, \{\mathbb{A}(x) \mid x \in \mathbb{X}\}, Q, c, \nu)$$

comprenant :

— un espace de Borel $\mathbb{X}$ appelé *espace d'états* ;

— un espace de Borel $\mathbb{A}$ appelé *espace d'actions* ;

— un ensemble $\{\mathbb{A}(x) \mid x \in \mathbb{X}\}$ de sous-ensembles mesurables non vides de $\mathbb{A}$, où $\mathbb{A}(x)$ est *l'ensemble des actions réalisables lorsque le système est dans l'état* $x \in \mathbb{X}$, tel que $\mathbb{K} = \{(x, a) \mid x \in \mathbb{X}, a \in \mathbb{A}(x)\}$ soit un sous-ensemble mesurable de $\mathbb{X} \times \mathbb{A}$;

— un noyau stochastique Q sur $\mathbb{X}$ sachant $\mathbb{K}$ appelé *loi de transition* ;

— une fonction mesurable $c : \mathbb{K} \to \mathbb{R}$ appelée *fonction de coût par transition* ;

— une loi de probabilité initiale ν sur $\mathbb{X}$.

Remarque. À la place d'une fonction de coût, on peut tout à fait considérer une fonction de récompense. La théorie qui en découle est presque la même : la seule différence réside dans le fait que nous calculerons alors des maxima plutôt que des minima. Nous aurons l'occasion, dans les chapitres qui suivent, de traiter des exemples de l'une ou l'autre situation.

Ce formalisme correspond donc à celui d'un processus stochastique contrôlé, c'est-à-dire que la transition dépend à la fois de l'état et de la décision prise. La figure 1.1 schématise ce qui se passe lors d'une transition.

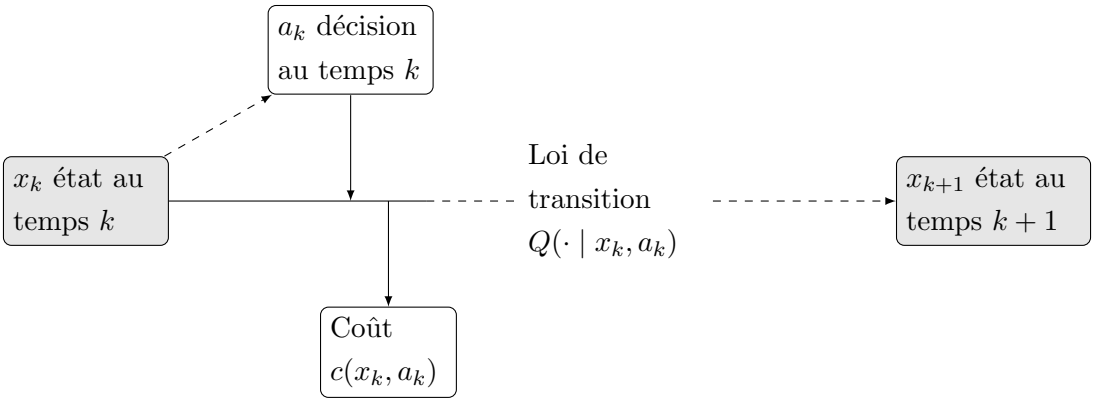

FIGURE 1.1 – Principe d'un processus markovien décisionnel

Dans la suite, nous nous donnons un modèle décisionnel markovien

$$(\mathbb{X}, \mathbb{A}, \{\mathbb{A}(x) \mid x \in \mathbb{X}\}, Q, c, \nu).$$

Afin de s'assurer que les notions que l'on s'apprête à aborder soient bien définies, on fait l'hypothèse suivante.

Hypothèse(s). *Il existe une fonction* $f : \mathbb{X} \to \mathbb{A}$ *telle que* $f(x) \in \mathbb{A}(x)$ *pour tout* $x \in \mathbb{X}$.

Maintenant, pour tout $k \in \mathbb{N}$, on définit l'ensemble H_k des *histoires admissibles* jusqu'au temps k par

$$\begin{cases} H_0 = \mathbb{X} \\ H_k = \mathbb{K}^k \times \mathbb{X} \end{cases}. \tag{1.1}$$

Un élément h_k de H_k est appelé *k-histoire admissible* et s'écrit $h_k = (x_0, a_0, \ldots, x_{k-1}, a_{k-1}, x_k)$ où pour tout $0 \leq l \leq k-1$, $(x_l, a_l) \in \mathbb{K}$.

L'optimisation décisionnelle va porter sur ce qu'on appelle les stratégies. Il en existe différentes sortes, que nous distinguerons dans la section suivante. La définition que nous donnons ici est la plus générale.

Définition 1.1.2.2. Une stratégie est une suite $\pi = (\pi_k)_{k\in\mathbb{N}}$ de noyaux stochastiques sur $\mathbb{A}$ sachant H_k tels que $\pi_k(\mathbb{A}(x_k) \mid h_k) = 1$ pour tout $h_k = (x_0, a_0, \ldots, x_k) \in H_k$, pour tout k dans $\mathbb{N}$. L'ensemble de toutes les stratégies est noté Π.

Intuitivement, une stratégie peut se voir comme une suite (a_k) de variables aléatoires à valeurs dans $\mathbb{A}$ (c'est-à-dire d'actions) telle que, pour toute histoire h_k, la loi de a_k est la mesure de probabilité $\pi_k(\cdot \mid h_k)$ concentrée sur $\mathbb{A}(x_k)$.

Construction canonique des MDP

Jusqu'à présent, nous n'avons évoqué que le cadre des modèles markoviens décisionnels. Or, le calcul du coût (ou de la récompense) nécessite naturellement de connaître la succession des états dans lequel le système s'est trouvé. Il faut alors définir convenablement un processus aléatoire sous-jacent au modèle. Une construction dite canonique est donnée ici.

On note $\Omega = (\mathbb{X} \times \mathbb{A})^{\mathbb{N}}$ l'ensemble dont les éléments sont les suites $\omega = (x_0, a_0, x_1, a_1, \ldots)$, avec $x_k \in \mathbb{X}$ et $a_k \in \mathbb{A}$ pour tout k (remarquons que Ω contient l'ensemble $\mathbb{K}^{\mathbb{N}}$ des histoires admissibles). On note $\mathcal{F}$ la tribu produit associée. On définit deux suites de variables aléatoires (X_k) et (A_k) sur Ω de la manière suivante : pour tout $\omega \in \Omega$ et tout $k \in \mathbb{N}$, on pose

$$X_k(\omega) = x_k \text{ et } A_k(\omega) = a_k. \tag{1.2}$$

Nous avons défini des variables aléatoires sur un espace mesurable, mais il reste à choisir une mesure de probabilité qui respecte le principe de la transition du MDP. La construction d'une telle mesure sur $(\Omega, \mathcal{F})$ s'appuie sur la donnée d'une stratégie π et de la distribution initiale ν. Le théorème de Ionescu-Tulcea (cf [HLL96], annexe C, théorème C.10) assure alors l'existence d'une unique probabilité $\mathbb{P}^\pi_\nu$ sur $(\Omega, \mathcal{F})$ telle que, pour tous $B \in \mathcal{B}(\mathbb{X})$ et $C \in \mathcal{B}(\mathbb{A})$,

$$\mathbb{P}^\pi_\nu \{X_0 \in B\} = \nu(B), \tag{1.3}$$

$$\mathbb{P}^\pi_\nu \{A_k \in C \mid X_0, A_0, \ldots, X_k\} = \pi_t(C \mid X_0, A_0, \ldots, X_k), \tag{1.4}$$

$$\mathbb{P}^\pi_\nu \{X_{k+1} \in B \mid X_0, A_0, \ldots, X_k, A_k\} = Q(B \mid X_k, A_k). \tag{1.5}$$

Dans la suite, on notera $\mathbb{E}_\nu^\pi$ l'espérance associée à $\mathbb{P}_\nu^\pi$. Formellement, la définition d'un MDP est donc la suivante.

Définition 1.1.2.3. Un processus markovien décisionnel à temps discret est le processus stochastique $(X_k)_{k\geq 0}$ défini sur l'espace probabilisé $(\Omega, \mathcal{F}, \mathbb{P}_\nu^\pi)$.

Remarques.

— Le MDP (X_k) ainsi défini dépend fortement de la stratégie et de la mesure de probabilité initiale choisies.

— L'équation (1.5) ressemble à une condition de Markov. En revanche, (X_k) n'est pas, en général, un processus de Markov au sens usuel. En effet, la transition ne dépend pas que de l'état de départ, elle dépend aussi de l'action. Cependant, dans la section suivante, on étudiera une condition sur la stratégie π pour que (X_k) soit un processus de Markov.

— Le modèle $\mathcal{M}$ tel qu'on l'a défini est stationnaire, c'est-à-dire que ses composantes ne dépendent pas du temps. Il est possible de définir des modèles non-stationnaires. Mais au prix d'un changement d'espace d'états, on peut se ramener au cas stationnaire.

Un exemple de MDP

On considère une population de poissons dont la pêche est ouverte toute l'année. Toutefois, elle est limitée à une certaine proportion de la population totale de poissons. On souhaite modéliser l'évolution de cette population.

Les MDP sont adaptés à un tel problème dans la mesure où une composante de décision va intervenir : le volume de poissons à laisser pour la reproduction. On suppose que la croissance de la population de poissons suit le modèle de Ricker (cf. équation (1.6)).

La pêche intervenant toute l'année, on peut prendre un pas de temps d'une saison par exemple. Chaque état x représente la variable d'intérêt, qui est ici la population de poissons. Donc l'espace d'états $\mathbb{X}$ sera l'ensemble des tailles de population possibles, donc $\mathbb{X} = \mathbb{R}_+$.

La décision intervient au niveau du volume de poissons que l'on souhaite laisser pour se reproduire. L'espace d'actions sera donc $\mathbb{A} = \mathbb{R}_+$. Le modèle de Ricker est donné par la relation suivante :

$$x_{k+1} = \theta_1 a_k \exp(-\theta_2 a_k + \xi_k), \tag{1.6}$$

où θ_1 et θ_2 sont des constantes connues et (ξ_k) est une suite de variables aléatoires indépendantes et identiquement distribuées à valeurs réelles dont la loi commune est connue.

Pour autant, il est évident que le nombre de poissons restant après la pêche ne peut excéder la population totale de départ. Donc pour tout état x, les actions réalisables à l'état x forment l'intervalle $\mathbb{A}(x) = [0; x]$.

L'évolution de la population de poissons est donnée par une relation de la forme $x_{k+1} = F(x_k, a_k, \xi_k)$. Si on appelle μ la loi commune des ξ_k, la loi de transition Q est donnée, pour tout

$B \in \mathcal{B}(\mathbb{X})$, $x \in \mathbb{X}$ et $a \in \mathbb{A}$, par

$$\begin{aligned} Q(B \mid x, a) &= \mu(\{y \in \mathbb{R} \mid \theta_1 a \exp(-\theta_2 a + y) \in B\}) \\ &= \int_{\mathbb{R}} \mathbb{1}_B(\theta_1 a \exp(-\theta_2 a + y))\mu(\mathrm{d}y). \end{aligned}$$

La fonction de coût dépend du point de vue que l'on adopte sur la situation que l'on modélise. Par exemple, si on se place dans le rôle du pêcheur, on peut considérer une fonction de récompense $r(x, a) = x - a$. Cette fonction peut être vue comme un « coût » pour un défenseur de la cause animale. Il s'agit de la même fonction, mais qui a une interprétation différente.

Enfin, la loi initiale du processus peut être une masse de Dirac en la population initiale x_0 de poissons. On a donc un modèle markovien décisionnel $(\mathbb{R}_+, \mathbb{R}_+, \{[0; x] \mid x \in \mathbb{R}_+\}, Q, r, \delta_{x_0})$ pour cette situation.

La modélisation d'un problème sous forme de MDP peut être très simple comme dans cet exemple. Toutefois, nous allons voir dans les chapitres qui vont suivre qu'elle peut se révéler fort complexe.

1.2 Stratégies et propriété de Markov

Parmi les stratégies de Π, nous pouvons distinguer des sous-ensembles de stratégies remarquables. Pour ce faire, introduisons une définition préliminaire.

Définition 1.2.0.1. On définit l'ensemble $\Phi = \{\varphi \in \mathcal{P}(\mathbb{A} \mid \mathbb{X}) \mid \varphi(\mathbb{A}(x) \mid x) = 1, \forall x \in \mathbb{X}\}$. On définit également $\mathbb{F} = \{f : \mathbb{X} \to \mathbb{A} \text{ mesurable} \mid f(x) \in \mathbb{A}(x), \forall x \in \mathbb{X}\}$. Les fonctions de $\mathbb{F}$ sont appelées sélecteurs de la fonction $x \mapsto \mathbb{A}(x)$.

Il est possible d'identifier une fonction f de $\mathbb{F}$ à un noyau stochastique φ de Φ au moyen de l'égalité $\varphi(C \mid x) = \mathbb{1}_C(f(x))$ pour tout $x \in \mathbb{X}$ et tout $C \in \mathcal{B}(\mathbb{A})$. On peut alors voir $\mathbb{F}$ comme un sous-ensemble de Φ.

Définition 1.2.0.2. Une stratégie $\pi = (\pi_k)_{k \geq 0}$ est dite :

— *markovienne aléatoire* s'il existe une suite (φ_k) d'éléments de Φ telle que

$$\pi_k(\cdot \mid h_k) = \varphi_k(\cdot \mid x_k) \text{ pour tout } h_k \in H_k. \tag{1.7}$$

L'ensemble des telles stratégies est noté Π_{MA}.

— *stationnaire aléatoire* s'il existe $\varphi \in \Phi$ tel que

$$\pi_k(\cdot \mid h_k) = \varphi(\cdot \mid x_k) \text{ pour tout } h_k \in H_k. \tag{1.8}$$

L'ensemble des telles stratégies est noté Π_{SA}.

— *déterministe* s'il existe une suite (g_k) de fonctions mesurables $g_k : H_k \to \mathbb{A}$ telle que, pour tout h_k dans H_k, $g_k(h_k) \in \mathbb{A}(x_k)$ et

$$\pi_k(C \mid h_k) = \mathbb{1}_C(g_k(h_k)) \text{ pour tout } C \in \mathcal{B}(\mathbb{A}). \tag{1.9}$$

L'ensemble des telles stratégies est noté Π_D.

— *déterministe markovienne* s'il existe une suite (f_k) de sélecteurs telle que

$$\pi_k(C \mid h_k) = \mathbb{1}_C(f_k(x_k)) \text{ pour tout } C \in \mathcal{B}(\mathbb{A}). \tag{1.10}$$

L'ensemble des telles stratégies est noté Π_{DM}.

— *déterministe stationnaire* s'il existe un sélecteur f tel que

$$\pi_k(C \mid h_k) = \mathbb{1}_C(f(x_k)) \text{ pour tout } C \in \mathcal{B}(\mathbb{A}). \tag{1.11}$$

L'ensemble des telles stratégies est noté Π_{DS}.

Remarque. On a clairement les inclusions $\Pi_{SA} \subset \Pi_{MA} \subset \Pi$ et $\Pi_{DS} \subset \Pi_{DM} \subset \Pi_D \subset \Pi$. Le fait que $\mathbb{F} \subset \Phi$ implique que $\Pi_{DS} \subset \Pi_{SA}$. Cela veut donc dire que l'hypothèse selon laquelle $\mathbb{F}$ est non vide est suffisante pour que tous les ensembles de stratégies définis précédemment (Π y compris) soient non vides.

Notations : on définit pour tout $x \in \mathbb{X}$

$$Q(\cdot \mid x, \varphi) = \int_{\mathbb{A}} Q(\cdot \mid x, a)\varphi(\mathrm{d}a \mid x).$$

En particulier pour $f \in \mathbb{F}$, on note $Q(B \mid x, f) = Q(B \mid x, f(x))$ et $c(x, f) = c(x, f(x))$. Par abus de notation, on confondra parfois une stratégie déterministe avec le(s) sélecteur(s) associé(s). De même on confondra une stratégie aléatoire avec le(s) noyau(x) de Φ associés.

La propriété de Markov

Définition 1.2.0.3. Soit X un espace de Borel. On se donne une suite $(R_k)_{k\geq 0}$ de noyaux stochastiques de $\mathcal{P}(X \mid X)$ et $(Z_k)_{k\geq 0}$ un processus à valeurs dans X sur un espace de probabilité $(\Xi, \mathcal{G}, \mathbb{P})$. On dit que (Z_k) est un processus de Markov inhomogène de noyaux de transition (R_k) si pour tout $B \in \mathcal{B}(X)$ et pour tout k,

$$\mathbb{P}\{Z_{k+1} \in B \mid Z_0, \ldots, Z_k\} = \mathbb{P}\{Z_{k+1} \in B \mid Z_k\} = R_k(B \mid Z_k). \tag{1.12}$$

Si tous les noyaux R_k sont égaux à un noyau stochastique R (c'est-à-dire que les R_k ne dépendent pas de k), alors on dit que (Z_k) est un processus de Markov homogène.

La première égalité dans (1.12) est appelée propriété de Markov. On énonce maintenant la proposition suivante qui donne une condition pour qu'un MDP soit un processus de Markov.

Proposition 1.2.0.4. *Soit ν une loi initiale de probabilité. Si $\pi = (\varphi_k)_{k\geq 0} \in \Pi_{MA}$, alors $(X_k)_{k\geq 0}$ construit de manière canonique est un processus de Markov inhomogène de noyaux de transition $(Q(. \mid ., \varphi_k))_{k\geq 0}$, c'est-à-dire que pour tout $B \in \mathcal{B}(\mathbb{X})$ et pour tout k,*

$$\begin{aligned}\mathbb{P}_\nu^\pi \{X_{k+1} \in B \mid (X_0, \ldots, X_k) = (x_0, \ldots, x_k)\} &= \mathbb{P}_\nu^\pi \{X_{k+1} \in B \mid X_k = x_k\} \\ &= Q(B \mid x_k, \varphi_k). \end{aligned} \tag{1.13}$$

Si $\pi = (f_k)$ est une stratégie déterministe markovienne, alors les égalités précédentes sont valables pour les noyaux $Q(\cdot \mid \cdot, f_k)$.
De plus, si π est une stratégie stationnaire, alors (X_k) est un processus de Markov homogène.

1.3 Critères de coût

Les problèmes liés au formalisme des MDP sont des questions d'optimisation. En effet, ils modélisent des situations où on a besoin de connaître la meilleure action à choisir et donc, par extension, de connaître la meilleure stratégie. Toutefois, le caractère aléatoire des problèmes considérés suppose de devoir optimiser un comportement moyen du processus. Alors, comment juger de la qualité d'une stratégie ? L'utilisation d'un coût par transition permet de quantifier ≪ localement ≫ l'optimalité d'une action. Mais cette dernière ayant potentiellement des conséquences sur le long terme, une décision qui est bonne à un certain instant peut, dans le futur, amener le processus dans des états très coûteux. Dès lors, on ne va pas chercher à optimiser le coût par transition, mais des critères de coût, c'est-à-dire des quantités qui utilisent naturellement le coût par transition mais qui prennent en compte les conséquences éventuelles d'une décision. Ces critères de coût seront des fonctions de la stratégie adoptée $\pi \in \Pi$ et de la loi initiale du processus $\nu \in \mathcal{P}(\mathbb{X})$. Le problème d'optimisation associé à un critère consiste à le minimiser (ou à le maximiser) sur l'ensemble des stratégies $\pi \in \Pi$.

Il existe un certain nombre de critères étudiés dans la littérature (cf. [HLL96]). Le choix de l'un ou de l'autre procède à la fois de la géométrie du problème et de l'interprétation que l'on a du coût. Dans les applications les plus courantes, on retrouve souvent les deux mêmes critères, que nous nous proposons de présenter ici.

1.3.1 Le critère à horizon fini

Soit un entier $N \in \mathbb{N}$. Le critère à horizon N fini est la quantité

$$V(\pi, \nu) = \mathbb{E}_\nu^\pi \left[\sum_{k=0}^{N-1} c(X_k, A_k) + h(X_N) \right], \tag{1.14}$$

où $h : \mathbb{X} \to \mathbb{R}$ est une fonction mesurable. Elle représente un coût terminal qui dépend uniquement de l'état final du système.

Ce critère est intuitivement la somme des coûts par transition obtenus lors du processus jusqu'à un instant fini et déterministe appelé horizon, à laquelle on ajoute un coût terminal. Il est utilisé tout naturellement dans les problèmes où le processus aléatoire considéré a une durée

finie fixe. On le retrouve notamment en finance, où l'horizon représente la maturité. L'application industrielle que nous allons étudier demandera par ailleurs d'optimiser un critère à horizon fini.

Remarque. Dans le cas d'un processus à horizon fini, le coût terminal h peut parfois être inclus dans le modèle markovien décisionnel associé, comme nous le ferons dans le chapitre 8.

1.3.2 Le critère pondéré

Soit un réel $\alpha \in]0;1[$. Le critère α-pondéré est la quantité

$$V_\alpha(\pi, \nu) = \mathbb{E}_\nu^\pi \left[\sum_{k=0}^{\infty} \alpha^k c(X_k, A_k) \right]. \tag{1.15}$$

C'est un critère à horizon infini dans lequel chaque coût par transition est pondéré. Dans la théorie, ce facteur α sert à assurer la convergence de la série sous certaines hypothèses. En revanche, selon les problèmes, il peut avoir une interprétation concrète (taux d'actualisation en finance, par exemple). Ce critère est notamment associé à des processus dont on ne connaît pas a priori la durée. Les problèmes de gestion d'inventaire ou de maintenance en font un large usage.

1.4 Processus markoviens décisionnels partiellement observables

Dans un certain nombre de problèmes, une partie des informations sur l'état courant du système peut ne pas être accessible au contrôleur. Ces informations non observables ne peuvent naturellement pas intervenir dans la prise de décision. Pourtant, c'est de l'état en entier (et de la décision) que dépend la transition vers l'état suivant, et éventuellement le coût par transition. On parle alors de problèmes *partiellement observables*. De nombreux domaines comportent des exemples de tels problèmes : la finance, où certains paramètres du marché sont inconnus, ou bien l'ingénierie. Le formalisme des MDP que l'on a expliqué précédement devient inadapté. Ici, nous allons nous attacher à décrire brièvement le formalisme des POMDP en suivant [BR11]. Il reste similaire à celui des MDP, mais le principal point à éclaircir est le fait que la décision ne dépend que d'une partie de l'état. Notons que nous n'allons envisager que des processus partiellement observables en horizon fini, ce que le formalisme des POMDP sous-entend.

Définition 1.4.0.1. Un modèle markovien décisionnel partiellement observable est la donnée d'un septuplet $(\mathbb{X} \times \mathbb{Y}, \mathbb{A}, \{\mathbb{A}(y) \mid y \in \mathbb{Y}\}, Q, c, h, \nu)$ comprenant :

- un espace d'états $\mathbb{X} \times \mathbb{Y}$ où $\mathbb{X}$ et $\mathbb{Y}$ sont des espaces de Borel, avec $\mathbb{X}$ l'espace des états *non observables* et $\mathbb{Y}$ l'espace des états *observables* ;
- un espace de Borel $\mathbb{A}$ qui est l'espace des actions ;
- un ensemble $\{\mathbb{A}(y) \mid y \in \mathbb{Y}\}$ de sous-ensembles mesurables non vides de $\mathbb{A}$, où $\mathbb{A}(y)$ est *l'ensemble des actions réalisables lorsque le système est dans l'état* $(x, y) \in \mathbb{X} \times \mathbb{Y}$ (il ne dépend donc que de la composante observable), tel que $\mathbb{K}' = \{(y, a) \mid y \in \mathbb{Y}, a \in \mathbb{A}(y)\}$ contienne le graphe d'une fonction mesurable de $\mathbb{Y}$ dans $\mathbb{A}$;

— un noyau stochastique Q sur $\mathbb{X} \times \mathbb{Y}$ sachant $\mathbb{X} \times \mathbb{K}'$ appelé *loi de transition* ;

— une fonction mesurable $c : \mathbb{X} \times \mathbb{K}' \to \mathbb{R}$ appelée *fonction de coût par transition* ;

— une fonction mesurable $h : \mathbb{X} \times \mathbb{Y} \to \mathbb{R}$ appelée *coût terminal* ;

— une loi de probabilité initiale ν sur $\mathbb{X}$.

Dans toute la suite du chapitre, on considérera un entier $N \in \mathbb{N}$ qui sera l'horizon du processus. Comme le montre la définition du modèle, la transition dépend à la fois de tout l'état et de la décision. En revanche, les actions ne peuvent se décider qu'en se basant sur la composante observable. La figure 1.2 schématise la transition d'un POMDP.

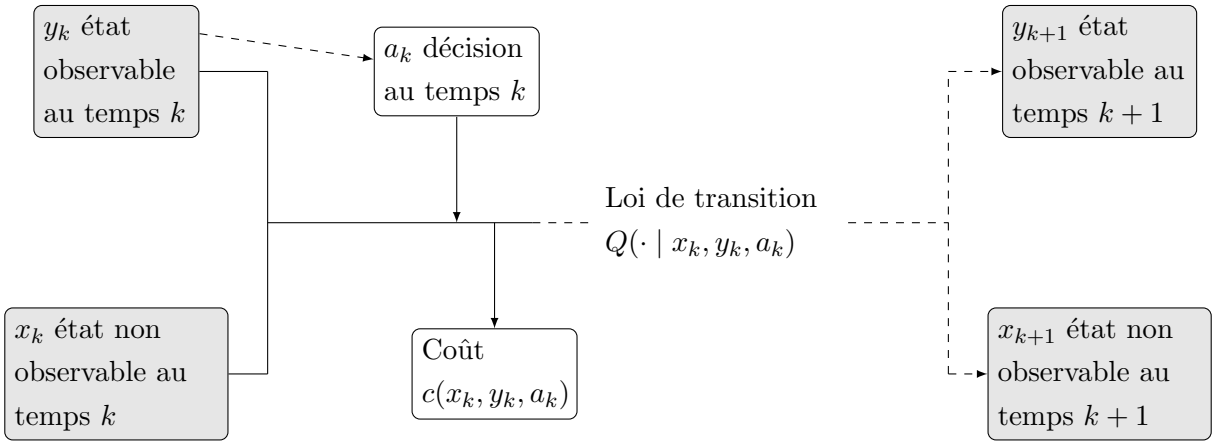

FIGURE 1.2 – Principe d'un processus markovien décisionnel partiellement observable

Dans la suite, on se donne un modèle markovien décisionnel partiellement observable

$$(\mathbb{X} \times \mathbb{Y}, \mathbb{A}, \{\mathbb{A}(y) \mid y \in \mathbb{Y}\}, Q, c, h, \nu).$$

On définit pour tout $0 \leq k \leq N$ l'ensemble des *histoires observables* jusqu'au temps k par récurrence par

$$\begin{cases} H_0^{\mathrm{o}} = \mathbb{Y} \\ H_k^{\mathrm{o}} = H_{k-1}^{\mathrm{o}} \times \mathbb{A} \times \mathbb{Y} \end{cases}. \tag{1.16}$$

Un élément de H_k^{o} est appelé *histoire observable au rang k*.

On peut à présent définir ce qu'est une stratégie pour ce type de processus. Pour faire un parallèle avec le formalisme des MDP, remarquons que l'on va définir des stratégies déterministes.

Définition 1.4.0.2. (a) On appelle *règle de décision H^{o}-dépendante au rang k* toute fonction f_k mesurable de H_k^{o} dans $\mathbb{A}$.

(b) On appelle *stratégie H^{o}-dépendante* une suite $\pi = (f_0, \ldots, f_{N-1})$ où pour $k \in [\![0; N-1]\!]$, chaque f_k est une règle de décision H^{o}-dépendante au rang k.

L'ensemble des stratégies H^{o}-dépendantes est noté Π^{o}.

Soient $\pi = (f_0, \dots, f_{N-1}) \in \Pi^{\mathrm{o}}$, $y \in \mathbb{Y}$ un état observable initial et $\Omega' = (\mathbb{X} \times \mathbb{Y})^{N+1}$. Pour tout élément $\omega = (x_0, y_0, \dots, x_N, y_N) \in \Omega'$, on définit les variables aléatoires

$$X_k(\omega) = x_k \text{ et } Y_k(\omega) = y_k \tag{1.17}$$

pour tout $k \in [\![0; N]\!]$. La suite des actions successives choisies selon la stratégie π est définie par récurrence par

$$\begin{cases} A_0 = f_0(Y_0) \\ A_k = f_k(Y_0, A_0, \dots, Y_{k-1}, A_{k-1}, Y_k) \text{ pour tout } 1 \leq k \leq N-1. \end{cases} \tag{1.18}$$

Soit $y \in \mathbb{Y}$ un état observable initial. Le théorème de Ionescu-Tulcea donne l'existence d'une unique mesure de probabilité $\mathbb{P}_y^\pi$ sur $(\Omega', \mathcal{G})$, $\mathcal{G}$ étant la tribu produit associée à Ω', qui vérifie les égalités suivantes pour tous $B \in \mathcal{B}(\mathbb{X})$, $C \in \mathcal{B}(\mathbb{Y})$, $D \in \mathcal{B}(\mathbb{A})$ et $h_k \in H_k^{\mathrm{o}}$:

$$\mathbb{P}_y^\pi \{(X_0, Y_0) \in B \times C\} = \nu(B)\delta_y(C), \tag{1.19}$$

$$\mathbb{P}_y^\pi \{A_k \in D \mid (Y_0, A_0, \dots, Y_{k-1}, A_{k-1}, Y_k) = h_k\} = \delta_{f_k(h_k)}(D), \tag{1.20}$$

$$\mathbb{P}_y^\pi \{(X_{k+1}, Y_{k+1}) \in B \times C \mid X_0, Y_0, A_0, \dots, X_k, Y_k, A_k\} = Q(B \times C \mid X_k, Y_k, A_k). \tag{1.21}$$

Soit également une mesure de probabilité $\mathbb{P}_{x,y}^\pi$ sur $\mathbb{X} \times \mathbb{Y}$ telle que

$$\mathbb{P}_y^\pi(\cdot) = \int_{\mathbb{X}} \mathbb{P}_{x,y}^\pi(\cdot)\nu(\mathrm{d}x). \tag{1.22}$$

On notera $\mathbb{E}_{x,y}^\pi$ l'espérance associée à $\mathbb{P}_{x,y}^\pi$. On définit dès lors ce qu'est un POMDP.

Définition 1.4.0.3. Un processus markovien décisionnel partiellement observable à temps discret est le processus stochastique $(X_k, Y_k)_{0 \leq k \leq N}$ défini sur l'espace probabilisé $(\Omega', \mathcal{G}, \mathbb{P}_y^\pi)$.

Le critère à horizon fini N est donc défini de la manière suivante : pour une stratégie $\pi \in \Pi^{\mathrm{o}}$ et $y \in \mathbb{Y}$ un état initial, on pose

$$V^{\mathrm{po}}(\pi, y) = \int_{\mathbb{X}} \mathbb{E}_{x,y}^\pi \left[\sum_{k=0}^{N-1} c(X_k, Y_k, A_k) + h(X_N, Y_N) \right] \nu(\mathrm{d}x). \tag{1.23}$$

Le problème d'optimisation associé à ce critère consiste à minimiser la quantité $V^{\mathrm{po}}(y, \pi)$ sur l'ensemble des stratégies $\pi \in \Pi^{\mathrm{o}}$.

2 Techniques classiques de résolution

Avec l'utilisation du formalisme des MDP ou des POMDP décrite au chapitre 1 vient la question de l'optimisation d'un critère de coût. Il va s'agir de calculer [1] la fonction valeur optimale du critère à horizon fini

$$x \mapsto V^*(x) = \inf_{\pi \in \Pi} V(\pi, \delta_x)$$

ou du critère α-pondéré

$$x \mapsto V_\alpha^*(x) = \inf_{\pi \in \Pi} V_\alpha(\pi, \delta_x)$$

et de trouver une stratégie π^* qui réalise la borne inférieure (ou supérieure). Dès 1970, des méthodes d'optimisation ont été développées en se basant sur des concepts tels que la programmation dynamique ([How60, Ber87]) ou la programmation linéaire ([Dan98, Mur09]). Ces travaux ont donné lieu à l'élaboration d'algorithmes classiques très utilisés dans les applications.

Dans ce chapitre, nous proposons une brève revue de ces techniques standard d'optimisation jalonnée de références vers des exemples d'utilisation. Il nous a paru intéressant de non seulement décrire les algorithmes et les méthodes, mais aussi de reprendre un petit bagage théorique pour justifier leur provenance. Afin de ne pas alourdir notre propos, nous n'allons pas donner les démonstrations des résultats énoncés ici. Ces dernières peuvent être trouvées dans la bibliographie associée. Les théorèmes sont essentiellement issus de [HLL96], de [HLL99] et de [BR11] (pour la partie sur les POMDP).

La partie 2.1 traite de la programmation dynamique, qui donne lieu à un ou deux algorithmes selon le critère à optimiser. Ensuite, dans la partie 2.2, nous expliquerons en quoi

1. Dans le cas d'une maximisation de récompense, il faudra considérer des sup à la place des inf.

consiste la résolution de problèmes via la programmation linéaire. Enfin, dans la partie 2.3, nous détaillerons la manière classique, développée dans [BR11], de transformer un POMDP en un MDP complètement observable pour tenter d'appliquer les techniques vues précédemment.

Dans toute la suite de ce chapitre, jusqu'à la partie sur les POMDP, on se donne un modèle markovien décisionnel

$$(\mathbb{X}, \mathbb{A}, \{\mathbb{A}(x) \mid x \in \mathbb{X}\}, Q, c, \nu)$$

et nous utiliserons largement les notations du chapitre 1.

2.1 Programmation dynamique

Le principe de programmation dynamique (ou principe d'optimalité de Bellman, du nom de Richard Bellman qui a introduit le concept dans les années 1950) consiste à optimiser des sous-problèmes plus simples pour parvenir à l'optimum du problème initial. Dans le cas des MDP, la fonction valeur optimale est approchée par une suite définie par récurrence qu'on appelle les *équations de Bellman* ou les *équations de programmation dynamique*. Elles peuvent prendre différentes formes selon le critère de coût choisi. Elles sont à la base d'algorithmes que l'on présente ici ([BS78, Put94, HLL96]) dans le cas d'une minimisation d'un coût. On déduit facilement les procédures qui en découlent pour une maximisation de récompense.

2.1.1 Critère à horizon fini $N \in \mathbb{N}$

On rappelle qu'étant donnée une fonction de coût terminale $h : \mathbb{X} \to \mathbb{R}$, le critère à horizon fini N est défini par

$$V(\pi, \nu) = \mathbb{E}_\nu^\pi \left[\sum_{k=0}^{N-1} c(X_k, A_k) + h(X_N) \right].$$

Dans la suite, pour $x \in \mathbb{X}$, on notera $V(\pi, x)$ et $\mathbb{E}_x^\pi$ plutôt que $V(\pi, \delta_x)$ et $\mathbb{E}_{\delta_x}^\pi$ respectivement.

L'optimisation de ce critère repose sur le théorème de programmation dynamique (que l'on peut trouver dans [HLL96] section 3.2) qui établit une récurrence initialisée à h.

Théorème 2.1.1.1. *Soient les fonctions $v_0, v_1, \ldots, v_N$ de $\mathbb{X}$ dans $\mathbb{R}$ définies par récurrence descendante par*

$$v_N(x) = h(x) \text{ pour tout } x \in \mathbb{X}, \tag{2.1}$$

et pour k allant de $N-1$ à 0

$$v_k(x) = \min_{a \in \mathbb{A}(x)} \left[c(x, a) + \int_{\mathbb{X}} v_{k+1}(y) Q(\mathrm{d}y \mid x, a) \right]. \tag{2.2}$$

Supposons que ces fonctions soient mesurables et que, pour tout k dans $[\![0; N-1]\!]$, il existe un sélecteur $f_t \in \mathbb{F}$ tel que $f_k(x) \in \mathbb{A}(x)$ atteigne le minimum dans (2.2). Alors la stratégie déterministe markovienne $\pi^* = \{f_0, \ldots, f_{N-1}\}$ est optimale (c'est-à-dire que $V^*(x) = V(\pi^*, x)$ pour tout $x \in \mathbb{X}$) et la fonction de valeur optimale V^* est égale à v_0.

Algorithme 1 Critère à horizon fini N

pour tout $x \in \mathbb{X}$ **faire**
  $v_N(x) \leftarrow h(x)$
fin pour
pour k **de** $N-1$ **à** 0 **faire**
  pour tout $x \in \mathbb{X}$ **faire**
    $$v_k(x) = \min_{a \in \mathbb{A}(x)} \left[c(x,a) + \int_{\mathbb{X}} v_{k+1}(y) Q(\mathrm{d}y \mid x, a) \right]$$
    $$\pi_k^*(x) = \operatorname*{argmin}_{a \in \mathbb{A}(x)} \left[c(x,a) + \int_{\mathbb{X}} v_{k+1}(y) Q(\mathrm{d}y \mid x, a) \right]$$
  fin pour
fin pour
retourner v_0, π^*

Ce théorème donne lieu à la procédure décrite par l'algorithme 1. Elle présente un certain nombre de points qui peuvent entraîner des difficultés.

1. Elle nécessite de connaître les fonctions v_k en chaque point $x \in \mathbb{X}$, en particulier pour le calcul d'intégrale. Donc, lorsque $\mathbb{X}$ est infini ou non dénombrable, l'application de cet algorithme risque d'être rendue difficile.
2. La recherche du minimum pour le calcul des v_k se complique lorsque $\mathbb{A}$ est très grand.
3. Le calcul des quantités mises en jeu nécessitent de connaître analytiquement la fonction de coût et la loi de transition. Ce n'est pas toujours le cas, comme nous le verrons dans l'application industrielle.

Les exemples d'utilisation de ce critère sont nombreux : parmi eux, on trouve la gestion d'inventaire ([Put94] section 4.6.1 ou [HLL96] section 3.7). Il s'agit d'une classe de problèmes où il faut gérer correctement le stock d'un certain produit pour pouvoir honorer les demandes des clients tout en évitant des coûts liés au stockage trop élevés. L'application industrielle des MDP que nous allons étudier dans la partie II de notre travail en est un cas très particulier. Certains problèmes d'arrêt optimal, comme l'exercice d'une option d'achat ou de vente ([Put94] section 3.4.4), peuvent faire appel à cet algorithme.

2.1.2 Critère pondéré

Soit $0 < \alpha < 1$. Le critère α-pondéré est la quantité

$$V_\alpha(\pi, \nu) = \mathbb{E}_\nu^\pi \left[\sum_{k=0}^{\infty} \alpha^k c(X_k, A_k) \right].$$

Les équations de programmation dynamique pour ce critère reposent sur le constat que la fonction valeur optimale V_α^* est solution de *l'équation d'optimalité α-pondérée*, c'est-à-dire qu'elle vérifie

$$V_\alpha^*(x) = \min_{a \in \mathbb{A}(x)} \left[c(x,a) + \alpha \int_{\mathbb{X}} V_\alpha^*(y) Q(\mathrm{d}y \mid x, a) \right]. \tag{2.3}$$

Notons la similarité avec les équations de Bellman dans le cas de l'horizon fini. Nous aurons besoin de la définition suivante.

Définition 2.1.2.1. Soient un espace de Borel E et une fonction mesurable $f : E \to \mathbb{R}$. On dit que f est semi-continue inférieurement en $x \in E$ si, pour toute suite (x_n) d'éléments de E qui converge vers x, $\liminf_{n\to+\infty} f(x_n) \geq f(x)$. Si f est semi-continue inférieurement en tout $x \in E$, on dit que f l'est sur E.

Le théorème suivant, issu de [HLL96] (section 4.2), donne lieu à deux algorithmes de résolution.

Théorème 2.1.2.2. *On suppose que :*

- *c est semi-continue inférieurement et minorée sur $\mathbb{K}$, et telle que $\{a \in \mathbb{A}(x) \mid c(x,a) \leq r\}$ est compact pour tous $x \in \mathbb{X}$ et $r \in \mathbb{R}$;*
- *pour toute fonction mesurable bornée $v : \mathbb{X} \to \mathbb{R}$, la fonction $(x,a) \mapsto \int_{\mathbb{X}} v(y)Q(\mathrm{d}y \mid x,a)$ est continue et bornée sur $\mathbb{K}$;*
- *il existe une stratégie $\pi \in \Pi$ telle que $V_\alpha(\pi,x) < +\infty$ pour tout $x \in \mathbb{X}$.*

Alors :

1. *la fonction V_α^* est la solution minimale à l'équation d'optimalité α-pondérée, i.e. pour toute autre solution u, $u \geq V_\alpha^*$;*
2. *il existe un sélecteur $f^* \in \mathbb{F}$ qui réalise le minimum dans (2.3) dont la stratégie déterministe stationnaire associée est optimale et réciproquement, si $f \in \Pi_{DS}$ est optimale, alors elle satisfait (2.3).*

Les équations de programmation dynamique approchent V_α^* par une suite récurrente en utilisant (2.3). Selon la manière d'interpréter la récurrence, on arrive à deux algorithmes : l'itération de la valeur ([Ber87, HLL96]) et l'itération de la stratégie ([BS78, Den03, How60, WW89]).

Itération de la valeur

On va définir la suite de fonctions suivante :

$$\begin{cases} v_0 \equiv 0 \\ v_n(x) = \min\limits_{a\in\mathbb{A}(x)} \left[c(x,a) + \alpha \int_{\mathbb{X}} v_{n-1}(y)Q(\mathrm{d}y \mid x,a) \right] \text{ pour tous } x \in \mathbb{X}, n \geq 1 \end{cases} \tag{2.4}$$

On a alors la proposition suivante

Proposition 2.1.2.3. *La suite (v_n) est une suite croissante qui converge vers V_α^*.*

L'algorithme d'itération de la valeur rend donc une approximation de la valeur optimale et de la stratégie optimale. La procédure est décrite dans l'algorithme 2.

Algorithme 2 Itération de la valeur

pour tout $x \in \mathbb{X}$ **faire**
  $v_0(x) \leftarrow 0$
fin pour
$n \leftarrow 0$
tant que $\|v_{n+1} - v_n\| > \varepsilon$ **faire**
  pour tout $x \in \mathbb{X}$ **faire**
    $v_{n+1}(x) = \min_{a \in \mathbb{A}(x)} \left[c(x,a) + \alpha \int_{\mathbb{X}} v_n(y) Q(\mathrm{d}y \mid x, a) \right]$
    $n \leftarrow n + 1$
  fin pour
fin tant que
pour tout $x \in \mathbb{X}$ **faire**
  $\pi(x) = \operatorname{argmin}_{a \in \mathbb{A}(x)} \left[c(x,a) + \alpha \int_{\mathbb{X}} v_{n+1}(y) Q(\mathrm{d}y \mid x, a) \right]$
fin pour
retourner v_{n+1},π

Itération de la stratégie

Comme le nom de l'algorithme l'indique, la récurrence va ici porter sur la stratégie. Soit g_0 une stratégie déterministe stationnaire arbitraire. On définit deux suites (g_n) et (w_n) de la façon suivante.

1. Un sélecteur g_n étant donné, on pose $w_n(x) = V_\alpha(g_n, x)$ pour tout x (on *évalue* la stratégie courante).
2. Le sélecteur qui réalise le minimum dans $\min_{a \in \mathbb{A}(x)} \left[c(x,a) + \alpha \int_{\mathbb{X}} w_n(y) Q(\mathrm{d}y \mid x, a) \right]$ est noté g_{n+1} (on *améliore* la stratégie).

On a alors le résultat ci-après.

Théorème 2.1.2.4. *S'il existe un entier n pour lequel* $w_n \equiv w_{n+1}$ [2]*, alors* w_n *réalise le minimum dans* (2.3)*. Si, de plus,* $\lim_{n \to +\infty} \alpha^n \mathbb{E}_x^\pi w_n(x_n) = 0$, $\forall x \in X$, $\forall \pi \in \Pi$ *telle que* $V_\alpha(\pi, x)$ *est fini pour tout* x*, alors* $w \equiv V_\alpha^*$ *et* $\pi^* = g_n$ *est optimale.*

Cet algorithme rend donc la fonction valeur optimale et une stratégie optimale. La procédure est décrite dans l'algorithme 3.

Remarque. Les deux algorithmes que nous venons de décrire souffrent des mêmes difficultés d'application que l'algorithme en horizon fini.

Les problèmes d'arrêt optimal sont de très bons exemples d'utilisation de ce critère ([Put94]

2. De façon générale, $(w_n) \searrow w$, avec w mesurable positive un minimum dans (2.3).

Algorithme 3 Itération de la stratégie

Entrée : $g_0 \in \Pi_{DS}$

$n \leftarrow 0$

tant que $w_{n+1} \not\equiv w_n$ **faire**

Résoudre $c(x, g_n(x)) + \alpha \int_{\mathbb{X}} w_n(y) Q(\mathrm{d}y \mid x, g_n(x)) = w_n(x)$ pour tout $x \in \mathbb{X}$

pour tout $x \in \mathbb{X}$ **faire**

$$g_{n+1}(x) = \underset{a \in \mathbb{A}(x)}{\operatorname{argmin}} \left[c(x,a) + \alpha \int_{\mathbb{X}} w_n(y) Q(\mathrm{d}y \mid x, a) \right]$$

fin pour

$n \leftarrow n + 1$

fin tant que

retourner w_n, g_{n+1}

section 3.4.2). Mais on le retrouve également en finance, avec la gestion de portefeuille ([HLL96] section 1.3.2).

2.2 Programmation linéaire

Cette technique d'optimisation consiste, comme son nom l'indique par ailleurs, à optimiser une fonction linéaire sur un ensemble de variables qui doivent respecter des contraintes sous forme de fonctions linéaires, ce qui en fait une procédure particulièrement intéressante pour des problèmes d'optimisation contraints ([DPR13]). La programmation linéaire a fait l'objet de nombreuses études ([Dan98, HLL96, Mur09]).

L'adaptation de cette technique aux MDP restreint l'étude aux critères à horizon infini (le critère pondéré dans notre cas). Elle suppose de changer de variable d'intérêt. En effet, optimiser le critère pondéré revient à trouver une stratégie π tel que $V_\alpha(\pi, \nu)$ soit optimal. Mais $\pi \mapsto V_\alpha(\pi, \nu)$ n'est pas linéaire en général ! On va alors transformer légèrement le critère. En effet, on commence d'abord par remarquer que l'application

$$\Gamma \mapsto \mu_\nu^\pi(\Gamma) = \sum_{k=0}^{+\infty} \alpha^k \mathbb{P}_\nu^\pi \{(X_k, A_k) \in \Gamma\}$$

définit une mesure sur $\mathbb{X} \times \mathbb{A}$ souvent appelée *fréquence état-action*. Comme pour tout entier $k \in \mathbb{N}$, $\mathbb{E}_\nu^\pi[c(X_k, A_k)] = \int_{\mathbb{X}\times\mathbb{A}} c(x,a) \mathbb{P}_\nu^\pi(\mathrm{d}x, \mathrm{d}a)$, alors on obtient la réécriture suivante du critère pondéré :

$$V_\alpha(\pi, \nu) = \int_{\mathbb{X}\times\mathbb{A}} c(x,a) \mu_\nu^\pi(\mathrm{d}x, \mathrm{d}a). \tag{2.5}$$

Donc, au lieu d'optimiser $\pi \mapsto V_\alpha(\pi, \nu)$, on va optimiser $\mu \mapsto \int_{\mathbb{X}\times\mathbb{A}} c(x,a) \mu(\mathrm{d}x, \mathrm{d}a)$, qui est une application linéaire.

Il est facile de remarquer que, pour tout $B \in \mathcal{B}(\mathbb{X})$,

$$\mu_\nu^\pi(B \times \mathbb{A}) - \alpha \int_{\mathbb{X}\times\mathbb{A}} Q(B \mid x, a) \mu_\nu^\pi(\mathrm{d}x, \mathrm{d}a) = \nu(B). \tag{2.6}$$

Cette contrainte est linéaire en μ_ν^π. Le problème d'optimisation linéaire associé à un MDP est donc formulé de la manière suivante :

$$\begin{aligned}&\text{optimiser } \int_{\mathbb{X}\times\mathbb{A}} c(x,a)\mu(\mathrm{d}x,\mathrm{d}a) \text{ sur l'ensemble des mesures } \mu \text{ positives}\\ &\text{telles que } \mu(B\times\mathbb{A}) - \alpha\int_{\mathbb{X}\times\mathbb{A}} Q(B\mid x,a)\mu(\mathrm{d}x,\mathrm{d}a) = \nu(B)\ \forall B\in\mathcal{B}(\mathbb{X}).\end{aligned} \tag{2.7}$$

L'équivalence entre l'optimisation sur les stratégies et l'optimisation linéaire est alors établie par le théorème suivant, issu de [HLL96] (section 6.3).

Théorème 2.2.0.1. *Soit w la fonction sur $\mathbb{K}$ à valeurs réelles définie par $w(x,a) = \overline{c} + c(x,a)$ où $\overline{c} > 0$ est arbitraire, et soit $\overline{w}(x) = \min_{a\in\mathbb{A}(x)} w(x,a)$.*

On suppose que la fonction $(x,a) \mapsto \dfrac{1}{w(x,a)}\displaystyle\int_{\mathbb{X}} \overline{w}(y)Q(\mathrm{d}y \mid x,a)$ est bornée et que les hypothèses du théorème 2.1.2.2 sont vérifiées. Alors

1. *Pour toute stratégie $\pi\in\Pi$ telle que $V_\alpha(\pi,\nu) < +\infty$, il existe une mesure positive μ^π qui respecte la contrainte* (2.6) *telle que $V_\alpha(\pi,\nu) = \displaystyle\int_{\mathbb{X}\times\mathbb{A}} c(x,a)\mu^\pi(\mathrm{d}x,\mathrm{d}a)$.*
2. *Réciproquement, si une mesure positive μ respecte la contrainte* (2.6)*, il existe une stratégie π^μ telle que $V_\alpha(\pi^\mu,\nu) < +\infty$ et $\displaystyle\int_{\mathbb{X}\times\mathbb{A}} c(x,a)\mu(\mathrm{d}x,\mathrm{d}a) = V_\alpha(\pi^\mu,\nu)$.*

Les deux problèmes d'optimisation sont donc équivalents sous ces hypothèses. Dans le cas où l'espace d'états $\mathbb{X}$ et l'espace d'actions $\mathbb{A}$ sont finis, il existe des algorithmes de résolution classiques, comme par exemple l'algorithme du simplexe ([Dan98, Mur09]). Toutefois, dans le cas général, optimiser sur un ensemble de mesures devient complexe, et il faut alors se tourner vers des méthodes d'approximation ([DPR13]) au préalable.

Un exemple de résolution via la programmation linéaire se trouve dans l'article [BAT13], qui traite de maintenance optimale. La programmation linéaire peut donc être utile dans un contexte proche de l'industrie.

2.3 Processus markoviens décisionnels partiellement observables

Pour cette section, on considère un modèle markovien décisionnel partiellement observable

$$(\mathbb{X}\times\mathbb{Y}, \mathbb{A}, \{\mathbb{A}(y) \mid y\in\mathbb{Y}\}, Q, c, h, \nu)$$

ainsi que le POMDP à horizon fini N associé $(X_k, Y_k)_{0\le k\le N}$ construit canoniquement. Nous reprenons ici les notations de la section 1.4.

L'algorithme de résolution du critère à horizon fini ne peut pas être appliqué tel quel du fait que l'action ne dépend pas de tout l'état. L'argument usuel pour pallier cette difficulté est de transformer ce modèle en un modèle complètement observable pour lequel les équations de programmation dynamique vont s'appliquer. Le changement principal va porter sur l'espace

des états : au lieu de considérer l'ensemble des états non observables, on va plutôt chercher à connaître leur loi de probabilité conditionnellement à l'histoire observable.

La construction donnée dans cette section est reprise du chapitre 5 de [BR11]. On fait l'hypothèse que la loi de transition Q possède une densité q par rapport à un produit tensoriel de mesures σ-finies λ et μ, c'est-à-dire qu'on peut écrire

$$Q(\mathrm{d}x', \mathrm{d}y' \mid x, y, a) = q(x', y' \mid x, y, a)\lambda(\mathrm{d}x')\mu(\mathrm{d}y'). \tag{2.8}$$

On définit alors un opérateur $\Phi : \mathcal{P}(\mathbb{X}) \times \mathbb{Y} \times \mathbb{A} \times \mathbb{Y} \to \mathcal{P}(\mathbb{X})$, appelé *opérateur de Bayes*, par

$$\Phi(\rho, y, a, y')(C) = \frac{\displaystyle\int_C \left(\int_{\mathbb{X}} q(x', y' \mid x, y, a)\rho(\mathrm{d}x) \right) \lambda(\mathrm{d}x')}{\displaystyle\int_{\mathbb{X}} \left(\int_{\mathbb{X}} q(x', y' \mid x, y, a)\rho(\mathrm{d}x) \right) \lambda(\mathrm{d}x')} \text{ pour tout } C \in \mathcal{B}(\mathbb{X}). \tag{2.9}$$

Cet opérateur va servir à construire une *équation de filtrage*, c'est-à-dire une suite récurrente de mesures de probabilités conditionnelles. Pour tous $k \in [\![0; N-1]\!]$, $h_{k+1} = (h_k, a_k, y_{k+1}) \in H^{\mathrm{o}}_{k+1}$, on définit

$$\begin{cases} m_0(B) = \nu(B) \\ m_{k+1}(B \mid h_k, a_k, y_{k+1}) = \Phi(m_k(\cdot \mid h_k), y_k, a_k, y_{k+1})(B) \end{cases} \tag{2.10}$$

pour tout $B \in \mathcal{B}(\mathbb{X})$. Cette équation de filtrage définit une loi conditionnelle sur X_k sachant $(Y_0, A_0, \ldots, Y_k)$.

Théorème 2.3.0.1. *Pour toute stratégie $\pi \in \Pi^{\mathrm{o}}$, tout $B \in \mathcal{B}(\mathbb{X})$ et tout $k \in [\![0; N]\!]$,*

$$m_k(B \mid Y_0, A_0, \ldots, Y_k) = \mathbb{P}^{\pi}_y \left\{ X_k \in B \mid Y_0, A_0, \ldots, Y_k \right\}. \tag{2.11}$$

On introduit alors le modèle markovien décisionnel complètement observable suivant :

$$\left(\mathcal{P}(\mathbb{X}) \times \mathbb{Y}, \mathbb{A}, \{ \mathbb{A}(\rho, y) = \mathbb{A}(y) \mid (\rho, y) \in \mathcal{P}(\mathbb{X}) \times \mathbb{Y} \}, Q', c', h', \delta_{(\nu,y)} \right)$$

où

— pour tous $(\rho, y) \in \mathcal{P}(\mathbb{X}) \times \mathbb{Y}$, $a \in \mathbb{A}(\rho, y)$, $B \in \mathcal{B}(\mathcal{P}(\mathbb{X}))$ et $C \in \mathcal{B}(\mathbb{Y})$,

$$Q'(B \times C \mid \rho, y, a) = \int_{\mathbb{X}} \left(\int_{\mathbb{X} \times C} \mathbb{1}_B(\Phi(\rho, y, a, y'))Q(\mathrm{d}x', \mathrm{d}y' \mid x, y, a) \right) \rho(\mathrm{d}x), \tag{2.12}$$

— $c' : \mathcal{P}(\mathbb{X}) \times \mathbb{Y} \times \mathbb{A} \to \mathbb{R}$ est donnée par $c'(\rho, y, a) = \displaystyle\int_{\mathbb{X}} c(x, y, a)\rho(\mathrm{d}x)$,

— $h' : \mathcal{P}(\mathbb{X}) \times \mathbb{Y} \to \mathbb{R}$ est donnée par $h'(\rho, y) = \displaystyle\int_{\mathbb{X}} h(x, y)\rho(\mathrm{d}x)$.

C'est sur ce modèle que le travail d'optimisation va porter. Soit $\pi \in \Pi$. On munit l'espace mesurable $\left((\mathcal{P}(\mathbb{X}) \times \mathbb{Y} \times \mathbb{A})^{N+1}, \mathcal{T} \right)$, avec $\mathcal{T}$ la tribu-produit associée, de l'unique mesure de probabilité $\mathbb{P}^{\pi}_{(\nu,y)}$ donnée par le théorème de Ionescu-Tulcea. On peut alors construire canoniquement des processus $(P_k, Y_k)_{0 \leq k \leq N}$ à valeurs dans $\mathcal{P}(\mathbb{X}) \times \mathbb{Y}$, $(A_k)_{0 \leq k \leq N-1}$ à valeurs dans $\mathbb{A}$.

Donc, on reprend le formalisme des MDP développé dans le chapitre 1. Entre autres, le critère à horizon fini pour le modèle complètement observable s'écrit

$$V(\pi, \nu, y) = \mathbb{E}^{\pi}_{(\nu,y)} \left[\sum_{k=0}^{N-1} c'(P_k, Y_k, A_k) + h'(P_N, Y_N) \right] \tag{2.13}$$

avec $\mathbb{E}^{\pi}_{(\nu,y)}$ l'espérance associée à $\mathbb{P}^{\pi}_{(\nu,y)}$. L'équivalence entre le problème partiellement observable et le problème complètement observable est donnée par le résultat suivant (cf. [BR11], section 5.3).

Théorème 2.3.0.2. *Soit $\pi \in \Pi$. Alors, pour tout $y \in \mathbb{Y}$,*

$$V^{\mathrm{po}}(\pi, y) = V(\pi, \nu, y). \tag{2.14}$$

Sous certaines hypothèses indiquées dans [BR11] (section 2.3), le théorème de programmation dynamique peut s'appliquer. Ainsi, il devient possible d'utiliser les équations de Bellman décrites dans la section 2.1.1. Toutefois, comme nous le verrons dans notre application théorique, si l'espace des états non observables $\mathbb{X}$ est général, on aura remplacé un espace éventuellement de dimension finie par un espace de mesures de probabilité de dimension infinie ! Il faut donc envisager des méthodes d'approximation.

La finance propose de nombreux problèmes partiellement observés : la section 6.2 de [BR11] est consacrée à un exemple de résolution. La partie III de notre travail, qui traite d'une application théorique, décrit plus en détail l'optimisation d'un problème partiellement observé.

2.4 Conclusion

Depuis les années 1970, la théorie des MDP s'est considérablement étoffée et des algortithmes de résolution ont vu le jour. Nous avons présenté dans ce chapitre les techniques qu'on peut qualifier de standard. Ce sont les procédures que l'on retrouve dans tout traité sur les MDP ([HLL96, Put94, BR11] par exemple). En fonction du point de vue adopté sur un problème donné, l'une ou l'autre peut se révéler plus ou moins judicieuse.

Toutefois, comme on a pu le constater, ces algorithmes semblent beaucoup plus adaptés au cas où l'espace d'états est fini (et de préférence petit, pour des questions de complexité). Lorsqu'il est quelconque, le calcul des intégrales peut être fastidieux, selon la forme du noyau de transition. À cette difficulté s'ajoute celle induite par l'espace des actions. En effet, s'il est quelconque, la recherche de l'argmin est en général difficile. Donc même si, en théorie, ces algorithmes convergent et rendent les valeurs optimales, leur implémentation numérique peut poser problème. De même, si la programmation linéaire a l'avantage d'être bien adaptée aux questions d'optimisation contrainte ([DP10, DPR13]), elle remplace l'optimisation sur les stratégies par l'optimisation sur un espace de mesures de dimension potentiellement infinie. Enfin, pour les POMDP, remplacer l'espace des états non observables par l'ensemble des mesures sur celui-ci peut amener à considérer un MDP complètement observable dont l'espace des états est de dimension infinie.

Dans le cas général, une étape de discrétisation est donc souvent nécessaire ([DPR12, DPR13, DPR15]). C'est d'ailleurs ce que nous allons faire dans le chapitre 8. Mais les difficultés peuvent également provenir du noyau de transition et de la fonction de coût. Comme nous le verrons dans le chapitre 4, où nous décrivons la modélisation adoptée pour le problème d'Airbus, nous n'avons pas toujours accès à une écriture analytique de la loi de transition, comme le requièrent les techniques standard. Il faut donc se tourner vers d'autres solutions pour l'optimisation. La recherche récente a permis notamment d'élaborer des algorithmes basés sur des simulations et la méthode de Monte-Carlo ([CFHM07, HHC12, CHFM13]). L'application industrielle sera notamment l'occasion d'appliquer de tels algorithmes.

Deuxième partie

Une application industrielle

3 Un processus d'intégration lanceur

La classe générale des problèmes de gestion d'inventaire (≪ inventory-production problems ≫) est souvent associée à des questions d'optimisation de coût ou de disponibilité des stocks. En effet, il faut gérer à la fois le stockage des produits, l'éventuel comportement aléatoire de la chaîne de production et la demande de la clientèle. Dès lors, un contrôleur extérieur doit décider d'un taux de production adéquat au regard de ce qu'il cherche à optimiser. En effet, une cadence de production trop lente induit des stocks bas mais ne permet pas de satisfaire la demande de la clientèle. À l'inverse, une cadence trop rapide permet d'honorer les commandes mais peut amener à produire des pièces inutilisées. Ces problèmes donnent lieu à beaucoup d'applications pratiques, notamment dans le domaine industriel. Pour cette raison, ils se rencontrent fréquemment dans la littérature ([CL88, JM74, SL96]), et de nombreuses techniques ont été utilisées pour les résoudre.

Le travail de cette thèse a été motivé en premier lieu par la résolution d'un problème industriel en collaboration avec Airbus. Il peut se voir comme un cas particulier d'un problème de gestion d'inventaire. Nous nous intéressons à un processus d'intégration lanceur qui comprend plusieurs étapes, de la production des sous-assemblages au lancement final. Les sous-assemblages subissent plusieurs sortes d'opérations telles que la préparation, l'intégration, le contrôle et le stockage. Ces opérations sont réparties entre plusieurs ateliers qui fonctionnent pour une partie en série, et l'autre partie en parallèle. Toutefois, ce processus présente plusieurs particularités. En effet, il intègre des contraintes de temps : chaque lancement doit être effectué selon un calendrier déterminé à l'avance. De plus, une partie de son architecture n'est pas fixée et est soumise à la décision du contrôleur. En effet, il se trouve un dépôt de stockage pour lequel il doit décider

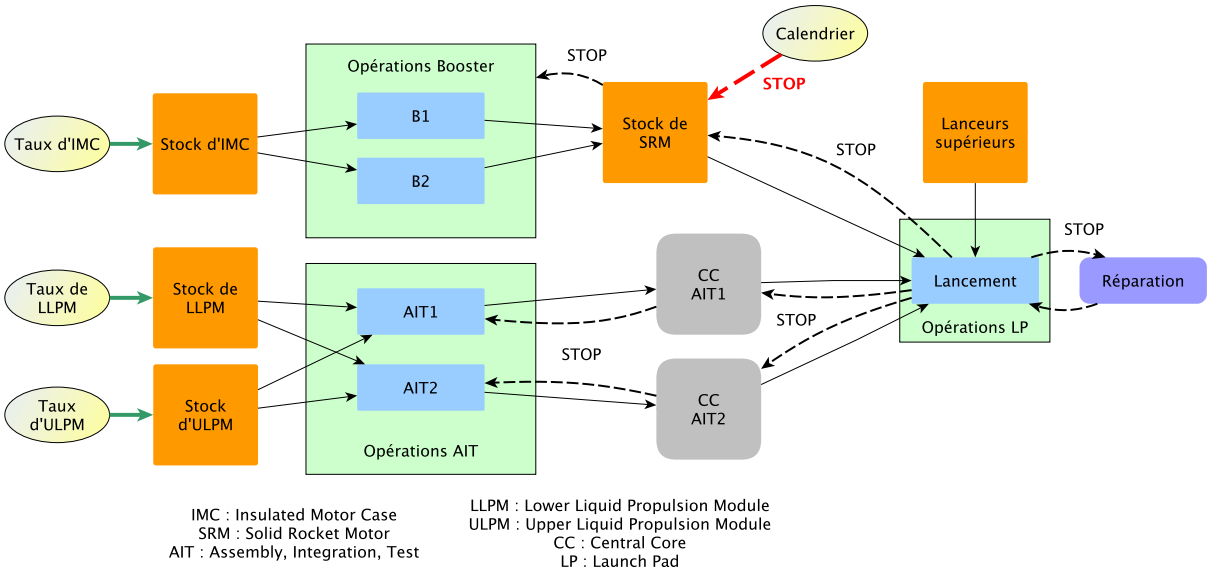

FIGURE 3.1 – Schéma du processus d'intégration

de sa capacité maximale avant toute mise en marche. L'optimisation portera sur cette capacité et sur les cadences de production des sous-assemblages de base, de sorte à minimiser les coûts d'exploitation. L'étude de cette chaîne de montage et son optimisation ont fait l'objet d'un exposé à la conférence ESREL qui a eu lieu à Zürich en septembre 2015 (cf. [NdSD+15]), ainsi que d'un article soumis à Journal of Risk and Reliability.

Nous consacrons ce chapitre à la description précise du processus d'intégration lanceur proposé. Nous partirons des composants de base d'un lanceur décrits dans la partie 3.1 pour expliquer les opérations qu'ils vont subir dans la partie 3.2. Les contraintes temporelles seront développées dans la partie 3.3. Les paramètres de contrôle seront listés dans la partie 3.4, puis enfin, dans la partie 3.5, nous définirons le problème d'optimisation associé à ce processus.

Pour un souci de clarté, la figure 3.1 schématise toutes les étapes successives que nous allons expliquer et répertorie toutes les abréviations que nous utiliserons.

3.1 Sous-assemblages

Un lanceur comporte deux parties : haute et basse. Des éléments de base, qu'on appelle sous-assemblages, s'intègrent dans les différents ateliers pour former chacune de ces parties qui s'assembleront entre elles. Ces sous-assemblages sont au nombre de quatre :

— les Insulated Motor Cases (IMC en abrégé), qui sont des propulseurs sans poudre et appartiennent au bas du lanceur,

— les Lower Liquid Propulsion Modules (LLPM) et les Upper Liquid Propulsion Modules (ULPM), qui forment la partie basse d'un lanceur,

— les lanceurs supérieurs, qui forment la coiffe (partie haute) d'un lanceur.

Les trois premiers sous-assemblages sont produits selon une cadence déterminée par le contrôleur. Quant aux lanceurs supérieurs, ils présentent la particularité d'être toujours disponibles lorsqu'on en a besoin. Pour cette raison, dans la modélisation et dans l'optimisation, on ne les prendra pas en compte.

Les durées de production des trois sous-assemblages IMC, LLPM et ULPM sont supposées aléatoires. Lorsque l'un d'entre eux est produit, il est transféré en temps négligeable dans l'entrepôt dédié dont la capacité est limitée à quatre unités. Ainsi, lorsqu'un stock est plein, la production du sous-assemblage correspondant est arrêtée, et reprend dès qu'une unité est destockée. Tant qu'on n'en a pas l'usage, le sous-assemblage attend dans son entrepôt.

Stocker des sous-assemblages a un coût. Le prix de stockage consiste en un montant journalier correspondant à une certaine proportion du prix de la pièce. Le tableau 3.1 récapitule les prix de stockage pour les trois sous-assemblages que l'on considère dans ce travail. Compte tenu des questions de confidentialité, tous les coûts de ce chapitre seront exprimés en pourcentages arbitraires d'une valeur de référence. Les proportions sont toutefois respectées.

Sous-assemblage	Coût de stockage par jour et par unité
IMC	2,6%
LLPM	55,94%
ULPM	35,59%

Tableau 3.1 – Coût de stockage pour chaque sous-assemblage de base

3.2 Ateliers

Le processus comprend diverses sortes d'opérations réparties en autant de catégories qu'il y a d'ateliers :

— les opérations Booster concernent l'intégration des propulseurs et leur chargement en poudre,

— les opérations d'Assemblage, d'Intégration et de Test (appelées opérations AIT dans la suite) concernent l'intégration de la partie basse du lanceur,

— les opérations de la zone de lancement (appelées opérations LP dans la suite) touchent à l'intégration du lanceur final, à son lancement effectif et à la maintenance du pas de tir.

Toutes ces opérations sont effectuées dans leur atelier dédié. Les ateliers des opérations Booster et AIT comprennent deux docks qui fonctionnent en parallèle. Ils peuvent donc accueillir deux sous-assemblages en même temps pour procéder à leur intégration. Cependant, la zone de lancement ne dispose que d'un dock. Les trois ateliers sont sujets à des pannes ou à des

problèmes manutentionnaires, ce qui implique que leur durée opératoire peut être considérée aléatoire. Décrivons maintenant ce qui se passe durant chaque opération. Les tableaux 3.2 et 3.3 répertorient les données chiffrées présentées dans cette partie. Notons que les durées nominales indiquées ne sont que des estimations issues d'expertises. Elles correspondent à une durée opératoire minimale. Par convention, dans cette partie et dans toute la suite, nous parlerons toujours de jours ouvrés, sauf mention explicite contraire. Nous considérons qu'il y a 261 jours ouvrés dans une année (365 jours auxquels nous avons retranché 52 week-ends).

3.2.1 Opérations Booster

Les opérations Booster débutent avec le déstockage d'un IMC. Ce dernier est acheminé vers un dock libre, s'il y en a un. La durée nominale d'intégration des IMC est de 5 jours. Le composant obtenu après intégration est appelé Solid Rocket Motor (SRM en abrégé). Le SRM est amené dans son stock. Ce dernier a une capacité de 4 ou 8 unités : il revient au contrôleur de décider laquelle est la moins coûteuse.

Lorsque la capacité du stock de SRM serait dépassée à la suite de l'intégration d'un IMC, les docks B1 et B2 (voir figure 3.1) ne sont plus autorisés à fonctionner et doivent attendre que des SRM quittent leur entrepôt. Cette situation se produit notamment lorsque leur stock est plein, ou qu'il ne reste de la place que pour une seule unité et qu'un IMC est déjà en cours d'intégration dans un dock.

Un lanceur nécessite 4 SRM : c'est donc exclusivement par groupes de quatre que les SRM quittent leur entrepôt pour être acheminés vers le pas de tir. Le coût de stockage des SRM est de 8,08% par unité et par jour.

3.2.2 Opérations AIT

Les opérations AIT nécessitent un LLPM et un ULPM. Lorsque les ressources sont suffisantes, ces deux sous-assemblages sont amenés ensemble vers un dock libre s'il y en a un. La durée nominale des opérations AIT est de 25 jours. Elles donnent en sortie un composant appelé Central Core (CC en abrégé), auquel viendront plus tard se greffer les SRM pour former la partie basse du lanceur.

Pour des raisons de sécurité, il n'y a pas de stock dédié pour les CC. Donc, lorsqu'un dock a terminé l'assemblage d'un CC, ce dernier doit y rester tant qu'on n'en a pas besoin dans la zone de lancement. Pendant ce temps, le dock est occupé et donc inutilisable pour toute autre intégration. Garder un CC dans son dock est considéré comme du stockage du point de vue financier, et coûte 100% par jour et par unité.

3.2.3 Opérations LP

Les opérations LP mettent en jeu trois types de composants : les Upper Launchers, les CC et les SRM. Plus précisément, il faut 1 Upper Launcher (qui est toujours disponible lorsqu'il le faut), 1 CC et 4 SRM. Lorsque les ressources sont suffisantes, elles sont acheminées vers le

pas de tir s'il est libre. Commence alors l'intégration du lanceur final, dont le lancement suit immédiatement après.

Toutefois, les opérations LP présentent une spécificité par rapport aux deux autres ateliers. En effet, elles ont la particularité de se dérouler en deux phases : une phase d'intégration et de lancement d'une part, et une phase de réparation d'autre part. La durée nominale de la première est de 10 jours. Une fois celle-ci terminée, le pas de tir a subi des dégradations suite au lancement qui vient d'être effectué. La phase de réparations s'enclenche alors. Elle dure 5 jours, de façon invariable. Il s'agit de la seule durée qui ne soit pas aléatoire. Bien qu'il n'y ait pas de lanceur présent sur le pas de tir à ce moment-là, la zone de lancement reste indisponible pour l'accueil de nouveaux composants et ces derniers doivent donc attendre la fin des réparations pour y être intégrés.

Composant	Coût de stockage par jour et par unité
SRM	8,08%
CC	100%

Tableau 3.2 – Coût de stockage pour les produits des opérations Booster et AIT

Opérations	Durée nominale en jours
Booster	5
AIT	25
LP (lancement)	10
LP (réparation)	5

Tableau 3.3 – Durées nominales des opérations dans les ateliers

3.3 Calendrier

Ce processus d'intégration comporte une contrainte temporelle sous la forme d'un calendrier de tirs. Chaque lancement doit être effectué à des dates prédéfinies imposées par les clients. Typiquement, le processus est conçu pour durer 30 ans. Un calendrier admissible doit respecter trois critères.

1. Les quatre premières années étant consacrées au démarrage du système, durant cette période, 18 tirs doivent être prévus et distribués de cette manière :
   — année 1 : 1 lancement,
   — année 2 : 2 lancements,
   — année 3 : 4 lancements,
   — année 4 : 11 lancements.

2. Pour chacune des 26 années suivantes, le nombre de lancements à effectuer doit se situer entre 6 et 12. Ils doivent être plus rapprochés en fin d'année qu'en début. Le nombre moyen de lancements à faire dans l'année doit être de 10.
3. Pour des raisons de sécurité, les lancements doivent être espacés d'au moins 15 jours.

Le calendrier a une action directe sur le processus dans le sens où, si le prochain tir à faire est trop loin dans le temps, le stock de SRM est bloqué de sorte qu'aucun SRM ne peut être acheminé vers la zone de lancement tant que la date du prochain tir n'est pas suffisamment proche. En effet, effectuer un tir en avance n'entraîne aucun gain. La notion de « suffisamment proche » est matérialisée par un seuil avant lequel le stock ne peut être débloqué. Étant donnée la durée nominale de lancement, la règle fixée est que le stock de SRM est débloqué à partir du 10^{e} jour avant la date du prochain tir.

À cause du caractère aléatoire des durées opératoires, des retards par rapport à la date prévue sont possibles. Ce cas de figure engendre un coût, dit « coût de retard ». En l'occurrence, on en distingue deux sortes.

— Si, lorsque l'intégration d'un nouveau lanceur démarre dans la zone de lancement, on sait qu'on va être en retard, il s'agit d'un retard dit anticipé. Ce scénario se produit notamment lorsque l'intégration commence moins de 10 jours avant la date de tir (on rappelle que la durée nominale des opérations LP est de 10 jours pour la première phase).

— Lorsque les opérations LP ont été mises en route en temps convenable (donc il n'y a pas d'anticipation d'un retard), mais que le lancement a quand même été effectué après la date prévue, il s'agit d'un retard dit a posteriori.

Les coûts de retard sont proportionnels à leur durée. Leurs estimations sont données dans le tableau 3.4.

Type de retard	Coût par jour
Anticipé	45,19%
A posteriori	80,13%

Tableau 3.4 – Coûts de retard

3.4 Décision

La description qui précède a fait apparaître quatre paramètres soumis à la décision d'un contrôleur extérieur.

1. Un paramètre *statique* : la capacité du dépôt de stockage des SRM. Elle doit être fixée pour toute la durée du processus avant sa mise en route. Elle est choisie parmi deux options : 4 ou 8 unités. La question consiste à choisir entre une capacité moindre, qui a un coût de stockage moins élevé mais qui peut ralentir le processus, et une capacité plus grande,

avec un coût d'exploitation supérieur mais qui permet de garder en réserve les ressources suffisantes pour 2 lanceurs, et donc qui est susceptible d'absorber certains retards. Puisque cette décision intervient avant que le processus ne démarre, il nous faudra comparer les performances obtenues dans les deux scénarios.

2. Trois paramètres *dynamiques* : les taux de production des trois sous-assemblages de base (IMC, LLPM et ULPM). Ils sont au cœur du problème d'optimisation puisque ce sont ces paramètres qui vont gouverner la vitesse du processus. Le contrôleur fixe les taux à la fréquence d'une fois par an. Ils sont choisis le premier jour pour toute l'année à venir. Ils se présentent sous la forme d'un nombre moyen de sous-assemblages produits en un an. L'ensemble des possibilités est décrite dans le tableau 3.5. Par exemple, si le contrôleur choisit de produire 40 IMC en moyenne par an, sachant qu'il y a 261 jours ouvrés dans une année, la durée moyenne de production sera de

$$\left\lfloor \frac{261}{40} \right\rfloor = 6 \text{ jours.}$$

Ainsi, ralentir la production peut s'avérer utile pour garder un niveau raisonnable des stocks et en limiter les coûts. Cependant, une production trop lente peut évidemment générer des retards.

Sous-assemblage	Nombre moyen produit à l'année
IMC	24, 28, 32, 36, 40, 44 ou 48
LLPM	6, 7, 8, 9, 10, 11 ou 12
ULPM	6, 7, 8, 9, 10, 11 ou 12

Tableau 3.5 – Plage des taux de production pour les trois sous-assemblages

3.5 Objectif

Les décisions prises pour les taux de production et pour l'architecture du processus engendrent des coûts. Ils sont calculés à la fin de chaque année. Le coût total est égal à la somme de tous ces coûts annuels. Il va s'agir de proposer les meilleurs choix pour les paramètres à décider afin non seulement de répondre à la demande du client (c'est-à-dire respecter le calendrier), mais aussi de minimiser le coût total. C'est tout à fait un problème de décision séquentielle dans l'incertain, ce que les processus markoviens décisionnels peuvent nous permettre de résoudre. Il nous faudra trouver une méthode de résolution numérique qui soit implémentable et qui rende une stratégie optimale explicite sous la forme d'un tableau qui donne l'action optimale à choisir en fonction de l'état et de l'année. Plusieurs outils sont à notre disposition, et la modélisation du problème va pouvoir nous éclairer sur celui qui est le plus judicieux.

4 Modélisation numérique et simulation

Le problème posé pour le processus d'intégration lanceur du chapitre 3 est un problème d'optimisation séquentielle dans l'incertain. Les principales méthodes étudiées dans la littérature pour y répondre nécessitent de connaître au moins deux éléments sous une forme analytique : la loi de transition et la fonction de coût. Une phase de modélisation de la chaîne de montage est donc nécessaire. Pour ce faire, plusieurs outils ont été développés. Dans les débuts de notre collaboration avec Airbus, nous avons travaillé sur un premier modèle ([EBBBS13]) en guise d'étude préliminaire, afin de développer certaines compétences de simulation. Tandis que leurs experts ont utilisé les réseaux de Petri pour les modéliser ([vdA94, Jen97]), nous avons orienté nos recherches vers les processus markoviens décisionnels ([HLL96, HLL99, BR11]).

S'agissant d'une classe de processus aléatoires contrôlés dont chaque état et chaque décision prise génère un coût, il nous a paru que ce cadre de travail était tout à fait adapté. Dès lors, la modélisation suppose une analyse complète du fonctionnement interne de la chaîne de montage pour identifier et préciser un espace d'états pour décrire le système, une loi de transition pour connaître la dynamique du système et une fonction de coût. Cette analyse a été effectuée en plusieurs étapes. Une réflexion sur le pas de temps à adopter nous a dirigés vers une manière bien particulière de concevoir la transition d'un état à un autre. Alors, nous avons pu dégager un schéma de modélisation numérique et donc une liste de variables utiles. Par la suite, nous avons implémenté un simulateur pour avoir à disposition des trajectoires précises.

Toutefois, ce chapitre préfigure une des plus grosses difficultés auxquelles nous allons faire face dans la phase d'optimisation. La complexité du système est en effet telle que malgré des hypothèses simplificatrices, l'espace d'états comprendra environ neuf milliards d'éléments. Cette constatation interdit toute procédure d'optimisation standard. Dans ce chapitre, nous commencerons à aborder cette question.

Nous découpons l'exposé en deux parties : d'une part nous détaillerons la modélisation numérique de la chaîne. Nous raisonnerons sur le pas de temps, puis en découleront la dynamique du système et les variables utiles. D'autre part, nous expliquerons comment nous avons simulé le processus à partir de la modélisation. Nous en profiterons pour parler de la manière dont nous avons implémenté les stratégies et la fonction de coût.

4.1 Modélisation numérique

4.1.1 Temps

Nous commençons par raisonner sur le pas de temps, car il va conditionner la dynamique du système, comme nous allons le voir. D'après le formalisme des processus markoviens décisionnels, la proposition naturelle pour le pas de temps est de le faire correspondre avec la fréquence des prises de décision, en l'occurrence ici une année. Nous pourrions ainsi considérer des processus à horizon fini de 30 ans, ce qui serait commode. En revanche, la difficulté qu'amène cette proposition réside dans la loi de transition d'un état à un autre. En effet, il faut déterminer tout ce qui se passe en une année pour calculer précisément l'état à l'année d'après. Alors, avec un pas de temps aussi large par rapport aux durées caractéristiques de la chaîne de montage, comment déterminer quelles opérations se sont succédées durant l'année ? Il s'agit là d'une tâche qui nous a semblé ne pouvoir être accomplie qu'en choisissant un autre pas de temps. Nous aurons par conséquent deux échelles de temps : une pour la simulation, et une pour l'optimisation.

L'analyse de la chaîne de montage amène à dégager des « événements », comme la production d'un sous-assemblage ou un lancement. À leur suite, ils déclenchent d'autres opérations et d'autres « événements ». Entre deux événements il ne se passe rien de pertinent d'un point de vue global. On en distingue un certain nombre qui sont :

1. la production d'un IMC,
2. la production d'un LLPM,
3. la production d'un ULPM,
4. la production d'un SRM par le dock B1 ou le dock B2,
5. la production d'un CC par le dock AIT1 ou le dock AIT2,
6. un lancement a été effectué ou bien la phase de réparations de la LP est terminée,
7. le calendrier autorise le déblocage du stock de SRM,
8. l'année courante est finie.

Ainsi, la transition du temps n au temps $n+1$ correspond au fait qu'un de ces événements s'est produit. Entre deux événements consécutifs, il n'y en a aucun autre. On va savoir décrire les conséquences de chacun de ces événements : il s'agit donc là d'une échelle de temps adaptée pour la simulation.

Il est possible de déterminer quel est le prochain événement à survenir parmi ceux-là en considérant certaines quantités particulières, comme par exemple un compte à rebours avant

qu'ils ne se produisent. Trouver le minimum parmi eux revient alors à déterminer le prochain événement.

Cependant, on ne peut se contenter d'un tel processus, puisqu'il ne rentre pas dans le formalisme classique (le pas de temps ne correspond pas à la fréquence des décisions). C'est la raison pour laquelle on a introduit l'événement « l'année courante est finie » : de cette manière, il sera aisé de ne garder que le premier état de chaque année. La succession de tels états forme le processus markovien décisionnel avec lequel nous allons travailler lors de la phase d'optimisation. On sera ainsi capables d'en simuler la transition grâce au pas de temps « événementiel ».

Soulignons également la nécessité d'une discrétisation du temps pour la simulation et pour l'optimisation. Afin de travailler avec un espace des états fini, nous avons, avec l'aide des experts d'Airbus, établi les hypothèses simplificatrices suivantes.

1. Le temps opératoire pour chacun des ateliers sera uniformément distibué entre deux ou trois valeurs proches de la durée nominale.
2. La durée opératoire de la production des sous-assemblages IMC, LLPM et ULPM sera distribuée selon une loi discrète sur un ensemble fini de valeurs centré autour de sa moyenne,
3. Le temps sera vu comme une succession de demi-journées.

La discussion avec les experts a abouti à l'établissement de lois pour la durée de production d'un sous-assemblage (cf. tableau 4.1, il s'agit d'une loi centrée autour de son espérance et d'écart-type 1,03) et pour la durée opératoire des ateliers (cf tableau 4.2).

Durée (en jours)	$T-2$	$T-1$	T	$T+1$	$T+2$
Probabilité	$\frac{3}{32}$	$\frac{5}{32}$	$\frac{1}{2}$	$\frac{5}{32}$	$\frac{3}{32}$

Tableau 4.1 – Loi de la durée de production des sous-assemblage pour un temps moyen de T jours

Atelier	Valeurs possibles
Docks Booster	5 ; 5,5
Docks AIT	25 ; 25,5 ; 26
Zone de lancement (LP)	10 ; 10,5

Tableau 4.2 – Valeurs sur lesquelles la durée opératoire de production dans les ateliers est uniformément distribuée

4.1.2 Variables

Afin de décrire le plus précisément possible l'état du système, nous avons besoin des variables suivantes. On appelle N_{tir} le nombre de tirs à faire en 30 ans. Le calendrier prendra alors la forme

d'un vecteur-ligne de taille N_{tir} dont la k^{e} composante représentera la date du k^{e} tir. On notera cette date `date_tir(k)` dans la suite. On appelle S_{SRM} la capacité maximale du stock de SRM.

- Les variables qui concernent la production des IMC sont
  - `tps_I` $\in \frac{1}{2}\mathbb{N} \cup \{+\infty\}$: temps restant de fabrication du prochain IMC,
  - `stock_I` $\in [\![0;4]\!]$: le nombre d'IMC en stock.
- Les variables qui concernent la production des LLPM sont
  - `tps_L` $\in \frac{1}{2}\mathbb{N} \cup \{+\infty\}$: temps restant de fabrication des LLPM,
  - `stock_L` $\in [\![0;4]\!]$: nombre de LLPM en stock.
- Les variables qui concernent la production des ULPM sont
  - `tps_U` $\in \frac{1}{2}\mathbb{N} \cup \{+\infty\}$: temps restant de fabrication des ULPM,
  - `stock_U` $\in [\![0;4]\!]$: nombre de ULPM en stock.
- Les variables qui concernent les opérations Booster sont
  - `stock_S` $\in [\![0;S_{\text{SRM}}]\!]$: stock de SRM (avec $S_{\text{SRM}} = 4$ ou 8),
  - `etat_B1` $\in \{0;1\}$: indique si le dock B1 est débloqué (0) ou pas (1),
  - `tps_B1` $\in [\![0;5{,}5]\!]_2 \cup \{+\infty\}$: donne le temps restant de fonctionnement du dock B1,
  - `etat_B2` $\in \{0;1\}$: indique si le dock B2 est débloqué ou pas,
  - `tps_B2` $\in [\![0;5{,}5]\!]_2 \cup \{+\infty\}$: temps restant de fonctionnement du dock B2.
- Les variables qui concernent les opérations AIT sont
  - `etat_AIT1` $\in \{0;1;2\}$: indique si le dock AIT1 est libre et à l'arrêt (0), occupé et en fonctionnement (1) ou occupé et à l'arrêt (2).
  - `tps_AIT1` $\in [\![0;26]\!]_2 \cup \{+\infty\}$: temps restant de fonctionnement du dock AIT1,
  - `etat_AIT2` $\in \{0;1;2\}$: indique si le dock AIT2 est libre et à l'arrêt, occupé et en fonctionnement ou occupé et à l'arrêt.
  - `tps_AIT2` $\in [\![0;26]\!]_2 \cup \{+\infty\}$: temps restant de fonctionnement du dock AIT2.
- Les variables qui concernent les opérations LP sont
  - `tps_lanc` $\in [\![0;10{,}5]\!]_2 \cup \{+\infty\}$: temps restant de lancement de la zone de lancement,
  - `tps_rep` $\in [\![0;5]\!] \cup \{+\infty\}$: temps restant de fonctionnement de la zone de lancement,
  - `nb_tirs` $\in [\![0;N_{\text{tir}}]\!]$: nombre de tirs effectués.
- Les variables qui concernent le calendrier des tirs sont
  - `tps_cal` $\in \frac{1}{2}\mathbb{N}$: temps restant avant la prochaine date de déblocage du stock de SRM,
  - `attente` $\in [\![0;N_{\text{tir}}]\!]$: file d'attente du calendrier en cas de retard,
  - `compteur` $\in [\![0;N_{\text{tir}}]\!]$: variable qui compte le nombre de déblocages.
- On a enfin trois variables générales
  - `tps_ppe` $\in \frac{1}{2}\mathbb{N}$: temps propre du système,

— `tps_an` $\in [\![0; 261]\!]_2$: temps restant avant que l'année en cours ne soit écoulée,

— `an` $\in [\![1; 30]\!]$: numéro de l'année en cours.

Dans toute la suite, on note μ_T la loi de probabilité discrète sur $[\![T-2; T+2]\!]$ décrite par le tableau 4.1.

Chaque année, il faudra décider des cadences de production des sous-assemblages IMC, LLPM et ULPM. On représente ces décisions par trois suites $(\theta_I^k)_{1\leq k\leq 30}$, $(\theta_L^k)_{1\leq k\leq 30}$ et $(\theta_U^k)_{1\leq k\leq 30}$ où, pour chaque $k \in [\![1; 30]\!]$,

— θ_I^k représente le nombre moyen d'IMC produits en une année pour la k^{e} année,

— θ_L^k représente le nombre moyen de LLPM produits en une année pour la k^{e} année,

— θ_U^k représente le nombre moyen d'ULPM produits en une année pour la k^{e} année.

Dans la suite, on choisit de représenter l'affectation à une variable d'une valeur ou d'une réalisation d'une loi de probabilité par une flèche pointée vers le nom de ladite variable.

À l'état initial, nous avons :

— `tps_B1`, `tps_B2`, `tps_AIT1`, `tps_AIT2`, `tps_lanc` et `tps_rep` $\leftarrow +\infty$,

— `tps_I` $\leftarrow \mu_{\lfloor 261/\theta_I^1 \rfloor}$,

— `tps_L` $\leftarrow \mu_{\lfloor 261/\theta_L^1 \rfloor}$,

— `tps_U` $\leftarrow \mu_{\lfloor 261/\theta_U^1 \rfloor}$,

— `tps_cal` $\leftarrow \max(\texttt{date\_tir(1)} - 10; 0)$,

— `tps_an` $\leftarrow 261$,

— `an` $\leftarrow 1$,

— toutes les autres variables $\leftarrow 0$.

Remarque. Pour les variables temporelles qui correspondent à des comptes à rebours, nous avons évoqué la possibilité de leur affecter la valeur $+\infty$. Il s'agit d'un moyen commode, comme on va le voir, de ne pas les prendre en compte lors de la détermination du prochain événement.

4.1.3 Dynamique

Ici, nous allons rentrer dans le détail de la transition d'un état à un autre via le système d'événements que nous avons exposé avant. Il va s'agir d'expliciter très précisément ce qui se passe lors de chacun des événements que l'on a listés.

Dans un premier temps, on détermine lequel des événements sera le prochain à se produire. On dispose, pour chacun des trois ateliers et des trois sous-assemblages, des durées restantes avant la prochaine production, et on dispose de toutes les variables de calendrier nécessaires (i.e. `tps_I`, `tps_L`, `tps_U`, `tps_B1`, `tps_B2`, `tps_AIT1`, `tps_AIT2`, `tps_lanc`, `tps_rep`, `tps_cal` et `tps_an`). Il ne reste qu'à en déterminer le minimum pour trouver l'événement à venir. On déduit alors la valeur de ce minimum des autres durées, et on incrémente `tps_ppe` de cette même

valeur. C'est là que les valeurs $+\infty$ se révéleront utiles pour les événements qui ne peuvent pas arriver.

Expliquons maintenant plus en détail ce qui se passe lors de chaque événement. Une flèche pointée vers la droite dénote une transformation de la variable correspondante : par exemple `variable` $\rightarrow +1$ signifie qu'on ajoute 1 à `variable`.

1. **Autorisation d'envoi des SRM dans la zone de lancement.**
   (a) `compteur` $\rightarrow +1$,
   (b) Si `compteur` $+ 1 \leq N_{\text{tir}}$:
      i. `tps_cal` $\leftarrow \max(\texttt{date\_tir(compteur + 1)} - (10 + \texttt{tps\_ppe}); 0)$.
   (c) Sinon `tps_cal` $\leftarrow +\infty$.
   (d) Si `etat_AIT1` $= 2$ ET `stock_S` ≥ 2 ET `tps_lanc` $= +\infty$ ET `tps_rep` $= +\infty$:
      i. `stock_S` $\rightarrow -2$,
      ii. `etat_AIT1` $\leftarrow 0$ (en position « libre et à l'arrêt »),
      iii. `tps_lanc` $\leftarrow \mathcal{U}(\{10; 10{,}5\})$.
      iv. Si `stock_L` ≥ 1 ET `stock_U` ≥ 1 :
         A. `stock_L` $\rightarrow -1$,
            — Si `tps_L` $= +\infty$, `tps_L` $\leftarrow \mu_{\lfloor 261/\theta_L^{\texttt{an}} \rfloor}$,
         B. `stock_U` $\rightarrow -1$,
            — Si `tps_U` $= +\infty$, `tps_U` $\leftarrow \mu_{\lfloor 261/\theta_U^{\texttt{an}} \rfloor}$,
         C. `etat_AIT1` $\leftarrow 1$ (en position « occupé et en fonctionnement »),
         D. `tps_AIT1` $\leftarrow \mathcal{U}(\{15; 15{,}5; 16\})$.
      v. Si `stock_I` ≥ 1 ET `etat_B1` $= 0$ ET (`stock_S` $\leq S_{\text{SRM}} - 2$ si le dock B2 est bloqué OU sinon `stock_S` $\leq S_{\text{SRM}} - 1$) :
         A. `stock_I` $\rightarrow -1$,
            — Si `tps_I` $= +\infty$, `tps_I` $\leftarrow \mu_{\lfloor 261/\theta_I^{\texttt{an}} \rfloor}$,
         B. `etat_B1` $\leftarrow 1$ (en position « bloqué »),
         C. `tps_B1` $\leftarrow \mathcal{U}(\{9; 9{,}5\})$.
      vi. Si `stock_I` ≥ 1 ET `etat_B2` $= 0$ ET (`stock_S` $\leq S_{\text{SRM}} - 2$ si le dock B1 est bloqué OU sinon `stock_S` $\leq S_{\text{SRM}} - 1$) :
         A. `stock_I` $\rightarrow -1$,
         B. `etat_B2` $\leftarrow 1$ (en position « bloqué »),
         C. `tps_B2` $\leftarrow \mathcal{U}(\{9; 9{,}5\})$.
   (e) Sinon, si `etat_AIT2` $= 2$ ET `stock_S` ≥ 2 ET `tps_lanc` $= +\infty$ ET `tps_rep` $= +\infty$:
      i. `stock_S` $\rightarrow -2$,

ii. `etat_AIT2` $\leftarrow 0$ (en position « libre et à l'arrêt »),

iii. `tps_lanc` $\leftarrow \mathcal{U}(\{10; 10{,}5\})$.

iv. Si `stock_L` ≥ 1 ET `stock_U` ≥ 1 :

A. `stock_L` $\rightarrow -1$,

— Si `tps_L` $= +\infty$, `tps_L` $\leftarrow \mu_{\lfloor 261/\theta_L^{\mathtt{an}} \rfloor}$,

B. `stock_U` $\rightarrow -1$,

— Si `tps_U` $= +\infty$, `tps_U` $\leftarrow \mu_{\lfloor 261/\theta_U^{\mathtt{an}} \rfloor}$,

C. `etat_AIT2` $\leftarrow 1$ (en position « occupé et en fonctionnement »),

D. `tps_AIT2` $\leftarrow \mathcal{U}(\{15; 15{,}5; 16\})$.

v. On passe au point 1.(d).v.

(f) Sinon :

i. `attente` $\rightarrow +1$.

(g) STOP.

Lorsque le calendrier autorise l'envoi des SRM dans la zone de lancement, on vérifie qu'on peut préparer un lancer et on procède au calcul de la nouvelle durée avant la prochaine autorisation. Si un lancer peut se faire, le stock de SRM a été vidé et un dock AIT s'est libéré : il faut donc vérifier si les docks B et AIT peuvent être mis en marche. Si un lancer ne peut être fait, alors on incrémente la file d'attente de 1.
Le recalcul de la durée avant la prochaine autorisation nécessite de savoir par rapport à quelle date calendaire se baser. C'est toute l'utilité de la variable `compteur`, à laquelle on ajoute 1 pour avoir la prochaine date. Si le numéro ainsi obtenu dépasse le nombre de tirs N_{tir} prévus, c'est que tout le calendrier a été passé en revue et donc il n'y a plus d'autorisation à donner dans la suite. Une fois la bonne date de tir trouvée, on lui retranche `tps_ppe` et 10 pour avoir le nombre de jours avant la prochaine autorisation.

2. **La zone de lancement a fini d'être réparée.**

(a) `tps_rep` $\leftarrow +\infty$.

(b) Si `stock_S` ≥ 2 ET `etat_AIT1` $= 2$ ET `attente` ≥ 1 :

i. `etat_AIT1` $\leftarrow 0$ (en position « libre et à l'arrêt »),

ii. `attente` $\rightarrow -1$,

iii. `stock_S` $\rightarrow -2$,

iv. `tps_lanc` $\leftarrow \mathcal{U}(\{10; 10{,}5\})$,

v. on passe au point 1.(d).iv.

(c) Sinon, si `stock_S` ≥ 2 ET `etat_AIT2` $= 2$ ET `attente` ≥ 1 :

i. `etat_AIT2` $\leftarrow 0$ (en position « libre et à l'arrêt »),

ii. `attente` $\rightarrow -1$,

iii. `stock_S` $\rightarrow -2$,

iv. `tps_lanc` $\leftarrow \mathcal{U}(\{10; 10{,}5\})$,

v. on passe au point 1.(e).iv.

(d) STOP.

Lorsque la phase de réparation de la zone de lancement est achevée, alors il est possible qu'elle puisse être remise en lancement. Si c'est le cas, un dock AIT est alors libéré et le stock de SRM est alors vidé, donc un dock B et un dock AIT peuvent être remis en fonctionnement.

3. **La zone de lancement a procédé à un lancement.**

(a) `nb_tirs` $\rightarrow +1$.

(b) `tps_lanc` $\leftarrow +\infty$.

(c) `tps_rep` $\leftarrow 5$.

(d) STOP.

Lorsque la zone de lancement a lancé, on incrémente le compteur de tir et on enclenche la phase de réparations.

4. **Le dock Bj a produit.** ($j \in \{1; 2\}$)

(a) `etat_Bj` $\leftarrow 0$ (en position « débloqué »).

(b) `tps_Bj` $\leftarrow +\infty$.

(c) Si `etat_AIT1` $= 2$ ET `stock_S` ≥ 1 ET `tps_lanc` $= +\infty$ ET `tps_rep` $= +\infty$ ET `attente` ≥ 1 :

i. `attente` $\rightarrow -1$,

ii. `stock_S` $\rightarrow -1$,

iii. `etat_AIT1` $\leftarrow 0$ (en position « libre et à l'arrêt »),

iv. `tps_lanc` $\leftarrow \mathcal{U}(\{10; 10{,}5\})$,

v. on passe au point 1.(d).iv.

(d) Sinon, si `etat_AIT2` $= 2$ ET `stock_S` ≥ 1 ET `tps_lanc` $= +\infty$ ET `tps_rep` $= +\infty$ ET `attente` ≥ 1.

i. `attente` $\rightarrow -1$,

ii. `stock_S` $\rightarrow -1$,

iii. `etat_AIT2` $\leftarrow 0$ (en position « libre et à l'arrêt »),

iv. `tps_lanc` $\leftarrow \mathcal{U}(\{10; 10{,}5\})$,

v. on passe au point 1.(e).iv.

(e) Sinon :

i. `stock_S` $\rightarrow +1$,

ii. on passe au point 1.(d).v.

(f) STOP.

Lorsqu'un dock B a produit, il se débloque et les opérations LP peuvent être déclenchées. Dans ce cas, un dock AIT et un dock B peuvent être remis en marche. Si ce n'est pas possible, le SRM produit part dans son stock et comme un dock B a été débloqué, il peut être remis en marche.

5. **Le dock AITj a produit.** ($j \in \{1;2\}$)

(a) `tps_AITj` $\leftarrow +\infty$.

(b) Si `stock_S` ≥ 2 ET `tps_lanc` $= +\infty$ ET `tps_rep` $= +\infty$ ET `attente` ≥ 1 :

i. `attente` $\rightarrow -1$,

ii. `etat_AITj` $\leftarrow 0$ (en position « libre et à l'arrêt »),

iii. `stock_S` $\rightarrow -2$,

iv. `tps_lanc` $\leftarrow \mathcal{U}(\{10;10{,}5\})$,

v. on passe au point 1.(d).iv. si $j = 1$ ou au point 1.(e).iv. si $j = 2$.

(c) Sinon : `etat_AITj` $\leftarrow 2$ (en position « occupé et à l'arrêt »).

(d) STOP.

Lorsqu'un dock AIT a produit, si les opérations LP peuvent être déclenchées, alors le CC y est envoyé et le dock qui l'a produit devient libre et à l'arrêt. Si ce n'est pas possible, le CC reste dans son dock qui est à l'arrêt mais toujours occupé.

6. **Un IMC arrive.**

(a) Si `etat_B1` $= 0$ ET (`stock_S` $\leq S_{\text{SRM}} - 2$ si le dock B2 est bloqué OU sinon `stock_S` $\leq S_{\text{SRM}} - 1$) :

i. `etat_B1` $\leftarrow 1$ (en position « bloqué »),

ii. `tps_B1` $\leftarrow \mathcal{U}(\{9;9{,}5\})$.

(b) Sinon, si `etat_B2` $= 0$ ET (`stock_S` $\leq S_{\text{SRM}} - 2$ si le dock B1 est bloqué OU sinon `stock_S` $\leq S_{\text{SRM}} - 1$) :

i. `etat_B2` $\leftarrow 1$ (en position « bloqué »),

ii. `tps_B2` $\leftarrow \mathcal{U}(\{9;9{,}5\})$.

(c) Sinon : `stock_I` $\rightarrow +1$.

(d) Si `stock_I` $= 4$, `tps_I` $\leftarrow +\infty$.

(e) Sinon, `tps_I` $\leftarrow \mu_{\lfloor 261/\theta_I^{\texttt{an}} \rfloor}$.

(f) STOP.

Lorsqu'un IMC arrive, on remet le temps de production de ce sous-assemblage à sa valeur initiale et on vérifie si un dock B peut être déclenché à ce moment. Si ce n'est pas possible, on l'envoie au stock.

7. **Un LLPM arrive.**
   (a) Si $\texttt{etat\_AIT1} = 0$ ET $\texttt{stock\_U} \geq 1$:
      i. $\texttt{stock\_U} \rightarrow -1$,
      ii. $\texttt{etat\_AIT1} \leftarrow 1$ (en position « occupé et en fonctionnement »),
      iii. $\texttt{tps\_AIT1} \leftarrow \mathcal{U}(\{15; 15{,}5; 16\})$,
   (b) Sinon, si $\texttt{etat\_AIT2} = 0$ ET $\texttt{stock\_U} \geq 1$:
      i. $\texttt{stock\_U} \rightarrow -1$,
      ii. $\texttt{etat\_AIT2} \leftarrow 1$ (en position « occupé et en fonctionnement »),
      iii. $\texttt{tps\_AIT2} \leftarrow \mathcal{U}(\{15; 15{,}5; 16\})$,
   (c) Sinon : $\texttt{stock\_L} \rightarrow +1$.
   (d) Si $\texttt{stock\_L} = 4$, $\texttt{tps\_L} \leftarrow +\infty$.
   (e) Sinon, $\texttt{tps\_L} \leftarrow \mu_{\lfloor 261/\theta_L^{\texttt{an}} \rfloor}$.
   (f) STOP.

Lorsqu'un LLPM arrive, on remet le temps de production de ce sous-assemblage à sa valeur initiale et on vérifie si un dock AIT peut être mis en marche. Si ce n'est pas le cas, le LLPM part au stock.

8. **Un ULPM arrive.**
   (a) Si $\texttt{etat\_AIT1} = 0$ ET $\texttt{stock\_L} \geq 1$:
      i. $\texttt{stock\_L} \rightarrow -1$,
      ii. $\texttt{etat\_AIT1} \leftarrow 1$ (en position « occupé et en fonctionnement »),
      iii. $\texttt{tps\_AIT1} \leftarrow \mathcal{U}(\{15; 15{,}5; 16\})$,
   (b) Sinon, si $\texttt{etat\_AIT2} = 0$ ET $\texttt{stock\_L} \geq 1$:
      i. $\texttt{stock\_L} \rightarrow -1$,
      ii. $\texttt{etat\_AIT2} \leftarrow 1$ (en position « occupé et en fonctionnement »),
      iii. $\texttt{tps\_AIT2} \leftarrow \mathcal{U}(\{15; 15{,}5; 16\})$,
   (c) Sinon : $\texttt{stock\_U} \rightarrow +1$.
   (d) Si $\texttt{stock\_U} = 4$, $\texttt{tps\_U} \leftarrow +\infty$.
   (e) Sinon, $\texttt{tps\_U} \leftarrow \mu_{\lfloor 261/\theta_U^{\texttt{an}} \rfloor}$.
   (f) STOP.

Lorsqu'un ULPM arrive, on remet le temps de production de ce sous-assemblage à sa valeur initiale et on vérifie si un dock AIT peut être mis en marche. Si ce n'est pas le cas, l'ULPM part au stock.

9. **Une année s'est écoulée**
   (a) `an` $\rightarrow +1$,
   (b) `tps_an` $\leftarrow 261$,
   (c) STOP.

   Lorsqu'une année s'est écoulée, on incrémente le compteur d'années et on initialise le temps restant à 261 jours.

On trouvera sur la figure 4.1 un schéma récapitulatif de l'enchaînement des opérations.

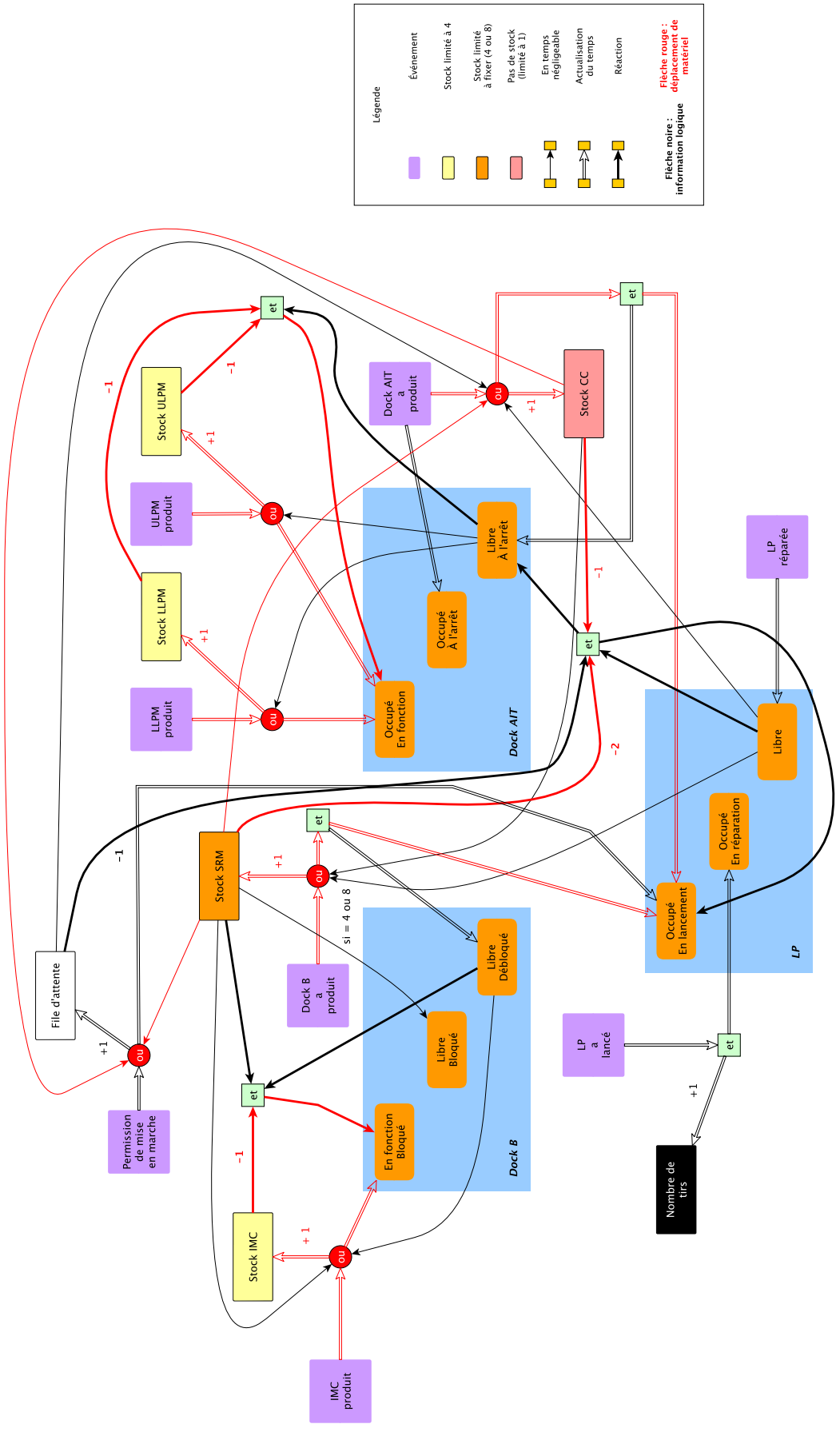

FIGURE 4.1 – Schéma de la modélisation de la chaîne de montage

4.2 Simulation numérique

Maintenant que nous avons décrit la dynamique du processus, nous allons exposer les principes de son implémentation. Nous isolerons d'abord les groupes d'instructions similaires qui reviennent souvent (les « routines »), puis nous expliquerons comment nous avons mis en place le calcul du coût et les stratégies afin de coder la dynamique de la transition du processus.

4.2.1 Routines

Dans la modélisation donnée à la partie précédente, on observe qu'un certain nombre d'instructions très similaires se répètent. On veut donc créer des routines qui se chargeraient de ces opérations. Seulement, ces opérations ne sont pas parfaitement identiques selon l'événement qui s'est produit et si on veut utiliser des routines, il faudra changer un peu la modélisation de sorte à les rendre identiques.

Par exemple, prenons les points 1.(d).v. et 6.(a). Ils concernent la mise en marche du dock B1. Les opérations communes sont

1. `etat_B1` $\leftarrow 1$,
2. `tps_B1` $\leftarrow \mathcal{U}(\{9;9{,}5\})$.

Seulement, le point 1.(d).v. comporte une opération supplémentaire et ce point est appliqué plusieurs fois ailleurs, notamment après le point 1.(d).iv. Cette opération est celle qui consiste à enlever une unité au stock d'IMC. On voudrait rajouter cette opération au point 6.(a). Autrement dit, on voudrait appliquer la routine suivante :

1. `stock_I` $\rightarrow -1$,
2. `etat_B1` $\leftarrow 1$,
3. `tps_B1` $\leftarrow \mathcal{U}(\{9;9{,}5\})$.

Pour retrouver le même résultat, il faut donc avoir auparavant ajouté une unité au stock d'IMC. Le point 6 sera donc ainsi modifié :

1. `stock_I` $\rightarrow +1$.
2. Si `etat_B1` $= 0$ ET (`stock_S` $\leq S_{\text{SRM}} - 2$ si le dock B2 est bloqué OU sinon `stock_S` $\leq S_{\text{SRM}} - 1$) :
   (a) `stock_I` $\rightarrow -1$...
3. Sinon si `etat_B2` $= 0$ ET (`stock_S` $\leq S_{\text{SRM}} - 2$ si le dock B2 est bloqué OU sinon `stock_S` $\leq S_{\text{SRM}} - 1$) :
   (a) `stock_I` $\rightarrow -1$...
4. Si `stock_I` $= 4$...

On trouve d'autres séries d'opérations similaires pour lesquelles le principe va être le même. Pour ces séries d'opérations, il va s'agir d'ajouter des instructions d'incrémentation ou de décrémentation des stocks.

Donc, les routines que l'on veut mettre en place concernent la mise en route des docks B, des docks AIT et de la zone de lancement. On veut donc créer trois fonctions, une pour chacun des ateliers. Nous allons également implémenter une routine pour calculer combien de temps il reste avant le prochain lancement.

— La fonction qui met en marche des docks B si les conditions sont remplies sera appelée `verifB`.

— Celle qui s'occupe des docks AIT sera appelée `verifAIT`.

— Celle qui déclenche les opérations LP sera appelée `verifLP`. Pour cette fonction, nous avons rassemblé les deux durées `tps_lanc` et `tps_rep` en une seule, notée `tps_LP` et qui est à valeurs dans $[\![0;10{,}5]\!]_2 \cup \{+\infty\}$ (cf. annexe A).

— Pour calculer automatiquement le compte à rebours `tps_cal` avant le prochain lancement, on crée la routine `veriftemps`.

Les codes de ces routines sont donnés dans l'annexe A, section A.1.1.

4.2.2 Fonction de coût

Le processus que l'on considère engendre des coûts à chaque transition. Ces coûts sont de deux natures : coûts de stockage et coûts de retard. Il s'agit de coûts journaliers, ce qui signifie que les variables temporelles vont être mises à contribution et qu'il faut déterminer à quel moment précisément le coût doit être calculé.

Coût de stockage

Cinq sous-assemblages sont concernés par les coûts de stockage : les IMC, les LLPM, les ULPM, les SRM et les CC. Stocker chacun de ces éléments va coûter, par jour, un certain pourcentage du prix de la pièce. Donc, pour chaque pièce entreposée au cours du processus, il faut savoir le temps exact passé dans son stock.

Chaque pièce rentre dans son stock et en sort lors d'un des événements explicités à la partie précédente. Le coût engendré par le stockage de cette pièce est donc proportionnel au temps écoulé entre l'événement qui a vu son entrée dans le stock et celui qui a permis sa sortie. Le schéma de la figure 4.2 illustre ce principe et nous permet de voir que le coût de stockage de chaque pièce est égal à la somme des coûts de stockage calculés entre chaque événement depuis l'entrée de la pièce dans la chaîne jusqu'à sa sortie.

Soient :

— c_I le coût de stockage par jour des IMC,

— c_L le coût de stockage par jour des LLPM,

— c_U le coût de stockage par jour des ULPM,

— c_S le coût de stockage par jour des SRM,

— c_C le coût de stockage par jour des CC.

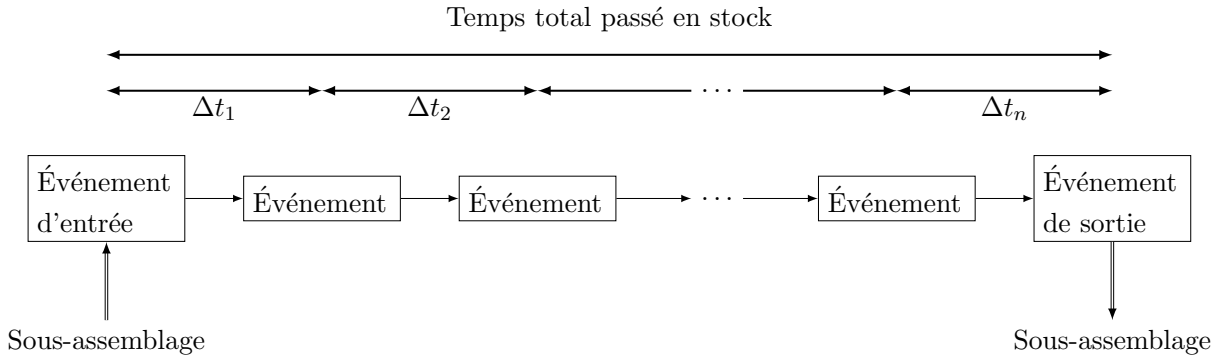

FIGURE 4.2 – Calcul du coût de stockage pour un sous-assemblage

Lors de chaque événement, il faut utiliser la valeur que l'on incrémente au temps propre (qui est la différence entre le temps propre actuel et le temps propre précédent). On la note Δt. Ainsi, on a accès au coût de stockage de chacune des pièces en stock via l'opération

$$[c_I \texttt{stock\_I} + c_L \texttt{stock\_L} + c_U \texttt{stock\_U} + c_S \texttt{stock\_S} + c_C(\delta_2(\texttt{etat\_AIT1}) + \delta_2(\texttt{etat\_AIT2}))] \times \Delta t.$$

Notons que ce calcul doit se faire avant d'appliquer les instructions relatives à l'événement qui s'est produit, de sorte à prendre en compte les pièces qui, éventuellement, seraient destockées.

Coût de retard anticipé

Le coût de retard anticipé est ce qu'il faut payer lorsqu'au moment où on met en marche la zone de lancement, on sait qu'on sera en retard sur la date calendaire. En effet, le lancer dure au moins 10 jours. Un retard sera donc anticipé lorsque la date calendaire se situe à ce moment-là à moins de 10 jours. Soit c_{RA} le coût par jour du retard anticipé. Le calcul de ce coût se fait donc via la formule

$$c_{RA}(\texttt{tps\_ppe} + \texttt{tps\_LP} - \texttt{date\_tir(nb\_tirs + 1)})_+.$$

Ce coût doit, par définition, être calculé au moment de la mise en marche de la LP après l'affectation de la variable `tps_LP`.

Coût de retard a posteriori

Le coût de retard a posteriori est ce qu'il faut payer lorsqu'un tir a été effectué en retard, mais que ce retard n'a pas été anticipé (c'est-à-dire que la zone de lancement a été mise en marche au moins 10 jours avant la date calendaire de tir). Soit c_{RP} le coût journalier de retard a posteriori. Étant donné qu'au moment de la mise en marche de la zone de lancement, on a accès à la valeur de la durée nécessaire pour effectuer le tir, on sait alors dire s'il va être effectué en retard. Seulement, il faut vérifier que le retard n'est pas anticipé, ce qu'on sait dire aussi (cf. paragraphe précédent). Le calcul du coût de retard a posteriori se fait donc via la formule

$$c_{RP}(\texttt{tps\_ppe} + \texttt{tps\_LP} - \texttt{date\_tir(nb\_tirs + 1)})_+,$$

ce coût étant nul si le retard a été anticipé, autrement dit si tir est commencé à moins de 10 jours de la date calendaire. Le coût de retard a posteriori doit donc être calculé lors de la mise en marche de la LP après l'affectation de la variable `tps_LP`.

Simulation

On introduit alors une nouvelle variable `cout` qui va servir à accumuler les coûts générés lors des transitions. À chaque itération, on regarde quelle valeur parmi `tps_I`, `tps_L`, `tps_U`, `tps_B1`, `tps_B2`, `tps_AIT1`, `tps_AIT2`, `tps_LP`, `tps_cal` et `tps_an` est minimale. On retranche ce minimum aux autres valeurs. Puis on incrémente `tps_ppe` du minimum trouvé, puis on calcule le coût de stockage grâce à ce minimum. On ajoute sa valeur à `cout`. Donc seule la fonction `verifLP` se trouve affectée par la présence des variables de coût. On trouvera son code dans l'annexe A, section A.1.1.

Pour les besoins de l'optimisation, on instaurera une pénalité financière par tir non effectué en fin de processus (qui est donc calculé lorsque les 30 années sont écoulées) que l'on rajoute à `cout` à la toute fin. Une fois le processus terminé, il est en effet facile de récupérer le nombre de tirs qui n'ont pas pu être faits.

4.2.3 Application d'une stratégie et première réduction de l'espace d'états

Les cadences de production des sous-assemblages sont déterminées par les trois suites de taux $(\theta_I^k)_{1\leq k\leq 30}$, $(\theta_L^k)_{1\leq k\leq 30}$ et $(\theta_U^k)_{1\leq k\leq 30}$. Ces taux ne sont pas déterminés à l'avance : ils sont fonction de l'état dans lequel se trouve le système au moment où on les choisit. Mais la trajectoire de la chaîne va dépendre de la stratégie choisie, c'est-à-dire d'une fonction qui à un état de la chaîne associe une décision (c'est donc une stratégie déterministe markovienne).

Tel que nous l'avons modélisé, le processus possède un espace d'états trop important pour la phase d'optimisation qui va suivre. En effet, on estime son cardinal à environ neuf milliards. Il faut donc trouver un moyen de réduire le nombre de variables d'états dont dépend la décision. On peut remarquer que physiquement, un contrôleur n'a pas accès à toutes les variables descriptives que l'on a mises en évidence lors de la description de la chaîne de montage. Seules certaines sont directement observables [1]. Ce sont :

- l'état des stocks : `stock_I`, `stock_L`, `stock_U` et `stock_S`,
- l'état des ateliers : `etat_B1`, `etat_B2`, `etat_AIT1`, `etat_AIT2` et `etat_LP`,
- le nombre de tirs effectués `nb_tirs`,
- les variables calendaires (`tps_cal`, `attente` et `compteur`) et les variables générales (`tps_ppe`, `tps_an` et `an`).

On définit également une autre variable observable, appelée `tirs_prevus`, qui compte le nombre de tirs à effectuer dans l'année. Elle est à valeurs dans $[\![1;17]\!]$. Comme deux tirs doivent être espacés d'au moins 15 jours, on ne peut faire qu'au plus 17 tirs en un an.

1. On pourrait finalement voir ce processus comme un POMDP. La complexité du problème n'en serait en revanche pas réduite via ce formalisme.

Dans la suite, on appellera également `stock_C` la variable à valeurs dans $\{0; 2\}$ qui mesure le nombre de CC en stock (autrement dit, $\texttt{stock\_C} = \delta_2(\texttt{etat\_AIT1}) + \delta_2(\texttt{etat\_AIT2})$).

Certaines de ces variables ne sont pas raisonnablement pertinentes pour l'établissement d'un taux de production. On modélise l'information « physique » utile au contrôleur par les six variables suivantes :

1. `tirs_prevus`,
2. `stock_I`,
3. `stock_U`,
4. `stock_L`,
5. `stock_S`,
6. `stock_C`.

On réduit ainsi l'état du système à ces six variables d'intérêt. Le nombre d'états réduits est noté $\#\mathbb{X}$. Selon que l'on autorise un stock de 4 SRM ou de 8 SRM, $\#\mathbb{X}$ vaut respectivement 26 251 ou 47 251. On ordonne l'ensemble de ces états d'une manière arbitraire et on les range dans une matrice appelée `matrice_etats` de taille $7 \times \#\mathbb{X}$, où chacune des six premières lignes, servant à la lisibilité de la matrice, correspond aux valeurs des variables citées au-dessus. Ainsi, les états réduits se retrouvent numérotés de 1 à $\#\mathbb{X}$. La septième ligne sert pour les besoins de la simulation : elle donne, pour chacun des états, un code unique en forme de nombre entier, afin de les retrouver plus facilement dans la matrice. On instaure alors une nouvelle variable `numero_etat` à valeurs dans $[\![0; \#\mathbb{X}]\!]$ qui donne le numéro de l'état réduit dans lequel se trouve le système à l'année nouvelle (0 étant mis par convention pour l'état terminal).

On peut faire de même avec les différentes décisions possibles. Chaque année, on décide d'un taux de production pour les trois sous-assemblages, c'est-à-dire d'un nombre moyen de pièces à produire en un an. On note $\#\mathbb{A}$ le nombre total de décisions possibles, ce qui donne $\#\mathbb{A} = 7^3 = 343$. On ordonne ainsi arbitrairement les décisions possibles et on les range dans une matrice appelée `matrice_actions` de taille $3 \times \#\mathbb{A}$, où chaque ligne correspond au taux d'un des trois sous-assemblages :

1. la première ligne correspond aux IMC,
2. la deuxième ligne correspond aux LLPM,
3. la troisième ligne correspond aux ULPM.

Ainsi, les décisions possibles se retrouvent numérotées de 1 à $\#\mathbb{A}$.

On prend une décision au début de chacune des 30 années que dure le processus. On peut alors modéliser une stratégie par une matrice $(m_{i,j})$ de taille $\#\mathbb{X} \times 30$ à valeurs dans $[\![1; \#\mathbb{A}]\!]$ telle que le cœfficient $m_{i,j}$ est le numéro de la décision à prendre la j^{e} année si le système se trouve dans l'état numéro i à ce moment-là.

C'est donc lorsqu'une année s'est écoulée qu'il faut chercher le numéro de l'état et calculer le nombre de tirs prévus. Mais il faut distinguer le cas où le processus a terminé et entre donc dans sa 31$^{\text{e}}$ année. Dans ce cas, il faut calculer le nombre de tirs qui n'ont pas été effectués (i.e. `compteur` − `nb_tirs`). Sinon, on aura besoin du nombre de tirs prévus dans l'année par le calendrier, qu'on note `tirs_calendrier(k)` pour la k^{e} année.

Le schéma 4.3 récapitule donc le nombre de variables et leur répartition. On trouvera un code de la dynamique de la transition dans l'annexe A, section A.1.2.

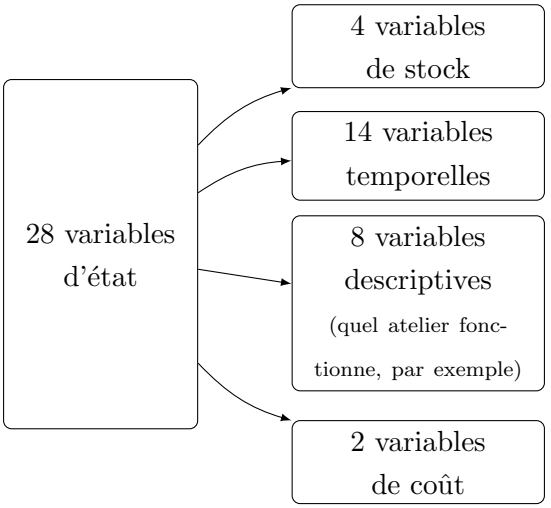

FIGURE 4.3 – Répartition des variables

4.3 Conclusion

Nous avons modélisé la chaîne de montage du chapitre 3 en utilisant un pas de temps particulier. En effet, nous avons considéré que le système changeait d'état lorsqu'un « événement » se produit, c'est-à-dire dès l'instant qu'une ou plusieurs variables changent. Nous avons distingué huit événements qui représentent autant de faits majeurs qui surviennent au cours du fonctionnement de la chaîne. À partir de cette modélisation, nous avons implémenté un simulateur en MATLAB. Ce dernier prend en entrée une stratégie et rend une trajectoire accompagnée de son coût.

L'avantage d'une telle modélisation est principalement la précision des trajectoires obtenues. Ainsi, on peut suivre exactement ce qui se passe dans la chaîne au cours du temps. Le MDP sous-jacent avec lequel nous allons travailler s'obtient en considérant uniquement les états nécessaires. C'est celui qui reprend l'état de la chaîne au début de chaque année, afin que l'échelle de temps du processus coïncide avec la fréquence de la décision. De même, l'implémentation des stratégies dans le simulateur nous a permis d'effectuer une première étape de réduction de l'espace des états. En effet, d'environ neuf milliards, nous nous sommes ramenés à un ensemble d'états de

cardinal au plus 47251 via une analyse de l'information physique sur laquelle le contrôleur se base pour décider d'un taux de production.

En revanche, notre démarche implique que nous ne connaissons que la loi de transition événementielle, pas celle du MDP sous-jacent. Il faudrait composer le noyau événementiel mais la loi du nombre de compositions nous est inconnue. De plus, il est difficile d'écrire analytiquement cette loi de transition. Cette simple constatation empêche l'application de toute procédure standard d'optimisation dont on a fait la revue au chapitre 2. La forme inhabituelle de la fonction de coût, à savoir que le coût d'une année ne peut se calculer qu'une fois icelle écoulée, corrobore cette conclusion. Il faut donc envisager d'autres techniques d'optimisation. Le chapitre 5 présente deux algorithmes qui pourraient potentiellement s'avérer adaptés au problème posé par Airbus.

5 Algorithmes stochastiques d'optimisation

La manière dont nous avons modélisé et simulé la chaîne de montage au chapitre 4 pose un problème majeur pour l'optimisation. En effet, trois échelles de temps coïncident :

— l'année pour l'optimisation, car c'est la fréquence de la décision,

— la journée pour le calcul des coûts, car on dispose de coûts journaliers,

— l'événement pour la simulation.

Or, si on est capable de simuler des trajectoires précises de la chaîne et, a fortiori, de récupérer les états qui forment le MDP associé, on ne saurait donner une écriture analytique de la loi de transition d'une année à une autre. En effet, on sait que cette loi est une itération du noyau « événementiel ». Mais on ne peut prédire combien d'événements se dérouleront en une année, et donc combien de fois il faut itérer ce noyau. Il est donc très difficile d'en avoir une forme explicite. Ajoutons à cette difficuté que la fonction de coût n'est pas de la forme usuelle, c'est-à-dire qu'elle n'est calculable qu'une fois l'année écoulée, donc elle ne dépend pas uniquement de l'état de départ et de la décision prise. Les techniques standard d'optimisation comme la programmation dynamique ou linéaire, qui requièrent la connaissance parfaite de la loi de transition et de la fonction de coût par transition, ne peuvent donc pas être utilisées ici.

Nous nous sommes donc tournés vers d'autres techniques ([CHFM13, HS04, HHC12]) et nous avons regardé de plus près deux procédures : l'algorithme MRAS (*Model Reference Adaptive Search*) et l'algorithme ASA (*Approximate Stochastic Annealing*) que l'on trouve dans [CHFM13]. Ces algorithmes proposent une approche originale de l'optimisation des processus markoviens décisionnels et utilisent très largement des simulations de trajectoires et la méthode de Monte-Carlo.

Considérons le problème général de minimisation suivant : soit $f : E \to \mathbb{R}$ une fonction minorée sur un certain ensemble non vide $E \subset \mathbb{R}^n$. On veut trouver un élément $x^* \in E$ tel que

$$x^* \in \underset{x \in E}{\operatorname{argmin}}\ f(x).$$

L'algorithme MRAS utilise une famille de lois de probabilités $\varphi(\cdot, \theta)$ sur E dépendant d'un paramètre θ pour générer des solutions potentielles au problème. La recherche d'un minimum pour f se transforme alors en recherche d'un paramètre θ^* optimal pour lequel les réalisations de la distribution $\varphi(\cdot, \theta^*)$ seraient les optima recherchés. La mise à jour du paramètre d'une itération à l'autre est déterminée par une suite de lois de référence $(g_k)_{k \geq 1}$ construites par récurrence selon la formule

$$g_k(x) = \frac{f(x) g_{k-1}(x)}{\displaystyle\int_E f(x) g_{k-1}(x)\, \mathrm{d}x} \text{ pour tout } x \in E$$

avec g_1 une loi initiale sur E. Le nouveau paramètre est celui qui minimise la projection de g_k sur la famille de lois $\varphi(\cdot, \theta)$ selon la *divergence de Kullback-Leibler* définie par

$$D(g_k || \varphi(\cdot, \theta)) = \int_E \log\left(\frac{g_k(x)}{\varphi(x, \theta)}\right) g_k(x)\, \mathrm{d}x.$$

La recherche de θ^* dépend donc fortement du choix de la famille $\varphi(\cdot, \theta)$. En pratique, on se tourne vers des familles pour lesquelles le calcul du paramètre θ qui minimise la divergence est facile. En l'occurrence, il en existe une pour laquelle le minimum possède une expression analytique ([ZFM10]) : la famille exponentielle. Il s'agit d'une famille de fonctions de la forme

$$\varphi(x, \theta) = \exp\left({}^t\theta\, \Gamma(x) - K(\theta)\right) h(x)$$

où $h : \mathbb{R}^n \to \mathbb{R}$, $\Gamma : \mathbb{R}^n \to \mathbb{R}^m$ sont des fonctions et $K(\theta) = \log \displaystyle\int_E \exp({}^t\theta\, \Gamma(x)) h(x)\, \mathrm{d}x$. De nombreuses lois usuelles font partie de cette famille, comme la loi normale, la loi de Poisson, la loi binomiale, la loi géométrique... C'est cette famille qui est utilisée dans l'algorithme que nous allons présenter ici. Le paramètre à optimiser est une matrice de probabilités à trois dimensions. La mise à jour de cette matrice à chaque itération fait intervenir une phase de sélection où le calcul d'un seuil (dans le cas des MDP, l'estimation d'un quantile de la loi du coût) fait en sorte d'écarter les solutions de mauvaise qualité.

L'algorithme ASA partage quelques similarités avec le MRAS. Mais son fonctionnement évoque également la méthode du recuit simulé ([BDM11]). Dans l'esprit général, il cherche à connaître le meilleur paramètre θ qui minimise la divergence de Kullback-Leibler $D(g_k || \varphi(\cdot, \theta))$ où $\varphi(\cdot, \theta)$ est une famille de lois de probabilités (nous nous servirons également de la famille exponentielle, pour simplifier les calculs). Comme pour le MRAS, ce paramètre prendra la forme d'une matrice de probabilités. Toutefois, la ressemblance avec la méthode de recuit simulé tient à la suite de lois de référence $(g_k)_{k \geq 1}$. En effet, on utilise une distribution de Boltzmann qui peut être écrite ainsi :

$$g_k(x) = \frac{\exp\left(-\dfrac{f(x)}{T_k}\right)}{\displaystyle\int_E \exp\left(-\frac{f(x)}{T_k}\right) \mathrm{d}x}$$

où T_k est un paramètre de température décroissant lorsque k tend vers l'infini. Cette distribution donne un poids plus fort aux éléments x de E avec $f(x)$ petit. Une différence notable avec le MRAS est l'absence de phase de sélection : toutes les solutions tirées avec la loi $\varphi(\cdot, \theta)$ sont mises à contribution.

Dans les deux cas, l'optimisation de la matrice de probabilités est très commode. En effet, le problème de minimisation du coût revient à minimiser un paramètre grâce auquel on est capables de simuler des stratégies. Ces stratégies ont la forme de matrices $M = (m_{i,t})$ où chaque cœfficient $m_{i,t}$, à valeurs dans $[\![1; 343]\!]$, représente le numéro de la décision à prendre si on est dans l'état numéro i au temps t. Ce sont donc des stratégies déterministes markoviennes directement utilisables avec le simulateur.

Naturellement, nous n'avons exposé dans cette introduction que les grands principes de ces deux algorithmes. Dans la suite du chapitre, ils vont apparaître sous une forme appliquée spécifiquement aux MDP. En l'occurrence, la fonction à minimiser sera celle qui à une stratégie associe son coût moyen. Ce dernier sera estimé via une méthode de Monte-Carlo. Ce chapitre propose de décrire avec précision ces deux algorithmes adaptés aux MDP. La première partie sera consacrée au MRAS, la seconde à l'ASA.

Nous reprenons ici certaines notations du chapitre précédent. En l'occurrence, nous noterons $\#\mathbb{X}$ le nombre total d'états, $\#\mathbb{A}$ le nombre total d'actions et H l'horizon du problème.

5.1 L'algorithme MRAS

5.1.1 Description générale de l'algorithme

L'algorithme opère sur une matrice à trois dimensions P où chaque cœfficient $P(i, j, t)$ indique la probabilité de choisir l'action a_j à l'instant t lorsqu'on se trouve dans l'état x_i. À chaque itération, l'algorithme met à jour cette matrice P en fonction des résultats rendus par la simulation du processus markovien décisionnel que l'on veut optimiser. On obtient ainsi une suite $(P_k)_{k\geq 0}$ de matrices qui représente la succession des mises à jour. L'objectif de l'algorithme est de faire en sorte de mettre un poids plus fort sur les stratégies les moins coûteuses.

Dans cette partie, nous allons nous attacher au fonctionnement général de l'algorithme. À chaque itération, il procède à plusieurs phases qui sont décrites ci-dessous.

Étape 1 : simulation des stratégies candidates

D'abord, on construit un certain nombre de stratégies, mais pas toutes de la même manière. En effet, certaines sont construites en tirant les actions selon la matrice de probabilité courante P, et les autres selon la matrice de probabilité initiale P_0.

Ainsi, simuler des stratégies selon P permet de garder les acquis des itérations précédentes et de diriger la recherche vers des stratégies potentiellement meilleures. La matrice initiale P_0 est cruciale car elle détermine un espace initial de solutions potentielles vers lequel l'algorithme est constamment renvoyé, et qu'il continue à explorer tout au long de son exécution.

Étape 2 : évaluation des performances

Pour chacune des stratégies construites à l'étape précédente, on évalue sa performance par la méthode de Monte-Carlo. Le nombre de simulations pour la méthode de Monte-Carlo doit être choisi selon des consignes précises que l'on détaillera plus tard.

Étape 3 : détermination des meilleures stratégies candidates

Toutes les stratégies simulées à l'étape 1 ne sont pas intéressantes. Certaines vont engendrer un coût plus élevé que d'autres. Cette étape consiste justement à se débarasser des mauvaises stratégies.

Pour ce faire, on calcule un seuil de performance et on ne garde que la proportion de stratégies candidates dont le coût est inférieur au seuil. Si le seuil calculé à cette itération est meilleur que le seuil de l'itération précédente, on le garde en mémoire. Si ce n'est pas le cas, il existe peut-être une stratégie qui engendre un coût moins élevé que le seuil de l'itération précédente, et ce dernier devient alors le nouveau seuil de performance.

Ce seuil est très important : par sa définition même, il est supposé décroître avec le nombre d'itérations et oriente donc la recherche vers des solutions toujours plus proches d'un optimum.

Étape 4 : mise à jour de P

C'est à cette étape que la matrice des probabilités P évolue, et donc que se redessine l'espace des solutions potentielles. Deux cas peuvent se présenter.

— Soit l'étape précédente n'a pas permis d'obtenir un meilleur seuil qu'auparavant, auquel cas P ne change pas.

— Soit on dispose d'un nouveau seuil. Dans ce cas, une formule permet de recalculer chaque cœfficient de P en ne faisant intervenir que les stratégies intéressantes mises en évidence à l'étape précédente (c'est-à-dire celles qui font mieux que le seuil) et éventuellement celles qui font un peu moins bien, selon une tolérance laissée à choisir en paramètre.

5.1.2 Dynamique de l'algorithme

Ici, nous allons donner précisément la suite des instructions à exécuter. On trouvera dans la section A.2.1 de l'annexe A le code utilisé pour l'implémentation du MRAS.

Paramètres

L'algorithme nécessite un certain nombre de paramètres d'entrée.

— $P_0 \in \mathcal{M}_{\#\mathbb{X},\#\mathbb{A},H}([0;1])$: matrice de probabilités initiale, qui peut être choisie selon une connaissance préalable du problème, ou qui, dans le cas contraire, doit être proche d'une loi uniforme,

— $\rho_0 \in]0;1]$: quantile initial qui va servir à calculer le seuil de performance, et donc qui détermine la proportion de candidats intéressants,

— $\varepsilon > 0$: tolérance pour la prise en compte de moins bonnes stratégies dans la mise à jour de P,

— $N_0 \geq 2$: nombre initial de stratégies candidates que l'on va simuler,

— $M_0 \geq 1$: nombre initial de simulations pour la méthode de Monte-Carlo,

— $\alpha > 1$: facteur par lequel on multiplie le nombre de candidats à simuler à la prochaine itération lorsqu'on ne trouve pas de seuil convenable,

— $\beta > 1$: facteur par lequel on multiplie, à la fin de chaque itération, le nombre de simulations pour la méthode de Monte-Carlo,

— $\lambda \in]0;1[$: cœfficient de mélange qui détermine la proportion de stratégies candidates qui seront simulées selon la matrice initiale P_0,

— $\nu \in]0;1]$: cœfficient de mixité qui est utilisé au moment de la mise à jour de P afin d'améliorer les performances numériques de l'algorithme,

— $\mathcal{H} : \mathbb{R} \to \mathbb{R}_+$ strictement décroissante : fonction qui sert à éviter que les cœfficients de P ne soient négatifs,

— $x_0 \in \mathbb{X}$: état initial pour le MDP,

— $k = 0$: compteur d'itérations,

— $K \in \mathbb{N}^*$: nombre limite d'itérations.

Fonctions particulières

L'algorithme d'optimisation fait appel à deux fonctions que l'on définit ici. D'une part, la fonction $\mathcal{I}$ est celle qui va permettre de quantifier la contribution de chacun des candidats potentiels en fonction du seuil de performance sélectionné. Elle est définie par :

$$\mathcal{I}(x,\chi) = \begin{cases} 0 \text{ si } x \geq \chi + \varepsilon \\ \dfrac{\chi + \varepsilon - x}{\varepsilon} \text{ si } \chi < x < \chi + \varepsilon \\ 1 \text{ si } x \leq \chi \end{cases} .$$

La figure 5.1 en donne une représentation graphique.

D'autre part, la fonction $\mathbf{f}$ apparaît dans le calcul de la mise à jour de la matrice P. Pour tous $1 \leq i \leq \#\mathbb{X}$, $1 \leq j \leq \#\mathbb{A}$, $1 \leq t \leq H$, on pose

$$\Pi_{i,j}(t) = \{\text{stratégies } \pi = (\pi_k)_{1 \leq k \leq H} \mid \pi_t(x_i) = a_j\}$$

et on définit $\mathbf{f}$ ainsi :

$$\mathbf{f}(\pi, P_k) = (1-\lambda) f(\pi, P_k) + \lambda f(\pi, P_0),$$

avec

$$f(\pi, P_k) = \prod_{t=0}^{H-1} \prod_{i=1}^{\#\mathbb{X}} \prod_{j=1}^{\#\mathbb{A}} P_k(i,j,t)^{\mathbb{1}_{\Pi_{i,j}(t)}(\pi)}.$$

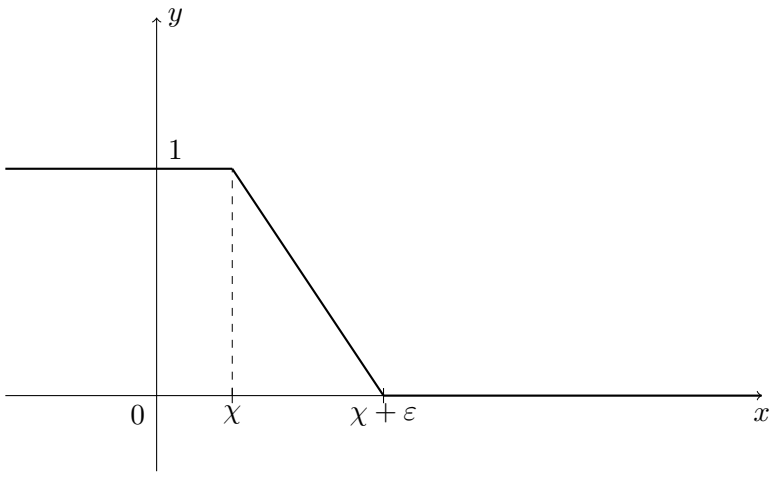

FIGURE 5.1 – Tracé de la fonction $x \mapsto \mathcal{I}(x, \chi)$

Instructions

TANT QUE $k \leq K$, exécuter les instructions suivantes.

1. **Simulation des stratégies candidates**
   Construire N_k stratégies notées π^n ($n \in [\![1; N_k]\!]$) de la manière suivante : pour chaque n,
   — avec probabilité λ, tirer les actions de π^n selon la matrice P_0 ;
   — avec probabilité $1 - \lambda$, tirer les actions de π^n selon la matrice P.

2. **Évaluation des performances (méthode de Monte-Carlo)**
   (a) Pour chaque stratégie π^n, simuler M_k trajectoires du MDP avec x_0 pour état initial.
   (b) Calculer, pour chaque m et chaque n, $V_{k,m}^n$ ($n \in [\![1; N_k]\!]$ et $m \in [\![1; M_k]\!]$), le coût engendré par la m^{e} trajectoire simulée avec la stratégie π^n.
   (c) Pour chaque n, calculer $\overline{V}_k^n = \dfrac{1}{M_k} \displaystyle\sum_{m=1}^{M_k} V_{k,m}^n$.

3. **Détermination des meilleures stratégies candidates**
   (a) Ranger les $\overline{V}_k^n$ par ordre décroissant et ainsi obtenir une suite $(\overline{V}_k^{(n)})$ de sorte que $\overline{V}_k^{(1)} \geq \ldots \geq \overline{V}_k^{(N_k)}$.
   (b) Calculer $\gamma_k(\rho_k, N_k) = \overline{V}_k^{(\lceil (1-\rho_k) N_k \rceil)}$ où $\lceil x \rceil$ est la partie entière supérieure de x.
   (c) Mettre à jour le seuil de performance $\overline{\gamma}_k$, le quantile et le nombre de stratégies à simuler de la manière suivante :
   — SI $k = 0$ OU $\gamma_k(\rho_k, N_k) \leq \overline{\gamma}_{k-1} - \varepsilon$
   $\hookrightarrow$ $\overline{\gamma}_k \leftarrow \gamma_k(\rho_k, N_k)$,

$\hookrightarrow$ $\rho_{k+1} \leftarrow \rho_k$,

$\hookrightarrow$ $N_{k+1} \leftarrow N_k$,

$\hookrightarrow$ $\boldsymbol{\pi}_k^* \leftarrow$ une stratégie π^n telle que $\overline{V}_k^{(n)} = \gamma_k(\rho_k, N_k)$.

— SINON on cherche le plus petit entier $\nu > \lceil (1-\rho_k)N_k \rceil$ tel que $\overline{V}_k^{(\nu)} \leq \overline{\gamma}_{k-1} - \varepsilon$

▷ Si ν existe :

$\hookrightarrow$ $\overline{\gamma}_k \leftarrow \overline{V}_k^{(\nu)}$,

$\hookrightarrow$ $\rho_{k+1} \leftarrow 1 - \dfrac{\nu}{N_k}$,

$\hookrightarrow$ $N_{k+1} \leftarrow N_k$,

$\hookrightarrow$ $\boldsymbol{\pi}_k^* \leftarrow$ une stratégie π^n telle que $\overline{V}_k^{(n)} = \overline{V}_k^{(\nu)}$.

▷ Si ν n'existe pas :

$\hookrightarrow$ $\overline{\gamma}_k \leftarrow \overline{V}_k^n$ où $\pi^n = \boldsymbol{\pi}_{k-1}^*$,

$\hookrightarrow$ $\rho_{k+1} \leftarrow \rho_k$,

$\hookrightarrow$ $N_{k+1} \leftarrow \lceil \alpha N_k \rceil$,

$\hookrightarrow$ $\boldsymbol{\pi}_k^* \leftarrow \boldsymbol{\pi}_{k-1}^*$.

4. **Calcul de la matrice P_{k+1}**

(a) SI $\forall\, n \in [\![1; N_k]\!]$, $\dfrac{[\mathcal{H}(\overline{V}_k^n)]^k}{\mathbf{f}(\pi^n, P_k)} \mathcal{I}(\overline{V}_k^n, \overline{\gamma}_k) = 0$,

$$P_{k+1} = P_k.$$

(b) SINON

i. on calcule d'abord une matrice auxiliaire $\hat{P}_{k+1}$ de la façon suivante : pour tous $1 \leq i \leq \#\mathbb{X}$, $1 \leq j \leq \#\mathbb{A}$, $1 \leq t \leq H$,

$$\hat{P}_{k+1}(i,j,t) \leftarrow \frac{\displaystyle\sum_{n=1}^{N_k} \frac{[\mathcal{H}(\overline{V}_k^n)]^k}{\mathbf{f}(\pi^n, P_k)} \mathcal{I}(\overline{V}_k^n, \overline{\gamma}_k) \mathbb{1}_{\Pi_{i,j}(t)}(\pi^n)}{\displaystyle\sum_{n=1}^{N_k} \frac{[\mathcal{H}(\overline{V}_k^n)]^k}{\mathbf{f}(\pi^n, P_k)} \mathcal{I}(\overline{V}_k^n, \overline{\gamma}_k)}.$$

ii. $P_{k+1} \leftarrow \nu \hat{P}_{k+1} + (1-\nu) P_k$.

(c) $M_{k+1} \leftarrow \lceil \beta M_k \rceil$.

(d) $k \leftarrow k+1$.

Remarque. Notons que $\mathbf{f}(\pi, P)$ ne peut valoir 0. En effet, si la stratégie testée (simulée à partir de la matrice P ou de la matrice P_0) préconise l'action a_j à l'état x_i et à l'instant t, c'est que $P(i,j,t)$ ou $P_0(i,j,t)$ n'est pas nul. Donc soit $f(\pi, P)$ n'est pas nul, soit $f(\pi, P_0)$ n'est pas nul.

5.1.3 Représentation des cœfficients des matrices $\hat{P}_k$

Le calcul des matrices $\hat{P}_k$ fait intervenir la fonction $\mathbf{f}$ qui rend une somme de produits de nombres positifs inférieurs à 1. Si $\#\mathbb{X}$, $\#\mathbb{A}$ ou H sont très grands, cette somme sera très proche de 0 si bien qu'elle aura une valeur inférieure à la précision machine, et sera numériquement traitée comme égale à 0. Toutefois, il est possible de calculer le logarithme naturel de la fonction f qui intervient dans le calcul de $\mathbf{f}$ (avec la convention numérique $\log(0) = -\infty$). Il faut donc adopter une représentation des nombres qui interviennent dans le calcul de $\hat{P}_k$.

Soit $k \in \mathbb{N}^*$. Pour tout $n \in [\![1; N_k]\!]$, on note

$$\alpha_n^{(k)} = \frac{A_n^{(k)}}{G_n^{(k)} + H_n^{(k)}} \tag{5.1}$$

avec

$$A_n^{(k)} = [\mathcal{H}(\overline{V}_k^n)]^k \mathcal{I}(\overline{V}_k^n, \overline{\gamma}_k),\ G_n^{(k)} = (1-\lambda) f(\pi^n, P_k) \text{ et } H_n^{(k)} = \lambda f(\pi^n, P_0)$$

de sorte que

$$\hat{P}_{k+1}(i,j,t) = \frac{\displaystyle\sum_{n=1}^{N_k} \alpha_n^{(k)} \mathbb{1}_{\Pi_{i,j}(t)}(\pi^n)}{\displaystyle\sum_{n=1}^{N_k} \alpha_n^{(k)}}. \tag{5.2}$$

On cherche à écrire $\alpha_n^{(k)}$ en fonction de $\log G_n^{(k)}$ et de $\log H_n^{(k)}$. On écrit alors

$$\alpha_n^{(k)} = \begin{cases} \dfrac{\exp(\log A_n^{(k)} - \log G_n^{(k)})}{1 + \exp(\log H_n^{(k)} - \log G_n^{(k)})} \text{ si } G_n^{(k)} \geq H_n^{(k)} \\ \\ \dfrac{\exp(\log A_n^{(k)} - \log H_n^{(k)})}{1 + \exp(\log G_n^{(k)} - \log H_n^{(k)})} \text{ si } H_n^{(k)} \geq G_n^{(k)}. \end{cases} \tag{5.3}$$

Le logarithme du numérateur est calculable (avec la convention numérique $\exp(-\infty) = 0$), et le dénominateur est calculable également car il est toujours compris entre 1 et 2. On représente donc les cœfficients $\alpha_n^{(k)}$ par le crochet suivant

$$\alpha_n^{(k)} = \left[a_n^{(k)}; b_n^{(k)}\right] \tag{5.4}$$

où

$$a_n^{(k)} = \begin{cases} \dfrac{1}{1 + \exp(\log H_n^{(k)} - \log G_n^{(k)})} \text{ si } G_n^{(k)} \geq H_n^{(k)} \\ \\ \dfrac{1}{1 + \exp(\log G_n^{(k)} - \log H_n^{(k)})} \text{ si } H_n^{(k)} \geq G_n^{(k)} \end{cases} \tag{5.5}$$

et

$$b_n^{(k)} = \begin{cases} \log A_n^{(k)} - \log G_n^{(k)} \text{ si } G_n^{(k)} \geq H_n^{(k)} \\ \log A_n^{(k)} - \log H_n^{(k)} \text{ si } H_n^{(k)} \geq G_n^{(k)} \end{cases} \tag{5.6}$$

de sorte que

$$\alpha_n^{(k)} = a_n^{(k)} \exp\left(b_n^{(k)}\right). \tag{5.7}$$

Enfin, pour calculer les matrices $\hat{P}_k$, on va normaliser chacun des cœfficients $\alpha_n^{(k)}$. Pour ce faire, on calcule

$$\hat{N} = \underset{n \in [\![1;N_k]\!]}{\operatorname{argmax}}\, b_n^{(k)}.$$

On factorise ensuite par $\alpha_{\hat{N}}^{(k)}$, de sorte que l'on ait

$$\hat{P}(i,j,t) = \frac{\displaystyle\sum_{n=1}^{N_k} \frac{\alpha_n^{(k)}}{\alpha_{\hat{N}}^{(k)}} \mathbb{1}_{\Pi_{i,j}(t)}(\pi^n)}{\displaystyle\sum_{n=1}^{N_k} \frac{\alpha_n^{(k)}}{\alpha_{\hat{N}}^{(k)}}} \tag{5.8}$$

où $\dfrac{\alpha_n^{(k)}}{\alpha_{\hat{N}}^{(k)}}$ est représenté par le crochet

$$\left[\frac{a_n^{(k)}}{a_{\hat{N}}^{(k)}} \,;\, b_n^{(k)} - b_{\hat{N}}^{(k)}\right]. \tag{5.9}$$

Ce crochet est calculable numériquement et permet d'obtenir la valeur des cœfficients de $\hat{P}$.

5.2 L'algorithme ASA

Nous proposons dans cette partie une deuxième technique d'optimisation adaptée à notre problème et que l'on peut retrouver dans [CHFM13] : l'algorithme ASA. Son principe est assez similaire à celui de l'algorithme MRAS (ils sont tous les deux basés sur le concept plus général de recuit simulé, cf. [BDM11]). La différence majeure entre les deux méthodes réside dans la fréquence de la mise à jour de la matrice de probabilités P_k. Dans le MRAS, cette matrice n'évolue que lorsqu'au moins une stratégie a donné une performance meilleure que le seuil (ou du moins, suffisamment proche), de sorte à diriger la recherche vers des stratégies plus prometteuses. En revanche, dans l'ASA, comme nous allons le voir, la matrice P_k est mise à jour à toutes les itérations et prend en compte toutes les stratégies simulées. Elle fait en sorte de mettre un poids plus fort sur celles qui ont donné un coût moindre.

5.2.1 Description générale de l'algorithme

L'algorithme ASA opère également sur une matrice à trois dimensions P où chaque cœfficient $P(i,j,t)$ indique la probabilité de choisir l'action a_j à l'instant t lorsqu'on se trouve dans l'état x_i. Les étapes de cet algorithme sont similaires à celle de l'algorithme MRAS.

Étape 1 : simulation des stratégies candidates

L'algorithme simule d'abord un certain nombre de stratégies. Certaines sont simulées selon la matrice initiale P_0, et les autres selon la matrice courante P. La proportion de stratégies simulées selon P_0 est amenée à décroître de manière au moins polynômiale au fur et à mesure des itérations. Le nombre de stratégies candidates doit croître de façon polynômiale.

Étape 2 : évaluation des performances

La méthode de Monte-Carlo est utilisée pour estimer la performance de chacune des stratégies construites à l'étape précédente. Le nombre de simulations utilisées doit croître suffisamment lentement.

Étape 3 : mise à jour de P

Contrairement à l'algorithme MRAS, la suite P_k va constamment évoluer. Une nouvelle matrice $\hat{P}$ est calculée en fonction des performances des stratégies construites à l'étape 1, en faisant en sorte de mettre un poids plus fort sur les meilleures stratégies grâce à un paramètre de température qui décroît au fur et à mesure des itérations. Puis une combinaison convexe de $\hat{P}$ et de P_k forme la matrice P_{k+1}. Ainsi, la nouvelle matrice des probabilités P_{k+1} ne s'éloigne pas trop de P_k, et permet une recherche plus efficace en évitant les extrema locaux.

5.2.2 Dynamique de l'algorithme

Ici, nous allons donner précisément la suite des instructions à exécuter. Dans la section A.2.2 de l'annexe A, on trouvera le code de l'ASA que nous avons implémenté.

Paramètres

L'algorithme nécessite un certain nombre de paramètres d'entrée.

— $P_0 \in \mathcal{M}_{\#\mathbb{X},\#\mathbb{A},H}([0;1])$: matrice de probabilités initiale, qui doit être initialisée à une loi uniforme (i.e. tous ses cœfficients doivent valoir $\frac{1}{\#\mathbb{A}}$),

— $\alpha_0 = \frac{1}{100^{0,501}}$: cœfficient de lissage initial qui contrôle l'évolution de la suite de matrices (P_k),

— $\beta_0 = 1$: cœfficient de mélange initial qui détermine la proportion de stratégies qui seront simulées selon la matrice P_0,

— $T_0 \in \mathbb{R}_+^*$: température initiale, qui gouverne la convergence de l'algorithme (elle ne doit être choisie ni trop élevée, pour ne pas ralentir la convergence de l'algorithme, ni trop proche de zéro, auquel cas l'algorithme pourrait se retrouver coincé dans un extremum local),

— $N_0 \in \mathbb{N}^*$: nombre initial de stratégies candidates à simuler,

— $M_0 \in \mathbb{N}^*$: nombre initial de simulations pour la méthode de Monte-Carlo,

— x_0 : état initial pour le MDP,
— $k = 0$: compteur d'itérations,
— $K \in \mathbb{N}^*$: nombre limite d'itérations.

Remarque. Les valeurs prises pour α_0 et β_0 sont déterminées par le choix explicite de suites (α_k) et (β_k) dont les formules générales sont données lors de la description des instructions.

Une distribution sur l'ensemble des stratégies

L'algorithme d'optimisation fait appel à la distribution de probabilité $\mathbf{f}$ sur l'ensemble des stratégies qu'on a évoquée pour l'algorithme MRAS. Elle apparaît dans le calcul de la mise à jour de la matrice P. On rappelle qu'elle est ainsi définie :

$$\mathbf{f}(\pi, P_k) = (1 - \beta_k) f(\pi, P_k) + \beta_k f(\pi, P_0),$$

avec

$$f(\pi, P_k) = \prod_{t=0}^{H-1} \prod_{i=1}^{\#\mathbb{X}} \prod_{j=1}^{\#\mathbb{A}} P_k(i,j,t)^{\mathbb{1}_{\Pi_{i,j}(t)}(\pi)}.$$

Instructions

TANT QUE $k \leq K$, exécuter les instructions suivantes.

1. **Simulation des stratégies candidates**
   Construire N_k stratégies notées π^n ($n \in [\![1; N_k]\!]$) de la manière suivante : pour chaque n,
   — avec probabilité β_k, tirer les actions de π^n selon la matrice P_0 ;
   — avec probabilité $1 - \beta_k$, tirer les actions de π^n selon la matrice P.

2. **Évaluation des performances**
   (a) Pour chaque stratégie π^n, simuler M_k trajectoires du MDP avec x_0 pour état initial.
   (b) Calculer, pour chaque m et chaque n, $V_{k,m}^n$ ($n \in [\![1; N_k]\!]$ et $m \in [\![1; M_k]\!]$), le coût engendré par la m^{e} trajectoire simulée avec la stratégie π^n.
   (c) Pour chaque n, calculer $\overline{V}_k^n = \dfrac{1}{M_k} \displaystyle\sum_{m=1}^{M_k} V_{k,m}^n$.

3. **Calcul de la matrice P_{k+1}**
   (a) on calcule d'abord une matrice auxiliaire $\hat{P}_{k+1}$ de la façon suivante : pour tous entiers $1 \leq i \leq \#\mathbb{X}$, $1 \leq j \leq \#\mathbb{A}$ et $1 \leq t \leq H$,

$$\hat{P}_{k+1}(i,j,t) = \frac{\displaystyle\sum_{n=1}^{N_k} \frac{\exp(-\overline{V}_k^n T_k^{-1})}{\mathbf{f}(\pi^n, P_k)} \mathbb{1}_{\Pi_{i,j}(t)}(\pi^n)}{\displaystyle\sum_{n=1}^{N_k} \frac{\exp(-\overline{V}_k^n T_k^{-1})}{\mathbf{f}(\pi^n, P_k)}}.$$

(b) $P_{k+1} \leftarrow \alpha_k \hat{P}_{k+1} + (1-\alpha_k)P_k$.

4. **Calcul des nouveaux paramètres de l'algorithme**

(a) $M_{k+1} \leftarrow \max(M_0, \lfloor 1{,}01 \log^3(k) \rfloor)$,

(b) $N_{k+1} \leftarrow \max(N_0, \lfloor k^{0{,}501} \rfloor)$,

(c) $\alpha_{k+1} \leftarrow \dfrac{1}{(k+100)^{0{,}501}}$,

(d) $\beta_{k+1} \leftarrow \dfrac{1}{\sqrt{k+1}}$,

(e) $T_{k+1} \leftarrow \dfrac{T_0}{\log(k+\mathrm{e})}$,

(f) $k \leftarrow k+1$.

Remarque. Pour les mêmes raisons que pour l'algorithme MRAS, la fonction $\mathbf{f}(\pi, P)$ ne peut valoir 0.

5.2.3 Représentation des cœfficients des matrices $\hat{P}_k$

Comme pour l'algorithme MRAS, le calcul des matrices $\hat{P}_k$ peut poser problème à cause de certaines valeurs numériques très proches de 0. On choisit d'adopter une représentation de ses cœfficients similaire : pour tous $k \in \mathbb{N}^*$ et $n \in [\![1; N_k]\!]$, on note

$$\alpha_n^{(k)} = \frac{A_n^{(k)}}{G_n^{(k)} + H_n^{(k)}}$$

avec

$$A_n^{(k)} = \exp(-\overline{V}_k^n T_k^{-1}),\ G_n^{(k)} = (1-\beta_k) f(\pi^n, P_k) \text{ et } H_n^{(k)} = \beta_k f(\pi^n, P_0)$$

de sorte que

$$\hat{P}_{k+1}(i,j,t) = \frac{\displaystyle\sum_{n=1}^{N_k} \alpha_n^{(k)} \mathbb{1}_{\Pi_{i,j}(t)}(\pi^n)}{\displaystyle\sum_{n=1}^{N_k} \alpha_n^{(k)}}.$$

Les cœfficients $\alpha_n^{(k)}$ sont représentés par des crochets identiques à ceux construits pour l'algorithme MRAS. On se réfèrera à la section 5.1.3 pour connaître le détail de cette construction.

5.3 Conclusion

Les deux algorithmes présentés dans ce chapitre sont des procédures d'optimisation basées sur des simulations. Ils consistent à optimiser une matrice de probabilités avec laquelle on génère des stratégies. Pour ce faire, ils simulent des stratégies et évaluent leur performance via la méthode de Monte-Carlo. La mise à jour de la matrice de probabilités est différente selon l'algorithme : le MRAS fait appel à un seuil de performance tandis que l'ASA calcule une sorte

de moyenne pondérée. Du fait de ces approches différentes, le MRAS nécessite le réglage d'un nombre conséquent de paramètres, contrairement à l'ASA.

La complexité du problème a nécessité quelques aménagements dans l'implémentation des algorithmes. En effet, dans les deux cas, la mise à jour de la matrice de probabilités nécessite de calculer deux produits de $H \times \#\mathbb{X} \times \#\mathbb{A}$ facteurs. Par exemple, pour un stock maximal de SRM de 4, auquel cas $\#\mathbb{X} = 26251$, on doit multiplier 270 122 790 nombres entre 0 et 1. Ce produit, théoriquement strictement positif, a une valeur inférieure à la précision machine. Il est donc numériquement traité comme valant 0. Nous avons donc adopté une représentation logarithmique des nombres en jeu pour pallier ce problème. Le chapitre 6 montrera comment on réduira encore plus l'espace des états.

Ces deux procédures sont très gourmandes en simulations de trajectoires. En effet, d'une part, il faut pouvoir assurer la précision des estimations des performances dans l'application de la méthode de Monte-Carlo. D'autre part, il faut effectuer un nombre suffisant d'itérations pour avoir une approximation acceptable de la matrice de probabilités optimale. Par exemple, les tests de l'ASA que nous présenterons au chapitre 6 ont requis aux alentours de soixante-quinze millions de trajectoires pour parvenir au résultat. Afin d'avoir des résultats en temps raisonnable, il faut donc envisager un portage du simulateur en un autre langage, plus adapté à l'architecture de son code informatique.

Maintenant que nous disposons de deux méthodes d'optimisation implémentées qui répondent bien au problème, la suite de notre travail va consister à comparer leurs performances, sélectionner celle qui convient le mieux et répondre au problème posé par Airbus.

6 Optimisation de la chaîne de montage

Nous disposons de deux algorithmes, le MRAS et l'ASA, implémentés et utilisables avec le simulateur que nous avons construit. Ils ne fonctionnent pas de la même manière : on peut donc légitimement supposer que leurs performances sont différentes. C'est ce qu'il faut examiner dans un premier temps, et garder comme objectif de déterminer lequel est le plus efficace. Cette étape, nécessaire, sera l'occasion d'éprouver notre méthode de simulation de la chaîne de montage. Dans un deuxième temps, nous pourrons tester l'algorithme sélectionné sur des calendriers générés aléatoirement pour comparer ses performances avec des stratégies de référence dites « naïves » qui consistent à produire le juste nombre de lanceurs. Ainsi, il sera temps d'examiner la question de la capacité optimale du stock de SRM.

Dès lors, nous pourrons aller plus loin en nous occupant de problématiques plus typiquement industrielles. Lors de notre collaboration avec Airbus, nous avons évoqué la possibilité de compressions calendaires, c'est-à-dire le choix, dans une année donnée, de rapprocher certains tirs pour compenser un arrêt éventuel de la chaîne, pour des raisons de maintenance par exemple. Cette question, restée en suspens durant la phase de simulation de la chaîne et d'implémentation des algorithmes, sera pour nous un prétexte à l'utilisation concrète de notre solution et un moyen d'aller plus loin.

Pour autant, avant même d'implémenter les deux algorithmes, nous avons procédé à une étape numérique nécessaire pour vérifier la justesse du simulateur et qui nous a permis de dégager quelques traits du comportement de la chaîne. Nous disposions ainsi de points de repère

pour l'optimisation stochastique. Cette étape a également été importante pour appréhender la fonction de coût et ses variations.

Donc, nous donnerons d'abord les quelques résultats numériques préliminaires obtenus. Puis nous nous consacrerons à l'optimisation stochastique en présentant d'abord une comparaison des résultats des algorithmes étudiés au chapitre 5 appliqués à la chaîne de montage du chapitre 3 et d'éventuelles modifications du simulateur décrit dans le chapitre 4 pour améliorer leurs performances. Puis nous verrons quelques exemples de calendriers et tenterons pour chacun de répondre aux questions posées pour la chaîne étudiée. Enfin, nous examinerons une question de robustesse des stratégies optimales.

6.1 Premiers résultats numériques

Pour les tests effectués dans cette partie, nous avons

— fixé la capacité du stock de SRM à 8 unités,

— établi un calendrier qui exige, pour les années 5 à 30, 10 tirs par an (les quatre premières années sont restées telles que décrites dans la section 3.3 du chapitre 3).

6.1.1 Vitesse maximale de production

On examine ici ce qui se passe lorsque l'agent extérieur décide de produire 12 lanceurs par an, soit 48 IMC, 12 LLPM et 12 ULPM. Dans ce cas, un LLPM (ou un ULPM) est produit en $\lfloor \frac{261}{12} \rfloor = 21$ jours. Un IMC est produit en $\lfloor \frac{261}{48} \rfloor = 5$ jours.

Comme on pouvait s'y attendre, la vitesse maximale de production ne génère quasiment aucun retard dans les dates de tir. D'après la figure 6.1, les retards sont d'au plus une demi-journée, ce qui s'explique par le faible aléa mis sur la durée opératoire de la LP.

En revanche, si on examine l'évolution du niveau des stocks, on peut s'apercevoir que cette vitesse de production est trop élevée. En effet, la figure 6.2 montre qu'il y a toujours au moins 4 SRM inutilisés dans le stock. De même, sur la figure 6.3, on voit que 3 LLPM restent aussi inutilisés durant le processus (il se produit le même phénomène pour les ULPM). Il y a donc une marge conséquente pour réduire les coûts, reportés dans le tableau 6.1.

Coût	Valeur
Stockage	2 821 700
Retard	4300
Total	2 826 000

Tableau 6.1 – Coût moyen sur 100 simulations de la vitesse maximale de production

6.1.2 Vitesse minimale de production

Ici, l'agent extérieur décide de produire 6 lanceurs par an, soit 24 IMC, 6 ULPM et 6 LLPM. Par conséquent, un LLPM (ou un ULPM) sera produit en 43 jours et un IMC sera produit en

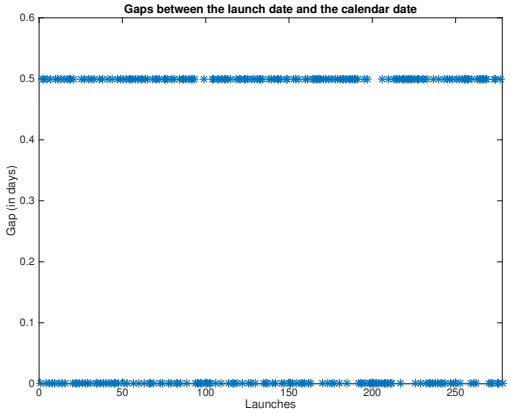

FIGURE 6.1 – Évolution des écarts au calendrier pour la vitesse maximale de production

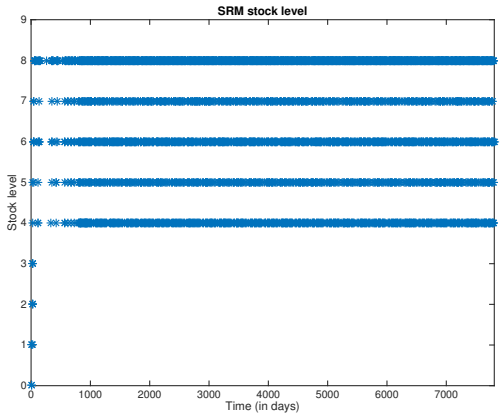

FIGURE 6.2 – Évolution du stock de SRM dans le temps pour la vitesse maximale de production

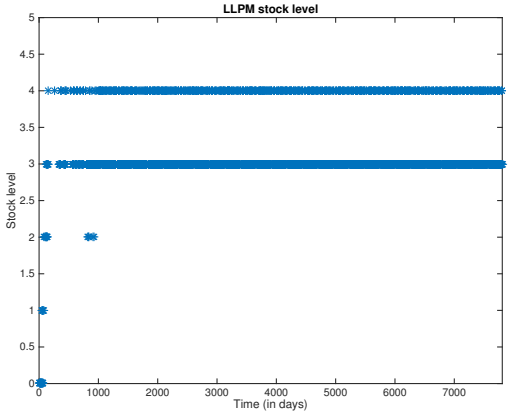

FIGURE 6.3 – Évolution du stock de LLPM dans le temps pour la vitesse maximale de production

10 jours. Intuitivement, d'aucuns s'attendent à un coût de retard plus élevé mais à un coût de stockage moindre, puisqu'ils seront en nombre insuffisant.

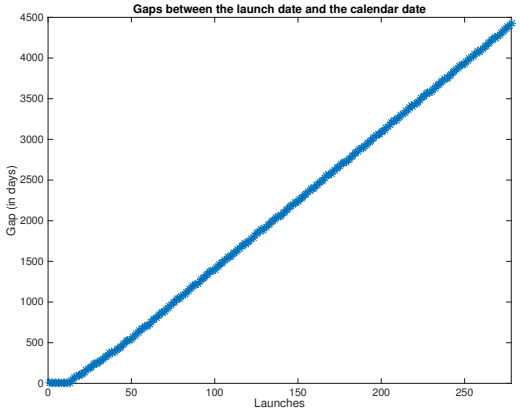

FIGURE 6.4 – Évolution des écarts au calendrier pour la vitesse minimale de production

Comme le montre la figure 6.4, les retards sur le calendrier explosent. Seuls 175 tirs sur 278 ont été faits dans les 30 ans et le dernier tir a 4423,5 jours de retard (soit environ 12 ans !). Cette vitesse n'est naturellement pas acceptable. Mais examinons tout de même l'évolution des stocks. Cette stratégie induit qu'on a besoin de beaucoup de temps pour produire un Central Core (environ 70 jours) tandis qu'un SRM est créé en 15 jours. Ces derniers attendent donc longtemps dans leur stock. C'est pour cette raison que le stock de SRM se comporte de manière similaire à la figure 6.2. En revanche, un LLPM (et ce qui suit vaut aussi pour les ULPM) n'attend donc quasiment pas dans son stock, comme le montre la figure 6.5. On trouvera dans le tableau 6.2 les coûts moyens de stockage et de retard avec cette vitesse.

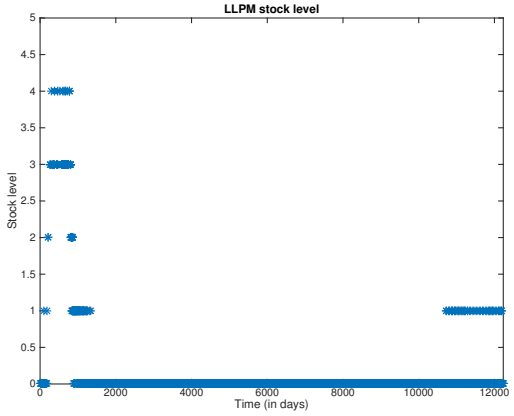

FIGURE 6.5 – Évolution du stock de LLPM dans le temps pour la vitesse minimale de production

Coût	Valeur
Stockage	1 029 000
Retard	18 130 000
Total	19 159 000

Tableau 6.2 – Coût moyen sur 100 simulations de la vitesse minimale de production

6.1.3 Vitesse moyenne de production

Considérons maintenant une stratégie qui consisterait à produire 10 lanceurs par an, c'est-à-dire 40 IMC, 10 LLPM et 10 ULPM en moyenne. Cette vitesse correspond au juste nombre de tirs à effectuer. Les retards ne devraient donc pas être trop importants : c'est ce que montre la figure 6.6. Les stocks devraient contenir moins de pièces inutilisées. Les figures 6.7, 6.8 et 6.9 montrent que c'est bien le cas. Toutefois, on remarquera qu'il reste encore des sous-assemblages stockés inutilement. Par exemple, il reste constamment 4 SRM en stock.

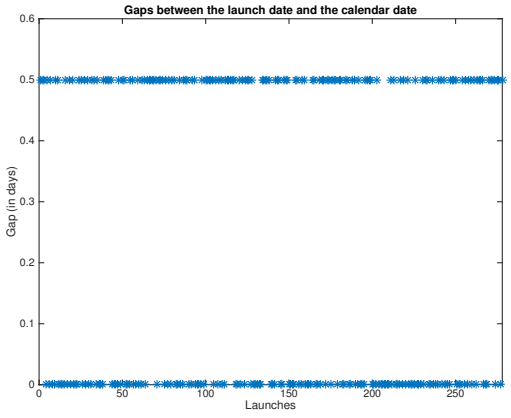

Figure 6.6 – Évolution des écarts au calendrier pour une vitesse moyenne de production

Coût	Valeur
Stockage	2 271 000
Retard	4300
Total	2 275 300

Tableau 6.3 – Coût moyen sur 100 simulations de une vitesse moyenne de production

Le tableau 6.3 montre que cette stratégie est indéniablement meilleure que la stratégie qui consiste à adopter la vitesse maximale. En effet, le coût de retard est identique, mais on gagne sur le stockage. Toutefois, avec les considérations sur le stock, on peut raisonnablement supposer qu'il y a une grande marge d'amélioration : changer la vitesse de production chaque année doit permettre au contrôleur de s'adapter aux niveaux des stocks (par exemple ralentir la production

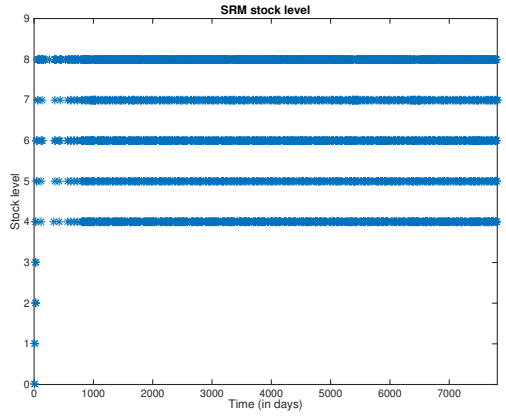

FIGURE 6.7 – Évolution du stock de SRM dans le temps pour une vitesse moyenne de production

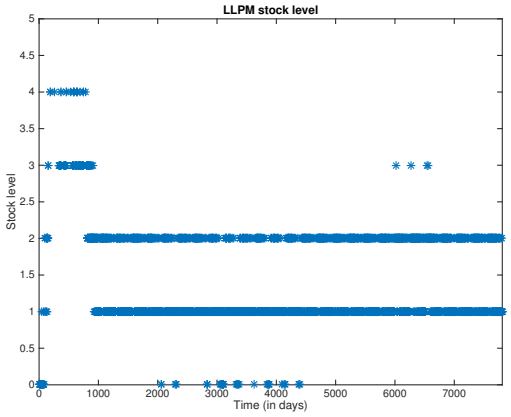

FIGURE 6.8 – Évolution du stock de LLPM dans le temps pour une vitesse moyenne de production

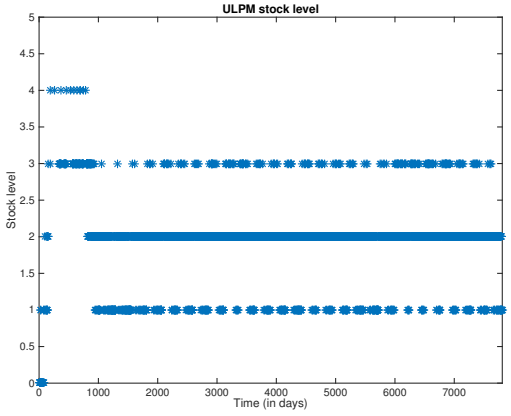

FIGURE 6.9 – Évolution du stock de ULPM dans le temps pour une vitesse moyenne de production

lorsqu'il y a beaucoup de sous-assemblages disponibles) et de diminuer encore plus les coûts. C'est précisément ce que les deux algorithmes explicités dans le chapitre 5 sont censés prouver.

6.2 Performances des deux algorithmes

Pour effectuer les tests, nous avons

— repris les conditions des tests de la partie précédente, à l'exception de la durée du calendrier qui ne s'étale que sur 10 ans pour ce qui va suivre,

— limité les cadences de production à une plage située entre 8 et 12 tirs, compte tenu du calendrier choisi,

— ajouté un coût supplémentaire : tout tir non fait à la fin de la période de temps considérée entraîne un surcoût de 10 000 000 afin de pénaliser très fortement les stratégies qui ne sont pas acceptables.

Notre objectif est de vérifier la cohérence des résultats rendus par les algorithmes MRAS et ASA en les comparant avec les performances de stratégies dites « naïves » décrites ci-après. Nous aurons à la fois à repenser le problème, et sélectionner la réponse la plus pertinente au regard de la question de l'optimisation. Dans cette section et dans toutes les suivantes de ce chapitre, la machine sur laquelle nous avons fait tourner nos algorithmes est un Mac Pro 2,7 GHz 12-Core avec 64 Go de RAM.

Toutefois, le MRAS et l'ASA font appel à la méthode de Monte-Carlo de nombreuses fois. Typiquement, les tests que nous avons effectué ont requis plusieurs dizaines de millions de simulations de la chaîne de montage (environ 75 000 000 pour les tests de l'ASA sur 30 ans). Il fallait donc que notre simulateur soit suffisamment rapide pour donner des résultats en temps raisonnable. Nous l'avons d'abord implémenté en MATLAB, qui rendait une trajectoire en 0,7 seconde. Les algorithmes étaient donc trop longs à l'exécution. Nous avons donc choisi de nous tourner vers le langage C, connu pour sa rapidité de traitement des boucles. Le simulateur en C rend une trajectoire en 0,003 seconde, ce qui représente une nette amélioration dans le temps d'exécution des algorithmes. Dans toute la suite du chapitre, c'est le simulateur en C que nous utilisons.

6.2.1 Stratégies « naïves »

Une stratégie dite « naïve » propose, pour chacun des trois sous-assemblages IMC, LLPM et ULPM, de faire produire en moyenne chaque année le même nombre d'unités. Tester de telles stratégies a du sens, étant donné le calendrier particulier que l'on s'est fixé. On trouvera dans le tableau 6.4 l'estimation de la performance de ces stratégies naïves.

Comme on pouvait s'y attendre, c'est la stratégie qui consiste à produire 40 IMC, 10 LLPM et 10 ULPM par an en moyenne qui est la moins coûteuse : elle permet de subvenir aux besoins sans excédent. La quatrième année, qui demande 11 tirs, ne pose pas de problème en raison du stock qui s'est formé auparavant. Produire moins de sous-assemblages génère beaucoup de retards, et en produire plus génère des coûts de stockage inutiles.

Nombre moyen d'unités par an			Performance
IMC	LLPM	ULPM	
32	8	8	123 770 000
36	9	9	45 666 000
40	10	10	809 540
44	11	11	945 340
48	12	12	972 440

TABLEAU 6.4 – Performance de stratégies naïves estimées par la méthode de Monte-Carlo avec 100 000 simulations

Donc, notre stratégie de référence sera celle qui consiste à produire 40 IMC, 10 LLPM et 10 ULPM par an. On ne sait pas si elle est optimale : c'est ce que les algorithmes auxquels nous avons songé devront confirmer ou infirmer.

6.2.2 Algorithme MRAS

Dans tous les tests effectués avec l'algorithme MRAS, sauf mention explicite contraire, nous avons fixé :

— le nombre initial de stratégies candidates N_0 à 100,

— le nombre initial de simulations pour la méthode de Monte-Carlo M_0 à 1000,

— le nombre limite d'itérations K à 100,

— la valeur du paramètre μ à 10^{-8},

— le quantile initial ρ_0 à 0,25.

Pour des raisons de mémoire, nous avons également limité à 600 le nombre de candidats à tester.

Tolérance ε et matrice de probabilités initiale P_0

Notre première préoccupation a été de vérifier la sensibilité de l'algorithme aux paramètres. Deux d'entre eux ont attiré notre attention en premier lieu : la tolérance ε, qui a une influence à la fois sur la sélection éventuelle d'un nouveau seuil et sur la mise à jour de la matrice P_k, et la matrice initiale P_0, qui détermine quel espace initial de solutions potentielles l'algorithme va considérer.

Les tableaux 6.5 et 6.6 récapitulent les résultats obtenus en faisant varier ε et l'espace de solutions initial. Pour ce dernier point, nous avons voulu comparer les résultats rendus par l'algorithme lorsqu'il doit chercher une solution simulée en utilisant tout l'espace des actions (ce qu'on nomme « espace entier ») et lorsqu'il part d'un ensemble de stratégies qui ne proposent que certaines actions (notamment qui excluent de produire 32 IMC dans l'année, ce qu'on nomme « espace réduit »).

Au premier abord, ce qui nous frappe est que l'algorithme n'a, dans aucun des cas, réussi à trouver de stratégie optimale (au mieux, il a trouvé une stratégie qui engendre un coût de 817

	$\varepsilon = 1000$	$\varepsilon = 1$
Espace entier	1 596 437	906 675
Espace réduit	850 515	817 855

TABLEAU 6.5 – Meilleur seuil trouvé selon la tolérance ε et l'espace de stratégies initial pour le MRAS avec $\alpha = 1{,}02$, $\beta = 1{,}001$, $\lambda = 0{,}4$ et $\nu = 0{,}5$

	$\varepsilon = 1000$	$\varepsilon = 1$
Espace entier	83	62
Espace réduit	22	93

TABLEAU 6.6 – Dernière itération où P_k a évolué selon la tolérance ε et l'espace de stratégies initial pour le MRAS avec $\alpha = 1{,}02$, $\beta = 1{,}001$, $\lambda = 0{,}4$ et $\nu = 0{,}5$

855, c'est-à-dire avec une erreur relative par rapport à notre stratégie de référence de 1,03%), bien qu'il soit indéniable que choisir $\varepsilon = 1$ amène à de meilleurs résultats. On pouvait s'y attendre : si ε est trop grand, l'algorithme peut ne pas faire évoluer P_k alors qu'un meilleur seuil a été trouvé. Il est donc préférable de choisir pour ce paramètre une valeur faible.

On pourra objecter que l'algorithme n'a pas assez cherché : en effet, nous ne l'avons fait tourner « que » sur 100 itérations. C'est là que le tableau 6.6, qui répertorie le numéro de la dernière itération où P_k a évolué, se révèle utile : on voit que l'algorithme cesse en fait de chercher assez vite (sauf lorsque $\varepsilon = 1$ et qu'on réduit l'espace des solutions, mais d'autres tests ont montré que ce n'était que le fruit du hasard : avec les mêmes paramètres, l'algorithme n'a plus fait évoluer la matrice P_k après la 40^{e} itération).

Si P_k n'évolue pas, c'est qu'aucune stratégie n'a donné une meilleure performance. Alors, le nombre de candidats augmente. Lorsque l'algorithme teste trop de candidats, la durée des itérations s'allonge considérablement (en moyenne, 100 itérations nécessitent environ 4 heures, avec une forte variance). Et, compte tenu de la taille des matrices utilisées, avoir trop de stratégies en mémoire fait geler l'ordinateur. C'est arrivé alors que nous testions l'algorithme MRAS sur 150 itérations : il en était venu à tester plus de 1000 candidats sans trouver de meilleur seuil !

Tester sur 100 itérations nous a donc paru être un bon compromis, d'autant plus que, comme nous allons le voir, il n'est pas flagrant que les paramètres ont une réelle influence sur la recherche d'une solution optimale.

Cœfficient de mélange λ

Nous avons donc sélectionné pour la suite des tests $\varepsilon = 1$. Nous nous sommes penchés sur le paramètre λ, qui détermine quelle proportion de stratégies sera construite à partir de la matrice initiale P_0. C'est ce paramètre qui permet à l'algorithme de garder la main sur tout l'espace des solutions. Il est naturel de penser qu'il joue un rôle certain. Le tableau 6.7 répertorie les résultats obtenus pour nos tests.

Le cœfficient λ semble ne pas beaucoup influer sur le résultat rendu par l'algorithme : on reste, avec ces trois valeurs, dans le même ordre de grandeur. Quant à la dernière itération utile (c'est-à-dire la dernière itération où P_k a évolué), on aurait pu penser qu'une plus petite valeur pour λ aurait donné une plus grande possibilité de trouver de meilleures stratégies, ce qui ne semble pas être le cas.

Valeur de λ	Meilleur seuil trouvé	Dernière itération utile
0,4	906 675	62
0,1	1 045 437	72
0,7	926 401	32

TABLEAU 6.7 – Résultats des tests en fonction de λ pour le MRAS, avec $\alpha = 1{,}02$, $\beta = 1{,}001$, $\nu = 0{,}5$, $\varepsilon = 1$ et en considérant tout l'espace des actions

Cœfficient de mixité ν

Une autre question que nous nous sommes posés est celle de la portée du cœfficient de mixité ν. Les auteurs de [CHFM13] établissent que ce paramètre doit améliorer les performances numériques de l'algorithme. Mais existe-t-il une manière judicieuse de le choisir ? Le tableau 6.8 répertorie nos résultats.

Valeur de ν	Meilleur seuil trouvé	Dernière itération utile
0,5	906 675	62
0,25	1 286 362	84
0,75	1 070 594	87

TABLEAU 6.8 – Résultats des tests en fonction de ν pour le MRAS, avec $\alpha = 1{,}02$, $\beta = 1{,}001$, $\lambda = 0{,}4$, $\varepsilon = 1$ et en considérant tout l'espace des actions

Il ne ressort pas de comportement particulier en fonction de ν. Il reste en tous les cas manifeste que le résultat de l'algorithme MRAS est très éloigné de la performance de la stratégie naïve de référence (au mieux, l'erreur relative est de 12%).

Cœfficients α et β

Enfin, nous nous sommes penchés sur le rôle des paramètres α et β, qui contrôlent la croissance respectivement du nombre de stratégies candidates testées et du nombre de simulations pour la méthode de Monte-Carlo. Afin de vérifier si l'algorithme MRAS était tout de même capable d'arriver au moins à la stratégie naïve de référence, nous avons effectué des tests avec l'espace réduit de stratégies décrit plus haut. Les résultats de ces tests sont présentés dans le tableau 6.9.

Comme pour les autres cœfficients, il ressort de nos tests que les paramètres d'entrée n'ont pas vraiment d'influence sur la convergence de l'algorithme. Il faut donc réfléchir à une autre approche.

6.2.3 Agrégation d'états

Les tests précédents montrent un fait : l'algorithme MRAS ne gère pas bien la géométrie particulière du problème. En effet, les solutions qu'il propose ne sont pas bonnes : bien que fini,

Valeur de α	Valeur de β	Meilleur seuil trouvé	Dernière itération utile
1,02	1,001	817 855	93
1,02	1,0205	839 081	35
1,01	1,0205	870 776	68

TABLEAU 6.9 – Résultats des tests en fonction de α et β pour le MRAS, avec $\lambda = 0{,}4$, $\nu = 0{,}5$, $\varepsilon = 1$ et en considérant l'espace d'actions réduit

l'espace de toutes les stratégies est toujours trop grand pour ce type d'algorithme de recherche (son cardinal vaut 125^{472510} pour nos tests). Il est donc déjà encourageant de voir que l'algorithme MRAS se rapproche malgré tout d'une stratégie optimale. Mais ce n'est pas suffisant. Donc, plutôt que de penser la convergence de l'algorithme en termes de réglage des paramètres, nous avons préféré réduire l'espace des états, de sorte que le cardinal de l'espace des solutions soit bien plus faible d'une part, et d'autre part que chaque itération soit exécutée plus rapidement.

Nous avons alors choisi d'agréger certains états. En l'occurrence, nous avons opéré les transformations suivantes :

— si, dans une année, le nombre de tirs à effectuer est supérieur à 12, on appelle cet état « 12 tirs et plus », de sorte que `tirs_prevus` ne soit plus qu'à valeurs dans $[\![1; 12]\!]$;

— le nombre d'IMC, de LLPM ou d'ULPM sera codé par un entier entre 1 et 3 comme décrit par le tableau 6.10 ;

— le nombre de SRM en stock sera codé par un entier entre 1 et 3 comme décrit par le tableau 6.11.

Nombre d'unités en stock	Code
0 ou 1	1
2 ou 3	2
4	3

TABLEAU 6.10 – Table de codage pour l'agrégation des états pour les stocks des sous-assemblages

Nombre d'unités en stock	Code
0, 1, 2 ou 3	1
4, 5, 6 ou 7	2
8	3

TABLEAU 6.11 – Table de codage pour l'agrégation des états du stock de SRM

Dans ses grandes lignes, ce codage revient à décrire qualitativement l'état des stocks selon trois états « vide ou peu rempli », « moyennement rempli » et « plein ». Ainsi, on recherche une stratégie optimale parmi un sous-ensemble de stratégies qui consistent à prendre la même décision selon certaines valeurs des stocks. Avec cette solution, on ramène le nombre d'états à 2188, et donc le nombre de stratégies possibles à 125^{21880} pour les tests de cette partie.

Nous avons donc effectué des tests de l'algorithme MRAS avec cette réduction d'états, dont les résultats sont repris dans les tableaux 6.12 et 6.13.

On constate que l'algorithme MRAS, bien qu'indéniablement plus performant, n'est arrivé qu'une fois à trouver une stratégie meilleure que la stratégie naïve. De plus, l'algorithme semble

	$K = 100$	$K = 150$
Espace entier	1 059 873	826 981
Espace réduit	806 871	811 619

TABLEAU 6.12 – Meilleur seuil trouvé selon le nombre d'itérations K et l'espace de stratégies initial pour le MRAS avec $\alpha = 1{,}02$, $\beta = 1{,}0205$, $\lambda = 0{,}4$, $\nu = 0{,}5$ et $\varepsilon = 1$

	$K = 100$	$K = 150$
Espace entier	42	31
Espace réduit	100	78

TABLEAU 6.13 – Dernière itération utile selon le nombre d'itérations K et l'espace de stratégies initial pour le MRAS avec $\alpha = 1{,}02$, $\beta = 1{,}0205$, $\lambda = 0{,}4$, $\nu = 0{,}5$ et $\varepsilon = 1$

ne pas plus chercher avec cette réduction d'états, comme en témoignent les dernières itérations utiles. Cependant, il est notable qu'en restreignant l'ensemble des stratégies, l'algorithme semble plus efficace. Il se peut donc qu'une stratégie optimale se trouve dans ce sous-ensemble restreint.

Nous pouvons donc tirer de ces tests trois enseignements.

1. La stratégie naïve de référence n'est pas optimale : l'algorithme MRAS a trouvé mieux.
2. L'algorithme MRAS est potentiellement très lent par sa convergence d'une part, et par sa vitesse d'exécution d'autre part (les itérations sont de plus en plus longues).
3. Compte tenu de la géométrie du problème, l'algorithme MRAS ne peut pas être considéré comme fiable.

Devant ces résultats, nous avons décidé de tester un autre algorithme d'optimisation : l'algorithme ASA.

6.2.4 Algorithme ASA

Dans tous les tests effectués avec l'algorithme ASA, sauf mention explicite contraire, nous avons fixé :

— le nombre initial de stratégies simulées N_0 à 100,

— le nombre initial de simulations pour la méthode de Monte-Carlo M_0 à 5000,

— la température initiale T_0 à 2.

De même, nous avons gardé l'agrégation d'états décrite à la sous-partie précédente.

Nous avons exécuté l'algorithme deux fois : une en fixant $K = 100$ itérations, et l'autre en fixant $K = 200$. Le tableau 6.14 répertorie les résultats (nous avons mis en regard les tests effectués avec le même nombre d'itérations mais sans l'agrégation d'états).

L'agrégation d'états que nous avons opérée convient vraiment à l'algorithme ASA qui, dans un cas comme dans l'autre, a induit une stratégie bien meilleure que la stratégie naïve de référence, et plus rapidement que le MRAS (chaque itération de l'algorithme ASA, avec nos paramètres et l'agrégation d'états, prend environ 45 secondes). Cependant, deux remarques sont à faire.

	$K = 100$	$K = 200$
Avec agrégation d'états	782 790	796 270
Sans agrégation d'états	908 190	831 180

Tableau 6.14 – Coût moyen de stratégies simulées selon la matrice de probabilités donnée par l'ASA

1. L'algorithme ASA, comme l'algorithme MRAS, rend une matrice P qui est une approximation de la matrice optimale P^*, puisque la convergence de ces algorithmes est asymptotique. Il faut donc un nombre d'itérations adapté pour obtenir une bonne stratégie. Ceux que nous avons choisis ne sont pas suffisamment élevés, mais conviennent pour le propos de cette section.
2. La remarque précédente et l'écart-type du coût (estimé dans ce cas à environ 10 000) expliquent pourquoi on a obtenu une moins bonne performance avec 200 itérations plutôt qu'avec 100 itérations. L'algorithme ASA étant stochastique, des phénomènes de ce type peuvent survenir.

On pourrait tester l'algorithme ASA avec d'autres agrégations d'états. Par exemple, on peut considérer une agrégation des états du stock de SRM comme celle décrite dans le tableau 6.15 et qui ne correspond à rien de concret.

Nombre d'unités en stock	Code
0 ou 1	1
2, 3, 5, 6, 7 ou 8	2
4	3

Tableau 6.15 – Table de codage pour l'agrégation des états du stock de SRM

Nous avons testé l'algorithme en le laissant tourner tant que les performances semblaient évoluer. Le tableau 6.16 répertorie le nombre d'itérations nécessaires et le coût moyen. À titre de curiosité, nous avons également fait tourner l'ASA sur les 30 ans normalement requis.

Horizon	Nombre d'itérations	Coût moyen	Coût de la meilleure stratégie naïve
10 ans	200	795 000	809 540
30 ans	900	2 301 500	2 331 700

Tableau 6.16 – Coût moyen de stratégies simulées selon la matrice de probabilités rendue par l'ASA en fonction de l'horizon

On constate que, malgré le codage peu sensé, l'algorithme ASA a tout de même rendu une matrice qui génère d'aussi bonnes stratégies que précédemment et il a convergé rapidement (le test a duré environ 2 heures pour l'horizon de 10 ans, et un peu moins de 2 jours pour l'horizon

de 30 ans).

Enfin, nous avons effectué une troisième série de tests : nous avons repris le codage du stock de SRM du tableau 6.11 et nous avons changé le codage des trois autres stocks (cf. tableau 6.17), de sorte qu'il corresponde à trois états qualitatifs « vide », « plein » et « ni vide ni plein ». Puis nous avons comparé les performances des deux algorithmes proposés. Les résultats sont repris dans les tableaux 6.18 et 6.19.

IMC, LLPM ou ULPM en stock	Code
0	1
1, 2 ou 3	2
4	3

TABLEAU 6.17 – Table de codage pour l'agrégation des états pour les stocks de sous-assemblages

	$K = 100$	$K = 150$
Espace entier	865 324	974 816
Espace réduit	797 656	780 054

TABLEAU 6.18 – Meilleur seuil trouvé selon le nombre d'itérations K et l'espace de stratégies initial pour le MRAS avec $\alpha = 1{,}02$, $\beta = 1{,}0205$, $\lambda = 0{,}4$, $\nu = 0{,}5$ et $\varepsilon = 1$

Itérations	$K = 100$	$K = 150$
Espace entier	792 833	784 591
Espace réduit	727 136	724 899

TABLEAU 6.19 – Coût moyen de stratégies simulées selon la matrice de probabilités rendue par l'ASA en fonction du nombre d'itérations K et de l'espace de stratégies initial

Ces résultats nous amènent à penser que cette dernière agrégation d'états sied mieux aux deux algorithmes que l'on teste. En réduisant l'espace des stratégies, l'algorithme MRAS en a trouvé une dont la performance est meilleure que toutes celles données par les tests précédents. Néanmoins, comme on peut le constater en comparant les tableaux 6.18 et 6.19, l'algorithme ASA a trouvé une meilleure solution, et plus rapidement. En effet, il aura fallu 11 heures avec le MRAS pour obtenir ce résultat, tandis que 2 heures seulement ont été nécessaires avec l'ASA. De plus, c'est en recherchant une solution sur un espace restreint de stratégies que nous avons obtenu le coût le plus bas. Ce ne sera pas toujours forcément le cas : ici, le calendrier est très régulier, ce qui exclut d'emblée certains taux de production qui ne conviennent évidemment pas. Dans le cas général, comme nous allons le voir à la partie suivante où nous aurons affaire à des calendriers plus vraisemblables, il est difficile de préjuger d'une forme des solutions optimales. C'est donc pour cette raison que nous choisirons pour toute la suite du chapitre le sous-ensemble des stratégies induit par l'agrégation d'états codée selon les tableaux 6.11 et 6.17.

6.3 Optimisation sur 30 ans

Maintenant que nous avons pu sélectionner un algorithme pour l'optimisation, nous allons l'appliquer au problème posé dans le chapitre 3. Il est donc question de trouver les meilleurs paramètres de la chaîne de montage lorsqu'on dispose d'un calendrier de tirs sur trente années. Afin de respecter au mieux les critères pour un calendrier admissible, nous avons élaboré un programme qui en génère selon la loi sur le nombre de tirs par an du tableau 6.20. Les dates de tir dans une année sont réparties selon le tableau 6.21.

Nombre de tirs par an	6	7	8	9	10	11	12
Probabilité	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{12}$	$\frac{3}{24}$	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{6}$

Tableau 6.20 – Loi sur le nombre de tirs par an pour la génération d'un calendrier

Nombre de tirs par an	1	2	4	6	7	8	9	10	11	12
Tir 1	130	87	52	37	32	29	27	26	23	21
Tir 2		174	104	74	64	58	54	52	46	42
Tir 3			156	111	96	87	81	78	69	63
Tir 4			208	148	128	116	111	107	92	84
Tir 5				185	160	145	136	129	121	117
Tir 6				222	192	174	161	151	141	135
Tir 7					224	203	186	173	161	153
Tir 8						232	211	195	181	171
Tir 9							236	217	201	189
Tir 10								239	221	207
Tir 11									241	225
Tir 12										243

Tableau 6.21 – Dates calendaires choisies en fonction du nombre de tirs à effectuer

Notre objectif, dans cette partie, sera essentiellement de tenter de répondre à la question du stock maximal de SRM. À titre de comparaison, nous allons également indiquer la performance moyenne de la stratégie naïve (celle qui prescrit de produire le juste nombre de lanceurs). L'optimisation se fait sur 343^{65640} stratégies. Afin d'exclure celles qui ne permettent pas de faire tous les tirs demandés dans les 30 ans, nous conservons ici et jusqu'à la fin du chapitre la pénalité de 10 000 000 par tir non effectué en fin de processus.

À partir de maintenant, nous sélectionnons la meilleure stratégie simulée par l'ASA avec la matrice optimale rendue. Par abus de langage, nous dirons que l'ASA rend une stratégie.

6.3.1 Premier exemple

Considérons le calendrier représenté sur la figure 6.10. Les nombres de tirs à faire par an ne sont pas très éloignés de la moyenne de 10 tirs. Nous avons fait tourner l'ASA sur 500 itérations (ce qui a requis approximativement 8 heures) et les résultats rendus sont répertoriés dans le tableau 6.22.

On constate d'une part que l'algorithme ASA donne de meilleures stratégies, et d'autre part que permettre de stocker 8 SRM amène à un meilleur résultat (gain de 23,38% par rapport à la stratégie naïve, et de 10,21% par rapport au cas où on stocke 4 SRM). Pour autant, on peut légitimement se demander si cette conclusion est générale, ou si elle dépend du calendrier choisi. Donc, regardons ce qui se passe en prenant un autre calendrier.

6.3.2 Deuxième exemple

Considérons maintenant le calendrier représenté sur la figure 6.11. Le nombre de tirs par année est assez dispersé. Les résultats du tableau 6.23 ont été obtenus avec 500 itérations de l'ASA.

Dans ce cas, c'est un stock maximal de 4 SRM qui induit un coût plus bas (5,6% de gain par rapport au stockage de 8 SRM). Le gain par rapport à la stratégie naïve est remarquable : il est 68,83% dans le cas de 4 SRM, et de 71,98% dans le cas de 8 SRM ! Naturellement, la performance dépend du calendrier (dans ce deuxième cas, il y a un certain nombre d'années où il ne faut faire qu'entre 6 et 8 tirs, ce qui induit des coûts de retard – et de stockage – moindres).

Plus important, le stock optimal de SRM dépend également du calendrier. La dispersion des nombres de tirs à faire semble toutefois être étrangère à la question : en effet, si on programme 10 tirs par an pour les années 5 à 30, c'est le stockage de 4 SRM qui induit un meilleur coût, avec un gain de 15,41% par rapport au stockage de 8 SRM (coût de 1 889 591 contre 2 233 795).

6.3.3 Troisième exemple

Prenons cette fois le calendrier représenté sur la figure 6.12. Les nombres de tirs sont assez dispersés, mais il y a peu d'années où ils sont faibles. Pour 500 itérations de l'ASA, les résultats obtenus sont dans le tableau 6.24.

Ici, bien que le stockage de 4 SRM induise un meilleur coût, le gain par rapport à 8 SRM n'est que de 0,61% (il est de 14,36% lorsqu'on compare avec la stratégie naïve). Compte tenu de la variance des coûts, dans ce cas, on peut conclure que les deux scénarios amènent des performances similaires.

En conclusion, au travers de ces trois exemples, nous avons pu nous rendre compte que la question du stock optimal de SRM n'est pas triviale et ne semble pas pouvoir être décidée a priori. Un élément de réponse peut être apporté par l'introduction de coûts statiques (en particulier des coûts d'exploitation des bâtiments, un stock plus vaste étant naturellement plus coûteux) dont nous ne disposons pas. La tendance paraît néanmoins pencher en faveur d'un stock maximal de 4 SRM.

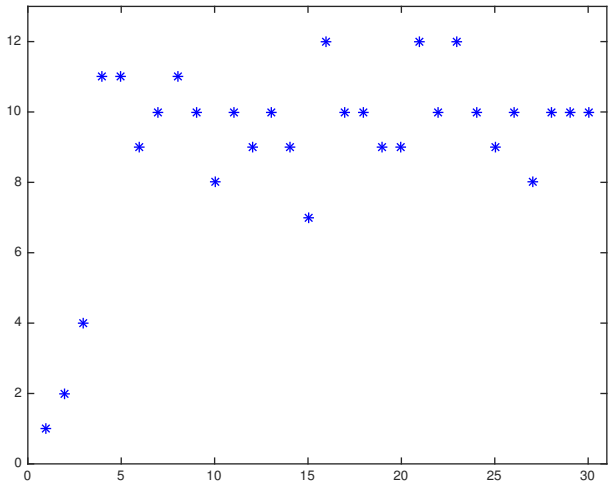

FIGURE 6.10 – Nombre de tirs par année pour le premier calendrier testé sur 30 ans

Stock maximal de SRM	4	8
ASA	2 158 990	1 938 512
Stratégie naïve	2 336 100	2 530 000

TABLEAU 6.22 – Comparaison entre les performances moyennes rendues par l'ASA et la stratégie naïve selon le stock maximal de SRM pour le premier calendrier testé

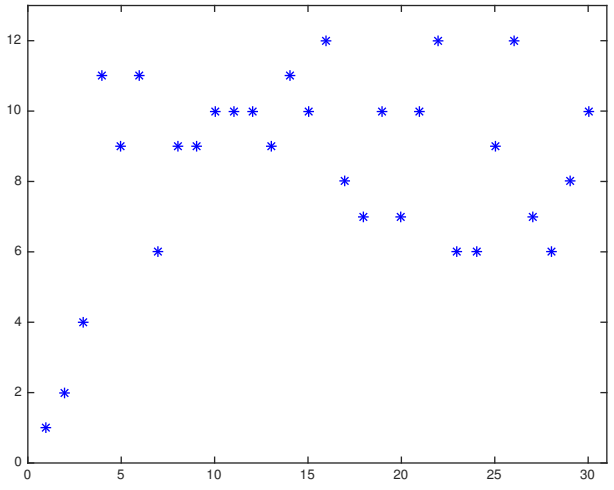

FIGURE 6.11 – Nombre de tirs par année pour le deuxième calendrier testé sur 30 ans

Stock maximal de SRM	4	8
ASA	1 452 914	1 539 095
Stratégie naïve	2 452 900	2 646 900

TABLEAU 6.23 – Comparaison entre les performances moyennes rendues par l'ASA et la stratégie naïve selon le stock maximal de SRM pour le deuxième calendrier testé

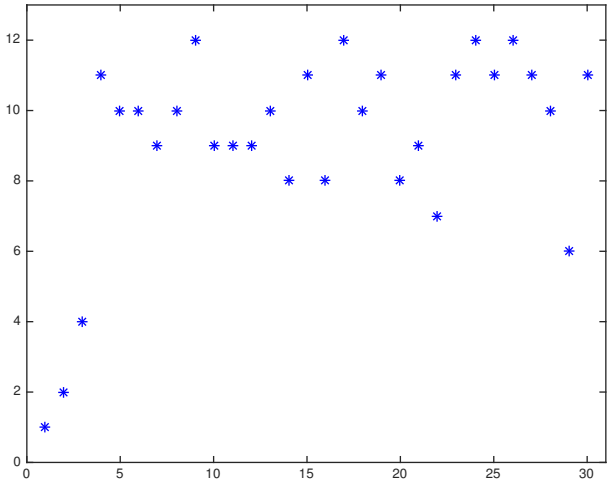

FIGURE 6.12 – Nombre de tirs par année pour le troisième calendrier testé sur 30 ans

Stock maximal de SRM	4	8
ASA	2 022 800	2 035 200
Stratégie naïve	2 362 000	2 555 700

TABLEAU 6.24 – Comparaison entre les performances moyennes rendues par l'ASA et la stratégie naïve selon le stock maximal de SRM pour le troisième calendrier testé

Enfin, il nous paraît important de préciser que les stratégies optimales rendues ne présentent pas de trait caractéristique particulier qui nous permettraient de les expliciter. Les stratégies optimales n'ont pas de forme triviale.

6.4 Compression et robustesse des stratégies optimales

Des raisons matérielles ou humaines peuvent, en cours de processus, bousculer les calendriers établis. Par exemple, un contrôle des installations, leur maintenance ou une mobilisation particulière des équipes font cesser tout fonctionnement de la chaîne. Pour autant, le nombre de tirs à faire dans l'année doit rester le même. Pour compenser ces périodes éventuelles d'inactivité, on se donne la possibilité de rapprocher localement les dates de tir prévues. C'est ce qu'on appelle une *compression de calendrier*. Lorsqu'un calendrier a subi une compression, on parle de *calendrier compressé*.

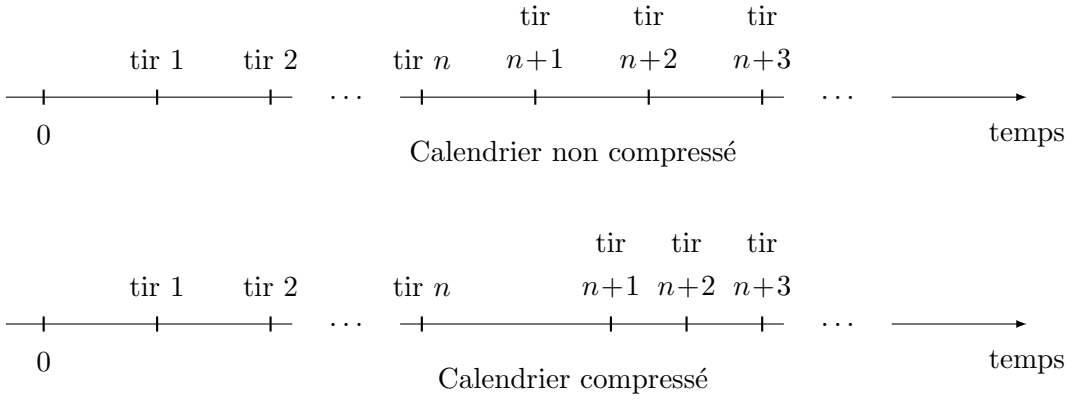

FIGURE 6.13 – Exemple de compression calendaire portant sur 3 tirs ($n+1$, $n+2$ et $n+3$)

La figure 6.13 illustre l'effet d'une compression sur le calendrier. Les dates de deux tirs ont été changées pour en rapprocher trois dans le temps (d'où le nom de compression). Lors de discussions avec Airbus ont été évoquées des compressions de 2 à 5 tirs dans une année. Ces compressions peuvent être décidées en cours d'année, alors même que la cadence de production des sous-assemblages a déjà été fixée.

Dès lors, d'aucuns pourront se demander si la stratégie optimale trouvée pour un calendrier non compressé reste optimale si une compression intervient. Naturellement, les conclusions de la partie précédente dirigent notre intuition vers une réponse négative, puisque la stratégie optimale dépend fortement du calendrier. C'est ce que nous allons examiner ici, et nous allons essayer de nous rendre compte dans quelles proportions une stratégie optimale pour un calendrier non compressé ne l'est plus en cas de compression. Pour ce faire, nous utiliserons deux calendriers. Le premier, très régulier, prescrit 10 tirs à effectuer pendant les années 5 à 30. Le deuxième sera tiré aléatoirement.

Pour tous les tests effectués dans cette section, nous avons considéré des compressions intervenant en toute fin d'année. La compression que nous opérons ramène à 15 jours l'intervalle

entre deux tirs. Nous fixons également la capacité du stock maximal de SRM à 4 unités.

Enfin, par souci de commodité, nous adopterons la notation suivante pour désigner une compression :

$$\begin{pmatrix} n_1 & n_2 & \dots & n_k \\ m_1 & m_2 & \dots & m_k \end{pmatrix}$$

où les n_i désignent les numéros des années où il y a une compression, et pour tout $1 \leq i \leq k$, m_i désigne le nombre de tirs compressés pour l'année n_i. Par exemple, $\begin{pmatrix} 6 & 10 & 18 \\ 2 & 3 & 2 \end{pmatrix}$ désigne une compression de 2 tirs la 6$^{\text{e}}$ année, de 3 tirs la 10$^{\text{e}}$ année et de 2 tirs la 18$^{\text{e}}$ année, les autres années restant inchangées par rapport au calendrier de base.

6.4.1 Calendrier régulier

Rappelons que la stratégie optimale rendue par l'ASA induit un coût moyen de 1 889 591. Commençons par examiner l'effet de petites compressions en début, milieu et fin de processus.

En observant le tableau 6.25, on remarque d'emblée que la stratégie optimale pour le calendrier non compressé cesse d'être optimale lorsqu'on introduit une compression. C'est bien cohérent avec nos conclusions de la partie précédente. En revanche, il est notable que les écarts relatifs sont très faibles (inférieurs à 2%). En effet, il ne faut pas oublier que, dans notre modélisation, la prise de décision se base non pas sur les dates de tir mais sur le nombre de tirs à effectuer dans une année. Introduire une compression va alors essentiellement induire un bouleversement de la politique de stockage qui peut avoir des répercussions sur la capacité à effectuer les lancers à la date prévue. Dans notre test, les compressions touchent un petit nombre d'années, ce qui en limite la portée et explique des écarts relatifs aussi bas.

Une telle conclusion amène nécessairement à s'interroger sur l'effet de compressions plus importantes. Pour ces tests, nous avons utilisé un programme MATLAB qui génère aléatoirement des compressions sur un calendrier passé en paramètre. Le tableau 6.26 reprend les résultats obtenus. Comme l'on pouvait s'y attendre, lorsqu'on introduit plus de compressions, les écarts relatifs peuvent être bien plus élevés. Donc, a priori, la robustesse de la stratégie optimale dépend du nombre de compressions qui interviennent lors du processus.

Enfin, nous pouvons comparer la valeur optimale du cas non compressé avec celles pour les calendriers compressés. Au mieux, on obtient des performances du même ordre. Dans les cas compressés, elles sont en général plus élevées que la valeur de base de 1 889 591 (les écarts relatifs vont de - 0,21 % jusqu'à + 9,34 %). On pense immédiatement à mettre en cause la régularité du calendrier. En effet, les perturbations introduites par les compressions semblent se propager et s'accentuer.

6.4.2 Un calendrier quelconque

Nous avons tiré un calendrier aléatoirement pour cette partie. Il est représenté sur la figure 6.14.

Compression	Optimum	Performance stratégie de base	Écart relatif
$\begin{pmatrix} 6 & 7 \\ 3 & 3 \end{pmatrix}$	2 066 100	2 085 900	+ 0,96%
$\begin{pmatrix} 15 & 16 \\ 3 & 3 \end{pmatrix}$	1 885 549	1 899 400	+ 0,73%
$\begin{pmatrix} 27 & 28 \\ 3 & 3 \end{pmatrix}$	2 037 262	2 067 700	+ 1,49%
$\begin{pmatrix} 6 & 8 & 9 \\ 2 & 4 & 5 \end{pmatrix}$	2 060 142	2 094 600	+ 1,67%

Tableau 6.25 – Comparaison des performances des stratégies optimales dans les cas compressé/non compressé pour de petites compressions du calendrier régulier

Compression	Optimum	Performance stratégie de base	Écart relatif
$\begin{pmatrix} 5 & 6 & 8 & 17 & 26 \\ 5 & 4 & 5 & 2 & 4 \end{pmatrix}$	1 947 870	2 330 400	+ 19,64%
$\begin{pmatrix} 5 & 7 & 13 & 18 & 19 & 25 \\ 2 & 2 & 3 & 3 & 5 & 5 \end{pmatrix}$	2 013 732	2 046 400	+ 1,62%
$\begin{pmatrix} 6 & 11 & 13 & 18 & 20 & 26 & 30 \\ 5 & 4 & 5 & 2 & 5 & 5 & 2 \end{pmatrix}$	1 904 735	2 349 700	+ 23,36%

Tableau 6.26 – Comparaison des performances des stratégies optimales dans les cas compressé/non compressé pour des compressions importantes du calendrier régulier

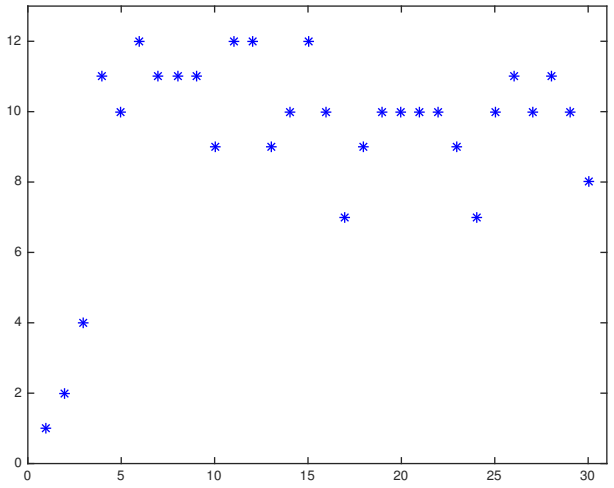

FIGURE 6.14 – Nombre de tirs par année pour le calendrier quelconque testé pour les compressions

La stratégie optimale rendue par l'ASA pour ce calendrier induit un coût de 1 777 454. Comme pour le calendrier régulier, nous avons commencé par effectuer des tests avec des petites compressions. Sur le tableau 6.27, les résultats montrent que les écarts relatifs, bien que toujours faibles (moins de 5,5%), sont plus élevés que pour le calendrier précédent. De petites compressions ont donc une portée limitée, légèrement amplifiée par la dispersion des dates de tir du calendrier choisi.

En revanche, selon le tableau 6.28, lorsqu'on introduit de grandes compressions, les écarts relatifs restent bas (moins de 7%). Les perturbations générées par les compressions sont comme absorbées par la géométrie irrégulière du calendrier. Cette observation est corroborée par les valeurs optimales trouvées par l'ASA pour les calendriers compressés : elles restent autour de la valeur optimale de base de 1 777 454. En effet, les écarts relatifs s'étalent de - 4,44% à + 3,15%, donc inférieurs à 4,5% en valeur absolue. Donc, dans ce cas, compresser les tirs n'amène pas des conséquences trop importantes. Les résultats tranchent radicalement avec ceux obtenus pour le calendrier précédent : il est donc raisonnable de penser que la robustesse des stratégies dépend de la dispersion du nombre de tirs à faire par an. Un calendrier aux nombres de tirs dispersés semble plus à même d'absorber les perturbations dues aux compressions.

6.5 Conclusion

Nous avons comparé les performances du MRAS et de l'ASA sur le problème posé par Airbus. Tant sur la qualité des résultats rendus que sur la vitesse d'exécution de l'algorithme, l'ASA nous a paru être la procédure la plus adaptée à notre cas. Devant la complexité du problème et les résultats rendus par les algorithmes, nous avons engagé une réflexion sur une nouvelle réduction

Compression	Optimum	Performance stratégie de base	Écart relatif
$\begin{pmatrix} 6 & 7 \\ 3 & 3 \end{pmatrix}$	1 698 823	1 779 100	+ 4,73%
$\begin{pmatrix} 15 & 16 \\ 3 & 3 \end{pmatrix}$	1 698 569	1 788 200	+ 5,28%
$\begin{pmatrix} 27 & 28 \\ 3 & 3 \end{pmatrix}$	1 759 280	1 818 100	+ 3,34%
$\begin{pmatrix} 6 & 8 & 9 \\ 2 & 4 & 5 \end{pmatrix}$	1 720 572	1 781 900	+ 3,56%

Tableau 6.27 – Comparaison des performances des stratégies optimales dans les cas compressé/non compressé pour de petites compressions du calendrier quelconque

Compression	Optimum	Performance stratégie de base	Écart relatif
$\begin{pmatrix} 5 & 6 & 8 & 17 & 26 \\ 5 & 4 & 5 & 2 & 4 \end{pmatrix}$	1 833 446	1 878 200	+ 2,44%
$\begin{pmatrix} 5 & 7 & 13 & 18 & 19 & 25 \\ 2 & 2 & 3 & 3 & 5 & 5 \end{pmatrix}$	1 713 077	1 832 700	+ 6,98%
$\begin{pmatrix} 6 & 11 & 13 & 18 & 20 & 26 & 30 \\ 5 & 4 & 5 & 2 & 5 & 5 & 2 \end{pmatrix}$	1 745 097	1 864 000	+ 6,81%

Tableau 6.28 – Comparaison des performances des stratégies optimales dans les cas compressé/non compressé pour des compressions importantes du calendrier régulier

de l'espace des états. Après l'avoir réduit aux seules variables pertinentes pour la décision, nous avons envisagé que la décision pourrait se baser non pas sur une information quantitative mais plutôt sur une information qualitative du niveau des stocks. Nous avons donc agrégé ces états quantitatifs pour n'en retenir que trois par stock, ce qui a ramené l'espace des états à 2281 éléments. Nous avons testé plusieurs agrégations d'états, et nous avons sélectionné la plus pertinente.

Dès lors, nous avons pu tenter de répondre au problème d'optimisation d'Airbus. Comme il n'a pas de solution mathématique formelle, nous avons eu besoin d'un critère pour estimer la qualité des stratégies rendues par l'ASA. Nous les avons donc comparées aux stratégies dites naïves. Nous obtenons des écarts relatifs parfois considérables (jusqu'à près de 70 %). L'ASA nous a alors paru être une méthode d'optimisation de bonne qualité qui rendait des stratégies utilisables en pratique. L'examen de la question du stock maximal de SRM a mis en lumière un fait surprenant : il dépend fortement du calendrier choisi. Nous avons rencontré tous les cas possibles, même si un stock maximal de 4 SRM semble convenir pour une petite majorité de calendriers testés. Donc, en fin de compte, nous avons été capables d'apporter des éléments de réponse à la problématique d'Airbus. Cet outil peut maintenant permettre d'aller plus loin en examinant des problèmes plus typiquement industriels, comme la robustesse des stratégies optimales. Nous nous sommes tournés vers les compressions de calendrier. La conclusion la plus raisonnable à tirer de nos tests est que la robustesse d'une stratégie optimale dépend de la dispersion du nombre de tirs.

L'industrialisation de cette technique d'optimisation est la suite logique à ce chapitre. En effet, maintenant que nous disposons d'un algorithme plutôt efficace et utilisable par l'industrie, il va s'agir de transformer la procédure pour qu'elle puisse s'adapter à des problèmes similaires, comme des chaînes de montage plus complexes.

Troisième partie

Une application théorique

7 Quantification d'une variable aléatoire

C'est au sein de la théorie du signal qu'est née la notion de quantification dans les années 1950. Il s'agissait alors de discrétiser un signal continu par un nombre fini de codes (ou quantifieurs). Ces codes devaient être choisis judicieusement de sorte à minimiser l'erreur commise. Désormais, la théorie de la quantification s'est liée aux probabilités, et on parle de quantification de lois de probabilité et de chaînes de Markov. Il s'agit maintenant de trouver la meilleure approximation d'une loi de probabilité quelconque par une loi de probabilité discrète à support fini (c'est-à-dire une combinaison de masses de Dirac). Longtemps limitée au cas unidimensionnel, la puissance de calcul actuelle permet d'effectuer la quantification de lois en plus grande dimension.

Désormais, la question a été largement étudiée et de nouveaux résultats et algorithmes ont été établis. Parmi les références les plus notables, on trouve les articles [Pag98], [PPP04] ou bien le manuel [GL00]. Nous nous proposons alors de résumer dans ce chapitre les notions essentielles pour bien comprendre cette théorie très féconde et utilisée maintenant dans de nombreux domaines, en particulier dans l'approximation de problèmes d'optimisation que nous allons aborder dans le chapitre 8. Nous donnerons les résultats de convergence les plus importants, sans leur preuve toutefois pour ne pas alourdir notre propos. On les trouvera dans les références citées au-dessus. En vue du chapitre 8, dans lequel nous allons faire grand usage des résultats de convergence présentés ici, nous allons également évoquer le procédé de quantification d'une chaîne de Markov, qui requiert d'être plus précautionneux à cause des qualités intrinsèques d'un tel processus aléatoire.

Dans un premier temps, nous parlerons des cellules de Voronoï, qui sont un élément important de la quantification. Ensuite, nous parlerons de la quantification d'une variable aléatoire et nous

enchaînerons sur l'algorithme de Kohonen, qui est un algorithme stochastique qui produit de bons résultats. Enfin, nous exposerons les principes de la quantification des chaînes de Markov.

Dans toute la suite, pour un espace probabilisé $(\Omega, \mathcal{F}, \mathbb{P})$, on notera $\mathrm{L}^p(\Omega, \mathcal{F}, \mathbb{P})$ l'ensemble des variables aléatoires X définies sur Ω à valeurs dans $\mathbb{R}^d$ telles que l'on ait $\|X\|_p^p = \mathbb{E}[|X|^p] < +\infty$ avec $\mathbb{E}$ l'espérance associée à $\mathbb{P}$. On fixe pour la suite un réel $p > 1$.

7.1 Les cellules de Voronoï

Soit une N-grille de $\mathbb{R}^d$ (c'est-à-dire un ensemble de points distincts de $(\mathbb{R}^d)^N$) notée $\Gamma = \{x_1, \ldots, x_N\}$. On définit une famille d'ensembles $(\overline{C}_i(\Gamma))_{1 \leq i \leq N}$ par

$$\overline{C}_i(\Gamma) = \left\{ \xi \in \mathbb{R}^d \;\middle|\; |\xi - x_i| = \min_{j \in [\![1;N]\!]} |\xi - x_j| \right\}.$$

Chacun de ces ensembles $\overline{C}_i(\Gamma)$ est donc associé à un point x_i de Γ et comprend tous les points de $\mathbb{R}^d$ pour lesquels x_i est le point de Γ le plus proche. Cette famille d'ensembles est appelée *mosaïque de Voronoï* ou *cellules fermées de Voronoï*. Par exemple, la figure 7.1 illustre une mosaïque de Voronoï et la grille à laquelle elle est associée.

Une mosaïque de Voronoï couvre $\mathbb{R}^d$, c'est-à-dire $\bigcup_{i=1}^{N} \overline{C}_i(\Gamma) = \mathbb{R}^d$. À l'instar des cellules fermées (qui sont des fermés dans $\mathbb{R}^d$), on introduit les *cellules ouvertes de Voronoï* qui sont définies par

$$C_i^o(\Gamma) = \left\{ \xi \in \mathbb{R}^d \;\middle|\; |\xi - x_i| < \min_{i \neq j} |\xi - x_j| \right\}.$$

Ce sont des ouverts de $\mathbb{R}^d$ disjoints deux à deux. Toutefois, elles ne couvrent pas $\mathbb{R}^d$.

Définition 7.1.0.1. On appelle *partition de Voronoï* associée à une N-grille Γ toute partition $(C_i(\Gamma))_{1 \leq i \leq N}$ de $\mathbb{R}^d$ telle que $C_i(\Gamma) \subset \overline{C}_i(\Gamma)$ pour tout i.

Remarque. On a aussi $C_i^o(\Gamma) \subset C_i(\Gamma)$.

Proposition 7.1.0.2. *Soit $(C_i(\Gamma))_{1 \leq i \leq N}$ une partition de Voronoï.*

(a) $\overline{C}_i(\Gamma)$ et $C_i^o(\Gamma)$ sont convexes dans $\mathbb{R}^d$.

(b) On a que $\mathrm{int}(\overline{C}_i(\Gamma)) = C_i^o(\Gamma)$, $\mathrm{adh}(C_i^o(\Gamma)) = \overline{C}_i(\Gamma)$ et $\partial C_i(\Gamma) = \partial \overline{C}_i(\Gamma) = \partial C_i^o(\Gamma)$.

(c) $\ell_d(\partial C_i(\Gamma)) = 0$.

7.2 Principe de la quantification et convergence

Soit une variable aléatoire $X \in \mathrm{L}^p(\Omega, \mathcal{F}, \mathbb{P})$ de loi de probabilité $\mathbb{P}_X$. L'idée principale de la quantification est de remplacer X par une variable aléatoire qui prend un nombre fini de valeurs de sorte à minimiser l'erreur en norme p entre X et sa version quantifiée. Autrement dit, on veut résoudre le problème de minimisation suivant :

$$\min \left\{ \|X - Y\|_p \;\middle|\; Y : \Omega \to \mathbb{R}^d \text{ mesurable et } \#Y(\Omega) \leq N \right\}. \tag{7.1}$$

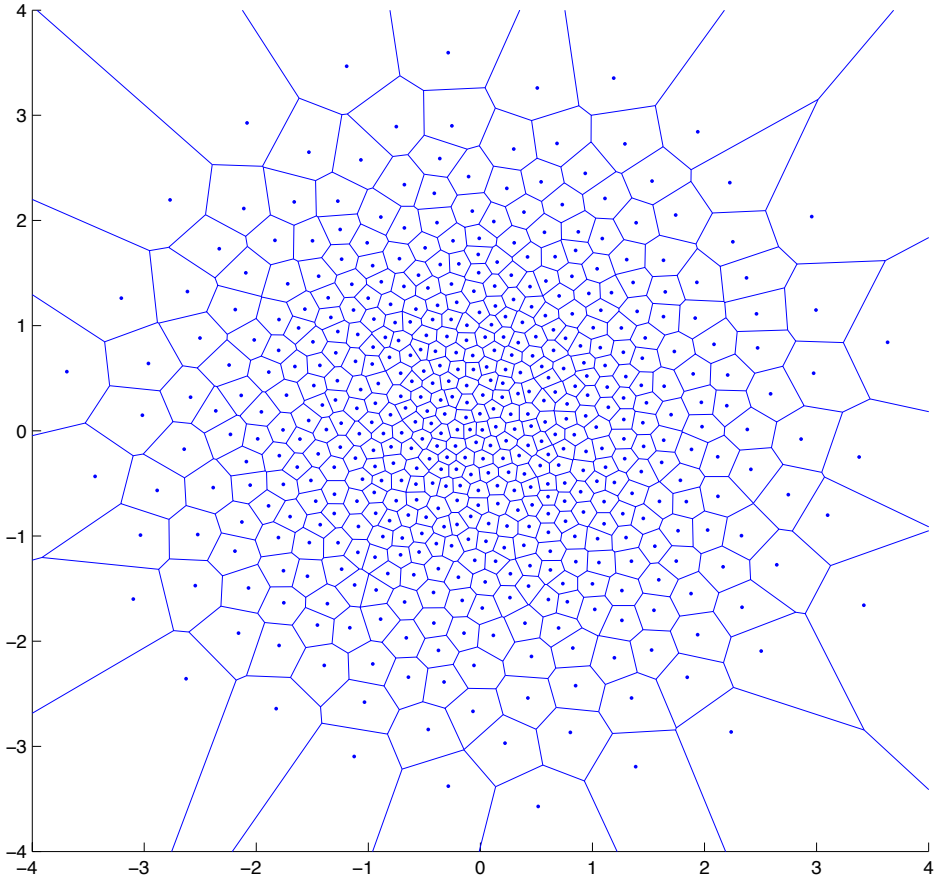

FIGURE 7.1 – Cellules de Voronoï d'une grille de 500 points dans $\mathbb{R}^2$

Soient une N-grille $\Gamma = \{x_1, \ldots, x_N\}$ de $\mathbb{R}^d$ et $(C_i(\Gamma))_{1\leq i\leq N}$ une partition de Voronoï associée à Γ. On appelle *quantifieur de Voronoï* de X la variable aléatoire X_N (ou X_N^Γ s'il y a ambiguïté) à valeurs dans Γ définie par

$$X_N = \sum_{i=1}^{N} x_i \mathbb{1}_{C_i(\Gamma)}(X).$$

Le quantifieur de Voronoï est la projection de X sur la grille Γ selon le plus proche voisin. La loi d'une telle variable aléatoire X_N est donc $\sum_{i=1}^{N} p_i \delta_{x_i}$ où

$$p_i = \mathbb{P}\{X_N = x_i\} = \mathbb{P}\{X \in C_i(\Gamma)\} = \mathbb{P}_X(C_i(\Gamma)).$$

Les p_i sont les *masses des cellules de Voronoï*. On identifiera Γ avec le vecteur $x = (x_1, \ldots, x_N)$ de $(\mathbb{R}^d)^N$ dans toute la suite et on parlera donc indifféremment de Γ ou de x. On notera les cellules de Voronoï $C_i(\Gamma)$ ou $C_i(x)$ selon le cas.

L'erreur commise lors de cette projection, appelée *distorsion*, dépend de x et s'écrit

$$D_N^X(x) = \mathbb{E}[|X - X_N|^p] = \sum_{k=1}^{N} \mathbb{E}\left[|X - x_i|^p \, \mathbb{1}_{C_i(x)}(X)\right]$$
$$= \mathbb{E}\left[\min_{i\in[\![1;N]\!]} |X - x_i|^p\right] = \int_{\mathbb{R}^d} d_N(x,u)\mathbb{P}_X(\mathrm{d}u). \tag{7.2}$$

où $d_N : (\mathbb{R}^d)^N \times \mathbb{R}^d \to \mathbb{R}_+$ est la *distorsion locale* définie par

$$d_N(x,u) = \min_{i=1,\dots,N} |x_i - u|^p .$$

La question que l'on se pose à ce stade est de savoir si cette fonction admet un minimum et si oui, comment ce minimum se comporte lorsque N tend vers l'infini. On dira que X_N est un *quantifieur optimal de* X si

$$\mathbb{E}[|X - X_N|^p] = \inf_{\Gamma\in(\mathbb{R}^d)^N} D_N^X(x) = \Delta_N^X.$$

On a alors la proposition suivante.

Proposition 7.2.0.1. *Pour toute N-grille Γ de $\mathbb{R}^d$, on a*

$$D_N^X(x) = \inf\left\{\mathbb{E}[|X - Y|^p] \;\middle|\; Y : \Omega \to \mathbb{R}^d \text{ mesurable et } Y(\Omega) \subset \{x_1, \dots, x_N\}\right\} \tag{7.3}$$

et alors

$$\Delta_N^X = \inf\left\{\mathbb{E}[|X - Y|^p] \;\middle|\; Y : \Omega \to \mathbb{R}^d \text{ mesurable et } \#Y(\Omega) \le N\right\}. \tag{7.4}$$

L'étude de la distorsion est nécessaire pour démontrer l'existence d'un quantifieur optimal. À cet effet, on a la proposition suivante, qui affirme son existence et en donne une caractérisation.

Proposition 7.2.0.2. *La fonction D_N^X est continue sur $(\mathbb{R}^d)^N$ et atteint son minimum. De plus, si on a que $\#\mathrm{supp}(\mathbb{P}_X) > N$, alors ce minimum est atteint en un N-uplet $(x_1^*, \dots, x_N^*)$ de $(\mathbb{R}^d)^N$ tel que $x_i^* \in \mathrm{conv}(\mathrm{supp}(\mathbb{P}_X))$ et $x_i^* \neq x_j^*$ pour $i \neq j$, et on a $\Delta_{k+1}^X < \Delta_k^X$ pour $1 \le k \le N$.*

Le résultat suivant de différentiabilité de la distorsion nous donne une piste pour une méthode de recherche numérique d'un quantifieur optimal.

Proposition 7.2.0.3. *(a) La fonction D_N^X est continûment différentiable en tout point $x = (x_1, \dots, x_N)$ de $(\mathbb{R}^d)^N$ tel que $x_i \neq x_j$ pour $i \neq j$ et $\mathbb{P}_X\left(\bigcup_{i=1}^N \partial C_i(x)\right) = 0$. Son gradient est donné par*

$$\nabla D_N^X(x) = \mathbb{E}[\nabla_x d_N(x, X)] \tag{7.5}$$

où

$$\nabla_x d_N(x,u) = \left(\frac{\partial d_N}{\partial x_i}(x,u)\right)_{1\le i\le N} = p\left(\frac{x_i - u}{|x_i - u|}\,|x_i - u|^{p-1}\,\mathbb{1}_{C_i(x)}(u)\right)_{1\le i\le N}.$$

avec la convention $\frac{0}{|0|} = 0$. Si $\mathbb{P}_X$ est absolument continue par rapport à la mesure de Lebesgue, la formule (7.5) est valable pour $p = 1$.

(b) Si $\mathbb{P}_X$ est absolument continue par rapport à la mesure de Lebesgue, D_N^X est différentiable en tout $x = (x_1, \dots, x_N) \in (\mathbb{R}^d)^N$ tel que $x_i \neq x_j$ pour $i \neq j$ et on a

$$\operatorname*{argmin}_{(\mathbb{R}^d)^N} D_N^X \subset \left\{\nabla D_N^X = 0\right\}. \tag{7.6}$$

On est donc assurés de l'existence (mais pas de l'unicité !) d'une grille qui minimise la distorsion. D'ailleurs, on dit qu'un quantifieur de Voronoï X_N^Γ de X est *stationnaire* lorsque la grille associée (qu'on identifie au vecteur x de $(\mathbb{R}^d)^N$) satisfait la relation

$$\nabla D_N^X(x) = 0.$$

Un quantifieur optimal est donc stationnaire (mais la réciproque est fausse en général). Les quantifieurs stationnaires possèdent la propriété suivante.

Proposition 7.2.0.4. *Si X_N est un quantifieur stationnaire de X, alors $\mathbb{E}[X \mid X_N] = X_N$.*

Occupons-nous des questions de convergence. La convergence de la distorsion minimale est donnée par la proposition suivante.

Proposition 7.2.0.5. *La suite $(\Delta_N^X)_N$ converge et on a*

$$\lim_{N \to +\infty} \Delta_N^X = 0. \tag{7.7}$$

Si ce résultat de convergence est facile à montrer, il est beaucoup plus ardu d'estimer la vitesse de convergence. Cette question a trouvé une réponse à l'heure actuelle, mais sa résolution a nécessité plusieurs étapes. On a donc le théorème suivant, que l'on peut trouver dans [GL00] :

Théorème 7.2.0.6. *Supposons que $\mathbb{E}[|X|^{p+\beta}] < +\infty$ pour un certain $\beta > 0$. Alors*

$$\lim_{N \to +\infty} \left(N^{\frac{p}{d}} \min_{\#\Gamma \leq N} \left\| X - X_N^\Gamma \right\|_p^p \right) = J_{p,d} \left(\int_{\mathbb{R}^d} g^{\frac{d}{d+p}}(\xi)\, \mathrm{d}\xi \right)^{1+\frac{p}{d}} \tag{7.8}$$

où $\mathbb{P}_X(\mathrm{d}\xi) = g(\xi)\ell_d(\mathrm{d}\xi) + \nu(\mathrm{d}\xi)$ avec $\nu \perp \ell_d$, et $J_{p,d}$ une constante réelle positive.

La vitesse de convergence de Δ_N^X est donc en $\mathrm{O}(N^{-\frac{p}{d}})$. On sait peu de choses sur la valeur de la constante $J_{p,d}$. Mais dans [PPP04], les auteurs donnent des informations sur sa valeur en dimension 1 : elle vaut $J_{p,1} = \dfrac{1}{2^p(p+1)}$. En dimension plus grande, on ne sait que donner un équivalent :

$$J_{p,d} \underset{d \to +\infty}{\sim} \left(\frac{d}{2\pi \mathrm{e}} \right)^{\frac{p}{2}}.$$

Notons qu'on en connaît aussi une valeur particulière : $J_{2,2} = \dfrac{5}{18\sqrt{3}}$.

7.3 Comment obtenir une grille optimale ?

On recherche donc un N-uplet $x = (x_1, \ldots, x_N) \in (\mathbb{R}^d)^N$ qui réalise le minimum de la distorsion Δ_N^X. L'équation (7.6) nous incite en rechercher un parmi les quantifieurs stationnaires de X, qui sont des minima locaux.

Sauf dans un petit nombre de cas de peu d'intérêt, on ne sait pas décrire ni donner la forme générale d'une grille optimale. Toutefois, il existe des algorithmes qui permettent d'en avoir une bonne approximation. Les premières recherches ont été menées dans le cas d'une approximation L^2 de variables réelles et ont abouti à l'algorithme dit de Lloyd (cf. [Pha07], section 1.4.1). Derrière cet algorithme se cache une méthode de point fixe. Cette méthode n'étant implémentable qu'en dimension 1, nous allons plutôt nous attarder sur un autre algorithme, plus général, basé sur une méthode de gradient stochastique. Il s'agit de l'algorithme dit de Kohonen (cf. [Pha07] section 1.4.3). C'est celui que nous utilisons pour l'application numérique du chapitre 8.

7.3.1 Principe

L'algorithme de Kohonen se base sur la représentation intégrale du gradient de la distorsion, c'est-à-dire

$$\nabla D_N^X(x) = \int_{\mathbb{R}^d} \nabla_x d_N(x, u)\, \mathrm{d}\mathbb{P}_X(u).$$

On suppose savoir simuler la loi $\mathbb{P}_X$. On considère donc la suite (x_n) de variables aléatoires à valeurs $(\mathbb{R}^d)^N$ définie par récurrence par

$$x_{n+1} = x_n + \frac{\gamma_n}{d} \nabla_x d_N(x_n, Y_{n+1}) \tag{7.9}$$

où (γ_n) est une suite de pas positifs telle que

$$\sum_{n\geq 0} \gamma_n = +\infty \text{ et } \sum_{n\geq 0} \gamma_n^2 < +\infty \tag{7.10}$$

et où (Y_n) est une suite de variables aléatoires indépendantes et identiquement distribuées de loi $\mathbb{P}_X$ indépendante de x_0. Cette suite récurrente fournit à chaque étape une nouvelle grille de points distincts. Sous des hypothèses appropriées sur la distorsion et sur la loi $\mathbb{P}_X$, la suite (x_n) converge presque sûrement vers une grille qui en réalise un minimum local.

Seulement, dans le cas général, D_N^X ne vérifie pas ces hypothèses (en particulier, ∇D_N^X devrait être lipschitzienne, par exemple), ce qui pose ces problèmes en théorie. En pratique toutefois, l'algorithme de Kohonen fournit des résultats satisfaisants. De plus, en affaiblissant les hypothèses, on peut obtenir des résultats de convergence presque sûre dans le cas quadratique (i.e. lorsque $p = 2$). Ce cas est par conséquent implémenté le plus souvent lors des applications, et l'algorithme porte alors le nom de CLVQ (*Competitive Learning Vector Quantization*, cf. [PPP04] section 2.2).

Explicitons plus en détails la procédure à suivre autour de la formule récursive (7.9). On peut en distinguer trois étapes. Soit la grille $x_n = (x_{n,1}, \ldots, x_{n,N})$ calculée à la n^{e} itération.

Étape 1 : *phase de compétition.* On simule une variable aléatoire Y_{n+1} de loi $\mathbb{P}_X$ et on sélectionne l'indice i_{n+1} tel que $i_{n+1} \in \underset{i\in[\![1;N]\!]}{\operatorname{argmin}} |Y_{n+1} - x_{n,i}|$. C'est l'indice du point de la grille x_n le plus proche de Y_{n+1}.

Étape 2 : *phase d'apprentissage.* On met à jour la grille x_n de la manière suivante :

$$\begin{cases} x_{n+1,i_{n+1}} = x_{n,i_{n+1}} - \gamma_{n+1} \dfrac{x_{n,i_{n+1}} - Y_{n+1}}{\left|x_{n,i_{n+1}} - Y_{n+1}\right|} \left|x_{n,i_{n+1}} - Y_{n+1}\right|^{p-1} \\ x_{n+1,i} = x_{n,i} \text{ si } i \neq i_{n+1} \end{cases} . \tag{7.11}$$

Étape 3 : *phase de mise à jour.* Cette phase facultative est dédiée à la mise à jour éventuelle de l'estimation des masses des cellules de Voronoï associées à la grille x_{n+1} et à celle de la distorsion minimale. Posons $p_{n,i}$ la masse de la cellule associée à $x_{n,i}$ et d_n l'estimation de la distorsion minimale calculées à la n^{e} itération. La mise à jour se fait selon les formules suivantes :

$$\begin{cases} p_{0,i} = \dfrac{1}{N} \\ p_{n+1,i} = p_{n,i} - \gamma_{n+1}(p_{n,i} - \delta_{i_{n+1}}(i)) \end{cases} \quad \text{pour tout } i \in [\![1;N]\!] \tag{7.12}$$

$$\begin{cases} d_0 = 0 \\ d_{n+1} = d_k - \gamma_{n+1}(d_k - \left|x_{n,i_{n+1}} - Y_{n+1}\right|^r) \text{ où } r \in]0;p]. \end{cases} \tag{7.13}$$

La proposition suivante justifie l'étape 3.

Proposition 7.3.1.1. *On suppose que la loi $\mathbb{P}_X$ de X est absolument continue par rapport à la mesure de Lebesgue. On suppose également que $\mathbb{E}[|X|^{p+\beta}] < +\infty$ pour un certain $\beta > 0$. Alors, sachant l'événement $\left\{x_k \underset{k\to+\infty}{\longrightarrow} x^*\right\}$, $p_{k,i} \underset{k\to+\infty}{\longrightarrow} \mathbb{P}_X(C_i(x^*))$ presque sûrement pour tout $i \in [\![1;N]\!]$ et $d_k \underset{k\to+\infty}{\longrightarrow} D_N^X(x^*)$ presque sûrement.*

Remarquons que ces convergences ont lieu sous l'événement $\left\{x_k \underset{k\to+\infty}{\longrightarrow} x^*\right\}$, peu importe la limite x^*, c'est-à-dire peu importe la grille retournée par l'algorithme. Les suites définies par les équations (7.12) et (7.13) sont donc consistantes.

7.3.2 Choix de la grille initiale et de la suite de pas (γ_n)

Jusqu'à présent, nous n'avons pas évoqué plus en détails la grille initiale x_0 et la suite de pas (γ_n), qui sont laissées au choix de l'utilisateur (tant que (γ_n) respecte les propriétés (7.10)). En pratique, c'est un réel problème qui se pose dans la mesure où ces choix déterminent la convergence de l'algorithme vers une solution satisfaisante. L'article [PP03] nous donne de précieuses informations sur cette question.

Le choix de la grille initiale x_0 peut provenir de deux méthodes. Lorsque N est petit (typiquement $N \geq 10$), on peut former une grille en simulant N réalisations indépendantes et de même loi $\mathbb{P}_X$. Lorsque N est plus grand, on peut avoir recours à une autre manière de choisir x_0 : il s'agit de la méthode dite *splitting-initializing.* Elle consiste à rajouter un point à une

grille optimale de taille $N-1$ (habituellement l'origine de $\mathbb{R}^d$, i.e. le quantifieur optimal de taille 1) pour obtenir une grille initiale de taille N. Toutefois, l'initialisation via cette méthode peut n'aboutir qu'à des grilles sous-optimales. Elle représente en revanche un bon compromis entre stabilité et précision.

Pour le choix de la suite (γ_n), on pense tout naturellement à prendre $\gamma_n = \dfrac{1}{n}$. Toutefois, ce choix n'est pas judicieux. En effet, pour la loi uniforme, $\gamma_n > \frac{2N^{\frac{3}{d}}}{\pi^2 n}$ est le seuil critique pour valider le théorème central limite quand n est assez grand. Donc, à partir de là, si on pose

$$\gamma_n = \gamma_0 \frac{a}{a + \gamma_0 bn} \tag{7.14}$$

avec $a = 4N^{\frac{1}{d}}$ et $b = \pi^2 N^{-\frac{2}{d}}$, alors $\gamma_n \sim \frac{2N^{\frac{3}{d}}}{\pi^2 n}$. Les expérimentations numériques montrent que ce choix de γ_n peut se révéler plus efficace, en particulier dans le cas de la quantification d'une loi normale. La figure 7.1 montre par ailleurs le résultat de la quantification d'une loi normale centrée réduite bivariée exécutée avec ce choix de la suite (γ_n). Reste maintenant à choisir γ_0. On peut le prendre égal à 1, ou s'inspirer de la méthode *splitting-initializing*. Dans ce cas, comme on initialise l'algorithme à une grille optimale à laquelle on a rajouté des points, pour garder le bénéfice des calculs déjà effectués, il est indiqué de prendre

$$\gamma_0 = \frac{1}{2} \min_{i \neq j} |x_i - x_j| .$$

Ce choix est motivé par l'inégalité

$$\min_{i \neq j} |x_i - x_j| \leq 2\sqrt{\mathbb{E}\left[|X - x_i|^2\right]}.$$

Dans nos calculs toutefois, nous choisirons $\gamma_0 = 1$. C'est aussi le choix opéré pour réaliser la figure 7.1.

7.4 Quantification d'une chaîne de Markov

Considérons maintenant une chaîne de Markov $(X_n)_{n \geq 0}$ définie sur un espace probabilisé $(\Omega, \mathcal{F}, \mathbb{P})$, à valeurs dans $\mathbb{R}^d$, de loi de transition du temps $n-1$ au temps n notée P_n et de loi initiale μ. On souhaite quantifier la chaîne (X_n), c'est-à-dire approcher la loi de (X_n) par la loi d'un processus $(X_n^{(N)})$ à valeurs dans des ensembles finis tout en prenant en compte les caractéristiques de la loi de transition. L'article [PPP04] expose deux techniques pour y parvenir : la quantification marginale et la quantification markovienne. Ces méthodes s'inspirent de la caractérisation complète d'une chaîne de Markov par sa loi initiale et sa loi de transition. Elles ont l'une et l'autre leurs avantages et leurs inconvénients. Entre autres, la quantification marginale donne de meilleurs résultats numériques mais pose des problèmes quant à la propriété de Markov, tandis que la quantification markovienne garde la propriété de Markov mais demande des hypothèses plus fortes. Le choix de l'une ou l'autre se motive par le type de problème à traiter. En l'occurrence, la quantification marginale ayant été développée pour des problèmes d'arrêt optimal, au regard du chapitre 8, c'est celle que nous allons exposer.

7.4.1 Quantification marginale

Comme son nom l'indique, cette méthode de quantification consiste à approcher la distribution de chacune des variables aléatoires X_n. Ainsi si, au rang n, on dispose d'une grille de N points $\Gamma_n = \{x_n^1, \dots, x_n^N\}$ de $\mathbb{R}^d$ de cellules de Voronoï associées $C_1(\Gamma_n), \dots, C_N(\Gamma_n)$, on définit $X_n^{(N)}$ comme la projection de X_n sur la grille Γ_n selon le plus proche voisin. Toutefois, comme la projection selon le plus proche voisin n'est pas injective, le processus $(X_n^{(N)})_{n\geq 0}$ n'est pas une chaîne de Markov.

On pose alors

$$p_{k-1}^i = \mathbb{P}\left\{X_{k-1}^{(N)} = x_{k-1}^i\right\}, \tag{7.15}$$

$$\beta_k^{i,j} = \mathbb{P}\left\{X_{k-1}^{(N)} = x_{k-1}^i \text{ et } X_k^{(N)} = x_k^j\right\} \tag{7.16}$$

et

$$p_k^{i,j} = \frac{\beta_k^{i,j}}{p_{k-1}^i} = \mathbb{P}\left\{X_k^{(N)} = x_k^j \;\middle|\; X_{k-1}^{(N)} = x_k^i\right\}. \tag{7.17}$$

Il existe une chaîne de Markov $(\tilde{X}_n^{(N)})_n$ de matrices de transition $(p_n^{i,j})_{i,j}$ et de loi initiale $(p_0^i)_i$ où chaque $\tilde{X}_n^{(N)}$ est à valeurs dans la grille Γ_n. La méthode de quantification marginale repose donc sur l'approximation de $(X_n)_n$ par la chaîne de Markov $(\tilde{X}_n^{(N)})_n$. Par définition, l'erreur de quantification $\left\|X_n - X_n^{(N)}\right\|_2$ tend vers 0 lorsque N tend vers $+\infty$ à la vitesse donnée par le théorème 7.2.0.6.

L'estimation de la qualité de ce type d'approximation est donnée par le théorème suivant, que l'on peut trouver dans [PPP04] ou [Pha07].

Théorème 7.4.1.1. *Soit $p \geq 1$. Supposons que, pour tout $1 \leq k \leq n$, la loi de transition P_k soit lipschitzienne de constante $[P_k]$, c'est-à-dire que pour toute fonction φ lipschitizenne sur $\mathbb{R}^d$ de constante $[\varphi]$,*

$$\left|\int_{\mathbb{R}^d} \varphi(y) P_k(x, \mathrm{d}y) - \int_{\mathbb{R}^d} \varphi(y) P_k(x', \mathrm{d}y)\right| \leq [P_k][\varphi]\left|x - x'\right|$$

pour tous $x, x' \in \mathbb{R}^d$. Posons $[P] = \max_{1\leq k\leq n} [P_k]$. Pour toutes fonctions φ_k, $1 \leq k \leq n$, lipschitziennes de rapport $[\varphi_k] \leq 1$ et bornées par 1,

$$\begin{aligned}&\left|\mathbb{E}\left[\varphi_0(X_0)\dots\varphi_n(X_n)\right] - \mathbb{E}\left[\varphi_0(\tilde{X}_0^{(N)})\dots\varphi_n(\tilde{X}_n^{(N)})\right]\right| \\ &\leq \sum_{k=0}^{n}\left(1 + (2 - \mathbb{1}_{\{p\}}(2))\frac{[P]^{n-k+1} - 1}{[P] - 1}\right)\left\|X_k - \tilde{X}_k^{(N)}\right\|_p\end{aligned} \tag{7.18}$$

avec la convention $\dfrac{u^m - 1}{u - 1} = m$ si $u = 1$ et $m \in \mathbb{N}$.

La faiblesse des hypothèses pour ce théorème est remarquable : la convergence ne demande que le caractère lipschitzien des transitions.

Remarque. Dans [PPP04], les auteurs donnent notamment que la vitesse de la convergence de la quantification marginale pour chaque rang n est en $\mathrm{O}\left(\dfrac{n^{1+\frac{1}{d}}}{N^{\frac{1}{d}}}\right)$.

7.4.2 Procédure

On étend donc l'algorithme CLVQ aux chaînes de Markov via la procédure suivante (c'est-à-dire que cette procédure réalise une L^2-quantification optimale), que l'on peut trouver dans [BP03] ou dans [PRS05]. En entrée, l'algorithme prend :

— un nombre de réplications S,

— une suite de pas $(\gamma_s)_{1\leq s\leq S}$ qui respectent les égalités (7.10),

— S simulations de la chaîne de Markov $(Z^1,\dots,Z^S)$ où $Z^i=(Z_0^i,\dots,Z_{N_0}^i)$ avec $Z_0^i=z_0$ un état initial et $N_0\in\mathbb{N}^*$.

Les étapes de la procédure se succèdent ainsi.

Étape 1 : *phase d'initialisation.* On construit

— N_0+1 grilles : $\Gamma_0^0=\{z_0\}$ et $\Gamma_n^0=\left\{z_n^{0,1},\dots,z_n^{0_N}\right\}$ pour tout $1\leq n\leq N_0$,

— N_0 vecteurs de poids : $p_0^0=1$ et $p_n^{0,i}=\dfrac{1}{N}$ pour tout $1\leq i\leq N$ et tout $1\leq n\leq N_0$,

— N_0 matrices de poids : $\beta_n^0=(\beta_n^{0,i,j})=(0)$ pour tout $1\leq i\leq N$, tout $1\leq j\leq N$ et tout $1\leq n\leq N_0$,

— N_0 distorsions : $D_n^0=0$ pour tout $1\leq n\leq N_0$.

Étape 2 : *phase d'apprentissage et de mise à jour.* Pour s allant de 0 à $S-1$, on exécute les instructions suivantes.

(a) **Cas $n=N_0$.**

— On sélectionne l'indice $i_{N_0}(s+1)$ tel que $i_{N_0}(s+1)\in\underset{1\leq i\leq N}{\operatorname{argmin}}\left|z_{N_0}^{s,i}-Z_{N_0}^s\right|$.

— On met à jour la grille : pour tout $1\leq i\leq N$,

$$z_{N_0}^{s+1,i}=z_{N_0}^{s,i}-\gamma_{s+1}\delta_{i_{N_0}(s+1)}(i)(z_{N_0}^{s,i}-Z_{N_0}^s).$$

— On met à jour la distorsion :

$$D_{N_0}^{s+1}=D_{N_0}^s-\gamma_{s+1}\left(D_{N_0}^s-\left|z_{N_0}^{s,i_{N_0}(s+1)}-Z_{N_0}^s\right|_2^2\right).$$

(b) **Cas n entre N_0-1 et 1.**

— On sélectionne l'indice $i_n(s+1)$ tel que $i_n(s+1)\in\underset{1\leq i\leq N}{\operatorname{argmin}}\left|z_n^{s,i}-Z_n^s\right|$.

— On met à jour la grille : pour tout $1\leq i\leq N$,

$$z_n^{s+1,i}=z_n^{s,i}-\gamma_{s+1}\delta_{i_n(s+1)}(i)(z_n^{s,i}-Z_n^s).$$

— On met à jour les poids : pour tous $1\leq i,j\leq N$,

$$p_n^{s+1,i}=p_n^{s,i}-\gamma_{s+1}(p_n^{s,i}-\delta_{i_n(s+1)}(i)),$$

$$\beta_{n+1}^{s+1,i,j}=\beta_{n+1}^{s,i,j}-\gamma_{n+1}(\beta_{n+1}^{s,i,j}-\delta_{i_n(s+1)}(i)\delta_{i_{n+1}(s+1)}(j)),$$

$$r_{n+1}^{s+1,i,j}=\frac{\beta_{n+1}^{s+1,i,j}}{p_n^{s+1,i}}.$$

— On met à jour la distorsion :

$$D_n^{s+1} = D_n^s - \gamma_{s+1}\left(D_n^s - \left|z_n^{s,i_n(s+1)} - Z_n^s\right|_2^2\right).$$

(c) **Cas $n = 0$.** On met à jour les poids : pour tout $1 \leq j \leq N$,

$$\beta_1^{s+1,1,j} = \beta_1^{s,1,j} - \gamma_{s+1}(\beta_1^{s,1,j} - \delta_{i_2(s+1)}(j)),$$

$$r_1^{s+1,1,j} = \beta_1^{s+1,1,j}.$$

L'algorithme rend en sortie les grilles, la distorsion et la matrice de transition $(r_n^{i,j})$ de la chaîne de Markov approchée.

Remarque. Jusqu'à présent, et dans l'algorithme également, nous avons considéré que toutes les grilles Γ_n étaient de même taille N. Il est tout à fait possible de quantifier une chaîne de Markov en prenant des grilles de tailles diverses. Les matrices de transition ne sont alors pas carrées.

À l'instar de la quantification d'une variable aléatoire, on peut démontrer que, conditionnellement à l'événement $\left\{\Gamma_n^s \underset{s\to+\infty}{\longrightarrow} \Gamma_n^{(N)}\right\}$, on a les convergences presque sûres suivantes :

$$p_n^{s,i} \underset{s\to+\infty}{\longrightarrow} \mathbb{P}\left\{Z_n^{(N)} = z_n^i\right\},$$

$$\beta_{n+1}^{s,i,j} \underset{s\to+\infty}{\longrightarrow} \mathbb{P}\left\{Z_{n+1}^{(N)} = z_{n+1}^j \text{ et } Z_n^{(N)} = z_n^i\right\},$$

$$D_n^s \underset{s\to+\infty}{\longrightarrow} D_n(\Gamma_n^{(N)})$$

où $D_n(\Gamma_n^{(N)})$ est la distorsion associée à la projection sur la grille $\Gamma_n^{(N)}$.

8 Arrêt optimal partiellement observable

Nous allons nous intéresser à un aspect applicatif des processus markoviens décisionnels : les problèmes d'arrêt optimal. Considérant un processus aléatoire, on cherche à trouver le moment opportun pour l'arrêter afin d'optimiser une fonction d'utilité. Ces problèmes ont été beaucoup étudiés et possèdent de très nombreuses applications dans tous les domaines. On les retrouve notamment en finance ([BR11]) : l'exercice d'options américaines en sont une parfaite illustration. Mais l'arrêt optimal concerne également l'industrie : on peut penser aux problèmes de maintenance prédictive ([dSDZE10]) par exemple. Enfin, certains jeux de société, comme le Not-Seven ([BDFN12]), consistent à arrêter le jeu pour avoir le plus de points possible. Dans ce chapitre, nous considérerons un cas particulier où le contrôleur n'a accès qu'à une partie des informations seulement : il s'agit de problèmes dits *partiellement observables*. La décision d'arrêt ne pourra alors se fonder que sur ce que le contrôleur sait de l'état courant du système. On cherchera alors le meilleur moment pour arrêter le système, mais également le gain moyen optimal espéré.

Dans ce chapitre, nous nous concentrerons sur cette dernière quantité. La théorie des POMDP (cf. chapitre 1, section 1.4) donne alors un cadre de travail qui convient particulièrement bien à notre propos. C'est dans ce formalisme que nous allons inscrire notre problème d'arrêt optimal, puis que nous allons le transformer en problème complètement observable. Pour ce faire, l'approche la plus courante est d'introduire un processus de filtrage auxiliaire à valeurs dans un espace de probabilités (appelé *belief space* en anglais). Ainsi, sur ce MDP complètement observable, on peut appliquer les outils de la théorie classique et calculer le gain moyen optimal. En revanche, le prix à payer est fort puisqu'en général, l'espace de probabilités utilisé est de dimension infinie. Il n'y a pas de manière triviale de discrétiser cet ensemble, et il faut donc trouver

un méthode qui représente un compromis raisonnable entre le temps de calcul et la précision de l'approximation.

Tandis que la littérature traitant du cas où les espaces d'états sont des ensembles de cardinal fini est abondante (on pourra par exemple lire [HF00] et [RPPCD08]), il y a encore peu de références sur le cas où les espaces d'états sont quelconques. Quelques méthodes ont été développées depuis quelques années pour tenter de résoudre des problèmes d'optimisation associés à des POMDP à espaces d'états continus. Les références les plus pertinentes sur le sujet sont [HP05, ZFM10, Zho13, YZ13]. Dans [HP05], les auteurs partitionnent l'espace des états observables par agrégation d'états pour obtenir la valeur optimale du POMDP associé. En effet, ils constatent que certains états observables amènent le contrôleur à prendre une seule et même décision. Ils proposent donc de discrétiser l'ensemble des états observables en regroupant sous un même état ceux qui mènent à prendre la même décision. Le calcul de la valeur optimale fait alors appel à des algorithmes auxiliaires non décrits dans ce travail de thèse. Cette méthode demande toutefois que l'ensemble des états non observables soit discret. Dans [ZFM10], les auteurs proposent de paramétrer l'espace de probabilités par la famille exponentielle afin d'en réduire sa dimension. Mais ils se concentrent sur des problèmes de contrôle généraux pour des critères de coût à horizon infini et des stratégies stationnaires. Ce cadre de travail ne convient pas aux problèmes d'arrêt optimal. Dans [Zho13], l'auteur s'appuie sur des propriétés de la valeur optimale (notamment des propriétés de régularité) pour en déduire un algorithme qui en calcule une approximation basée sur une méthode de filtrage et sur des simulations de trajectoires. Elle suppose que toutes les lois possèdent des densités par rapport à la mesure de Lebesgue et que la fonction de récompense est convexe. La convergence de la procédure y est démontrée, sans information sur sa vitesse. Enfin, dans [YZ13], les auteurs utilisent une formulation duale du problème d'optimisation en termes de martingales pour calculer non pas une approximation de la fonction valeur optimale, mais des bornes. Ils ne requièrent pas d'hypothèses particulières sur la chaîne de Markov si ce n'est la capacité d'en simuler des trajectoires, mais ne donnent pas de vitesse de convergence de leur procédure non plus.

Considérant un problème d'arrêt optimal partiellement observable, nous proposons dans ce chapitre de développer une méthode d'approximation par quantification de son gain moyen optimal calculable numériquement dans le cas où les espaces d'états sont quelconques. On trouvera au chapitre 7 de plus amples informations sur la quantification. Nous situerons notre travail dans le domaine des problèmes à horizon fini, c'est-à-dire que les processus considérés comporteront une suite finie de variables aléatoires. Notre méthode, inspirée de l'approche des auteurs de [PRS05], présentera l'avantage d'avoir une vitesse de convergence connue sous des hypothèses relativement faibles. L'idée motrice de [PRS05] est de quantifier simultanément le processus des observations et le processus de filtrage.

La quantification permettra l'utilisation des résultats classiques sur les processus markoviens décisionnels et le calcul de la fonction valeur optimale passera alors par ce que l'on va appeler les équations de Bellman, qui définissent une suite récurrente de fonctions. Il faudra donc travailler sur ces équations, et plus particulièrement sur l'opérateur de Bellman sur lequel elles reposent.

Le travail d'approximation que nous nous apprêtons à accomplir sera organisé en deux temps. D'une part, nous ramènerons l'espace des états non observables à un ensemble de cardinal fini afin de pouvoir utiliser des outils déjà développés. D'autre part, il nous faudra vérifier que nous pouvons utiliser ces outils. Notre but affiché est d'obtenir, à la fin, un processus ne mettant en jeu que des quantités calculables numériquement. Pour ces deux étapes, nous utiliserons donc les techniques de quantification et les résultats que l'on peut trouver dans [BP03] ou [PPP04].

Dans la section 8.1, nous définirons le problème d'arrêt optimal sur lequel nous allons travailler et nous montrerons qu'il est équivalent à un problème d'optimisation en horizon fini pour un POMDP. Puis, nous montrerons que ce problème partiellement observable est équivalent au problème d'optimisation d'un MDP complètement observable. Ensuite, dans la section 8.2, nous jetterons les bases d'une première discrétisation du MDP ainsi obtenu. Nous établirons les résultats de convergence nécessaires et nous en donnerons la vitesse. Dans la section 8.3 suivante, nous procéderons à la deuxième discrétisation du MDP, afin d'obtenir un problème calculable que l'on peut résoudre numériquement. La section 8.4 présente un exemple sur lequel nous appliquerons notre résultat.

Dans la suite, nous utiliserons les notations suivantes. Soit (E, d) un espace métrique. On définit $\mathbb{L}(E)$ comme l'ensemble des fonctions lipschitziennes bornées sur E. On munit cet ensemble de la norme $\|\cdot\|_{\mathbb{L}}$ définie par

$$\|f\|_{\mathbb{L}} = \sup_{x \in E} |f(x)| + \sup_{x \neq y} \frac{|f(x) - f(y)|}{\mathrm{d}(x, y)}.$$

On spécifie alors un sous-ensemble de $\mathbb{L}(E)$, noté $\mathbb{L}^*(E)$, défini par

$$\mathbb{L}^*(E) = \{f \in \mathbb{L}(E) \ \mid \ \|f\|_{\mathbb{L}} \leq 1\}.$$

On munira $\mathcal{P}(E)$ de la norme de Kantorovich-Rubinstein notée $\|\cdot\|_{KR}$ et définie par

$$\|\mu\|_{KR} = \sup_{f \in \mathbb{L}^*(E)} \left| \int_E f(x) \mu(\mathrm{d}x) \right|$$

pour toute mesure $\mu \in \mathcal{P}(E)$ (cf. [Bog07], section 8.3). Cette dernière est bien définie dans la mesure où (E, d) est un espace métrique.

8.1 Formulations successives du problème

8.1.1 Description et hypothèses préliminaires

Dans cette partie, on adaptera une partie du formalisme développé dans [DP10] (section 2) à notre cas : nous allons décrire notre problème d'arrêt optimal en utilisant une formulation dite *faible*. Ainsi, il nous sera plus aisé de démontrer son équivalence à un POMDP bien choisi.

Soient $\mathbb{X} \subset \mathbb{R}^m$ et $\mathbb{Y} \subset \mathbb{R}^n$ des espaces de Borel, et $P(\cdot \mid \cdot)$ un noyau stochastique sur $\mathbb{X} \times \mathbb{Y}$ sachant $\mathbb{X} \times \mathbb{Y}$. Considérant une chaîne de Markov à horizon fini $N_0 \in \mathbb{N}$, à espace d'états $\mathbb{X} \times \mathbb{Y}$, de loi de transition P et d'état initial $(x^0, y^0) \in \mathbb{X} \times \mathbb{Y}$, l'objectif de notre problème sera de

trouver un temps d'arrêt optimal dans $[\![0; N_0]\!]$ afin de maximiser une fonction de récompense. Nous supposons que la composante à valeurs dans $\mathbb{X}$ est *non observable*, tandis que la composante à valeurs dans $\mathbb{Y}$ est *observable*. La règle d'arrêt que l'on décidera ne pourra se baser que sur la partie observable. Soit $R : \mathbb{X} \times \mathbb{Y} \to \mathbb{R}_+$ la fonction de récompense positive que l'on cherche à maximiser. Cette section est consacrée à la définition formelle de ce type de problème.

Dans un souci de simplicité, nous établissons ici les quatre hypothèses que nous ferons sur notre modèle. La première suppose que le noyau P possède une densité par rapport à un produit tensoriel de mesures de probabilité, et assure que l'on va pouvoir utiliser des techniques de quantification décrites dans le chapitre 7.

Hypothèse(s) A.

(1) Il existe des mesures de probabilité σ-finies λ sur $\mathbb{X}$ et ν sur $\mathbb{Y}$ ainsi qu'une fonction mesurable $q : (\mathbb{X} \times \mathbb{Y})^2 \to \mathbb{R}_+$ telles que

$$P(\mathrm{d}x', \mathrm{d}y' \mid x, y) = q(x', y' \mid x, y)\lambda(\mathrm{d}x')\nu(\mathrm{d}y').$$

(2) La mesure λ a un moment d'ordre $2 + \beta$ fini pour un certain $\beta > 0$.

La deuxième hypothèse impose à la densité q quelques propriétés de régularité qui seront essentielles pour la suite de notre raisonnement.

Hypothèse(s) B. *Il existe des constantes positives $\overline{q}$ et L_q telles que, pour tous $x, \tilde{x}, x', x'' \in \mathbb{X}$, $y, \tilde{y}, y' \in \mathbb{Y}$,*

(1) $q(x', y' \mid x, y) \leq \overline{q}$;

(2) $|q(x', y' \mid x, y) - q(x'', y' \mid \tilde{x}, \tilde{y})| \leq L_q[|x' - x''| + |x - \tilde{x}| + |y - \tilde{y}|]$.

Avant d'établir la troisième hypothèse, on définit, pour tous $x \in \mathbb{X}$ et $y, y' \in \mathbb{Y}$, la quantité

$$q(\lambda, y' \mid x, y) = \int_{\mathbb{X}} q(x', y' \mid x, y)\lambda(\mathrm{d}x').$$

Elle apparaîtra dans le formalisme du MDP complètement observable que l'on va construire plus tard. Notons que, par l'hypothèse A(1), $y' \mapsto q(\lambda, y' \mid \cdot, \cdot)$ est une densité sur $\mathbb{Y}$ par rapport à ν. L'hypothèse C qui suit n'est qu'une condition suffisante pour avoir notre résultat principal.

Hypothèse(s) C. *Il existe un réel $\delta > 0$ tel que, pour tous $x \in \mathbb{X}$ et $y, y' \in \mathbb{Y}$, $q(\lambda, y' \mid x, y) \geq \dfrac{1}{\delta}$.*

Quant à l'hypothèse D, elle impose une certaine régularité à la fonction de récompense R.

Hypothèse(s) D. *La fonction R appartient à $\mathbb{L}(\mathbb{X} \times \mathbb{Y})$.*

8.1.2 Arrêt optimal

Définissons formellement notre problème d'arrêt optimal. D'abord, nous avons besoin d'une notion de *contrôle* du processus.

Définition 8.1.2.1. On appelle *contrôle* un sextuplet

$$\lambda = (\Xi, \mathcal{F}, \mathbf{P}, (\mathcal{F}_k)_{0\leq k\leq N_0}, (\mathcal{X}_k, \mathcal{Y}_k)_{0\leq k\leq N_0}, \tau)$$

où

— $(\Xi, \mathcal{F}, \mathbf{P}, (\mathcal{F}_k)_{0\leq k\leq N_0})$ est un espace de probabilité filtré, avec $\mathbf{E}$ l'espérance associée à $\mathbf{P}$;

— $(\mathcal{X}_k, \mathcal{Y}_k)_{0\leq k\leq N_0}$ est une chaîne de Markov définie sur $(\Xi, \mathcal{F}, \mathbf{P})$, de loi de transition P, à valeurs dans $\mathbb{X}\times\mathbb{Y}$, adaptée à la filtration $(\mathcal{F}_k)_{0\leq k\leq N_0}$ et d'état initial (x^0, y^0) (c'est-à-dire que la loi de $(\mathcal{X}_0, \mathcal{Y}_0)$ est $\delta_{(x^0,y^0)}$) ;

— τ est un temps d'arrêt à valeurs dans $[\![0; N_0]\!]$ pour la filtration $(\mathcal{F}_k^{\mathbb{Y}})_{0\leq k\leq N_0}$ définie pour tout $k \in [\![0; N_0]\!]$ par $\mathcal{F}_k^{\mathbb{Y}} = \sigma(\mathcal{Y}_0, \ldots, \mathcal{Y}_k)$.

L'ensemble de tels contrôles est noté Λ.

On définit notre problème d'arrêt optimal ainsi : pour $\lambda \in \Lambda$ et $(x, y) \in \mathbb{X}\times\mathbb{Y}$ un état initial, on considère les fonctions $\mathcal{R}$ et $\mathcal{R}^*$ définies par

$$\mathcal{R}(x, y, \lambda) = \mathbf{E}[R(\mathcal{X}_\tau, \mathcal{Y}_\tau)]$$

et

$$\mathcal{R}^*(x, y) = \sup_{\lambda\in\Lambda} \mathcal{R}(x, y, \lambda).$$

Notre objectif, dans ce chapitre, est d'approcher la valeur $\mathcal{R}^*(x^0, y^0)$. Notons que notre problème est bien défini car l'hypothèse D assure que pour tout $(x, y) \in \mathbb{X}\times\mathbb{Y}$, $\mathcal{R}^*(x, y) \leq \|R\|_{\mathbb{L}} < +\infty$.

8.1.3 Un modèle markovien décisionnel partiellement observable

On introduit le modèle markovien décisionnel partiellement observable

$$\mathcal{M} = \big(\mathbb{S}, \mathbb{A}, Q, r, h, (x^0, y^0, 0)\big)$$

dont les éléments sont définis ci-après.

— $\mathbb{S} = \mathbb{X}\times\mathbb{Y}\times\{0; 1\}$ est *l'espace des états*. La troisième variable d'état, qui est à valeurs dans $\{0; 1\}$, indique si l'on s'est arrêté (auquel cas on la pose égale à 1) ou pas (elle vaut alors 0). L'espace $\mathbb{X}$ est l'espace des états non observables, et $\mathbb{Y}\times\{0; 1\}$ est l'espace des états observables.

— $\mathbb{A} = \{0; 1\}$ est *l'espace des actions*, avec 0 mis pour l'action « continuer » et 1 mis pour l'action « arrêter ».

— Q est la *loi de transition* du modèle : il s'agit d'un noyau stochastique sur $\mathbb{S}$ sachant $\mathbb{S}\times\mathbb{A}$ défini pour tous $B \in \mathcal{B}(\mathbb{X})$, $C \in \mathcal{B}(\mathbb{Y})$ et $D \subset \{0; 1\}$ par

$$Q(B\times C\times D \mid x, y, z, a) = P(B\times C \mid x, y)\left[\delta_z(D)\mathbb{1}_{\{0\}}(a) + \delta_1(D)\mathbb{1}_{\{1\}}(a)\right]$$

avec P le noyau de transition de notre problème d'arrêt optimal de départ.

— $r : \mathbb{S} \times \mathbb{A} \to \mathbb{R}_+$ est la *fonction de récompense par transition* qu'on définira par

$$r(x, y, z, a) = R(x, y)\mathbb{1}_{\{(0,1)\}}(z, a)$$

pour tout $(x, y, z, a) \in \mathbb{S} \times \mathbb{A}$. Cette définition de la fonction r fait en sorte que l'on ne calcule la récompense qu'au moment où on s'arrête.

— $h : \mathbb{S} \to \mathbb{R}_+$ est la *fonction de récompense terminale* définie par

$$h(x, y, z) = R(x, y)\mathbb{1}_{\{0\}}(z)$$

pour tout $(x, y, z) \in \mathbb{S}$. Ainsi, si l'on ne s'est pas arrêté avant l'horizon, la fonction de récompense est calculée à ce moment-là. C'est une manière de forcer l'arrêt au temps N_0.

— $(x^0, y^0, 0) \in \mathbb{S}$ est *l'état initial*. Par souci de simplification, on parlera dans la suite de l'état initial (x^0, y^0) au lieu de $(x^0, y^0, 0)$.

Détaillons maintenant la notion de stratégie utilisée pour ce modèle. Comme dans le paragraphe 1.4 du chapitre 1, on définit les ensembles des *histoires observables* par récurrence par

$$\begin{cases} H_0^{\mathrm{o}} = \mathbb{Y} \times \{0; 1\} \\ H_k^{\mathrm{o}} = H_{k-1}^{\mathrm{o}} \times \mathbb{A} \times (\mathbb{Y} \times \{0; 1\}), \text{ pour tout } 1 \leq k \leq N_0. \end{cases} \tag{8.1}$$

Soit $\Omega = \mathbb{S}^{N_0+1}$. On se donne une stratégie $\pi = (f_0, \ldots, f_{N_0-1}) \in \Pi^{\mathrm{o}}$ et on définit sur Ω les variables aléatoires

$$X_k(\omega) = x_k,\ Y_k(\omega) = y_k \text{ et } Z_k(\omega) = z_k \tag{8.2}$$

pour tout $k \in [\![0; N_0]\!]$ où $\omega = (x_0, y_0, z_0, \ldots, x_{N_0}, y_{N_0}, z_{N_0}) \in \Omega$. Par récurrence, on définit aussi la suite des actions successives choisies selon la stratégie π par

$$\begin{cases} A_0 = f_0(Y_0, Z_0) \\ A_k = f_k(Y_0, Z_0, A_0, \ldots, Y_{k-1}, Z_{k-1}, A_{k-1}, Y_k, Z_k) \text{ pour tout } 1 \leq k \leq N_0 - 1. \end{cases} \tag{8.3}$$

Soit $(x, y) \in \mathbb{X} \times \mathbb{Y}$ un état initial. Le théorème de Ionescu-Tulcea assure l'existence d'une unique mesure de probabilité $\mathbb{P}^{\pi}_{x,y}$ sur $(\Omega, \mathcal{G})$ avec $\mathcal{G}$ la tribu produit associée, telle que, pour tous $B \in \mathcal{B}(\mathbb{X})$, $C \in \mathcal{B}(\mathbb{Y})$, $D \subset \{0; 1\}$ et $h_k \in H_k^{\mathrm{o}}$,

$$\mathbb{P}^{\pi}_{x,y}\{(X_0, Y_0, Z_0) \in B \times C \times D\} = \delta_x(B)\delta_y(C)\delta_0(D), \tag{8.4}$$

$$\mathbb{P}^{\pi}_{x,y}\{A_k \in D \mid (Y_0, Z_0, A_0, \ldots, Y_{k-1}, Z_{k-1}, A_{k-1}, Y_k, Z_k) = h_k\} = \delta_{f_k(h_k)}(D), \tag{8.5}$$

$$\begin{aligned} &\mathbb{P}^{\pi}_{x,y}\{(X_{k+1}, Y_{k+1}, Z_{k+1}) \in B \times C \times D \mid X_0, Y_0, Z_0, A_0, \ldots, X_k, Y_k, Z_k, A_k\} \\ &= Q(B \times C \times D \mid X_k, Y_k, Z_k, A_k). \end{aligned} \tag{8.6}$$

On notera $\mathbb{E}^{\pi}_{x,y}$ l'espérance associée à $\mathbb{P}^{\pi}_{x,y}$.

On définit alors le problème d'optimisation en horizon fini associé à $\mathcal{M}$ ainsi : pour une stratégie $\pi \in \Pi^{\mathrm{o}}$ et $(x, y) \in \mathbb{X} \times \mathbb{Y}$ un état initial, on pose

$$\mathcal{R}_{\mathcal{M}}(x, y, \pi) = \mathbb{E}^{\pi}_{x,y}\left[\sum_{k=0}^{N_0-1} r(X_k, Y_k, Z_k, A_k) + h(X_{N_0}, Y_{N_0}, Z_{N_0})\right]$$

et

$$\mathcal{R}^*_{\mathcal{M}}(x, y) = \sup_{\pi \in \Pi^{\mathrm{o}}} \mathcal{R}_{\mathcal{M}}(x, y, \pi).$$

On cherche à montrer qu'il est équivalent à notre problème d'arrêt optimal. Notons que ce problème est bien défini, car l'hypothèse D assure que $\mathcal{R}^*_{\mathcal{M}}(x, y) \leq N_0 \|R\|_{\mathbb{L}} < +\infty$ pour tout $(x, y) \in \mathbb{X} \times \mathbb{Y}$.

Le théorème suivant, inspiré du théorème 2.1 de [DP10], ainsi que sa démonstration, montre l'équivalence souhaitée.

Théorème 8.1.3.1. *Pour tout contrôle $\lambda \in \Lambda$, pour tout état initial $(x, y) \in \mathbb{X} \times \mathbb{Y}$, il existe une stratégie $\pi \in \Pi^{\mathrm{o}}$ telle que*

$$\mathcal{R}_{\mathcal{M}}(x, y, \pi) = \mathcal{R}(x, y, \lambda).$$

Réciproquement, pour toute stratégie $\pi \in \Pi^{\mathrm{o}}$, pour tout état initial $(x, y) \in \mathbb{X} \times \mathbb{Y}$, il existe un contrôle $\lambda \in \Lambda$ tel que

$$\mathcal{R}(x, y, \lambda) = \mathcal{R}_{\mathcal{M}}(x, y, \pi).$$

Démonstration. Consulter l'annexe B, section B.1. □

Remarque. Ce théorème implique en particulier que $\mathcal{R}^*(x, y) = \mathcal{R}^*_{\mathcal{M}}(x, y)$.

Désormais, dans ce travail, nous nous occuperons de résoudre le problème d'optimisation associé à ce POMDP.

8.1.4 Un modèle markovien décisionnel complètement observable

Comme nous l'avons vu dans la section 2.3 du chapitre 2, l'argument standard pour résoudre ce genre de problèmes est de se ramener à un processus complètement observable. Dans un premier temps, on fixe quelques notations : dans toute la suite, pour toute mesure de probabilité $\rho \in \mathcal{P}(\mathbb{X})$, on notera pour tous $x' \in \mathbb{X}$ et $y, y' \in \mathbb{Y}$

$$q(x', y' \mid \rho, y) = \int_{\mathbb{X}} q(x', y' \mid x, y)\rho(\mathrm{d}x),$$

et on pose pour tous $y, y' \in \mathbb{Y}$

$$q(\lambda, y' \mid \rho, y) = \int_{\mathbb{X}} q(\lambda, y' \mid x, y)\rho(\mathrm{d}x).$$

Vérifions si ces quantités respectent des hypothèses similaires à celles que l'on a posées au début.

Proposition 8.1.4.1. *Sous les hypothèses B et C, pour tous $x', x'' \in \mathbb{X}$, $y, y', \tilde{y} \in \mathbb{Y}$, $\rho, \tilde{\rho} \in \mathcal{P}(\mathbb{X})$,*

(1) $q(x', y' \mid \rho, y) \leq \overline{q}$;

(2) $|q(x', y' \mid \rho, y) - q(x'', y' \mid \rho, y)| \leq L_q |x' - x''|$;

(3) $|q(x', y' \mid \rho, y) - q(x', y' \mid \tilde{\rho}, \tilde{y})| \leq (\overline{q} + L_q)[\|\rho - \tilde{\rho}\|_{KR} + |y - \tilde{y}|]$;

(4) $q(\lambda, y' \mid \rho, y) \geq \dfrac{1}{\delta}$.

Démonstration. Les points (1), (2) et (4) sont évidents. Pour démontrer (3), on considère $x' \in \mathbb{X}$, $y', \tilde{y} \in \mathbb{Y}$ et $\rho, \tilde{\rho} \in \mathcal{P}(\mathbb{X})$. On écrit que

$$\begin{aligned}\left|q(x', y' \mid \rho, y) - q(x', y' \mid \tilde{\rho}, \tilde{y})\right| \leq & \int_{\mathbb{X}} \left|q(x', y' \mid x, y) - q(x', y' \mid x, \tilde{y})\right| \rho(\mathrm{d}x) \\ & + \left| \int_{\mathbb{X}} q(x', y' \mid x, \tilde{y}) \rho(\mathrm{d}x) - \int_{\mathbb{X}} q(x', y' \mid x, \tilde{y}) \tilde{\rho}(\mathrm{d}x) \right|. \end{aligned} \tag{8.7}$$

L'hypothèse B(2) donne que $|q(x', y' \mid x, y) - q(x', y' \mid x, \tilde{y})| \leq L_q |y - \tilde{y}|$. De plus, à x', y' et $\tilde{y}$ fixés, la fonction $x \mapsto q(x', y' \mid x, \tilde{y})$ est bornée par $\overline{q}$ et L_q-lipschitzienne sur $\mathbb{X}$. Il vient donc qu'à x', y' et $\tilde{y}$ fixés, la fonction $x \mapsto \dfrac{q(x', y' \mid x, \tilde{y})}{\overline{q} + L_q}$ est dans $\mathbb{L}^*(\mathbb{X})$. Alors,

$$\left|q(x', y' \mid \rho, y) - q(x', y' \mid \tilde{\rho}, \tilde{y})\right| \leq (\overline{q} + L_q) \|\rho - \tilde{\rho}\|_{KR} + L_q |y - \tilde{y}|. \tag{8.8}$$

Puisque $\overline{q}$ est positif, on a bien le résultat voulu. □

Il faut vérifier est que le noyau Q possède une densité par rapport à un produit de mesures σ-finies (cf. chapitre 2 section 2.3). On écrit donc

$$Q(\mathrm{d}x', \mathrm{d}y', \mathrm{d}z' \mid x, y, z, a) = w(x', y', z' \mid x, y, z, a) \lambda(\mathrm{d}x') \nu(\mathrm{d}y') \mu(\mathrm{d}z')$$

où

$$w(x', y', z' \mid x, y, z, a) = \begin{cases} 2\mathbb{1}_{\{z\}}(z') q(x', y' \mid x, y) \text{ si } a = 0 \\ 2\mathbb{1}_{\{1\}}(z') q(x', y' \mid x, y) \text{ si } a = 1 \end{cases}$$

et $\mu(\mathrm{d}z') = \frac{1}{2}(\delta_0 + \delta_1)(\mathrm{d}z')$.

Soit $C \in \mathcal{B}(\mathbb{X})$. Pour $y, y' \in \mathbb{Y}$, $\rho \in \mathcal{P}(\mathbb{X})$, et $z, z', a \in \{0; 1\}$ tels que $w(x', y', z' \mid x, y, z, a) \neq 0$, on a que :

$$\frac{\displaystyle\int_C \left(\int_{\mathbb{X}} w(x', y', z' \mid x, y, z, a) \rho(\mathrm{d}x) \right) \lambda(\mathrm{d}x')}{\displaystyle\int_{\mathbb{X}} \left(\int_{\mathbb{X}} w(x', y', z' \mid x, y, z, a) \rho(\mathrm{d}x) \right) \lambda(\mathrm{d}x')} = \frac{1}{q(\lambda, y' \mid \rho, y)} \int_C q(x', y' \mid \rho, y) \lambda(\mathrm{d}x').$$

Le membre de gauche ne dépend en fait ni de z, ni de z', ni de a. Introduisons donc la fonction $\Phi : \mathbb{Y} \times \mathcal{P}(\mathbb{X}) \times \mathbb{Y} \to \mathcal{P}(\mathbb{X})$ définie, pour tous $\rho \in \mathcal{P}(\mathbb{X})$, $y, y' \in \mathbb{Y}$ et $C \in \mathcal{B}(\mathbb{X})$, par

$$\Phi(y', \rho, y)(C) = \frac{1}{q(\lambda, y' \mid \rho, y)} \int_C q(x', y' \mid \rho, y) \lambda(\mathrm{d}x').$$

Si $w(x', y', z' \mid x, y, z, a) = 0$, on pose

$$\frac{\displaystyle\int_C \left(\int_{\mathbb{X}} w(x', y', z' \mid x, y, z, a) \rho(\mathrm{d}x) \right) \lambda(\mathrm{d}x')}{\displaystyle\int_{\mathbb{X}} \left(\int_{\mathbb{X}} w(x', y', z' \mid x, y, z, a) \rho(\mathrm{d}x) \right) \lambda(\mathrm{d}x')} = \Phi(y', \rho, y)(C).$$

L'application Φ ainsi définie est appelée *opérateur de Bayes*.

Si on applique strictement le formalisme de [BR11], on va construire un modèle markovien décisionnel complètement observable dont le noyau de transition, noté Q', sera défini, pour tous $B \in \mathcal{B}(\mathcal{P}(\mathbb{X}))$, $C \in \mathcal{B}(\mathbb{Y})$, $D \subset \{0; 1\}$, $\rho \in \mathcal{P}(\mathbb{X})$, $y \in \mathbb{Y}$ et $z, a \in \{0; 1\}$, par

$$Q'(B \times C \times D \mid \rho, y, z, a) = \int_{\mathbb{X}} \int_{C \times D} \mathbb{1}_B \left(\Phi(y', \rho, y) \right) Q^{\mathbb{Y},Z}(\mathrm{d}y', \mathrm{d}z' \mid x, y, z, a) \rho(\mathrm{d}x) \tag{8.9}$$

où

$$Q^{\mathbb{Y},Z}(\mathrm{d}y', \mathrm{d}z' \mid x, y, z, a) = \int_{\mathbb{X}} Q(\mathrm{d}x', \mathrm{d}y', \mathrm{d}z' \mid x, y, z, a).$$

Or, on a que

$$\begin{aligned} \int_{\mathbb{X}} Q^{\mathbb{Y},Z}(\mathrm{d}y', \mathrm{d}z' \mid x, y, z, 0) \rho(\mathrm{d}x) &= 2 \mathbb{1}_{\{z\}}(z') \nu(\mathrm{d}y') \mu(\mathrm{d}z') \int_{\mathbb{X}} \int_{\mathbb{X}} q(x', y' \mid x, y) \lambda(\mathrm{d}x') \rho(\mathrm{d}x) \\ &= \mathbb{1}_{\{z\}}(z') q(\lambda, y' \mid \rho, y) \nu(\mathrm{d}y') (\delta_0(\mathrm{d}z') + \delta_1(\mathrm{d}z')). \end{aligned} \tag{8.10}$$

En remarquant que $\displaystyle\int_D \mathbb{1}_{\{z\}}(z')(\delta_0(\mathrm{d}z') + \delta_1(\mathrm{d}z')) = \int_D \delta_z(\mathrm{d}z') = \delta_z(D)$, il vient

$$Q'(B \times C \times D \mid \rho, y, z, 0) = \delta_z(D) \left(\int_C \mathbb{1}_B\big(\Phi(y', \rho, y)\big) q(\lambda, y' \mid \rho, y) \nu(\mathrm{d}y') \right). \tag{8.11}$$

Un calcul similaire amène à

$$Q'(B \times C \times D \mid \rho, y, z, 1) = \delta_1(D) \left(\int_C \mathbb{1}_B\big(\Phi(y', \rho, y)\big) q(\lambda, y' \mid \rho, y) \nu(\mathrm{d}y') \right). \tag{8.12}$$

On définit alors un noyau stochastique S sur $\mathcal{P}(\mathbb{X}) \times \mathbb{Y}$ sachant $\mathcal{P}(\mathbb{X}) \times \mathbb{Y}$ par

$$S(B \times C \mid \rho, y) = \int_C \mathbb{1}_B\big(\Phi(y', \rho, y)\big) Q^{\mathbb{Y}}(\mathrm{d}y' \mid \rho, y)$$

pour tous $B \in \mathcal{B}(\mathcal{P}(\mathbb{X}))$, $C \in \mathcal{B}(\mathbb{Y})$, $\rho \in \mathcal{P}(\mathbb{X})$ et $y \in \mathbb{Y}$, où $Q^{\mathbb{Y}}(\mathrm{d}y' \mid \rho, y) = q(\lambda, y' \mid \rho, y) \nu(\mathrm{d}y')$.

On construit donc le modèle décisionnel markovien complètement observable $\mathcal{M}'$ suivant :

$$\mathcal{M}' = \big(\mathbb{S}', \mathbb{A}, Q', r', h', (\delta_{x^0}, y^0, 0)\big) \tag{8.13}$$

avec

— $\mathbb{S}' = \mathcal{P}(\mathbb{X}) \times \mathbb{Y} \times \{0; 1\}$ est le nouvel espace d'états dont toutes ses composantes sont observables ;

— Q' est la loi de transition définie, pour $B \in \mathcal{B}\big(\mathcal{P}(\mathbb{X})\big)$, $C \in \mathcal{B}(\mathbb{Y})$ et $D \subset \{0;1\}$, par

$$Q'(B \times C \times D \mid \rho, y, z, a) = S(B \times C \mid \rho, y) \left[\delta_z(D)\mathbb{1}_{\{0\}}(a) + \delta_1(D)\mathbb{1}_{\{1\}}(a)\right];$$

— $r' : \mathbb{S}' \times \mathbb{A} \to \mathbb{R}_+$ est la fonction de récompense par transition, définie par

$$r'(\rho, y, z, a) = R'(\rho, y)\mathbb{1}_{\{(0,1)\}}(z, a) \text{ où } R'(\rho, y) = \int_{\mathbb{X}} R(x, y)\rho(\mathrm{d}x);$$

— $h' : \mathbb{S}' \to \mathbb{R}_+$ est la fonction de récompense terminale définie par

$$h'(\rho, y, z) = \mathbb{1}_{\{0\}}(z)R'(\rho, y).$$

Remarque. Pour la topologie induite par la norme de Kantorovich-Rubinstein, $\mathcal{P}(\mathbb{X})$ est un espace de Borel. En effet, on note d'abord que, $\mathbb{X}$ étant un espace métrique, le corollaire 6.3.5 de [Bog07] donne que les tribus de Borel et de Baire sur $\mathbb{X}$ sont égales. Donc toute mesure de Borel est de Baire et inversement. Comme $\mathbb{X}$ est un espace de Borel, on peut appliquer le théorème 8.3.2 de [Bog07]. Il montre que la topologie engendrée par la métrique associée à $\|\cdot\|_{KR}$ coïncide avec la topologie faible induite sur $\mathcal{P}(\mathbb{X})$. On conclut avec le théorème 8.9.5 de [Bog07] que $\mathcal{P}(\mathbb{X})$ est aussi un espace de Borel.

Soient $\Omega' = \mathbb{S}'^{N_0+1}$ et $\mathcal{G}'$ sa tribu produit associée. On construit canoniquement sur $(\Omega', \mathcal{G}')$ un processus (P_k, Y_k, Z_k). Pour une stratégie $\pi \in \Pi^{\mathrm{o}}$, on considère une suite de variables aléatoires (A_k) définie par les équations (8.3). On considère également l'unique probabilité $\mathbb{P}^{\pi}_{\delta_x, y}$ sur $(\Omega', \mathcal{G}')$ donnée par le théorème de Ionescu-Tulcea. On note $\mathbb{E}^{\pi}_{\delta_x, y}$ l'espérance prise avec la probabilité $\mathbb{P}^{\pi}_{\delta_x, y}$.

Notre problème d'optimisation à horizon fini pour $\mathcal{M}'$ s'exprime donc ainsi : on pose, pour un état initial $(x, y) \in \mathbb{X} \times \mathbb{Y}$ et $\pi \in \Pi^{\mathrm{o}}$ une stratégie,

$$\mathcal{R}_{\mathcal{M}'}(\delta_x, y, \pi) = \mathbb{E}^{\pi}_{\delta_x, y}\left[\sum_{k=0}^{N_0-1} r'(P_k, Y_k, Z_k, A_k) + h'(P_{N_0}, Y_{N_0}, Z_{N_0})\right] \tag{8.14}$$

puis

$$\mathcal{R}^*_{\mathcal{M}'}(\delta_x, y) = \sup_{\pi \in \Pi^{\mathrm{o}}} \mathcal{R}_{\mathcal{M}'}(\delta_x, y, \pi). \tag{8.15}$$

Avoir pu donner une densité w pour le noyau Q et voir que le problème d'optimisation associé à $\mathcal{M}$ soit bien défini par l'hypothèse D sont les deux éléments qui nous permettent d'appliquer le théorème 2.3.0.2, et avoir que $\mathcal{R}_{\mathcal{M}}(x, y, \pi) = \mathcal{R}_{\mathcal{M}'}(\delta_x, y, \pi)$ pour tous $(x, y) \in \mathbb{X} \times \mathbb{Y}$ et $\pi \in \Pi^{\mathrm{o}}$ (et donc a fortiori $\mathcal{R}^*_{\mathcal{M}}(x, y) = \mathcal{R}^*_{\mathcal{M}'}(\delta_x, y)$).

Désormais, on ne s'occupera de résoudre que le problème posé pour $\mathcal{M}'$. Par abus de notation, dans toute la suite, on notera respectivement R, r et h au lieu de R', r' et h'.

8.1.5 Les équations de Bellman

La résolution du problème d'optimisation posé pour $\mathcal{M}'$ passe par le calcul des équations de Bellman définies dans le théorème 5.3.3 de [BR11]. On s'intéresse donc à la relation de récurrence suivante, pour tout $k \in [\![1; N_0]\!]$ et tout élément (ρ, y, z) de $\mathbb{S}'$:

$$\begin{cases} v_0(\rho, y, z) = h(\rho, y, z) \\ v_k(\rho, y, z) = \max\limits_{a\in\{0;1\}} \left\{ r(\rho, y, z, a) + \displaystyle\int_{\mathbb{S}'} v_{k-1}(\rho', y', z') Q'(\mathrm{d}\rho', \mathrm{d}y', \mathrm{d}z' \mid \rho, y, z, a) \right\}. \end{cases}$$

On commence par remarquer que pour toute fonction mesurable $g : \mathbb{S}' \to \mathbb{R}$ telle que $g(\rho, y, 1) = 0$, on obtient

$$\max_{a\in\{0;1\}} \left\{ r(\rho, y, 1, a) + \int_{\mathbb{S}'} g(\rho', y', z') Q'(\mathrm{d}\rho', \mathrm{d}y', \mathrm{d}z' \mid \rho, y, 1, a) \right\} = 0$$

par définition de r qui impose que $r(\rho, y, 1, 0) = r(\rho, y, 1, 1) = 0$. Donc, puisque par définition de la fonction h, on a $h(\rho, y, 1) = 0$, il vient que $v_k(\rho, y, 1) = 0$ pour tout $k \in [\![0; N_0]\!]$. On peut alors restreindre cette relation de récurrence à l'ensemble des fonctions mesurables $g : \mathbb{S}' \to \mathbb{R}$ telles que $g(\rho, y, 1) = 0$ pour tout $(\rho, y) \in \mathcal{P}(\mathbb{X}) \times \mathbb{Y}$.

Enfin, comme on a

$$\begin{aligned} &\max_{a\in\{0;1\}} \left\{ r(\rho, y, 0, a) + \int_{\mathbb{S}'} g(\rho', y', z') Q'(\mathrm{d}\rho', \mathrm{d}y', \mathrm{d}z' \mid \rho, y, 0, a) \right\} \\ &= \max \left\{ R(\rho, y); \int_{\mathbb{Y}} g(\Phi(y', \rho, y), y', 0) q(\lambda, y' \mid \rho, y) \nu(\mathrm{d}y') \right\}, \end{aligned} \tag{8.16}$$

par abus de notation, on réécrit la récurrence précédente en termes de fonctions mesurables $f : \mathcal{P}(\mathbb{X}) \times \mathbb{Y} \to \mathbb{R}$ en posant, pour tout $(\rho, y) \in \mathcal{P}(\mathbb{X}) \times \mathbb{Y}$,

$$\mathcal{B}f(\rho, y) = \max \{R(\rho, y); Sf(\rho, y)\}$$

où

$$Sf(\rho, y) = \int_{\mathbb{Y}} f\left(\Phi(y', \rho, y), y'\right) Q^{\mathbb{Y}}(\mathrm{d}y' \mid \rho, y).$$

L'opérateur $\mathcal{B}$, défini sur l'ensemble des fonctions mesurables de $\mathcal{P}(\mathbb{X}) \times \mathbb{Y}$ dans $\mathbb{R}$, est appelé *opérateur de Bellman*. On définit alors l'équation de Bellman par :

$$\begin{cases} v_0(\rho, y) = h(\rho, y, 0) = R(\rho, y) \\ v_k(\rho, y) = \mathcal{B}v_{k-1}(\rho, y) = \mathcal{B}^k v_0(\rho, y) \end{cases} \quad \text{pour tous } k \in [\![1; N_0]\!] \text{ et } (\rho, y) \in \mathcal{P}(\mathbb{X}) \times \mathbb{Y},$$

où $\mathcal{B}^k = \underbrace{\mathcal{B} \circ \ldots \circ \mathcal{B}}_{k \text{ fois}}$, permet de calculer la valeur optimale $\mathcal{R}_{\mathcal{M}'}$ de notre problème. En effet, on a le résultat suivant.

Théorème 8.1.5.1. *Pour tous $x \in \mathbb{X}$ et $y \in \mathbb{Y}$, $\mathcal{R}^*_{\mathcal{M}'}(\delta_x, y) = v_{N_0}(\delta_x, y)$.*

Démonstration. Consulter l'annexe B, section B.2. □

Étant donnée la forme des équations de Bellman, trouver un espace de fonctions stabilisé par $\mathcal{B}$ (et dans lequel se trouve la fonction v_0) permettra de déterminer dans quel espace se trouve la fonction de valeur optimale. Dès lors, en construisant une approximation appropriée de l'opérateur de Bellman, nous pourrons contrôler l'erreur commise. Notre objectif sera donc de vérifier que la discrétisation que nous proposerons répond bien à cette exigence.

8.2 Première discrétisation de l'espace d'états

Dans cette partie, nous allons chercher à approcher numériquement la fonction $\mathcal{R}_{\mathcal{M}'}$. À cet effet, dans la suite, on considérera X une variable aléatoire de loi λ à valeurs dans $\mathbb{X}$ et Y une variable aléatoire de loi ν à valeurs dans $\mathbb{Y}$, toutes deux définies sur un espace de probabilité $(\mathcal{E}, \mathcal{T}, \mathbb{P})$. On notera $\mathbb{E}$ l'espérance prise pour $\mathbb{P}$. Ici, nous allons discrétiser l'espace des états non observables $\mathbb{X}$. En effet, l'hypothèse A(2) nous permet de procéder à la L^2-quantification optimale de la variable aléatoire X comme le font les auteurs de [PPP04] : il en résulte une variable aléatoire X_N de loi λ_N à valeurs dans $\mathbb{X}$ définie sur $(\mathcal{E}, \mathcal{T}, \mathbb{P})$, où $N \in \mathbb{N}^*$ représente le nombre de valeurs que peut prendre X_N, c'est-à-dire que

$$X_N(\mathcal{E}) = \left\{x_1^{(N)}, \dots, x_N^{(N)}\right\} \subset \mathbb{X}.$$

L'ensemble $\left\{x_1^{(N)}, \dots, x_N^{(N)}\right\}$ est la grille de quantification optimale pour X. Soit

$$\mathbb{X}_N = \left\{x^0, x_1^{(N)}, \dots, x_N^{(N)}\right\} \tag{8.17}$$

la grille optimale de quantification de X augmentée du point x^0. Pour simplifier, on suppose que $x^0 \notin \left\{x_1^{(N)}, \dots, x_N^{(N)}\right\}$. Notons que $\mathbb{P}\left\{X_N = x^0\right\} = 0$. L'ensemble $\mathbb{X}_N$ sera notre espace d'états non observables pour le modèle discrétisé.

On notera dans la suite $\varepsilon_N = \|X - X_N\|_2$ pour tout $N \in \mathbb{N}^*$. Par le théorème 7.2.0.6, ε_N tend vers 0 lorsque N tend vers $+\infty$ avec une vitesse en $\mathrm{O}(N^{-\frac{1}{m}})$. Dès lors, il existe un entier $M_1 \in \mathbb{N}^*$ tel que

$$\varepsilon_N \leq \frac{1}{2L_q}\left(1 \wedge \delta^{-1}\right), \forall N \geq M_1. \tag{8.18}$$

Occupons-nous maintenant du modèle approché.

8.2.1 Construction du modèle

On va construire des approximations des quantités utilisées dans la partie précédente, de sorte à garder une structure similaire pour mesurer plus facilement l'erreur commise. Partons de la densité $q(\lambda, \cdot \mid \cdot, \cdot)$. Pour en construire une approximation, on pose pour tout $N \in \mathbb{N}^*$

$$q(\lambda_N, y' \mid \rho, y) = \int_{\mathbb{X}} q(x', y' \mid \rho, y)\lambda_N(\mathrm{d}x')$$

et

$$q(\lambda_N, \nu \mid \rho, y) = \int_{\mathbb{Y}} q(\lambda_N, y' \mid \rho, y)\nu(\mathrm{d}y') = \int_{\mathbb{X}\times\mathbb{Y}} q(x', y' \mid \rho, y)\lambda_N(\mathrm{d}x')\nu(\mathrm{d}y')$$

pour tous $\rho \in \mathcal{P}(\mathbb{X})$ et $y \in \mathbb{Y}$. Montrons un premier résultat.

Lemme 8.2.1.1. *Sous l'hypothèse B(2), pour tous $N \in \mathbb{N}^*$, $\rho \in \mathcal{P}(\mathbb{X})$ et $y, y' \in \mathbb{Y}$, on a*

$$\big|q(\lambda, y' \mid \rho, y) - q(\lambda_N, y' \mid \rho, y)\big| \leq L_q \varepsilon_N. \tag{8.19}$$

Démonstration. On a que

$$\big|q(\lambda, y' \mid \rho, y) - q(\lambda_N, y' \mid \rho, y)\big| \leq \mathbb{E}[\big|q(X, y' \mid \rho, y) - q(X_N, y' \mid \rho, y)\big|]. \tag{8.20}$$

On peut appliquer le (2) de la proposition 8.1.4.1 pour avoir le résultat. □

En particulier, ce lemme implique le corollaire suivant.

Corollaire 8.2.1.2. *Sous les hypothèses A et B(2), on a que, pour tous $\rho \in \mathcal{P}(\mathbb{X})$, $y \in \mathbb{Y}$, $N \in \mathbb{N}^*$,*

$$|q(\lambda_N, \nu \mid \rho, y) - 1| \leq L_q \varepsilon_N. \tag{8.21}$$

Démonstration. On note que $|q(\lambda_N, \nu \mid \rho, y) - 1| \leq \int_{\mathbb{Y}} \big|q(\lambda_N, y' \mid \rho, y) - q(\lambda, y' \mid \rho, y)\big|\,\nu(\mathrm{d}y')$ et on utilise le lemme 8.2.1.1 pour avoir le résultat. □

Ce corollaire nous assure alors la convergence uniforme de $(q(\lambda_N, \nu \mid \cdot, \cdot))_N$ vers 1. Donc il existe un entier $M_2 \in \mathbb{N}^*$ tel que pour tous $N \geq M_2$, $\rho \in \mathcal{P}(\mathbb{X})$ et $y \in \mathbb{Y}$, $q(\lambda_N, \nu \mid \rho, y) > 0$. On peut alors poser, pour tout $N \geq M_2$,

$$\tilde{q}(\lambda_N, y' \mid \rho, y) = \frac{q(\lambda_N, y' \mid \rho, y)}{q(\lambda_N, \nu \mid \rho, y)}$$

pour tous $\rho \in \mathcal{P}(\mathbb{X})$, $y, y' \in \mathbb{Y}$.

On remarque qu'ainsi, la fonction $y' \mapsto \tilde{q}(\lambda_N, y' \mid \cdot, \cdot)$ est une densité sur $\mathbb{Y}$ par rapport à ν. On définit maintenant, pour tout $N \geq M_2$, un noyau stochastique $Q_N^{\mathbb{Y}}$ sur $\mathbb{Y}$ sachant $\mathcal{P}(\mathbb{X}) \times \mathbb{Y}$ par

$$Q_N^{\mathbb{Y}}(C \mid \rho, y) = \int_C \tilde{q}(\lambda_N, y' \mid \rho, y)\nu(\mathrm{d}y')$$

pour tous $C \in \mathcal{B}(\mathbb{Y})$, $\rho \in \mathcal{P}(\mathbb{X})$ et $y \in \mathbb{Y}$. C'est l'approximation de $Q^{\mathbb{Y}}$ que l'on cherche. Il reste à construire une approximation de l'opérateur de Bayes.

Grâce au lemme 8.2.1.1, on sait que la suite de fonctions $(q(\lambda_N, \cdot \mid \cdot, \cdot))_N$ converge uniformément vers $q(\lambda, \cdot \mid \cdot, \cdot)$, qui est strictement positive (cf. (4) de la proposition 8.1.4.1). Donc il existe $M_3 \in \mathbb{N}^*$ tel que pour tous $N \geq M_3$, $\rho \in \mathcal{P}(\mathbb{X})$ et $y, y' \in \mathbb{Y}$, $q(\lambda_N, y' \mid \rho, y) > 0$. On pose donc

$$M_0 = \max\{M_1; M_2; M_3\}$$

et pour garder la même structure que Φ, on définit, pour tout $N \geq M_0$,

$$\Phi_N(y', \rho, y)(\mathrm{d}x') = \frac{q(x', y' \mid \rho, y)\lambda_N(\mathrm{d}x')}{q(\lambda_N, y' \mid \rho, y)}.$$

Ainsi définie, Φ_N est une mesure de probabilité sur $\mathbb{X}_N$. Enfin, on pose

$$S_N(B \times C \mid \rho, y) = \int_C \mathbb{1}_B\big(\Phi_N(y', \rho, y)\big) Q_N^{\mathbb{Y}}(\mathrm{d}y' \mid \rho, y)$$

pour tous $N \geq M_0$, $B \in \mathcal{B}(\mathcal{P}(\mathbb{X}))$, $C \in \mathcal{B}(\mathbb{Y})$ et $(\rho, y) \in \mathcal{P}(\mathbb{X}) \times \mathbb{Y}$. Remarquons que S_N ainsi défini est bien un noyau stochastique sur $\mathcal{P}(\mathbb{X}) \times \mathbb{Y}$ sachant $\mathcal{P}(\mathbb{X}) \times \mathbb{Y}$, concentré sur $\mathcal{P}(\mathbb{X}_N) \times \mathbb{Y}$.

8.2.2 Un opérateur de Bellman approché

On va à présent définir une approximation de l'opérateur de Bellman, que l'on va noter $\mathcal{B}_N$. Pour garder la même structure que $\mathcal{B}$, on va poser, pour $f : \mathcal{P}(\mathbb{X}) \times \mathbb{Y} \to \mathbb{R}$ et pour $N \geq M_0$,

$$\mathcal{B}_N f(\rho, y) = \max\{R(\rho, y); S_N f(\rho, y)\}$$

où

$$S_N f(\rho, y) = \int_{\mathbb{Y}} f\big(\Phi_N(y', \rho, y), y'\big) Q_N^{\mathbb{Y}}(\mathrm{d}y' \mid \rho, y). \tag{8.22}$$

L'approximation de l'équation de Bellman que l'on choisit est donc

$$\begin{cases} v_0^{(N)}(\rho, y) = R(\rho, y) \\ v_k^{(N)}(\rho, y) = \mathcal{B}_N v_{k-1}^{(N)}(\rho, y) = \mathcal{B}_N^k v_0^{(N)}(\rho, y) \end{cases} \quad \text{pour } k \in [\![1; N_0]\!] \text{ et } (\rho, y) \in \mathcal{P}(\mathbb{X}) \times \mathbb{Y}. \tag{8.23}$$

où $\mathcal{B}_N^k = \underbrace{\mathcal{B}_N \circ \ldots \circ \mathcal{B}_N}_{k \text{ fois}}$. On définit alors la fonction $\mathcal{R}_{\mathcal{M}',N}$ comme

$$\mathcal{R}_{\mathcal{M}',N} \equiv v_{N_0}^{(N)} \tag{8.24}$$

Il reste à savoir si cette fonction approche la fonction $\mathcal{R}_{\mathcal{M}'}$. Démontrons dans un premier temps quelques lemmes utiles.

8.2.3 Résultats préliminaires de régularité

Pour simplifier les notations, dans toute la suite, on posera pour tout $N \in \mathbb{N}^*$

$$E_N = 1 + 2\delta L_q \varepsilon_N.$$

Régularité de la fonction q

Lemme 8.2.3.1. *Sous les hypothèses B, pour tous $\rho, \tilde{\rho} \in \mathcal{P}(\mathbb{X})$, et $y, y', \tilde{y} \in \mathbb{Y}$, on a que*

$$\big|q(\lambda, y' \mid \rho, y) - q(\lambda, y' \mid \tilde{\rho}, \tilde{y})\big| \leq (\overline{q} + L_q)[\|\rho - \tilde{\rho}\|_{KR} + |y - \tilde{y}|]. \tag{8.25}$$

Démonstration. On a que $q(\lambda, y' \mid \rho, y) = \mathbb{E}[q(X, y' \mid \rho, y)]$. Il vient alors

$$\left|q(\lambda, y' \mid \rho, y) - q(\lambda, y' \mid \tilde{\rho}, \tilde{y})\right| \leq \mathbb{E}[\left|q(X, y' \mid \rho, y) - q(X, y' \mid \tilde{\rho}, \tilde{y})\right|]. \tag{8.26}$$

On applique le (3) de la proposition 8.1.4.1 pour avoir le résultat. □

Lemme 8.2.3.2. *Sous les hypothèses A et B, pour tous $\rho, \tilde{\rho} \in \mathcal{P}(\mathbb{X})$, $y, y', \tilde{y} \in \mathbb{Y}$ et $N \geq M_0$, on a que*

$$\left|q(\lambda_N, y' \mid \rho, y) - q(\lambda_N, y' \mid \tilde{\rho}, \tilde{y})\right| \leq (\overline{q} + L_q)[\|\rho - \tilde{\rho}\|_{KR} + |y - \tilde{y}|]. \tag{8.27}$$

Démonstration. La démonstration est similaire à la précédente. □

Corollaire 8.2.3.3. *Sous les hypothèses A et B, on a que, pour tous $N \geq M_0$, $y, y' \in \mathbb{Y}$ et $\rho, \rho' \in \mathcal{P}(\mathbb{X})$,*

$$\left|q(\lambda_N, \nu \mid \rho, y) - q(\lambda_N, \nu \mid \rho', y')\right| \leq (\overline{q} + L_q)[\left\|\rho - \rho'\right\|_{KR} + \left|y - y'\right|]. \tag{8.28}$$

Démonstration. On note que

$$\left|q(\lambda_N, \nu \mid \rho, y) - q(\lambda_N, \nu \mid \rho', y')\right| \leq \int_{\mathbb{Y}} \left|q(\lambda_N, y'' \mid \rho, y) - q(\lambda_N, y'' \mid \rho', y')\right| \nu(\mathrm{d}y'') \tag{8.29}$$

et on applique le lemme précédent pour avoir le résultat. □

Lemme 8.2.3.4. *Sous les hypothèses A, B(2) et C, pour tous $N \geq M_0$, $\rho \in \mathcal{P}(\mathbb{X})$, $y, y' \in \mathbb{Y}$, on a que*

$$\left|\frac{1}{q(\lambda, y' \mid \rho, y)} - \frac{1}{q(\lambda_N, y' \mid \rho, y)}\right| \leq 2\delta^2 L_q \varepsilon_N. \tag{8.30}$$

Démonstration. Le lemme 8.2.1.1 donne que

$$q(\lambda, y' \mid \rho, y) - L_q\varepsilon_N \leq q(\lambda_N, y' \mid \rho, y) \leq q(\lambda, y' \mid \rho, y) + L_q\varepsilon_N. \tag{8.31}$$

Notons que, puisque $N \geq M_0$, $\varepsilon_N \leq \dfrac{1}{2\delta L_q}$ et donc $q(\lambda, y' \mid \rho, y) - L_q\varepsilon_N \geq \dfrac{1}{2\delta} > 0$. On peut alors écrire que

$$\begin{aligned}\frac{1}{q(\lambda, y' \mid \rho, y) + L_q\varepsilon_N} - \frac{1}{q(\lambda, y' \mid \rho, y)} &\leq \frac{1}{q(\lambda_N, y' \mid \rho, y)} - \frac{1}{q(\lambda, y' \mid \rho, y)} \\ &\leq \frac{1}{q(\lambda, y' \mid \rho, y) - L_q\varepsilon_N} - \frac{1}{q(\lambda, y' \mid \rho, y)}.\end{aligned} \tag{8.32}$$

On en déduit alors que

$$\begin{aligned}\left|\frac{1}{q(\lambda_N, y' \mid \rho, y)} - \frac{1}{q(\lambda, y' \mid \rho, y)}\right| &\leq \frac{1}{q(\lambda, y' \mid \rho, y) - L_q\varepsilon_N} - \frac{1}{q(\lambda, y' \mid \rho, y)} \\ &= \frac{1}{q(\lambda, y' \mid \rho, y)}\left(\frac{q(\lambda, y' \mid \rho, y)}{q(\lambda, y' \mid \rho, y) - L_q\varepsilon_N} - 1\right) \\ &\leq \frac{\delta L_q \varepsilon_N}{q(\lambda, y' \mid \rho, y) - L_q\varepsilon_N} \leq 2\delta^2 L_q \varepsilon_N,\end{aligned} \tag{8.33}$$

la pénultième inégalité ayant été obtenue grâce au (4) de la proposition 8.1.4.1. □

Remarque. C'est dans la démonstration de ce lemme que l'hypothèse C a trouvé toute sa justification. En effet, la stricte positivité de $q(\lambda, y' \mid \rho, y) - L_q\varepsilon_N$ implique que $\varepsilon_N < \dfrac{q(\lambda, y' \mid \rho, y)}{L_q}$ pour tous $y, y' \in \mathbb{Y}$ et $\rho \in \mathcal{P}(\mathbb{X})$, ce qui signifie que la quantité $q(\lambda, y' \mid x, y)$ doit être minorée.

Corollaire 8.2.3.5. *Sous les hypothèses A, B(2) et C, pour tous $N \geq M_0$, $\rho \in \mathcal{P}(\mathbb{X})$, $y, y' \in \mathbb{Y}$, on a que*

$$\frac{1}{q(\lambda_N, y' \mid \rho, y)} \leq \delta E_N. \tag{8.34}$$

Corollaire 8.2.3.6. *Sous les hypothèses A, B(2) et C, on a que, pour tous $N \geq M_0$, $\rho \in \mathcal{P}(\mathbb{X})$, $y \in \mathbb{Y}$,*

$$\frac{1}{q(\lambda_N, \nu \mid \rho, y)} \leq \delta E_N. \tag{8.35}$$

Démonstration. Il suffit de remarquer que $q(\lambda_N, \nu \mid \rho, y) = \displaystyle\int_{\mathbb{Y}} q(\lambda_N, y' \mid \rho, y)\nu(\mathrm{d}y')$ et d'utiliser le corollaire précédent pour avoir le résultat. □

Lemme 8.2.3.7. *Sous les hypothèses A et B(2), on a que, pour tous $N \geq M_0$, $\rho \in \mathcal{P}(\mathbb{X})$, $y \in \mathbb{Y}$,*

$$\left| \frac{1}{q(\lambda_N, \nu \mid \rho, y)} - 1 \right| \leq 2L_q\varepsilon_N. \tag{8.36}$$

Démonstration. Le raisonnement est similaire à celui utilisé pour le lemme 8.2.3.4 en partant du corollaire 8.2.1.2. □

Régularité de l'opérateur de Bayes

Lemme 8.2.3.8. *Sous les hypothèses B et C, pour tous $\rho, \tilde{\rho} \in \mathcal{P}(\mathbb{X})$, $y, y', \tilde{y} \in \mathbb{Y}$, on a que*

$$\left\|\Phi(y', \rho, y) - \Phi(y', \tilde{\rho}, \tilde{y})\right\|_{KR} \leq \delta(\overline{q} + L_q)(1 + \overline{q}\delta)[\|\rho - \tilde{\rho}\|_{KR} + |y - \tilde{y}|]. \tag{8.37}$$

Démonstration. Soit $f \in \mathbb{L}^*(\mathbb{X})$. Alors

$$\begin{aligned}
&\left| \int_{\mathbb{X}} f(x)\Phi(y', \rho, y)(\mathrm{d}x) - \int_{\mathbb{X}} f(x)\Phi(y', \tilde{\rho}, \tilde{y})(\mathrm{d}x) \right| \\
&\leq \mathbb{E}\left[|f(X)| \left| \frac{q(X, y' \mid \rho, y)}{q(\lambda, y' \mid \rho, y)} - \frac{q(X, y' \mid \tilde{\rho}, \tilde{y})}{q(\lambda, y' \mid \tilde{\rho}, \tilde{y})} \right| \right] \\
&\leq \mathbb{E}\left[\frac{1}{q(\lambda, y' \mid \rho, y)} \left| q(X, y' \mid \rho, y) - q(X, y' \mid \tilde{\rho}, \tilde{y}) \right| \right] \\
&\quad + \mathbb{E}\left[q(X, y' \mid \tilde{\rho}, \tilde{y}) \left| \frac{1}{q(\lambda, y' \mid \rho, y)} - \frac{1}{q(\lambda, y' \mid \tilde{\rho}, \tilde{y})} \right| \right].
\end{aligned} \tag{8.38}$$

On utilise la proposition 8.1.4.1 pour majorer cette somme, ce qui donne

$$\begin{aligned}
\left| \int_{\mathbb{X}} f(x)\Phi(y', \rho, y)(\mathrm{d}x) - \int_{\mathbb{X}} f(x)\Phi(y', \tilde{\rho}, \tilde{y})(\mathrm{d}x) \right| &\leq \delta(\overline{q} + L_q)[\|\rho - \tilde{\rho}\|_{KR} + |y - \tilde{y}|] \\
&\quad + \overline{q}\delta^2 \left| q(\lambda, y' \mid \rho, y) - q(\lambda, y' \mid \tilde{\rho}, \tilde{y}) \right|.
\end{aligned} \tag{8.39}$$

On conclut en utilisant le lemme 8.2.3.1. □

Lemme 8.2.3.9. *Sous les hypothèses A, B et C, pour tous $N \geq M_0$, $\rho, \tilde{\rho} \in \mathcal{P}(\mathbb{X})$ et $y, y', \tilde{y} \in \mathbb{Y}$, on a que*

$$\left\|\Phi_N(y', \rho, y) - \Phi_N(y', \tilde{\rho}, \tilde{y})\right\|_{KR} \leq \delta E_N(\overline{q} + L_q)(1 + \overline{q}\delta E_N)[\|\rho - \tilde{\rho}\|_{KR} + |y - \tilde{y}|]. \tag{8.40}$$

Démonstration. La démonstration de ce lemme est similaire à celle du lemme 8.2.3.8. On pourra la trouver à l'annexe B, section B.3. □

Lemme 8.2.3.10. *Sous les hypothèses A, B et C, pour tous $N \geq M_0$, $\rho \in \mathcal{P}(\mathbb{X})$ et $y, y' \in \mathbb{Y}$, on a que*

$$\left\|\Phi(y', \rho, y) - \Phi_N(y', \rho, y)\right\|_{KR} \leq \delta\big(L_q(2\delta\overline{q} + 1) + \overline{q}\big)\varepsilon_N. \tag{8.41}$$

Démonstration. Soit $f \in \mathbb{L}^*(\mathbb{X})$. Alors

$$\begin{aligned}
&\left|\int_{\mathbb{X}} f(x)\Phi(y', \rho, y)(\mathrm{d}x) - \int_{\mathbb{X}} f(x)\Phi_N(y', \rho, y)(\mathrm{d}x)\right| \\
&\leq \mathbb{E}\left[\left|f(X)\frac{q(X, y' \mid \rho, y)}{q(\lambda, y' \mid \rho, y)} - f(X_N)\frac{q(X_N, y' \mid \rho, y)}{q(\lambda_N, y' \mid \rho, y)}\right|\right] \\
&\leq \mathbb{E}\left[q(X_N, y' \mid \rho, y)\,|f(X_N)|\left|\frac{1}{q(\lambda, y' \mid \rho, y)} - \frac{1}{q(\lambda_N, y' \mid \rho, y)}\right|\right] \\
&\quad + \mathbb{E}\left[\frac{|f(X)|}{q(\lambda, y' \mid \rho, y)}\left|q(X, y' \mid \rho, y) - q(X_N, y' \mid \rho, y)\right|\right] \\
&\quad + \mathbb{E}\left[\frac{q(X_N, y' \mid \rho, y)}{q(\lambda, y' \mid \rho, y)}\,|f(X) - f(X_N)|\right].
\end{aligned} \tag{8.42}$$

Le lemme 8.2.3.4 et la proposition 8.1.4.1 permettent d'obtenir

$$\begin{aligned}
&\left|\int_{\mathbb{X}} f(x)\Phi(y', \rho, y)(\mathrm{d}x) - \int_{\mathbb{X}} f(x)\Phi_N(y', \rho, y)(\mathrm{d}x)\right| \\
&\leq \overline{q}\left|\frac{1}{q(\lambda, y' \mid \rho, y)} - \frac{1}{q(\lambda_N, y' \mid \rho, y)}\right| + \delta L_q\varepsilon_N + \delta\overline{q}\varepsilon_N \\
&\leq 2\overline{q}\delta^2 L_q\varepsilon_N + \delta L_q\varepsilon_N + \delta\overline{q}\varepsilon_N,
\end{aligned} \tag{8.43}$$

d'où l'inégalité. □

Régularité de la fonction R

Proposition 8.2.3.11. *Sous l'hypothèse D, la fonction R appartient à l'ensemble $\mathbb{L}(\mathcal{P}(\mathbb{X}) \times \mathbb{Y})$.*

Démonstration. Soient $\rho, \rho' \in \mathcal{P}(\mathbb{X})$ et $y, y' \in \mathbb{Y}$. D'une part, on a que $|R(\rho, y)| \leq \|R\|_{\mathbb{L}}$. D'autre part, on écrit que

$$\left|R(\rho, y) - R(\rho', y')\right| \leq \int_{\mathbb{X}} \left|R(x, y) - R(x, y')\right| \rho(\mathrm{d}x) + \left|\int_{\mathbb{X}} R(x, y')\rho(\mathrm{d}x) - \int_{\mathbb{X}} R(x, y')\rho'(\mathrm{d}x)\right|. \tag{8.44}$$

L'hypothèse D implique que $|R(x,y) - R(x,y')| \leq \|R\|_{\mathbb{L}} |y - y'|$. Elle implique également que, à $y' \in \mathbb{Y}$ fixé, la fonction $x \mapsto R(x,y')$ est dans $\mathbb{L}(\mathbb{X})$. Donc la fonction $x \mapsto \dfrac{R(x,y')}{\|R\|_{\mathbb{L}}}$, à $y' \in \mathbb{Y}$ fixé, est dans $\mathbb{L}^*(\mathbb{X})$. L'équation précédente devient alors

$$\big|R(\rho,y) - R(\rho',y')\big| \leq \|R\|_{\mathbb{L}} \left[\big\|\rho - \rho'\big\|_{KR} + \big|y - y'\big|\right]. \tag{8.45}$$

On a donc bien le résultat voulu. □

Nous pouvons maintenant démontrer notre résultat d'approximation.

8.2.4 Un résultat d'approximation

Propriété stabilisatrice des opérateurs de Bellman

Dans toute la suite, nous utiliserons l'inégalité suivante, valable pour tous réels a, a', b, b' :

$$\big|\max(a;b) - \max(a';b')\big| \leq \big|a - a'\big| + \big|b - b'\big|. \tag{8.46}$$

Exhibons maintenant un espace stabilisé par les opérateurs de Bellman.

Proposition 8.2.4.1. *Sous les hypothèses B, C et D, l'opérateur de Bellman $\mathcal{B}$ stabilise l'ensemble $\mathbb{L}(\mathcal{P}(\mathbb{X}) \times \mathbb{Y})$. De plus, pour toute fonction $f \in \mathbb{L}(\mathcal{P}(\mathbb{X}) \times \mathbb{Y})$, on a que*

$$\|\mathcal{B}f\|_{\mathbb{L}} \leq \max\{\|R\|_{\mathbb{L}}\,; \|f\|_{\mathbb{L}}\} + \|R\|_{\mathbb{L}} + \|f\|_{\mathbb{L}} (\overline{q} + L_q)\big(1 + \delta\overline{q}(1 + \overline{q}\delta)\big). \tag{8.47}$$

Démonstration. Soient $f \in \mathbb{L}(\mathcal{P}(\mathbb{X}) \times \mathbb{Y})$, $\rho, \rho' \in \mathcal{P}(\mathbb{X})$ et $y, y' \in \mathbb{Y}$. Alors d'une part,

$$\begin{aligned}
|\mathcal{B}f(\rho,y)| &\leq \max\{|R(\rho,y)|\,; |Sf(\rho,y)|\} \\
&\leq \max\left\{\|R\|_{\mathbb{L}}\,; \int_{\mathbb{X}\times\mathbb{Y}} \big|f\big(\Phi(y',\rho,y),y'\big)\big|\, q(x',y' \mid \rho,y)\lambda(\mathrm{d}x')\nu(\mathrm{d}y')\right\} \\
&\leq \max\{\|R\|_{\mathbb{L}}\,; \|f\|_{\mathbb{L}}\}.
\end{aligned} \tag{8.48}$$

D'autre part, on a que

$$\begin{aligned}
\big|Sf(\rho,y) - Sf(\rho',y')\big| \leq{}& \mathbb{E}\left[\big|f\big(\Phi(Y,\rho,y),Y\big)\big|\,\big|q(X,Y \mid \rho,y) - q(X,Y \mid \rho',y')\big|\right] \\
&+ \mathbb{E}\left[q(X,Y \mid \rho',y')\,\big|f\big(\Phi(Y,\rho,y),Y\big) - f\big(\Phi(Y,\rho',y'),Y\big)\big|\right].
\end{aligned} \tag{8.49}$$

La proposition 8.1.4.1 permet d'obtenir que

$$\begin{aligned}
\big|Sf(\rho,y) - Sf(\rho',y')\big| \leq{}& \|f\|_{\mathbb{L}} (\overline{q} + L_q)\left[\big\|\rho - \rho'\big\|_{KR} + \big|y - y'\big|\right] \\
&+ \overline{q}\,\|f\|_{\mathbb{L}}\, \mathbb{E}\left[\big\|\Phi(Y,\rho,y) - \Phi(Y,\rho',y')\big\|_{KR}\right].
\end{aligned} \tag{8.50}$$

Le lemme 8.2.3.8 permet de majorer $\mathbb{E}\left[\|\Phi(Y,\rho,y) - \Phi(Y,\rho',y')\|_{KR}\right]$ et donc d'avoir

$$\big|Sf(\rho,y) - Sf(\rho',y')\big| \leq \|f\|_{\mathbb{L}} (\overline{q} + L_q)\big(1 + \overline{q}\delta(1 + \overline{q}\delta)\big)\left[\big\|\rho - \rho'\big\|_{KR} + \big|y - y'\big|\right]. \tag{8.51}$$

Donc en utilisant la proposition 8.1 et l'inégalité (8.46),

$$\begin{aligned}\left|\mathcal{B}f(\rho,y)-\mathcal{B}f(\rho',y')\right| &\leq \left|R(\rho,y)-R(\rho',y')\right|+\left|Sf(\rho,y)-Sf(\rho',y')\right| \\ &\leq \Big(\|R\|_{\mathbb{L}}+\|f\|_{\mathbb{L}}\left(\overline{q}+L_q\right)\big(1+\delta\overline{q}(1+\overline{q}\delta)\big)\Big)\left[\left\|\rho-\rho'\right\|_{KR}+\left|y-y'\right|\right]\end{aligned} \tag{8.52}$$

d'où le résultat. □

Proposition 8.2.4.2. *Sous les hypothèses A, B, C et D, pour tout $N \geq M_0$, l'opérateur de Bellman $\mathcal{B}_N$ stabilise l'ensemble $\mathbb{L}(\mathcal{P}(\mathbb{X})\times\mathbb{Y})$. De plus, pour toute $f\in\mathbb{L}(\mathcal{P}(\mathbb{X})\times\mathbb{Y})$, on a que*

$$\|\mathcal{B}_N f\|_{\mathbb{L}} \leq \max\left\{\|R\|_{\mathbb{L}}\,;\|f\|_{\mathbb{L}}\,\delta E_N\right\}+\|R\|_{\mathbb{L}}+\delta E_N\,\|f\|_{\mathbb{L}}\,(\overline{q}+L_q)(1+\overline{q}\delta E_N)^2. \tag{8.53}$$

Démonstration. La démonstration est similaire à celle de la proposition 8.2.4.1. On pourra la trouver à l'annexe B, section B.4. □

On a donc exhibé un espace stabilisé par les deux opérateurs $\mathcal{B}$ et $\mathcal{B}_N$. On peut désormais montrer notre résultat d'approximation.

Résultat principal

On commence par montrer trois propositions importantes avant de montrer le théorème principal.

Proposition 8.2.4.3. *Soient $N \geq M_0$ et $f\in\mathbb{L}(\mathcal{P}(\mathbb{X})\times\mathbb{Y})$. Sous les hypothèses A, B et C, on a que, pour tout $(\rho,y)\in\mathcal{P}(\mathbb{X})\times\mathbb{Y}$,*

$$|Sf(\rho,y)-S_Nf(\rho,y)| \leq \theta\,\|f\|_{\mathbb{L}}\,\varepsilon_N \tag{8.54}$$

où

$$\theta = L_q(1+2\overline{q}L_q)+\delta\overline{q}(\overline{q}+L_q(1+2\delta\overline{q})). \tag{8.55}$$

Démonstration. On écrit que

$$\begin{aligned}|Sf(\rho,y)-S_Nf(\rho,y)| \leq{}& \mathbb{E}\left[q(X,Y\mid\rho,y)\left|f\big(\Phi(Y,\rho,y),Y\big)-f\big(\Phi_N(Y,\rho,y),Y\big)\right|\right] \\ &+\mathbb{E}\left[\left|f\big(\Phi_N(Y,\rho,y),Y\big)\right|\left|q(X,Y\mid\rho,y)-q(X_N,Y\mid\rho,y)\right|\right] \\ &+\mathbb{E}\left[\left|f\big(\Phi_N(Y,\rho,y),Y\big)\right|q(X_N,Y\mid\rho,y)\left|\frac{1}{q(\lambda_N,\nu\mid\rho,y)}-1\right|\right].\end{aligned} \tag{8.56}$$

La proposition 8.1.4.1 implique que

$$\begin{aligned}|Sf(\rho,y)-S_Nf(\rho,y)| \leq{}& \overline{q}\,\|f\|_{\mathbb{L}}\,\mathbb{E}\left[\|\Phi(Y,\rho,y)-\Phi_N(Y,\rho,y)\|_{KR}\right] \\ &+\|f\|_{\mathbb{L}}\,L_q\varepsilon_N+\|f\|_{\mathbb{L}}\,\overline{q}\left|\frac{1}{q(\lambda_N,\nu\mid\rho,y)}-1\right|.\end{aligned} \tag{8.57}$$

Maintenant, on utilise le lemme 8.2.3.10 et le lemme 8.2.3.7. On a alors que

$$|Sf(\rho,y)-S_Nf(\rho,y)| \leq \overline{q}\,\|f\|_{\mathbb{L}}\,\delta\big(L_q(2\delta\overline{q}+1)+\overline{q}\big)\varepsilon_N+\|f\|_{\mathbb{L}}\,L_q\varepsilon_N+2\,\|f\|_{\mathbb{L}}\,\overline{q}L_q\varepsilon_N, \tag{8.58}$$

d'où le résultat. □

Corollaire 8.2.4.4. *Soient $N \geq M_0$ et $f \in \mathbb{L}(\mathcal{P}(\mathbb{X}) \times \mathbb{Y})$. Sous les hypothèses A, B et C, on a que, pour tout $(\rho, y) \in \mathcal{P}(\mathbb{X}) \times \mathbb{Y}$,*

$$|\mathcal{B}f(\rho, y) - \mathcal{B}_N f(\rho, y)| \leq \theta \|f\|_{\mathbb{L}} \varepsilon_N. \tag{8.59}$$

Démonstration. Il suffit de savoir majorer $|Sf(\rho, y) - S_N f(\rho, y)|$, d'après l'inégalité (8.46). On utilise alors la proposition précédente pour avoir l'inégalité voulue. □

Proposition 8.2.4.5. *Soient $N \geq M_0$ et $f \in \mathbb{L}\big(\mathcal{P}(\mathbb{X}) \times \mathbb{Y}\big)$. Pour tout $k \in [\![1; N_0]\!]$, sous les hypothèses A, B et C, pour tout couple $(\rho, y) \in \mathcal{P}(\mathbb{X}) \times \mathbb{Y}$,*

$$\left|\mathcal{B}^k f(\rho, y) - \mathcal{B}_N^k f(\rho, y)\right| \leq \theta \varepsilon_N \sum_{l=0}^{k-1} \left\|\mathcal{B}^l f\right\|_{\mathbb{L}}. \tag{8.60}$$

Démonstration. Montrons cette proposition par récurrence sur k. Le cas $k = 1$ est démontré par le corollaire 8.2.4.4. Supposons maintenant que la proposition soit vraie pour un certain rang $k \in [\![1; N_0 - 1]\!]$. On écrit alors que

$$\begin{aligned}\left|\mathcal{B}^{k+1} f(\rho, y) - \mathcal{B}_N^{k+1} f(\rho, y)\right| \leq{}& \left|\mathcal{B}(\mathcal{B}^k f)(\rho, y) - \mathcal{B}_N(\mathcal{B}^k f)(\rho, y)\right| \\ &+ \left|\mathcal{B}_N(\mathcal{B}^k f)(\rho, y) - \mathcal{B}_N(\mathcal{B}_N^k f)(\rho, y)\right|. \end{aligned} \tag{8.61}$$

On utilise le corollaire 8.2.4.4 pour majorer le premier terme puisque, par la proposition 8.2.4.1, $\mathcal{B}$ stabilise l'ensemble $\mathbb{L}\big(\mathcal{P}(\mathbb{X}) \times \mathbb{Y}\big)$. Pour le deuxième terme, d'après l'inégalité (8.46), on doit majorer la quantité $\left|S_N(\mathcal{B}^k f)(\rho, y) - S_N(\mathcal{B}_N^k f)(\rho, y)\right|$. On écrit que

$$\begin{aligned}&\left|S_N(\mathcal{B}^k f)(\rho, y) - S_N(\mathcal{B}_N^k f)(\rho, y)\right| \\ &\leq \int_{\mathbb{Y}} \left|\mathcal{B}^k f\big(\Phi_N(y', \rho, y), y'\big) - \mathcal{B}_N^k f\big(\Phi_N(y', \rho, y), y'\big)\right| Q_N^{\mathbb{Y}}(\mathrm{d}y' \mid \rho, y) \\ &\leq \theta \varepsilon_N \sum_{l=0}^{k-1} \left\|\mathcal{B}^l f\right\|_{\mathbb{L}} \end{aligned} \tag{8.62}$$

par hypothèse de récurrence. On obtient alors que

$$\left|\mathcal{B}^{k+1} f(\rho, y) - \mathcal{B}_N^{k+1} f(\rho, y)\right| \leq \theta \left\|\mathcal{B}^k f\right\|_{\mathbb{L}} \varepsilon_N + \theta \varepsilon_N \sum_{l=0}^{k-1} \left\|\mathcal{B}^l f\right\|_{\mathbb{L}}. \tag{8.63}$$

La récurrence est donc établie. □

On sait désormais contrôler l'erreur de chaque quantité $\left|v_k - v_k^{(N)}\right|$. Il ne nous reste plus qu'à énoncer notre théorème final, qui sera facile à démontrer grâce aux propositions préliminaires de cette sous-partie.

Théorème 8.2.4.6. *Sous les hypothèses A, B, C et D, pour tout $N \geq M_0$,*

$$\left|\mathcal{R}^*_{\mathcal{M}'}(\delta_{x^0}, y^0) - \mathcal{R}^*_{\mathcal{M}', N}(\delta_{x^0}, y^0)\right| \leq \theta \varepsilon_N \sum_{l=0}^{N_0 - 1} \left\|\mathcal{B}^l R\right\|_{\mathbb{L}}. \tag{8.64}$$

En particulier, on a donc que $\left|\mathcal{R}^*_{\mathcal{M}'}(\delta_{x^0}, y^0) - \mathcal{R}^*_{\mathcal{M}', N}(\delta_{x^0}, y^0)\right| = \mathrm{O}(N^{-\frac{1}{m}})$.

DÉMONSTRATION. On a que $\mathcal{R}^*_{\mathcal{M}} \equiv \mathcal{B}^{N_0} R \equiv v_{N_0}$ et $\mathcal{R}^*_{\mathcal{M},N} \equiv \mathcal{B}_N^{N_0} R \equiv v_{N_0}^{(N)}$. Or, par la proposition 8.1, $R \in \mathbb{L}\big(\mathcal{P}(\mathbb{X}) \times \mathbb{Y}\big)$. Donc on applique la proposition 8.2.4.5 et on conclut au résultat énoncé. La vitesse est donnée par le théorème 7.2.0.6. □

Remarque. Ce théorème nous assure donc que $\left|\mathcal{R}^*(x^0, y^0) - \mathcal{R}^*_{\mathcal{M}',N}(\delta_{x^0}, y^0)\right| = \mathrm{O}(N^{-\frac{1}{m}})$.

8.3 Seconde discrétisation de l'espace d'états

Les algorithmes de résolution que nous connaissons pour le formalisme des MDP ne sont pas encore applicables en l'état. En effet, si on s'est ramenés à un processus complètement observable à espace d'états de dimension finie, il est toujours a priori infini. Nous devons donc discrétiser une nouvelle fois l'espace d'états. Pour ce faire, nous allons reprendre les idées contenues dans l'article [BP03] qui consistent à quantifier une chaîne de Markov à valeurs dans $\mathbb{R}^p$ associée à un problème d'arrêt optimal complètement observable. Dans la suite, on identifiera $\mathcal{P}(\mathbb{X}_N)$ comme un simplexe de $[0;1]^N$ et par abus de notation, on notera $|\rho|$ la norme euclidienne de la mesure ρ vue comme un vecteur de $[0;1]^N$.

Notons un fait important : pour toute fonction $f \in \mathbb{L}^*(\mathbb{X})$ et toutes mesures $\rho, \rho' \in \mathcal{P}(\mathbb{X}_N)$ telles que $\rho = \sum_{i=1}^N \rho_i \delta_{x_i}$ et $\rho' = \sum_{i=1}^N \rho'_i \delta_{x_i}$, on a que

$$\left|\int_{\mathbb{X}} f \mathrm{d}(\rho - \rho')\right| \leq \sum_{i=1}^N \left|\rho_i - \rho'_i\right| . \left|f(x_i)\right| \leq \sum_{i=1}^N \left|\rho_i - \rho'_i\right| \leq \sqrt{N} \left|\rho - \rho'\right| .$$

Il vient alors l'inégalité

$$\left\|\rho - \rho'\right\|_{KR} \leq \sqrt{N} \left|\rho - \rho'\right| \tag{8.65}$$

pour toutes mesures $\rho, \rho' \in \mathcal{P}(\mathbb{X}_N)$.

8.3.1 Discrétisation du processus

Soient deux entiers naturels $M \in \mathbb{N}^*$, $N \geq M_0$ et $(P_k^{(N)}, Y_k^{(N)})_{0 \leq k \leq N_0}$ une chaîne de Markov à valeurs dans $\mathcal{P}(\mathbb{X}_N) \times \mathbb{Y}$, de loi de transition S_N et d'état initial (δ_{x^0}, y^0). On va donc en effectuer la L^2-quantification optimale : on obtient alors une chaîne de Markov $(P_k^{(M,N)}, Y_k^{(M,N)})_{0 \leq k \leq N_0}$ non stationnaire qui, au temps $k \in [\![0; N_0]\!]$, est à valeurs dans un ensemble $\Gamma_{M,N}^k$ fini de cardinal M.

Dans toute la suite, nous noterons

$$\eta_M^{(N,k)} = \left\|(P_k^{(N)}, Y_k^{(N)}) - (P_k^{(M,N)}, Y_k^{(M,N)})\right\|_2 .$$

Le théorème 7.2.0.6 donne que $\eta_M^{(N,k)} = \mathrm{O}(M^{-\frac{1}{N+n}})$.

8.3.2 Enveloppe de Snell et convergence

Nous allons montrer que les équations de Bellman approchées définies avec les équations (8.23) peuvent être interprétées comme l'enveloppe de Snell d'un problème d'arrêt optimal associé à la chaîne de Markov $(P_k^{(N)}, Y_k^{(N)})_{0\leq k\leq N_0}$.

Nous avons écrit les équations de Bellman approchées de cette manière :

$$\begin{cases} v_0^{(N)}(\rho, y) = R(\rho, y) \\ v_k^{(N)}(\rho, y) = \mathcal{B}_N v_{k-1}^{(N)}(\rho, y) \end{cases} \quad \forall\, (\rho, y) \in \mathcal{P}(\mathbb{X}) \times \mathbb{Y}.$$

Définissons donc la suite de variables aléatoires $(T_k^{(N)})_{0\leq k\leq N_0}$ par

$$T_k^{(N)} = v_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) \text{ pour tout } k \in [\![0; N_0]\!].$$

Par définition, le noyau S_N peut être vu comme un noyau sur $\mathcal{P}(\mathbb{X}_N)\times\mathbb{Y}$ sachant $\mathcal{P}(\mathbb{X}_N)\times\mathbb{Y}$: c'est la loi conditionnelle de $(P_k^{(N)}, Y_k^{(N)})$ sachant $(P_{k-1}^{(N)}, Y_{k-1}^{(N)})$ pour tout $1 \leq k \leq N_0$. Vient alors l'égalité

$$\begin{aligned} S_N v_k^{(N)}(P_{N_0-k-1}^{(N)}, Y_{N_0-k-1}^{(N)}) &= \int_{\mathcal{P}(\mathbb{X})\times\mathbb{Y}} v_k^{(N)}(\rho', y') S_N(\mathrm{d}\rho', \mathrm{d}y' \mid P_{N_0-k-1}^{(N)}, Y_{N_0-k-1}^{(N)}) \\ &= \mathbb{E}\left[v_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) \;\middle|\; (P_{N_0-k-1}^{(N)}, Y_{N_0-k-1}^{(N)}) \right] \\ &= \mathbb{E}\left[T_k^{(N)} \;\middle|\; (P_{N_0-k-1}^{(N)}, Y_{N_0-k-1}^{(N)}) \right]. \end{aligned} \tag{8.66}$$

S'ensuit alors une autre définition des variables $T_k^{(N)}$ qui s'exprime par récurrence par

$$\begin{cases} T_0^{(N)} = R(P_{N_0}^{(N)}, Y_{N_0}^{(N)}), \\ T_k^{(N)} = \mathcal{B}_N v_{k-1}^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) = \max\left\{ R(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}); \mathbb{E}\left[T_{k-1}^{(N)} \;\middle|\; (P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) \right] \right\}. \end{cases}$$

Nous venons d'obtenir l'enveloppe de Snell que nous cherchions. Remarquons au passage que $T_{N_0}^{(N)} = v_{N_0}^{(N)}(P_0^{(N)}, Y_0^{(N)}) = \mathcal{R}^*_{\mathcal{M}',N}(\delta_{x^0}, y^0)$.

Donc, de la même manière, on définit la suite de variables aléatoires $(T_k^{(M,N)})_{0\leq k\leq N_0}$ par récurrence par

$$\begin{cases} T_0^{(M,N)} = R(P_{N_0}^{(M,N)}, Y_{N_0}^{(M,N)}), \\ T_k^{(M,N)} = \max\left\{ R(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)}); \mathbb{E}\left[T_{k-1}^{(M,N)} \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)}) \right] \right\}. \end{cases}$$

On pose $T_{N_0}^{(M,N)} = \mathcal{R}^*_{\mathcal{M}',N,M}(\delta_{x^0}, y^0)$. Démontrons que cette quantité approche $\mathcal{R}^*_{\mathcal{M}',N}(\delta_{x^0}, y^0)$.

Théorème 8.3.2.1. *Sous les hypothèses A, B, C et D, pour tous $k \in [\![1; N_0]\!]$, $N \geq M_0$ et $M \in \mathbb{N}^*$, on a que*

$$\left\| T_k^{(N)} - T_k^{(M,N)} \right\|_2 \leq \sqrt{N} \sum_{l=0}^{k} d_l^{(N)} \eta_M^{(N,N_0-l)}. \tag{8.67}$$

où

$$d_l^{(N)} = \begin{cases} \sqrt{2}\,\|R\|_{\mathbb{L}} \ \ si\ l = 0, \\ \|R\|_{\mathbb{L}} + 2\left\|v_{l-1}^{(N)}\right\|_{\mathbb{L}} \delta E_N(\overline{q} + L_q)(1 + \overline{q}\delta E_N)^2 \ \ si\ l \in [\![1; k]\!]. \end{cases} \tag{8.68}$$

En particulier, pour $k = N_0$*,*

$$\left|\mathcal{R}^*_{\mathcal{M}',N}(\delta_{x^0}, y^0) - \mathcal{R}^*_{\mathcal{M}',N,M}(\delta_{x^0}, y^0)\right| \leq \sqrt{N} \sum_{l=0}^{N_0} d_{N_0-l}^{(N)} \eta_M^{(N,l)}. \tag{8.69}$$

Démonstration. Consulter l'annexe B, section B.5. □

Remarque. Dans ce théorème, N est supposé fixé, ce qui ne pose pas de problème pour la convergence.

8.4 Une application numérique

Dans cette section, nous allons présenter un exemple numérique d'application des résultats démontrés dans ce chapitre. On se propose donc d'étudier un problème d'arrêt optimal qui vérifie nos hypothèses. Cette application est librement adaptée d'un problème de contrôle de réservoir d'eau que l'on peut trouver dans la section 1.3 de [HL89]. Ce type d'application est crucial notamment dans les pays soumis à un important stress hydrique.

Considérons un réservoir d'eau de capacité $K > 0$ finie rempli par les eaux de pluie. On contrôle le volume du réservoir à intervalles réguliers. Chaque pluie remplit le réservoir. Le volume de liquide contenu dans le réservoir n'est accessible qu'à travers des mesures d'icelui. Ces mesures ne sont toutefois pas exactes. L'objectif du problème est de fermer le réservoir lorsqu'il contient un volume d'eau au plus près d'une valeur $\alpha \in]0; K[$ sur la seule connaissance des mesures dont on dispose.

On modélise ce problème par une chaîne de Markov $(\mathcal{X}_k^1, \mathcal{X}_k^2, \mathcal{Y}_k)_{0 \leq k \leq N_0}$ dont l'horizon N_0 est fini. Les variables non observables $(\mathcal{X}_k^1, \mathcal{X}_k^2)$ sont à valeurs dans

$$\mathbb{X} = (]0; K[\times \{0\}) \cup \{(0; -1)\} \cup \{(K; 1)\}\,.$$

Les variables observables $(\mathcal{Y}_k)$ sont à valeurs dans

$$\mathbb{Y} = [0; K].$$

Ici, $(\mathcal{X}_k^1)$ représente la suite des volumes d'eau que contient le réservoir. Les variables $(\mathcal{Y}_k)$ symbolisent les mesures que l'on fait de ces volumes.

Remarque. Les variables $\mathcal{X}_k^2$ pourront prendre trois valeurs selon celle de $\mathcal{X}_k^1$: -1 si $\mathcal{X}_k^1$ vaut 0, 1 si $\mathcal{X}_k^1$ vaut K et 0 sinon. Elles ne sont pas observables, mais permettent d'imposer à l'espace $\mathbb{X}$ une topologie particulière qui fera en sorte que nos hypothèses soient vérifiées.

On modélise la dynamique du processus $(\mathcal{X}_k^1, \mathcal{X}_k^2, \mathcal{Y}_k)$ par le système suivant

$$\begin{cases} \mathcal{X}_{k+1}^1 = (\mathcal{X}_k^1 + \xi_k) \wedge K \\ \mathcal{X}_{k+1}^2 = \mathbb{1}_{\{K\}}(\mathcal{X}_{k+1}^1) - \mathbb{1}_{\{0\}}(\mathcal{X}_{k+1}^1) \quad \forall\, k \in [\![0; N_0 - 1]\!] \\ \mathcal{Y}_{k+1} = (\mathcal{X}_{k+1}^1 + \psi_k)_+ \wedge K \end{cases} \tag{8.70}$$

où (ξ_k) et (ψ_k) sont des suites de variables aléatoires indépendantes et identiquement distribuées de densités respectivement f sur $\mathbb{R}_+$ et g sur $\mathbb{R}$.

Enfin, on considère la fonction de récompense suivante

$$R(x_1, x_2, y) = K - |x_1 - \alpha|$$

dont une représentation graphique de cette fonction est donnée sur la figure 8.1.

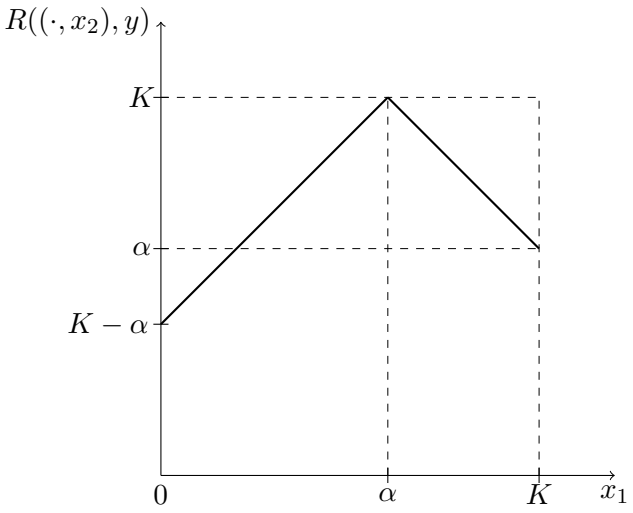

FIGURE 8.1 – Tracé de la fonction $x_1 \mapsto R((x_1, x_2), y)$

La proposition suivante établit que ce problème d'arrêt optimal respecte bien nos hypothèses, sous quelques conditions aisément vérifiables en pratique.

Proposition 8.4.0.1. *On suppose que f est L_f-lipschitzienne sur $[0; K]$, de sorte que sa fonction de répartition F vérifie $F(K) < 1$. On suppose que g est strictement positive sur $[-K; K]$ et L_g-lipschitzienne sur cet intervalle. On suppose enfin que la fonction de répartition G de g vérifie $G(-K) > 0$ et $G(0) < 1$. Sous ces conditions, ce problème vérifie les hypothèses A, B, C et D.*

DÉMONSTRATION. Consulter l'annexe B, section B.6. □

Remarque. Les mesures λ et ν exhibées dans cette démonstration ne sont pas absolument continues par rapport à la mesure de Lebesgue. Ainsi, cette application ne rentre pas dans le cadre de travail de [YZ13, ZFM10, Zho13]. L'exemple d'arrêt optimal proposé dans ces travaux ne rentre pas dans notre propre cadre, notamment car la densité de la loi de transition n'est pas bornée.

Pour l'application numérique, on suppose que les ξ_k suivent une loi exponentielle de paramètre 5. Nous avons choisi pour les ψ_k une loi normale $\mathcal{N}(0;\sigma^2)$, où nous avons considéré deux valeurs pour la variance : $\sigma^2 = 0{,}001$ (ce qui correspond au cas où les mesures sont assez précises) et $\sigma^2 = 0{,}01$ (qui entraîne un écart-type de 0,1, ce qui est un bruit raisonnable au regard du problème). On prendra de plus $N_0 = 10$, $K = 1$ et $\alpha = 0{,}5$. Nous supposerons que l'état initial $(\mathcal{X}_0^1, \mathcal{X}_0^2, Y_0)$ est $(0;-1;0)$.

Nous allons illustrer les convergences que l'on a démontrées précédemment. Les tests de cette partie ont été effectués avec un Mac Pro 2,7 GHz 12-Core avec 64 Go de RAM.

Le théorème 8.2.4.6 assure la convergence de $\mathcal{R}^*_{\mathcal{M}',N}(\delta_{(0,-1)},0)$ vers $\mathcal{R}^*(0,-1,0)$ à une vitesse en $\mathrm{O}(N^{-1})$. De plus, si on fixe N, le théorème 8.3.2.1 assure la convergence de $\mathcal{R}^*_{\mathcal{M}',N,M}(\delta_{(0,-1)},0)$ vers $\mathcal{R}^*_{\mathcal{M}',N}(\delta_{(0,-1)},0)$ à une vitesse en $\mathrm{O}(M^{-\frac{1}{N+1}})$. En suivant la méthode développée dans ce chapitre, nous avons effectué deux quantifications successives, l'une suivant l'algorithme CLVQ pour les variables aléatoires (cf. chapitre 7, section 7.3.1), et l'autre suivant l'algorithme CLVQ pour les chaînes de Markov (cf. chapitre 7, section 7.4.2). Nous avons ainsi pu déterminer la valeur optimale approchée $\mathcal{R}^*_{\mathcal{M}',N,M}(\delta_{(0,-1)},0)$ via l'algorithme standard de résolution des problèmes à horizon fini exposé au chapitre 2, section 2.1.1.

Le tableau 8.1 et le graphe 8.2 montrent la variation de la valeur optimale approchée en fonction du nombre de points M des grilles $\Gamma^k_{M,N}$ pour deux valeurs de N. Les tableaux 8.2 et 8.3, comme les graphes 8.3 et 8.4, montrent la variation de la valeur optimale en fonction du nombre de points N de la grille $\mathbb{X}_N$ lorsqu'on fixe le nombre de points M. On doit observer deux phénomènes de convergence : le premier est une convergence lorsque M tend vers $+\infty$ à N fixé, et le second est une convergence lorsque N tend vers $+\infty$. Pour toutes les valeurs de N testées, la convergence lorsque M tend vers l'infini est manifeste d'après le tableau 8.1 et sur la figure 8.2. Le phénomène de convergence lorsque N croît est corroboré par ce qu'on observe sur les tableaux 8.2 et 8.3, ainsi que sur les figures 8.3 et 8.4.

Il est manifeste de constater que la valeur optimale approchée se comporte très différemment selon la valeur de la variance de la loi normale qui bruite les mesures des $\mathcal{X}_k^1$. En effet, la fonction de récompense est calculée à partir des seules valeurs du processus $(\mathcal{X}_k^1)$. Lorsque la variance vaut $\sigma^2 = 0{,}001$ (autrement dit l'écart-type est environ de 0,03), la précision des mesures entraîne qu'il est plus facile de s'arrêter au bon moment. En revanche, lorsque la variance vaut $\sigma^2 = 0{,}01$ (donc l'écart-type est de 0,1), les mesures sont plus dispersées, et l'arrêt intervient plus souvent à un moment inopportun alors que l'on pouvait observer une valeur proche de 0,5.

Enfin, il est à signaler un fait important : le fléau de la dimension ne nous permet pas d'obtenir des résultats en temps raisonnable pour des valeurs de N trop grandes, puisque nous procédons à la quantification d'une chaîne de Markov de dimension $N+1$ (pour $N = 300$, le calcul avec $M = 10000$ demande environ deux jours). En effet, une quantification précise demande une grande valeur de N, mais aussi une grande valeur de M : plus N est grand, plus M devra l'être aussi, ce qui demandera de manipuler des données d'une taille considérable et allongera d'autant plus le temps de calcul. Pour les résultats présentés dans cette section, nous avons poussé à leur limite les moyens informatiques dont nous disposions.

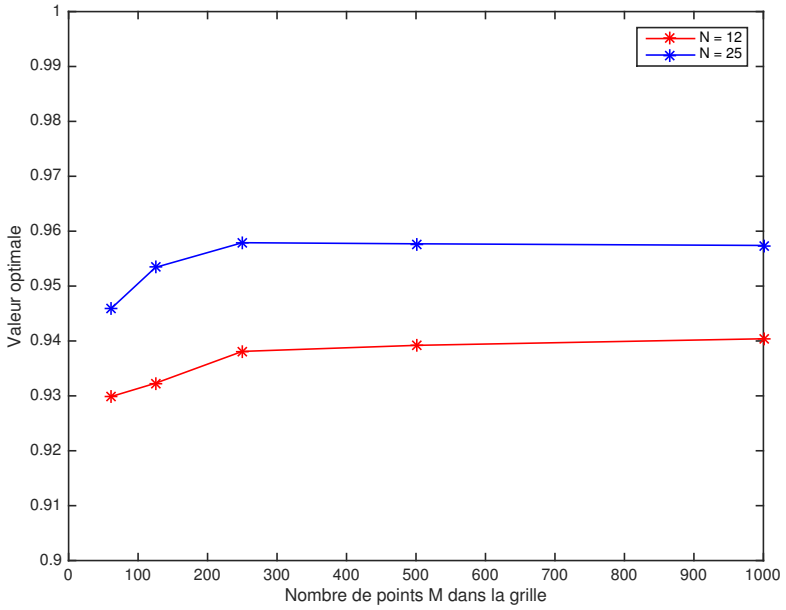

FIGURE 8.2 – Évolution de la valeur optimale $\mathcal{R}^*_{\mathcal{M}',N,M}(\delta_{(0,-1)}, 0)$ en fonction du nombre de points M des grilles $\Gamma^k_{M,N}$, pour différentes valeurs de N.

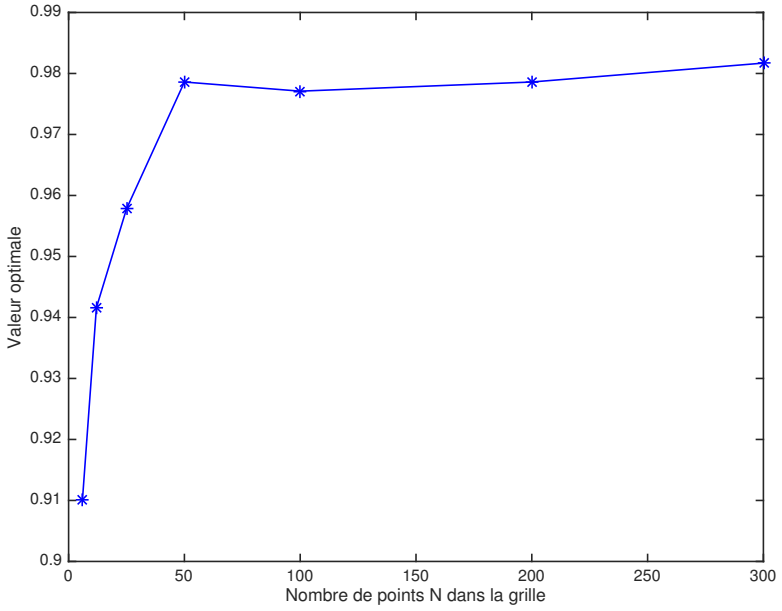

FIGURE 8.3 – Évolution de la valeur optimale $\mathcal{R}^*_{\mathcal{M}',N,M}(\delta_{(0,-1)}, 0)$ calculée pour $M = 10000$ points et $\sigma^2 = 0{,}001$ en fonction de N.

Valeurs de M	$N = 12$	$N = 25$
62	0,9299	0,946
125	0,9323	0,9534
250	0,9381	0,9579
500	0,9392	0,9577
1000	0,9404	0,9574

TABLEAU 8.1 – Valeurs de $\mathcal{R}^*_{\mathcal{M}',N,M}(\delta_{(0,-1)}, 0)$ en fonction du nombre de points M des grilles $\Gamma^k_{M,N}$, pour différentes valeurs de N et $\sigma^2 = 0{,}001$.

Valeurs de N	Valeur optimale
6	0,9102
12	0,9416
25	0,9578
50	0,9786
100	0,9771
200	0,9786
300	0,9817

TABLEAU 8.2 – Valeurs de $\mathcal{R}^*_{\mathcal{M}',N,M}(\delta_{(0,-1)}, 0)$ calculée pour $M = 10000$ points et $\sigma^2 = 0{,}001$ en fonction de N.

Valeurs de N	Valeur optimale
6	0,9078
12	0,9063
25	0,8659
50	0,8526
100	0,6577
200	0,6587
300	0,6595

TABLEAU 8.3 – Valeurs de $\mathcal{R}^*_{\mathcal{M}',N,M}(\delta_{(0,-1)}, 0)$ calculée pour $M = 5000$ points et $\sigma^2 = 0{,}01$ en fonction de N.

8.5 Conclusion

Le travail que nous avons présenté dans ce chapitre s'est déroulé en plusieurs étapes. Nous sommes partis de notre problème d'arrêt optimal partiellement observé pour en construire une modélisation suivant le formalisme des POMDP. Il a été nécessaire de démontrer l'équivalence des deux formulations. Selon la technique expliquée dans [BR11], nous avons dérivé de ce POMDP un MDP complètement observable dont l'espace des états est de dimension infinie. Nous avons donc procédé à un premier travail de discrétisation qui consistait à approcher le MDP complètement observable par un autre à espace d'états de dimension finie. Nous avons utilisé pour ce faire la quantification, dont le principe et les principaux résultats ont été détaillés au chapitre 7. Enfin, une deuxième étape de discrétisation nous a permis d'approcher notre MDP à espace d'états de dimension finie par un MDP à espace d'états fini. Ainsi, la valeur optimale de ce dernier est calculable par ordinateur. Les résultats que nous avons montrés dans ce chapitre établissent la convergence de la valeur optimale approchée vers la valeur optimale du POMDP initial.

Toutefois, si la convergence théorique est assurée, la méthode que nous proposons souffre du fléau de la dimension. En effet, on effectue en fin de compte deux quantifications successives. Plus la première est précise, plus la deuxième nécessitera du temps. Les moyens informatiques dont nous disposions nous ont permis d'avoir des résultats assez précis néanmoins. Cette observation

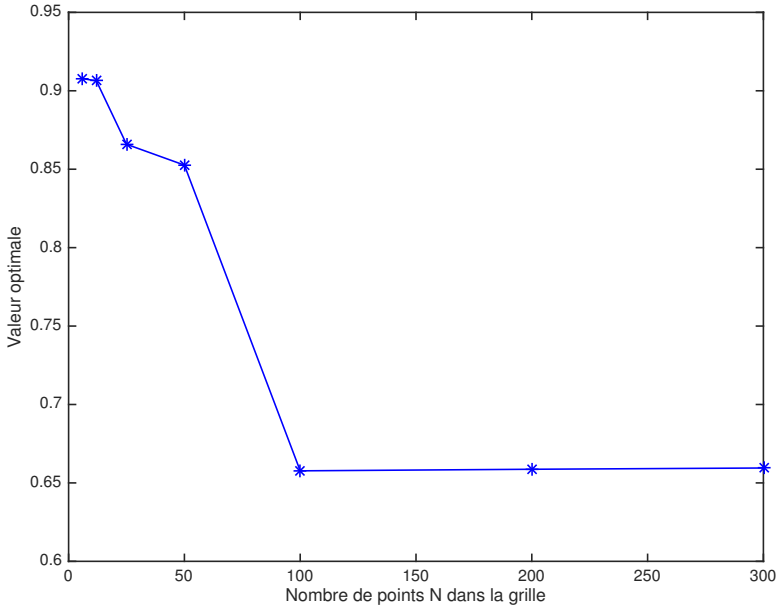

FIGURE 8.4 – Évolution de la valeur optimale $\mathcal{R}^*_{\mathcal{M}',N,M}(\delta_{(0,-1)}, 0)$ calculée pour $M = 5000$ points et $\sigma^2 = 0{,}01$ en fonction de N.

soulève naturellement la question d'une méthode d'approximation alternative qui serait moins coûteuse en temps. De même, parmi les hypothèses qui assurent la convergence théorique s'en trouve une qui est assez forte : il faut savoir minorer uniformément une espérance. L'affaiblissement de cette hypothèse constitue une éventuelle suite à ce travail.

Conclusion

Les processus markoviens décisionnels apportent un élément de réponse à la question de la bonne (ou plutôt de la meilleure) décision. En effet, il s'agit d'un formalisme que l'on retrouve dans des domaines très variés et qui convient pour des problèmes d'optimisation décisionnelle séquentielle. Ce travail de thèse a été l'occasion d'entrevoir l'étendue de cette théorie au travers de deux applications, l'une industrielle, l'autre théorique. Dans les deux cas ressortent deux difficultés majeures.

D'une part, la modélisation d'une situation ou d'un système est un véritable travail à part entière, surtout si le problème est complexe. Le MDP proposé pour l'application d'Airbus a procédé d'un raisonnement sur le pas de temps. Nous nous sommes en effet aperçus que plusieurs échelles temporelles coexistaient dans la chaîne de montage, ce qui a aiguillé notre réflexion vers un modèle dont la loi de transition est événementielle, c'est-à-dire que le système passe à l'état suivant dès qu'une des variables qui le décrivent change. Cette modélisation, commode pour plusieurs raisons, induit un problème de taille : les techniques d'optimisation standard ne sont pas applicables. Même pour des problèmes en apparence plus simples, comme l'exemple de POMDP étudié au chapitre 8, il nous a fallu adapter l'espace des états pour respecter le cadre théorique développé. En l'occurrence, changer la topologie de l'espace d'états en ajoutant une variable permet de vérifier les hypothèses avec la distance euclidienne usuelle.

D'autre part, la phase d'optimisation peut présenter quelques embûches. Par exemple, comme notre étude de l'arrêt optimal l'a montré, il est nécessaire, dans des problèmes où l'espace des états n'est pas fini, de se concentrer sur des problèmes plus simples qui approchent d'une certaine manière les quantités à calculer de sorte à pouvoir utiliser les algorithmes classiques. Dans ce cas, selon la méthode d'approximation choisie, des difficultés numériques peuvent survenir. Quant à l'application d'Airbus, comme nous l'avons évoqué avant, il a fallu se tourner vers des techniques qui calculent une approximation de la stratégie et de la valeur optimales. Ces algorithmes, développés depuis un peu plus d'une décennie, nécessitent seulement de savoir simuler le MDP à optimiser. Ce cadre de travail s'est avéré compatible avec notre modélisation de la chaîne de montage.

L'application industrielle d'Airbus a donc été l'occasion de modéliser un problème concret et complexe. Ce travail a permis l'implémentation d'un simulateur en MATLAB dans un premier temps, puis d'un portage d'icelui en langage C afin d'améliorer son temps d'exécution. Après une brève revue des solutions d'optimisation envisageables, nous en avons sélectionné une, l'algorithme ASA, pour répondre au problème posé. En l'occurrence, cet algorithme nous permet d'obtenir une stratégie de bonne qualité (la convergence de l'algorithme étant asymptotique, il s'agit d'une approximation d'une stratégie optimale) sous une forme implémentable et utilisable en pratique par les industriels. Il s'agit en fin de compte d'une stratégie déterministe markovienne en forme de tableau qui donne la décision à prendre selon l'état et le temps. Ainsi, nous avons pu établir que la question de la capacité optimale de stockage des SRM n'a pas de réponse triviale : elle dépend en effet du calendrier. C'est donc un premier pas vers l'industrialisation du procédé voulue par Airbus. C'est une partie de la suite du travail prévue. Dès lors, il est naturel d'envisager d'appliquer l'algorithme ASA à des modèles plus complexes. En effet, la chaîne de montage décrite dans le chapitre 3 n'est qu'une partie de la chaîne de montage globale du lanceur. Cette dernière est toutefois toujours en cours d'élaboration par les ingénieurs d'Airbus. On peut également prolonger le travail fourni dans d'autres directions, notamment en travaillant sur une hypothèse simplificatrice bien particulière : celle qui énonce que le calendrier est connu sur les trente ans (ou, ce qui est équivalent, découvert d'année en année). Les discussions avec Airbus nous ont fait envisager le cas où le calendrier serait connu un petit nombre d'années à l'avance. Dès lors, il faut envisager une toute autre modélisation du problème notamment au niveau de la décision.

L'application théorique des MDP aux problèmes d'arrêt optimal partiellement observables consiste en une proposition d'approximation de tels problèmes via la technique de la quantification. Nous avons tenté de contribuer à une branche des MDP pour laquelle la littérature ne présente encore que peu de références. Il s'agit d'une approximation en deux temps : d'abord, nous discrétisons l'espace des états non observables afin de se ramener à un espace de dimension finie. Ensuite, nous utilisons des techniques développées par d'autres auteurs, comme [BP03] ou [PRS05], afin d'obtenir un MDP complètement observable à espace d'états fini. Seulement, ces deux quantifications successives engendrent des problèmes numériques importants, liés au fléau de la dimension. Naturellement, ces problèmes nous amènent à nous demander si d'autres approches seraient plus efficaces, comme par exemple l'utilisation d'inégalités de concentration à l'instar de [DPR15]. Mais plus directement liée au travail exposé dans le chapitre 8, la question de l'affaiblissement de l'hypothèse C, qui demande la minoration uniforme d'une espérance, est un possible prolongement de ce travail. On peut également étudier ce qui se passe lorsqu'on considère une fonction de récompense non bornée. Néanmoins, on pourrait songer à d'autres axes d'étude. Par exemple, nous n'avons pas du tout évoqué l'approximation du temps d'arrêt optimal. Cette question a été abordée en partie dans [BP03], où les auteurs montrent la convergence en probabilité du temps d'arrêt optimal pour le MDP observable à espace d'états fini vers le temps d'arrêt optimal pour le MDP observable à espace d'états de dimension finie. Quid de sa convergence vers le temps d'arrêt optimal du POMDP initial ? C'est une question qu'il serait très intéressant d'examiner. Enfin, pour aller encore plus loin, on peut imaginer l'ajout de contraintes

au problème d'arrêt optimal partiellement observé. Dès lors, comme dans [DP10] ou [DPR13], on pourrait envisager d'utiliser des techniques de programmation linéaire. La démarche à adopter serait alors très différente.

En fin de compte, les contributions apportées par cette thèse posent les jalons d'un travail de longue haleine à la fois théorique et appliqué, pour lequel beaucoup de questions passionnantes cherchent encore leur (éventuelle ?) réponse. Nous avons tenté, à notre modeste niveau, de créer un nouveau lien entre recherche mathématique et industrie autour d'une belle théorie aux nombreuses applications. Puisse ce lien s'avérer aussi fructueux que nous l'espérons.

Quatrième partie

Annexes

A Simulation et implémentation

Cette annexe reprend les parties les plus techniques concernant la simulation de la chaîne de montage et l'implémentation des algorithmes de résolution.

A.1 Simulateur de la chaîne de montage

Dans cette partie, on s'intéresse aux codes de la simulation de la chaîne décrite au chapitre 3. Les notations et noms des variables sont ceux donnés au chapitre 4.

A.1.1 Routines

Les quatre routines présentées ici prennent en argument l'état courant du processus et ses paramètres.

Fonction `verifB`

1. Si `stock_I` ≥ 1 ET `etat_B1` $= 0$ ET (`stock_S` $\leq S_{\text{SRM}} - 2$ OU (`etat_B2` $= 0$ ET `stock_S` $= S_{\text{SRM}} - 1$)) :
   (a) `stock_I` $\rightarrow -1$,
   (b) `etat_B1` $\leftarrow 1$,
   (c) `tps_B1` $\leftarrow \mathcal{U}(\{9;9{,}5\})$,
   (d) Si `tps_I` $= +\infty$, `tps_I` $\leftarrow \mu_{\lfloor 261/\theta_I^{\texttt{an}} \rfloor}$.
2. Si `stock_I` ≥ 1 ET `etat_B2` $= 0$ ET `stock_S` $\leq S_{\text{SRM}} - 2$:
   (a) `stock_I` $\rightarrow -1$,

(b) $\texttt{etat\_B2} \leftarrow 1$,

(c) $\texttt{tps\_B2} \leftarrow \mathcal{U}(\{9;9{,}5\})$,

(d) Si $\texttt{tps\_I} = +\infty$, $\texttt{tps\_I} \leftarrow \mu_{\lfloor 261/\theta_I^{\texttt{an}} \rfloor}$.

3. RENVOYER l'état actualisé du processus.

Notons que les conditions de mise en marche sont différentes selon qu'il s'agit du dock B1 ou du dock B2. En effet, dans le cas où il ne reste qu'un emplacement disponible pour les SRM, afin de ne pas dépasser la capacité maximale, nécessairement les deux docks doivent être débloqués et dans ce cas, c'est le dock B1 qui sera bloqué s'il le peut. Il ne faut donc vérifier que celui-là.

Fonction `verifAIT`

1. Si $\texttt{stock\_L} \geq 1$ ET $\texttt{stock\_U} \geq 1$ ET $\texttt{etat\_AIT1} = 0$:

(a) $\texttt{stock\_L} \rightarrow -1$.

(b) Si $\texttt{tps\_L} = +\infty$, $\texttt{tps\_L} \leftarrow \mu_{\lfloor 261/\theta_L^{\texttt{an}} \rfloor}$.

(c) $\texttt{stock\_U} \rightarrow -1$.

(d) Si $\texttt{tps\_I} = +\infty$, $\texttt{tps\_U} \leftarrow \mu_{\lfloor 261/\theta_U^{\texttt{an}} \rfloor}$.

(e) $\texttt{etat\_AIT1} \leftarrow 1$,

(f) $\texttt{tps\_AIT1} \leftarrow \mathcal{U}(\{15;15{,}5;16\})$.

2. Si $\texttt{stock\_L} \geq 1$ ET $\texttt{stock\_U} \geq 1$ ET $\texttt{etat\_AIT2} = 0$:

(a) $\texttt{stock\_L} \rightarrow -1$.

(b) Si $\texttt{tps\_L} = +\infty$, $\texttt{tps\_L} \leftarrow \mu_{\lfloor 261/\theta_L^{\texttt{an}} \rfloor}$.

(c) $\texttt{stock\_U} \rightarrow -1$.

(d) Si $\texttt{tps\_I} = +\infty$, $\texttt{tps\_U} \leftarrow \mu_{\lfloor 261/\theta_U^{\texttt{an}} \rfloor}$.

(e) $\texttt{etat\_AIT2} \leftarrow 1$,

(f) $\texttt{tps\_AIT2} \leftarrow \mathcal{U}(\{15;15{,}5;16\})$.

3. RENVOYER l'état actualisé du processus.

Cette fonction vérifie bien les conditions pour mettre en marche un dock AIT : s'il est libre, et si les sous-assemblages sont en nombre suffisant.

Fonction `verifLP`

On a choisi de changer un petit peu le traitement des variables associées à la zone de lancement, afin de ne plus avoir deux durées distinctes pour un même atelier (`tps_lanc` et `tps_rep`) mais une seule durée, que l'on va appeler `tps_LP` à valeurs dans $[\![0;10{,}5]\!]_2 \cup \{+\infty\}$ qui correspond au minimum des deux durées `tps_lanc` et `tps_rep`. Pour savoir si la zone de lancement est libre, en lancement ou en réparation, on fera appel à un indicateur nommé `etat_LP` et qui est vaut :

— 0 si la LP est libre,

— 1 si la LP est en lancement,

Zone de lancement	Modélisation	Simulation
En lancement	$\texttt{tps\_lanc} \in [\![0;10{,}5]\!]_2$ $\texttt{tps\_rep} = +\infty$	$\texttt{tps\_LP} \in [\![0;10{,}5]\!]_2$ $\texttt{etat\_LP} = 1$
En réparation	$\texttt{tps\_lanc} = +\infty$ $\texttt{tps\_rep} \in [\![0;10{,}5]\!]_2$	$\texttt{tps\_LP} \in [\![0;10{,}5]\!]_2$ $\texttt{etat\_LP} = 2$
Libre	$\texttt{tps\_lanc} = +\infty$ $\texttt{tps\_rep} = +\infty$	$\texttt{tps\_LP} = +\infty$ $\texttt{etat\_LP} = 0$

TABLEAU A.1 – Équivalence entre la modélisation et la simulation

— 2 si la LP est en réparation.

Dès lors, $\texttt{tps\_LP} \neq +\infty$ si `etat_LP` vaut 1 ou 2, et on en conclut que les deux formulations sont donc équivalentes (cf. tableau A.1). On peut maintenant passer au cœur de la fonction `verifLP` :

1. Si (`etat_AIT1` $= 2$ OU `etat_AIT2` $= 2$) ET `stock_S` ≥ 2 ET `etat_LP` $= 0$ ET `attente` ≥ 1 :
   (a) `etat_LP` $\leftarrow 1$,
   (b) `tps_LP` $\leftarrow \mathcal{U}(\{10;10{,}5\})$,
   (c) `stock_S` $\rightarrow -2$,
   (d) `attente` $\rightarrow -1$.
   (e) Si $(\texttt{tps\_ppe} + 10 - \texttt{date\_tir(nb\_tirs + 1)})_+ = 0$:
      i. `cout` $\rightarrow +c_{RP}(\texttt{tps\_ppe} + \texttt{tps\_LP} - \texttt{date\_tir(nb\_tirs + 1)})_+$.
   (f) Sinon
      i. `cout` $\rightarrow +c_{RA}(\texttt{tps\_ppe} + \texttt{tps\_LP} - \texttt{date\_tir(nb\_tirs + 1)})_+$.
   (g) Si `etat_AIT1` $= 2$,
      i. `etat_AIT1` $\leftarrow 0$.
   (h) Sinon,
      i. `etat_AIT2` $\leftarrow 0$.
2. RENVOYER l'état actualisé du processus.

Cette fonction vérifie d'abord les conditions remaniées de déclenchement des opérations LP et si elles sont réalisées, elle initie un lancement. Ensuite, elle regarde quel dock AIT a produit. Si c'est le dock AIT1, elle lui enlève son CC et le remet en position « libre et à l'arrêt ». Sinon, c'est que c'est le dock AIT2 qui a produit et elle lui applique ces mêmes opérations.

Cette fonction requiert que les variables `etat_AIT1` et `etat_AIT2` soient systématiquement égalées à 2 dès que le dock AIT associé a produit. Ainsi, si un lancement peut commencer suite à la production d'un CC, celui-ci est directement pris de son dock et n'apparaîtra pas à l'état suivant. Elle requiert également que le stock de SRM soit incrémenté de 1 après toute production

par un dock B, de sorte que le stock soit immédiatement vidé si la zone de lancement peut être mise en marche consécutivement à cet événement. Enfin, la variable `attente` devra être toujours incrémentée de 1 lorsqu'une autorisation d'envoi est donnée.

Fonction `veriftemps`

1. Si `compteur` $+ 1 \leq N_{\text{tir}}$: `tps_cal` $\leftarrow \max($`date_tir(compteur + 1)` $-$ `tps_ppe` $- 10; 0)$.
2. Sinon : `tps_cal` $\leftarrow +\infty$.
3. RENVOYER l'état actualisé du processus.

A.1.2 Implémentation de la loi de transition

Ici, nous allons présenter comment nous avons implémenté la loi de transition du simulateur, en prenant en compte toutes les considérations de la section 4.2 du chapitre 4.

La transition d'un état à un autre fera appel aux matrices `matrice_etats` et `matrice_actions`, ainsi qu'à une stratégie qu'on va appeler `Strat`. Par définition du calendrier, il ne faut faire qu'un tir la première année et par convention, l'état réduit $(1; 0; 0; 0; 0; 0)$ est l'état numéro 1.

À l'état initial,

— `tps_B1`, `tps_B2`, `tps_AIT1`, `tps_AIT2`, `tps_LP` $\leftarrow +\infty$,

— `tps_cal` $\leftarrow \max($`date_tir(1)` $- 10; 0)$

— `tps_I` $\leftarrow \mu_{\lfloor 261/\theta_I^1 \rfloor}$,

— `tps_L` $\leftarrow \mu_{\lfloor 261/\theta_L^1 \rfloor}$,

— `tps_U` $\leftarrow \mu_{\lfloor 261/\theta_U^1 \rfloor}$,

— `tps_an` $\leftarrow 261$,

— `an` $\leftarrow 1$,

— `tirs_prevus` $\leftarrow 1$,

— `numero_etat` $\leftarrow 1$,

— toutes les autres variables $\leftarrow 0$.

À chaque itération, on regarde quelle valeur parmi `tps_I`, `tps_L`, `tps_U`, `tps_B1`, `tps_B2`, `tps_AIT1`, `tps_AIT2`, `tps_LP`, `tps_cal` et `tps_an` est minimale. On la retranche aux huit autres et on l'ajoute à `tps_ppe`.

1. **Autorisation d'envoi des SRM dans la LP**
   (a) `compteur` $\rightarrow +1$,
   (b) `attente` $\rightarrow +1$,
   (c) on applique `veriftemps`,
   (d) on applique `verifLP`,

(e) on applique `verifAIT`,

(f) on applique `verifB`.

2. **La LP a procédé à un lancement ou a fini d'être réparée.**

— Si `etat_LP` $= 1$:

(a) `nb_tirs` $\rightarrow +1$,

(b) `etat_LP` $\leftarrow 2$,

(c) `tps_LP` $\leftarrow 5$,

— sinon :

(a) `etat_LP` $\leftarrow 0$,

(b) `tps_LP` $\leftarrow +\infty$,

(c) on applique `verifLP`,

(d) on applique `verifAIT`,

(e) on applique `verifB`.

3. **Le dock Bj a produit.** ($j \in \{1; 2\}$)

(a) `stock_S` $\rightarrow +1$,

(b) `etat_Bj` $\leftarrow 0$,

(c) `tps_Bj` $\leftarrow +\infty$,

(d) on applique `verifLP`,

(e) on applique `verifAIT`,

(f) on applique `verifB`.

4. **Le dock AITj a produit.** ($j \in \{1; 2\}$)

(a) `etat_AITj` $\leftarrow 2$,

(b) `tps_AITj` $\leftarrow +\infty$,

(c) on applique `verifLP`,

(d) on applique `verifAIT`,

(e) on applique `verifB`.

5. **Un IMC arrive.**

(a) `stock_I` $\rightarrow +1$.

(b) Si `stock_I` $= 4$, `tps_I` $\leftarrow +\infty$.

(c) Sinon `tps_I` $\leftarrow \mu_{\lfloor 261/\texttt{matrice\_actions(1,Strat(numero\_etat,an))}\rfloor}$.

(d) On applique `verifB`.

6. **Un LLPM arrive.**
   (a) `stock_L` $\rightarrow +1$,
   (b) Si `stock_L` $= 4$, `tps_L` $\leftarrow +\infty$.
   (c) Sinon `tps_L` $\leftarrow \mu_{\lfloor 261/\texttt{matrice\_actions(2,Strat(numero\_etat,an))}\rfloor}$.
   (d) on applique `verifAIT`.

7. **Un ULPM arrive.**
   (a) `stock_U` $\rightarrow +1$,
   (b) Si `stock_U` $= 4$, `tps_U` $\leftarrow +\infty$.
   (c) Sinon `tps_U` $\leftarrow \mu_{\lfloor 261/\texttt{matrice\_actions(3,Strat(numero\_etat,an))}\rfloor}$.
   (d) on applique `verifAIT`.

8. **Une année s'est écoulée**
   (a) `an` $\rightarrow +1$,
   (b) `tps_an` $\leftarrow 261$.
   (c) SI `an` > 30 :
      i. `tirs_prevus` $\leftarrow (\texttt{compteur} - \texttt{nb\_tirs})_+$,
      ii. `numero_etat` $\leftarrow 0$.
   (d) SINON
      i. `tirs_prevus` $\leftarrow \min(\texttt{tirs\_calendrier(an)} + (\texttt{compteur} - \texttt{nb\_tirs})_+; 17)$,
      ii. `numero_etat` $\leftarrow$ le numéro de l'état réduit
      $(\texttt{tirs\_prevus}; \texttt{stockI}; \texttt{stockU}; \texttt{stockL}; \texttt{stockS}; \texttt{stockC})$.

A.2 Algorithmes d'optimisation

Dans cette section, nous allons nous intéresser aux codes des algorithmes MRAS et ASA décrits dans le chapitre 5.

A.2.1 Algorithme MRAS

Variables

Pour programmer cet algorithme, nous aurons besoin des variables suivantes.

— `rho` $\in]0; 1]$: variable qui va accueillir les valeurs successives des ρ_k,

— `N` $\in [\![2; +\infty[\![$: variable qui va prendre les valeurs successives des N_k,

— `N_suiv` $\in [\![2; +\infty[\![$: variable tampon qui va être utilisée pour garder en mémoire la valeur de N_{k+1} lorsqu'elle sera différente de N_k,

— `M` $\in \mathbb{N}^*$: variable qui va prendre les valeurs successives des M_k,

— `P` $\in \mathcal{M}_{\#\mathbb{X},\#\mathbb{A},H}([0;1])$: matrice qui va accueillir les valeurs successives des P_k,

— `Q` $\in \mathcal{M}_{\#\mathbb{X},\#\mathbb{A},H}([0;1])$: matrice auxiliaire qui va accueillir les valeurs successives des $\hat{P}_k$,

— `seuil` $\in \mathbb{R}_+$: variable qui va accueillir les valeurs successives des $\gamma_k(\rho_k, N_k)$,

— `gamma_prec` $\in \mathbb{R}_+$: variable tampon qui va servir à garder en mémoire la valeur de $\overline{\gamma}_{k-1}$,

— `gamma` $\in \mathbb{R}_+$: variable qui va accueillir les valeurs successives des $\overline{\gamma}_k$,

— `k` $\in \mathbb{N}^*$: variable qui va compter les itérations,

— `strategie` $\in \mathcal{M}_{\#\mathbb{X},H}([\![1; \#\mathbb{A}]\!])$: une matrice qui va accueillir les valeurs successives des $\boldsymbol{\pi}_k^*$,

— `memoire_cout` $\in \mathbb{R}_+$: variable tampon qui va servir à garder en mémoire le coût engendré par les $\boldsymbol{\pi}_k^*$.

Nous prenons pour la fonction $\mathcal{H}$ la fonction $x \mapsto \mathrm{e}^{-\mu x}$ avec $\mu > 0$ fixé.

À l'état initial,

— `rho` $\leftarrow \rho_0$,

— `N` $\leftarrow N_0$,

— `N_suiv` $\leftarrow {N_0}^1$,

— `M` $\leftarrow M_0$,

— `P` $\leftarrow P_0$,

— `Q` $\leftarrow {P_0}^1$,

— `seuil` $\leftarrow 0^1$,

— `gamma_prec` $\leftarrow 1^1$,

— `gamma` $\leftarrow 1^1$,

— `k` $\leftarrow 1$,

— `strategie` $\leftarrow (0)^1$,

— `memoire_cout` $\leftarrow 0^1$.

Routines

Pour la programmation de l'algorithme, on fait appel à des fonctions extérieures qui vont calculer certaines quantités. Elles sont au nombre de trois.

La fonction `simulation` : cette fonction est au cœur de l'algorithme puisqu'elle simule une trajectoire du MDP étudié et en rend son coût. Son fonctionnement est détaillé au chapitre 4, section 4.2.

1. Valeur arbitraire.

La fonction `f` : cette fonction calcule les quantités $\ln f(\pi, P)$. Elle prend en entrée une matrice `S` appartenant à $\mathcal{M}_{\#\mathbb{X},H}([\![1;\#\mathbb{A}]\!])$ et une matrice `P` appartenant à $\mathcal{M}_{\#\mathbb{X},\#\mathbb{A},H}([0;1])$. Elle rend en sortie un nombre réel négatif.

1. Créer une matrice `Indic` appartenant à $\mathcal{M}_{\#\mathbb{X},\#\mathbb{A},H}(\{0;1\})$ initialisée à la matrice nulle.
2. POUR t entre 1 et H FAIRE
   (a) Créer une matrice creuse $\mathtt{A} \in \mathcal{M}_{\#\mathbb{X},\#\mathbb{A}}(\{0;1\})$ telle que $\mathtt{A}(i,j) = \delta_{a_j}(\mathtt{S}(i,t))$.
   (b) Affecter à chaque cœfficient $\mathtt{Indic}(i,j,t)$ la valeur $\mathtt{A}(i,j)$.

   FIN POUR
3. Rendre la somme de tous les cœfficients de la matrice de terme général
$$\Big(\log[\mathtt{P}(i,j,t)^{\mathtt{Indic}(i,j,t)}]\Big)_{i,j,t}.$$

La fonction `I` : cette fonction calcule les valeurs de la fonction $\mathcal{I}$ que l'on trouve dans la formule de $\hat{P}_k$. Elle prend en entrée trois nombres réels `z`, `k` et `eps` et rend en sortie un nombre réel entre 0 et 1.

1. SI $\mathtt{z} \leq \mathtt{k}$, rendre 1.
2. SINON SI $\mathtt{z} > \mathtt{k}$ ET $\mathtt{z} < \mathtt{k} + \mathtt{eps}$, rendre $\dfrac{\mathtt{k} + \mathtt{eps} - \mathtt{z}}{\mathtt{eps}}$.
3. SINON rendre 0.

Instructions

Le MRAS est donc ainsi programmé.

TANT QUE $\mathtt{k} \leq K$

1. **Simulation des stratégies candidates**
   (a) Créer une matrice $\mathtt{candidats} \in \mathcal{M}_{\#\mathbb{X},\#\mathbb{A},\mathtt{N}}([\![1;\#\mathbb{A}]\!])$ initialisée à la matrice nulle.
   (b) POUR n entre 1 et `N` FAIRE
      i. Construire une matrice $\mathtt{u} \in \mathcal{M}_{\#\mathbb{A},\#\mathbb{X},H}([0;1])$ en répliquant un vecteur-ligne de réalisations de la loi uniforme sur [0 ;1] sur $\#\mathbb{A}$ lignes.
      ii. Tirer un réel `v` selon une loi uniforme sur $[0;1]$.
      iii. SI $\mathtt{v} \leq \lambda$:
         A. Calculer la fonction de répartition de la matrice P_0.
         B. Calculer H vecteurs d'actions tirées à l'aide de la fonction de répartition de P_0 et de `u`.
         C. Concaténer ces vecteurs pour les affecter à $\mathtt{candidats}(:,:,n)$.
      iv. SINON

A. Calculer la fonction de répartition de la matrice `P`.

B. Calculer H vecteurs d'actions tirées à l'aide de la fonction de répartition de `P` et de `u`.

C. Concaténer ces vecteurs pour les affecter à $\texttt{candidats}(:,:,n)$.

FIN POUR

2. **Évaluation des performances**

(a) Créer une matrice $\texttt{couts} \in \mathcal{M}_{2,\texttt{N}}(\mathbb{R})$ initialisée à la matrice nulle.

(b) POUR n entre 1 et `N` FAIRE

i. Simuler `M` trajectoires avec `simulation` en utilisant la stratégie $\texttt{candidats}(:,:,n)$ et calculer leur coût.

ii. Affecter la moyenne des `M` coûts à $\texttt{couts}(1,n)$.

iii. $\texttt{couts}(2,n) \leftarrow n$.

FIN POUR

3. **Détermination des meilleures stratégies candidates**

(a) Créer une matrice $\texttt{couts\_trie} \in \mathcal{M}_{2,\texttt{N}}(\mathbb{R})$ qui contient la matrice `couts` dont les colonnes auront été triées dans l'ordre décroissant selon la première ligne.

(b) Créer une variable `seuil` qui contiendra $\texttt{couts\_trie}(1, \lceil \texttt{N}(1-\texttt{rho}) \rceil)$.

(c) $\texttt{gamma\_prec} \leftarrow \texttt{gamma}$.

(d) SI $\texttt{k} = 1$:

i. $\texttt{gamma} \leftarrow \texttt{seuil}$,

ii. $\texttt{strategie} \leftarrow \texttt{candidats}(:,:,\texttt{couts\_trie}(2, \lceil \texttt{N}(1-\texttt{rho}) \rceil))$,

iii. $\texttt{memoire\_cout} \leftarrow \texttt{seuil}$,

iv. $\texttt{N\_suiv} \leftarrow \texttt{N}$.

(e) SINON SI $\texttt{seuil} \leq \texttt{gamma} - \texttt{eps}$:

i. $\texttt{gamma} \leftarrow \texttt{seuil}$,

ii. $\texttt{strategie} \leftarrow \texttt{candidats}(:,:,\texttt{couts\_trie}(2, \lceil \texttt{N}(1-\texttt{rho}) \rceil))$,

iii. $\texttt{memoire\_cout} \leftarrow \texttt{seuil}$,

iv. $\texttt{N\_suiv} \leftarrow \texttt{N}$.

(f) SINON :

i. créer une variable `existe` initialisée à 0,

ii. créer une variable `num` initialisée à $\lceil \texttt{N}(1-\texttt{rho}) \rceil + 1$,

iii. TANT QUE $\texttt{num} \leq \texttt{N}$ ET $\texttt{existe} = 0$ FAIRE

A. SI $\texttt{couts\_trie}(1,\texttt{num}) \leq \texttt{gamma\_prec} - \texttt{eps}$, affecter 1 à `existe`,

B. SINON $\texttt{num} \leftarrow \texttt{num} + 1$,

FIN TANT QUE

iv. SI $\texttt{existe} = 1$:

A. $\texttt{gamma} \leftarrow \texttt{couts\_trie}(1, \texttt{num})$,

B. $\texttt{strategie} \leftarrow \texttt{candidats}(:, :, \texttt{couts\_trie}(2, \texttt{num}))$,

C. $\texttt{memoire\_cout} \leftarrow \texttt{couts\_trie}(1, \texttt{num})$,

D. $\rho \leftarrow 1 - \dfrac{\texttt{num}}{\texttt{N}}$,

E. $\texttt{N\_suiv} \leftarrow \texttt{N}$,

v. SINON :

A. $\texttt{gamma} \leftarrow \texttt{memoire\_cout}$,

B. $\texttt{N\_suiv} \leftarrow \lceil \alpha \texttt{N} \rceil$.

4. **Calcul de la matrice P_{k+1}**

(a) Créer un vecteur `logsA` de `N` composantes tel que chaque valeur $\texttt{logsA}(j)$ continenne la quantité $-\mu(\texttt{k}-1)\texttt{couts}(1,j) + \log(\texttt{I}(\texttt{couts}(1,j), \texttt{gamma}, \varepsilon))$.

(b) Créer un vecteur `F0` à `N` composantes tel que chaque valeur $\texttt{F0}(j)$ contienne $\texttt{f}(\texttt{candidats}(:,:,j), P_0) + \log\lambda$.

(c) Créer un vecteur `Fk` à `N` composantes tel que chaque valeur $\texttt{Fk}(j)$ contienne $\texttt{f}(\texttt{candidats}(:,:,j), \texttt{P}) + \log(1-\lambda)$.

(d) SI $\max(\texttt{logsA}) > -\infty$:

i. créer une matrice $\texttt{coeffs} \in \mathcal{M}_{2,\texttt{N}}(\mathbb{R})$ initialisée à la matrice nulle,

ii. créer une matrice $\texttt{D} \in \mathcal{M}_{\#\mathbb{X},\#\mathbb{A},\texttt{N}}(\mathbb{R})$ initialisée à la matrice nulle,

iii. POUR j entre 1 et `N` FAIRE

A. SI $\texttt{F0}(j) > \texttt{Fk}(j)$, $\texttt{coeffs}(1,j) \leftarrow \dfrac{1}{1+\exp(\texttt{Fk}(j) - \texttt{F0}(j))}$,

B. SINON $\texttt{coeffs}(1,j) \leftarrow \dfrac{1}{1+\exp(\texttt{F0}(j) - \texttt{Fk}(j))}$,

FIN POUR

iv. POUR j entre 1 et `N` FAIRE $\texttt{coeffs}(2,j) \leftarrow \texttt{logsA}(j) - \max(\texttt{F0}(j), \texttt{Fk}(j))$ FIN POUR,

v. créer deux variables `vmax` qui contient la quantité $\max_{j \in [\![1;\texttt{N}]\!]} (\texttt{coeffs}(2,j))$ et `pmax` qui contient la quantité $\operatorname{argmax}_{j \in [\![1;\texttt{N}]\!]} (\texttt{coeffs}(2,j))$,

vi. diviser toutes les valeurs de la première ligne de `coeffs` par $\texttt{coeffs}(1, \texttt{pmax})$,

vii. ôter à toutes les valeurs de la deuxième ligne de `coeffs` la quantité `vmax`,

viii. POUR t entre 1 et H FAIRE

A. POUR n entre 1 et `N` FAIRE

— créer une matrice creuse de taille $\#\mathbb{X} \times \#\mathbb{A}$ de terme général $\left(\delta_{a_j}(\texttt{candidats}(i,t,n))\right)_{i,j}$,

— affecter cette matrice creuse à $\texttt{D}(:,:,n)$,

FIN POUR

B. calculer la matrice auxiliaire D' de taille $\#\mathbb{X} \times \#\mathbb{A}$ telle que $D'(i,j) = \sum_{n=1}^{\texttt{N}} \texttt{D}(i,j,n)$,

C. calculer le nombre $s = \sum_{n=1}^{\texttt{N}} \texttt{coeffs}(1,n)\exp(\texttt{coeffs}(2,n))$,

D. $\texttt{Q}(:,:,t) \leftarrow \frac{1}{s}D'$,

FIN POUR

ix. $\texttt{P} \leftarrow (1-\nu)\texttt{P} + \nu\texttt{Q}$.

(e) $\texttt{k} \leftarrow \texttt{k} + 1$.

(f) $\texttt{M} \leftarrow \lceil \beta \texttt{M} \rceil$.

(g) $\texttt{N} \leftarrow \texttt{N\_suiv}$.

FIN TANT QUE

A.2.2 Algorithme ASA

Variables

Pour programmer cet algorithme, nous aurons besoin des variables suivantes.

— $\texttt{N} \in [\![2; +\infty[\![$: variable qui va prendre les valeurs successives des N_k,

— $\texttt{M} \in \mathbb{N}^*$: variable qui va prendre les valeurs successives des M_k,

— $\texttt{P} \in \mathcal{M}_{\#\mathbb{X},\#\mathbb{A},H}([0;1])$: matrice qui va accueillir les valeurs successives des P_k,

— $\texttt{Q} \in \mathcal{M}_{\#\mathbb{X},\#\mathbb{A},H}([0;1])$: matrice auxiliaire qui va accueillir les valeurs successives des $\hat{P}_k$,

— $\texttt{alpha} \in [0;1]$: variable qui va accueillir les valeurs successives des α_k,

— $\texttt{beta} \in [0;1]$: variable qui va accueillir les valeurs successives des β_k,

— $\texttt{T} \in \mathbb{R}_+$: variable qui va accueillir les valeurs successives des T_k,

— $\texttt{k} \in \mathbb{N}^*$: variable qui va compter les itérations.

À l'état initial,

— $\texttt{N} \leftarrow N_0$,

— $\texttt{M} \leftarrow M_0$,

— $\texttt{P} \leftarrow P_0$,

— $\texttt{Q} \leftarrow P_0$ (valeur arbitraire),

— `alpha` $\leftarrow \alpha_0$,

— `beta` $\leftarrow \beta_0$,

— `T` $\leftarrow T_0$,

— `k` $\leftarrow 1$.

Routines

Pour la programmation de l'algorithme ASA, on fait appel à des fonctions extérieures qui vont calculer certaines quantités. Ces fonctions sont au nombre de deux.

La fonction `simulation` : cette fonction simule des trajectoires du MDP étudié et en rend une estimation du coût moyen. Son fonctionnement est détaillé dans le chapitre 4, partie 4.2.

La fonction `f` : cette fonction calcule les quantités $\ln f(\pi, P)$. Elle prend en entrée une matrice `S` de $\mathcal{M}_{\#\mathbb{X},H}(\llbracket 1;\#\mathbb{A}\rrbracket)$ et une matrice `P` appartenant à $\mathcal{M}_{\#\mathbb{X},\#\mathbb{A},H}([0;1])$. Elle rend en sortie un nombre réel négatif.

1. Créer une matrice `Indic` appartenant à $\mathcal{M}_{\#\mathbb{X},\#\mathbb{A},H}(\{0;1\})$ initialisée à la matrice nulle.
2. POUR t entre 1 et H FAIRE
   (a) Créer une matrice creuse $\mathtt{A} \in \mathcal{M}_{\#\mathbb{X},\#\mathbb{A}}(\{0;1\})$ telle que $\mathtt{A}(i,j) = \delta_{a_j}(\mathtt{S}(i,t))$.
   (b) Affecter à chaque cœfficient $\mathtt{Indic}(i,j,t)$ la valeur $\mathtt{A}(i,j)$.

   FIN POUR
3. Rendre la somme de tous les cœfficients de la matrice de terme général $\left(\log[\mathtt{P}(i,j,t)^{\mathtt{Indic}(i,j,t)}]\right)_{i,j,t}$.

Instructions

Le corps de l'ASA est donc ainsi programmé.

TANT QUE `k` $\leq K$

1. **Simulation des stratégies candidates**
   (a) Créer une matrice `candidats` $\in \mathcal{M}_{\#\mathbb{X},\#\mathbb{A},\mathtt{N}}(\llbracket 1;\#\mathbb{A}\rrbracket)$ initialisée à la matrice nulle.
   (b) POUR n entre 1 et `N` FAIRE
      i. Construire une matrice $\mathtt{u} \in \mathcal{M}_{\#\mathbb{A},\#\mathbb{X},H}([0;1])$ en répliquant un vecteur-ligne de réalisations de la loi uniforme sur [0 ;1] sur $\#\mathbb{A}$ lignes.
      ii. Tirer un réel `v` selon une loi uniforme sur $[0;1]$.
      iii. SI `v` $\leq$ `beta` :
         A. Calculer la fonction de répartition de la matrice P_0.
         B. Calculer H vecteurs d'actions tirées à l'aide de la fonction de répartition de P_0 et de `u`.

C. Concaténer ces vecteurs pour les affecter à $\texttt{candidats}(:,:,n)$.

iv. SINON

A. Calculer la fonction de répartition de la matrice `P`.

B. Calculer H vecteurs d'actions tirées à l'aide de la fonction de répartition de `P` et de `u`.

C. Concaténer ces vecteurs pour les affecter à $\texttt{candidats}(:,:,n)$.

FIN POUR

2. **Évaluation des performances**

(a) Créer un vecteur $\texttt{couts} \in \mathcal{M}_{1,\texttt{N}}(\mathbb{R})$ initialisée à la matrice nulle.

(b) POUR n entre 1 et `N` FAIRE

i. Affecter à $\texttt{couts}(1,n)$ la valeur rendue par `simulation`, avec un nombre de réplications de Monte-Carlo égal à `M`.

FIN POUR

3. **Calcul de la matrice P_{k+1}**

(a) Créer un vecteur `logsA` de `N` composantes tel que chaque valeur $\texttt{logsA}(j)$ continenne la quantité $-\dfrac{\texttt{couts}(1,j)}{\texttt{T}}$.

(b) créer un vecteur `F0` à `N` composantes tel que chaque $\texttt{F0}(j)$ vale $\texttt{f}(\texttt{candidats}(:,:,j), P_0) + \log(\texttt{beta})$,

(c) créer un vecteur `Fk` à `N` composantes tel que chaque $\texttt{Fk}(j)$ vale $\texttt{f}(\texttt{candidats}(:,:,j), \texttt{P}) + \log(1 - \texttt{beta})$,

(d) créer une matrice $\texttt{coeffs} \in \mathcal{M}_{2,\texttt{N}}(\mathbb{R})$ initialisée à la matrice nulle,

(e) créer une matrice $\texttt{D} \in \mathcal{M}_{\#\mathbb{X},\#\mathbb{A},\texttt{N}}(\mathbb{R})$ initialisée à la matrice nulle,

(f) POUR j entre 1 et `N` FAIRE

i. SI $\texttt{F0}(j) > \texttt{Fk}(j)$, $\texttt{coeffs}(1,j) \leftarrow \dfrac{1}{1 + \exp(\texttt{Fk}(j) - \texttt{F0}(j))}$,

ii. SINON $\texttt{coeffs}(1,j) \leftarrow \dfrac{1}{1 + \exp(\texttt{F0}(j) - \texttt{Fk}(j))}$,

iii. $\texttt{coeffs}(2,j) \leftarrow \texttt{logsA}(j) - \max(\texttt{F0}(j), \texttt{Fk}(j))$,

FIN POUR

(g) créer deux variables `vmax` qui contient la quantité $\max\limits_{j \in [\![1;\texttt{N}]\!]} (\texttt{coeffs}(2,j))$ et `pmax` qui contient la quantité $\operatorname{argmax}\limits_{j \in [\![1;\texttt{N}]\!]} (\texttt{coeffs}(2,j))$,

(h) diviser toutes les valeurs de la première ligne de `coeffs` par $\texttt{coeffs}(1, \texttt{pmax})$,

(i) ôter à toutes les valeurs de la deuxième ligne de `coeffs` la quantité `vmax`,

(j) POUR t entre 1 et H FAIRE

i. POUR n entre 1 et $\mathtt{N}$ FAIRE

— créer une matrice creuse de taille $\#\mathbb{X} \times \#\mathbb{A}$ de terme général $\big(\delta_{a_j}(\mathtt{candidats}(i,t,n))\big)_{i,j}$,

— affecter cette matrice creuse à $\mathtt{D}(:,:,n)$,

FIN POUR

ii. calculer la matrice auxiliaire D' de taille $\#\mathbb{X} \times \#\mathbb{A}$ telle que $D'(i,j) = \sum_{n=1}^{\mathtt{N}} \mathtt{D}(i,j,n)$,

iii. calculer le nombre $s = \sum_{n=1}^{\mathtt{N}} \mathtt{coeffs}(1,n)\exp(\mathtt{coeffs}(2,n))$,

iv. $\mathtt{Q}(:,:,t) \leftarrow \frac{1}{s} D'$,

FIN POUR

(k) $\mathtt{P} \leftarrow (1-\mathtt{alpha})\mathtt{P} + \mathtt{alpha}.\mathtt{Q}$.

4. **Calcul des nouveaux paramètres de l'algorithme**

(a) $\mathtt{M} \leftarrow \max(M_0, \lfloor 1{,}01 \log^3(\mathtt{k}) \rfloor)$,

(b) $\mathtt{N} \leftarrow \max(N_0, \lfloor \mathtt{k}^{0{,}501} \rfloor)$,

(c) $\mathtt{alpha} \leftarrow \frac{1}{(\mathtt{k}+100)^{0{,}501}}$,

(d) $\mathtt{beta} \leftarrow \frac{1}{\sqrt{\mathtt{k}+1}}$,

(e) $\mathtt{T} \leftarrow \frac{T_0}{\log(\mathtt{k}+\mathrm{e})}$,

(f) $\mathtt{k} \leftarrow \mathtt{k}+1$.

FIN TANT QUE

B Démonstrations complémentaires

Dans cette annexe, nous proposons de détailler la démonstration de certains résultats du chapitre 8.

B.1 Démonstration du théorème 8.1.3.1

Soit $\lambda = (\Xi, \mathcal{F}, \mathbf{P}, (\mathcal{F}_k)_{0\leq k\leq N_0}, (\mathcal{X}_k, \mathcal{Y}_k)_{0\leq k\leq N_0}, \tau)$ un contrôle de Λ. Notre objectif sera de construire un POMDP et une stratégie à partir du contrôle donné de sorte à avoir l'égalité voulue. Pour ce faire, sur l'espace probabilisé $(\Xi, \mathcal{F}, \mathbf{P})$, on définit les processus $(\mathcal{A}_k)_{0\leq k\leq N_0}$ et $(\mathcal{Z}_k)_{0\leq k\leq N_0}$ par

$$\mathcal{A}_k = \mathbb{1}_{\{\tau\leq k\}} \text{ et } \begin{cases} \mathcal{Z}_k = \mathcal{A}_{k-1} \\ \mathcal{Z}_0 = 0. \end{cases}$$

Le processus $(\mathcal{A}_k)$ indiquera si, selon le temps d'arrêt τ, on décide de s'arrêter ou non et $(\mathcal{Z}_k)$ informe du fait que l'arrêt a été effectué ou non.

Puisque τ est un temps d'arrêt adapté à la filtration $(\mathcal{F}_k^{\mathbb{Y}})$, $(\mathcal{A}_k)$ est aussi $(\mathcal{F}_k^{\mathbb{Y}})$-adapté et $(\mathcal{Z}_k)$ est $(\mathcal{F}_k^{\mathbb{Y}})$-prévisible. Alors, pour tout k, il existe une fonction mesurable $f_k : \mathbb{Y}^{k+1} \to \{0; 1\}$ telle que

$$\mathcal{A}_k = f_k(\mathcal{Y}_0, \ldots, \mathcal{Y}_k). \tag{B.1}$$

Comme le processus $(\mathcal{X}_k, \mathcal{Y}_k)$ est $(\mathcal{F}_k)$-adapté, alors $\mathcal{F}_k^{\mathbb{Y}} \subset \mathcal{F}_k$. Donc $(\mathcal{A}_k)$ est aussi $(\mathcal{F}_k)$-adapté et $(\mathcal{Z}_k)$ est aussi $(\mathcal{F}_k)$-prévisible. On définit maintenant la filtration

$$\mathcal{T}_k = \sigma\left(\mathcal{X}_l, \mathcal{Y}_l, \mathcal{Z}_l,\, l \in [\![0; k]\!]\right) \forall\, k \in [\![0; N_0]\!].$$

Il est clair que $\mathcal{F}_k^{\mathbb{Y}} \subset \mathcal{T}_k \subset \mathcal{F}_k$.

Pour tous $B \in \mathcal{B}(\mathbb{X})$, $C \in \mathcal{B}(\mathbb{Y})$ et $D \subset \{0; 1\}$, on a que

$$\mathbf{P}\left\{(\mathcal{X}_{k+1}, \mathcal{Y}_{k+1}, \mathcal{Z}_{k+1}) \in B \times C \times D \mid \mathcal{T}_k\right\} = \mathbf{E}\left[\mathbb{1}_{\{(\mathcal{X}_{k+1}, \mathcal{Y}_{k+1}, \mathcal{Z}_{k+1}) \in B \times C \times D\}} \mid \mathcal{T}_k\right]. \tag{B.2}$$

Or, on peut écrire que

$$\begin{aligned}\mathbb{1}_{\{(\mathcal{X}_{k+1}, \mathcal{Y}_{k+1}, \mathcal{Z}_{k+1}) \in B \times C \times D\}} &= \mathbb{1}_{\{\mathcal{A}_k = 0\}} \mathbb{1}_{\{\mathcal{Z}_{k+1} \in D\}} \mathbb{1}_{\{(\mathcal{X}_{k+1}, \mathcal{Y}_{k+1}) \in B \times C\}} \\ &\quad + \mathbb{1}_{\{\mathcal{A}_k = 1\}} \mathbb{1}_{\{\mathcal{Z}_{k+1} \in D\}} \mathbb{1}_{\{(\mathcal{X}_{k+1}, \mathcal{Y}_{k+1}) \in B \times C\}}.\end{aligned} \tag{B.3}$$

Comme $\mathcal{T}_k \subset \mathcal{F}_k$, $\mathcal{A}_k$ étant $\mathcal{T}_k$-mesurable et $\mathcal{Z}_k$ étant $(\mathcal{F}_k^{\mathbb{Y}})$-prévisible, alors

$$\begin{aligned}&\mathbf{E}\left[\mathbb{1}_{\{\mathcal{A}_k = 0\}} \mathbb{1}_{\{\mathcal{Z}_{k+1} \in D\}} \mathbb{1}_{\{(\mathcal{X}_{k+1}, \mathcal{Y}_{k+1}) \in B \times C\}} \mid \mathcal{T}_k\right] \\ &= \mathbb{1}_{\{\mathcal{A}_k = 0\}} \mathbb{1}_{\{\mathcal{Z}_{k+1} \in D\}} \mathbf{E}\left[\mathbf{E}\left[\mathbb{1}_{\{(\mathcal{X}_{k+1}, \mathcal{Y}_{k+1}) \in B \times C\}} \mid \mathcal{F}_k\right] \mid \mathcal{T}_k\right] \\ &= \mathbb{1}_{\{\mathcal{A}_k = 0\}} \mathbb{1}_{\{\mathcal{Z}_{k+1} \in D\}} \mathbf{E}\left[P(B \times C \mid \mathcal{X}_k, \mathcal{Y}_k) \mid \mathcal{T}_k\right] \\ &= \mathbb{1}_{\{\mathcal{A}_k = 0\}} \mathbb{1}_{\{\mathcal{Z}_{k+1} \in D\}} P(B \times C \mid \mathcal{X}_k, \mathcal{Y}_k).\end{aligned} \tag{B.4}$$

Or $\{\mathcal{A}_k = 0\} = \{\mathcal{Z}_{k+1} = 0\}$. Donc

$$\mathbf{E}\left[\mathbb{1}_{\{\mathcal{A}_k = 0\}} \mathbb{1}_{\{\mathcal{Z}_{k+1} \in D\}} \mathbb{1}_{\{(\mathcal{X}_{k+1}, \mathcal{Y}_{k+1}) \in B \times C\}} \mid \mathcal{T}_k\right] = \mathbb{1}_{\{\mathcal{A}_k = 0\}} \delta_0(D) P(B \times C \mid \mathcal{X}_k, \mathcal{Y}_k). \tag{B.5}$$

De la même manière, en utilisant l'égalité $\{\mathcal{A}_k = 1\} = \{\mathcal{Z}_{k+1} = 1\}$, on obtient que

$$\mathbf{E}\left[\mathbb{1}_{\{\mathcal{A}_k = 1\}} \mathbb{1}_{\{\mathcal{Z}_{k+1} \in D\}} \mathbb{1}_{\{(\mathcal{X}_{k+1}, \mathcal{Y}_{k+1}) \in B \times C\}} \mid \mathcal{T}_k\right] = \mathbb{1}_{\{\mathcal{A}_k = 1\}} \delta_1(D) P(B \times C \mid \mathcal{X}_k, \mathcal{Y}_k). \tag{B.6}$$

Notons que $\{\mathcal{A}_k = 0\} \subset \{\mathcal{A}_{k-1} = 0\} = \{\mathcal{Z}_k = 0\}$. Par conséquent,

$$\begin{aligned}&\mathbf{P}\left\{(\mathcal{X}_{k+1}, \mathcal{Y}_{k+1}, \mathcal{Z}_{k+1}) \in B \times C \times D \mid \mathcal{T}_k\right\} \\ &= \mathbb{1}_{\{\mathcal{A}_k = 0\}} \delta_{\mathcal{Z}_k}(D) P(B \times C \mid \mathcal{X}_k, \mathcal{Y}_k) + \mathbb{1}_{\{\mathcal{A}_k = 1\}} \delta_1(D) P(B \times C \mid \mathcal{X}_k, \mathcal{Y}_k) \\ &= Q(B \times C \times D \mid \mathcal{X}_k, \mathcal{Y}_k, \mathcal{Z}_k, \mathcal{A}_k).\end{aligned} \tag{B.7}$$

Enfin, la loi de $\mathcal{A}_k$ possède la propriété suivante : pour $D \subset \{0; 1\}$,

$$\mathbf{P}\left\{\mathcal{A}_k \in D \mid (\mathcal{Y}_0, \mathcal{Z}_0, \mathcal{A}_0, \ldots, \mathcal{Y}_k, \mathcal{Z}_k) = (y_0, z_0, a_0, \ldots, y_k, z_k)\right\} = \delta_{f_k(y_0, \ldots, y_k)}(D). \tag{B.8}$$

On introduit la suite $\pi = (g_k)_{0 \leq k \leq N_0 - 1}$ où chaque g_k est une règle de décision H^{o}-dépendante au rang k définie par

$$g_k(y_0, z_0, a_0, \ldots, y_{k-1}, z_{k-1}, a_{k-1}, y_k, z_k) = f_k(y_0, \ldots, y_k). \tag{B.9}$$

Dès lors, la propriété d'unicité du théorème de Ionescu-Tulcea, conjointement avec les équations (B.7) et (B.8), nous permet d'avoir que, pour tout rectangle mesurable $G = G_0 \times G_1 \times \ldots \times G_{N_0}$,

$$\mathbf{P}\left\{(\mathcal{X}_0, \mathcal{Y}_0, \mathcal{Z}_0, \mathcal{A}_0, \ldots, \mathcal{X}_{N_0}, \mathcal{Y}_{N_0}, \mathcal{Z}_{N_0}) \in G\right\} = \mathbb{P}^{\pi}_{x,y}\left\{(X_0, Y_0, Z_0, A_0, \ldots, X_{N_0}, Y_{N_0}, Z_{N_0}) \in G\right\}. \tag{B.10}$$

On a alors que

$$\begin{aligned}\mathcal{R}(x, y, \lambda) &= \mathbf{E}[R(\mathcal{X}_\tau, \mathcal{Y}_\tau)] \\ &= \mathbf{E}\left[\sum_{k=0}^{N_0-1} R(\mathcal{X}_k, \mathcal{Y}_k)\mathbb{1}_{\{\tau=k\}} + R(\mathcal{X}_{N_0}, \mathcal{Y}_{N_0})\mathbb{1}_{\{\tau=N_0\}}\right].\end{aligned} \tag{B.11}$$

On distingue maintenant deux cas.

— si $\tau = N_0$, alors $\mathcal{A}_k = 0$ pour tout $k \in [\![0; N_0 - 1]\!]$ et a fortiori $\mathcal{Z}_{N_0} = 0$. Par la définition de la fonction h, nous avons l'égalité $R(\mathcal{X}_{N_0}, \mathcal{Y}_{N_0})\mathbb{1}_{\{\tau=N_0\}} = h(\mathcal{X}_{N_0}, \mathcal{Y}_{N_0}, \mathcal{Z}_{N_0})$,

— si $\tau = k < N_0$, alors $\mathcal{A}_l = \mathbb{1}_{[\![k;N_0-1]\!]}(l)$ et donc $\mathcal{Z}_l = \mathbb{1}_{[\![k+1;N_0]\!]}(l)$. En particulier, $\mathcal{A}_k = 1$ et $\mathcal{Z}_k = 0$. La définition que l'on a choisie pour la fonction r nous permet alors d'écrire que $R(\mathcal{X}_k, \mathcal{Y}_k)\mathbb{1}_{\{\tau=k\}} = r(\mathcal{X}_k, \mathcal{Y}_k, \mathcal{Z}_k, \mathcal{A}_k)$.

Il vient que

$$\begin{aligned}\mathcal{R}(x, y, \alpha) &= \mathbf{E}\left[\sum_{k=0}^{N_0-1} r(\mathcal{X}_k, \mathcal{Y}_k, \mathcal{Z}_k, \mathcal{A}_k) + h(\mathcal{X}_{N_0}, \mathcal{Y}_{N_0}, \mathcal{Z}_{N_0})\right] \\ &= \mathbb{E}^{\pi}_{x,y}\left[\sum_{k=0}^{N_0-1} r(X_k, Y_k, Z_k, A_k) + h(X_{N_0}, Y_{N_0}, Z_{N_0})\right]\end{aligned} \tag{B.12}$$

par l'égalité (B.10). On a donc bien que $\mathcal{R}(x, y, \lambda) = \mathcal{R}_{\mathcal{M}}(x, y, \pi)$.

Réciproquement, soit une stratégie H^{o}-dépendante $\pi = (f_0, \ldots, f_{N_0-1}) \in \Pi^{\mathrm{o}}$ et soit un état initial $(x, y) \in \mathbb{X} \times \mathbb{Y}$. On considère un processus $(X_k, Y_k, Z_k)_{0 \leq k \leq N_0}$ défini sur l'espace de probabilité $\left(\Omega, \mathcal{G}, \mathbb{P}^{\pi}_{x,y}\right)$ par les équations (8.2), et on considère également la suite $(A_k)_{0 \leq k \leq N_0}$ définie par les équations (8.3). On définit les filtrations suivantes : $\mathcal{F}_k = \sigma(X_0, Y_0, Z_0, \ldots, X_k, Y_k, Z_k)$, $\mathcal{F}_k^{\mathbb{Y},Z} = \sigma(Y_0, Z_0, \ldots, Y_k, Z_k)$ et $\mathcal{F}_k^{\mathbb{Y}} = \sigma(Y_0, \ldots, Y_k)$. Le processus (X_k, Y_k) est $(\mathcal{F}_k)$-adapté, à valeurs dans $\mathbb{X} \times \mathbb{Y}$, de loi de transition P et d'état initial $(x, y) \in \mathbb{X} \times \mathbb{Y}$. Soit maintenant la variable aléatoire τ définie par

$$\tau = \begin{cases} \inf\{k \in [\![0; N_0 - 1]\!] \mid A_k = 1\} \text{ si cette quantité existe,} \\ N_0 \text{ sinon.} \end{cases}$$

Par définition, τ est un temps d'arrêt adapté à la filtration $(\mathcal{F}_k^{\mathbb{Y},Z})$. Mais, selon la définition d'un contrôle, il faudrait qu'il soit adapté à $(\mathcal{F}_k^{\mathbb{Y}})$. On démontre donc le lemme suivant.

Lemme B.1.0.1. *Pour tout $k \in [\![0; N_0]\!]$, il existe une fonction mesurable $F_k : \mathbb{Y}^{k+1} \to \{0; 1\}$ telle que $Z_k = F_k(Y_0, \ldots, Y_k)$.*

DÉMONSTRATION. On va démontrer ce résultat par récurrence. Notons d'abord que $Z_0 = 0$ par définition. On écrit alors $Z_0 = F_0(Y_0)$ où F_0 est la fonction identiquement nulle sur $\mathbb{Y}$, qui est donc mesurable.

Soit $k \in [\![1; N_0]\!]$. On suppose la propriété vraie pour tout $l \in [\![0; k-1]\!]$. Montrons qu'elle est alors vraie au rang k. On écrit

$$\begin{aligned} A_{k-1} &= f_{k-1}(Y_0, Z_0, A_0, \dots, Y_{k-1}, Z_{k-1}) \\ &= f_{k-1}(Y_0, F_0(Y_0), A_0, \dots, Y_{k-1}, F_{k-1}(Y_0, \dots, Y_{k-1})) \end{aligned} \tag{B.13}$$

par hypothèse de récurrence. Donc on peut écrire $A_{k-1} = \tilde{f}_{k-1}(Y_0, \dots, Y_{k-1})$ où $\tilde{f}_{k-1}$ est mesurable par composition.

Alors, par définition de Z_k,

$$\begin{aligned} Z_k &= \mathbb{1}_{\{A_{k-1}=1\}} + Z_{k-1}\mathbb{1}_{\{A_{k-1}=0\}} \\ &= \mathbb{1}_{\left\{\tilde{f}_{k-1}(Y_0,\dots,Y_{k-1})=1\right\}} + F_{k-1}(Y_0, \dots, Y_{k-1})\mathbb{1}_{\left\{\tilde{f}_{k-1}(Y_0,\dots,Y_{k-1})=0\right\}}. \end{aligned} \tag{B.14}$$

Donc on peut écrire $Z_k = F_k(Y_0, \dots, Y_k)$ où F_k est mesurable par composition. La récurrence est établie □

Ce lemme prouve notamment que pour tout k, $\mathcal{F}_k^{\mathbb{Y},Z} = \mathcal{F}_k^{\mathbb{Y}}$. Donc τ est adapté à la filtration $(\mathcal{F}_k^{\mathbb{Y}})$. Dès lors, par définition des fonctions r et h, et par définition de τ,

$$\begin{aligned} \mathbb{E}^{\pi}_{x,y}\left[\sum_{k=0}^{N_0-1} r(X_k, Y_k, Z_k, A_k) + h(X_{N_0}, Y_{N_0}, Z_{N_0})\right] &= \mathbb{E}^{\pi}_{x,y}\left[\sum_{k=0}^{N_0} R(X_k, Y_k)\mathbb{1}_{\{\tau=k\}}\right] \\ &= \mathbb{E}^{\pi}_{x,y}[R(X_\tau, Y_\tau)]. \end{aligned} \tag{B.15}$$

Donc, le contrôle $\lambda = \left(\Omega, \mathcal{G}, \mathbb{P}^{\pi}_{x,y}, (\mathcal{F}_k)_{0\le k\le N_0}, (X_k, Y_k)_{0\le k\le N_0}, \tau\right)$ appartient à Λ et vérifie l'égalité $\mathcal{R}_{\mathcal{M}}(x, y, \pi) = \mathcal{R}(x, y, \lambda)$.

B.2 Démonstration du théorème 8.1.5.1

Il suffit de vérifier que notre modèle $\mathcal{M}'$ respecte l'hypothèse de structure du chapitre 2 de [BR11]. Ainsi, on pourra appliquer le théorème 5.3.3 de [BR11] et avoir le résultat au prix d'un changement d'indice.

On note $\mathbb{B}(\mathbb{S}')$ l'ensemble des fonctions mesurables bornées sur $\mathbb{S}'$. Pour f un élément de $\mathbb{B}(\mathbb{S}')$, on définit $\|f\| = \sup_{(\rho,y,z)\in\mathbb{S}'} |f(\rho, y, z)|$.

(i) Il est évident que $h \in \mathbb{B}(\mathbb{S}')$ car $R \in \mathbb{L}(\mathcal{P}(\mathbb{X}) \times \mathbb{Y})$.

(ii) Soit $f \in \mathbb{B}(\mathbb{S}')$. L'intégrale $\displaystyle\int_{\mathbb{S}'} f(\rho', y', z')Q'(\mathrm{d}\rho', \mathrm{d}y', \mathrm{d}z' \mid \rho, y, z, a)$ existe car f est bornée. De plus, Q' étant un noyau stochastique, la fonction

$$(\rho, y, z, a) \mapsto r(\rho, y, z, a) + \int_{\mathbb{S}'} f(\rho', y', z')Q'(\mathrm{d}\rho', \mathrm{d}y', \mathrm{d}z' \mid \rho, y, z, a)$$

est mesurable par composition et bornée par $\|R\|_{\mathbb{L}} + \|f\|$. Il vient alors que la fonction

$$(\rho, y, z) \mapsto \sup_{a \in \mathbb{A}} \left[r(\rho, y, z, a) + \int_{\mathbb{S}'} f(\rho', y', z') Q'(\mathrm{d}\rho', \mathrm{d}y', \mathrm{d}z' \mid \rho, y, z, a) \right]$$

appartient à $\mathbb{B}(\mathbb{S}')$.

(iii) Soit $f \in \mathbb{B}(\mathbb{S}' \times \mathbb{A})$ et soit $\tilde{f}(\rho, y, z) = \sup_{a \in \mathbb{A}} f(\rho, y, z, a)$. Notre espace d'actions $\mathbb{A}$ étant est un ensemble fini, il est clair qu'il existe une fonction mesurable g de $\mathbb{S}'$ dans $\mathbb{A}$ telle que $\tilde{f}(\rho, y, z) = f(\rho, y, z, g(\rho, y, z))$ pour tout $(\rho, y, z) \in \mathbb{S}'$.

L'hypothèse de structure est donc respectée par notre modèle $\mathcal{M}'$.

B.3 Démonstration du lemme 8.2.3.9

Soit $f \in \mathbb{L}^*(\mathbb{X})$. Alors

$$\begin{aligned}
&\left| \int_{\mathbb{X}} f(x) \Phi_N(y', \rho, y)(\mathrm{d}x) - \int_{\mathbb{X}} f(x) \Phi_N(y', \tilde{\rho}, \tilde{y})(\mathrm{d}x) \right| \\
&\leq \mathbb{E}\left[|f(X_N)| \left| \frac{q(X_N, y' \mid \rho, y)}{q(\lambda_N, y' \mid \rho, y)} - \frac{q(X_N, y' \mid \tilde{\rho}, \tilde{y})}{q(\lambda_N, y' \mid \tilde{\rho}, \tilde{y})} \right| \right] \\
&\leq \mathbb{E}\left[\frac{1}{q(\lambda_N, y' \mid \rho, y)} \left| q(X_N, y' \mid \rho, y) - q(X_N, y' \mid \tilde{\rho}, \tilde{y}) \right| \right] \\
&\quad + \mathbb{E}\left[q(X_N, y' \mid \tilde{\rho}, \tilde{y}) \left| \frac{1}{q(\lambda_N, y' \mid \rho, y)} - \frac{1}{q(\lambda_N, y' \mid \tilde{\rho}, \tilde{y})} \right| \right]. \qquad \text{(B.16)}
\end{aligned}$$

Le corollaire 8.2.3.5 et la proposition 8.1.4.1 permettent d'avoir

$$\begin{aligned}
&\left| \int_{\mathbb{X}} f(x) \Phi_N(y', \rho, y)(\mathrm{d}x) - \int_{\mathbb{X}} f(x) \Phi_N(y', \tilde{\rho}, \tilde{y})(\mathrm{d}x) \right| \\
&\leq \delta E_N (\overline{q} + L_q) [\|\rho - \tilde{\rho}\|_{KR} + |y - \tilde{y}|] + \overline{q} (\delta E_N)^2 \left| q(\lambda_N, y' \mid \rho, y) - q(\lambda_N, y' \mid \tilde{\rho}, \tilde{y}) \right|.
\end{aligned}$$

On conclut en utilisant le lemme 8.2.3.2.

B.4 Démonstration de la proposition 8.2.4.2

Soient $f \in \mathbb{L}(\mathcal{P}(\mathbb{X}) \times \mathbb{Y})$, $N \geq M_0$, $\rho, \rho' \in \mathcal{P}(\mathbb{X})$ et $y, y' \in \mathbb{Y}$. Alors d'une part,

$$\begin{aligned}
|\mathcal{B}_N f(\rho, y)| &\leq \max\{|R(\rho, y)|\,; |S_N f(\rho, y)|\} \\
&\leq \max\left\{ \|R\|_{\mathbb{L}}\,; \frac{1}{q(\lambda_N, \nu \mid \rho, y)} \int_{\mathbb{X} \times \mathbb{Y}} \left| f\big(\Phi_N(y', \rho, y), y'\big) \right| q(x', y' \mid \rho, y) \lambda_N(\mathrm{d}x') \nu(\mathrm{d}y') \right\} \\
&\leq \max\{\|R\|_{\mathbb{L}}\,; \|f\|_{\mathbb{L}}\, \delta E_N\}. \qquad \text{(B.17)}
\end{aligned}$$

D'autre part, on a

$$\begin{aligned}
&\left|S_N f(\rho,y) - S_N f(\rho',y')\right| \\
&\leq \mathbb{E}\left[\frac{\left|f\big(\Phi_N(Y,\rho,y),Y\big)\right|}{q(\lambda_N,\nu \mid \rho,y)}\left|q(X_N,Y \mid \rho,y) - q(X_N,Y \mid \rho',y')\right|\right] \\
&\quad + \mathbb{E}\left[q(X_N,Y \mid \rho',y')\left|f\big(\Phi_N(Y,\rho,y),Y\big)\right|\left|\frac{1}{q(\lambda_N,\nu \mid \rho,y)} - \frac{1}{q(\lambda_N,\nu \mid \rho',y')}\right|\right] \\
&\quad + \mathbb{E}\left[\frac{q(X_N,Y \mid \rho',y')}{q(\lambda_N,\nu \mid \rho',y')}\left|f\big(\Phi_N(Y,\rho,y),Y\big) - f\big(\Phi_N(Y,\rho',y'),Y\big)\right|\right]. \qquad \text{(B.18)}
\end{aligned}$$

L'utilisation du corollaire 8.2.3.6, du lemme 8.2.3.3 et de la proposition 8.1.4.1 amène à

$$\begin{aligned}
\left|S_N f(\rho,y) - S_N f(\rho',y')\right| \leq\ & \delta E_N \left\|f\right\|_{\mathbb{L}} (\overline{q} + L_q)[\left\|\rho - \rho'\right\|_{KR} + \left|y - y'\right|] \\
& + \overline{q}\left\|f\right\|_{\mathbb{L}} (\delta E_N)^2 (\overline{q} + L_q)[\left\|\rho - \rho'\right\|_{KR} + \left|y - y'\right|] \\
& + \overline{q}\left\|f\right\|_{\mathbb{L}} \delta E_N \mathbb{E}\left[\left\|\Phi_N(Y,\rho,y) - \Phi_N(Y,\rho',y')\right\|_{KR}\right]. \qquad \text{(B.19)}
\end{aligned}$$

On termine par la majoration de $\mathbb{E}\left[\left\|\Phi_N(Y,\rho,y) - \Phi_N(Y,\rho',y')\right\|_{KR}\right]$ en utilisant le lemme 8.2.3.9 pour avoir

$$\left|S_N f(\rho,y) - S_N f(\rho',y')\right| \leq \delta E_N \left\|f\right\|_{\mathbb{L}} (\overline{q} + L_q)(1 + \overline{q}\delta E_N)^2 [\left\|\rho - \rho'\right\|_{KR} + \left|y - y'\right|]. \qquad \text{(B.20)}$$

Donc en utilisant la proposition 8.1 et l'inégalité (8.46),

$$\begin{aligned}
&\left|\mathcal{B}_N f(\rho,y) - \mathcal{B}_N f(\rho',y')\right| \\
&\leq \left|R(\rho,y) - R(\rho',y')\right| + \left|S_N f(\rho,y) - S_N f(\rho',y')\right| \\
&\leq \big(\left\|R\right\|_{\mathbb{L}} + \delta E_N \left\|f\right\|_{\mathbb{L}} (\overline{q} + L_q)(1 + \overline{q}\delta E_N)^2\big)[\left\|\rho - \rho'\right\|_{KR} + \left|y - y'\right|]. \qquad \text{(B.21)}
\end{aligned}$$

d'où le résultat.

B.5 Démonstration du théorème 8.3.2.1

La démonstration de ce théorème est très similaire à celle du théorème 2 de [BP03] (section 2.2.2). En revanche, notre processus ne respecte pas tout à fait les hypothèses de cet article. Les propriétés qu'il vérifie sont toutefois assez proches. Nous appliquons donc le même raisonnement, mais avec nos hypothèses.

Dans toute cette preuve, nous noterons

$$\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) = \mathbb{E}\left[T_{k-1}^{(N)} \;\middle|\; (P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)})\right]$$

et

$$\varphi_k^{(M,N)}(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)}) = \mathbb{E}\left[T_{k-1}^{(M,N)} \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right].$$

Soit $k \in [\![1; N_0]\!]$. Par l'inégalité (8.46), on écrit que

$$\begin{aligned}\left|T_k^{(N)} - T_k^{(M,N)}\right| &\leq \left|R(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) - R(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right| \\ &\quad + \left|\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) - \varphi_k^{(M,N)}(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right| \\ &\leq \|R\|_{\mathbb{L}} \sqrt{N} \eta_M^{(N,N_0-k)} \\ &\quad + \left|\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) - \mathbb{E}\left[\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right]\right| \\ &\quad + \left|\mathbb{E}\left[\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right] - \varphi_k^{(M,N)}(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right|\end{aligned} \tag{B.22}$$

Écrivons maintenant que

$$\begin{aligned}&\left|\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) - \mathbb{E}\left[\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right]\right| \\ &\leq \left|\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) - \varphi_k^{(N)}(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right| \\ &\quad + \left|\varphi_k^{(N)}(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)}) - \mathbb{E}\left[\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right]\right|.\end{aligned} \tag{B.23}$$

Or, comme $\varphi_k^{(N)}(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})$ est $\sigma\left(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)}\right)$-mesurable, alors

$$\varphi_k^{(N)}(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)}) = \mathbb{E}\left[\varphi_k^{(N)}(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)}) \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right] \tag{B.24}$$

et donc

$$\begin{aligned}&\left|\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) - \mathbb{E}\left[\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right]\right| \\ &\leq \left|\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) - \varphi_k^{(N)}(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right| \\ &\quad + \mathbb{E}\left[\left|\varphi_k^{(N)}(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)}) - \varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)})\right| \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right].\end{aligned} \tag{B.25}$$

Or l'équation (8.66) donne

$$\begin{aligned}&\left|\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) - \mathbb{E}\left[\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right]\right| \\ &\leq \left|S_N v_{k-1}^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) - S_N v_{k-1}^{(N)}(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right| \\ &\quad + \mathbb{E}\left[\left|S_N v_{k-1}^{(N)}(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)}) - S_N v_{k-1}^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)})\right| \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right].\end{aligned} \tag{B.26}$$

On utilise ici l'inéquation (B.20) et on obtient alors

$$\begin{aligned}&\left|\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) - \mathbb{E}\left[\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right]\right| \\ &\leq \left\|v_{k-1}^{(N)}\right\|_{\mathbb{L}} \sqrt{N} \delta E_N (\overline{q} + L_q)(1 + \overline{q}\delta E_N)^2 \Bigg[\left(\left|P_{N_0-k}^{(N)} - P_{N_0-k}^{(M,N)}\right| + \left|Y_{N_0-k}^{(N)} - Y_{N_0-k}^{(M,N)}\right|\right) \\ &\quad + \mathbb{E}\left[\left(\left|P_{N_0-k}^{(N)} - P_{N_0-k}^{(M,N)}\right| + \left|Y_{N_0-k}^{(N)} - Y_{N_0-k}^{(M,N)}\right|\right) \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right]\Bigg].\end{aligned} \tag{B.27}$$

On en conclut que

$$\begin{aligned}&\left\|\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) - \mathbb{E}\left[\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right]\right\|_2 \\ &\leq 2\sqrt{N} \left\|v_{k-1}^{(N)}\right\|_{\mathbb{L}} \delta E_N (\overline{q} + L_q)(1 + \overline{q}\delta E_N)^2 \eta_M^{(N,N_0-k)}.\end{aligned} \tag{B.28}$$

Notons maintenant que

$$\mathbb{E}\left[\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right] = \mathbb{E}\left[T_{k-1}^{(N)} \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right] \tag{B.29}$$

car $\sigma\left(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)}\right) \subset \sigma\left(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}\right)$. Il vient alors que

$$\begin{aligned}&\left|\mathbb{E}\left[\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right] - \varphi_k^{(M,N)}(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right| \\ &\leq \mathbb{E}\left[\left|T_{k-1}^{(N)} - T_{k-1}^{(M,N)}\right| \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right].\end{aligned} \tag{B.30}$$

Donc, on obtient que

$$\left\|\mathbb{E}\left[\varphi_k^{(N)}(P_{N_0-k}^{(N)}, Y_{N_0-k}^{(N)}) \;\middle|\; (P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right] - \varphi_k^{(M,N)}(P_{N_0-k}^{(M,N)}, Y_{N_0-k}^{(M,N)})\right\|_2 \leq \left\|T_{k-1}^{(N)} - T_{k-1}^{(M,N)}\right\|_2. \tag{B.31}$$

Par conséquent, on trouve que

$$\left\|T_k^{(N)} - T_k^{(M,N)}\right\|_2 \leq \eta_M^{(N,N_0-k)}\sqrt{N}\left(\|R\|_{\mathbb{L}} + 2\left\|v_{k-1}^{(N)}\right\|_{\mathbb{L}}\delta E_N(\overline{q}+L_q)(1+\overline{q}\delta E_N)^2\right) \tag{B.32}$$

$$+ \left\|T_{k-1}^{(N)} - T_{k-1}^{(M,N)}\right\|_2 \tag{B.33}$$

et donc une récurrence rapide permet d'arriver à la formule voulue.

B.6 Démonstration de la proposition 8.4.0.1

Il est à noter que la densité g étant lipschitzienne sur $[-K;K]$, elle est continue sur ce compact donc majorée par un réel positif noté $\|g\|_\infty$. Ainsi, pour tous $y, y' \in [-K;K]$,

$$\left|\int_{-\infty}^{y} g(t)\,\mathrm{d}t - \int_{-\infty}^{y'} g(t)\,\mathrm{d}t\right| = \left|\int_{y'}^{y} g(t)\,\mathrm{d}t\right| \leq \|g\|_\infty \left|y - y'\right|. \tag{B.34}$$

Donc G est lipschitzienne sur $[-K;K]$ de rapport au plus $\|g\|_\infty$. Le même argument donne que f est majorée sur $[0;K]$ par un réel positif noté $\|f\|_\infty$ et que F est lipschitzienne sur $[0;K]$ de rapport au plus $\|f\|_\infty$. Nous utiliserons ces propriétés tout au long de cette preuve.

Commençons par montrer les hypothèses A :

A(1). Soient $B_1 \in \mathcal{B}([0;K])$, $B_2 \subset \{-1;0;1\}$ et $C \in \mathcal{B}([0;K])$. La loi de transition P du système considéré est

$$\begin{aligned}&P(B_1 \times B_2 \times C \mid x_1, x_2, y) \\ &= \delta_0(B_2)\int_{-\infty}^{K-x_1}\int_{\mathbb{R}^2} \mathbb{1}_{B_1}\left((x_1+\xi)\wedge K\right)\mathbb{1}_C\left(\left((x_1+\xi)\wedge K + \psi\right)_+ \wedge K\right) f(\xi)g(\psi)\mathrm{d}\xi\mathrm{d}\psi \\ &\quad + \delta_1(B_2)\int_{K-x_1}^{+\infty}\int_{\mathbb{R}^2} \mathbb{1}_{B_1}\left((x_1+\xi)\wedge K\right)\mathbb{1}_C\left(\left((x_1+\xi)\wedge K + \psi\right)_+ \wedge K\right) f(\xi)g(\psi)\mathrm{d}\xi\mathrm{d}\psi \\ &= \delta_0(B_2)\int_{-\infty}^{K-x_1}\int_{\mathbb{R}^2} \mathbb{1}_{B_1}\left(x_1+\xi\right)\mathbb{1}_C\left(\left(x_1+\xi+\psi\right)_+ \wedge K\right) f(\xi)g(\psi)\mathrm{d}\xi\mathrm{d}\psi \\ &\quad + \delta_1(B_2)\int_{K-x_1}^{+\infty}\int_{\mathbb{R}^2} \mathbb{1}_{B_1}\left(K\right)\mathbb{1}_C\left(\left(K+\psi\right)_+ \wedge K\right) f(\xi)g(\psi)\mathrm{d}\xi\mathrm{d}\psi.\end{aligned} \tag{B.35}$$

Dès lors, les changements de variable successifs $\xi \mapsto x_1 + \xi$ et $\psi \mapsto \xi + \psi$ dans la première intégrale et $\psi \mapsto \psi + K$ dans la deuxième intégrale donnent

$$P(B_1 \times B_2 \times C \mid x_1, x_2, y) = \delta_0(B_2) \int_{B_1 \cap [x_1; K[} f(\xi - x_1)\mathfrak{M}_2(C, \xi)\mathrm{d}\xi + \delta_K(B_1)\delta_1(B_2)(1 - F(K - x_1))\mathfrak{M}_3(C). \tag{B.36}$$

où

$$\mathfrak{M}_1(C, \xi) = \delta_0(C)G(-\xi) + \int_C g(\psi - \xi)\,\mathrm{d}\psi + \delta_K(C)(1 - G(K - \xi)), \tag{B.37}$$

$$\mathfrak{M}_2(C) = \delta_0(C)G(-K) + \int_C g(\psi - K)\,\mathrm{d}\psi + \delta_K(C)(1 - G(0)). \tag{B.38}$$

Soit maintenant la fonction $q : (\mathbb{X} \times \mathbb{Y})^2 \to \mathbb{R}_+$ définie par

$$q(x_1', x_2', y' \mid x_1, x_2, y) = 2K\mathbb{1}_{[x_1; K[\times \{0\}}(x_1', x_2')f(x_1' - x_1)\mathfrak{m}_1(y', x_1') + 2\mathbb{1}_{\{(K;1)\}}(x_1', x_2')(1 - F(K - x_1))\mathfrak{m}_2(y') \tag{B.39}$$

où

$$\mathfrak{m}_1(y', x_1') = 4G(-x_1')\mathbb{1}_{\{0\}}(y') + 2K\mathbb{1}_{]0;K[}(y')g(y' - x_1') + 4(1 - G(K - x_1'))\mathbb{1}_{\{K\}}(y'), \tag{B.40}$$

$$\mathfrak{m}_2(y') = 4G(-K)\mathbb{1}_{\{0\}}(y') + 2K\mathbb{1}_{]0;K[}(y')g(y' - K) + 4(1 - G(0))\mathbb{1}_{\{K\}}(y') \tag{B.41}$$

sont des fonctions positives.

On déduit de l'écriture que P que ce noyau stochastique admet q pour densité par rapport au produit tensoriel des mesures de probabilité λ sur $\mathbb{X}$ et ν sur $\mathbb{Y}$ suivantes :

$$\lambda(B_1 \times B_2) = \frac{\ell_1(B_1)\delta_0(B_2)}{2K} + \frac{\delta_K(B_1)\delta_1(B_2)}{2}, \tag{B.42}$$

$$\nu(C) = \frac{\delta_0(C)}{4} + \frac{\ell_1(C)}{2K} + \frac{\delta_K(C)}{4} \tag{B.43}$$

pour $B_1 \in \mathcal{B}([0; K])$, $B_2 \subset \{-1; 0; 1\}$ et $C \in \mathcal{B}([0; K])$. Ces deux mesures sont évidemment σ-finies.

A(2). Il est clair que la mesure λ possède des moments de tous ordres.

Ensuite, vérifions si la densité q que l'on a exhibée respecte les hypothèses B.

B(1). Il est facile de voir que les fonctions $\mathfrak{m}_1$ et $\mathfrak{m}_2$ sont majorées par $8 + 2K \|g\|_\infty$. On obtient donc que

$$q(x_1', x_2', y' \mid x_1, x_2, y) \leq 2(K \|f\|_\infty + 1)(8 + 2K \|g\|_\infty). \tag{B.44}$$

B(2). Pour démontrer que la densité q vérifie l'hypothèse B(2), nous allons procéder en deux temps. D'une part, nous allons montrer que $(x_1', x_2', y', x_1, x_2, y) \mapsto q(x_1', x_2', y' \mid x_1, x_2, y)$ est lipschitzienne par rapport à (x_1', x_2'). Pour ce faire, on va fixer $y, y' \in \mathbb{Y}$ et $(x_1, x_2) \in \mathbb{X}$ et distinguer deux cas.

— Si $x_1', x_1'' \in [x_1; K[$ (c'est-à-dire $x_2' = x_2'' = 0$) :

$$\begin{aligned}
&\left|q(x_1', 0, y' \mid x_1, x_2, y) - q(x_1'', 0, y' \mid x_1, x_2, y)\right| \\
&\leq 2K\mathfrak{m}_1(y', x_1') \left|f(x_1' - x_1) - f(x_1'' - x_1)\right| + 2Kf(x_1'' - x_1) \left|\mathfrak{m}_1(y', x_1') - \mathfrak{m}_1(y', x_1'')\right| \\
&\leq 2K(L_f(8 + 2K \|g\|_\infty) + \|f\|_\infty (8 \|g\|_\infty + 2KL_g)) \left|x_1' - x_1''\right| \\
&\leq 2K(L_f(8 + 2K \|g\|_\infty) + \|f\|_\infty (8 \|g\|_\infty + 2KL_g)) \left[\left|x_1' - x_1''\right| + \left|x_2' - x_2''\right|\right]. \quad \text{(B.45)}
\end{aligned}$$

En effet, $|x_2' - x_2''| = 0$ dans ce cas et

$$\begin{aligned}
&\left|\mathfrak{m}_2(y', x_1') - \mathfrak{m}_2(y', x_1'')\right| \leq 4 \left|G(-x_1') - G(-x_1'')\right| \\
&\quad + 2K \left|g(y' - x_1') - g(y' - x_1'')\right| + 4 \left|G(K - x_1') - G(K - x_1'')\right| \\
&\leq (8 \|g\|_\infty + 2KL_g) \left|x_1' - x_1''\right|. \quad \text{(B.46)}
\end{aligned}$$

— Si $x_1' \in [x_1; K[$ et $x_1'' = K$ (c'est-à-dire $x_2' = 0$ et $x_2'' = 1$), le cas symétrique étant analogue :

$$\begin{aligned}
&\left|q(x_1', 0, y' \mid x_1, x_2, y) - q(K, 1, y' \mid x_1, x_2, y)\right| \\
&\leq 2K\mathfrak{m}_1(y', x_1') \left|f(x_1' - x_1) - f(K - x_1)\right| \\
&\quad + 2Kf(K - x_1)\mathfrak{m}_1(y', x_1') + 2(1 - F(K - x_1))\mathfrak{m}_2(y') \\
&\leq 2\mathfrak{a}(8 + 2K \|g\|_\infty) \left[\left|x_1' - K\right| + 1\right] \\
&\leq 2\mathfrak{a}(8 + 2K \|g\|_\infty) \left[\left|x_1' - x_1''\right| + \left|x_2' - x_2''\right|\right] \quad \text{(B.47)}
\end{aligned}$$

où

$$\mathfrak{a} = \max\left(KL_f; K \|f\|_\infty + 1\right). \quad \text{(B.48)}$$

On a donc démontré ce que l'on souhaitait.

D'autre part, nous allons montrer que $(x_1', x_2', y', x_1, x_2, y) \mapsto q(x_1', x_2', y' \mid x_1, x_2, y)$ est lipschitzienne par rapport à (x_1, x_2, y) sur $\mathbb{X}\times\mathbb{Y}$. Considérons $(x_1', x_2'), (x_1, x_2), (\tilde{x_1}, \tilde{x_2}) \in \mathbb{X}$, avec $x_1' \geq x_1$ et $x_1' \geq \tilde{x_1}$, et $y, y', \tilde{y} \in \mathbb{Y}$.

$$\begin{aligned}
&\left|q(x_1', x_2', y' \mid x_1, x_2, y) - q(x_1', x_2', y' \mid \tilde{x_1}, \tilde{x_2}, \tilde{y})\right| \\
&\leq 2K\mathfrak{m}_1(y', x_1') \left|f(x_1' - x_1) - f(x_1' - \tilde{x_1})\right| \\
&\quad + 2\mathfrak{m}_2(y') \left|F(K - x_1) - F(K - \tilde{x_1})\right| \\
&\leq 2(8 + 2K \|g\|_\infty)(KL_f + \|f\|_\infty) \left|x_1 - \tilde{x_1}\right| \\
&\leq 2(8 + 2K \|g\|_\infty)(KL_f + \|f\|_\infty)\left[\left|x_1 - \tilde{x_1}\right| + \left|x_2 - \tilde{x_2}\right| + \left|y - \tilde{y}\right|\right]. \quad \text{(B.49)}
\end{aligned}$$

Il vient alors que la densité q respecte l'hypothèse B(2).

Maintenant, démontrons que la densité q vérifie l'hypothèse C. Considérons $y, y' \in \mathbb{Y}$ et $(x_1, x_2) \in \mathbb{X}$. On a que

$$\int_{\mathbb{X}} q(x_1', x_2', y' \mid x_1, x_2, y)\lambda(\mathrm{d}x_1', \mathrm{d}x_2') = \int_{x_1}^{K} f(x_1' - x_1)\mathfrak{m}_1(y', x_1')\,\mathrm{d}x_1' + (1 - F(K - x_1))\mathfrak{m}_2(y'). \quad \text{(B.50)}$$

Les deux termes du membre de droite étant positifs, on peut écrire

$$\int_{\mathbb{X}} q(x_1', x_2', y' \mid x_1, x_2, y)\lambda(\mathrm{d}x_1', \mathrm{d}x_2') \geq (1 - F(K - x_1))\mathfrak{m}_3(y'). \tag{B.51}$$

Or, g est continue sur le compact $[-K; K]$ et strictement positive sur cet intervalle. Elle y est donc minorée par un réel $\gamma^- > 0$. On a alors que

$$\mathfrak{m}_2(y') \geq \min\big(4G(-K); 2K\gamma^-; 4(1 - G(0))\big), \tag{B.52}$$

ce qui entraîne la minoration

$$\int_{\mathbb{X}} q(x_1', x_2', y' \mid x_1, x_2, y)\lambda(\mathrm{d}x_1', \mathrm{d}x_2') \geq (1 - F(K)) \min\big(4G(-K); 2K\gamma^-; 4(1 - G(0))\big). \tag{B.53}$$

Par hypothèse, $F(K) < 1$, $G(-K) > 0$ et $G(0) < 1$. Donc

$$(1 - F(K)) \min\big(4G(-K); 2K\gamma^-; 4(1 - G(0))\big) > 0.$$

On a bien ce qu'on cherchait.

Enfin, il est facile de voir que R respecte l'hypothèse D. La proposition est donc vraie.

Bibliographie

[BAT13] J.S. Borrero et R. Akhavan-Tabatabaei : Time and inventory dependent optimal maintenance policies for single machine workstations : An MDP approach. *European Journal of Operational Research*, 228(3):545 – 555, 2013.

[BDFN12] P. Bianchi, L. Decreusefond, G. Fort et J. Najim : Cours de probabilités – MDI 104, version de novembre 2012.

[BDM11] Nathalie Bartoli et Pierre Del Moral : *Simulation & algorithmes stochastiques*. Éditions Cépaduès, 2011.

[Ber87] Dimitri P. Bertsekas : *Dynamic programming : deterministic and stochastic models*. Prentice Hall, Inc., Englewood Cliffs, NJ, 1987.

[Bog07] Vladimir I. Bogachev : *Measure Theory*, volume 2. Springer Berlin Heidelberg, 2007.

[BP03] Vlad Bally et Gilles Pagès : A quantization algorithm for solving multi-dimensional discrete-time optimal stopping problems. *Bernoulli*, 9(6):1003 – 1049, 2003.

[BR11] Nicole Bäuerle et Ulrich Rieder : *Markov decision processes with applications to finance*. Universitext. Springer, Heidelberg, 2011.

[BS78] Dimitri P. Bertsekas et Steven E. Shreve : *Stochastic optimal control : the discrete-time case*, volume 139 de *Mathematics in Science and Engineering*. Academic Press, Inc. [Harcourt Brace Jovanovich, Publishers], New-York-London, 1978.

[CFHM07] Hyeong Soo Chang, Michael C. Fu, Jiaqiao Hu et Steven I. Marcus : An asymptotically efficient simulation-based algorithm for finite horizon stochastic dynamic programming. *IEEE Trans. Automat. Control*, 52(1):89 – 94, 2007.

[CHFM13] Hyeong Soo Chang, Jiaqiao Hu, Michael C. Fu et Steven I. Marcus : *Simulation-based algorithms for Markov decision processes*. Communications and Control Engineering Series. Springer, London, 2013.

[CL88] Morris A. Cohen et Hau L. Lee : Strategic analysis of integrated production-distribution systems : models and methods. *Operations research*, 36(2):216 – 228, 1988.

[Dan98] George B. Dantzig : *Linear programming and extensions.* Princeton Landmwarks in Mathematics. Princeton University Press, Princeton, NJ, corrected édition, 1998.

[Den03] Eric V. Denardo : *Dynamic programming : models and applications.* Dover Publications, Inc., Mineola, NY, 2003.

[DP10] François Dufour et Alexey Piunovskiy : Multiobjective stopping problem for discrete-time Markov processes : convex analytic approach. *Journal of Applied Probability*, 47:947–966, 2010.

[DPR12] François Dufour et Tomás Prieto-Rumeau : Approximation of Markov decision processes with general state space. *Journal of Mathematical Analysis and Applications*, 388(2):1254 – 1267, 2012.

[DPR13] François Dufour et Tomás Prieto-Rumeau : Finite linear programming approximations of constrained discounted Markov decision processes. *Society for Industrial and Applied Mathematics Journal on Control and Optimization*, 51(2):1298 – 1324, 2013.

[DPR15] François Dufour et Tomás Prieto-Rumeau : Approximation of average cost Markov decision processes using empirical distributions and concentration inequalities. *Stochastics. An International Journal of Probability and Stochastic Processes*, 87(2):273 – 307, 2015.

[dSDZE10] Benoîte de Saporta, François Dufour, Huilong Zhang et Charles Elegbede : Arrêt optimal pour la maintenance prédictive. *In 17e Congrès de Maîtrise des Risques et de Sûreté de Fonctionnement, 5 – 7 octobre 2010, La Rochelle*, pages 4A–3, France, 2010.

[EBBBS13] Charles Elegbede, Damien Bérard-Bergery, Jacques Béhar et Tristan Stauffer : Dynamical modelling and stochastic optimization for the design of a launcher integration process. *In* R.D.J.M. Steenbergen et al., éditeur : *Safety, Reliability and Risk analysis : beyond the horizon*, pages 3039 – 3046. CRC Press, 2013.

[GL00] Siegfried Graf et Harald Luschgy : *Foundations of quantization for probability distributions*, volume 1730 de *Lecture Notes in Mathematics.* Springer-Verlag, Berlin, 2000.

[HF00] Eric Hansen et Zhengzhu Feng : Dynamic programming for POMDPs using a factored state representation. *In AIPS*, pages 130 – 139, 2000.

[HFM08] Jiaqiao Hu, Michael C. Fu et Steven I. Marcus : A model reference adaptive search method for stochastic global optimization. *Communications in Information and Systems*, 8(3):245 – 275, 2008.

[HHC12] Jiaqiao Hu, Ping Hu et Hyeong Soo Chang : A stochastic approximation framework for a class of randomized optimization algorithms. *IEEE Trans. Automat. Control*, 57(1):165 – 178, 2012.

[HL89] Onésimo Hernández-Lerma : *Adaptive Markov control processes*, volume 79 de *Applied Mathematical Sciences*. Springer-Verlag, New York, 1989.

[HLL96] Onésimo Hernández-Lerma et Jean Bernard Lasserre : *Discrete-Time Markov Control Processes : Basic Optimality Criteria*, volume 30 de *Applications of Mathematics*. New-York : Springer-Verlag, 1996.

[HLL99] Onésimo Hernández-Lerma et Jean Bernard Lasserre : *Further topics on discrete-time Markov control processes*, volume 42 de *Applications of Mathematics*. Springer-Verlag, New York, 1999.

[How60] Ronald A. Howard : *Dynamic programming and Markov processes*. The Technology Press of M.I.T., Cambridge, Mass. ; John Wiley & Sons, Inc., New-York-London, 1960.

[HP05] Jesse Hoey et Pascal Poupart : Solving POMDPs with continuous or large discrete observation spaces. *In IJCAI*, pages 1332 – 1338, 2005.

[HS04] Daniel P. Heyman et Matthew J. Sobel : *Stochastic models in operations research Vol II*. Dover Publications, Inc., Mineola, NY, 2004.

[Jen97] Kurt Jensen : *Coloured Petri nets. Basic concepts, analysis methods and practical use*. Monographs in Theoretical Computer Science. An EATCS Series. Springer-Verlag, Berlin, 1997.

[JM74] Lynwood A. Johnson et Douglas C. Montgomery : *Operations research in production planning, scheduling, and inventory control*, volume 6. Wiley, New-York, 1974.

[Mur09] Katta G. Murty : Linear programming. *In Operations reserach methodologies*, Oper. Res. Ser., pages 1 – 35. CRC Press, Boca Raton, FL, 2009.

[NdSD+15] Christophe Nivot, Benoîte de Saporta, François Dufour, Jacques Béhar, Damien Bérard-Bergery et Charles Elegbede : Modeling and optimization of a launcher integration process. *In* Luca Podofillini et al., éditeur : *Safety and Reliability of Complex Engineered Systems : ESREL 2015*, pages 2281 – 2288. CRC Press, 2015.

[Pag98] Gilles Pagès : A space quantization method for numerical integration. *Journal of Computational and Applied Mathematics*, 89(1):1 – 38, 1998.

[Pha07] Huyên Pham : Méthodes de quantification optimale et applications en finance. Cours de Master 2, version de 2006-2007.

[PP03] Gilles Pagès et Jacques Printems : Optimal quadratic quantization for numerics : the gaussian case. *Monte Carlo Methods and Applications*, 9(2):135 – 165, 2003.

[PPP04] Gilles Pagès, Huyên Pham et Jacques Printems : Optimal quantization methods and applications to numerical problems in finance. *In Handbook of computational and numerical methods in finance*, pages 253–297. Birkhäuser Boston, Boston, MA, 2004.

[PRS05] Huyên Pham, Wolfgang Runggaldier et Afef Sellami : Approximation by quantization of the filter process and applications to optimal stopping problems under partial observation. *Monte Carlo Methods and Applications*, 11(1):57 – 81, 2005.

[Put94] Martin L. Puterman : *Markov decision processes : discrete stochastic dynamic programming*. Wiley Series in Probability and Mathematical Statistics : Applied Probability and Statistics. John Wiley & Sons, Inc., New-York, 1994.

[RPPCD08] Stéphane Ross, Joelle Pineau, Sébastien Paquet et Brahim Chaib-Draa : Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research*, 32:663 – 704, 2008.

[SL96] MM Srinivasan et H-S Lee : Production-inventory systems with preventive maintenance. *IIE transactions*, 28(11):879 – 890, 1996.

[vdA94] W.M.P. van der Aalst : Putting high-level Petri nets to work in industry. *Computers in industry*, 25(1):45 – 54, 1994.

[WW89] Chelsea C. White, III et Douglas J. White : Markov decision processes. *European Journal of Operational Research*, 39(1):1 – 16, 1989.

[YZ13] Fan Ye et Enlu Zhou : Optimal stopping of partially observable Markov processes : a filtering-based duality approach. *Automatic Control, IEEE Transactions on*, 58(10):2698 – 2704, 2013.

[ZFM10] Enlu Zhou, Michael C. Fu et Steven I. Marcus : Solving continuous-state POMDPs via density projection. *IEEE Trans. Automat. Control*, 55(5):1101 – 1116, 2010.

[Zho13] Enlu Zhou : Optimal stopping under partial observation : near-value iteration. *Automatic Control, IEEE Transactions on*, 58(2):500 – 506, 2013.