Caractérisation des erreurs de séquençage non aléatoires : application aux mosaïques et tumeurs hétérogènes

Chadi Saad

Université de Lille

Université de Lille
École Doctorale Biologie-Santé de Lille
UMR 9189 - CRIStAL
UMR-S 1172 - JPArc

Caractérisation des erreurs de séquençage non aléatoires

application aux mosaïques et tumeurs hétérogènes

Thèse

Présentée pour l'obtention du grade de :
Docteur de l'Université de Lille

Spécialité : Bio-informatique

Par

Chadi Saad

Soutenue le 26 Septembre 2018 devant un jury composé de :

Présidente de jury	**Isabelle Van Seuningen**	(DR CNRS, JPArc, Lille)
Rapporteur	**Jacques Nicolas**	(DR Inria, Inria & Irisa, Rennes)
Rapportrice	**Morgane Thomas-Chollier**	(MCF ENS, IBENS, Paris)
Examinatrice	**Marie De Tayrac**	(MCU-PH univ. Rennes, IGDR, Rennes)
Examinateur	**Gilles Didier**	(CR CNRS, I2M, Marseille)
Encadrant	**Martin Figeac**	(IR univ. Lille, Lille)
Co-Directrice de thèse	**Marie-Pierre Buisine**	(PU-PH univ. Lille, JPARC, Lille)
Co-Directrice de thèse	**Hélène Touzet**	(DR CNRS, CRIStAL & Inria, Lille)

AFFILIATIONS

Thèse effectuée en collaboration entre les deux UMR :

UMR 9189 - CRIStAL
Université de Lille, CNRS, Inria
Avenue Carl Gauss
59655 Villeneuve d'Ascq Cedex
France

UMR-S 1172 - JPArc
Université de Lille, INSERM, CHRU de Lille
1, place de Verdun
59045 Lille Cedex
France

PARTENAIRES

Centre Hospitalier Régional
Universitaire de Lille

Cette thèse a bénéficié du soutien financier de la Région Hauts-de-France et du Centre hospitalier régional universitaire de Lille

"... I can't be as confident about computer science as I can about biology. Biology easily has 500 years of exciting problems to work on. It's at that level."

— Donald E. Knuth (1993)

"Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful."

— George E. P. Box, Norman Richard Draper (1987)

Remerciements

J'aimerais adresser mes premiers remerciements à mes directrices de m'avoir accepté en cette thèse. Merci à Hélène Touzet, pour tous ses précieux conseils et son soutien scientifique comme moral (à chaque fois qu'elle remarquait sur mon visage que quelque chose n'allait pas, elle venait me demander si tout allait bien), c'était vraiment rassurant de travailler sous ta direction. Merci également à ma co-directrice Marie-Pierre Buisine, de m'avoir gardé en thèse, après m'avoir accepté en apprentissage dans son laboratoire durant mon M2. Même si géographiquement elle était plus loin, elle était toujours là à mon écoute et elle m'a suivi durant toute la période de thèse, avec tout son soutien. J'ai eu vraiment de la chance de vous avoir comme directrices de ma thèse. Vous étiez présentes malgré vos nombreuses responsabilités. Merci également pour vos contributions aux nombreuses corrections de mon manuscrit de thèse.

Je remercie très chaleureusement mes encadrants. Martin Figeac est, sans doute, la personne avec laquelle j'ai interagi le plus. C'est avec Martin que j'ai découvert le monde du travail en bio-informatique en Master en 2012. Depuis 6 ans, il est mon encadrant au jour le jour, nous avons partagé de nombreux moments tant personnels que professionnels, ce qui a fait de lui bien plus qu'un encadrant. Son optimisme et sa motivation sont sans limite ! Je suis ravi de pouvoir continuer à travailler avec toi. Merci à Julie Leclerc qui depuis l'encadrement de mon M2 avec Martin m'encadre également depuis 6 ans ! Merci pour tous tes conseils et ta disponibilité permanente à chaque fois que j'avais besoin d'aide et que j'avais des questions qui touchaient à la biologie. Merci également pour tes nombreuses aides pour la correction de tous ce que je rédigeais rapports/posters/manuscrit. Merci à Laurent Noé, mon troisième encadrant (eh oui, j'ai été gâté avec trois encadrants). Laurent m'apportait le soutien pour tout ce qui touche principalement à l'informatique. Chaque fois que je discutais avec toi, j'apprenais de nouvelles choses intéressantes. Certes, je venais parfois avec une question et sortais avec dix, mais c'est ça la recherche... Merci à vous de votre aide et votre disponibilité, je vous en suis extrêmement reconnaissant.

Je tiens sincèrement à remercier les membres de mon jury de thèse. Je remercie Mme Isabelle Van Seuningen (Directeur de Recherche CNRS) de m'avoir accepté comme membre de son équipe et d'avoir accepté de présider mon jury. Je remercie M. Jacques Nicolas (Directeur de Recherche, INRIA) et Mme Morgane Thomas-Chollier (Maître de conférences à l'ENS Paris) qui m'ont fait l'honneur de juger mes travaux, ainsi que

Mme Marie De Tayrac (Maître de conférences des Universités-praticien hospitalier de l'Université de Rennes) et M. Gilles Didier (Chargé de recherche, CNRS) pour avoir accepté de faire partie de mon jury.

Merci à Hugues Richard (Maître de conférences à l'Université Pierre & Marie Curie) avec qui j'ai eu de nombreux et précieux échanges durant ma thèse.

Merci à Florian Vanhems que j'ai eu l'occasion d'encadrer avec Laurent durant son stage de L3. Il a travaillé sur l'implémentation de DiNAMO en C++. C'était un plaisir de travailler avec lui.

Merci à l'équipe pédagogique de Polytech, pour leur confiance et pour l'expérience enrichissante que j'ai eue en enseignant l'informatique durant mon ATER.

Merci au CHRU de Lille et à la Région Hauts-de-France pour le financement de ma thèse. Merci à tous les membres de l'équipe Bonsaï pour l'accueil et tous les bons moments que nous avons partagés durant les pauses de 16h, les pique-niques du Mardi midi, les journées au vert... L'ambiance et l'environnement de travail étaient vraiment sympas. Merci à Rayan mon collègue de bureau, et qui apporte tout le temps des choses à manger :)

Merci à tous mes amis de l'équipe BioInfo du CHRU de Lille et de la plate-forme de génomique de l'Université de Lille, avec lesquels j'ai passé des moments innombrables et de bons souvenirs.

Grand Merci à mes amis ! Une thèse est une épreuve qui n'est pas très facile moralement, leur support et les moments de joie que nous avons passés ensemble étaient vraiment importants pour moi. Vous êtes ma deuxième famille en France.

Last, but not least, un énorme Merci à mes parents, mes frères et sœurs pour leur soutien inconditionnel. Même si j'étais loin d'eux pendant ces dernières années, ils étaient toujours là, proches de moi.

RÉSUMÉ

L'arrivée des technologies de séquençage d'ADN à haut-débit a représenté une révolution dans le domaine de la génomique personnalisée, en raison de leur résolution et leur faible coût. Toutefois, ces nouvelles technologies présentent un taux d'erreur élevé, qui varie entre 0,1% et 1% pour les séquenceurs de seconde génération. Cette valeur est problématique dans le cadre de la recherche de variants de faible ratio allélique, comme ce qui est observé dans le cas des tumeurs hétérogènes. En effet, un tel taux d'erreur peut mener à des milliers de faux positifs. Chaque région de l'ADN étudié doit donc être séquencée plusieurs fois, et les variants sont alors filtrés en fonction de critères basés sur leur profondeur. Malgré ces filtres, le nombre d'artefacts reste important, montrant la limite des approches conventionnelles et indiquant que certains artefacts de séquençage ne sont pas aléatoires.

Dans le cadre de cette thèse, nous avons développé un algorithme exact de recherche des motifs d'ADN dégénérés sur-représentés en amont des erreurs de séquençage non aléatoires et donc potentiellement liés à leur apparition. Cet algorithme a été mis en œuvre dans un logiciel appelé $DiNAMO$, qui a été testé sur des données de séquençage issues des technologies IonTorrent et Illumina.

Les résultats expérimentaux ont mis en évidence plusieurs motifs, spécifiques à chacune de ces deux technologies. Nous avons ensuite montré que la prise en compte de ces motifs dans l'analyse, réduisait considérablement le taux de faux positifs. $DiNAMO$ peut donc être utilisé en aval de chaque analyse, comme un filtre supplémentaire permettant d'améliorer l'identification des variants, en particulier des variants à faible ratio allélique.

Characterization of non-random sequencing errors

application to mosaicism and heterogeneous tumors

ABSTRACT

The advent of Next Generation DNA Sequencing technologies has revolutionized the field of personalized genomics through their resolution and low cost. However, these new technologies are associated with a relatively high error rate, which varies between 0.1% and 1% for second-generation sequencers. This value is problematic when searching for low allelic ratio variants, as observed in the case of heterogeneous tumors. Indeed, such error rate can lead to thousands of false positives. Each region of the studied DNA must therefore be sequenced several times, and the variants are then filtered according to criteria based on their depth. Despite these filters, the number of errors remains significant, showing the limit of conventional approaches and indicating that some sequencing errors are not random.

In the context of this thesis, we have developed an exact algorithm for over-represented degenerate DNA motifs discovery on the upstream of non-random sequencing errors and thus potentially linked to their appearance. This algorithm was implemented in a software called $DiNAMO$, which was tested on sequencing data from IonTorrent and Illumina technologies.

The experimental results revealed several motifs, specific to each of these two technologies. We then showed that taking these motifs into account in the analysis reduced significantly the false-positive rate. $DiNAMO$ can therefore be used downstream of each analysis, as an additional filter to improve the identification of variants, especially, variants with low allelic ratio.

SOMMAIRE

LISTE DES TABLEAUX

LISTE DES FIGURES

FIGURE **Page**

Liste des abréviations

ADN Acide DésoxyriboNucléique

ARN Acide RiboNucléique

BER Base Excision Repair

BQSR Base Quality Score Recalibration

ChIP-seq Chromatin ImmunoPrecipitation sequencing

CNV Copy Number Variation

ddNTP didésoxyriboNucléoside Tri-Phosphate

dNMP désoxyriboNucléoside Mono-Phosphate

dNTP déoxyriboNucléoside Tri-Phosphate

EM Espérance-Maximisation

ESP Exome Sequencing Project

FFPE Formaldehyde-Fixed and Paraffin Embedded

FT Facteurs de Transcription

GATK Genome Analysis Toolkit

GIAB Genome In A Bottle

HC High-Confidence

HMM Hidden Markov Model

InDel Insertion/Délétion

IUPAC International Union of Pure and Applied Chemistry consortium

MMR MisMatch Repair

NER Nucleotide Excision Repair

ONT Oxford Nanopore Technologies

PCR Polymerase Chain Reaction

PFM Position Frequency Matrix

PSSM Position Specific Scoring Matrix

PWM Position Weight Matrix

RMSE Root-Mean-Square Error

SAM Sequence Alignment/Map

SMRT Single-Molecule Real-Time

SNV Single Nucleotide Variation

SSE Sequence-Specific Errors

SV Structural Variations

TFBS Transcription Factor Binding Site

VAF Variant Allele Frequency

VCF Variant Call Format

INTRODUCTION

Les technologies de séquençage de nouvelle génération (NGS) représentent une avancée majeure en médecine personnalisée, car elles permettent de séquençer simultanément un très grand nombre de régions nucléotidiques et de populations cellulaires distinctes, le tout avec un débit élevé et un coût modeste. Ces performances rendent possibles la recherche des variants génétiques à grande échelle dans le génome humain dans un but clinique. La détection de ces variants permet un meilleur diagnostic de la maladie et une meilleure prise en charge des patients..

L'utilisation du NGS est particulièrement pertinente dans le diagnostic des tumeurs et des maladies génétiques hétérogènes. Dans le cas des tumeurs, le NGS permet de détecter les variants minoritaires et donc d'avoir une vue globale sur l'ensemble des clones cellulaires présents dans la tumeur d'un individu. L'identification des différents variants dans une tumeur hétérogène permet, par exemple, d'identifier une éventuelle résistance au traitement liée à un clone parfois minoritaire. Dans le cas des maladies rares, le NGS, par l'analyse concomitante de multiples gènes, permet d'accélérer le diagnostic. Il peut aussi permettre de détecter des mutations en mosaïque. Pour cette raison, les équipes du CHRU et universitaires de Lille ont investi dans divers séquenceurs de nouvelle génération, qui sont maintenant utilisés en routine clinique pour une meilleure caractérisation des tumeurs et des maladies génétiques.

L'analyse des données de NGS nécessite des étapes de traitement bioinformatique pour identifier les variants de manière fiable. La difficulté vient du fait que le NGS présente un taux d'erreur élevé, entre 0,1% et 1%. À l'échelle d'un génome complet, cela représente des milliers de faux positifs. Plusieurs critères sont utilisés pour filtrer ces erreurs de séquençage, qui exploitent la profondeur de séquençage. Ces filtres sont efficaces contre les erreurs aléatoires, mais ne permettent pas d'éliminer les erreurs

non-aléatoires, qui peuvent être facilement confondues avec des variants génétiques de faible ratio allélique, augmentant ainsi le taux de faux positifs.

Il a été montré que les erreurs de séquençage non aléatoires pouvaient être liées à leur contexte nucléotidique [16, 131, 141]. Les erreurs de séquençage seraient alors causées par la présence au sein de l'ADN de motifs spécifiques, difficiles à séquencer par la ou les technologie(s) de séquençage. L'objectif de cette thèse était d'explorer cette hypothèse, en proposant un cadre bioinformatique complet pour l'identification et la caractérisation des motifs d'ADN liés aux erreurs de séquençage artéfactuelles, ainsi que la prise en compte de ces motifs pour améliorer la recherche de variants.

Ce travail a été effectué au sein de l'équipe « Mucines, différenciation et cancérogenèse épithéliales » dirigée par Dr. Isabelle Van Seuningen, affiliée au laboratoire Inserm UMR-S 1172 (JPARC), en collaboration étroite avec l'équipe Bonsai, spécialisée dans l'algorithmique des séquences biologiques, affiliée au laboratoire CRIStAL (UMR 9189) et dirigée par Dr. Hélène Touzet, ainsi qu'avec la plate-forme de génomique de Lille dirigée par Dr. Martin Figeac.

Le manuscrit est organisé en trois grandes parties.

La première partie aborde les notions de biologie et de bioinformatique nécessaires à la compréhension des travaux. Cette partie est divisée en deux chapitres. Le chapitre 1 présente les notions de base en biologie moléculaire (section 1.1), ainsi que les grands principes de la recherche de variants génomiques par séquençage d'ADN (section 1.2). Le chapitre 2 donne une vue globale sur les méthodes et modèles utilisés pour la recherche des motifs d'ADN, notamment les motifs dégénérés. Ces méthodes sont utilisées actuellement majoritairement pour la recherche de motifs correspondant à des sites de fixation de facteurs de transcription, mais les principes de base seront utiles et utilisés par la suite.

La deuxième partie comporte un chapitre (chapitre 3), qui présente la problématique des erreurs de séquençage non aléatoires et l'état de l'art sur les méthodes utilisées actuellement pour les identifier (section 3.1). Ce chapitre présente également les limites des approches existantes, justifiant les développements proposés dans cette thèse.

La troisième partie, composée de trois chapitres, est consacrée aux travaux de la

thèse. Le chapitre 4 présente le nouvel algorithme exact développé pour la recherche des motifs d'ADN sous la forme d'expression IUPAC. Cet algorithme est implémenté dans un outil, appelé $DiNAMO$, et adapté à une utilisation dans le cadre des erreurs de séquençage non aléatoires. $DiNAMO$ a été comparé à plusieurs outils connus, et les résultats de cette évaluation sont présentés dans le chapitre 5. Ces deux derniers chapitres ont fait l'objet d'une publication dans la revue internationale *BMC Bioinformatics*, sous l'identifiant doi : 10.1186/s12859-018-2215-1. Le chapitre 6 montre ensuite comment l'utilisation de $DiNAMO$ permet d'améliorer le processus d'appel de variants en réduisant le taux de faux positifs.

Les conclusions et perspectives de ce travail sont présentées dans un dernier chapitre, le chapitre 7.

Première partie

Contexte général de la thèse

CHAPITRE 1

GÉNÉRALITÉS SUR LA RECHERCHE DE VARIANTS GÉNOMIQUES

1.1 Notions de génétique moléculaire

1.1.1 L'ADN

L'acide désoxyribonucléique (ADN) est le support de base de l'information génétique de l'individu. Il est formé d'une suite de monomères appelés *nucléotides*. Chaque nucléotide, appelé également désoxyriboNucléoside Mono-Phosphate (dNMP), est composé d'un pentose (sucre à cinq atomes de carbone), le désoxyribose, d'un groupement phosphate et d'une base nucléique (Figure 1.1). Les substrats utilisés pour synthétiser l'ADN sont les nucléosides triphosphates (dNTP, désoxyriboNucleoside Tri-Phosphate).

Il existe quatre bases nucléiques différentes dans l'ADN : l'adénine, la guanine, la cytosine et la thymine. Les deux premières forment le groupe des *purines*, alors que les deux autres font partie du groupe des *pyrimidines* (Figure 1.2).

1.1.1.1 Structure de l'ADN

Les nucléotides sont assemblés sous forme d'un polymère par l'intermédiaire de leur sucre (le désoxyribose) et du groupement phosphate. Une molécule de phosphate relie

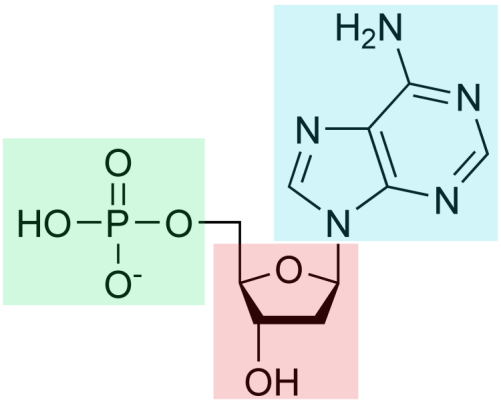


FIGURE 1.1 – La structure d'un nucléotide (dAMP, DésoxyriboAdénosine Mono-Phosphate). En bleu, la base nucléique (Adénine dans cet exemple). En rose, le pentose (désoxyribose). En vert, le groupement phosphate.

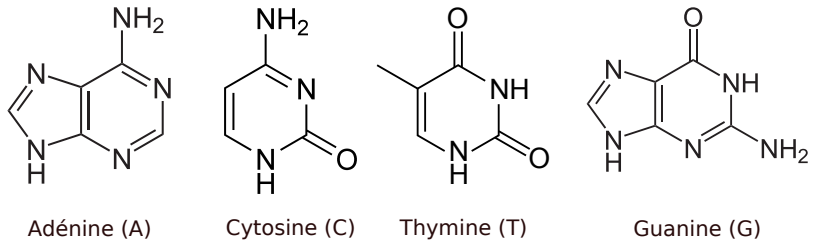


FIGURE 1.2 – Les quatre bases nucléiques de l'ADN.

le carbone en position 3' du premier sucre au carbone en position 5' du sucre suivant, qui est relié à son tour par son carbone en position 3' à un autre nucléotide formant un assemblage orienté. Ainsi, le carbone 5' du premier sucre dans la chaîne (le début de la séquence) reste libre, formant l'extrémité 5'. À l'opposé, le carbone 3' du dernier sucre (la fin de la séquence) est aussi libre, formant l'extrémité 3' (Figure 1.3).

Lors de la synthèse d'une molécule d'ADN, le nucléotide incorporé vient s'ajouter à l'extrémité 3' de la molécule en cours de synthèse. La synthèse d'une molécule d'ADN se fait donc exclusivement dans le sens **5'→3'** [111].

Les molécules d'ADN ont une structure de type hélice double brin (Figure 1.4.a). Les deux brins sont liés grâce aux appariements (liaisons hydrogène) entre les paires de bases nucléiques, dites complémentaires (Figure 1.4.b). L'adénine est appariée à la

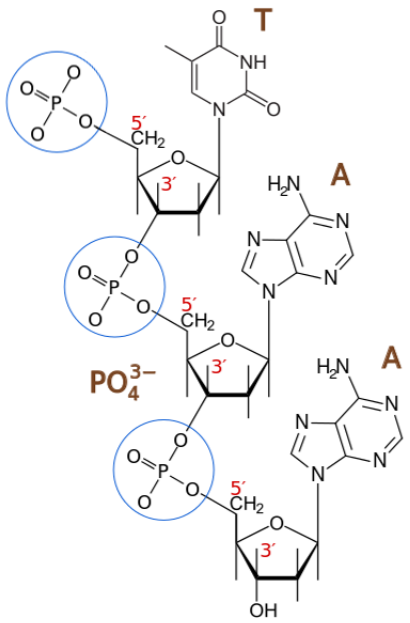


FIGURE 1.3 – Polymère d'ADN. Un exemple d'une séquence TAA. La base T forme l'extrémité 5' (début de séquence), alors que le A, à l'autre extrémité, forme l'extrémité 3' (fin de séquence).

thymine par deux liaisons hydrogène, tandis que la guanine est appariée à la cytosine par trois liaisons hydrogène. Ce type d'appariement entre bases complémentaires est appelé « appariement de Watson-Crick » faisant référence aux deux chercheurs *James Watson* et *Francis Crick* qui ont contribué à la mise en évidence de cette structure d'ADN bicaténaire en 1953 [203].

L'extrémité 5' du premier brin s'apparie à l'extrémité 3' du brin complémentaire. Les deux brins anti-parallèles sont dits « sens » et « anti-sens ». Le deuxième brin présente la séquence reverse-complémentaire du premier. Étant donnée la nature double brin de l'ADN, sa longueur est comptée en paires de bases (pb).

Les molécules d'ADN double brin sont associés à des protéines permettant leur compaction et leur organisation sous forme de chromosomes. Le nombre, la taille et la forme des chromosomes diffèrent d'une espèce à une autre. On compte 46 chromosomes pour l'espèce humaine contenant environ 6,4 milliards de paires de bases.

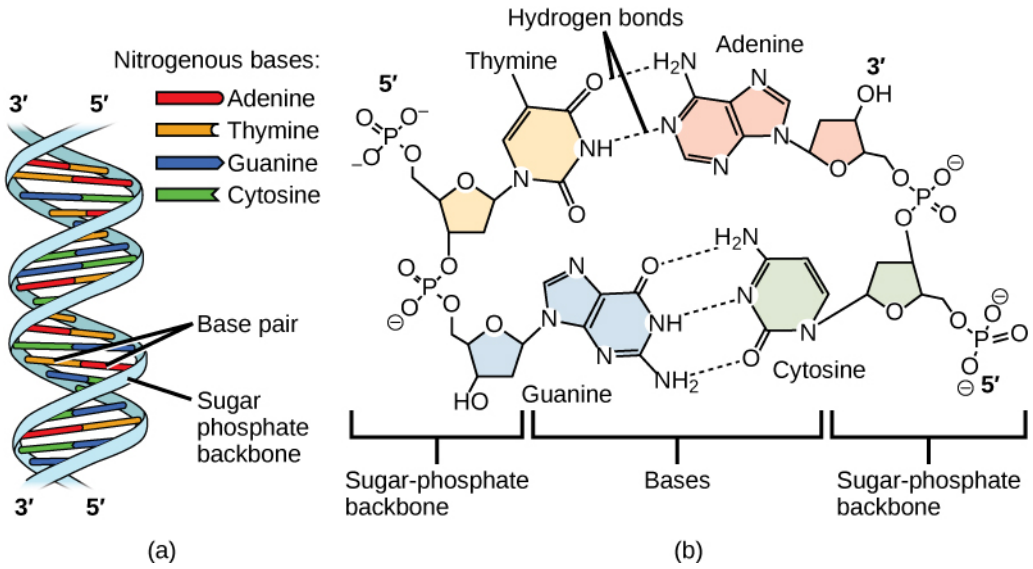


FIGURE 1.4 – Structure de la double hélice d'ADN. Source de la figure : [1]

1.1.1.2 Les gènes

L'ADN porte l'information génétique de l'individu. Cette information est distribuée sur différentes régions de l'ADN en différentes entités fonctionnelles, appelées gènes, capables de délivrer l'information nécessaire à la production d'une molécule fonctionnelle, acide ribonucléique (ARN) ou protéine. Chaque gène est lui même organisé en deux types de régions : celles contenant l'information à exprimer et les régions régulatrices, dont la région promotrice située en 5' du gène, qui contrôlent le niveau d'expression du gène.

Certains gènes sont traduits sous forme de protéine. La synthèse directe des protéines à partir des séquences d'ADN double brin n'est pas possible et une étape intermédiaire est nécessaire. Cette étape est appelée la transcription. La transcription est le mécanisme qui permet de copier ou transcrire l'information codée dans le gène sous forme d'une autre molécule apparentée mais simple brin appelée acide ribonucléique (ARN) [79].

Un même gène peut exister en différentes versions, distinguées par des variations au sein de leur séquence nucléotidique. Chaque version est appelée allèle. Par exemple, le gène codant le groupe sanguin ABO chez l'humain existe sous forme de plusieurs allèles différents. Si les deux chromosomes d'une même paire portent le même allèle, on

dit que le gène est homozygote, tandis que si les deux allèles sont différents, on dit qu'il est hétérozygote.

1.1.2 L'ARN

L'acide ribonucléique (ARN) est un polymère qui est très proche chimiquement de l'ADN. L'ARN diffère principalement de l'ADN par l'absence de la thymine, qui est remplacée par l'uracile, par le sucre du type ribose (désoxyribose pour l'ADN), ainsi que par sa structure monocaténaire (un seul brin). L'ARN est obtenu à partir de l'ADN par le processus de transcription, dans lequel des enzymes appelées *ARN polymérases* effectuent la copie de l'ADN vers l'ARN.

1.1.2.1 Familles d'ARN

Il existe différents types d'ARN. Seuls les acteurs principaux intervenant dans le processus de synthèse des protéines à partir de l'ADN sont décrits.

1.1.2.1.1 L'ARN messager (ARNm) est le résultat de la transcription d'un gène. C'est une copie du contenu du gène dont il est issu. Cet ARN est produit dans le noyau de la cellule, puis transféré vers le cytoplasme pour être traduit en protéines. Cet ARN messager subit une suite d'étapes de modifications, notamment l'épissage qui consiste à éliminer les introns, avant de devenir un ARNm mature. L'ARNm est un ARN informatif qui sert d'intermédiaire dans le processus de la traduction de l'ADN d'un gène en protéine. Il est le seul ARN à être traduit [79].

Tous les autres types d'ARN décrits par la suite, sont des ARN qui ne sont jamais traduits en protéines.

1.1.2.1.2 Les ARN ribosomaux (ARNr) participent à la constitution du ribosome, qui est un complexe ARN-protéines de grande taille. Les ribosomes sont les éléments chargés de lire la séquence d'ARNm et de la traduire en séquence protéique.

1.1.2.1.3 Les ARN de transfert (ARNt) sont chargés du transfert des acides aminés (monomères de base d'une protéine) vers le ribosome. Ces ARN ont une structure dite en « feuille de trèfle » avec 3 boucles et un bras accepteur, site d'attachement de l'acide aminé. La séquence d'ARN est lue par triplets de nucléotides, appelés codons. Chaque

codon code pour un seul acide aminé (Table 1.5). Il existe un ARN de transfert pour chacun des acides aminés.

1.1.3 Les protéines

Les protéines sont des polymères d'acides aminés obtenus par la traduction des molécules d'ARNm. Les acides aminés sont liés entre eux par une liaison peptidique. Chez l'Homme, il existe 20 acides aminés différents (Table 1.1).

Acide Aminé	**Abréviation**	**Acide Aminé**	**Abréviation**
Glycine	Gly (G)	Thréonine	Thr (T)
Alanine	Ala (A)	Cystéine	Cys (C)
Valine	Val (V)	Asparagine	Asn (N)
Leucine	Leu (L)	Glutamine	Gln (Q)
Isoleucine	Ile (I)	Tyrosine	Tyr (Y)
Méthionine	Met (M)	Aspartate	Asp (D)
Proline	Pro (P)	Glutamate	Glu (E)
Phénilalanine	Phe (F)	Lysine	Lys (K)
Tryptophane	Trp (W)	Arginine	Arg (R)
Sérine	Ser (S)	Histidine	His (H)

TABLE 1.1 – Les 20 acides aminés avec leur abréviation.

Le processus de traduction d'ARNm en protéine nécessite de nombreux acteurs. Brièvement, le ribosome se fixe à l'ARNm et le lit codon par codon (triplet de nucléotides) à partir du codon initiateur qui est très majoritairement AUG et qui code une méthionine. L'acide aminé correspondant à chaque codon est véhiculé par l'ARNt, qui le livre au ribosome, pour y être lié à l'acide aminé précédent par une liaison peptidique. Le processus de lecture continue jusqu'à la rencontre d'un codon *stop* (UAG, UAA ou UGA) qui marque la fin de la traduction. Plusieurs codons peuvent coder pour le même acide aminé (il existe 64 triplets de nucléotides pour 20 acides aminés) ; on parle de la dégénérescence du code génétique (Table 1.5).

Les protéines assurent des fonctions très diverses chez les organismes vivants. Elles ont par exemple un rôle structural (peau, ongles, poils,...), dans la réponse immunitaire (les immunoglobulines), enzymatique, hormonal et autres. Dans le paragraphe suivant de cette thèse, nous nous intéresserons plus spécifiquement à une classe particulière de protéines ayant un rôle dans la régulation de l'expression des gènes, les facteurs de transcription.

1re base	2e base								3e base
	U		C		A		G		
U	UUU	F Phe	UCU	S Ser	UAU	Y Tyr	UGU	C Cys	U
	UUC	F Phe	UCC	S Ser	UAC	Y Tyr	UGC	C Cys	C
	UUA	L Leu	UCA	S Ser	UAA	Stop	UGA	Stop	A
	UUG	L Leu	UCG	S Ser	UAG	Stop	UGG	W Trp	G
C	CUU	L Leu	CCU	P Pro	CAU	H His	CGU	R Arg	U
	CUC	L Leu	CCC	P Pro	CAC	H His	CGC	R Arg	C
	CUA	L Leu	CCA	P Pro	CAA	Q Gln	CGA	R Arg	A
	CUG	L Leu	CCG	P Pro	CAG	Q Gln	CGG	R Arg	G
A	AUU	I Ile	ACU	T Thr	AAU	N Asn	AGU	S Ser	U
	AUC	I Ile	ACC	T Thr	AAC	N Asn	AGC	S Ser	C
	AUA	I Ile	ACA	T Thr	AAA	K Lys	AGA	R Arg	A
	AUG	M Met & initiation	ACG	T Thr	AAG	K Lys	AGG	R Arg	G
G	GUU	V Val	GCU	A Ala	GAU	D Asp	GGU	G Gly	U
	GUC	V Val	GCC	A Ala	GAC	D Asp	GGC	G Gly	C
	GUA	V Val	GCA	A Ala	GAA	E Glu	GGA	G Gly	A
	GUG	V Val	GCG	A Ala	GAG	E Glu	GGG	G Gly	G

FIGURE 1.5 – Table des codons ARN. Figure adaptée de : [2]

1.1.3.1 Facteurs de transcription

Les facteurs de transcription (FT) sont des protéines régulatrices qui contrôlent l'initiation et la vitesse de la transcription des gènes (ADN → ARN) en se fixant sur l'ADN (Figure 1.6). Les facteurs de transcription sont répartis en deux classes [96] :

— Les FT généraux : qui s'associent avec l'ARN polymérase pour composer la machinerie transcriptionnelle de base [148].

— Les FT spécifiques de séquences : qui reconnaissent des séquences d'ADN spécifiques, auxquelles ils se fixent pour réguler la transcription du gène. Ils peuvent fonctionner comme activateurs, en stimulant la transcription du gène ou bien comme répresseurs en inhibant la transcription. Les facteurs de transcription spécifiques sont divisés en deux groupes [25] :

- Les FT dont la régulation est liée au développement. Ils sont spécifiques d'un type de cellule, et interviennent lors de la différenciation cellulaire. Par exemple, le FT GATA1 qui est un facteur de transcription essentiel à la différenciation des cellules érythroïdes [74].

- Les FT spécifiques dont la régulation est liée à un signal biologique (hormonal par exemple) ou à un récepteur cellulaire. Par exemple, le FT STAT3 qui est activé pour un récepteur membranaire associé à une enzyme de type Janus kinase (JAK) quand une cellule entre en contact avec des cytokines (notamment la cytokine IL22) et des facteurs de croissance (comme EGFR, CSF1R, PDGFR) [100].

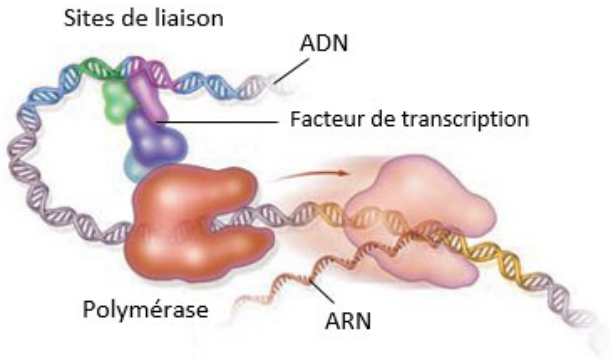


FIGURE 1.6 – La régulation de l'expression du gène par les facteurs de transcription. Source de la figure : [3]

Les FT possèdent un domaine de fixation à l'ADN qui reconnaît et se fixe au niveau de séquences spécifiques, situées dans la région régulatrice du gène, et appelées sites de fixation des facteurs de transcription (TFBS, *Transcription Factor Binding Site*). La taille de ces sites est relativement petite (généralement entre 6 et 10 pb). Cette interaction ADN-protéine au niveau des TFBS est spécifique mais n'est pas stricte. Cela signifie qu'un même facteur de transcription peut reconnaître et se fixer sur plusieurs sites qui diffèrent de une à quelques bases (Figure 1.7).

Plusieurs FT peuvent réguler un même gène. Ces FT peuvent interagir entre eux à travers des domaines de fixation protéine-protéine (Figure 1.8).

```
GGAC   A GAT AAAC ACTT
GGAC GAC GAT AAGG ACTT
GGAC ACA GAT AAC  ACTT
GGAC TCT GAT AACA TCTT
GGAC GTT GAT AAGT ACTT
GGAC GAA GAT AATC ACTT
GGAC  CA GAT AAC  ACTT
GGAC CGT GAT AAAC ACTT
GGAC ACC GAT AACA ACTT
GGAC GAT GAT AACA ACTT
GGAT AAA GAT AACA ACTT
GGAC GCA GAT AACA ACTT
GGAC GCA GAT ATCA ACTT
```

FIGURE 1.7 – Un ensemble de sites de fixation sur l'ADN reconnus par le FT GATA1 [103]. Au centre des séquences, en encadré, une sous-séquence commune et conservée (GAT), entourée par des séquences plus au moins variables.

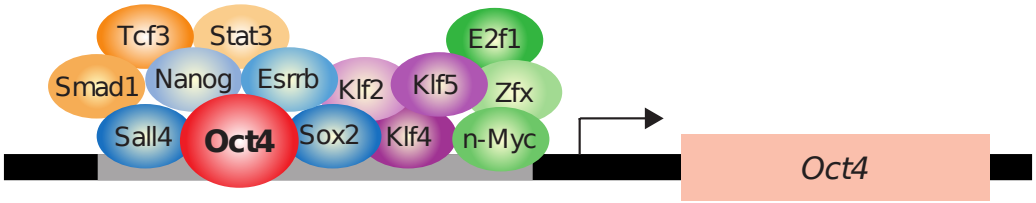


FIGURE 1.8 – Un ensemble de facteurs de transcription régulant le gène *OCT4*. La zone rose représente la séquence codante du gène (à transcrire). La zone grise représente la zone régulatrice qui contient les sites de fixation des facteurs de transcription. La flèche indique le sens de transcription (5'→3') et le site d'initiation de la transcription. Figure adaptée de : [145]

1.2 Les variants génétiques

Les variants génétiques sont des modifications de la séquence d'ADN et donc de l'information contenue dans cette séquence. La modification de la base est appelée « mutation ». Les conséquences de ces variations dépendent de la partie du génome touchée.

1.2.1 Origine des variants

L'apparition des variants peut être spontanée ou bien induite. Les mutations spontanées peuvent survenir lors du processus de réplication d'ADN [78]. En effet, l'ADN

polymérase (l'enzyme qui synthétise la nouvelle molécule d'ADN) peut incorporer une mauvaise base pendant la réplication de l'ADN. Le taux d'erreur de cette enzyme est de l'ordre de 10^{-5} (une erreur sur cent mille bases répliquées) [155]. Il existe d'autres types de mutations spontanées, qui sont dues au métabolisme cellulaire et à la température corporelle, comme la dépurination de l'ADN par exemple. La dépurination est un processus dans lequel l'ADN perd des bases de type purines par hydrolyse [72].

Les mutations induites résultent d'une interaction entre l'ADN et un agent extérieur, physique ou chimique, appelé mutagène [40]. Les facteurs environnementaux sont des facteurs majeurs dans l'apparition des variants. L'exemple le plus connu d'agent mutagène physique est la lumière ultraviolette (UV), qui induit des mésappariements de type C-T dans l'ADN [178]. Les produits chimiques peuvent aussi induire des modifications de nucléotides ou bien des mauvais appariements de bases. Par exemple, le benzo[a]pyrene, une substance contenue dans la cigarette, est connu comme étant une substance mutagène qui induit l'apparition de lésions au niveau des guanines notamment dans des gènes suppresseurs de tumeur (tel que le *TP53*), augmentant ainsi le risque d'apparition du cancer du poumon [52].

Il existe de nombreux mécanismes de détection des mésappariements et de correction d'erreurs, tels que les systèmes de réparation MMR (*MisMatch Repair*), NER (*Nucleotide Excision Repair*) et BER (*Base Excision Repair*) [82]. Malgré cette multitude de mécanismes de réparation, quelques erreurs peuvent échapper et être maintenues dans les cellules filles [152].

1.2.2 Types de variants et conséquences

Il existe plusieurs types de variants, distingués notamment selon la taille de la région affectée.

1.2.2.1 Variants ponctuels

les variants ponctuels correspondent à des substitutions, délétions ou insertions ne touchant qu'un seul nucléotide ou un petit nombre de nucléotides.

— **Substitutions (SNV, *Single Nucleotide Variation*)** Ces variants correspondent au remplacement d'une base par une autre. Elles sont divisées en deux classes, les

transitions et les transversions. Une transition est un remplacement d'une base par une autre de la même catégorie chimique (purine par une autre purine, par exemple A→G, ou bien pyrimidine par une autre pyrimidine, par exemple T→C). Par contre, la transversion est un remplacement d'une base par une autre d'une catégorie chimique différente (purine par pyrimidine ou *vice versa*).

Les conséquences de ce type de variants dépendent de deux facteurs principaux, la position dans le gène (région codante ou non codante), et le type de la base remplaçante (Figure 1.9).

- **Substitution silencieuse ou synonyme ou même-sens :** la variation modifie la séquence d'un codon sans en modifier la signification, ce qui est possible grâce à la dégénérescence du code génétique (Table 1.5).
- **Substitution faux-sens :** la variation modifie la séquence d'un codon qui code un acide aminé différent.
- **Substitution non-sens :** le codon est remplacé par un codon *stop*. Ces variants conduisent à la terminaison prématurée de la traduction, et à la production d'une protéine tronquée.

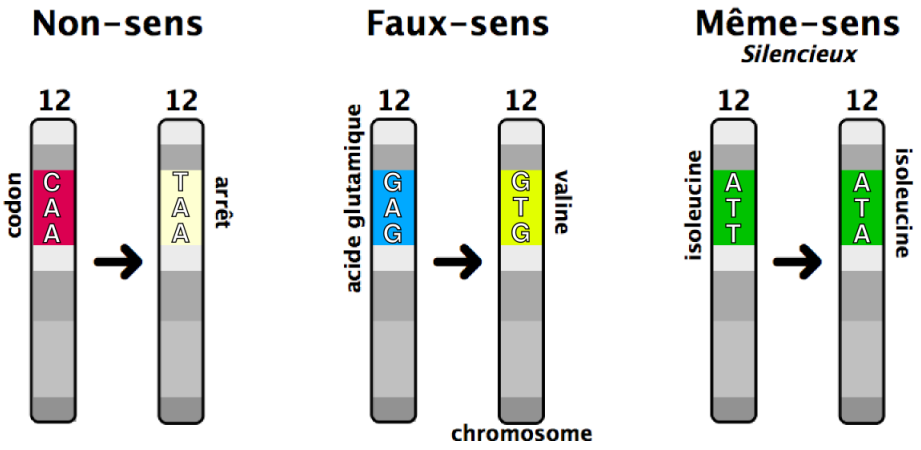


FIGURE 1.9 – Les différentes conséquences d'un variant ponctuel. Non-sens : La mutation de CAA (codant la Glutamine) en TAA (codon *Stop*) entraîne l'arrêt prématuré de la traduction de la protéine. Faux-sens : La mutation de CAA (codant l'Acide glutamique) en TAA (codant la Valine) entraîne un changement d'acide aminé dans la protéine. Même-sens : Les 2 codons ATT et ATA codent le même acide aminé, l'Isoleucine. Source de la figure : [4]

— **Insertions/Délétions (InDel) :** se sont des délétions ou des insertions d'une ou de plusieurs bases. Ce type de variants induit un décalage du cadre de lecture à

partir de la position du variant si la taille du fragment inséré/délété n'est pas un multiple de trois, entraînant souvent l'apparition d'un codon *stop* prématuré [79].

Les mutations peuvent survenir aussi en dehors de la région codante. Ces mutations peuvent par exemple altérer le niveau d'expression du gène si elles touchent la région promotrice ou avoir des conséquences sur le mécanisme de maturation des ARN, si elles touchent par exemple les sites d'épissage dans les introns. Les conséquences des variants introniques sont moins étudiées et plus difficiles à prédire que celles des variants exoniques.

1.2.2.2 Variants structuraux

Les variants structuraux (SV, *Structural Variations*) sont des variations qui affectent une grande région génique ou chromosomique. Ils sont définis comme étant des altérations génomiques impliquant des segments d'ADN ayant généralement une taille plus grande que 1 kb [15, 65, 189]. Il en existe plusieurs types (Figure 1.10) :

— **Les variations du nombre de copies (CNV, *Copy Number Variation*) :** elles sont définies comme étant des segments d'ADN qui sont présents en un nombre de copies différent de celui du génome de référence [161]. Il peut s'agir de duplication ou de délétion.

— **Les inversions :** l'orientation d'un segment d'ADN est inversée par rapport au reste du chromosome.

— **Les translocations :** ce sont des réarrangements touchant un ou plusieurs chromosomes et correspondant au déplacement d'un fragment de chromosome plus ou moins long. Il peut s'agir d'échanges réciproques de matériel chromosomique entre des chromosomes différents, qui peuvent être équilibrés. Les translocations peuvent être sans conséquence notamment si elles n'entraînent pas d'interruption de gènes. Si, par contre, le point de cassure d'une translocation se situe à l'intérieur d'un gène, la translocation provoque une interruption de ce gène et aura un effet délétère.

— **Les insertions :** Les insertions correspondent à l'introduction d'une séquence qui peut être une séquence endogène mobile, ou une séquence exogène virale.

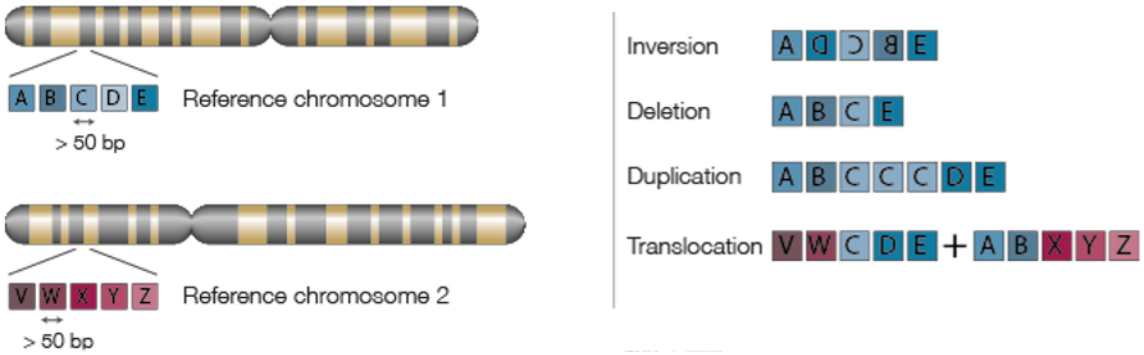


FIGURE 1.10 – Différents types de variants structuraux. Figure adaptée de : [5]

1.2.3 Variants germinaux et somatiques

Les variants peuvent survenir dans les cellules germinales ou dans les cellules somatiques. On parle alors respectivement de variants germinaux et de variants somatiques (Figure 1.12) [79].

1.2.3.1 Variants germinaux

Un variant germinal est un variant présent dans les cellules germinales (ovocyte ou spermatozoïde). De ce fait, il sera transmis à la descendance et l'embryon sera porteur du variant dans toutes ses cellules. Il s'agit alors pour l'embryon d'un variant constitutionnel.

1.2.3.2 Variants somatiques

Un variant somatique est un variant qui survient dans une cellule somatique (non germinale) et par définition n'est pas transmis à la descendance. Cette cellule mutée donnera naissance à une population de cellules porteuses du même variant si le variant est compatible avec la survie de la cellule. Cette population de cellules identiques, issue d'une même cellule d'origine, est appelée clone. Selon le stade d'apparition du variant somatique lors du développement et la modification apportée au fonctionnement de la cellule par la mutation (avantage prolifératif notamment), la proportion des cellules mutantes (taille du clone) peut être plus ou moins importante chez l'individu (Figure 1.11).

Les régions (ou les gènes) touchées par les variants se trouvent alors sous différentes

formes alléliques au sein de la population cellulaire. La fréquence d'un allèle au sein de la population est appelée ratio allélique.

On distinguera plus particulièrement, dans le cadre de cette thèse, deux types de variants somatiques, les variants en mosaïque et les variants observés dans les tumeurs :

— Mosaïcisme : si le variant survient lors des premières divisions post-zygotiques, après la fécondation, une large proportion de cellules pourra être touchée (entraînant la présence du variant au sein de plusieurs organes), formant ainsi des mosaïques. Le mosaïcisme est défini comme une co-existence d'au moins deux populations de cellules génétiquement différentes au sein d'un même individu [17, 67]. Il est possible dans certains cas de mosaïques que le variant somatique survenu lors du développement post-zygotique touche une future cellule germinale et soit ainsi transmis à la descendance. Un tel variant devient alors un variant germinal dans la descendance.

— Tumeurs hétérogènes : les variants somatiques survenant dans un tissu donné peuvent être à l'origine du développement d'une tumeur s'ils touchent par exemple des gènes responsables de la régulation du cycle cellulaire, dits proto-oncogènes, ou des gènes suppresseurs de tumeurs [67, 79]. Le taux de mutation dans une cellule tumorale au cours des divisions successives, est généralement plus élevé que dans une cellule normale. Il passe de l'ordre de 10^{-8} variants/pb/génération [42] à environ 10^{-4} variants/pb/génération [197]. Cela conduit à l'apparition de plusieurs clones cellulaires distincts à l'intérieur d'une tumeur. La proportion de chaque clone dépend du stade d'apparition du variant dans la tumeur et de l'avantage prolifératif qu'il confère (résistance à l'apoptose ou encore par exemple résistance aux traitements) (Figure 1.11).

1.2.4 Recherche de variants par séquençage

La détection des variants génomiques représente une tâche essentielle de la génétique moléculaire, pour la recherche fondamentale comme pour les applications médicales.

La détection des variants permet le diagnostic moléculaire de certaines maladies, c'est-à-dire l'identification du gène et de la mutation responsables de la maladie. Cela

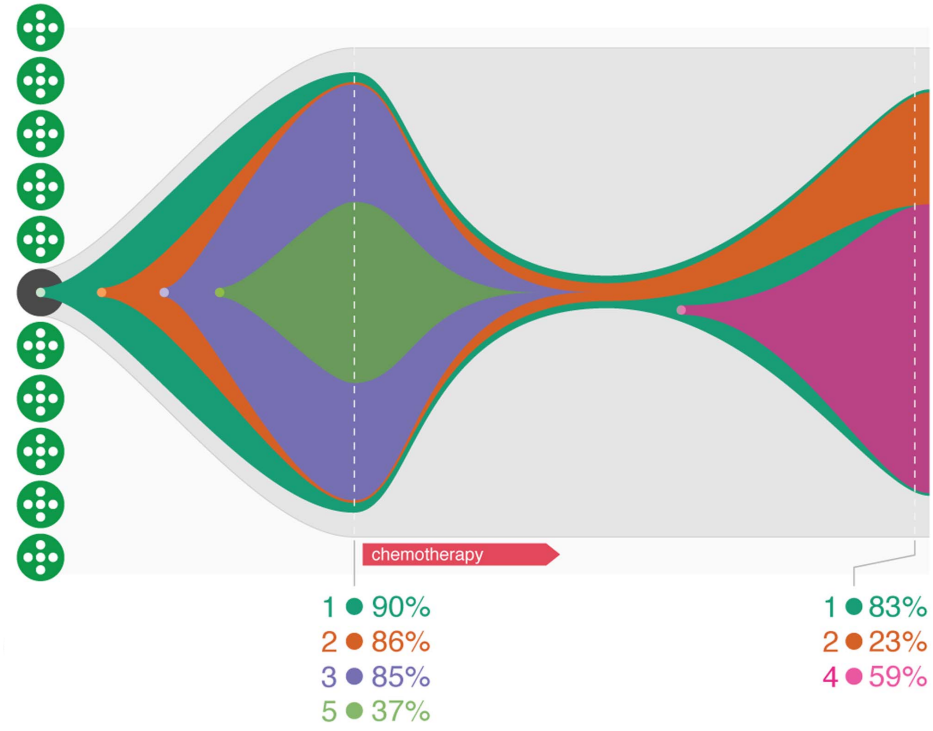


FIGURE 1.11 – Schéma de l'évolution clonale d'une tumeur hétérogène, à partir d'une seule cellule souche hématopoïétique, montrant le pourcentage de cellules appartenant à chaque clone à différents stades (avant et après traitement de la tumeur par chimiothérapie). Figure adaptée de : [134]

peut aider à poser un diagnostic et permettre une prise en charge adaptée du patient et de sa famille. À titre d'exemple, le syndrome de Lynch est responsable de 3 à 5% des cancers colorectaux [70]. Ce syndrome est dû à un variant constitutionnel d'un des gènes MMR (*MisMatch Repair*) intervenant dans la réparation des mésappariements de l'ADN. Le risque de développer un cancer colorectal avant l'âge de 70 ans chez les personnes présentant le syndrome de Lynch varie entre 35% et 50% [56]. La détection d'un variant délétère dans un gène MMR permet de poser le diagnostic de syndrome de Lynch et de proposer des stratégies de prise en charge adaptées aux patients atteints, comme la surveillance par coloscopie qui permet de réduire le risque de cancer colorectal de 63%, et également de réduire la mortalité [22, 93]. Par ailleurs, en cas de cancer, la recherche de certaines mutations dans la tumeur peut permettre d'orienter le traitement. À titre d'exemple, l'identification d'une mutation activatrice du proto-oncogène *BRAF* dans un mélanome, permet de traiter le patient par un inhibiteur spécifique de la protéine BRAF mutée [35].

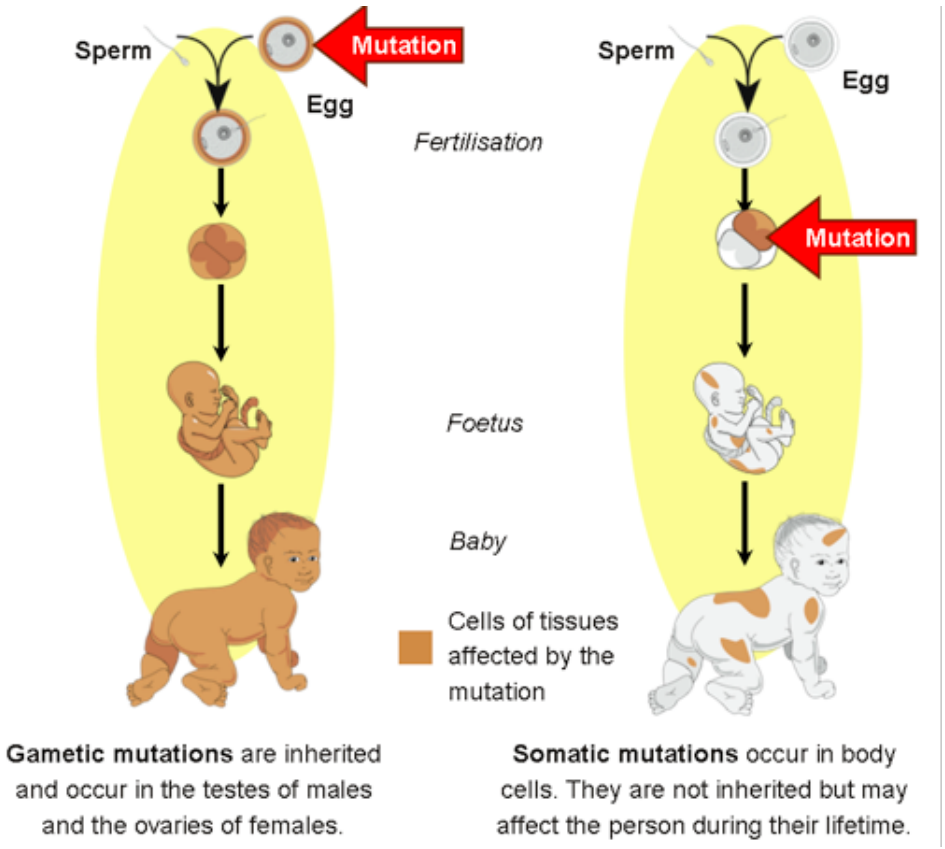


FIGURE 1.12 – Différence entre un variant somatique et un variant germinal. À gauche, un variant germinal qui survient dans l'ovule et qui est transmis à l'enfant. À droite, un variant somatique qui survient dans les stades précoces de la formation du fœtus ce qui affecte plusieurs tissus chez l'enfant (mosaïque). Source de la figure : [6]

Pour rechercher des variants sans aucun a priori, on a recours aux technologies de séquençage. Le séquençage d'ADN consiste à déterminer la succession de nucléotides qui composent la séquence d'ADN. Connaître la séquence d'ADN permet donc de détecter des variations par rapport au génome de référence. Un génome de référence est un génome modèle intégrant les informations les plus à jour que nous connaissons sur sa séquence.

1.2.4.1 Séquençage Sanger

L'une des premières méthodes utilisées pour séquencer l'ADN est la méthode *Sanger* [169]. Cette méthode a été mise en point par le chercheur britannique Frederick Sanger en 1977, et elle est aujourd'hui encore considérée comme la méthode de référence.

Cette technique est basée sur la synthèse du brin d'ADN complémentaire à partir d'un brin d'ADN matrice. Elle repose sur l'utilisation de didésoxyribonucléosides triphosphates (ddNTP), qui sont des nucléotides dont le groupement hydroxyle (OH) en 3' du désoxy-ribose est remplacé par un H. De ce fait, suite à l'incorporation d'un ddNTP dans la séquence en cours de synthèse, le nucléotide suivant ne peut plus s'incorporer (terminaison de chaîne). En général, en utilise des ddNTP marqués par des fluorochromes différents (ce qui permet de distinguer les quatre ddNTPs : ddATP, ddCTP, ddGTP, ddTTP), mélangés en petites quantités avec les autres dNTPs (dATP, dCTP, dGTP, dTTP). Etant donné que l'incorporation des ddNTPs dans la séquence en cours de synthèse est aléatoire, les fragments néosynthétisés sont de différentes tailles et migreront plus ou moins vite par électrophorèse sur gel capillaire, puisque la vitesse de migration des fragments dans le gel dépend de leur taille. Les fragments les plus courts sortent du gel en premier, tandis que les fragments les plus longs sortent en dernier. Lorsqu'un fragment sort du gel, le fluorochrome est excité par un laser, ce qui permet de détecter la longueur d'onde d'émission. Ainsi, à partir de la suite de couleurs enregistrées par le détecteur, la séquence du fragment d'ADN originale peut être construite (Figure 1.13). Les données enregistrées par le détecteur consistent en une série de pics d'intensité de fluorescence, comme indiqué dans l'électrophorégramme de la figure 1.14.

Dans un électrophorégramme, un pic unique indique une base non mutée ou bien un variant homozygote (si différent de la séquence de référence), alors que dans le cas d'un variant hétérozygote, on obtient 2 pics de couleurs différentes à la même position (Figure 1.14).

La technique de Sanger permet de séquencer de façon fiable des petits fragments d'ADN (jusqu'à 400 à 900 pb), avec un faible débit (la machine automatisée la plus récente ABI 3730XL est capable de séquencer jusqu'à 2100 kb par jour). À titre d'exemple, la première version de la séquence du génome humain a nécessité onze ans de travail collaboratif au sein d'un consortium international, composé d'une vingtaine de centres de recherche, et a coûté plus de 3 milliards de dollars [43].

Le débit de cette technologie de séquençage, ainsi que son coût élevé, limite son utilisation à la détection de variants dans des régions de petite taille (~1 kb) ou à la validation de variants détectés par une autre technique. De plus, il est impossible de détecter des variants minoritaires qui ont un faible ratio allélique, comme cela peut être

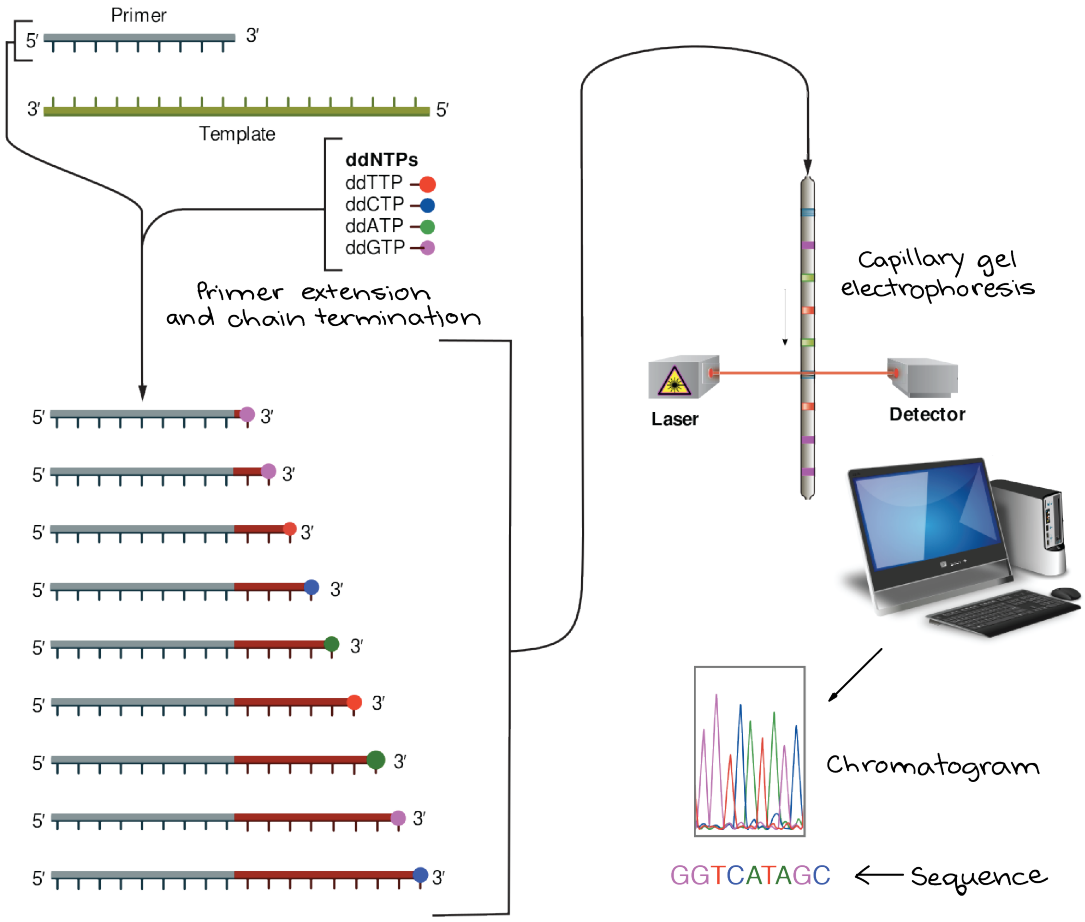


FIGURE 1.13 – Séquençage Sanger. Les fragments d'ADN synthétisés sont séparés en fonction de leur taille par électrophorèse sur gel et leur terminaison fluorescente est lue par le lecteur laser qui va identifier le ddNTP correspondant. Source de la figure : [7]

le cas pour les variants tumoraux, car le seuil de détection d'un variant par séquençage Sanger est d'environ 15-20% d'allèle muté [105].

1.2.4.2 Séquençage haut-débit

Apparus en 2004, les séquenceurs dits de nouvelle génération (*NGS, Next Generation Sequencing*) ont permis une révolution technologique en augmentant significativement le débit d'analyse des séquences d'ADN. Ces machines ont permis le passage d'une échelle de quelques milliers de paires de bases à plusieurs milliards, tout en réduisant d'une manière drastique le coût de séquençage (Figure 1.15).

Les séquenceurs disponibles aujourd'hui peuvent être séparés en deux groupes. Les

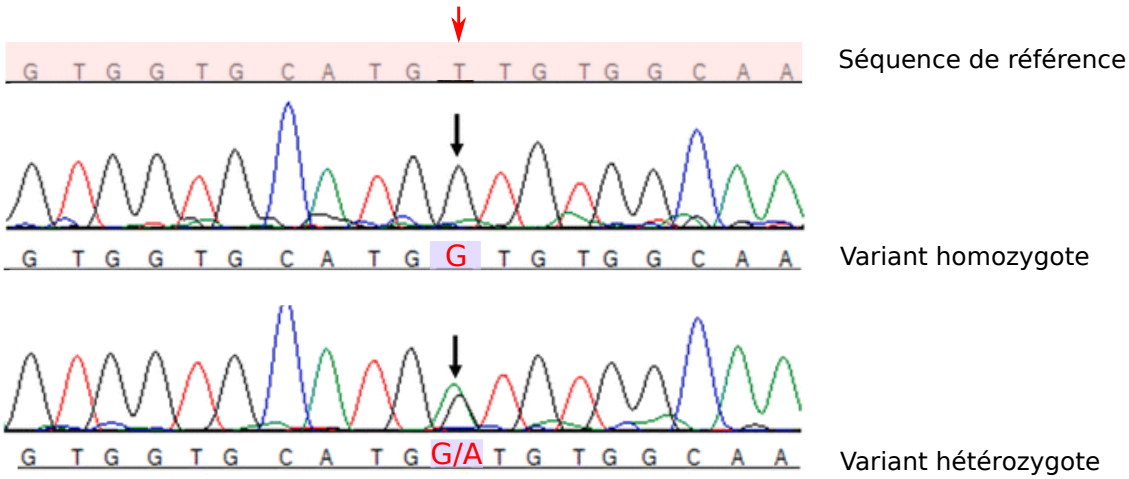


FIGURE 1.14 – Exemple d'un SNV potentiel détecté par séquençage Sanger. Sur le premier électrophorégramme, au niveau de la flèche noire, on remarque un unique pic qui correspond à un variant homozygote $T \rightarrow G$. Tandis que dans le deuxième électrophorégramme, il y a deux pics à la même position, qui correspondent à un variant hétérozygote $T \rightarrow A/G$. Figure adaptée de : [99]

séquenceurs dits de 2ème génération, capables de séquencer des fragments d'ADN courts (50 à 400 pb), tels que les séquenceurs Illumina et IonTorrent Thermofisher, et les séquenceurs de 3ème génération, capables de séquencer des longs fragments d'ADN (taille allant jusqu'à 200kb), tels que les séquenceurs PacBio (Pacific Biosciences) et Nanopore. Les séquenceurs de 3ème génération se distinguent de ceux de la 2ème génération aussi par l'absence de l'étape d'amplification par PCR (*Single molecule real time sequencing, SMRT*)

Les séquenceurs à haut-débit possèdent des caractéristiques différentes. Ils diffèrent principalement par leur débit, par la taille des fragments séquençés et par leur taux d'erreur. Ce dernier varie entre 0.1% et 1% pour les séquenceurs de 2ème génération, et peut monter jusqu'à 13% pour les séquenceurs de 3ème génération (Table 1.2).

Grâce au débit élevé, la recherche de variants génomiques est devenue possible au niveau d'un panel constitué de centaines de gènes, voir d'un exome ou d'un génome complet. Le séquençage haut-débit permet également de détecter plusieurs types de variants et pas uniquement des variants de petites tailles (SNV et InDel). Avec la possibilité de séquençer un génome entier par les séquenceurs NGS, il est possible, à l'inverse

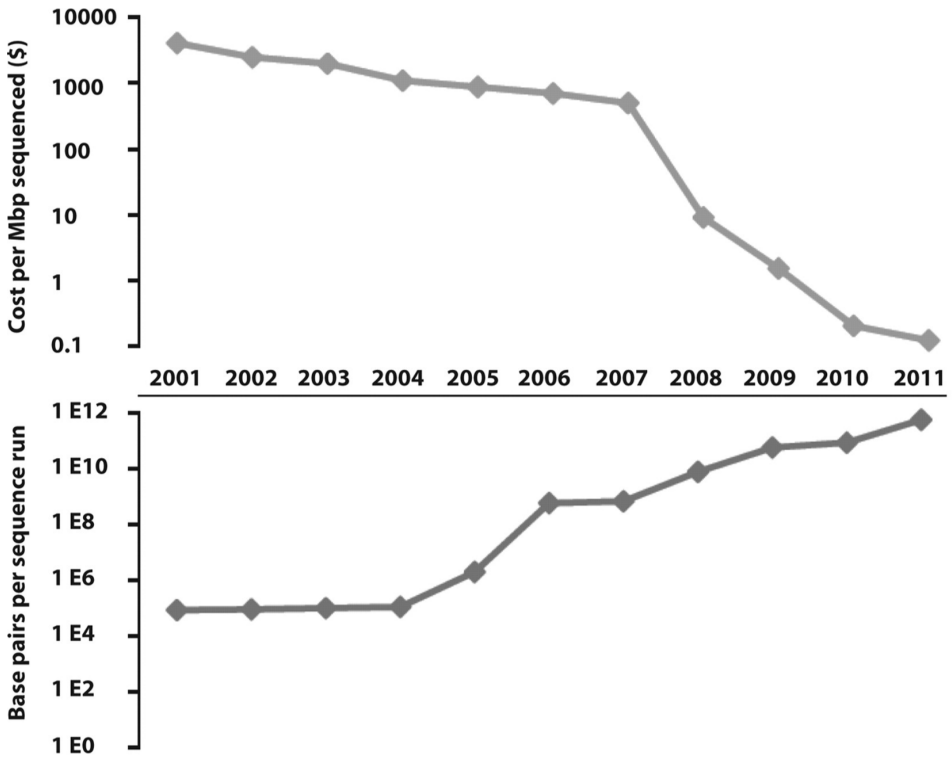


FIGURE 1.15 – Débit et coût de séquençage depuis l'achèvement du projet sur le génome humain. Figure adaptée de : [54]

du séquençage classique Sanger, de détecter des variants de grandes tailles comme les variants structuraux (SV). Il est possible, par exemple, de détecter les CNV en analysant la profondeur de séquençage à chaque position du génome. Ainsi, une profondeur plus élevée que la profondeur de séquençage moyenne dans certaines régions indiquera une région qui est dupliquée [192].

Dans le cadre de cette thèse, nous nous intéressons au séquençage NGS de 2ème génération. Les différentes étapes de cette technique, de la préparation des échantillons, au séquençage et au traitement des données avec différents logiciels bio-informatiques, sont décrites ci-dessous :

	Technologie de séquençage	Longueur des *reads*	Débit d'un *run*	Taux d'erreurs	Durée d'un *run*	Coût par Gb
ABI 3730xl	Séquençage par terminaison de chaîne	400 à 900 pb	2100 Kb	0.001%	1-3 hr	2 400 000 $
Illumina MiSeq V3	Séquençage par synthèse	300 pb (PE)	15 Gb	0.1%, substitution	21-56 hr	100 $
Ion Proton	Séquençage par synthèse	300 pb (SE)	10 Gb	1%, InDel	2-4 hr	80 $
Oxford Nanopore MinION	ONT	200 Kb	1.5 Gb	~12%, InDel	48 hr	750 $
Pacific Biosciences RSII	SMRT	20 Kb	1 Gb	13%, en lecture unique	4 hr	1 000 $

TABLE 1.2 – Comparaison de quelques séquenceurs de différentes générations [77]. En rose, un séquenceurs Sanger. En orange, deux séquenceurs NGS de seconde génération. En bleu, deux séquenceurs de troisième génération. SE : *Single-End*. PE : *Paired-End*. SMRT : *Single-Molecule Real-Time*.

1.2.4.2.1 Préparation de la librairie La première étape consiste à préparer les fragments d'ADN à séquençer. Il existe deux grandes stratégies pour l'obtention de l'ADN à séquencer, selon la région d'intérêt que l'on souhaite analyser.

Séquençage du génome entier : Le séquençage du génome entier consiste à fragmenter et séquencer l'intégralité du génome. Il existe deux méthodes pour fragmenter l'ADN :

— Fragmentation mécanique : par sonication (en utilisant des ultra-sons qui cassent l'ADN) ou par nébulisation.

— Fragmentation enzymatique : en utilisant des enzymes de restriction qui coupent l'ADN au niveau de sites de restriction.

La fragmentation permet d'obtenir des fragments d'ADN de taille compatible avec la technologie de séquençage. L'étape suivante de préparation de la librairie consiste à ajouter aux extrémités de ces fragments des adaptateurs permettant leur fixation sur le support de séquençage pour l'amplification et le séquençage (Figure 1.16).

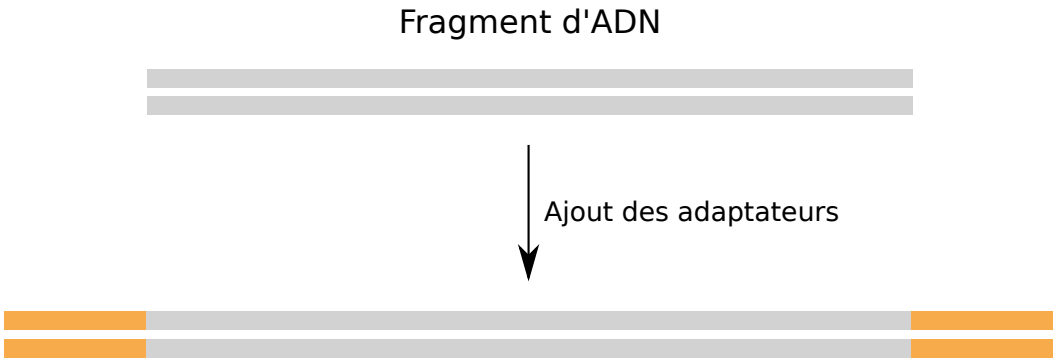


FIGURE 1.16 – Préparation de la librairie. Les adaptateurs sont ajoutés aux fragments d'ADN qu'on souhaite séquencer.

Séquençage ciblé : Dans certains cas, on s'intéresse à des variants dans des gènes spécifiques, notamment pour des applications cliniques où on se limite généralement à un panel de gènes ou bien à l'ensemble des exons (dit *exome*). Il est donc inutile de séquençer le génome entier.

Il existe deux grandes approches pour cibler et enrichir la librairie en régions d'intérêt. La première est l'approche « amplicons » qui consiste à utiliser des amorces encadrant les régions d'intérêt, pour pouvoir les amplifier par PCR (*Polymerase Chain Reaction*). La deuxième est l'approche « capture », qui consiste à fragmenter l'ADN et à capturer les régions d'interêt avec des sondes nucléotidiques d'une centaine de paires de bases. L'approche amplicon est généralement utilisée pour séquencer un nombre limité de régions d'intérêt, tandis que l'approche capture est utilisée pour capturer des grands panels de gènes (Figure 1.17).

Il est possible de séquencer l'ADN de plusieurs individus en un seul *run* de séquençage. Ceci permet d'optimiser le coût et le temps de séquençage en profitant au maximum du débit de la machine utilisée. Des petites séquences individuelles appelées « codes-barres », sont ajoutées à chaque fragment d'ADN pendant l'étape de préparation de la librairie. Ces petites séquences permettent d'identifier l'origine de chaque *read* après le séquençage (Figure 1.18).

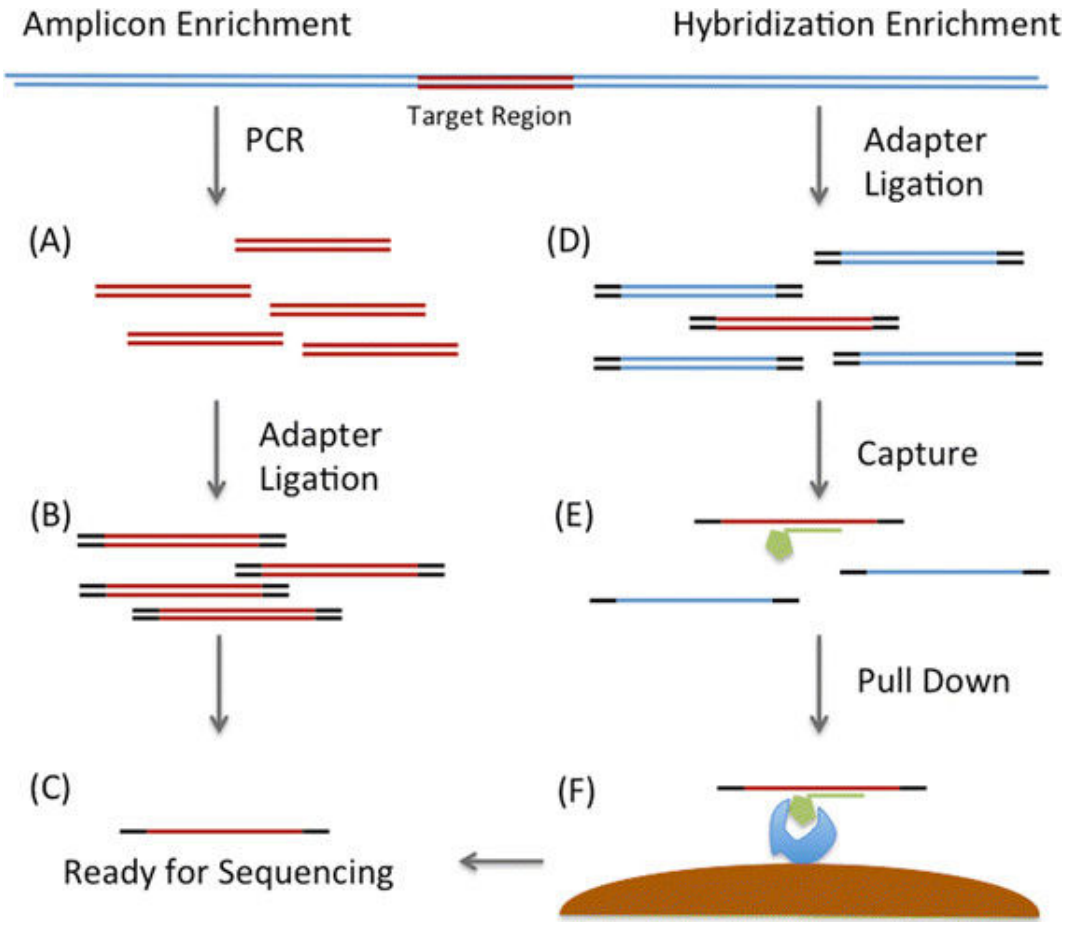


FIGURE 1.17 – Les deux principales méthodes d'enrichissement d'ADN. À gauche, la méthode d'enrichissement par amplification PCR. À droite, la méthode d'enrichissement par hybridation et capture. Source de la figure : [205]

1.2.4.2.2 Amplification clonale Les fragments attachés aux adaptateurs sont déposés et distribués aléatoirement sur le support de séquençage (*FlowCells* dans le cas de la technologie illumina, *Ion Sphere Particles (ISP)* dans le cas de la technologie Ion-Torrent par exemple). Chaque fragment d'ADN subit ensuite une amplification clonale, c'est-à-dire plusieurs cycles d'amplification successives par PCR, pour obtenir un clone de molécules identiques à la molécule de départ. Ce clone est appelé *cluster* dans le cas d'Illumina (Figure 1.19).

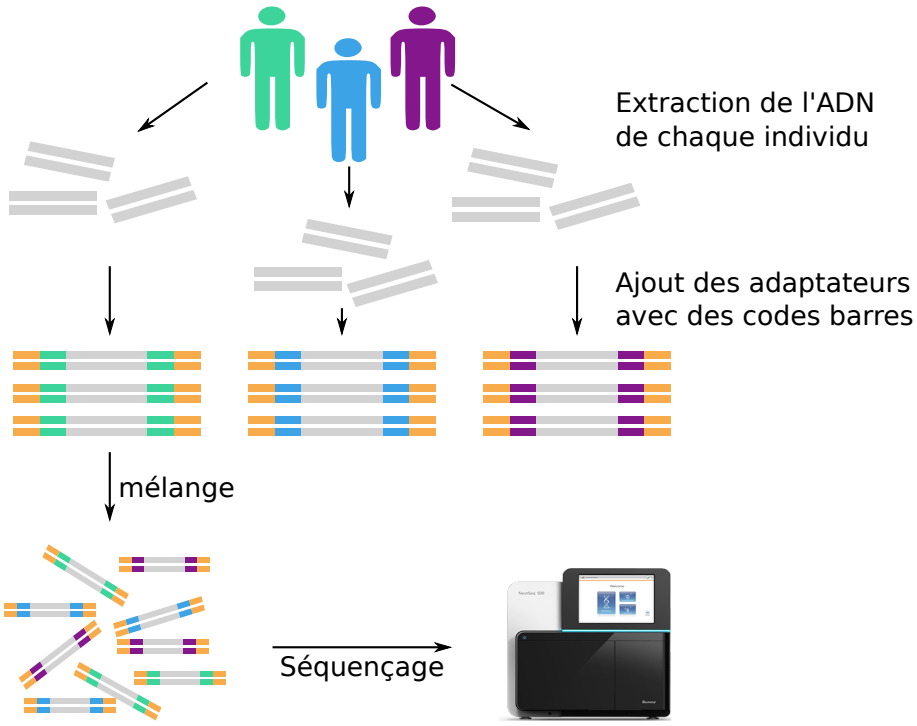


FIGURE 1.18 – Multiplexage à l'aide des codes barre. Séquençage d'une librairie construite à partir de fragments d'ADN provenant de plusieurs individus différents. Les fragments d'ADN de chaque individu sont identifiés grâce à une petite séquence (code barre) ajoutée avec les adaptateurs.

1.2.4.2.3 Séquençage et génération de données Il existe plusieurs technologies de séquençage en fonction du séquenceur utilisé. On peut citer le séquençage par synthèse, par ligation ou bien le séquençage en temps réel (SMRT) (Table 1.2). Les séquenceurs de 2ème génération utilisent la technologie de séquençage par synthèse. Les différentes technologies ne seront pas détaillées, seules les grandes lignes communes à tous les séquenceurs seront décrites.

Les fragments d'ADN préparés et amplifiés précédemment (la librairie) sont séquencés par le séquenceurs. Chaque base séquencée génère un signal qui lui est spécifique. Ce signal brut, qui peut être un signal lumineux dans le cas d'Illumina ou une différence de pH dans le cas d'IonTorrent, est converti par le séquenceur en une séquence nucléotidique. Chaque séquence nucléotidique obtenue par séquençage d'un fragment d'ADN est appelée un *read* (une lecture). Ces *reads* sont enregistrés dans un fichier au format Fastq, contenant les séquences nucléotidiques et leurs scores de qualité (Figure 1.20).

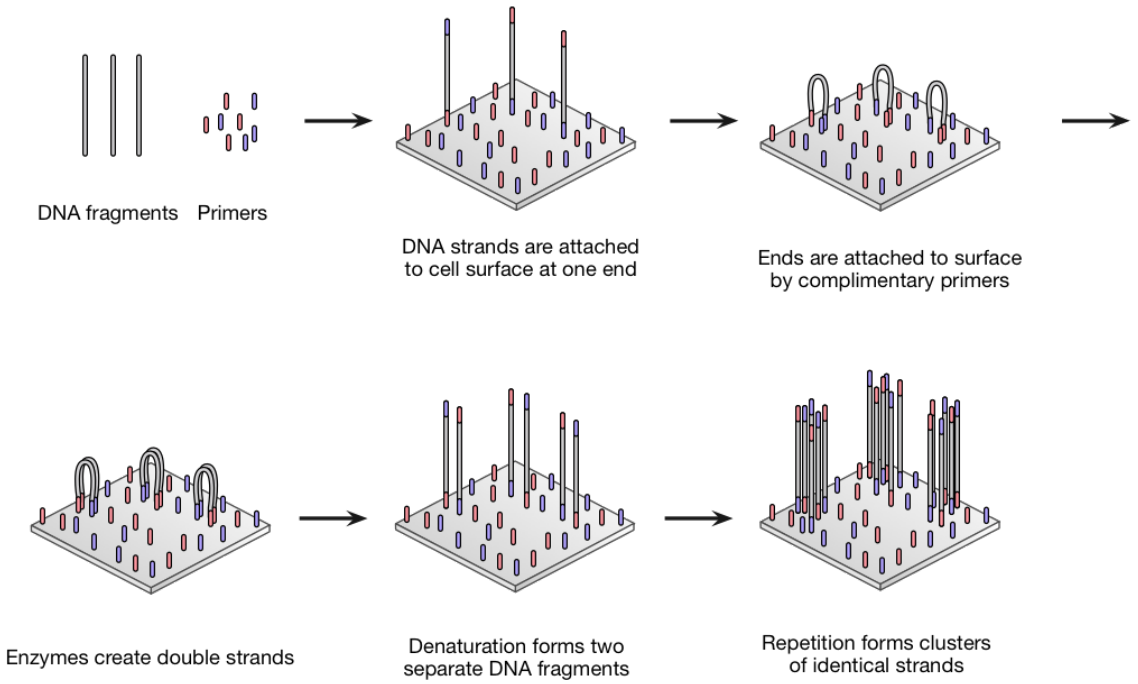


FIGURE 1.19 – Amplification clonale sur *FlowCell* d'Illumina. Source de la figure : [8]

read 1
```
@SRR062641.6751359
CGCCCGGCCAATCATTGTGGTTTTAAGTCACTAAGTTTGAGGCTATTTTGTTTTACAGCAAAAGCTAACTGATGCA
+
CBLNPGJQQQJPPQPPQPQRGPPPPRRQQRPSPGRQQQRLRRRMEPQQPMJHQQEHKMMFIIRH?SIIHKNJIKRL
```

read 2
```
@SRR062634.16249693
CTAAGTTTGAGGCTATTTTGTTTTACAGCAAAAGCTAACTGATGCAGACAGGGACAAGTCAGTCTCATCTCTGTGC
+
ALKMOOOOPPQJQOPPPPPQPPPPPPPRJQRQQQQQRPQPRQQPFQSQQPRLIMHKSNRJQORMFELRPQNQRQJQR
```

read 3
```
@SRR062634.20060465
CTCCCAGCTTCCAACAGACCCTGTCCCAGCTCCCTCCAAGCTGAGTGTTGGCCTGATACCTACCAGTGGAGCGAGG
+
D?KMPQEPGCPQQNPQIQIGR@DPERQHEKBED=HCHG8EHFDCD6<329@<:69A<6,;<967>;=C:>AA8BBE
```

FIGURE 1.20 – Fichier de *reads* au format Fastq. Chaque *read* utilise au minimum 4 lignes. La première ligne (en rouge) commence par un caractère "@" suivi de l'identifiant de la séquence et éventuellement d'une description. La deuxième ligne (en vert) contient la séquence nucléotidique. La troisième ligne commence par un caractère "+", qui peut être facultativement suivi d'une répétition de la première ligne. La quatrième ligne (en bleu) contient les scores de qualité associés à chacune des bases de la séquence de la ligne 2 et doit avoir exactement le même nombre de symboles que la ligne 2. Les valeurs de score de qualité sont codées au format ASCII.

La librairie d'ADN peut être séquençée de deux manières différentes (Figure 1.21) :

— le séquençage *Single-End* : le fragment d'ADN est séquencé dans une seule direction (un seul brin). Le choix du brin séquencé est aléatoire.

— le séquençage *Paired-end* : les deux brins sont séquencés. Les *reads* produits dans les deux directions peuvent se chevaucher, en fonction de la longueur du fragment d'ADN initial.

FIGURE 1.21 – Séquençage *Single-End* et *Paired-End*. À gauche, le séquençage *Single-End* : le séquençage se fait uniquement dans une direction d'une façon aléatoire (brin sens ou anti-sens). À droite, le séquençage *Paired-End* : le séquençage se fait dans les deux directions (brin sens et anti-sens).

Chaque position nucléotidique est séquencée plusieurs fois, et le nombre de fois où une base est séquencée correspond à la *profondeur de séquençage* (Figure 1.22).

La *profondeur de séquençage moyenne* (P) est le nombre moyen de *reads* qui couvrent une base. Par exemple, $30X$ veut dire qu'en moyenne, une base est couverte par 30 *reads*. $P = L.N/G$, où L est la longueur moyenne des *reads*, N le nombre de *reads* et G la longueur du génome haploïde.

La *couverture de séquençage* (C) correspond au pourcentage de la région d'intérêt bien couverte par des *reads*. $C_x = B_x/G$, avec B_x le nombre de bases couvertes par au moins x *reads* et G la longueur du génome haploïde.

1.2.4.2.4 Analyses bio-informatiques pour la recherche de variants Les données générées par les séquenceurs (les *reads*) sont analysées à l'aide d'outils bioinformatiques. Les outils utilisés, ainsi que les différentes étapes d'analyse, varient d'une application à une autre. Voici les principales étapes d'analyse bio-informatique pour la recherche de variants :

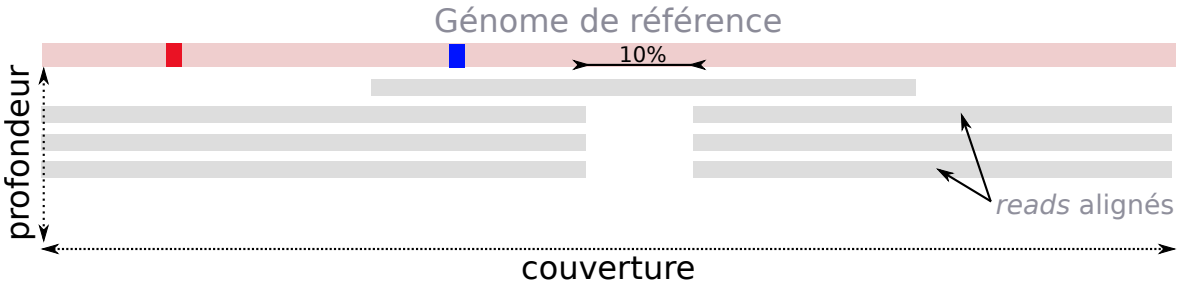


FIGURE 1.22 – Profondeur et couverture de séquençage. La prodondeur de séquençage au niveau de la base rouge est égale à $3X$, alors qu'au niveau de la base bleue elle est égale à $4X$. La profondeur de séquençage moyenne est environ égale à $3X$. La couverture de séquençage est de 90% (10% de bases sont mal couvertes), si on considère qu'une profondeur de $3X$ est suffisante.

1. Nettoyage des fichiers de *reads* : cette étape consiste à éliminer les *reads* ou les bases de mauvaise qualité, ainsi que les extrémités de séquences correspondant aux adaptateurs.
2. Alignement des *reads* : les *reads* sont alignés contre le génome de référence, avec des aligneurs tel que BWA [116] et Bowtie [110]. Le format de fichier de sortie est le SAM (*Sequence Alignment/Map*) ou son équivalent binaire BAM (Figure 1.23). Les *reads* qui s'alignent sur le brin sens de référence sont appelés *reads forward*, alors que les *reads* qui s'alignent sur le brin anti-sens (dans la direction 3'→5' par rapport au brin sens) sont appelés *reads reverse*.
3. Appel des variants : cette étape consiste à identifier les différences par rapport à la séquence de référence. Les fichiers BAM sont donc parcourus position par position pour détecter les variants génomiques. Les outils effectuant l'appel des variants sont appelés *variant callers*. Il en existe des dizaines [86] et ils sont classés en plusieurs catégories (détection des variants constitutionnels, somatiques, des CNV, des SV, ...). Le fichier de sortie est au format VCF (*Variant Call Format*) [165] (Figure 1.25 et 1.24).

1.2.4.2.5 Problématique des variants de faible ratio allélique Comme expliqué dans la section 1.2.3, il existe différents types de variations, germinales et somatiques. Dans le cadre d'une recherche de variants germinaux (variants présents dans toutes les

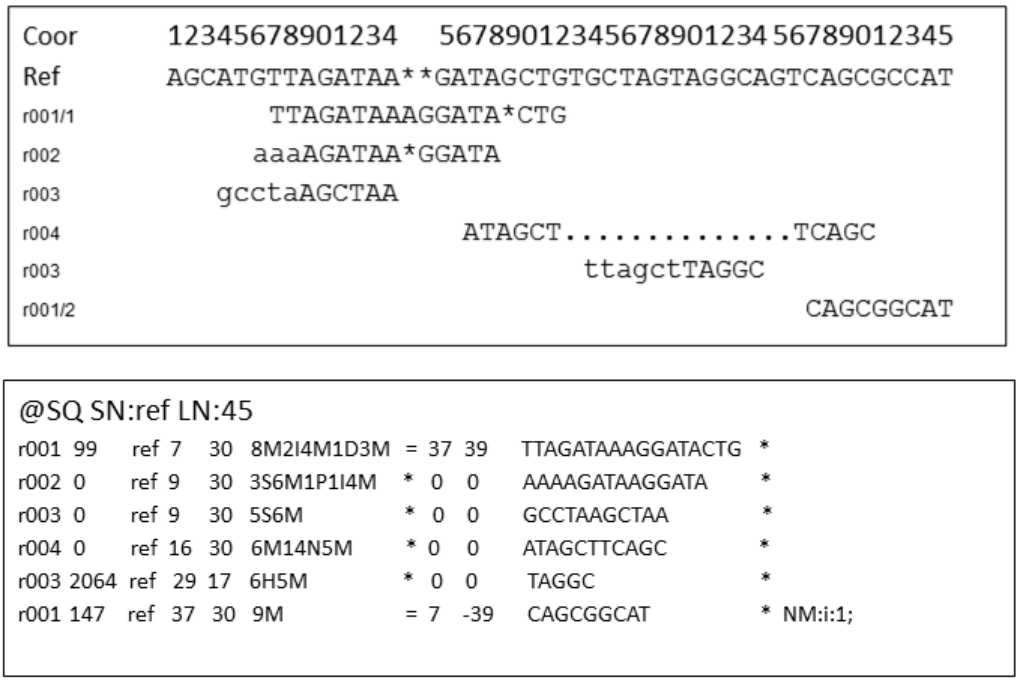


FIGURE 1.23 – Alignement de *reads* encodé dans un fichier au format SAM. En haut, un alignement de 6 *reads* contre la séquence de référence. En bas, l'encodage de cet alignement en format SAM, qui décrit à quelle position chaque *read* est aligné et comment (le nombre de mismatches, InDels, ...). Chaque ligne correspond à un *read*, et contient 11 champs obligatoires. Parmi ces champs, le premier correspond au nom du *read*, le troisième correspond au nom de la séquence de référence, suivi de la première position de l'alignement et le dixième champ correspond à la séquence du *read*.

cellules de l'individu), le taux de détection de variants est généralement relativement élevé (ratio allélique > 30% pour un ratio allélique théorique de 50%). Tandis que dans le cas d'une recherche de variants somatiques (séquençage d'une tumeur hétérogène par exemple), on cherche des variants qui peuvent avoir un ratio allélique beaucoup plus faible que celui des variants germinaux (ratio allélique aussi bas que 1%).

L'analyse des variants somatiques ayant un faible ratio allélique est devenue possible avec le NGS grâce à son débit élevé permettant de séquencer et de lire individuellement les différents allèles contenus dans différents clones cellulaires. Les *reads*, provenant de plusieurs cellules différentes, sont en effet visualisés individuellement au lieu d'avoir un signal global pour chaque position nucléotidique comme c'est le cas pour le séquençage Sanger. Le NGS se caractérise donc par une meilleure sensibilité et par sa capacité à détecter des variants minoritaires (ratio allélique ~1%) [105]. La proportion

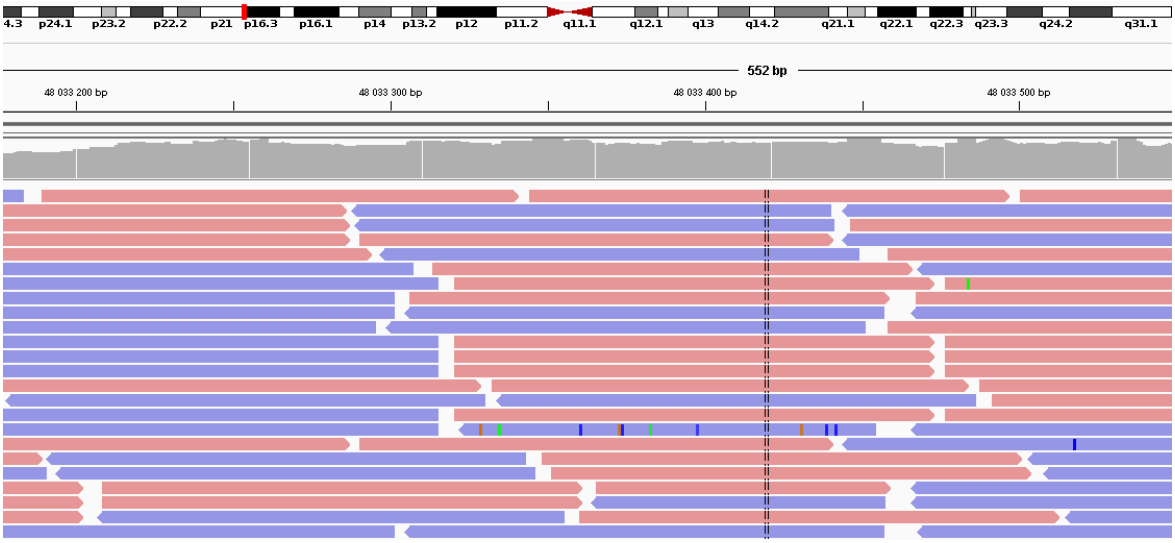


FIGURE 1.24 – Capture d'écran du logiciel de visualisation d'alignement IGV [196]. Ce logiciel prend des fichiers d'alignement au format BAM. En rouge sont représentés les *reads sens*. En bleu sont représentés les *reads anti-sens*. Les couleurs à l'intérieur d'un *read* correspondent à des mismatchs par rapport à la séquence de référence.

```
#CHROM POS     ID        REF     ALT     QUAL FILTER INFO
20     14370   rs6054257 G       A       29   PASS   NS=3;DP=14;AF=0.5;DB;H2
20     17330   .         T       A       3    q10    NS=3;DP=11;AF=0.017
20     1110696 rs6040355 A       G,T     67   PASS   NS=2;DP=10;AF=0.333,0.667;AA=T;DB
```

FIGURE 1.25 – Fichier au format VCF. Chaque ligne correspond à un variant. Les deux premières colonnes correspondent à la position du variant (chromosome et position), la troisième colonne contient le nucléotide de référence et la quatrième contient les nucléotides alternatifs.

de *reads* qui contiennent un variant donné est appelé VAF (Variant Allele Frequency). La VAF reflète le ratio allélique d'un variant au sein de l'échantillon séquencé. Ainsi, un variant présent dans toutes les cellules d'un individu aura une VAF~1 (~100% de *reads* porteurs du variant) dans le cas d'un variant homozygote et une VAF~0.5 (~50% de *reads*) dans le cas d'un variant hétérozygote.

Dans le cas d'un variant somatique identifié dans des cellules tumorales, la VAF dépend de deux facteurs :

— La pureté de l'échantillon : il est rare qu'un échantillon tumoral ne contienne que

des cellules tumorales (présence également de cellules non tumorales).

— L'hétérogénéité (clonalité) de la population de cellules tumorales : dans une même tumeur, on peut avoir différents sous-clones avec des variants différents, et par conséquent des ratio alléliques variables en fonction de la proportion de chaque sous-clone.

À cause de ces deux facteurs, les variants somatiques peuvent avoir une VAF faible (Figure 1.26). Prenons l'exemple d'un variant hétérozygote somatique présent dans une tumeur et un prélèvement contenant 60% de cellules tumorales réparties en 3 clones représentant chacun approximativement 50, 30 et 20% de la tumeur. Si le variant hétérozygote est présent dans les deux clones minoritaires, il sera détecté avec une VAF d'environ 15%.

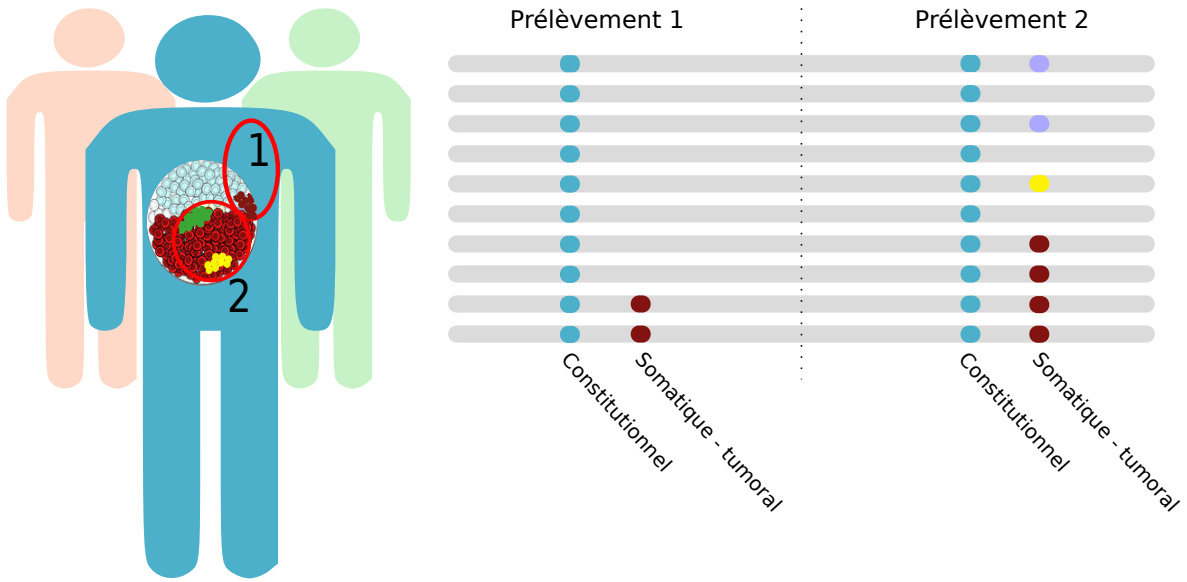


FIGURE 1.26 – Différents types de variants. Un variant constitutionnel homozygote se trouve sur la totalité des *reads*. Pour les variants somatiques (comme dans le cas des mosaïques et des tumeurs), ils peuvent se trouver sur une faible proportion de *reads*, en fonction de la clonalité et de la pureté de l'échantillon.

CHAPITRE 2

L'INFÉRENCE DE MOTIFS APPROCHÉS SUR-REPRÉSENTÉS

2.1 Mots et motifs

Nous avons vu dans le chapitre 1 que les quatre bases nucléotidiques d'ADN pouvaient être représentées comme des mots sur l'alphabet $\{A, C, G, T\}$. Il est souvent intéressant de pouvoir considérer et manipuler des ensembles de mots similaires, qui partagent par exemple une fonction commune. Nous appelons de tels ensembles de mots des *motifs*.

Ces motifs fréquents peuvent être porteurs d'un signal biologique, et être donc associés à une fonction particulière. A titre d'exemple, on peut citer les sites de clivage des enzymes de restriction, tels que le motif A↓AGCTT reconnu par Hind II [98, 153], les sites de fixation de facteurs de transcription (TFBS) qui participent à la régulation transcriptionelle des gènes [168] et les cibles de microARN [207].

Dans cette thèse, nous nous intéressons plus particulièrement aux motifs liés aux erreurs de séquençage systématiques (SSE). À la différence des exemples cités précédemment, il ne s'agit pas d'un signal biologique reconnu par une molécule, telle qu'une protéine ou un ARN. Ces motifs sont des séquences difficiles à séquençer par les techno-

logies utilisées actuellement et qui vont favoriser localement l'apparition d'une erreur de séquençage. Nous commençons par présenter l'état de l'art sur les méthodes utilisées en informatique pour la recherche de motifs, et nous reviendrons en détails sur le lien entre les motifs et la qualité du séquençage dans le chapitre 3.

2.2 Brève perspective historique sur la recherche de motifs

2.2.1 Recherche de mots

La recherche de motifs est l'un des problèmes les plus étudiés en informatique et plus spécifiquement dans le cadre de l'algorithmique de texte. Dans sa version la plus simple, ce problème consiste à chercher les occurrences exactes d'un mot x de taille m dans un texte y de taille n. Un mot est simplement une suite de caractères. Un algorithme naïf basé sur une fenêtre glissante permet de chercher toutes les occurrences avec un temps d'exécution quadratique $\mathcal{O}(n \times m)$.

1: **function** LOCALISER-NAÏVEMENT (x, m, y, n)
2: **for** $j \leftarrow 0$ **to** $n - m$ **do**
3: **function** SIGNALER-SI $(y[j..j + m - 1] = x)$

En 1970, Donald Knuth et Vaughan Pratt ont proposé un algorithme plus efficace, appelé algorithme de Knuth-Morris-Pratt (KMP) [102, 137], permettant de trouver les occurrences d'un mot avec une complexité linéaire : $\mathcal{O}(n+m)$ dans le pire cas. Ce progrès est rendu possible par l'exploitation de la structure interne du mot, et des caractères répétés. L'algorithme comprend une étape de pré-traitement du mot, avec la construction d'une structure de données, la table de sauts. Cet algorithme a marqué le début d'une longue série d'améliorations, vers toujours plus d'efficacité. En 1977, Robert Boyer et J. Strother Moore ont publié un algorithme nommé algorithme de BOYER-MOORE [28] dont le temps d'exécution peut être sous-linéaire (n/m) et qui utilise une table des suffixes. L'ouvrage de CROCHEMORE *et al.* [46] est une bonne référence concernant les algorithmes utilisés pour la recherche de mots.

Une variante de ce problème, consiste à rechercher un motif constitué de plusieurs

mots dans un texte. Les travaux de Searls [174–177] montrent que certaines séquences biologiques se formalisent par des langages, suivant la hiérarchie de Chomsky [38]. La hiérarchie de Chomsky définit différentes classes de langages, dont les langages rationnels (expressions régulières) reconnus par les automates finis, et les langages algébriques, engendrés par une grammaire hors contexte et par les automates à piles. Ces derniers permettent de décrire plus de structures biologiques, notamment les structures secondaires (tige-boucle par exemple), par rapport aux langages rationnels.

2.2.2 Recherche de mots approchée

Dans le cadre d'applications telles que les moteurs de recherche et les correcteurs orthographiques, le problème de la recherche exacte de mots n'est pas suffisant et on a besoin d'effectuer une recherche approchée de mots. Il s'agit de rechercher toutes les occurrences approchées qui sont à distance au plus k d'un mot x. Les distances qui sont généralement utilisées sont la distance de HAMMING et la distance de LEVENSHTEIN [142]. Les algorithmes développés pour la recherche approchée sont également très nombreux. Ils peuvent être divisés en quatre catégories principales : 1) les algorithmes basés sur la programmation dynamique, tels que l'algorithme développé par SELLERS [179] ($\mathcal{O}(m \times n)$) et plus tard celui de LANDAU & VISHKIN [109] ($\mathcal{O}(n \times k)$), 2) les algorithmes basés sur les automates, dont l'algorithme de KURTZ [108], 2) les techniques de *Bit-parallélisme* qui exploitent le parallélisme de l'ordinateur pour paralléliser les travaux précédents basés sur les automates, tel que l'algorithme de BAEZA-YATES & NAVARRO [19] ou bien les matrices de programmation dynamique, tel que l'algorithme de MYERS [140], 4) les algorithmes de filtrage, qui consiste à écarter les zones du texte qui ne peuvent pas contenir des occurrences de motif. En pratique, les outils de la dernière catégorie sont les plus rapides pour la recherche approchée de motifs complexes. Par exemple, l'outil nr-grep basé sur l'algorithme BNDM [143] a une complexité dans le pire des cas en $\mathcal{O}(m \times n)$, mais est sous linéaire en pratique si l'alphabet est uniformément représenté dans le texte et les lettres indépendantes. Pour une discussion plus détaillée sur l'ensemble des algorithmes de recherche approchée de mots avec leur complexité en temps d'exécution et d'espace, nous renvoyons le lecteur vers l'article de revue de NAVARRO [142].

2.2.3 Structures d'indexation de texte

La grande majorité des méthodes citées ci-dessus se basent sur une étape de pré-traitement du ou des mots recherchés, dans le but d'accélérer la phase de recherche dans le texte. Ces méthodes sont dites *On-line*, et sont utilisées pour la recherche dans des textes d'une taille relativement petite, de l'ordre du millier voire du million de caractères.

Le séquençage massif de séquences biologiques à partir des années 1990 apporte le besoin de traiter des grands volumes de données. La quantité de données à traiter a augmenté, passant à l'échelle du milliard de nucléotides. Des nouvelles méthodes ont été développées pour la recherche de mots. Ces méthodes se basent principalement sur le pré-traitement du texte (génome de référence humain par exemple) en créant un index permettant d'interroger et localiser rapidement un ou plusieurs mots dans le texte (méthodes *Off-line*). Plusieurs types d'index sont couramment utilisés en bioinformatique, comme l'arbre des suffixes [130, 204], la table des suffixes [126] ou bien le FM-index [64]. Même si le temps de construction et l'espace de ces index peuvent être lourds, ils ont un intérêt majeur lorsqu'un nombre élevé de requêtes est effectué sur le même texte (ici le génome de référence). En effet la recherche d'un motif de taille m sur un texte indexé peut se faire avec une complexité en temps en $\mathcal{O}(m)$.

2.2.4 Modélisation des motifs biologiques

Le traitement des séquences biologiques pose également la question de l'expressivité des modèles de motifs. Motif s'entend ici comme un ensemble de mots correspondant à une même fonction, ou un même signal. Les modèles utilisés en informatique restent peu pertinents pour certaines applications bioinformatiques. Pour la localisation des sites biologiques par exemple, il est désormais convenu que les motifs consensus, avec erreurs uniformes, ne sont pas adaptés à la description des sites de fixation de motifs de transcription. De ce fait, la majorité des algorithmes utilisent des représentations matricielles plus complexes, qui seront discutées dans les sections qui suivent, et qui ne sont pas compatibles avec les algorithmes de recherche décrits précédemment.

L'analyse de séquences biologiques pose également un nouveau type de question. Le motif recherché peut être inconnu, et le problème est de l'identifier. Il s'agit alors

d'un problème « d'inférence de motifs » (ou de découverte de motifs), c'est-à-dire de l'extraction de motifs, dont la forme est inconnue au départ, à partir d'un ensemble de séquences. Ces motifs sont détectés grâce à des techniques, apparentées aux méthodes utilisées dans le domaine de la fouille de données [12, 120], et qui consistent à analyser la fréquence des mots afin de repérer, grâce à une fonction de score, des motifs qui sont sur-représentés dans l'ensemble de séquences par rapport à un modèle de base. C'est dans ce cadre que nous nous plaçons.

Dans le présent chapitre, nous présentons les principes généraux de l'inférence de motifs sur-représentés en bioinformatique, tant sur le versant algorithmique que statistique. On peut diviser en deux grandes catégories les méthodes existantes : les méthodes énumératives et les méthodes probabilistes, qui diffèrent à la fois par leur façon de modéliser les motifs et par leur algorithme de recherche de motifs sur-représentés. D'un point de vue algorithmique, on peut aussi distinguer les approches gloutones et les approches globales.

2.3 Modèles de représentation des motifs d'ADN en bioinformatique

Différents modèles existent pour représenter les motifs d'ADN. On peut les classer en deux catégories : les modèles *à base de chaînes de caractères*, présentés en sous-section 2.3.1, et les modèles *probabilistes*, présentés en sous-section 2.3.2.

2.3.1 Modèles basés sur les chaînes de caractères

Dans ce type de modèle, le motif est défini explicitement comme un ensemble de mots, qui est décrit lui-même par une chaîne de caractères. C'est donc une représentation discrète.

2.3.1.1 Oligo-mers

Un oligo-mer est simplement un mot, c'est-à-dire que c'est un motif sans dégénérescence. Il ne permet pas de décrire une variabilité (des régions peu conservées) au sein du motif.

Ce type de modèle a été utilisé par les premiers outils pour la recherche des motifs d'ADN sur-représentés dans le génome de la levure [186, 200] (Figure 2.1.a).

a)

A A C T G

b)

A	A	C	T	C
A	A	C	T	G
A	G	C	T	C
A	G	C	T	G
A	A	C	T	G
A	A	C	T	G

c)

A	A	C	T	C
A	A	C	T	G
A	G	C	T	C
A	G	C	T	G
A	A	C	T	G
A	R	C	T	S

FIGURE 2.1 – Les modèles basés sur les chaînes de caractères. a) le modèle oligo simple. b) le modèle mismatch ($e = 2$). c) le modèle IUPAC.

2.3.1.2 Mismatch (consensus)

Le motif est décrit à partir d'un mot sur l'alphabet $\{A,C,G,T\}$, appelé séquence *consensus*, et un nombre borné d'erreurs e autorisées par rapport au consensus. Les erreurs s'entendent ici au sens de la distance de Hamming : il s'agit de substitutions, ou de remplacements de lettres (Figure 2.1.b).

Ce modèle est utilisé par exemple par PAVESI *et al.* dans l'outil Weeder [150]. À la différence du modèle oligo, le modèle « mismatch » permet une certaine variabilité en autorisant un nombre fixé e de mutations par rapport au motif de base (la séquence consensus).

La caractéristique principale de ce modèle est qu'il utilise un seuil de variation e qui s'applique à l'ensemble du motif, au lieu de spécifier les variations possibles sur chaque lettre indépendamment. Un ensemble de mots similaires est représenté par le motif consensus, autrement dit, à chaque position on garde la lettre la plus présente.

2.3.1.3 Modèle IUPAC

Le codage IUPAC (*International Union of Pure and Applied Chemistry consortium*) est utilisé pour symboliser les différentes combinaisons de bases nucléiques ou d'acides aminés sous forme de lettres. Pour l'alphabet $\{A,C,G,T\}$, le codage IUPAC contient 15 lettres, correspondant aux 15 sous-ensembles non vides de $\{A,C,G,T\}$. Le détail est donné dans la table 2.1.

Les bases représentant plusieurs lettres, telles que W, B ou N, sont dites des bases *dégénérées*. Elles représentent des positions ambiguës dans le motif, qui acceptent des nucléotides alternatifs. Par exemple, les deux séquences $AAA\mathbf{T}$ et $AAA\mathbf{A}$ peuvent être représentées par la séquence IUPAC $AAA\mathbf{W}$ en utilisant la base dégénérée W (Figure 2.1.c).

Code IUPAC	**Bases**
A	**A**denine
C	**C**ytosine
G	**G**uanine
T (ou U)	**T**hymine (ou **U**racil)
R	A ou G
Y	C ou T
S	G ou C
W	A ou T
K	G ou T
M	A ou C
B	C ou G ou T
D	A ou G ou T
H	A ou C ou T
V	A ou C ou G
N	A ou C ou G ou T

TABLE 2.1 – Code IUPAC

2.3.2 Modèles probabilistes

Une description plus statistique consiste à représenter un motif donné en le décrivant d'une manière probabiliste. Dans les modèles probabilistes, la probabilité de chaque nucléotide est évaluée à chaque position donnée dans le motif [162]. Les paramètres du modèle sont estimés à partir d'un jeu de données en utilisant la méthode du maximum de vraisemblance ou d'inférence bayésienne [47]. Deux modèles probabilistes sont généralement utilisés pour la représentation des motifs d'ADN, les matrices PWM et les HMM.

2.3.2.1 Matrices PWM

La matrice PWM (*Position Weight Matrix*), ou PSSM (*Position Specific Scoring Matrix*), est sans doute le modèle probabiliste le plus simple. Ce modèle a été introduit par STORMO *et al.* en 1982 [188] comme une alternative aux modèles basés sur les

chaînes de caractères. Le motif est représenté sous forme d'une matrice de dimension (c, ℓ), où c est la taille de motif et ℓ la taille de l'alphabet (4 dans le cas d'ADN et 20 pour les protéines).

Une matrice PWM est calculée à partir d'un ensemble de séquences alignées de même longueur, qui représentent le motif. On commence par compter la fréquence de chaque nucléotide à chaque position. Cela donne une matrice d'occurrences appelée PFM (*Position Frequency Matrix*).

Ces PFM sont ensuite transformées en PPM (*Position Probability Matrix*) en normalisant le nombre d'occurrences de chaque nucléotide à chaque position par rapport au nombre total de séquences. Dans les matrices PPM, on suppose que les colonnes sont indépendantes les unes des autres.

La matrice PWM est ensuite calculée à partir de la matrice PPM en normalisant les probabilités en fonction de la fréquence de chaque nucléotide dans le modèle de base pour obtenir son contenu en information (Figure 2.2).

Le contenu en information permet d'évaluer la divergence d'une position de motif par rapport au modèle de base. La quantité d'information pour chaque lettre à chaque position est calculé avec la formule suivante :

$$W_{i,j} = \log_2(P_{i,j}/P_b)$$

où $P_{i,j}$ est égal à la probabilité de la lettre i à la position j dans le motif, obtenue à partir de la PPM, et P_b est égal à la probabilité de la même lettre dans le modèle de base.

Dans le cas d'un faible nombre de séquences, il est possible d'obtenir des valeurs nulles dans les PFM, conduisant à des valeurs nulles dans la PPM et une division par zéro dans le PWM. Cela est souvent considéré comme indésirable [57], car des nucléotides peuvent être absents d'un petit échantillon par hasard, et leur attribuer une probabilité nulle est trop pénalisant [147]. Il est donc courant d'ajouter des pseudo-comptes lors du calcul de la PFM, afin d'éviter des probabilités nulles. Différentes valeurs de pseudo-comptes ont été utilisées, telles que 0,01 [32], 1 [89], 2 [135]. Ces pseudo-comptes

sont divisés par 4 et ajoutés à chaque élément de la PFM.

La quantité d'information à une position donnée de la matrice est obtenu en sommant les quantités d'information de toutes les lettres à cette position. Il est mesuré en *bits*. Les positions parfaitement conservées ($P_{i,j} = 1$) contiennent 2 bits d'information, tandis que les positions ayant les quatre lettres représentées d'une façon équiprobable ($P_{i,j} = 0.25$ et $P_b = 0.25$) ont un contenu informationnel nul. Le contenu d'information d'une matrice est la somme des contenus d'information de toutes les colonnes.

La limitation principale du modèle PWM est que les distributions de probabilité pour les positions individuelles sont indépendantes. Cela signifie que la PWM n'est pas capable de capturer des dépendances nucléotidiques d'ordre supérieur [167]. Ce modèle ne permet pas non plus de prendre en compte la possibilité d'insertions ou de délétions dans le motif.

2.3.2.2 Modèle de Markov caché

Les *modèles de Markov cachés* (ou HMM, pour *Hidden Markov Model* en anglais) peuvent être vus comme une extension des PWM. Les HMM, à la différence des PWM, autorisent des insertions et des délétions.

Chaque état a une probabilité d'émission différente pour chaque lettre de l'alphabet (4 dans le cas d'ADN), et une probabilité de transition vers la base suivante [57, 80, 122].

Plusieurs architectures, qui dérivent du *Profil HMM*, sont utilisées selon les outils [60] (Figure 2.3). Le *Profil HMM* est une description, sous forme de HMM, du consensus d'un alignement multiple de séquence. À chaque nœud de cet HMM il existe trois états possibles : un *match* avec les probabilités d'émission pour chaque nucléotide, une délétion ou bien une insertion d'une ou plusieurs nucléotides. Parmi les différentes variantes simplifiée du *Profil HMM* utilisées [107], on peut citer celui de META-MEME [80] qui modélise une série de motifs séparés par des insertions (Figure 2.3.a) et le modèle appelé « Plan 7 » de l'outil HMMER [58, 59], qui autorise des insertions et des délétions après chaque état de match. On peut citer aussi l'outil Discrover [122], le plus récent de cette catégorie, qui autorise une insertion d'une seule base après chaque état de *match*.

a) Séquences

$$\begin{array}{lccccc} s1= & A & A & C & T & C \\ s2= & A & A & C & T & G \\ s3= & A & A & C & T & T \\ s4= & A & G & C & T & G \end{array}$$

$\Downarrow$

b) PFM

(sans pseudo-comptes)

$$F = \begin{array}{c} \\ A \\ C \\ G \\ T \end{array} \begin{bmatrix} 4 & 3 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 0 & 4 & 1 \end{bmatrix}$$

$\Downarrow$

c) PPM

$$M = \begin{array}{c} \\ A \\ C \\ G \\ T \end{array} \begin{bmatrix} 1 & 0.75 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0.25 \\ 0 & 0.25 & 0 & 0 & 0.5 \\ 0 & 0 & 0 & 1 & 0.25 \end{bmatrix}$$

$\Downarrow$

d) PWM

sous un modèle de fond équiprobable $\left(W_{i,j} = log_2(P_{i,j}/0.25)\right)$

$$W = \begin{array}{c} \\ A \\ C \\ G \\ T \end{array} \begin{bmatrix} 2 & 1.58 & -\infty & -\infty & -\infty \\ -\infty & -\infty & 2 & -\infty & 0 \\ -\infty & 0 & -\infty & -\infty & 1 \\ -\infty & -\infty & -\infty & 2 & 0 \end{bmatrix}$$

FIGURE 2.2 – Construction d'une matrice PWM à partir d'un ensemble de séquences.

Toutefois, vu le nombre important de paramètres à estimer, ce modèle nécessite un grand jeu de données pour son entraînement, ce qui peut le rendre moins adapté pour la recherche des motifs dans des petits jeux de données et pour la détection des motifs rares.

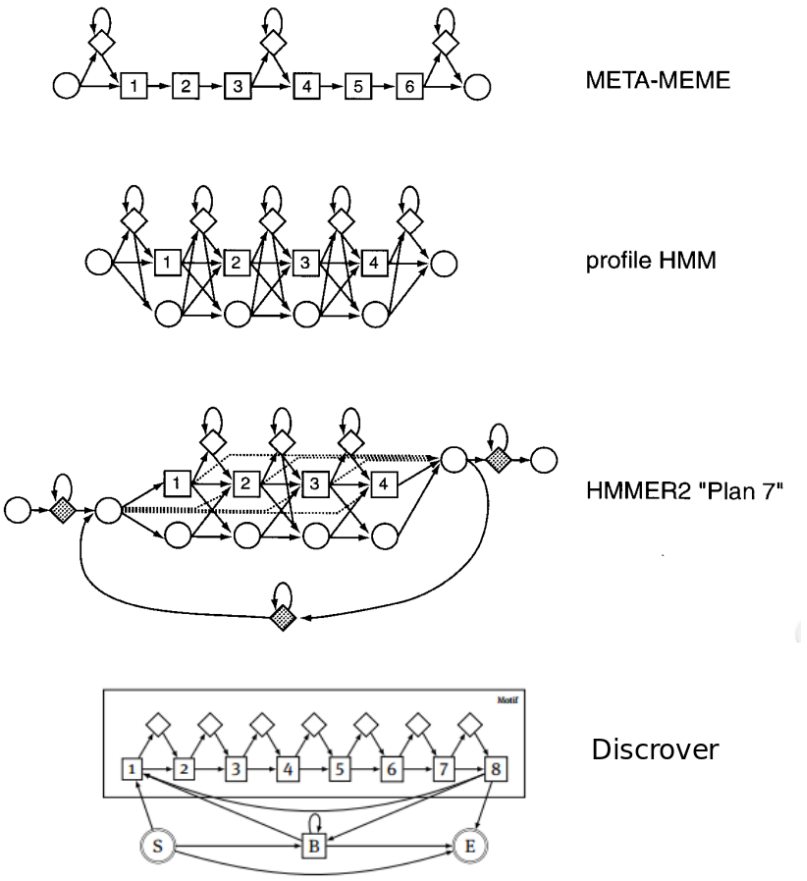


FIGURE 2.3 – Différentes architectures de modèles HMM utilisées par différents outils. Figure adaptée de : [60] et [122].

2.3.2.3 Représentation graphique des motifs probabilistes

Les deux modèles probabilistes présentés ci-dessus, PWM et HMM, sont difficiles à analyser à l'œil nu. Une première présentation graphique a été proposée par SCHNEIDER & STEPHENS [172] et consiste à présenter l'alignement de séquences sous-jacent sous la forme d'un *logo*. Ce *logo* se compose de piles de lettres (les 4 nucléotides A, C, G, T) à chaque position du motif. La hauteur de la pile indique le contenu informationnel de la position, entre 0 et 2 bits, et donc la conservation de la position. La hauteur de chaque symbole au sein de la pile indique sa fréquence. Le *logo* permet de décrire simplement un motif et de savoir quelles sont les positions les plus conservées dans le motif. (figure 2.4).

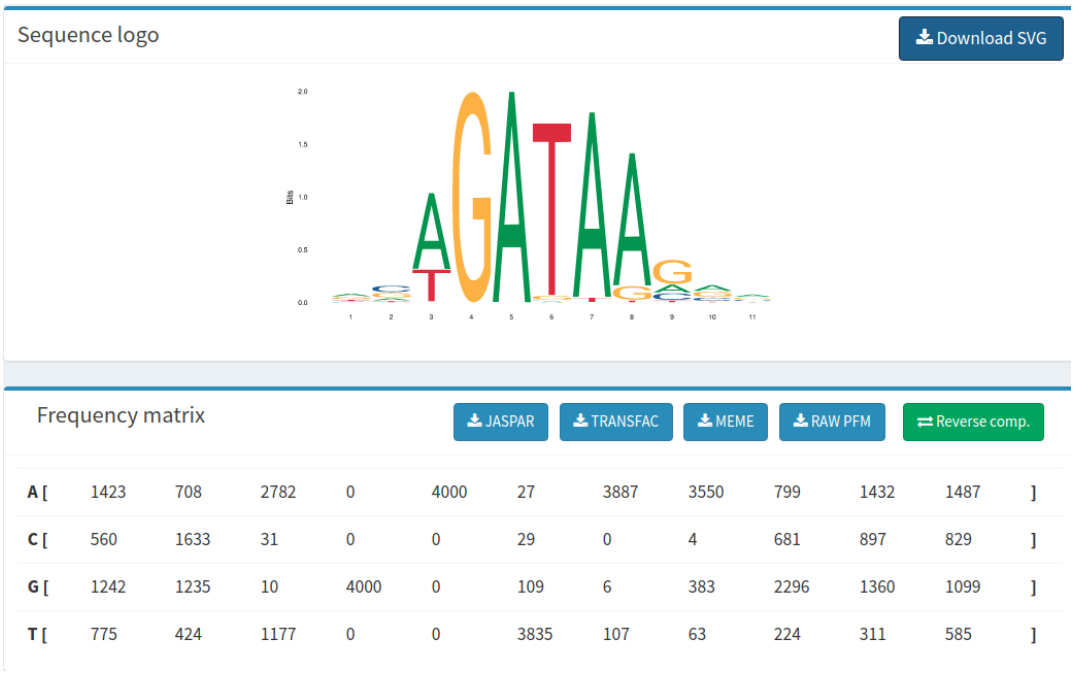


FIGURE 2.4 – Représentation en *logo* du motif du TFBS GATA1 et la matrice PFM correspondante. Source de la figure : [9]

2.4 Modèle de base

La découverte des motifs est possible grâce à leur sur-représentativité dans le jeu de données positif par rapport à un modèle de base (ou modèle de fond). Un motif est sur-représenté s'il est présent avec une fréquence plus élevée que ce qui est attendu sous un modèle de base.

Le choix d'un modèle de base est l'un des paramètres les plus cruciaux pour la découverte de modèles. Le modèle de base est utilisé pour estimer les fréquences attendues des motifs dans une séquence dépourvue de tout signal biologique spécifique (séquence neutre) [50]. Les paramètres de ce modèle peuvent être estimés à partir des séquences d'entrée elles-mêmes ou bien à partir d'un ensemble de séquences de base, par exemple l'ensemble des séquences promotrices [119] ou un ensemble de séquences aléatoirement choisies sur le génome [83]. Ces paramètres peuvent être directement fournis par l'utilisateur aussi.

Différents modèles de base ont été utilisés pour la recherche de motifs sur-représentés. Le plus simple est le modèle *IID*. Dans ce modèle, la probabilité d'occurrence d'un nu-

cléotide à une position donnée est indépendante de la position, ainsi que des nucléotides présents aux autres positions. Autrement dit, le modèle est complètement spécifié avec quatre paramètres : p_A, p_C, p_G et p_T, qui représentent les probabilités d'occurrence de A, C, G et T respectivement ($p_A + p_C + p_G + p_T = 1$).

Le modèle *IID* est très simple et ne reflète pas la complexité des génomes, où les positions successives ne sont pas indépendantes. Des chaînes de Markov avec un ordre plus élevé sont ainsi utilisées. C'est le cas par exemple des outils BioProspector [119] qui utilise une chaîne de Markov dont l'ordre varie entre 0 et 3 (choix d'utilisateur) et Peak-motifs [195] dont l'ordre de la chaîne de Markov (choisi par l'utilisateur) peut aller de 0 jusqu'à $m-2$, avec m la taille des motifs recherchés. Quant à l'outil MoSDi [128], L'utilisateur a le choix entre un modèle *IID* et un modèle Markovien.

Une variante du problème de recherche de motifs est la recherche de motifs *discriminants*. Cette variante consiste à chercher des motifs sur-représentés dans un ensemble de séquences, appelé jeu de données *positif* (ou *signal*), par rapport à un autre ensemble de séquences, appelé jeu de données *négatif* (ou *contrôle*). Le jeu de données négatif est utilisé dans ce cas pour l'estimation des paramètres du modèle de base. Parmi les outils qui font de la recherche discriminante, on peut citer DREME [20], DIPS [185] et Discrover [123].

2.5 Algorithmes de recherche de motifs sur-représentés

Étant donné un ensemble de séquences, la recherche de motifs sur-représentés consiste à inférer des motifs récurrents et qui s'y trouvent avec une fréquence significativement plus élevée que celle attendue sous le modèle de base.

Les algorithmes de recherche de motifs sont divisés en deux grandes catégories. Les *algorithmes énumératifs* qui parcourent de manière systématique l'espace des motifs possibles et testent la significativité de chaque motif analysé, et les *algorithmes probabilistes* basés sur des méthodes de recherche locale et utilisés avec des modèles probabilistes PWM et HMM.

$\mathscr{O}$, Occurrence \ $\mathscr{C}$, Condition	positif	négatif
présence du motif m	a	b
absence du motif m	c	d

TABLE 2.2 – Table de contingence construite pour chaque motif et utilisée pour le calcul du score. Le nombre a (resp. b) est le nombre de séquences de $\mathscr{P}$ (resp. $\mathscr{N}$) qui contiennent au moins une occurrence du motif m. Le nombre c (resp. d) et le nombre de séquences qui ne contiennent pas d'occurrences du motif m dans $\mathscr{P}$ (resp. $\mathscr{N}$). Ces quatre entiers sont utilisés pour estimer les probabilités conjointes $P(\mathscr{O}_i, \mathscr{C}_j)$ ainsi que les probabilités marginales $P(\mathscr{O}_i)$ et $P(\mathscr{C}_j)$.

2.5.1 Algorithmes énumératifs

Les algorithmes énumératifs sont particulièrement bien adaptés aux modèles basés sur les chaînes de caractères par leur nature discrète (section : 2.3.1).

Le principe consiste à explorer de manière systématique toutes les chaînes de caractère correspondant à un motif, à compter le nombre d'occurrences de chaque motif et à le comparer au nombre attendu sous le modèle de base ou dans le jeu de données négatif. La sur-représentativité de chaque motif est donc évaluée à l'aide d'une fonction de score par rapport au jeu de données négatif, s'il est fourni, ou bien au compte attendu calculé à partir du modèle de base.

2.5.1.1 Fonctions de score

La fonction de score associée à un motif doit permettre d'exprimer le caractère exceptionnel de son nombre d'occurrences dans le jeu positif, par rapport à ce qui est observé dans le jeu négatif.

Considérons les deux variables aléatoires $\mathscr{O}$ (Occurrence) qui correspond à la présence/absence d'un motif donné m dans la séquence, et $\mathscr{C}$ (Condition) qui décrit les deux conditions possibles, les jeux de données positif/négatif.

À partir des effectifs correspondant à la conjonction des ces deux variables, $\mathscr{O}$ et $\mathscr{C}$, on peut construire une table de contingence qui servira à son tour au calcul de score (Table 2.2). Cette table indique le nombre de séquences qui contiennent au moins une occurrence d'un motif donné. Elle ne donne pas le nombre total d'occurrences du motif dans le jeux de données, ce qui permet d'éviter le problème des motifs auto-corrélés.

$\mathcal{O}$, Occurrence \ $\mathcal{C}$, Condition	positif 1	négatif 2	..	négatif j	..	négatif k
présence du motif m	a	b_2	..	b_j	..	b_k
absence du motif m	c	d_2	..	d_j	..	d_k

TABLE 2.3 – Table de contingence de taille $2 \times k$. La première colonne représente le jeu de donnée positif, et les autres représentent les $k-1$ jeux de données négatifs.

À partir de cette table de contingence, il est possible d'évaluer la sur-représentativité d'un motif grâce à une fonction de score. Bien entendu, différentes fonctions de score peuvent être utilisées pour accomplir cette tâche.

2.5.1.1.1 Z-score Le Z-score est le nombre d'écarts-types duquel l'observation est écartée de la moyenne. Ainsi, un score positif représente une observation supérieure à la moyenne, tandis qu'un score négatif représente une observation inférieure à la moyenne. Il est applicable sur une table de contingence de taille $2 \times k$ (k jeux de données). On considère que le jeu de données positif est représenté par la première colonne $j = 1$, les autres colonnes ($2 < j < k$) représentant les jeux de données négatifs (Table 2.3).

Le z-score est calculé avec cette formule :

$$z = \frac{x - \mu}{\alpha}$$

avec $x = \frac{a}{a+c}$ la fréquence relative des séquences contenant le motif m dans le jeu de données positif (première colonne de la table de contingence) et μ, α représentant, respectivement, la fréquence relative moyenne et l'écart-type des fréquences relatives dans les jeux de données négatifs (contrôle).

La fréquence relative (f_j) des séquences contenant le motif dans le jeu de données négatif j, la moyenne (μ) et l'écart-type (α) sont donnés par les formules suivantes :

$$f_j = \frac{b_j}{b_j + dj}$$

$$\mu = \frac{1}{k-1} \sum_{j=2}^{k} f_j$$

$$\alpha = \sqrt{\frac{1}{k-1}\sum_{j=2}^{k}(f_j - \mu)^2}$$

Parmi les outils utilisant le z-score comme fonction de score, citons RSAT [132], AMD [182] et YMF [187].

2.5.1.1.2 Test exact de Fisher Le test exact de Fisher [66] est un test statistique qui est utilisé pour mesurer l'indépendance entre deux variables aléatoires au sein d'une table de contingence. Il est basé sur les probabilités de queue de la distribution hypergéométrique. Le test de Fisher est valable pour des échantillons avec des faibles effectifs [66].

Pour une table de contingence de taille 2×2 (Table 2.2), le calcul de la probabilité se fait avec cette formule :

$$p = \frac{\binom{a+b}{a}\binom{c+d}{c}}{\binom{n}{a+c}} = \frac{(a+b)!(c+d)!(a+c)!(b+d)!}{a!b!c!d!n!}$$

où $\binom{n}{k}$ est un coefficient binomial, et $n = a+b+c+d$.

Une faible valeur p (en général < 0.05), permet de rejeter l'hypothèse nulle d'indépendance.

L'outil DREME [20], utilise la P-valeur de test exact de Fisher comme fonction de score.

2.5.1.1.3 Information mutuelle Dans la théorie de l'information, l'information mutuelle (MI) de deux variables aléatoires est une quantité mesurant leur dépendance statistique [194]. Elle se mesure en *bit*. L'information mutuelle de deux variables indépendantes est nulle, et croit lorsque leur dépendance augmente.

Dans le cadre de recherche de motifs sur-représentés, il est possible d'évaluer la dépendance d'un motif m au jeu de données. On calcule donc l'information mutuelle entre les séquences contenant le motif m et les conditions (jeux de données). Elle est définie comme suit :

$$MI(\mathscr{O};\mathscr{C}) = \sum_{i\in[0,1],j\in[\mathscr{P},\mathscr{N}]} p(\mathscr{O}_i,\mathscr{C}_j)\log_2\left(\frac{p(\mathscr{O}_i,\mathscr{C}_j)}{p(\mathscr{O}_i)\times p(C_j)}\right)$$

La variable aléatoire $Occurrence$ ($\mathscr{O}$) correspond à l'absence/présence du motif m (0 pour absence, 1 pour présence dans une séquence) et la variable aléatoire $Condition$ ($\mathscr{C}$) décrit les deux conditions possibles ($\mathscr{P}$ ou $\mathscr{N}$).

Dans le cas d'un échantillon à taille finie, les distributions de probabilité pour $p(\mathscr{O}_i)$ et $p(\mathscr{C}_j)$ sont inconnues, nous les approximons donc par leurs probabilités empiriques [76] obtenues à partir de la table de contingence (Table 2.2). $p(\mathscr{O}_i,\mathscr{C}_j)$ correspond aux probabilités conjointes, tandis que, $p(\mathscr{O}_i)$ et $p(\mathscr{C}_j)$ correspondent aux probabilités marginales.

$$p(\mathscr{O}_i,\mathscr{C}_j) \approx \frac{\#\left(\mathscr{O}_i\ ,\mathscr{C}_j\right)}{\sum_{i,j}\#(\mathscr{O}_i\ ,\mathscr{C}_j)}$$

Discrover [123] dans son étape de recherche de graines, nommée Plasma [123] et Fire [62] sont deux outils qui utilisent l'information mutuelle comme fonction de score.

2.5.1.2 Algorithmes Énumératifs

L'énumération des motifs peut être exhaustive et exacte, ou bien limitée à un sous-ensemble de motifs.

2.5.1.2.1 Énumération gloutonne L'énumération et le test de tous les motifs d'une taille donnée est difficilement réalisable pour des grandes tailles de motifs ($L > 6$), car le nombre de motifs croit exponentiellement en fonction de leur taille : par exemple, il existe $n = 15^L$ motifs ADN dégénérés d'une taille l, en utilisant tout l'alphabet IUPAC. Pour la taille $L = 8$, il existe $15^8 \approx 2.6 \times 10^9$ motifs possibles).

Une façon de simplifier cet espace de recherche est d'utiliser des algorithmes gloutons. Le principe consiste à sélectionner des choix localement optimaux à chaque étape. FRAENKEL *et al.* [69] ont publié un algorithme qui énumère tous les motifs exacts sur l'alphabet A,C,G,T d'une petite taille (< 10) avec un nombre maximal de mismatches autorisés. Le modèle utilisé est donc celui de « mismatch » (cf. section 2.3.1). D'autres outils comme DREME [20], FIRE [62] et Plasma [123] cherchent les motifs exacts sur-

représentés en premier temps, puis dégénèrent les k premiers motifs, position par position, en essayant de remplacer le nucléotide A, C, G, T par une lettre IUPAC dégénérée. Si le motif avec la lettre dégénérée est plus significatif, ils le conservent et passent à l'itération suivante (autre position) et ainsi de suite. Une autre approche gloutonne consiste à énumérer les motifs d'une petite taille, puis à les étendre des deux côtés et de tester à chaque étape la significativité du motif. C'est le cas par exemple des deux outils AMD [182] et SeAMotE [14].

Cette approche permet donc d'explorer des motifs relativement longs (8 à 10 nucléotides) avec des grands jeux de données (séquençage haut-débit), tout en utilisant l'ensemble de l'alphabet IUPAC. Mais par sa nature gloutonne (stratégie localement optimale), beaucoup de possibilités ne sont pas explorées, et rien ne garantit de trouver un motif qui soit optimal de manière globale.

2.5.1.2.2 Énumération exacte

Génération de tous les motifs Les outils utilisant cette approche ont été les premiers à être développés. QUEEN *et al.* [159] étaient les premiers à énumérer et chercher des motifs simples d'une façon exacte, en utilisant le modèle « mismatch » (consensus), et en autorisant un nombre d'erreurs maximal (e) dans le motif. On peut aussi citer Oligo-Analysis développé par van HELDEN *et al.* [200] et Oligo-Diff [132], conçus pour la recherche des TFBS. Cet outil calcule la fréquence de chaque oligo (motif exact), avec la probabilité d'occurrence selon la loi binomiale en se basant sur les fréquences attendues. Ces dernières sont calculées à partir de l'ensemble des régions régulatrices du génome de la levure. Cette approche a permis de détecter des TFBS validés expérimentalement. Toutefois, elle est limitée à la recherche des motifs courts, de taille 6 pour cette étude, et simples avec une séquence exacte en représentant les motifs sous forme d'oligo-mers. TOMPA [198] a publié une méthode similaire basée sur le calcul de z-score en comparant les occurrences de chaque motif au nombre d'occurrences attendu, calculé à partir des séquences aléatoires générées par une chaîne de Markov se basant sur la distribution des di-nucléotides. De même, l'outil YMF a été développé par SINHA & TOMPA [187] en se basant sur le z-score, mais en autorisant des N au milieu de leurs motifs appelés *spacers*, et en autorisant les lettres R,Y,W et S de l'alphabet IUPAC dans les deux extrémités du motif (par exemple : YCCNNNNCCT). HELDEN *et al.* ont aussi

développé un outil, appelé *dyad-analysis* [85], utilisant les *spacers* au milieu des motifs, sans l'utilisation de l'alphabet IUPAC complet pour dégénérer le reste du motif. Ces deux derniers outils sont adaptés à des motifs où la partie dégénérée est située au centre. Dans la même catégorie, on trouve également HOMER [84] qui utilise, à la place du z-score, la distribution hypergéométrique pour évaluer l'enrichissement des motifs. HOMER n'impose pas de contraintes sur la position des nucléotides dégénérés, mais son alphabet IUPAC n'inclut pas les lettres dites de troisième niveau (B,D,H,V, Table 2.1) pour réduire son espace de recherche.

Tous ces outils imposent donc des contraintes spécifiques afin de réduire l'espace de recherche de motifs (alphabet IUPAC réduit, taille de motif, position de lettres dégénérés, ...). Ils ont été développés au départ pour l'étude des régions régulatrices dans le génome de la levure avant l'ère du séquençage haut-débit. L'augmentation exponentielle de la quantité des données à analyser ces dernières années a rendu la plupart de ces outils non adaptés au volume des données. L'utilisation des structures plus sophistiquées s'avère donc indispensable pour la recherche exacte des motifs plus complexes dans des jeux de données qui ne cessent d'augmenter en taille.

Génération limitée aux motifs possibles Pour éviter l'énumération de tous les motifs, certains outils utilisent des structures de données plus adaptées notamment des structures d'indexation, telles que les arbres de suffixes, les tableaux de suffixes ou bien des automates [45, 201]. Ces structures peuvent être adaptées pour l'extraction d'un ensemble de mots communs à un ensemble de séquences. Le principe de ces méthodes consiste à calculer le nombre d'occurrences de chaque motif exact trouvé dans le jeu de données dans un premier temps, puis à calculer par la suite le nombre d'occurrences des motifs dégénérés possibles en sommant les occurrences des motifs exacts correspondants. L'inconvénient des méthodes énumératives est que celles-ci génèrent beaucoup de motifs équivalents qui sont souvent des représentations multiples du même motif. C'est pourquoi on a souvent recours à une étape supplémentaire de regroupement de motifs (*clustering*) afin de synthétiser le résultat.

L'algorithme de Sagot [164] était le premier à avoir utilisé le modèle IUPAC (expression régulière) pour chercher des motifs complexes et dégénérés en utilisant une partie ou l'intégralité de l'alphabet IUPAC. Toutefois, il manquait l'étape finale de re-

groupement, et les motifs résultants étaient trop nombreux pour être tous analysés et exploités. Dans cette catégorie d'algorithmes, on peut citer l'outil Weeder [151] qui utilise l'arbre des suffixes pour accélérer le calcul. Il utilise le modèle *mismatch* pour la recherche des motifs. La séquence consensus gardée est celle qui a la meilleur valeur de z-score. PRATT [94] utilise également une structure d'index, à base de graphes, pour explorer exhaustivement les motifs, mais cet outil est optimisé pour les motifs courts et très conservés [127].

Une autre approche présentée par MARSCHALL & RAHMANN [128] consiste à élaguer l'espace de recherche des motifs, en exploitant les propriétés de monotonie de l'approximation de Poisson composée [170]. Cette propriété permet d'écarter un ensemble d'instances dont leur nombre d'occurrences et plus petit que le nombre d'occurrences k minimal pour avoir une p-valeur significative. L'inconvénient de cet outil, c'est qu'il ne permet par de faire une recherche de motifs discriminants entre deux ensembles de séquences. Il cherche des motifs sur-représentés par rapport à un modèle de base de type *IID*, qui est trop simple pour modéliser des séquences nucléotidiques. Il est possible d'utiliser un modèle Markovien, mais la procédure dans ce cas est heuristique, et se base sur une première étape de filtrage qui consiste à évaluer rapidement l'ensemble de motifs sous le modèle *IID* en utilisant le nombre d'occurrences minimal requis (k) en première étape, puis à re-évaluer les motifs trouvés (significatifs sous *IID*) sous un modèle de Markov. Il est donc possible de manquer des motifs qui seraient significatifs par rapport à un modèle de base Markovien et pas le modèle *IID*. De plus, certaines règles sont imposées pour le modèle IUPAC utilisé, en restreignant le nombre de lettres dégénérées dans le motif (au maximum : 6 lettres de second degré (R, Y, W, S, K, M), 0 de troisième degré (B, D, H, V) et deux de quatrième degré (N)).

2.5.2 Algorithmes probabilistes

Les algorithmes probabilistes sont utilisés pour chercher et optimiser des motifs représentés par des modèles probabilistes (cf. section 2.3.2). Étant donné qu'il s'agit de représentations avec des variables continues, il est impossible d'énumérer et tester tous les motifs possibles à l'aide des approches combinatoires citées précédemment. Le principe de cette seconde catégorie d'algorithmes est basé sur le maximum de vraisemblance, pour les outils les plus utilisés. La méthode démarre en général avec un motif

aléatoire et procède à son optimisation en effectuant des modifications successives jusqu'à l'obtention d'un motif sur-représenté et relativement stable.

Deux algorithmes principaux sont utilisés pour l'optimisation des motifs continus. Le premier, et sans doute le plus utilisé, est l'algorithme « Espérance-Maximisation » (EM). Le deuxième, qui est une variante du premier, est l'algorithme de Gibbs sampling.

2.5.2.1 L'algorithme Espérance-Maximisation

C'est un algorithme itératif qui permet de trouver les paramètres du maximum de vraisemblance d'un modèle probabiliste, autrement dit, les paramètres de la matrice PWM ou d'une chaîne de Markov cachée qui représentent un motif complexe (dégénéré). L'outil le plus connu utilisant cet algorithme est MEME [21]. L'algorithme EM est déterministe, cela signifie que pour chaque lancement, avec les mêmes entrées, on obtient toujours les mêmes résultats.

Dans le cas d'une matrice PWM, on initialise l'algorithme avec une matrice PWM aléatoire. Ceci est fait en choisissant une position aléatoire par séquence, comme position d'une instance du motif.

Chaque itération de l'algorithme est composée des deux étapes (Figure 2.5) :

2.5.2.1.1 Étape d'espérance (E) Elle consiste à estimer la matrice PWM à partir des motifs qui se trouvent aux positions sélectionnées à l'itération précédente. Les motifs sont alignés et les fréquences de chaque nucléotide pour chaque colonne de la matrice PWM sont calculées.

2.5.2.1.2 Étape de maximisation (M) Chaque position de chaque séquence est par la suite évaluée avec la matrice obtenue dans l'étape précédente en calculant sa probabilité sous la matrice PWM obtenue dans l'étape espérance. Pour chaque séquence, la position ayant la plus grande probabilité d'être générée par la matrice est sélectionnée. L'ensemble des nouvelles positions est alors utilisé dans l'étape espérance de l'itération suivante.

L'algorithme est lancé pour plusieurs itérations jusqu'à la convergence sur un motif stable, quand l'ensemble des positions sélectionnées ne change plus. L'algorithme peut

s'arrêter avant la convergence si le nombre d'itérations maximal fixé dans les paramètres de l'outil est atteint.

L'algorithme de Baum-welch est une variante de l'algorithme EM, qui est adaptée aux motifs représentés par un HMM [51, 123]. De la même façon, à chaque itération, les paramètres de la HMM sont mis à jours à partir de l'itération précédente en considérant tous les chemins possibles de la HMM pour chaque séquence dans le jeu de données d'apprentissage. L'algorithme s'arrête quand un nombre fixe d'itérations est atteint, ou bien quand le changement de la vraisemblance devient suffisamment petit.

L'inconvénient principal de cette approche est qu'il n'y a aucune garantie d'obtenir le maximum global. Cela est particulièrement vrai lorsqu'on utilise les HMM comme modèles pour représenter les motifs, à cause du nombre élevé de paramètres à estimer. En plus, le résultat obtenu dépend fortement du jeu de données initial, ce qui peut également conduire l'algorithme à un point de convergence inapproprié.

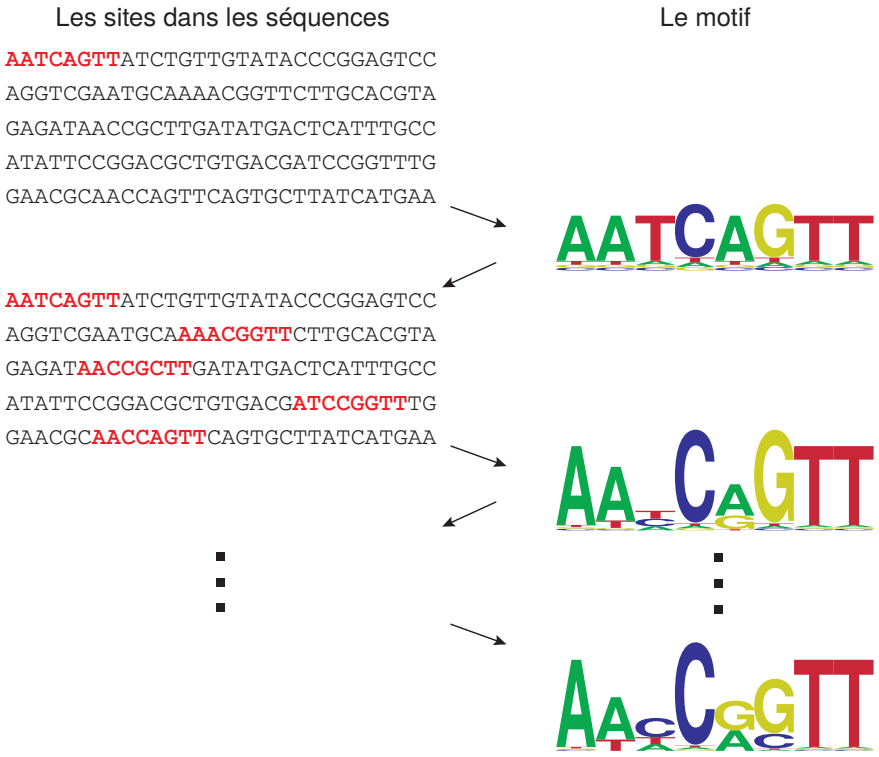


FIGURE 2.5 – Recherche de motif par EM. À partir d'un seul site, l'algorithme d'EM alterne entre l'attribution de sites à un motif (en rouge, à gauche) et la mise à jour du modèle de motif (à droite). Figure adaptée de : [55]

2.5.2.2 L'algorithme *Gibbs sampling*

L'algorithme *Gibbs sampling* est une variante stochastique de l'algorithme d'EM, dont le but est de réduire l'impact du jeu de données initial sur la convergence du modèle. Cette approche a été développée par LAWRENCE *et al.* [112] et implémentée dans plusieurs outils tel que MotifSampler [193] et BioProspector [119]. Ce dernier apporte quelques modifications par rapport à MotifSampler, notamment un ordre plus élevé pour la chaîne de Markov utilisée comme modèle du fond.

À la différence de l'algorithme EM classique, on choisit à chaque itération une séquence aléatoire X_i, et on estime les paramètres d'une PWM à partir des autres séquences en excluant X_i. Par contre, l'étape de maximisation est elle effectuée uniquement sur la séquence X_i. Chaque position de cette séquence est donc évaluée avec la matrice PWM, et un poids est ainsi attribué à chacune de ces positions en fonction de sa probabilité sous la matrice PWM. Une étape d'échantillonnage est ensuite effectuée pour choisir une position tout en respectant les poids attribués. Plus la probabilité d'une position est élevée, plus elle a de chance d'être choisi, contrairement à l'approche EM, qui choisit systématiquement la position avec la meilleur probabilité. Ce processus est répété jusqu'à convergence.

Le *Gibbs sampling* est aussi utilisé par nombreux outils pour l'estimation des paramètres des HMM[117, 144].

LAWRENCE *et al.* ont prouvé que l'algorithme de Gibbs sampling convergeait toujours, mais pas nécessairement vers un maximum global [201].

2.6 Conclusions

Chaque approche décrite précédemment présente des avantages et des inconvénients. Les méthodes énumératives exactes testent exhaustivement tous les motifs possibles et garantissent l'obtention d'un optimum global. Ceci impacte le temps de calcul qui augmente significativement avec la longueur et la complexité des modèles choisi (degré de dégénérescence). Les algorithmes probabilistes permettent de chercher rapidement des motifs avec des modèles continus complexes (PWM ou HMM), mais le résultat optimal n'est pas garanti et dépend fortement du jeu de données (nécessite un

grand jeu de données). De ce fait, ces derniers ne sont pas adaptés pour le recherche des motifs rares.

Un autre point important à souligner est la recherche multiple de motifs. Un jeu de données peut contenir plusieurs motifs sur-représentés, comme c'est le cas des analyses ChIP-seq où les séquences peuvent contenir plusieurs motifs différents correspondant à des sites différents. Afin de les trouver tous, les outils effectuent en général plusieurs itérations. À chaque itération, le motif trouvé est masqué dans le jeu de séquences afin de permettre l'identification d'un nouveau motif. Le fait de relancer le programme plusieurs fois peut augmenter significativement le temps de calcul.

Deuxième partie

Erreurs de séquençage et objectifs de la thèse

CHAPITRE 3

LE PROBLÈME DES ERREURS DE SÉQUENÇAGE

Dans ce chapitre, nous présentons en détails la problématique des erreurs de séquençage dans le cadre du séquençage de nouvelle génération, et le traitement bioinformatique de ces erreurs pour l'identification des variants. Cela nous conduit à formuler en fin de chapitre les objectifs généraux de la thèse.

3.1 Les erreurs de séquençage

Les technologies de séquençage à haut-débit présentent un taux d'erreur plus élevé que le séquençage conventionnel de Sanger, et qui varie approximativement entre 0,1 et 1% pour les séquenceurs dits de seconde génération (Table 3.1). L'origine de ces erreurs est multiple. Celles-ci peuvent survenir lors de l'étape de préparation de la librairie ou bien durant le processus de séquençage [75]. Lors de la préparation de l'échantillon, des erreurs peuvent être induites par l'ADN polymérase pendant l'étape de PCR utilisée pour l'amplification du matériel génétique [95]. Des erreurs préalables au séquençage peuvent également apparaître durant l'amplification clonale des fragments d'ADN sur le support de séquençage (génération des *clusters* ou génération des *ISP*). Durant l'étape de séquençage, les erreurs sont dûes à la mauvaise incorporation de bases par l'ADN polymérase ou à des erreurs du traitement du signal. Le taux d'erreur n'est pas constant le long du read. Il augmente considérablement avec la position de la base le long du *read*

ou en fonction du type d'erreur (insertion versus substitution). Outre ces erreurs survenues pendant la préparation des librairies ou au moment du séquençage, des erreurs dans l'appel des variants peuvent être liées également à des alignements incorrects des *reads*.

Ces erreurs de séquençage compliquent l'analyse des données, car elles peuvent être confondues avec des variants de faible ratio allélique, induisant ainsi des mauvais appels de variants (faux positifs). L'impact des erreurs de séquençage sur la qualité d'appel des variants peut être diminué en effectuant un séquençage avec une profondeur élevée et par l'utilisation de plusieurs filtres qui sont basés principalement sur des critères de comptage des *reads*. Les filtres communément appliqués par les programmes d'appel de variants sont les suivants [115] :

1. Filtre des régions de faible complexité (LC) : filtrer les régions du génome composées de peu de bases différentes (répétitions de AT par exemple). Ces régions induisent beaucoup d'erreurs d'alignement et par conséquent d'appel de variants. Des outils comme RepeatMasker [13] sont utilisés pour la détection de telles régions et pour filtrer les variants identifiés dans ces régions.
2. Filtre de profondeur maximale : filtrer les sites couverts par un nombre excessif de *reads*. Les faux positifs issus des régions avec une forte profondeur sont principalement dûs aux variants de type CNV ou à l'existence de séquences paralogues qui ne sont pas présentes dans le génome de référence humain utilisé, ce qui induit des erreurs d'alignement. Li [115] a proposé d'utiliser un seuil de profondeur maximale égale à $d + 3\sqrt{d}$, avec d la profondeur moyenne du *run*, pour réduire le taux de faux positifs.
3. Biais de brin : filtrer les variants dont la détection est corrélée à la direction de lecture, c'est-à-dire que le variant est plus fréquemment détecté dans une direction de séquençage (*sens* ou *anti-sens*) par rapport à un autre. En général, une table de contingence de taille 2×2 est construite avec le nombre des *matches*/*mismatches* sur les deux directions de lecture forward/reverse, et la corrélation est évaluée avec le test exact de Fisher.
4. Filtre de ratio allélique : filtrer les variants dont la VAF est inférieure à un seuil donné. Ce filtre peut être appliqué sur l'ensemble des *reads* ou en testant séparément les *reads forward* et *reverse*. Ce filtre peut être utilisé pour la recherche de

variants constitutionnels, mais induit des risques de faux négatifs dans le cadre de la recherche de variants somatiques ou en mosaïque.

5. Filtre de qualité : éliminer les variants dont la qualité est inférieure à un seuil fixé. La qualité d'un variant est obtenue en appliquant une formule (moyenne arithmétique par exemple) à la qualité du signal obtenu en sortie du séquenceur pour la lecture du nucléotide.

Plate-forme	**Type d'erreurs le plus fréquent**	**Taux d'erreurs**
454 GS Junior	Délétions	1%
IonTorrent PGM	Délétions courtes	1%
Illumina MiSeq	substitutions	0,1%
Illumina HiSeq2000	substitutions	0,1%

TABLE 3.1 – Taux d'erreurs par technologie de séquençage [68].

Toutefois, quand on recherche des variants de faible ratio allélique, l'application de tels filtres devient problématique. De tels variants peuvent être recherchés dans un échantillon mixte pouvant contenir du matériel génétique de différentes origines comme dans le cadre d'une analyse somatique des mosaïques [44, 156, 157] et de tumeurs hétérogènes [146] où un seuil de VAF à 1% est nécessaire. Un niveau de détection plus bas, allant jusqu'à 0,1%, est parfois souhaité pour d'autres applications comme dans le cadre de suivi de la maladie résiduelle [49], la métagénomique [113, 125], le dépistage prénatal de l'aneuploïdie fœtale [37, 63] et l'analyse d'ADN circulant [24].

Pour arriver à une telle sensibilité et pouvoir détecter des variants ayant un faible ratio allélique, la méthode de consensus basée sur une grande profondeur de séquençage devient insuffisante. Des logiciels spécialisés dans la recherche des variants somatiques, en utilisant des modèles statistiques plus sophistiqués, ont été développés. Ces modèles se basent principalement sur la profondeur des variants. Parmi les plus connus, on peut citer MuTect [39], HaplotypeCaller [53], LoFreq [206], VarScan [104] et FreeBayes [71]. Ces logiciels ont été largement utilisés et comparés dans d'autres études [34, 118, 166, 208]. Ils sont capables de détecter des variants ayant une faible ratio allélique avec une bonne sensibilité (VAF allant jusqu'à 1%), en diminuant les seuils de filtres basés sur la couverture des variants. Pour garder une bonne spécificité tout en diminuant les seuils de détection, il est nécessaire d'effectuer un séquençage ultra-profond ($> 1000\times$)

3.2 Erreurs de séquençage non aléatoires

Le séquençage ultra-profond permet de résoudre le problème des erreurs de séquençage aléatoires, c'est-à-dire qui apparaissent de manière indépendante entre les *reads*. À côté de cela, il existe des erreurs *non aléatoires*, appelées SSE (pour *Sequence-Specific Errors*), qui restent un problème majeur pour les plateformes de séquençage haut-débit [26, 141, 171, 214]. Contrairement aux erreurs aléatoires, les SSE se produisent plusieurs fois à la même position. De ce fait, il est difficile de les distinguer des vrais variants génomiques en utilisant uniquement l'information de la profondeur de séquençage. De plus, il a été montré que l'augmentation de la profondeur du séquençage ne fait qu'aggraver le problème et augmenter le nombre de faux positifs (c'est-à-dire les erreurs systématiques considérées comme de vrais variants) [202, 211].

Il existe des outils qui prennent en considération la position du variant dans l'étape d'appel de variants. C'est le cas des logiciels DeepSNV [73], EBcall [184], LoLoPicker [31], MERIT [81] et OutLyzer [139] qui estiment un taux d'erreur qui est spécifique à la position. Ces outils estiment le modèle d'erreur à partir de différents données. EBcall par exemple, estime les paramètres du modèle d'erreur à partir d'autres jeux de données de contrôle, tandis que DeepSNV le fait à partir d'un échantillon normal et un autre tumoral du même *run*. À partir du modèle d'erreur, position-spécifique, ils évaluent le variant en utilisant un test statistique comme le test Binomial (DeepSNV et LoLoPicker) ou le test de Tau de Thompson (OutLyzer).

L'utilisation d'un modèle d'erreur qui est position-spécifique est particulièrement utile pour l'appel de variants dans des échantillons de mauvaise qualité comme c'est le cas des tissus fixés à la formaldéhyde et inclus en paraffine (*Formaldehyde-Fixed and Paraffin Embedded*, FFPE), où le taux d'erreur est plus élevé et variable d'un site à un autre en comparaison à un échantillon frais [208]. Toutefois, l'estimation du taux d'erreur spécifique à la position nécessite un séquençage d'un grand nombre d'échantillons, ce qui n'est pas toujours faisable.

NAKAMURA *et al.* [141] ont montré que les erreurs systématiques dépendent du contexte nucléotidique en amont plutôt que de la position, et qu'il existe des motifs surreprésentés en amont des positions d'erreur de séquençage dans les *reads* Illumina, en particulier le motif GGT (Figure 3.1).

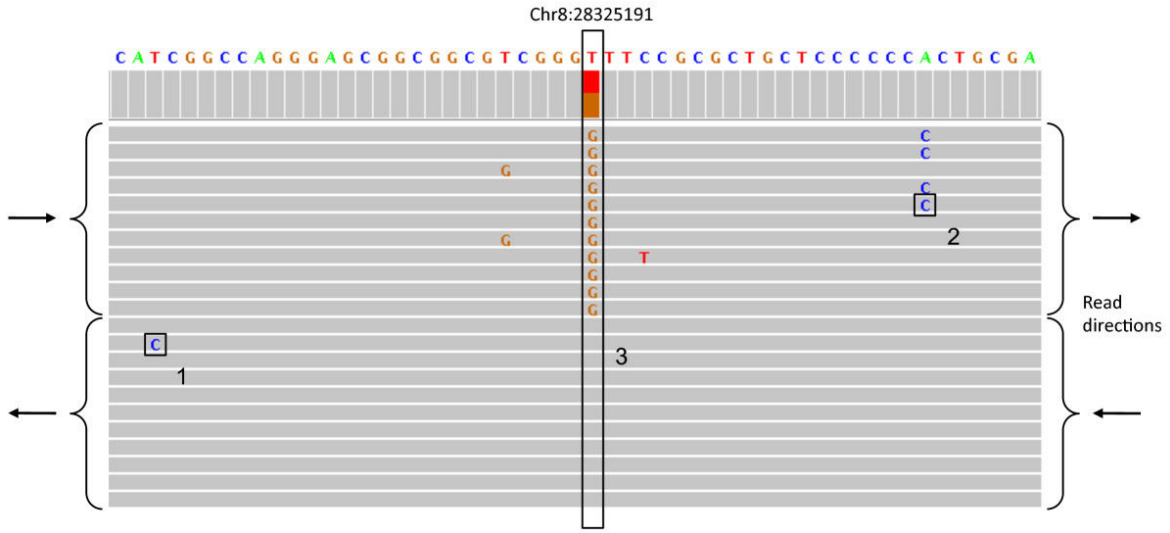


FIGURE 3.1 – Types d'erreurs de séquençage. Trois types d'erreurs dans les *reads* obtenu par séquençage Illumina : (1) Une erreur aléatoire. (2) Un variant à faible ratio allélique ou bien une erreur non aléatoire. (3) Erreur systématique, probablement dû au motif GGT. Source de la figure : [131].

Peu d'outils ont été développés pour l'analyse du contexte nucléotidique en amont, notamment pour la recherche des motifs, afin d'améliorer la qualité d'appel des variants et mieux distinguer les vrais variants des erreurs de séquençage systématiques. Une recherche bibliographique permet de recenser quatre outils d'analyse d'erreurs systématiques : SSECF, SysCall, Discovering-cse, GATK/BQSR.

3.2.1 SSECF

SSECF [183] est un outil de filtrage de *reads* à base de motifs. Le principe de l'outil consiste à aligner chaque *read* contre la séquence de référence afin de répertorier tout les *mismatchs*. Les séquences flanquantes (en amont et en aval conjointement) des positions des *mismatchs* sont extraites par la suite, pour l'inférence de motifs qui sont sur-représentés. Les auteurs proposent d'utiliser ces motifs pour filtrer des *reads* contenant des erreurs de séquençage potentielles, sans proposer une méthode claire pour le faire.

Leur méthode est limitée aux motifs exacts, sans aucune dégénérescence. Malheureusement, il n'existe aucun outil utilisable facilement pour analyser les données. Les scripts fournis permettent l'analyse des *reads* issus de la technologie FLX, FLX+, ou GS

Junior, technologies de séquençage qui ne sont plus maintenues.

3.2.2 SysCall

SysCall [131] est un classificateur par régression logistique. Étant donné un alignement de *reads* sur un génome de référence et une position l, SysCall construit un vecteur x_l, comme suit : tout d'abord, la direction de séquençage qui contient la plus grande proportion de bases différentes de la référence est choisie (*forward* ou *reverse*). SysCall considère uniquement les positions auxquelles il y a au moins une base qui diffère de la référence. Soit q_{l1} et q_{l2} les proportions des bases différentes dans les directions choisies et non choisies respectivement. Soit b_i le nucléotide qui est à i positions de l dans la direction choisie et w_i le vecteur de scores de qualité à la position de b_i. Un vecteur de caractéristiques x_l est ensuite annoté pour l, avec :

$$x_l = \left(b_{-2}, b_{-1}, b_0, q_{l1} - q_{l2}, q_{l1}, PT(w_0, w_1)\right)$$

où $PT(w_0, w_1)$ est le résultat du *test-t apparié* [212] sur les deux vecteurs w_0 et w_1.

Les paramètres de SysCall sont estimés à partir d'un jeu de données d'entraînement obtenu par un séquençage de type méthyl-Seq, dans lequel il est possible de distinguer les variants hétérozygotes des erreurs systématiques grâce au chevauchement des *reads* pairés, chaque paire étant lue dans un sens différent. Différentes profondeurs sont utilisées pour l'estimation des paramètres de SysCall (7×, 14×, 21× et 28×).

SysCall prend en entrée la liste de coordonnées génomiques des variants et évalue pour chacun la probabilité qu'il soit une erreur ou bien un vrai variant en utilisant les paramètres du modèle obtenus du jeu d'apprentissage avec la couverture la plus proche de celle du jeu de données à analyser (Figure 3.2).

Cet outil présente plusieurs limites. Il n'est utilisable qu'avec des données Illumina compatibles avec celles du jeu d'apprentissage : profondeur de séquençage proche et utilisation de la même chimie. SysCall, du fait de l'utilisation de jeu d'apprentissage spécifique (étude de la méthylation avec chevauchement des paires de reads) ne permet pas de regénérer un jeu d'apprentissage et de s'adapter à chaque *run* alors que [48, 180, 214] montrent que les erreurs de séquençage non aléatoires peuvent être variables d'un

run à un autre. D'autre part, SysCall ne prend en compte qu'un contexte nucléotidique de taille 3.

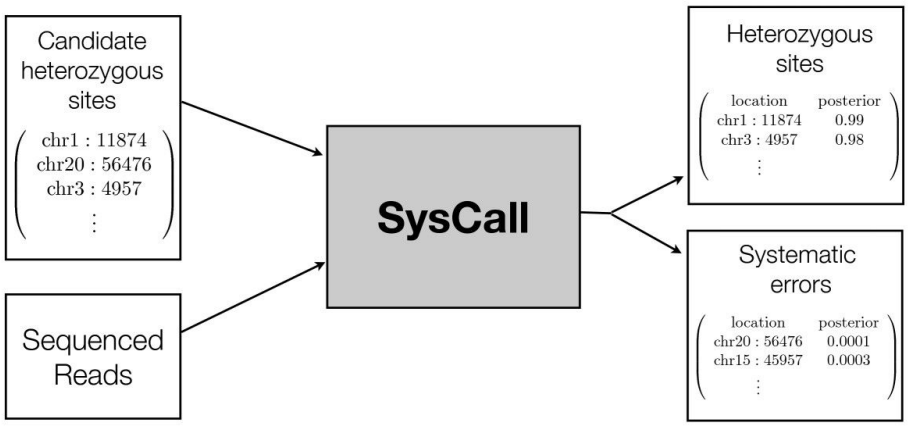


FIGURE 3.2 – Le processus de classement des variants par le logiciel *SysCall* [131]. Il prend en entrée un fichier de variants à classer et des *reads* d'un jeu de données de référence (jeu de données d'apprentissage), et donne en sortie deux fichiers dont un contenant ce qu'ils considère comme des vrais variants et un autre contenant les erreurs de séquençage (SSE).

3.2.3 Discovering-cse

ALLHOFF *et al.* [16] ont publié un outil qui recherche de motifs induisant l'apparition d'erreurs de séquençage. Leur approche se base sur le biais de brin, autrement dit, sur le déséquilibre de représentation d'un variant sur le brin sens ou anti-sens.

Pour un motif donné m, ils localisent toutes ses occurrences dans le génome de référence. Ensuite, une table de contingence agrégée est construite, dont les entrées sont nommées a, b, c, d comme dans la table 3.2 et la figure 3.3.

	Match	Mismatch	Total
Forward	a	b	f
Backward	c	d	k
Total	m	s	n

TABLE 3.2 – Table de contingence construite pour chaque motif 2×2 ; a, b, c, d : nombre de *reads* ; f, k, m, s : marginales ; $n = a + b + c + d = f + k = m + s$.

Le test exact de Fisher [66] est ensuite appliqué sur ces tables de contingence afin de calculer une P-valeur et de trouver les motifs qui favorisent l'apparition des erreurs

de séquençage.

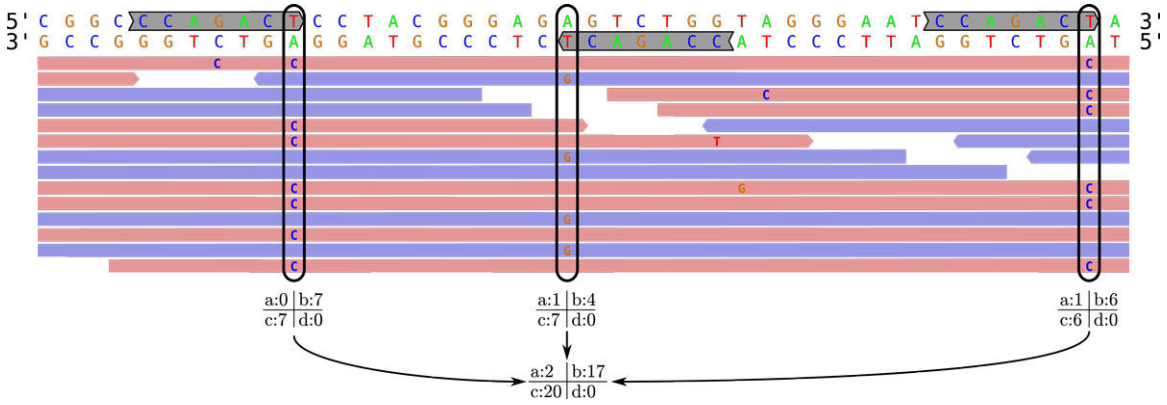


FIGURE 3.3 – Table de contingence agrégée construite pour le motif CCAGACT. Les *reads forward* en rouge, Les *reads reverse* en bleu. Source de la figure : [16]

Plusieurs jeux de donnés ont été testés dans leur étude, ils correspondent à différents séquenceurs, tous de type Illumina. Les motifs trouvés sont constitués principalement de GGT (Table 3.3), ce qui confirme les motifs précédemment trouvés par NAKAMURA *et al.* [141].

Bien que leur méthode permette de découvrir des motifs liés aux erreurs de séquençage, elle reste non optimale pour plusieurs raisons : elle permet de découvrir les motifs mais n'attribue pas de score aux variants, il est donc impossible de classer les variants en erreurs ou vrais variants ; leur approche est limitée aux motifs exacts tout en autorisant seulement la lettre N et non la totalité de l'alphabet IUPAC. Finalement, leur outil n'est pas adapté à la taille d'un génome humain, vu qu'il n'est capable d'analyser qu'un seul chromosome à la fois, ce qui limite sa puissance statistique.

3.2.4 GATK/BQSR

L'outil GATK (*Genome Analysis Toolkit*) [53] réalise une étape de re-calibration des bases durant le processus d'appel de variants. Cette étape est effectuée par le module BQSR (*Base Quality Score Recalibration*).

Le BQSR est un processus basé sur l'apprentissage automatique pour modéliser empiriquement les erreurs et ajuster le score de qualité des bases. Il est composé de deux étapes. La première étape consiste à construire le modèle de co-variation en se

Rang	Motif
1	ACGGCGGT
2	GTGGCGGT
3	GCGGCGGT
4	GTGGCTGT
5	ATGGCGGT
6	NCGGCGGT
7	GTGGCTTG
8	GNGGCGGT
9	GCGGCTGT
10	ACGGCTGT
	GGC

TABLE 3.3 – Les 10 premiers motifs trouvés avec la technologie GAIIX, en séquençant un génome de bactérie par ALLHOFF *et al.* [16]. Les motifs en une taille de 8 avec 4 bases dégénérées (N) possibles. On remarque la présence d'une « graine » commune à tous les motifs GGC.

basant sur les données de séquençage et un jeu de données de variants connus (base de données publique de polymorphismes comme dbSNP [181]). La deuxième étape consiste à ajuster les scores de qualité de chaque base dans les données en se basant sur le modèle construit dans la première étape (Figure 3.4). Étant donné que les outils d'appel de variants se basent fortement sur la qualité des bases, cette étape de re-calibration améliore la qualité d'appel des variants et limite les faux positifs.

Quatre caractéristiques sont prises en compte dans le modèle de co-variation :

— Le *read group* du *read*

— Le score de qualité de la base

— Le cycle de la machine produisant cette base

— La base + la base précédente (dinucléotide)

Autrement dit, GATK prend en compte les motifs de taille 2 (dinucléotide) pour la re-calibration des bases des SNV et des motifs de tailles 3 pour les InDels. Son principal

défaut est qu'il est limité à l'étude des dinucléotides et ne permet pas l'exploration des motifs plus longs et complexes.

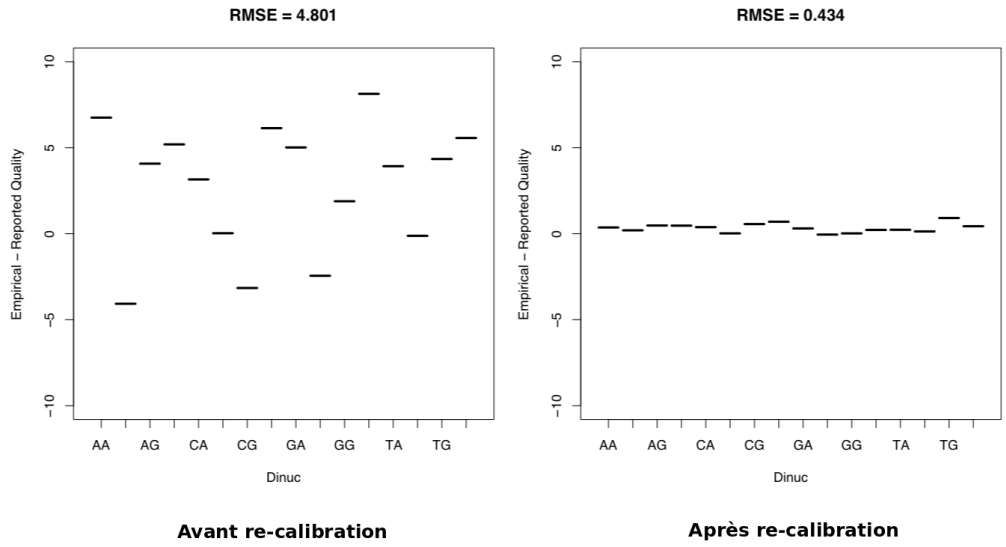


FIGURE 3.4 – La racine carrée de l'erreur type (RMSE, *Root-Mean-Square Error*) de la qualité des bases après chaque dinucléotide. La RMSE représente l'estimation des différences entre les valeurs de qualité prédites par le modèle de re-calibration de BQSR et les valeurs observées. On remarque qu'à droite, après re-calibration de bases par BQSR, l'écart entre les valeurs observées et les valeurs prédites diminue. Source de la figure : [10]

3.3 Objectifs de la thèse

Comme expliqué dans le chapitre 1, les séquenceurs de nouvelle génération sont caractérisés par leur débit élevé permettant l'utilisation du séquençage dans des nouvelles applications, notamment la recherche de variants génomiques à l'échelle de panels de gènes, voir du génome entier. Ces séquenceurs sont aussi caractérisés par un taux d'erreur, plus grand que celui du séquençage classique Sanger, pouvant aller jusqu'à 1% d'erreur. Ce taux d'erreur est problématique dans le cadre de la recherche de variants et spécifiquement les variants somatiques ayant un très faible ratio allélique et qui peuvent être confondus avec les erreurs de séquençage.

Plusieurs outils spécialisés dans la recherche de variants à faible ratio alléliques ont été développés (cf. section 3.1) et permettent de distinguer les vrais variants des erreurs de séquençage en utilisant des modèles probabilistes basés sur les comptages de *reads*. En dépit de cela, un type d'erreurs reste difficile à distinguer des vrais variants, ce sont les erreurs de séquençage non-aléatoires.

Pour bien identifier les erreurs de séquençage systématiques, des nouvelles approches dédiées ont été développées : l'approche statistique qui consiste à estimer un taux d'erreur spécifique à la position des variants, et l'approche combinatoire qui consiste à analyser le contexte nucléotidique en amont des erreurs de séquençage à la recherche des motifs qui sont potentiellement responsables de leur apparition. L'approche combinatoire permet d'augmenter la puissance statistique en combinant toutes les positions ayant le même contexte nucléotidique, à l'inverse de l'approche statistique qui nécessite des grands jeux de données afin de bien estimer les paramètres du modèle d'erreurs.

Toutefois, les outils existant pour la recherche de motifs liés aux erreurs de séquençage présentent deux principales limites :

— Le choix d'un modèle de fond : l'outil SysCall [131] par exemple, utilise un modèle de fond construit à partir de jeux de données d'entraînement obtenu par un séquençage Illumina. Cet outil est donc utilisable exclusivement pour l'analyse de données Illumina obtenu avec une chimie similaire. Le modèle de fond limite alors le spectre d'utilisation de l'outil. De plus, il a été montré que les erreurs systématiques (SSE) sont parfois spécifiques au *run* [48, 180, 214], ce qui nécessite un outil rapide capable d'analyser les données de chaque *run* en temps raisonnable,

au lieu d'utiliser un jeu de donnée indépendant.

— La complexité des motifs cherchés (taille et niveau de dégénérescence) : la généralisation des motifs permet deux grandes améliorations. Du point de vu informatif, les bases dégénérées permettent de savoir quelle sous-partie du motif est plus impliquée dans la génération d'erreur. Du point de vue statistique, les motifs dégénérés sont une combinaison de plusieurs motifs exacts, permettant ainsi d'augmenter les occurrences et par conséquence, d'augmenter le pouvoir statistique et la sensibilité sans devoir augmenter la taille du jeu d'apprentissage, ce qui est souvent le cas dans les applications cliniques où on réalise du séquençage ciblé avec potentiellement une faible diversité dans les régions séquençées. Les motifs dégénérés permettent alors de détecter des motifs qui sont faiblement présents dans le jeu de données. L'augmentation de la puissance statistique permet également d'augmenter la longueur des motifs recherchés, permettant à son tour d'augmenter la spécificité des motifs identifiés et évite l'effet additif des motifs normaux [16, 97].

Le chapitre 2 présente les différents modèles et algorithmes utilisés pour la recherche de motifs d'ADN plus complexe. Ces différentes approches ont été utilisées pour la recherche de motifs d'ADN dans le cadre d'une application pour la recherche de sites de fixation de facteur de transcription, mais n'ont jamais été appliqués à la recherche de motifs liés aux erreurs de séquençage systématique. Ces outils nécessitent une adaptation pour pouvoir les utiliser dans le cadre d'une application au SSE, notamment, pour pouvoir les utiliser pour la recherche de motifs à une position fixe dans la séquence, au lieu de balayer toutes les positions (par fenêtre glissante). De plus, la majorité de ces outils utilisent des méthodes gloutonnes ou des modèles probabilistes pour rechercher rapidement des motifs complexes et de grandes tailles dans un nombre élevé de séquences. De ce fait, ils manquent de sensibilités et ils ne sont pas adaptés à la recherche des motifs rares. Le seul outil recensé dans la littérature, qui est capable de faire une recherche exhaustive et exacte de motifs dégénérés est l'outil MoSDi [128], mais il est limité à un modèle de fond qui est trop simple pour décrire des séquences biologiques (*IID*). Dans le cadre de l'application aux SSE, le fait de limiter la recherche de motifs à une position fixe dans les séquences, réduit drastiquement l'espace de recherche et permet d'effectuer une recherche exhaustive exacte de motifs dégénérés, augmentant ainsi la sensibilité de la méthode.

Dans ce contexte, les objectifs de cette thèse sont de :

Objectif 1 : Proposer un algorithme énumératif, exact et rapide pour la recherche de motifs d'ADN en utilisant le modèle IUPAC, inspiré des outils d'analyse ChIP-seq et compatible avec une utilisation dans le cadre de la recherche d'erreurs de séquençage. L'algorithme sera implémenté dans un outil facilement utilisable.

Objectif 2 : Proposer une fonction de score, qui permette d'attribuer un score à chaque variant détecté en fonction de son contexte nucléotidique (motifs en amont).

Objectif 3 : Tester l'outil et la fonction de score, et analyser leur impact sur la qualité d'appel des variants.

Troisième partie

Les contributions de la thèse

CHAPITRE 4

DÉVELOPPEMENT DU LOGICIEL $\mathscr{D}iNAMO$

Ce chapitre présente la première contribution de la thèse, avec la définition d'un nouvel algorithme de recherche de motifs sur-représentés dans un premier ensemble de séquences d'ADN par rapport à un second. Il s'agit d'un algorithme utilisant le modèle IUPAC et qui appartient à la famille des algorithmes énumératifs exacts, pour reprendre la classification du chapitre 2. L'idée est d'énumérer d'une façon intelligente les motifs IUPAC à partir du jeu de données pour réduire l'espace de recherche et rendre la recherche exhaustive des motifs dégénérés possible. Cet algorithme a été implémenté dans le logiciel $DiNAMO$ (pour DNA MOtif) [163].

4.1 L'algorithme

4.1.1 Principe général

$DiNAMO$ prend en entrée deux fichiers de séquences au format multi-fasta, qui correspondent respectivement au jeu de données positif (ou signal), nommé $\mathscr{P}$, et au jeu de données négatif (ou contrôle), nommé $\mathscr{N}$.

Son algorithme est divisé en deux étapes principales. La première consiste à générer tous les motifs IUPAC possibles à partir du jeu de données $\mathscr{P}$ (cf. section 4.1.2). La seconde consiste à évaluer les motifs trouvés pour garder ceux qui ont le meilleur pouvoir

discriminant (cf. section 4.1.3). Il admet deux paramètres : L la longueur des motifs, et d le degré de dégénérescence (nombre de lettres dégénérées dans le motif, $d \leq L$).

4.1.2 Génération des motifs IUPAC et construction du demi-treillis

4.1.2.1 Comptage des motifs

La première étape consiste à recenser les $L-mers$ présents dans les séquences de $\mathscr{P}$. Le nombre d'occurrences de chcun de ces $L-mer$ dans les deux fichiers $\mathscr{P}$ et $\mathscr{N}$ est calculé, et stocké dans une table de hachage. Cette structure de données garantit un accès rapide aux informations dans les étapes suivantes de l'algorithme. À partir de ce point, seule cette table de hachage est utilisée, les séquences elles-mêmes n'étant plus nécessaires.

Les fichiers peuvent être analysés en deux modes : en *fenêtres glissantes*, où toutes les fenêtres de longueur L de chaque séquence sont analysées, en les décalant d'une position à chaque fois, ou le mode *position fixe*, où seuls les motifs de longueur L apparaissant à une position spécifique dans les séquences sont pris en compte. Le choix dépend de l'application.

4.1.2.2 Construction du demi-treillis

À partir de la table de hachage des $L-mers$, nous générons tous les motifs IUPAC de longueur L pour lesquels toutes les instances sont présentes dans la table de hachage. Nous appelons ces L-mers des L-mers *utiles*.

Cette étape est essentielle car elle évite d'explorer l'espace de tous les motifs dégénérés. Elle est effectuée en utilisant une structure de demi-treillis, qui permet d'organiser les différents motifs IUPAC.

Un *demi-treillis* est un ensemble d'éléments partiellement ordonné, pour lequel chaque couple admet une borne inférieure et une borne supérieure. On peut représenter un demi-treillis sous forme de graphe orienté.

Comme le montre la figure 4.1, les 15 caractères de l'alphabet IUPAC sont naturellement partiellement ordonnés par inclusion et forment un demi-treillis qui comporte

a)

IUPAC	Level	Bases
N	4	A ,C ,G ,T
B	3	C ,G ,T
V	3	A ,C ,G
H	3	A ,C ,T
D	3	A ,G ,T
S	2	C ,G
Y	2	C ,T
K	2	G ,T
M	2	A ,C
R	2	A ,G
W	2	A ,T
C	1	C
G	1	G
T	1	T
A	1	A

b)

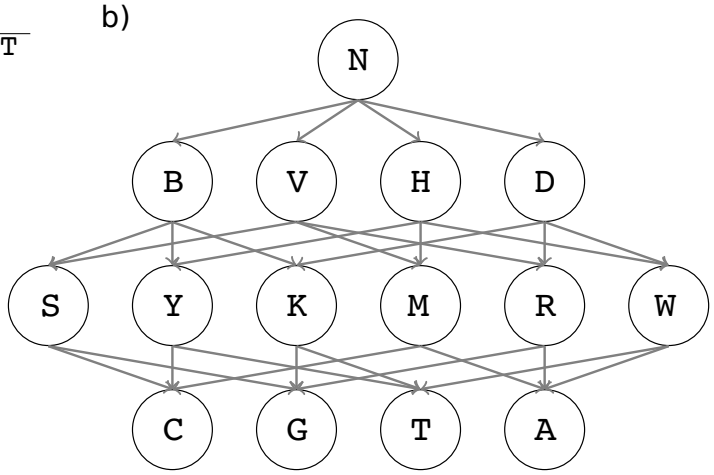


FIGURE 4.1 – Structure de demi-treillis de l'alphabet IUPAC. a) La table des lettres IUPAC avec leur niveau de dégénérescence et les nucléotides qu'elles représentent. b) Le demi-treillis représentant la relation d'ordre (inclusion) entre les différentes lettres IUPAC.

quatre niveaux : le niveau 1 pour les lettres non ambiguës (A, C, G et T), qui forment les feuilles du demi-treillis et ne contiennent aucune autre lettre, le niveau 2 pour les symboles IUPAC combinant deux lettres non ambiguës (K, M, R, S, W et Y), le niveau 3 pour les symboles IUPAC combinant trois lettres non ambiguës (B, D , H, V) et enfin le niveau 4, qui est le niveau supérieur, pour la lettre N (aNy), qui inclut toutes les lettres et est donc une borne supérieure.

À partir de ce demi-treillis de lettres IUPAC, nous définissons le *demi-treillis des motifs IUPAC de longueur L* pour $\mathscr{P}$ de la façon suivante. Les nœuds sont des motifs IUPAC de longueur L et il y a un arc entre deux motifs M_1 et M_2, si M_1 et M_2 diffèrent exactement à une position, disons i, telle que la $i^{\text{ème}}$ lettre de M_1 est directement connectée à l'$i^{\text{ème}}$ lettre de M_2 dans le demi-treillis des lettres IUPAC.

Pour construire cette structure de données, nous commençons par construire les nœuds correspondant à tous les $L-mers$ présents dans $\mathscr{P}$, qui sont donnés par la table de hachage des L-mers construite précédemment (sous-section 4.1.2.1). Cela constitue les feuilles du demi-treillis.

4.1.2.2.1 Dégénérescence des feuilles Nous généralisons ensuite progressivement chaque motif en ajoutant un caractère ambigu à la fois (Figure 4.2). Nous traitons toutes

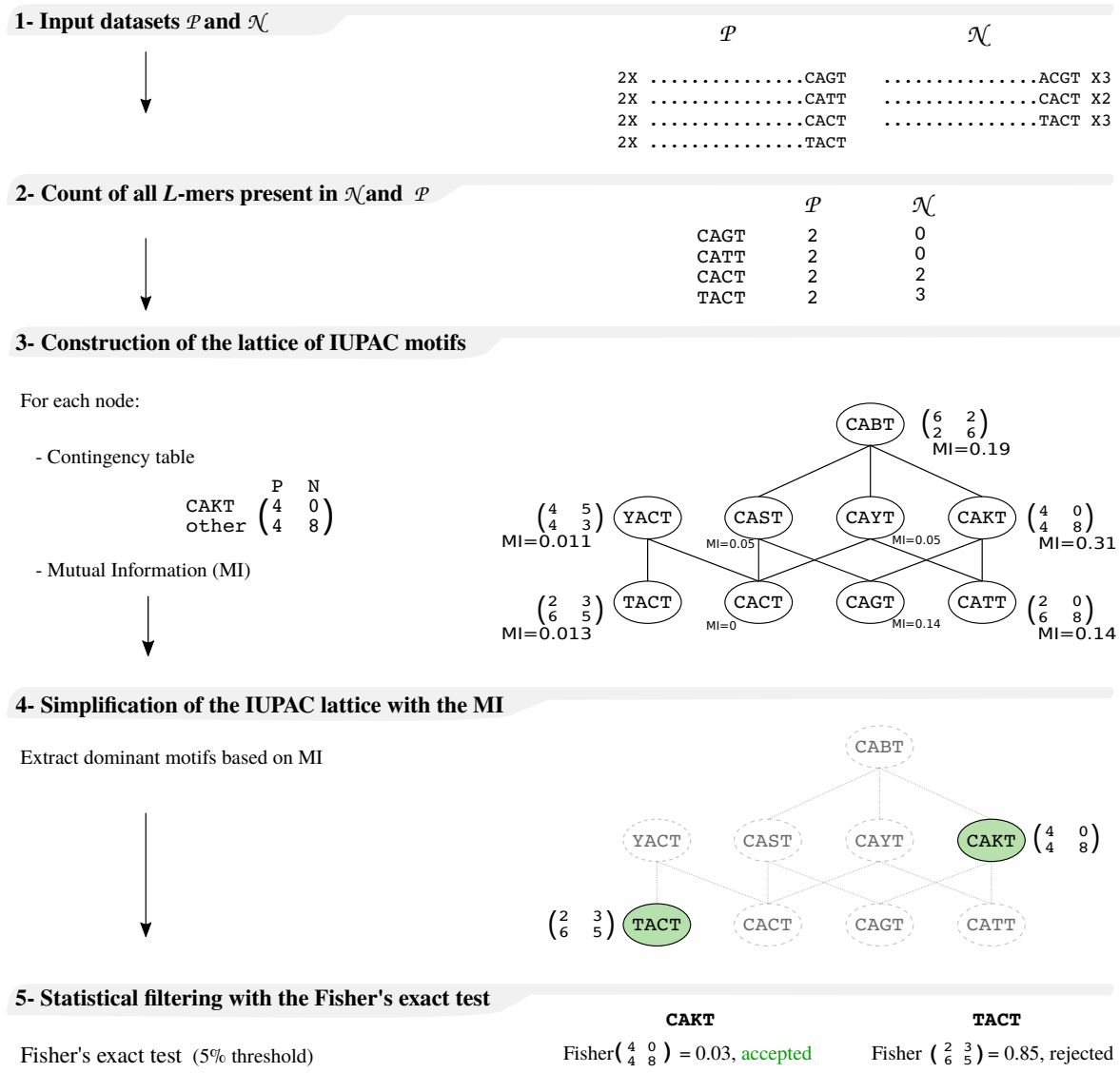


FIGURE 4.2 – Les différentes étapes de l'algorithme de *DiNAMO* avec le jeu de paramètres for paramètres $L = 4, d = 4, p = 0.05$ et mode *position fixe*.

les positions d'un motif donné successivement, en remplaçant le nucléotide actuel par un autre nucléotide. Par exemple, pour le mot $\boldsymbol{C}ACT$, nous examinons d'abord la première position et testons si $\boldsymbol{A}ACT$, $\boldsymbol{G}ACT$ et $\boldsymbol{T}ACT$ sont présents dans la table de hachage. Selon les motifs trouvés, nous générons les motifs IUPAC correspondants. Dans l'exemple présenté à la figure 2, nous générons seulement $\boldsymbol{Y}ACT$ pour cette position puisque seul $\boldsymbol{T}ACT$, en plus de $\boldsymbol{C}ACT$, est présent. En regardant la troisième position, nous générons $CA\boldsymbol{S}T$, $CA\boldsymbol{Y}T$, $CA\boldsymbol{K}T$ et enfin $CA\boldsymbol{B}T$, puisque $CA\boldsymbol{G}T$ et $CA\boldsymbol{T}T$, en plus de $\boldsymbol{C}ACT$, sont présents.

Cette opération est répétée pour chaque position dans le $L-mer$, et cela jusqu'à ce que le seuil de dégénérescence d soit atteint. Cela est réalisé rapidement grâce aux accès rapides aux éléments de la table de hachage (voir l'algorithme 1). Les nouveaux motifs IUPAC sont ajoutés à la table de hachage en incluant leur nombre total d'occurrences, qui est simplement calculé en additionnant le nombre d'occurrences de toutes les instances exactes qui le constituent.

À la fin du processus, le demi-treillis contient exactement l'ensemble des motifs IUPAC utiles de $\mathscr{P}$. À chaque nœud de ce demi-treillis, on associe une table de contingence qui contient les comptes pour le motif correspondant dans les deux fichiers $\mathscr{P}$ et $\mathscr{N}$ (table 4.1).

	$\mathscr{P}$	$\mathscr{N}$
	Jeu de données positif	Jeu de données négatif
présence du motif m	a	b
absence du motif m	c	d

TABLE 4.1 – La table de contingence correspondant à chaque motif (nœud dans le demi-treillis). a (resp. b) représente le nombre de séquences de $\mathscr{P}$ (resp. $\mathscr{N}$) qui contiennent au moins une occurrence du motif m. c (resp. d) représente le nombre d'autres séquences de $\mathscr{P}$ (resp. $\mathscr{N}$).

4.1.3 Simplification du demi-treillis

À cette étape, les motifs IUPAC utiles du jeu de données positif $\mathscr{P}$ ont été recensés et organisés dans une structure de demi-treillis. Le but est maintenant d'identifier les

Algorithm 1 Dégénération d'un ensemble de motifs exacts $\mathscr{P}$

```
1: function SearchForNeighbors(m, i, P)
2:     Input :
       — m : the motif (string in IUPAC alphabet)
       — i : position of m to degenerate (integer)
       — P : set of motifs
3:     Output : Neighbors : set of motifs                          ▷ neighbors of m
4:     Neighbors ← ∅
5:     for each l ∈ {A,C,G,T} do
6:         m[i] ← l
7:         if m ⊂ P then
8:             Neighbors ← Neighbors ∪ m
9:     return Neighbors
10:
11: function Combine(Neighbors)
12:     Input : Neighbors : set of motifs                          ▷ neighbors combination
13:     Output : Neighbors : set of possible IUPAC motifs
14:     Possible_IUPAC_motifs ← ∅
    ▷ generate possible IUPAC combinations from the set Neighbors
    ▷ example : AA,TA,GA → WA,RA,KA,DA
15:     return Possible_IUPAC_motifs
16:
17: function Degenerate(P, d)
18:     Input :
       — P : set of exact motifs to degenerate
       — d : degeneracy level (integer)
19:     Output : All_degenerate_motifs          ▷ set of degenerate motifs, obtained from P
20:     All_degenerate_motifs ← ∅
21:     degeneracy_level ← 0
22:     while degeneracy_level ≤ d do
23:         P' ← ∅
24:         for each motif m ∈ P do
25:             for each position i ∈ [1..|m|] do
26:                 Neighbors ← SearchForNeighbors(m, i, P)
27:                 P' ← P' ∪ Combine(Neighbors)
28:         P ← P'
29:         All_degenerate_motifs ← All_degenerate_motifs ∪ P'
30:         degeneracy_level ← degeneracy_level + 1
31:     RETURN All_degenerate_motifs
```

motifs qui sont significativement sur-représentés dans le jeu de données $\mathscr{P}$ par rapport au jeu de données négatif $\mathscr{N}$.

Comme expliqué en section 2.5.1.1, différentes fonctions de score existent dans la littérature pour réaliser cette tâche. Par exemple, la P-valeur du test exact de Fisher [16], le z-score [9], ou encore l'information mutuelle [17, 18, 19].

Dans notre algorithme, nous utilisons à la fois l'information mutuelle (cf. section 2.5.1.1.3) pour explorer et simplifier le demi-treillis, puis le test exact de Fisher (cf. section 2.5.1.1.2) pour garder uniquement les motifs significatifs.

L'utilisation de l'information mutuelle pour le processus de dégénérescence, permet de dégénérer les motifs en se basant uniquement sur leur pouvoir discriminant envers les deux jeux de données (déséquilibre du nombre d'occurrences entre les jeux de données positif et négatif). La P-valeur utilisée à la dernière étape pour calculer la significativité d'un motif dominant sélectionné préalablement est sensible à l'augmentation des comptes. Or, les motifs dégénérés ont par définition des comptes plus élevés, entraînant un biais de comparaison avec les motifs moins dégénérés. C'est pourquoi, seule l'information mutuelle est utilisée pour sélectionner l'ensemble des motifs dominants. Ensuite, les motifs dominants sélectionnés sont évalués indépendamment les uns des autres avec le test exact de Fisher.

4.1.3.1 Calcul et comparaison des informations mutuelles

Pour chaque nœud du demi-treillis, on calcule son information mutuelle à partir de la table de contingence associé Pour rappel, le but de l'information mutuelle est de mesurer la dépendance entre chaque motif et les deux conditions ($\mathscr{P}$ et $\mathscr{N}$). Elle est définie comme suit :

$$MI(m;c) = \sum_{i\in[0,1],j\in[P,N]} p(m_i,c_j)\log\left(\frac{p(m_i,c_j)}{p(m_i)p(c_j)}\right) \tag{4.1}$$

Puisque les distributions de probabilité pour $P(m)$ et $P(c)$ sont inconnues, nous utilisons une approximation obtenue à partir des probabilités empiriques données par la table de contingence en utilisant l'équation 4.2.

FIGURE 4.3 – Dégénérescence de motifs. Chaque nœud contient un motif avec son information mutuelle. À gauche, un exemple avec une dégénérescence acceptée. À droite, un exemple avec une dégénérescence refusée car le motif WAA (MI=0.15) possède un descendant qui a une plus grande MI (TAA, MI=0.2).

$$p(m_i, c_j) = \frac{\#\,(m_i\ ,\ c_j)}{\sum_{i,j}(m_i\ ,\ c_j)} \tag{4.2}$$

On considère que la fusion de deux motifs IUPAC en un nouveau motif IUPAC est *acceptée* si l'information mutuelle du nouveau motif est supérieure à celles des deux motifs séparément (Figure 4.3).

Pour identifier les dégénérescences de motifs acceptables dans le demi-treillis des motifs IUPAC utiles, on introduit la notion de « dominance » :

Définition 1 (motif dominant)**.** *Un motif est dit dominant si son information mutuelle est plus grande que celle de tous ses successeurs et prédécesseurs dans le demi-treillis.*

Définition 2 (motif dominé)**.** *Un motif est dit dominé si son information mutuelle est plus faible que celle d'au moins un de ses successeurs ou prédécesseurs dans le demi-treillis.*

En utilisant ces deux définitions, la recherche des motifs dégénérés **optimaux** (qui maximisent l'information mutuelle), revient à chercher tous les motifs *dominants* dans le demi-treillis.

Pour accélérer le processus de recherche des motifs dominants, on utilise la propriété de transitivité et de la relation d'ordre du demi-treillis construit à l'étape précédente et qui servira comme structure d'indexation pour cette étape. Premièrement, la liste des

motifs est triée par ordre décroissant en fonction de leur MI. Dans cette liste triée, le premier motif est nécessairement un motif dominant, puisque sa MI est maximale sur l'ensemble du demi-treillis. Par conséquent, nous l'ajoutons à la liste finale des résultats, et supprimons tous ses descendants et ascendants (dans le demi-treillis) de la liste triée (algorithme 2, ligne 7-8), car ceux-ci sont tous dominés. Nous continuons le traitement avec le prochain motif non supprimé ayant la MI maximale, jusqu'à ce que tous les motifs aient été sélectionnés ou supprimés de la liste (voir algorithme 2).

4.1.3.1.1 Calcul de la distribution complète de l'information mutuelle

Dans notre algorithme, nous avons fait le choix d'estimer l'information mutuelle en se basant sur des tables de contingences, c'est-à-dire sur les fréquences observées dans l'échantillon. En effet, les probabilités pour $P(m)$ et $P(c)$ sont inconnues (équation 4.1). Mais l'estimation ponctuelle $\hat{\text{MI}}$ ne donne aucune indication sur la fiabilité de la valeur si la taille de l'échantillon n est finie. Par exemple, pour deux variables aléatoires indépendantes X et Y, $\text{MI}(X;Y) = 0$ alors que $\hat{\text{MI}}(X;Y) = \mathcal{O}(n^{-1/2})$ en raison du bruit dans les données [90].

Pour avoir un indice de fiabilité pour le calcul des $\hat{\text{MI}}$, il est possible de construire des barres d'erreur ou des intervalles de confiance autour de la valeur estimée $\hat{\text{MI}}(X,Y)$. Le but est donc de déterminer l'intervalle T dans lequel la vraie valeur de $\text{MI}(X;Y)$, se trouve avec une probabilité δ, par exemple $\delta = 0.95$ [76].

Il a été montré que $\hat{\text{MI}}(\mathscr{O};\mathscr{C})$ suit approximativement une distribution *Gamma non-centrale* [76, 90] (définie dans [101]) avec les paramètres suivants :

— paramètre de forme, $\alpha = (|\mathscr{O}|-1)(|\mathscr{C}|-1)/2$

— paramètre d'échelle, $\beta = 1/(N \ln 2)$, avec N défini comme le total de la table de contingence

— paramètre de non-centralité, $\lambda = MI(\mathscr{O};\mathscr{C})$

Parmi ces trois paramètres, seul le paramètre de non-centralité, qui correspond à l'information mutuelle recherchée (λ), est inconnu. À partir d'une information mutuelle estimée ($\hat{\text{MI}}$), il est donc possible de calculer la distribution dont le quantile est la valeur de $\hat{\text{MI}}$ observée [76]. De cette manière, il est possible de calculer un intervalle dans

Intervalle de recherche au départ	**Temps de calcul pour chaque motif (en secondes)**	**Temps de calcul estimé pour tous les motifs de taille 6 (15^6 motifs)**
$[0;0,05]$	0,02	2,7 jours
$[0;0,1]$	0,035	4,6 jours
$[0;0,5]$	0,125	16 jours
$[0;1]$	0,23	30,5 jours

TABLE 4.2 – Temps de calcul nécessaire pour calculer les intervalles de confiance pour une information mutuelle estimée, en fonction de différents intervalles de recherche au départ. Les calculs sont faits avec le logiciel *R* [160] en utilisant la fonction *uniroot* ([29]) avec les paramètres par défaut (précision = 10^{-4} et nombre d'itération maximal = 10.000). L'augmentation des ces deux derniers paramètres ralentit encore le calcul. Dans le cas d'un jeu de données non maîtrisé (informations mutuelles et taille de jeu de données inconnus), il est difficile de choisir l'intervalle de départ, et on est obligé de partir de l'intervalle de recherche le plus large $[0;1]$.

lequel MI($\mathcal{O},\mathcal{C}$) se trouve avec une probabilité de 0.95 ($MI \in [MI_1;MI_2]$). Les bornes de cet intervalle de confiance sont donc calculées selon ces deux formules :

borne inférieure :

$$\Gamma^{nc}_{0.975}(\alpha,\beta,\boldsymbol{\lambda}=\boldsymbol{MI_1}) = \hat{\text{MI}}$$

borne supérieure :

$$\Gamma^{nc}_{0.025}(\alpha,\beta,\boldsymbol{\lambda}=\boldsymbol{MI_2}) = \hat{\text{MI}}$$

Le calcul du paramètre λ se fait alors à partir des deux quantiles composant l'intervalle de confiance de la MI. Ce calcul se fait en utilisant la méthode de recherche par dichotomie, ce qui pèse sur le temps de calcul (Table 4.2).

En plus du problème de temps de calcul, la notion de dominance entre les motifs doit être modifiée pour prendre en compte ces intervalles de confiance, et non plus des valeurs simples. Cette méthode de comparaison des intervalles s'est révélée compliquée, notamment pour gérer le cas de deux intervalles qui se chevauchent ou bien le cas d'un intervalle qui est inclus dans un autre. Différents cas de figure sont présentés dans la figure 4.4.

Au vu de toutes ces difficultés supplémentaires de l'utilisation des intervalles d'information mutuelle, cette méthode a été écartée et nous avons préféré utiliser la valeur de la $\hat{\text{MI}}$, calculée à partir des tables de contingence.

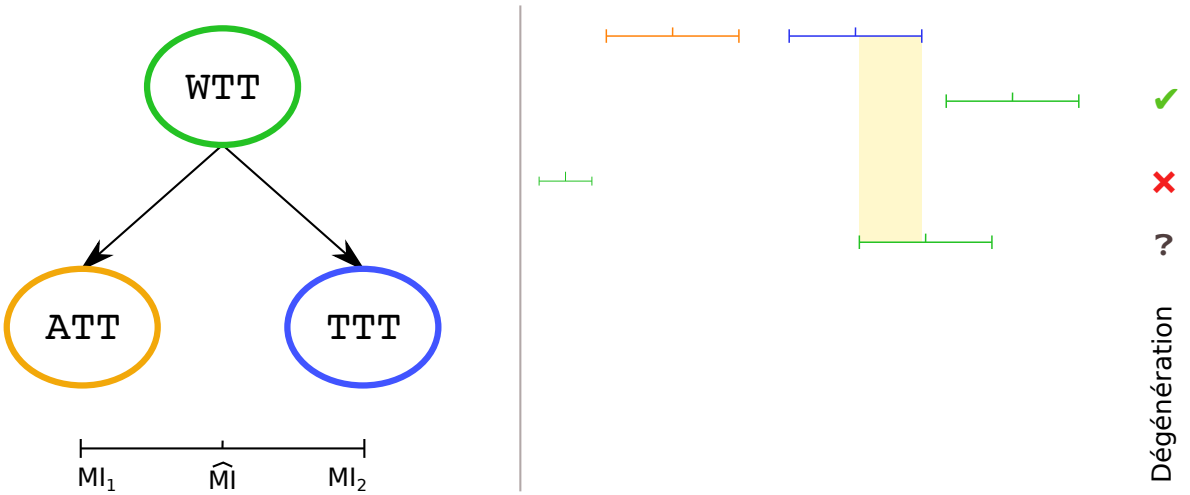


FIGURE 4.4 – Généralisation des motifs en utilisant les intervalles de confiances des M̂I : étude de l'intervalle de confiance du motif généralisé WTT vis-à-vis des motifs exacts ATT et TTT. Les intervalles jaunes et bleus correspondent aux deux motifs ATT et TTT, et l'intervalle vert correspond au motif WTT. Les deux extrémités des flèches correspondent au bornes de l'intervalle de confiance $[MI_1, MI_2]$. Le trait au centre des flèches correspond aux informations mutuelles estimées M̂I. À droite, sont représentés trois cas possibles pour l'intervalle de confiance du motif WTT. Dans le premier cas, MI_1 de WTT est toujours plus grande que MI_2 de ATT et TTT. La généralisation est donc acceptée. Dans le deuxième cas, MI_2 de WTT est toujours plus petite que le MI_1 de ATT et TTT, et la généralisation est refusée. Dans le dernier cas, les deux intervalles correspondant à WTT et TTT se chevauchent, rendant impossible le fait de prendre une décision.

Algorithm 2 Algorithme de simplification du treillis des motifs IUPAC G.

```
1:  function SimplifyLattice(G, pValue=0.05)
2:      Input : G : lattice of degenerate IUPAC motifs
3:      Output : Final_motifs : set of significantly over-represented motifs
4:      Final_motifs ← ∅
5:      while G ≠ ∅ do
6:          max_node ← MaxMiNode(G)                          ▷ FIGURE 4.5
7:          DeleteAncestors(max_node,G)
8:          DeleteDescendants(max_node,G)
9:          DeleteAncestorsOfDescendants(max_node,G)
10:         IF FisherExactTest_Pvalue(max_node) < pValue THEN
11:             Final_motifs ← Final_motifs ∪ max_node
12:         Delete(max_node,G)
13:     RETURN Final_motifs
```

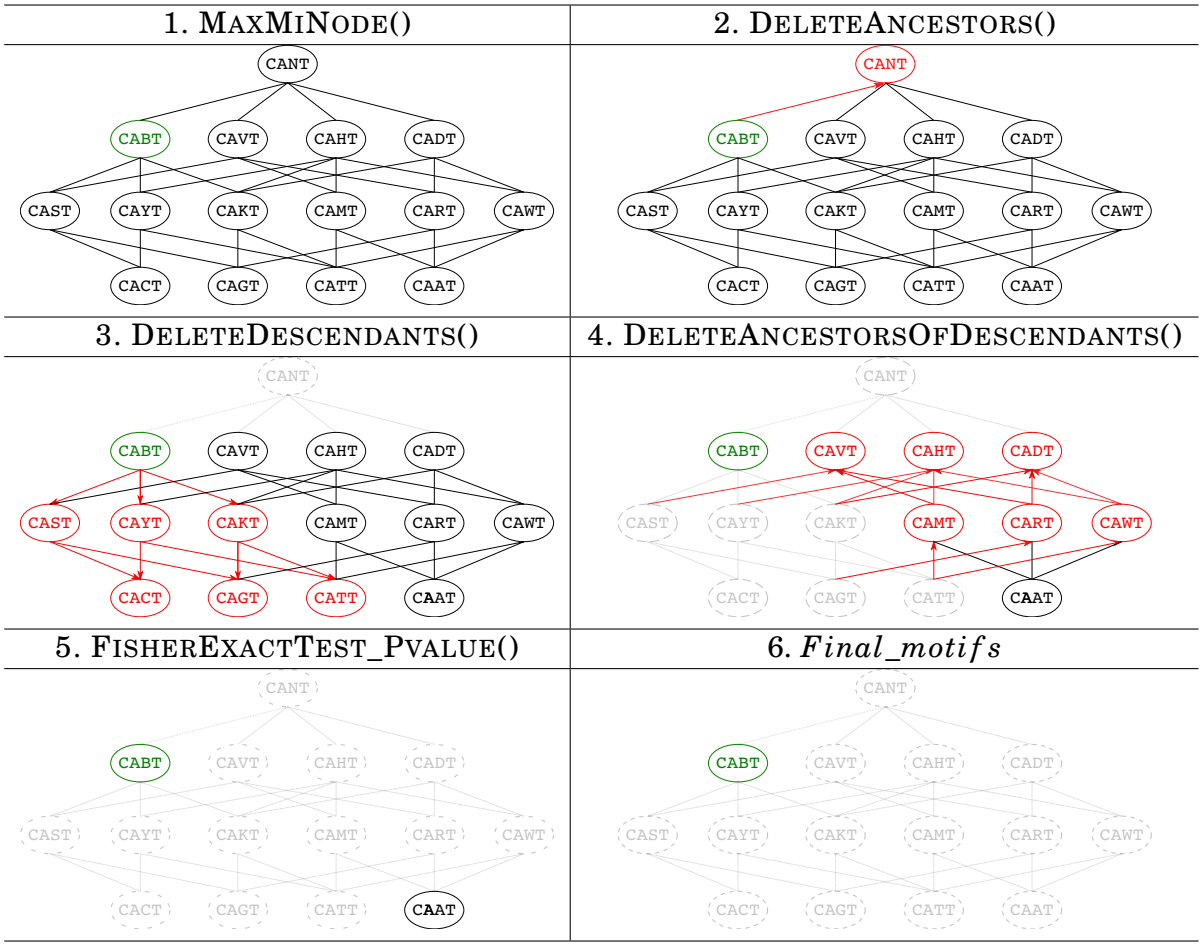


FIGURE 4.5 – Exemple graphique de nœuds impactés pour chaque fonction de l'algorithme 2. En vert, le nœud n sélectionné et ayant la MI maximale. En rouge, les nœuds supprimés par le nœud n.

4.1.3.2 Test exact de Fisher

La dernière étape de l'algorithme consiste à calculer la P-valeur du test exact de Fisher (cf. section 2.5.1.1.2) pour chaque motif retenu dans l'étape précédente. On applique la méthode Holm-Bonferroni [88] pour ajuster les P-valeurs et contrer le problème des comparaisons multiples. Nous conservons uniquement les motifs dont la P-valeur est inférieure à un seuil α donné en paramètre (par défaut : $\alpha = 0,05$).

4.1.4 Détection des motifs secondaires

Les motifs secondaires sont des motifs qui sont significativement sur-représentés dans le jeu de données positif $\mathscr{P}$, tout en ne présentant aucune instance en commun

avec des motifs précédemment détectés avec une meilleure information mutuelle.

Les motifs secondaires sont présents lorsque le jeu de données contient plusieurs motifs sur-représentés, indépendamment l'un de l'autre. De tels motifs se trouvent dans des séquences issues d'une analyse ChIP-seq, où les séquences régulatrices peuvent contenir des sites de fixations pour plusieurs facteurs de transcription différents (Figure 1.8).

Habituellement, de tels motifs sont détectés en masquant toutes les instances des motifs précédemment trouvés dans les séquences, et en exécutant de nouveau l'algorithme dans son intégralité. Ici, nous profitons de l'organisation des motifs en demi-treillis, et nous masquons toutes les instances directement dans le demi-treillis, en éliminant les prédécesseurs des successeurs de chaque motif trouvé. Cela permet une exécution plus rapide.

4.1.5 Regroupement des motifs similaires

Comme mentionné dans la section 4.1.2.1, $DiNAMO$ fonctionne avec deux modes différents pour la lecture des fichiers d'entrée, qui sont : le mode *fenêtre glissante* et le mode *position fixe*.

En mode *fenêtre glissante*, l'algorithme doit prendre en compte les motifs qui se chevauchent. Par exemple, si le motif AMGT est sur-représenté dans un jeu de données, alors MGTN, GTNN, NAMG qui chevauchent tous partiellement AMGT auront tendance à être également sur-représentés et pourraient être détectés par l'algorithme.

Afin d'éviter ces motifs redondants et ne détecter que les motifs principaux et représentatifs, un post-traitement a été ajouté. Il consiste à regrouper les motifs sur-représentés en fonction de leur similarité de séquence. Le motif avec la MI la plus élevée est d'abord sélectionné comme motif de référence, et tous les autres motifs sont alignés contre celui-ci, en autorisant un décalage d'un nombre borné de positions. Dans cet alignement, nous considérons que deux symboles IUPAC peuvent correspondre si l'intersection de leur ensemble d'instances exactes est non vide. Le motif principal (la référence) est gardé et tous les autres motifs qui s'alignent contre lui sont éliminés. Afin d'éviter de regrouper trop de motifs dans le cas d'ensembles de données de faible complexité, nous ne permettons pas l'extension des motifs principaux à plus de la moitié

de leur taille de chaque côté (Figure 4.6). Cette méthode gloutonne est proche de la méthode de regroupement utilisée par l'outil *RSAT - pattern-assembly* [132], à la différence qu'elle fonctionne avec des motifs IUPAC.

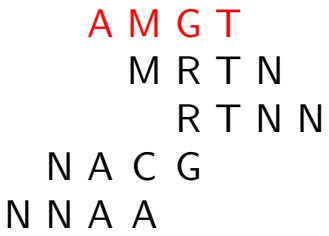


FIGURE 4.6 – Regroupement des motifs chevauchants. Le motif de référence est AMGT (en rouge). La longueur du motif est 4, donc la longueur de chevauchement minimale entre le motif de référence et tous les autres motifs du groupe est 4/2 = 2. Le motif MRTN chevauche avec AMGT car la lettre G est incluse dans R. De même, les motifs NACG et NNAA chevauchent avec AMGT parce que les lettres A et C sont incluses dans M.

4.2 Implementation

DiNAMO est implémenté en C++, et utilise les bibliothèques Sparsepp [154] et boost [106]. Il est disponible librement sous la licence GNU Affero General Public License, version 3. Il peut être facilement installé (les binaires pour Linux, MacOS et Windows sont disponibles) et utilisé sur une machine de bureau standard (< 8 Go de RAM).

CHAPITRE 5

ÉVALUATION DU LOGICIEL *DiNAMO*

Nous avons décrit dans le chapitre précédent le logiciel *DiNAMO*, qui met en œuvre un nouvel algorithme d'identification de motifs IUPAC. Dans ce chapitre, nous présentons des résultats expérimentaux de *DiNAMO* sur différents types de jeux de données afin d'évaluer ses performances. Le premier jeu étudié est un jeu de données synthétique contenant des motifs implantés. Le second jeu est constitué de séquences de pics de ChIP-seq, qui est une application importante de la recherche de motifs.

Pour chacun de ces deux jeux, nous avons comparé *DiNAMO* avec trois autres programmes, MEMECHIP [124], HOMER [84] et Discrover [123], qui utilisent chacun un modèle différent pour la représentation des motifs et une fonction de score pour le calcul de la significativité des motifs. MEME-CHIP appartient à la suite MEME et exécute deux algorithmes de découverte de motifs, MEME [21] et DREME [20]. MEME repose sur une approche de EM pour découvrir les motifs modélisés par des PWM, alors que DREME utilise des modèles IUPAC assortis d'un test exact de Fisher. Discrover est basé sur les HMM et utilise l'algorithme de Baum-Welch [23] pour l'estimation des paramètres. Enfin, HOMER utilise un modèle de *mismatch* simple et la distribution hypergéométrique pour évaluer l'enrichissement des oligo-nucléotides. Pour Discrover, nous devions spécifier en paramètre le nombre de motifs à rechercher, et avons fixé cette valeur à 10. Pour HOMER, nous avons gardé le paramètre par défaut ($-n = 25$). Nous utilisons la même longueur de motifs pour chaque outil, en gardant le reste des

paramètres par défaut.

5.1 Évaluation sur des données synthétiques

Dans cette expérience, nous avons simulé un ensemble de jeux de données à partir de séquences aléatoires. L'intérêt de travailler sur un tel jeu de données est double. D'une part, cela permet de contrôler le nombre, la fréquence et le niveau global de dégénérescence des motifs sur-représentés. D'autre part, cela rend la mesure la sensibilité et la spécificité des outils utilisés, puisque les motifs implantés et leurs positions sont alors connus.

5.1.1 Génération d'ensembles aléatoires de motifs IUPAC

Nous avons généré dans un premier temps plusieurs ensembles de motifs IUPAC d'une longueur fixe ($L = 6$). Chaque ensemble de motifs est caractérisé par le nombre de motifs qu'il contient (de 1 à 4) et par leur niveau de dégénérescence. Le niveau de dégénérescence d'un motif est calculé en additionnant le niveau de dégénérescence de chacune des lettres du motif (Figure 4.1.a). Pour le motif ANANAH par exemple, on obtient une valeur égale à 14, car $A = 1$, $N = 4$, $H = 3$. Nous avons limité la génération des motifs à implanter aux niveaux de dégénérescence suivants : 6,8,10,12 et 14. La valeur la plus basse (6) correspond à des motifs exacts sans caractères dégénérés, tandis que la plus élevée (14) correspond à des motifs tels que ANANAH, MRNWYY ou BAVCHB. Nous avons considéré toutes les combinaisons possibles des deux paramètres, nombre de motifs et niveau de dégénérescence, ce qui donne un total de 20 combinaisons. Pour chaque combinaison, nous avons généré 5 ensembles de motifs, donnant naissance à *100 ensembles de motifs* différents (Table A.1).

5.1.2 Implantation des motifs IUPAC dans des séquences aléatoires et recherche de motifs

Deux fichiers de 5000 séquences d'ADN aléatoires, de taille 100pb chacune, ont été construits avec le générateur de séquences de RSAT [132], en utilisant une distribution nucléotidique indépendante et équiprobable. Ces deux fichiers sont utilisés pour chaque ensemble de motifs implanté. Le premier fichier sert de jeu de négatif (contrôle). Le se-

cond sert de jeu de données positif (signal), dans lequel nous avons implanté les motifs IUPAC générés précédemment (Figure 5.1), à 6 fréquences différentes : 5%, 4%, 3%, 2%, 1% et 0,5% du nombre de séquences initiales. Les motifs sont implantés systématiquement en dernière position des séquences. Les séquences du jeu de données positif sont par la suite coupées au niveau de la position du motif et seules les 6 dernières positions sont gardées afin de forcer les logiciels d'analyse ChIP-seq à chercher les motifs à une position fixe. Chaque motif IUPAC est uniformément représenté par ses instances (tirage équiprobable), et tous les motifs de l'ensemble sont implantés avec la même fréquence. Nous avons répété cette opération 100 fois pour chaque ensemble de motifs, ce qui génère $100 \times 6 \times 100 = 60\,000$ jeux de données différents.

Chacun de ces jeux de données est analysé avec les 4 outils de recherche de motifs (*DiNAMO*, MEME-CHIP, HOMER et Discrover), pour rechercher des motifs de taille 6, sans restrictions sur le niveau de dégénérescence.

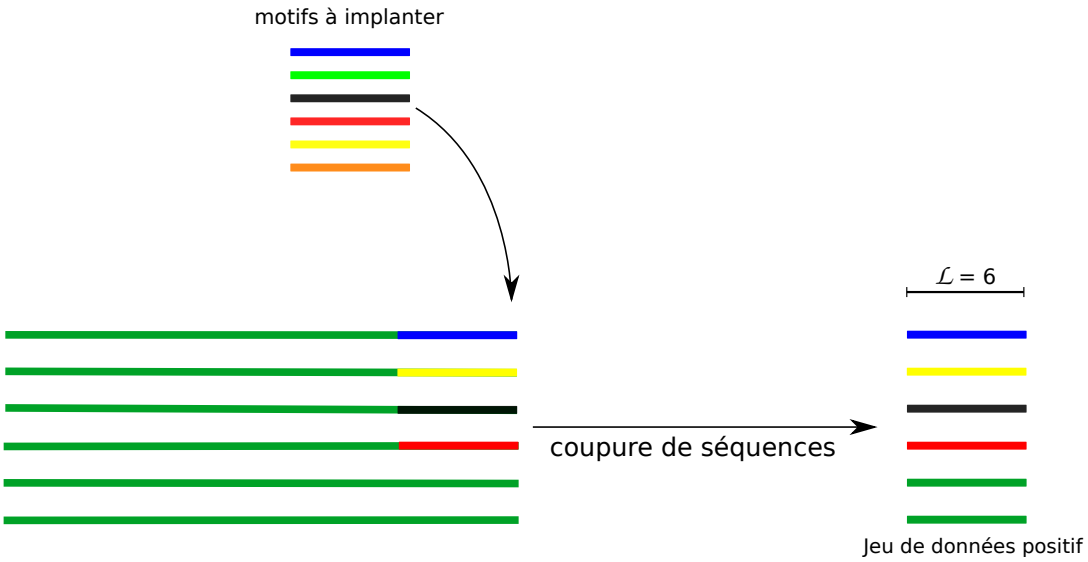


FIGURE 5.1 – Implantation des motifs dans des séquences aléatoires.

5.1.3 Évaluation des résultats

Pour pouvoir évaluer un outil de recherche de motifs et comparer les outils entre eux, il est nécessaire de pouvoir quantifier le résultat. Pour cela, nous utilisons le *coefficient de corrélation au niveau nucléotidique (nCC)* [30, 199].

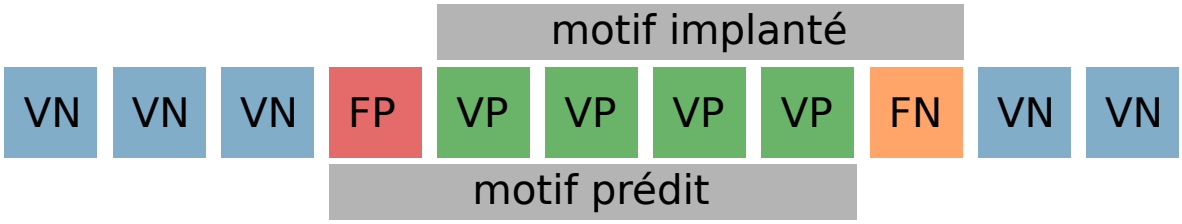


FIGURE 5.2 – Calcul du nCC. Classification des positions nucléotidiques en vrais positifs (VP), faux positifs (FP), faux négatifs (FN) et vrais négatifs (VN), en fonction de leur chevauchement entre les motifs implantés et les motifs prédits.

5.1.3.1 Coefficient de corrélation au niveau nucléotidique (nCC)

Ce coefficient a l'avantage d'évaluer simultanément la spécificité et la sensibilité d'un outil. Il est calculé en utilisant la formule suivante :

$$nCC = \frac{VP \times VN - FN \times FP}{\sqrt{(VP+FN)(VN+FP)(VP+FP)(VN+FN)}}$$

où

- **VP** (vrais positifs) est le nombre de positions, en nucléotides, qui appartiennent au motif et qui sont prédites par l'outil.
- **FP** (faux positifs) est le nombre de positions qui n'appartiennent pas au motif mais sont prédites par l'outil.
- **VN** (vrais négatifs) est le nombre de positions qui n'appartiennent pas au motif et qui ne sont pas prédites par l'outil.
- **FN** (faux négatifs) est le nombre de positions qui appartiennent au motif mais ne sont pas prédites par l'outil (Figure 5.2).

La valeur de nCC varie entre -1 et $+1$. La valeur -1 indique une anti-corrélation parfaite entre les résultats attendus et prédits, $+1$ une corrélation parfaite, et 0 indique qu'il n'y a pas de corrélation.

5.1.3.2 Synthèse

Dans le cas de multiples jeux de données (ce qui est le cas avec nos 60 000 jeux de données à évaluer), il existe trois méthodes différentes pour calculer un nCC qui synthétise les résultats de toutes les expériences [199] :

1. **Moyenne** : cela consiste à calculer la moyenne de tous les nCC obtenus pour chaque expérience.
2. **Moyenne normalisée** : cela consiste à normaliser la valeur de nCC de chaque expérience en soustrayant la moyenne et en divisant par l'écart-type. La moyenne de tous ces nCC normalisés est ensuite calculée.
3. **nCC combiné** : cela consiste à sommer les valeurs des VP/FP/VN/FN obtenues dans chaque expérience et à calculer un nCC à partir de ces valeurs.

TOMPA *et al.* [199] signalent peu de différences qualitatives entre ces trois méthodes de synthèse, à l'exception de la méthode de nCC moyenne qui tend à récompenser les méthodes qui ne prédisent aucun motif sur de nombreux jeux de données. La méthode « nCC combiné » est donc retenue pour ce travail.

5.1.4 Résultats

La figure 5.3 présente les résultats comparés de $DiNAMO$, Discrover, HOMER et MEME-CHIP en terme de nCC combiné, en faisant varier le nombre de motifs implantés, le niveau de dégénérescence et la fréquence d'implantation.

Cette figure montre que $DiNAMO$ réalise globalement la meilleure détection de motifs par rapport à tous les autres outils, avec une meilleure valeur de nCC dans tous les cas.

HOMER atteint les valeurs de nCC les plus proches des valeurs de $DiNAMO$. Cependant, les motifs identifiés sont presque exacts et faiblement dégénérés. En effet, au lieu de trouver un seul motif dégénéré, HOMER rapporte plusieurs instances exactes représentant le même motif. Par exemple, pour le motif WWWHCB inséré avec une fréquence de 5%, $DiNAMO$ trouve un seul motif (WWWHCB), alors que HOMER identifie 24 motifs différents, dont la liste est donnée dans la table 5.1. Ces motifs sont redondants et représentent souvent un sous-ensemble de motifs initialement implantés.

MEME-CHIP obtient de bons résultats avec des motifs exacts (niveau de dégénérescence égal à 6 sur la figure 4.b), mais sa valeur de nCC diminue rapidement avec le niveau de dégénérescence croissant des motifs. MEME-CHIP semble donc adapté à la recherche des motifs peu dégénérés.

AWWHCB	ATAHCB	ATTTCB	TWWCCG
HWAWCY	AWTYCY	MWTMCC	ATAACC
AAWWCB	AWWWCS	AWAMCY	WWWCCB
HAWACY	WWAMCY	ATWCCB	HAWTCC
TWWHCB	AWTCCY	MWAHCT	HAAACC
HAAHCG	TWTWCC	WATACC	YAWACY

TABLE 5.1 – Les 24 motifs trouvés par HOMER dans le jeu de données de séquences avec le motif WWWHCB implanté.

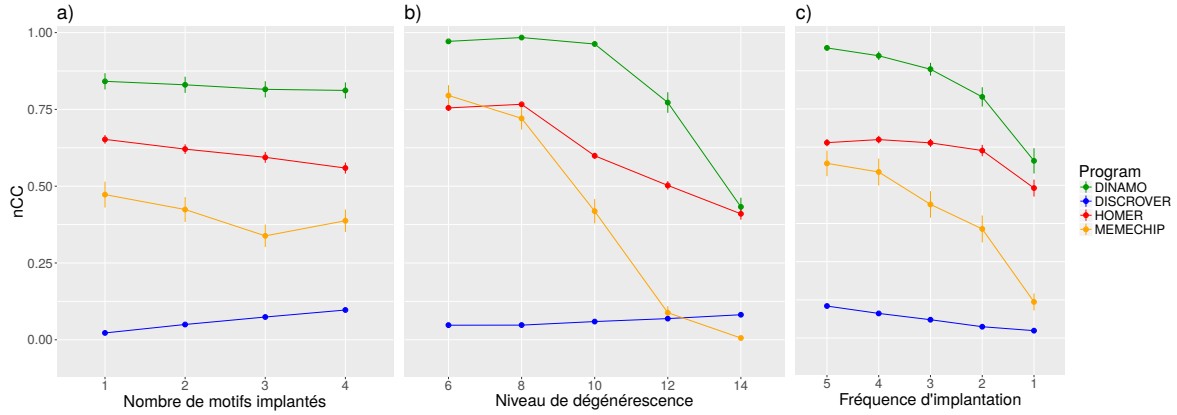


FIGURE 5.3 – L'impact de chaque paramètre de simulation sur la valeur de nCC combiné des 4 logiciels comparés. Dans chaque graphique, un seul paramètre varie tandis que tous les autres sont combinés. a) Nombre de motifs, b) niveau de dégénérescence, c) fréquence d'implantation.

Discrover a la plus faible valeur de nCC. Cela est probablement dû au modèle HMM utilisé qui n'est pas adapté à la recherche des motifs rares et peu représentés (< 5%) dans ces expériences. Des tests supplémentaires réalisés avec 75% de séquences contenant le motif, montrent que Discrover arrive à le détecter. Cela renforce l'hypothèse que la faible valeur de nCC pour Discrover est due à la faible fréquence d'implantation.

On peut aussi remarquer que le nombre de motifs implantés dans le jeu de données n'affecte pas la qualité des résultats (Figure 5.3.a).

Nous avons ensuite étudié l'impact de chacun des deux paramètres, respectivement, le *niveau de dégénérescence* et *le nombre de motifs implantés* à des fréquences variables d'implantation sur le résultat. D'après la figure 5.4, on peut remarquer que pour *DiNAMO*, la valeur de nCC ne varie pas beaucoup avec le niveau de dégénérescence entre 6 et 10 (a). La fréquence d'implantation a un impact plus élevé sur les résul-

tats en utilisant un niveau de dégénérescence élevé (12 et 14). Pour MEME-CHIP, la valeur nCC est élevée (> 0.75) pour les motifs exacts (niveau de dégénérescence = 6), pour les motifs moyennement dégénérés (niveau de dégénérescence $6 < d < 12$), mais elle se dégrade fortement lorsque la fréquence d'implantation diminue. Pour les motifs fortement dégénérés (niveau de dégénérescence ≥ 12), la valeur de nCC est faible pour toutes les fréquences d'implantation (< 0.25) (b). En d'autres termes, il est difficile pour MEME-CHIP de détecter des motifs dégénérés implantés à faible fréquence. Pour HOMER, nous remarquons que la valeur de la nCC diminue proportionnellement au niveau de dégénérescence. Cela signifie que HOMER est également sensible au niveau de dégénérescence, mais moins sensible que MEME-CHIP. Pour Discrover, nous pouvons remarquer que la fréquence d'implantation est le paramètre le plus impactant, car quelle que soit la valeur du paramètre de niveau de dégénérescence, la valeur nCC est faible et semble fortement corrélée à la fréquence (d).

On peut également remarquer que le nombre de motifs implantés n'affecte pas la valeur de nCC des quatre logiciels étudiés. Pour chaque outil, les quatre courbes ont approximativement le même aspect visuel (a).

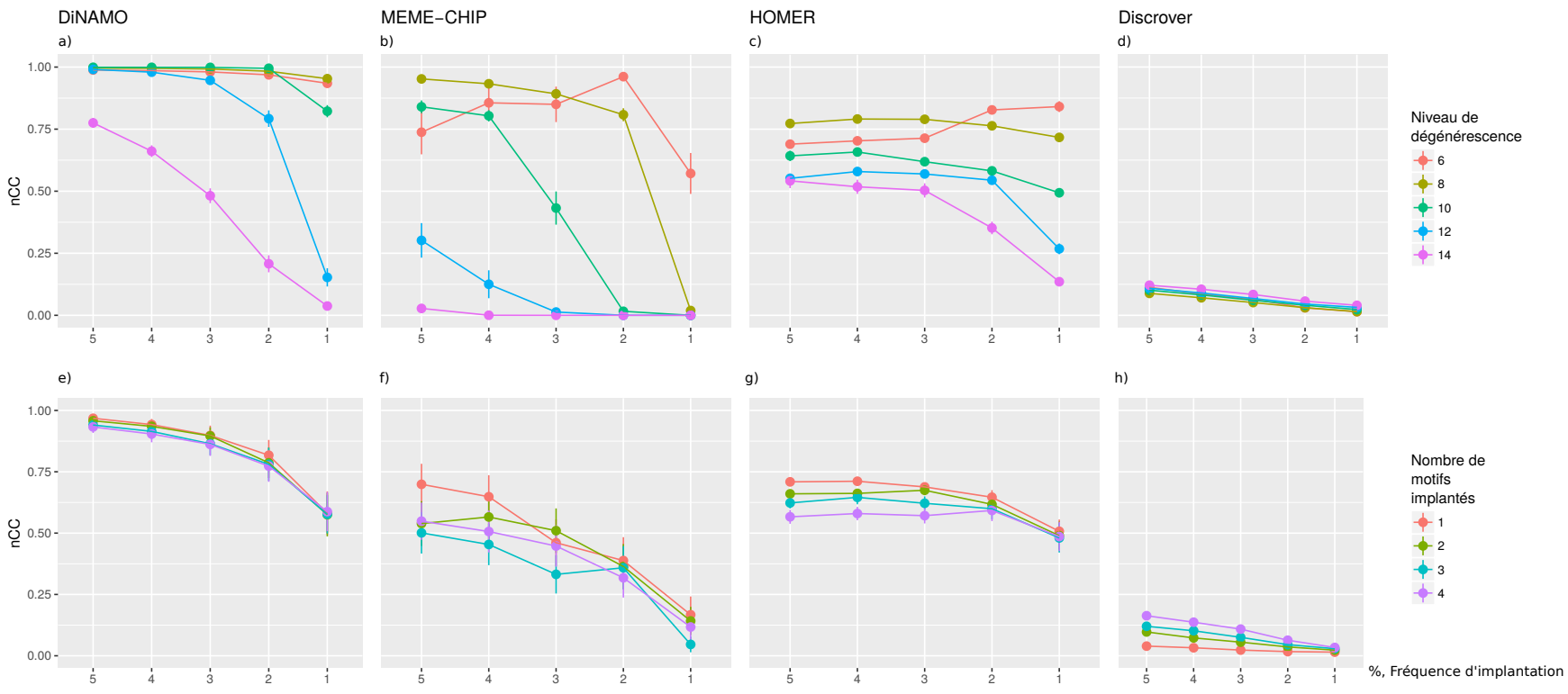


FIGURE 5.4 – Impact de chaque paramètre sur la qualité de détection du motif à chaque fréquence d'implantation des motifs.

Nous avons également évalué la *spécificité* de chaque outil. La spécificité (SPC) mesure la proportion de vrais négatifs identifiés dans l'ensemble de données ($SPC = VN/N$, où N correspond au nombre total de nucléotides dans le jeu de données). Nous avons généré 100 ensembles de données de manière aléatoire, sans aucun motif implanté. Nous avons utilisé les mêmes paramètres que pour les tests effectués précédemment (cf. section 5.1), et avons cherché des motifs de longueur 6 en autorisant jusqu'à 6 positions dégénérées. Les résultats montrent que tous les outils ont une spécificité élevée ($> 0,98$) (Figure 5.5).

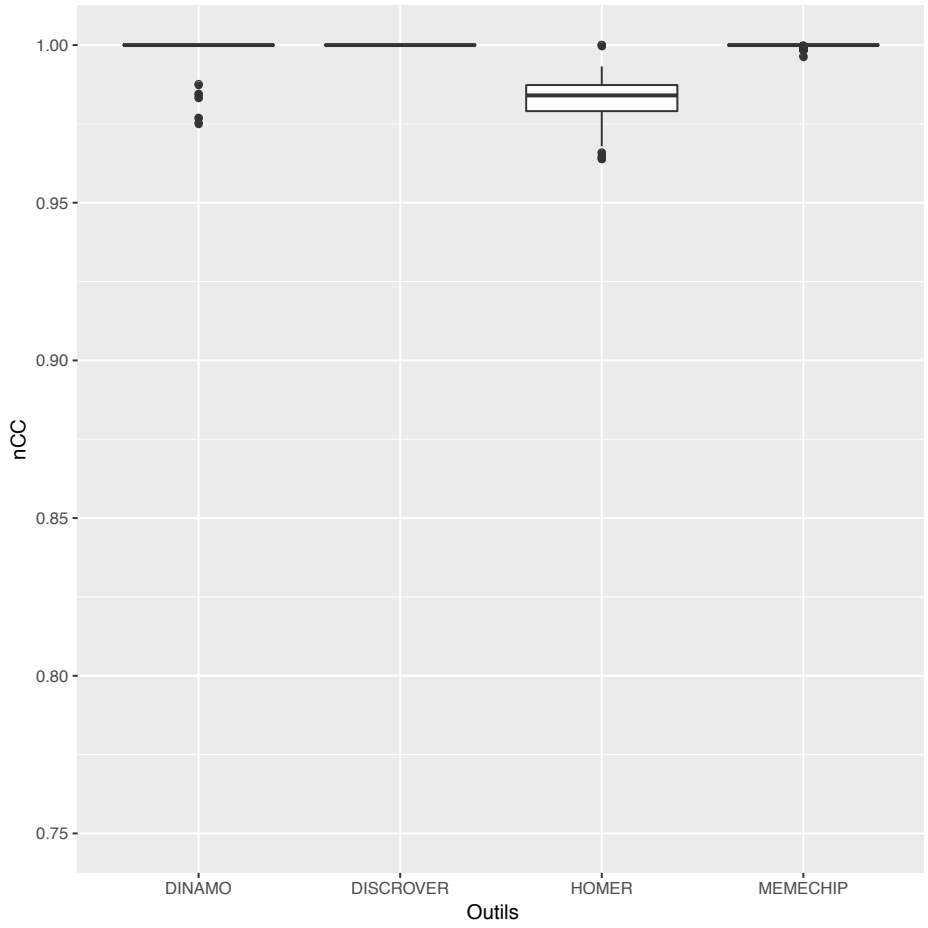


FIGURE 5.5 – Spécificité de chaque outil, calculée sur des jeux de données aléatoires. 100 jeux de données aléatoires, ne contenant aucun motif implanté, sont générés (loi de Bernoulli) afin d'évaluer le taux de faux positifs (5000 séquences aléatoires pour chaque jeu de données).

5.2 Évaluation sur données de ChIP-seq

Les logiciels testés avec *DiNAMO*, MEME-CHIP, HOMER et DISCROVER, sont en fait usuellement utilisés pour l'analyse de régions régulatrices obtenues par CHIP-seq. Le second ensemble de jeux de données que nous testons relève de ce domaine d'application, afin de placer ces outils dans des conditions favorables.

5.2.1 Introduction au ChIP-seq

L'immuno-précipitation de la chromatine suivie d'un séquençage (ChIP-seq) est une technique pour le profilage des interactions entre ADN et protéines, qui permet de ca-

ractériser et de cartographier les interactions entre les facteurs de transcription et leurs sites de fixation à l'ADN, à l'échelle du génome. Le ChIP-seq offre une résolution élevée, un faible niveau de bruit et une grande couverture. Ces qualités, ajoutées au coût décroissant du séquençage à haut débit, a fait du ChIP-seq un outil indispensable pour l'étude de la régulation des gènes [149].

Le ChIP-seq consiste à immuno-précipiter la protéine d'intérêt, fixée sur l'ADN, avec un anticorps spécifique attaché à des billes magnétiques. L'ADN est ensuite séparé de la protéine et séquencé à l'aide des technologies de séquençage haut-débit (Figure 5.6). Les *reads* obtenus sont ensuite alignés contre le génome de référence pour identifier les régions du génome enrichies en *reads*, appelées « pics ». Ces régions sont ensuite analysées par des logiciels de recherche de motifs afin de caractériser le site de fixation du facteur de transcription étudié (TFBS). Cette identification est possible grâce à la conservation du motif et à sa sur-représentativité dans les régions étudiées (cf. section 1.1.3.1).

5.2.2 Matériel et méthodes

Pour évaluer $DiNAMO$, nous utilisons cinq jeux de données correspondant à un facteur de transcription déjà connu et validé dans des études antérieures, et qui avaient été utilisés par les auteurs de DREME pour l'évaluation de leur outil [20]. Il s'agit de trois jeux de données de cellules souches embryonnaires de souris [33], correspondant respectivement aux facteurs de transcription **Oct4**, **Stat3** et **Sox2** et deux jeux de données d'érythrocytes de souris [36, 190], correspondant respectivement aux facteurs de transcription **Gata1** et **Klf1**. Les fichiers de séquences de pics ont été récupérés depuis la base de données publique *Gene Expression Omnibus* (GEO) [61]. Le nombre de séquences dans chaque jeux de données est indiqué dans le tableau 5.2.

Pour chacun de ces cinq jeux de données, le fichier positif a été construit en extrayant des séquences de 100 pb centrées autour de chaque centre de pic. Le jeu de données négatif est obtenu en mélangeant les séquences du jeu de données positif avec l'outil *dinucleotide shuffling* de la suite MEME [21].

Nous avons exécuté $DiNAMO$, MEME-CHIP, HOMER et Discrover pour chercher des motifs de longueur $L = 7$ contenant jusqu'à $d = 3$ lettres dégénérées. Comme tous les

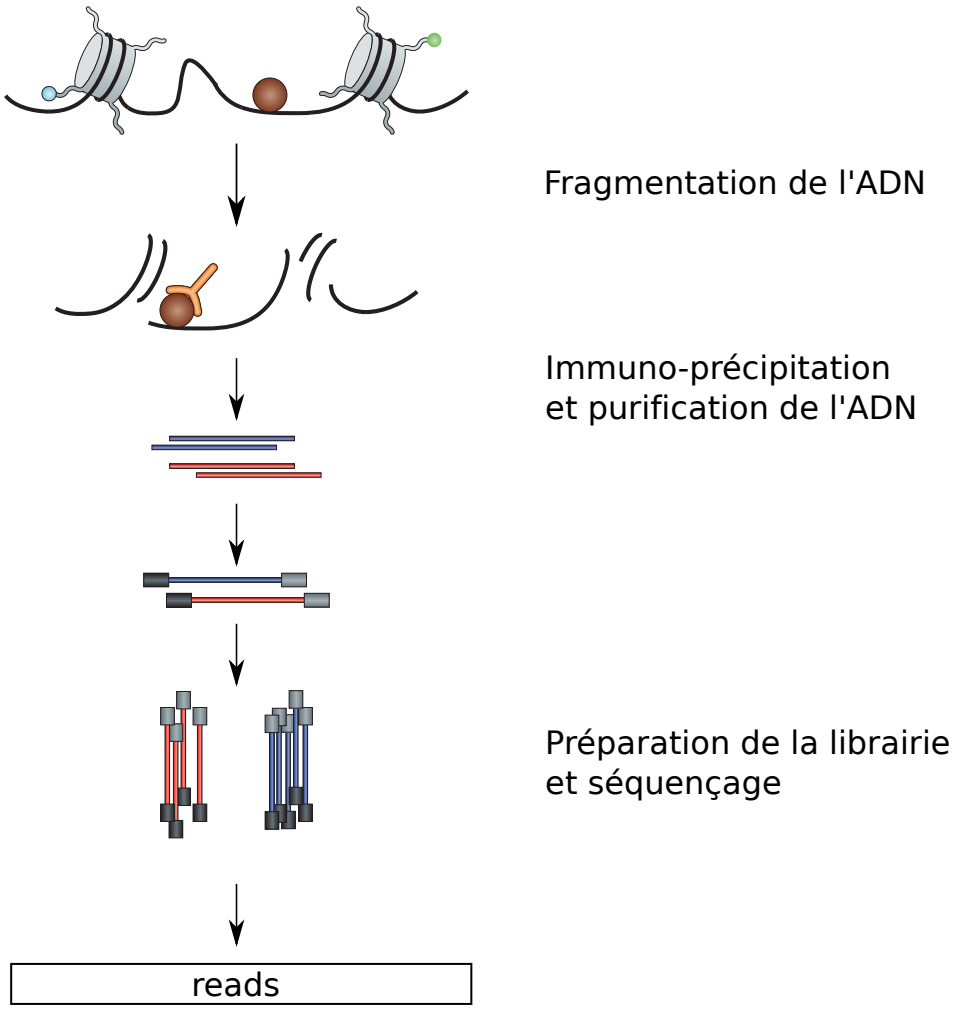


FIGURE 5.6 – Vue d'ensemble d'une expérience ChIP-seq. Figure adaptée de : [149]

outils de découverte de TFBS, nous avons utilisé le mode *fenêtre glissante* de *DiNAMO*, parcourant chaque position des séquences. Ensuite, nous avons appliqué la procédure de regroupement de motifs similaires décrite dans la section 4.1.5.

Les motifs prédits sont ensuite comparés aux matrices de fréquences de la base de données JASPAR [129], en utilisant l'outil TOMTOM de la suite MEME [21]. Plusieurs matrices de JASPAR pouvant correspondre au même motif, nous considérons les dix premières correspondances significatives données par TOMTOM.

Jeux de données	Nombre de séquences	Identifiant GEO
GATA1	14 351	GSE18164
SOX2	4 526	GSE11431
OCT4	3 761	GSE11431
STAT3	2 546	GSE11431
KLF1	945	GSE20478

TABLE 5.2 – Nombre de séquences de pics dans les cinq jeux de données CHIP-seq avec leur identifiant dans la base de données GEO.

	GATA1	SOX2	OCT4	STAT3	KLF1
DiNAMO	18	17	13	17	5
MEME-CHIP	11	16	10	3	2
HOMER	9	10	12	7	6
Discrover	3	4	4	2	2

TABLE 5.3 – Nombre de co-facteurs prédits par chaque outil dans chaque jeu de données.

5.2.3 Résultats

5.2.3.1 Jeux de données complets

Toutes les méthodes ont bien fonctionné sur les jeux de données complets (100% des séquences de pics) et ont correctement identifié les facteurs de transcription attendus. En outre, plusieurs motifs secondaires ont été trouvés, correspondant potentiellement à des co-facteurs de transcription. Le nombre de co-facteurs prédits par chaque outil est noté dans la table 5.3.

Afin de savoir si les co-facteurs prédits sont corrects ou bien s'il s'agit de faux positifs, nous les avons comparés à une base de données contenant des interactions de facteurs de transcription qui ont été validées expérimentalement. Cette analyse est faite en utilisant l'outil *Ingenuity Pathway Analysis* (IPA) de la société Qiagen. Les résultats montrent que la plupart des motifs prédits comme étant des co-facteurs ont une interaction connue avec le facteur de transcription principal (Table A.2).

5.2.3.2 Sous-échantillonnage

Chaque ensemble de données positives a ensuite été sous-échantillonné afin d'évaluer la sensibilité des outils en fonction de la taille de l'échantillon. Nous avons généré

FIGURE 5.7 – Graphe montrant la sensibilité des outils pour la détection du motif étudié. Pour chaque jeu de données, nous effectuons un échantillonnage avec différentes proportions. L'axe des X indique la proportion de séquences de pics du fichier original. Pour chaque proportion, l'échantillonnage a été répétée 100 fois. L'axe des Y représente le nombre de fois où le motif a été détecté parmi les 100 répétitions.

sept échantillons différents correspondant à un nombre de séquences décroissantes : 50%, 20%, 10%, 5%, 2%, 1% et 0,5% des fichiers de séquences de pics originaux. Pour chaque taille d'échantillon, l'expérience a été répétée 100 fois. Les différents outils ont été lancés sur chacun de ces fichiers et nous avons compté le nombre de fois où le motif attendu, correspondant au site de liaison du facteur de transcription, a été correctement trouvé (Figure 5.7).

On constate que $DiNAMO$ et MEME-CHIP prédisent le vrai motif dans la plupart des cas (près de 100% des cas jusqu'à 5% des séquences de pics), mais $DiNAMO$ montre une meilleure sensibilité aux basses fréquences. Il est important de noter que MEME-CHIP utilise deux outils (MEME & DREME) pour atteindre une telle sensibilité, ce qui affecte également le temps d'exécution du programme (Table 5.4).

HOMER obtient également de bons résultats, mais il est moins sensible que MEME-CHIP et $DiNAMO$. La différence de détection de motifs TFBS atteint environ 20%. DISCROVER a la sensibilité la plus faible, probablement due au modèle HMM non adapté aux petits échantillons.

Les résultats obtenus en analysant les séquences de pics montrent que $DiNAMO$ a une très bonne sensibilité, comparable aux outils les plus connus pour les analyses ChIP-seq. Toutefois, $DiNAMO$ détecte plus de motifs secondaires que les autres outils.

Fenêtre glissante

	CPU (sec)	RAM (MB)
DiNAMO	100	3400
MEMECHIP	1280	70
HOMER	18	555
Discrover	170	244

Position fixe

	CPU (sec)	RAM (MB)
DiNAMO	0.75	41
MEMECHIP	1.78	30
HOMER	0.88	120
Discrover	192	141

TABLE 5.4 – Temps d'exécution (*CPU*) et l'empreinte mémoire (*RAM*). Le tableau est obtenu avec le jeu de données GATA1 (14 351 séquences de longueur 100, mode *fenêtre glissante*). Le tableau de droite concerne le jeu de données synthétique (5000 séquences, mode *position fixe*). Les calculs sont réalisés sur un processeur Intel Xeon 1.96GHZ.

5.3 Conclusion

Les résultats sur les deux types de jeux de données montrent que $DiNAMO$ donne globalement de meilleurs résultats que les autres outils de recherche de motifs utilisés dans notre étude. Il a la meilleure sensibilité tout en gardant une très bonne spécificité (≈ 1). Il est aussi capable de gérer une meilleure dégénérescence des motifs pour identifier des motifs fortement dégénérés, alors que les autres outils sont limités à la recherche des motifs exacts ou bien ayant un faible niveau de dégénérescence. Ces propriétés seront utiles pour le cas d'application visé principalement et qui est présenté au chapitre suivant : la détection de contextes nucléotidiques liés aux erreurs de séquençage non aléatoires.

En raison de la nature exhaustive de l'algorithme de $DiNAMO$, la consommation mémoire est plus importante que celle des autres outils testés. Mais cela n'empêche pas en pratique son utilisation sur un ordinateur de bureau.

CHAPITRE 6

APPLICATION AUX ERREURS DE SÉQUENÇAGE NON ALÉATOIRES

Comme nous l'avons vu dans le chapitre 3, les séquenceurs de nouvelle génération sont caractérisés par un taux d'erreurs élevé. Les erreurs aléatoires sont les plus faciles à détecter et sont filtrées sur des critères de profondeur. Par contre, les erreurs non aléatoires (SSE) sont plus problématiques car elles surviennent par définition régulièrement à une même position et peuvent être confondues avec des variants de faible ratio allélique.

Des outils spécialisés dans l'identification des SSE en fonction de leur contexte nucléotidique sont présentés dans la section 3.1. Le principe de ces outils consiste à identifier les motifs d'ADN qui sont sur-représentés en amont des erreurs de séquençage. Toutefois, les outils existants sont limités à la recherche de motifs relativement simples (de faible niveau de dégénérescence ou de petite taille).

Dans ce chapitre, nous présentons, au travers de différents jeux de données de séquençage, l'utilisation de $DiNAMO$ comme outil capable de détecter les motifs favorisant l'apparition des SSE. Nous montrerons sur un jeu de données de haute confiance que l'utilisation des motifs détectés par $DiNAMO$ améliore la qualité des résultats en réduisant le taux de faux positifs détectés par un logiciel d'appel de variants.

6.1 Recherche de motifs liés au SSE avec *DiNAMO*

Nous souhaitons identifier des motifs liés à des erreurs de séquençage non aléatoires (*SSE*). Pour cela, nous devons construire un jeu de données positif ($\mathscr{P}$) correspondant à des séquences contenant des erreurs de séquençage non aléatoires et un jeu de données négatif ($\mathscr{N}$) correspondant à des séquences ne contenant pas d'erreurs non aléatoires. La première étape sera de répertorier les positions des erreurs de séquençage dans un jeu de données, puis de construire ($\mathscr{P}$) et sa contrepartie ($\mathscr{N}$) à partir de ces positions.

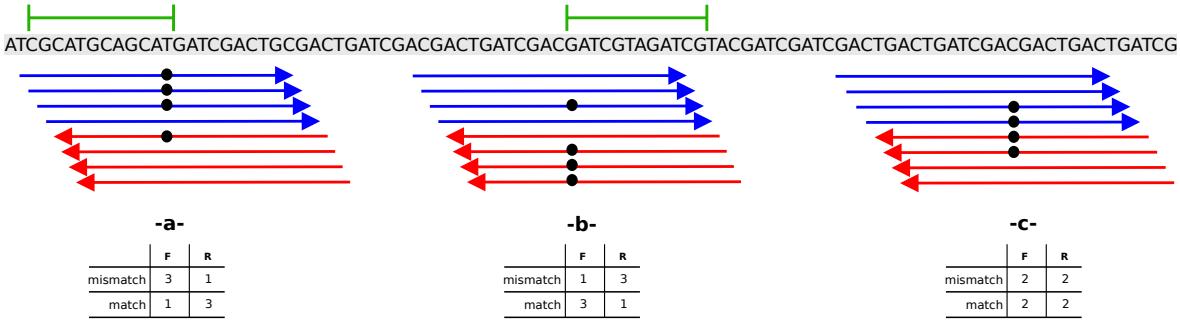
6.1.1 Préparation des données

6.1.1.1 Extraction du contexte nucléotidique des SSE

Nous partons d'un fichier de *reads* alignés, au format BAM, et supposons qu'un vrai variant doit être retrouvé de façon équilibrée dans les *reads* correspondant aux deux directions de séquençage, alors qu'une erreur de séquençage non aléatoire (SSE) sera présente préférentiellement dans une direction de séquençage par rapport à une autre (biais de sens). Le biais de sens est une caractéristique fondamentale des erreurs de séquençage qui dépendent du contexte nucléotidique en amont. À noter que cela ne permet néanmoins pas d'identifier les erreurs de PCR survenues pendant la construction de la librairie ou durant l'amplification clonale (emPCR, clusters), car l'erreur est ensuite amplifiée indépendamment du contexte nucléotidique et sera présente sur les *reads* dans les deux directions de séquençage.

Chaque position de l'alignement des *reads* sur la référence (dans le fichier BAM), est donc testée pour déterminer si le nombre de *reads* comportant un *mismatch* (une base qui diffère de la référence) à cette position, est en déséquilibre entre les deux directions de séquençage (biais de sens). l'outil pysamstats [133] a été utilisé pour déterminer le nombre de *matchs*/*mismatchs* à chaque position sur les deux directions de *reads*. Pour chaque position, nous pouvons ainsi construire à partir de ces comptages une table de contingence de taille 2×2 comme l'a fait ALLHOFF *et al.* [16] (Table 3.2), sans toutefois construire de table de contingence agrégée. Un test exact de Fisher est ensuite appliqué sur cette table pour déterminer s'il existe une sur-représentation du *mismatch* dans une direction de séquençage par rapport à l'autre. Si les *mismatchs* sont sur-représentés dans la direction *sens*, on extrait le contexte qui est en amont dans le sens 5' → 3'. S'ils

sont sur-représentés dans la direction *anti-sens*, on extrait le contexte qui est en amont dans la direction 3' → 5' (Figure 6.1). La séquence extraite est de taille 42 pb (la base à la position d'erreur + 41 bases en amont).



	F	R
mismatch	3	1
match	1	3

	F	R
mismatch	1	3
match	3	1

	F	R
mismatch	2	2
match	2	2

FIGURE 6.1 – Localisation des erreurs de séquençage par le biais de sens. En bleu, les *reads* alignés en *sens*. En rouge, ceux qui sont alignés en *anti-sens*. Les points noirs représentent les *mismatchs*. Les lignes vertes représentent les séquences extraites sur le génome de référence. a) Les *mismatchs* se trouvent plus fréquemment sur les *reads sens*, la séquence en amont est donc extraite. b) Les *mismatchs* se trouvent plus fréquemment sur les *reads anti-sens*, la séquence en aval est donc extraite. c) Les *mismatchs* sont représentés d'une façon équivalente dans les deux directions de séquençage, le variant est considéré comme un vrai variant et aucune séquence n'est extraite.

6.1.1.2 Préparation des jeux de données positif et négatif

La deuxième étape est de construire les 2 jeux de données positif et négatif qui seront analysés par *DiNAMO* pour la recherche de motifs. Cette étape est critique car il faut s'assurer de générer un jeu de données négatif sans biais, pour éviter de trouver des motifs sur-représentés dans le jeu de données positif, dus à l'introduction de biais dans le modèle de fond. Le principal biais à éviter, le « biais de composition », est dû à une composition différente en nucléotides entre deux séquences en raison de leur provenance de régions génomiques différentes. Par exemple, les séquences codantes et non-codantes ne contiennent pas le même pourcentage de GC. Un autre biais issu des séquences codantes peut être identifié en analysant la table des codons (Table 1.5). On s'aperçoit que la fréquence des quatre nucléotides (A,C,G,T) diffère entre les trois positions du codon. D'autre part, d'un organisme à un autre, certains codons sont préférentiellement utilisés, donnant lieu à une sur-représentation de ceux-ci dans les séquences

codantes (biais d'usage des codons). Tous ces biais peuvent conduire à la découverte de motifs sur-représentés, qui ne sont pas liés aux erreurs de séquençage.

Afin d'éviter le biais de composition, le biais d'usage des codons et tenter de garantir un jeu de données non biaisé, les 2 jeux de donnée positif et négatif sont construits à partir du même ensemble de séquences, extrait dans l'étape précédente. Ces séquences extraites dans l'étape précédente sont divisées en deux parties de taille égale (42/2 = 21). Les demi-séquences qui contiennent le SSE constituent le jeu de données positif $\mathscr{P}$, tandis que les autres demi-séquences constituent le jeu de données négatif $\mathscr{N}$ (Figure 6.2). Le fait que les 2 jeux de données soient construits avec les même séquences garantit que les différentes régions génomiques sont représentées avec les mêmes proportions dans ces deux jeux de données. De plus, la différence de 21 bases (multiple de 3) entre la dernière position des demi-séquences, permet d'éviter les biais dus à la variabilité de la composition en nucléotides entre les différentes positions du codon. Les séquences qui contiennent plus d'un SSE sont écartées afin d'avoir un contexte nucléotidique pur et spécifique à un seul SSE.

6.1.2 Jeu de données IonTorrent

Nous avons testé $DiNAMO$ sur plusieurs jeux de données provenant de différentes technologies de séquençage, afin de mettre en évidence les motifs liés aux erreurs de séquençage qui caractérisent chaque séquenceur.

Le premier jeu de données concerne la technologie IonTorrent Proton™. Ces données correspondent au séquençage de 11 exomes humains, issus de 3 projets de séquençage d'exomes tumoraux distincts. Les 11 exomes correspondent tous à des échantillons non tumoraux, utilisés dans l'étude initiale pour soustraire les variants constitutionnels (échantillons prélevés en phase de rémission complète, fibroblastes ou fractions cellulaires triées correspondant aux cellules saines). Ces exomes ne contiennent pas de variants somatiques, permettant ainsi d'éviter de biaiser le jeu de données avec des variants de faible ratio allélique.

Les *reads* obtenus ont été alignés avec le logiciel *Torrent Suite* [173], qui est l'outil par défaut fourni avec les séquenceurs IonTorrent. Les fichiers BAM obtenus sont ensuite analysés avec la stratégie décrite ci-dessus (cf. section 6.1.1.2), afin de répertorier

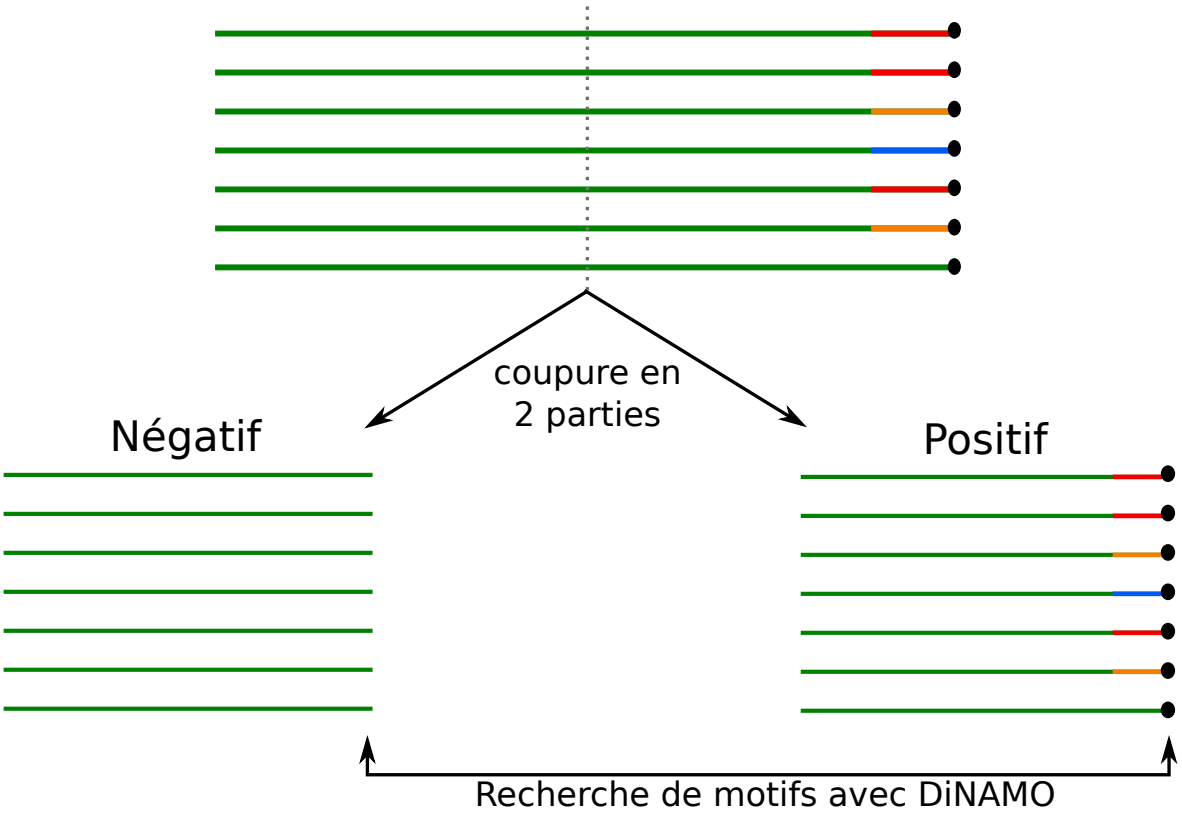


FIGURE 6.2 – Préparation du jeu de données négatif. Les séquences de taille 42 extraites en amont des SSE sont coupées en 2 parties. La partie qui contient l'erreur constitue le jeu de données positif, la deuxième partie constitue le jeu de données négatif (contrôle). Les couleurs à l'extrémité 3' des séquences correspondent aux motifs potentiellement sur-représentés à cette position (à proximité des SSE).

les positions des artefacts de séquençage (erreurs non aléatoires) et extraire le contexte nucléotidique en amont. Environ 24 milliers de séquences ont ainsi été obtenues. Ces séquences ont ensuite été analysées avec *DiNAMO* pour rechercher des motifs sur-représentés.

Pour la recherche des motifs, nous avons exécuté *DiNAMO* avec les paramètres $-l = 6$ et $-p = 1$ pour chercher des motifs de taille 6 à l'avant dernière position dans les séquences, c'est à dire la position qui précède l'erreur de séquençage. Nous avons essayé plusieurs valeurs de niveau de dégénérescence afin d'évaluer l'impact de ce paramètre.

Les 10 premiers motifs (c'est-à-dire ceux avec l'information mutuelle la plus élevée) identifiés par notre méthode sont fournis dans la table 6.1. Les motifs obtenus sont principalement des homopolymères, qui sont déjà connus pour être une source majeure

d'erreurs de séquençage pour la technologie IonTorrent [121, 158, 209].

En effectuant une recherche de motifs exacts (d=0), on remarque qu'il y a plusieurs motifs qui sont similaires, par exemple, les motifs **A**AAAAA et **G**AAAAA qui diffèrent par leur première base seulement. Ces premiers résultats montrent que les motifs identifiés peuvent être généralisés en combinant les motifs similaires afin d'identifier le meilleur sous motif et éviter le sur-apprentissage.

Dans le cas de $d = 6$, on constate que le motif **NN**AAAA a une information mutuelle plus grande que celle de **NNNN**AA (plus dégénéré) et celle de **N**AAAAA (moins dégénéré), ce qui veut dire que **NN**AAAA a un meilleur pouvoir discriminant entre les jeux de données positif et négatif. On peut supposer que **N**AAAAA, non retenu, est plus spécifique mais moins sensible que **NN**AAAA, alors que **NNN**AAA, non retenu également, est plus sensible mais moins spécifique que le motif **NN**AAAA retenu.

d=0	d=2	d=4	d=6
AAAAAA	**NNAAAA**	NNNVCC	NNNVCC
TTTTTT	NNCCCC	**NNAAAA**	**NNAAAA**
TAAAAA	NNTTTT	NNTTTK	NNTTTK
CCCCCA	NCCCCD	NMCCND	NMCCND
TTAAAA	NNGGGG	NNGGGV	NNGGGS
GAAAAA	GGDNCC	NAAAWB	NAAAAB
TCCCCC	AAAANB	NNBAAA	NNBAAA
GCCCCC	CNDCCC	HKCCCW	HKCCCW
ATTTTT	CMCCAD	HHHKGG	HHHKGG
CAAAAA	NDCCCA	NNVTTT	NNVTTT
...	...	...	...

TABLE 6.1 – Les motifs obtenus dans les données Ion-Torrent en fonction du niveau de dégénérescence. Seuls les 10 premiers motifs par ordre d'information mutuelle sont représentés dans la table. Les motifs sont principalement des homopolymères. En rouge, les bases de l'homopolymère AAAAAA qui sont combinés ensemble en utilisant l'alphabet IUPAC (les lettres A,C,G,T à la première position sont remplacées par N).

Une recherche de motifs sur-représentés a été effectuée par la suite sur chacune des positions dans le jeu de données positif pour évaluer la portée des motifs et leurs positions par rapport au SSE. Les résultats laisse penser que les motifs sur-représentés se trouvent seulement à la fin des séquences, ce qui montre qu'il n'y a pas besoin d'augmenter la taille des séquences extraites (> 42) en amont des erreurs de séquençage (Figure 6.3).

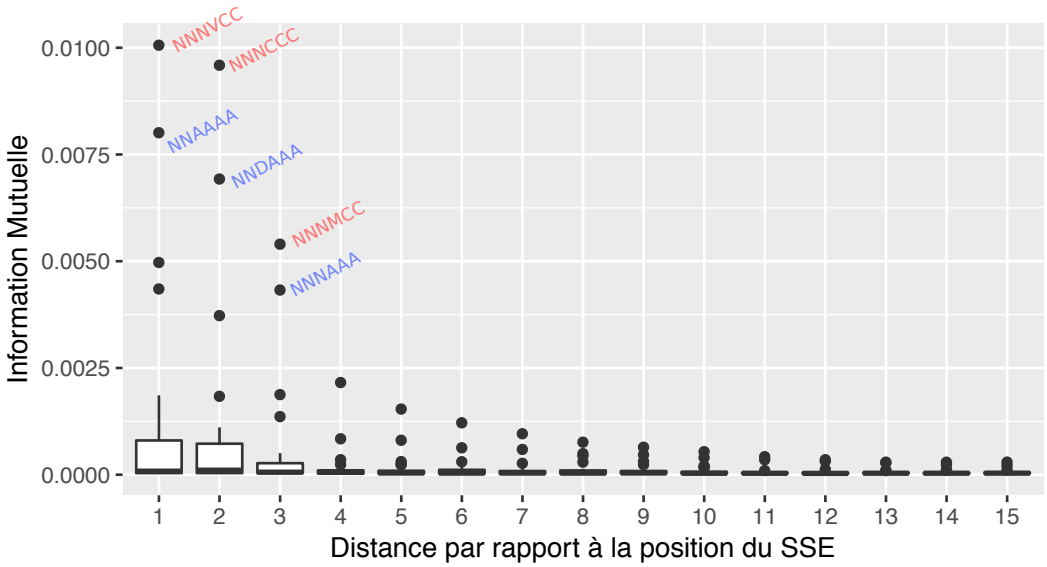


FIGURE 6.3 – Recherche de motifs à différentes positions en amont des SSE. Pour chaque SSE identifié dans le jeu de données IonTorrent de 11 exomes, l'information mutuelle de chaque motif de taille 6 en amont du SSE est calculée et représentée sur l'axe des ordonnées, avec sa distance par rapport à la position du SSE représentée sur l'axe des abscisses. Les séquences des motifs avec l'information mutuelle la plus élevée jusqu'à une distance de 3 par rapport au SSE sont également affichées : en rouge les séquences du motif avec la MI la plus grande, en bleu les séquences du motif avec la deuxième MI la plus grande. On remarque d'autre part, que les motifs sur-représentés se trouvent à proximité des erreurs de séquençage.

6.1.3 Jeux de données Illumina

6.1.3.1 Jeu de données publique : SRR2530739

Afin de diversifier les données et par conséquent les motifs identifiés, un deuxième jeu de données issu de la technologie Illumina a été testé. Il s'agit d'un exome humain séquencé avec le séquenceur Hi-Seq2000 d'Illumina et accessible sous l'identifiant SRR2530739 via la base de données publique SRA [114]. Les *reads* étaient déjà alignés à l'aide de l'outil BWA [116]. Une recherche de motifs de taille 8 a été effectuée afin de pouvoir les comparer aux motifs publiés par ALLHOFF *et al.* [16].

La recherche de motifs par *DiNAMO* révèle des motifs sur-représentés en amont des SSE issues de la technologie Illumina (Table 6.2). Les motifs obtenus sont similaires à ceux obtenus par ALLHOFF *et al.* [16], même si les jeux de données utilisés ne sont pas les mêmes, ce qui conforte l'existence de motifs spécifiques à la technologie Illumina et

d=0	
Alhoff	**DiNAMO**
TGGCGGGT	**TGGCAGGT**
CGGCGGGT	**TGGCGGGT**
GTGGCTCG	GGCGGGGT
CGGCAGGT	TTTTTTTG
TGGCTGGT	**CGGCGGGT**
TGGCAGGT	GCGGGGGT
CGGCTGGT	**AGGCGGGT**
GCGGCGGT	TTTTTTTC
AGGCGGGT	TGGCTGGT
GTGGCGGT	GGCAGGGT
...	...

d=1	
Alhoff	**DiNAMO**
TGGCNGGT	**NGGCGGGT**
NGGCGGGT	**TGGCNGGT**
CGGCNGGT	GGCNGGGT
TNGCGGGT	NGGCAGGT
NCGGCGGT	TTTTTTTV
ANGCGGGT	GGCGGGGH
GTGGCTNG	CGGCDGGT
TGNCGGGT	GGNGGGGT
NTGGCGGT	GNGGGGGT
TGGCGGNT	TGGVAGGT
...	...

d=8	
Alhoff	**DiNAMO**
	NGGSRGGT
	SNGSRGGT
	DGSNGGGT
	DGGVRGGT
	NGGVGGGT
	YGGCDGGT
	VNGGRGGT
	KSSSGGGT
	BSGSGGGT
	HGGSGGGW
	...

TABLE 6.2 – Motifs trouvés avec $DiNAMO$ dans le jeu de donnés Illumina Hi-Seq2000 (SRR2530739), comparés aux motifs trouvés dans l'étude de ALLHOFF *et al.* [16]. Seuls les 10 premiers motifs sont représentés dans les tables. Les motifs en gras sont les motifs communs aux deux logiciels. En rouge, les bases conservées et non dégénérées pour $d = 8$. La dégénérescence de l'intégralité du motif ($-d = 8$), est possible uniquement avec $DiNAMO$. Ce niveau de dégénérescence élevé permet de révéler les parties des motifs les plus conservées et les plus importantes. Il permet notamment de mettre en avant le motif GGT.

reproductibles d'un jeu de données à un autre.

Une recherche de motifs avec un niveau de dégénérescence plus élevé ($d = l = 8$) permet de révéler le sous-motif le plus conservé. Cette analyse montre que les trois dernières bases du motif sont conservées, alors que les autres bases sont dégénérées. Le sous motif conservé est le **GGT**, déjà mis en évidence par toutes les études antérieures comme étant responsable de l'apparition des erreurs de séquençage pour les séquenceurs Illumina [16, 131, 141].

6.1.3.2 Jeu de données de SysCall

Un autre jeu de données a été construit à partir des coordonnées génomiques annotées comme étant des erreurs de séquençage non aléatoires (SSE) et listées dans la publication de SysCall [131]. Les données sont obtenues par un séquençage de monocytes humains en utilisant le séquenceur Illumina GAIIx (qui n'est plus utilisé). À partir des 3272 positions fournies par SysCall, le contexte nucléotidique est extrait et analysé par $DiNAMO$, en cherchant des motifs de taille 6, sans restriction sur le nombre de bases dégénérées possible (paramètres $-l = -d = 6$).

Le premier motif trouvé par DiNAMO est **NNRGGT** (P-valeur ajustée $< 10^{-324}$), ce qui confirme le motif **GGT** initialement rapporté et lié à la technologie illumina. *DiNAMO* trouve également **SBTGGW**[1] et **NBGGGA**, ce qui montre que le motif **GGA** est aussi impliqué dans l'apparition des erreurs de séquençage pour le séquenceur GAIIx. Ce motif a déjà été rapporté dans l'étude antérieure ALLHOFF *et al.* [16].

6.2 Application

Les analyses effectuées avec *DiNAMO* en utilisant des données provenant de plusieurs séquenceurs (cf. section 6.1) ont montré une sur-représentativité de plusieurs motifs en amont des erreurs de séquençage, prouvant ainsi leur implication dans l'apparition des erreurs de séquençage. Cependant, des traitements supplémentaires sont encore nécessaires pour tirer profit de ces motifs et utiliser cette information pour l'amélioration de la qualité des résultats de l'appel des variants.

Cette section explique la méthode, notamment la fonction de score, utilisée pour classer les variants, et les jeux de données construits pour la validation de la méthode.

6.2.1 Fonction de score

Afin de classer les variants en SSE ou en non SSE nous avons établi une fonction de score tenant compte pour chaque variant de son contexte nucléotidique (motif en amont). Ce score servira de paramètre supplémentaire pour filtrer les variants et améliorer la qualité d'appel des variants.

Pour un variant avec un motif m en amont, notre fonction de score dépend de deux mesures, le taux d'erreur attendu en aval du motif, qu'on suppose pouvoir estimer, et le taux d'erreur observé à la position du variant testé. L'idée de cette fonction de score est d'évaluer si le taux d'erreur observé est plus important que le taux d'erreur attendu (bruit de fond).

La comparaison des taux d'erreur attendu et observé est évaluée par le test exact de Fisher, en utilisant la table de contingence suivante :

1. $W = A$ ou T

	Match	Mismatch
attendu	a	b
observé	c	d

TABLE 6.3 – Table de contingence 2×2 pour le variant à évaluer ; a et b représentent le nombre de *matchs* / *mismatchs* attendu qui sont calculés à partir du jeu de données en regardant toutes les occurrences du motif dans les régions séquencées ; c, d représentent le nombre de *matchs* / *mismatchs* observé sur les *reads* à la position du variant à évaluer.

Si le test exact de Fisher montre que les comptes observés des *mismatchs* ne sont pas significativement plus grands que les comptes attendus derrière le motif m, on considère qu'il s'agit d'une erreur de séquençage non aléatoire (SSE) due au motif m (Figure 6.4). Sinon, on considère qu'il s'agit d'un vrai variant.

Dans le cas où il existe des motifs générateurs d'erreurs des deux côtés du variant (en amont et en aval), le calcul de la P-valeur se fait pour les deux directions de *reads* séparément car le motif qui est en amont et celui qui est en aval du variant ne sont pas forcement les mêmes. Dans ce cas, pour qu'un variant soit considéré comme un vrai variant, on exige qu'il passe le test exact de Fisher pour les deux motifs, sinon il est considéré comme SSE. En effet, dans le cas d'un vrai variant, le nombre de *reads* portant le variant doit être supérieur au bruit de fond généré par le motif et ce dans les deux sens (équation 6.1). Pour un variant v à la position p et un seuil α de significativité (par défaut, $\alpha = 0.05$) :

$$F(motif, p) = \begin{cases} 0 & \text{si le motif n'est pas significatif} \\ FisherTest(a,b,c,d) & \text{sinon} \end{cases}$$

$$score(v) = MAX \begin{cases} F(\text{motif } sens, \text{ p}) \\ F(\text{motif } anti\text{-}sens, \text{ p}) \end{cases} \tag{6.1}$$

$$score(v) \begin{cases} < \alpha \rightarrow \text{variant} \\ > \alpha \rightarrow \text{SSE} \end{cases}$$

Calcul des comptes attendus : Les comptes attendus sont estimés à partir de l'intégralité du jeu de données alignés. Il s'agit du taux d'erreur attendu après un motif donné.

Pour calculer ce taux d'erreur par motif, l'ensemble des fichiers BAM issus du *run* de séquençage est parcouru. À chaque position, pour le motif observé, en sens puis en anti-sens, est compté le nombre de *matchs* et le nombre de *mismatchs*. Les vrais variants séquencés pourraient légèrement modifier ces comptes et induire une sur-estimation du nombre de *mismatchs* bien qu'en pratique le nombre de variants pour un motif donné doit être faible pour un jeu de données suffisamment grand. Pour éviter ce potentiel problème, nous éliminons les variants les plus évidents en ne comptant le nombre de *matchs* et de *mismatchs* que pour les positions où le ratio allélique est inférieur à 25% (VAF, *Variant Allele Frequency*).

Bien qu'il puisse encore y avoir des variants à faible ratio allélique (VAF<25%), les motifs étant dégénérés et le jeu de données suffisamment grand, le nombre de tels variants est négligeable par rapport au nombre de positions séquencées et par conséquent leur impact est très limité sur le calcul des comptes attendus.

Ainsi, en reprenant la table de contingence 6.3 pour un motif m donné :

— a = nombre attendu de *reads* avec un *match* après le motif m

— b = nombre attendu de *reads* avec un *mismatch* après le motif m

Ces comptes sont calculés à partir de tous les fichiers BAM [2].

Calcul des comptes observés : Les comptes observés (c et d dans la table de contingence 6.3) correspondent aux nombres de *reads* portant des *matchs*/*mismatchs* à la position du variant qu'on souhaite évaluer.

6.2.2 Matériels et méthodes - GIAB

Pour tester l'impact de l'utilisation de la fonction de score utilisant les motifs sur la qualité d'appel des variants, un jeu de données répondant à plusieurs contraintes a été préparé. Il doit contenir un nombre élevé de variants de faible ratio allélique et qui sont validés, afin de permettre une meilleure évaluation de la méthode. Un tel jeu de données a été préparé *in silico* à partir de données publiques.

2. Le fait de calculer le taux d'erreur attendu sur l'intégralité des fichiers BAM d'un même *run* permet d'avoir plus de diversité en régions génomiques et donc plus d'occurrences des motifs.

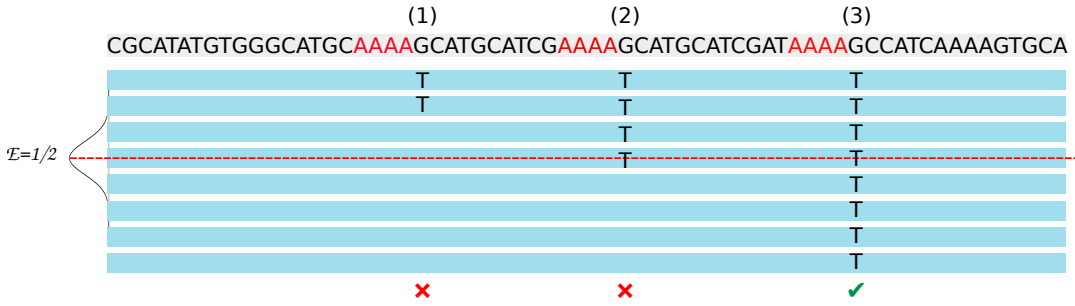


FIGURE 6.4 – Utilisation de la fonction de score utilisée pour évaluer les variants et les classer en vrais variants génomiques ou SSE. Une seule direction de séquençage est représentée dans cet exemple. On considère que le motif AAAA génère des erreurs avec un taux de $E = 1/2$. Le nombre de variants sur les *reads* est donc comparé au taux d'erreurs attendu avec le test exact de Fisher avec un seuil de significativité $\alpha = 0.05$. Le taux d'erreur observé pour les variants 1 et 2 est inférieur ou égal au taux d'erreur attendu. Il sont donc considérés comme erreurs de séquençage. Le variant 3 se trouve avec une proportion plus élevée que celle attendue, donc considéré comme vrai variant.

6.2.2.1 Description des données originales

Le projet *Genome in a Bottle* (GIAB) [215] est un projet de séquençage du génome humain, réalisé par le NIST (*National Institute of Standards and Technology*), dont le but est de mettre à disposition des données de séquençage bien caractérisées et de haute qualité qui peuvent servir de références pour les analyses cliniques et pour l'évaluation des outils bioinformatiques.

Actuellement, sept génomes humains ont été séquencés dans le cadre de ce projet, en utilisant plusieurs méthodes de séquençage différentes (Table 6.4), et analysés : le génome pilote NA12878 et deux trios (parents et leur enfant), un d'ascendance juive ashkénaze et l'autre d'ascendance chinoise.

Pour obtenir une liste de variants de haute confiance (*HC,High-confidence*) pour chaque individu, 14 jeux de données (11 génomes et 3 exomes), provenant de 5 plateformes de séquençage différentes et analysés avec 7 aligneurs différents (Table 6.4), ont été générés. Trois logiciels d'appel de variants différents (GATK UnifiedGenotyper [53], GATK HaplotypeCaller [53] et Cortex [92]) ont ensuite été exécutés sur ces données afin d'identifier tous les variants (SNV et InDels). Les variants concordants entre les différents logiciels et méthodes de séquençage sont considérés comme vrais et sont utilisés par la suite pour l'entraînement du module VQSR de GATK, pour classer les variants

Source	Plate-forme	Aligneur	Couverture	Longueur de reads	Génome/exome
1000 Genomes	Illumina GaIIx	BWA	39	44	Genome
1000 Genomes	Illumina GaIIx	BWA	30	54	Exome
1000 Genomes	454	Ssaha2	16	239	Genome
XPrize	Illumina HiSeq	Novoalign	37	100	Genome
XPrize	SOLiD 4	Lifescope	24	40	Genome
Complete Genomics	Complete Genomics	CGTools 2.0	73	33	Genome
Broad	Illumina HiSeq	BWA	68	93	Genome
Broad	Illumina HiSeq	BWA	66	66	Exome
Illumina	Illumina HiSeq	CASAVA	80	100	Genome
Illumina	Illumina HiSeq – PCR-free	BWA	56	99	Genome
Illumina	Illumina HiSeq – PCR-free	BWA	190	99	Genome
Life Technologies	Ion Torrent	tmap	80	237	Exome
Illumina	Illumina HiSeq – PCR-free	BWA-MEM	60	250	Genome
Life Technologies	Ion Torrent	tmap	12	200	Genome

TABLE 6.4 – Description des jeux de données et de leur traitement pour la génération des fichiers BAM utilisés pour la recherche les variants de haute-confiance pour chacun des 7 individus du projet GIAB [216].

discordants [216]. Une liste de variants dits de Haute Confiance (HC) est ensuite obtenue en appliquant un seuil sur la qualité des variants recalibrés par VQSR. La liste finale de variants HC correspond aux variants concordants et aux variants discordants avec une valeur de qualité recalibrée au dessus du seuil fixé. (Figure 6.5). L'ensemble des jeux de données et des résultats produits est accessible en ligne [11].

6.2.2.2 Traitement et préparation du jeu de données d'évaluation

6.2.2.2.1 Récupération des données : Parmi les 7 génomes étudiés dans le projet GIAB, seuls 4 ont été utilisés, car les données des parents du trio chinois n'étaient pas disponibles en ligne et le génome de l'enfant juive ashkénaze n'apportait pas de nouveaux variants par rapport à ses parents (Table 6.5). Seules les données de séquençage IonTorrent Proton ont été testées. Pour chaque individu deux fichiers sont téléchargés :

1. Le fichier d'alignement (format BAM) : contenant les *reads* déjà alignés sur le génome de référence. Seul l'exome a été séquençé avec la technologie IonTorrent Proton et non le génome complet.

2. Le fichier des variants HC (format VCF) : contenant les variants de haute-confiance obtenus par le protocole décrit précédemment (cf. section 6.2.2.1).

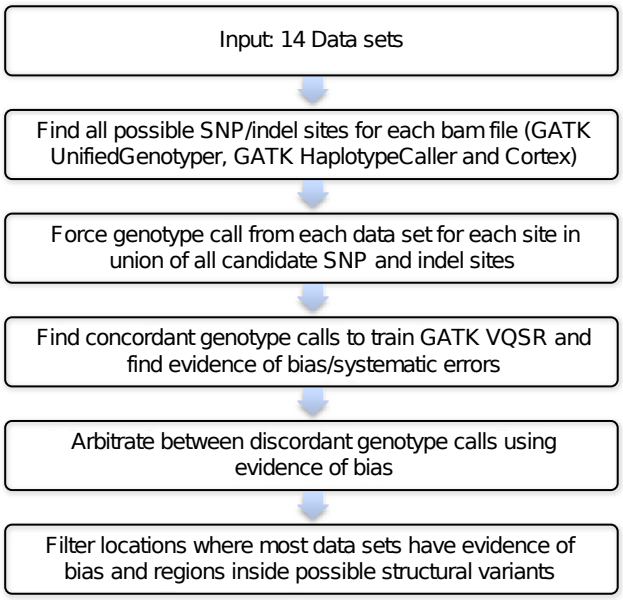


FIGURE 6.5 – Description de la méthode utilisée pour l'appel des variants génomiques de haute-confiance (HC). La méthode utilise les données générées par les différentes technologies de séquençage et analysées par plusieurs logiciels d'appel de variants afin de sélectionner un ensemble de variants HC [216].

Génome	**Id NIST**	**Échantillon NCBI**	**Taille du fichier BAM (en GO)**	**Nombre de Variants HC**
NA12878	HG001	SAMN03492678	25.1	3 775 182
Père AJ	HG003	SAMN03283345	43.8	3 560 920
Mère AJ	HG004	SAMN03283346	50.6	3 597 475
Enfant chinois	HG005	SAMN03283350	48.1	3 627 296

TABLE 6.5 – Les données des quatre individus GIAB utilisés.

6.2.2.2.2 Recherche de motifs : La première étape d'analyse consiste à chercher les motifs favorisant l'apparition des erreurs de séquençage pour la technologie de séquençage testée, l'IonTorrent Proton. Pour cela, une recherche des sites de SSE potentielles a été effectuée selon le protocole décrit dans la section 6.1. Les SSE trouvées chez tous les individus sont regroupés ensemble (373 767 positions) et leur contexte nucléotidique est extrait et analysé par $DiNAMO$.

Plusieurs tailles de motifs avec plusieurs niveaux de dégénérescence ont été testées. Les motifs de taille 3 sont donnés dans la table 6.6

Motif	MI	P-valeur Fisher exact
AAA	0.0067772	0
CCC	0.0029865	0
TTT	0.0029578	0
GGG	0.0008690	1.32881e-134
MWT	0.0003101	1.88485e-49
TKG	0.0001527	2.20516e-25
CCT	6.52481e-05	6.68284e-12
GTT	4.44444e-05	1.14931e-08

TABLE 6.6 – Motifs de taille 3 identifiés dans les données GIAB en utilisant la méthode décrite dans la section 6.1.1.1 et en recherchant les motifs avec $DiNAMO$ en utilisant les paramètre $-l = 3, -d = 3$. On remarque que les premiers motifs dans la liste sont les homopolymers, largement connus comme étant une source d'erreur pour la technologie IonTorrent.

6.2.2.2.3 Création de variants à faible ratio allélique : Pour pouvoir évaluer l'impact de l'utilisation de motifs sur l'appel de variants, un ensemble de variants à faible ratio allélique est nécessaire. Cet ensemble a été créé par dilution des variants HC de chaque individu dans les données des autres individus.

Les *reads* des quatre individus ont été mélangés ensemble pour créer un seul fichier BAM. Les variants qui sont spécifiques à un ou quelques individus se trouvent dilués par les *reads* provenant des autres individus. Le niveau de dilution dépend du nombre d'individus qui portent le variant. Ainsi, le ratio allélique d'un variant qui se trouve chez un seul individu sera divisé par environ quatre (dans le cas d'une couverture de séquençage équivalente chez les 4 individus) (Figure 6.6).

Un seul fichier BAM, d'une taille de 180 Go, regroupant tous les individus est produit, simulant le séquençage d'un seul exome avec la présence de variants somatiques dont le ratio allélique est faible et qui peuvent être facilement confondus avec les erreurs de séquençage non-aléatoires.

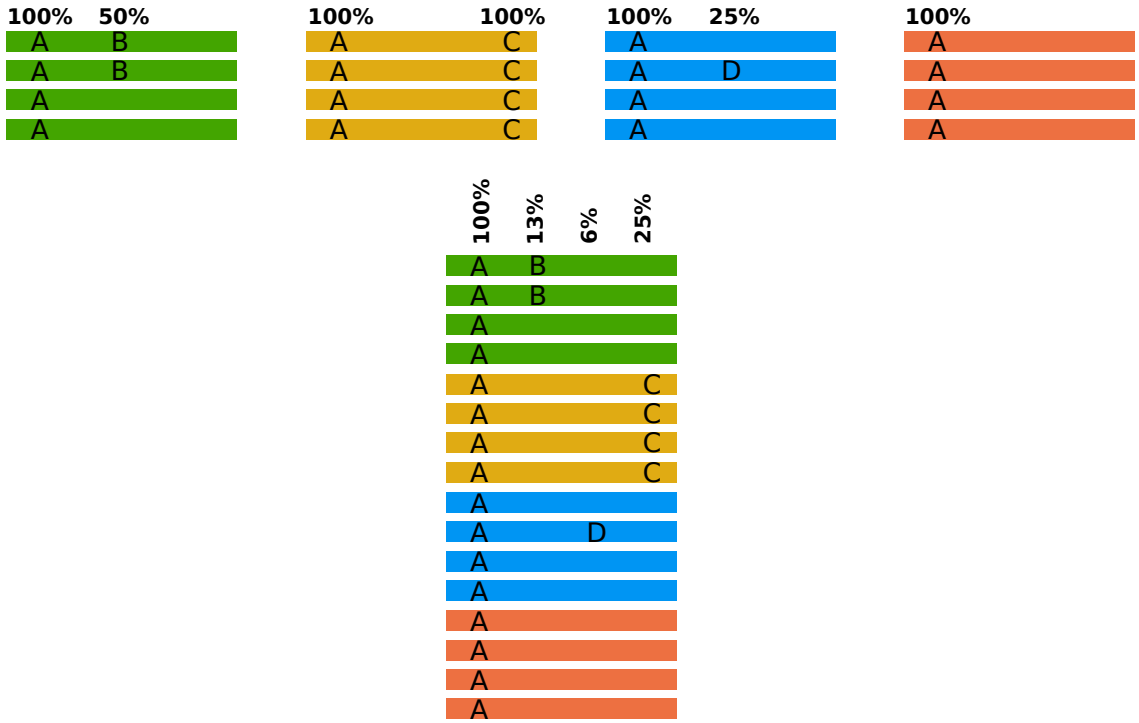


FIGURE 6.6 – Dilution des variants des quatre individus GIAB pour obtenir des variants à faible ratio allélique. Les variants communs à tout les individus (A) gardent le même ratio allélique, tandis que les variants spécifiques à quelques individus (B,C,D) seront dilués par les *reads* qui ne les portent pas et qui proviennent d'autres individus.

6.2.2.2.4 Recherche de variants dans le BAM généré : Une recherche de variants a été réalisée sur le fichier BAM généré. Cette analyse a été effectuée avec l'outil d'appel de variants *Freebayes* [71]. Le choix de *Freebayes* est motivé par le fait qu'il soit adapté pour l'analyse des données d'IonTorrent, une raison pour laquelle ce logiciel a été intégré dans la suite de logiciels (*Torrent Suite*) fourni avec les séquenceurs [173]. De plus, *Freebayes* est un outil très utilisé et bien évalué dans la littérature [41, 91, 216]. Le paramètre du ratio allélique minimal des variants a été fixé à $-F = 0.05$ (Soit une VAF minimale de 5%) pour détecter les variants dilués et augmenter le nombre de faux positifs (qui seront utiles pour l'évaluation de la méthode par la suite).

Seuls les variants de type SNV sont analysés. *Freebayes* identifie 161 856 SNV.

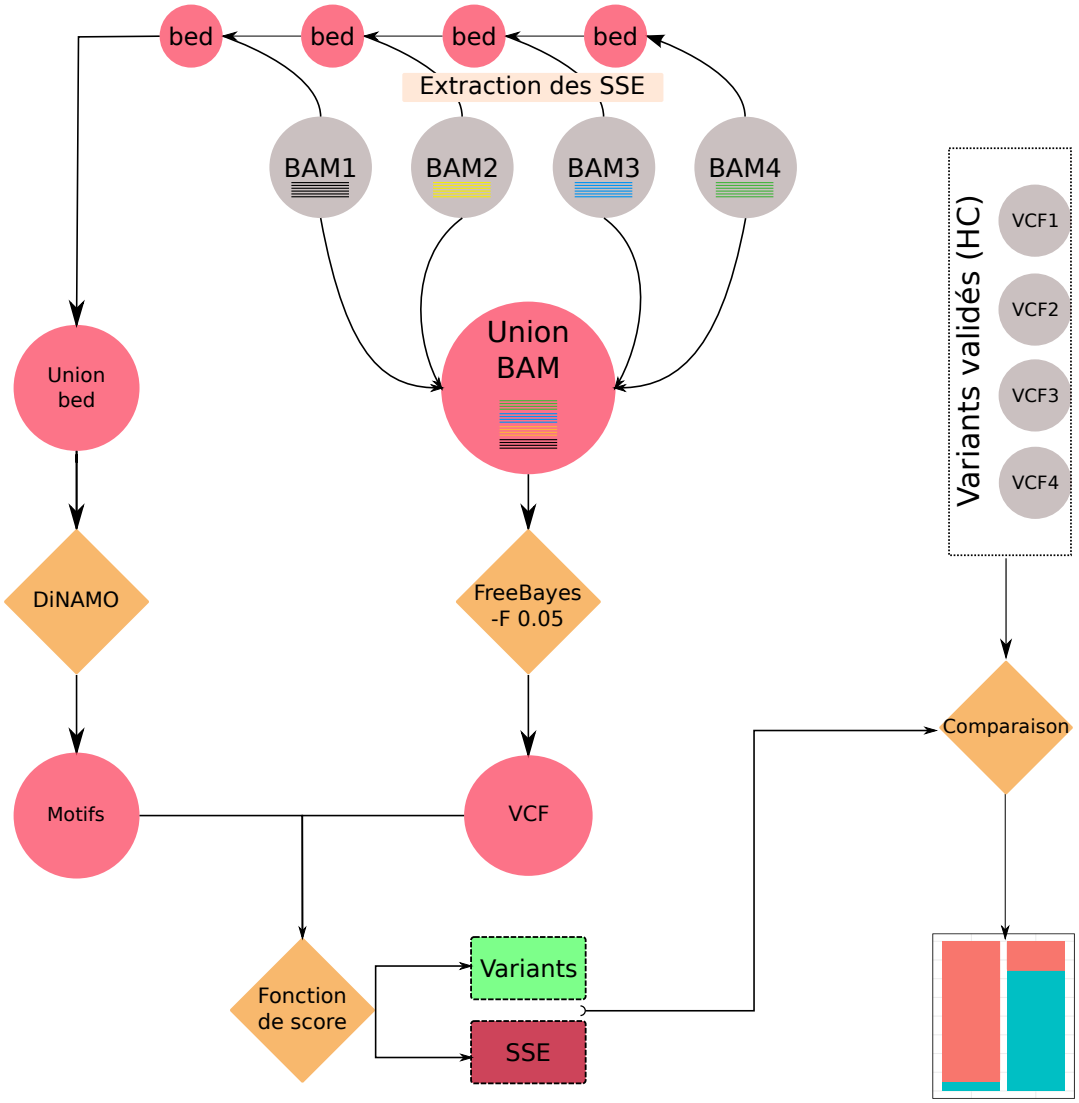


FIGURE 6.7 – Le protocole de préparation et d'analyse des données GIAB. En gris, les fichiers bruts de données GIAB. En rouge, les fichiers générés. Les variants classés par notre fonction de score sont comparés aux variants validés (HC) pour l'évaluation de la méthode.

6.2.3 Résultats

6.2.3.1 Analyse préliminaire

Une analyse préliminaire a été effectuée sur les variants détectés par *Freebayes*. Ces derniers ont été comparés aux variants HC des quatre individus GIAB. On considère qu'un variant est vrai s'il est présent chez au moins un individu. Dans le cas contraire, il est considéré comme erreur de séquençage (probablement non aléatoire, SSE).

Parmi les 161 856 variants trouvés par *Freebayes* sur le fichier BAM généré, 114 528 variants (71%) correspondent à des variants HC, et sont donc considérés comme des vraies mutations, et 47 328 variants (29%) ne correspondent pas à des variants HC et donc considérés comme erreurs de séquençage (Figure 6.8).

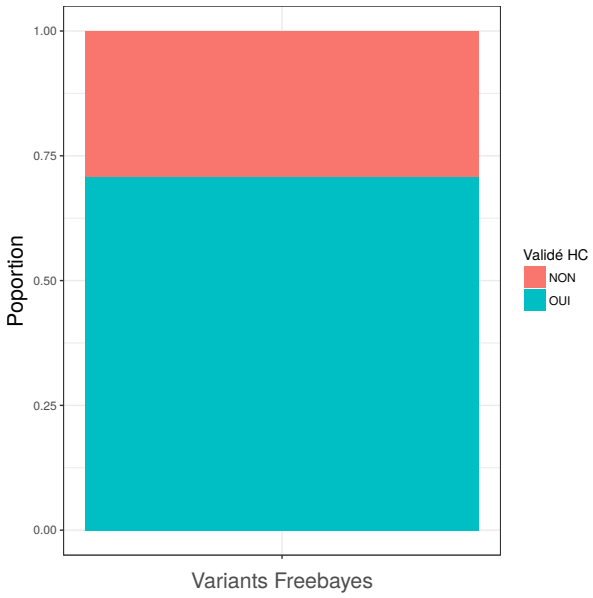


FIGURE 6.8 – Proportion des variants détectés par *Freebayes* et appartenant aux variants HC. 71% des variants sont des variants HC (en bleu), et 29% des variants ne le sont pas et sont donc considérés comme erreurs de séquençage (en rouge).

6.2.3.2 Filtrage par la fonction de score utilisant les motifs en amont et en aval

L'ensemble de ces variants a été évalué en fonction de leur contexte nucléotidique en amont et en aval, en utilisant la fonction de score décrite dans la section 6.2.1 avec un seuil $\alpha = 0.05$, et a été classé en deux catégories : erreurs de séquençage non aléatoire (SSE) et vrais variants.

En utilisant les motifs de taille 4, complètement dégénérés ($-d = 4$), 21 658 variants sont classés comme étant des erreurs de séquençage et 140 198 variants sont classés comme étant des vrais variants.

Dans ces deux catégories, la proportion des variants HC a été mesurée. Les comptes, fournis dans la table 6.7 et représentés dans la figure 6.9, montrent que la prise en compte du contexte nucléotidique des variants a un net impact sur la qualité de la classification des variants. 93% des variants classés comme des erreurs de séquençage, selon les motifs en amont et en aval, ne sont pas dans les variants HC et sont vraisemblablement de réelles erreurs de séquençage. Le filtre sur les SSE permet d'éliminer 20 209 faux positifs. La fraction des vrais positifs (variants détectés par *Freebayes* et validés HC) augmente de 71% à 81%. De l'autre côté, parmi les variants classés comme SSE, seuls 6% sont des variants HC (Faux négatifs), tandis que 94% des variants classés en SSE ne sont pas des variants HC (Vrais négatifs).

Classe / **HC**	**Variant**	**SSE**	**Freebayes sans motifs**
Oui	113 079 **(81%)**	1 449 **(7%)**	114 528 (71%)
Non	27 119 **(19%)**	20 209 **(93%)**	47 328 (29%)
Total	140 198 **(100%)**	21 658 **(100%)**	161 856 (100%)

TABLE 6.7 – Proportion des variants HC dans les deux catégories (SSE et variants) données par la fonction de score en utilisant les motifs de $DiNAMO$. La proportion des variants HC augmente de 71% à 81% par rapport aux variants *Freebayes* sans motifs.

6.2.3.3 Impact de la longueur et du niveau de dégénérescence des motifs sur le classement des variants

L'effet de la longueur ainsi que du niveau de dégénérescence des motifs sur la qualité de classement des variants a été évalué. Les variants ont été analysés avec des motifs

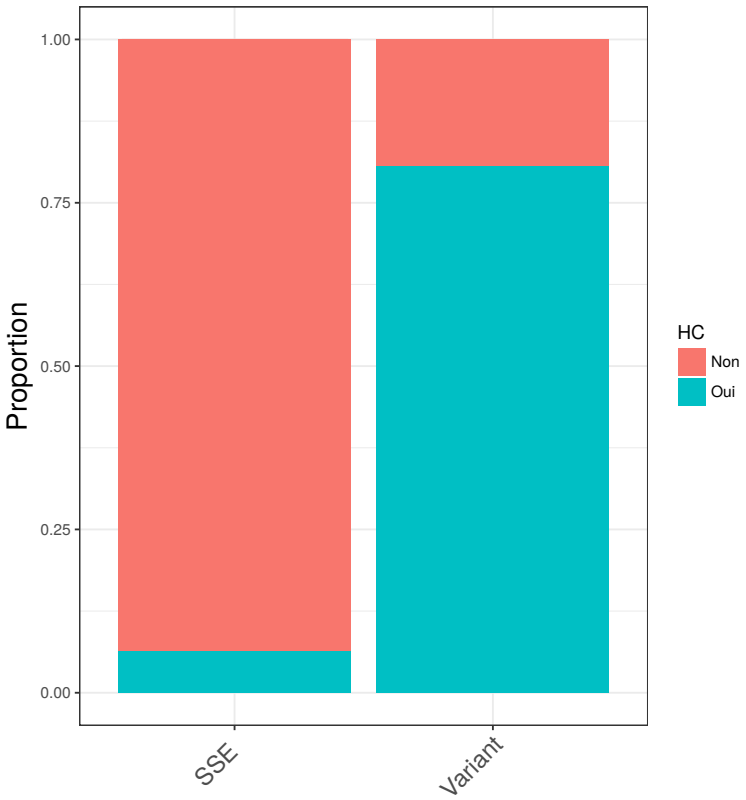


FIGURE 6.9 – Proportion des variants HC dans les deux catégories (SSE et variants) donnée par la fonction de score en utilisant les motifs de $DiNAMO$. La proportion des variants HC est de 81% dans les variants classés en vrais variants, contre 6% dans les variants classés en SSE.

de taille l allant de 2 à 6. Pour chaque taille de motif, nous avons testé les motifs exacts ($d = 0$) et dégénérés (sans limite sur le nombre de bases dégénérées : $d = l$).

Pour chaque valeur de l et d nous évaluons la sensibilité et la spécificité de la méthode avec les formules ci-dessous :

$$\textbf{sensibilité} = \frac{VP}{VP+FN} \qquad \textbf{spécificité} = \frac{VN}{VN+FP}$$

avec :

— VP : les vrais positifs. Les variants **HC** et classés comme **variants** par la fonction de score.

— VN : les vrais négatifs. Les variants **non HC** et classés comme **SSE** par la fonction de score.

— FN : les faux négatifs. Les variants **HC** et classés comme **SSE** par la fonction de score.

— FP : les faux positifs. Les variants **non HC** et classés comme **variants** par la fonction de score.

		Classement	
		Variant	**SSE**
HC	**Oui**	VP	FN
	Non	FP	VN

TABLE 6.8 – Calcul des valeurs de VP, FP, FN et VN.

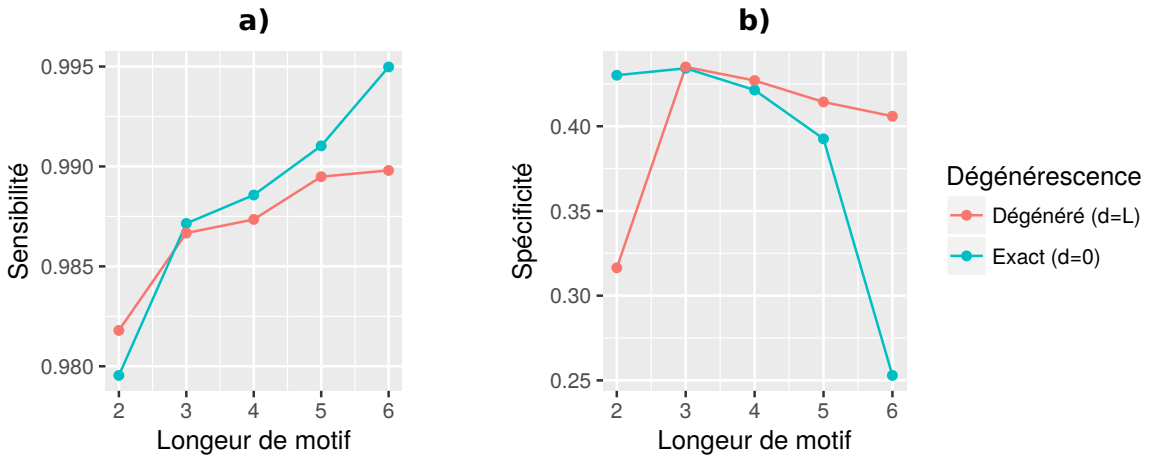


FIGURE 6.10 – Les valeurs de la sensibilité (à gauche) et de la spécificité (à droite) en fonction de la taille et du niveau de dégénérescence des motifs.

Les résultats présentés dans la figure 6.10.a montrent que la sensibilité de la méthode est élevée (>0.98) et qu'elle augmente avec la taille des motifs. La différence entre la sensibilité des motifs dégénérés et exacts n'est pas significative. Par exemple, Le plus grand écart est de 0.005 à la taille 6.

Par contre, pour la spécificité, l'écart est plus important entre les motifs exacts et dégénérés (Figure 6.10.b). On remarque que pour les motifs de petite taille ($l = 2$) les motifs exacts ont une meilleure spécificité, tandis qu'à partir de la taille $l = 3$ la spécificité est meilleure pour les motifs dégénérés. Pour les motifs dégénérés, la spécificité reste ensuite à peu près stable avec l'augmentation de la taille des motifs (> 40%) alors que pour les motifs exacts la spécificité chute en augmentant la taille des motifs pour atteindre une valeur de 25% pour des motifs de taille 6.

Ces résultats montrent deux choses :

1. Augmenter la taille des motifs permet d'améliorer la sensibilité. L'augmentation de la sensibilité pour les données GIAB (Figure 6.10.a) est de l'ordre de 1% entre les motifs de taille 2 et 6, soit 1 006 variants.
2. La dégénération des motifs permet de maintenir la spécificité (Figure 6.10.b), tout en augmentant la taille des motifs.

6.2.3.3.1 Discussion de la faible spécificité La figure 6.10 montre une spécificité globalement faible (<45%). Cette faible valeur est due au nombre élevé des FP (27 119 pour les motifs de taille 4, voir table 6.7). Les FP étant des variants trouvés par *Freebayes* et non présents parmi les variants HC, ces variants peuvent en réalité correspondre à une de ces deux hypothèses :

— H_1 : Une partie des variants FP sont en réalité des VP, en cause un jeu de données HC non complet.

— H_2 : Les FP sont bien des FP, mettant en exergue les limites de *FreeBayes* et/ou de la technologie Ion-Torrent.

Pour évaluer le taux de faux positifs de *Freebayes*, les quatre jeux de données de départ ont été analysés avec *Freebayes* en gardant le paramètre de ratio allélique minimal à sa valeur par défaut ($-F = 0.25$) et en conservant uniquement les variants ayant une qualité correcte ($Q > 40$). La proportion des variants détectés et validés est représentée dans la figure 6.11. Les résultats montrent que même parmi les variants de bonne qualité, plus de 25% des variants identifiés par *Freebayes* ne sont pas contenus dans le set HC (contre 19% avec l'utilisation de $DiNAMO$). Ceci montre que le nombre élevé de faux positifs observés après l'application de notre fonction de score, n'est pas dû à notre méthode, et qu'il était déjà élevé en analysant les jeux de données initiaux, ce qui renforce l'idée que ces faux positifs sont dus à une des 2 hypothèses précédemment évoquées.

6.2.4 Comparaison aux outils DREME et Discrover

Afin de comparer $DiNAMO$ aux approches gloutonnes, un test a été réalisé avec les deux outils DREME (de MEME-CHIP) et Discrover. Pour simuler une analyse dans

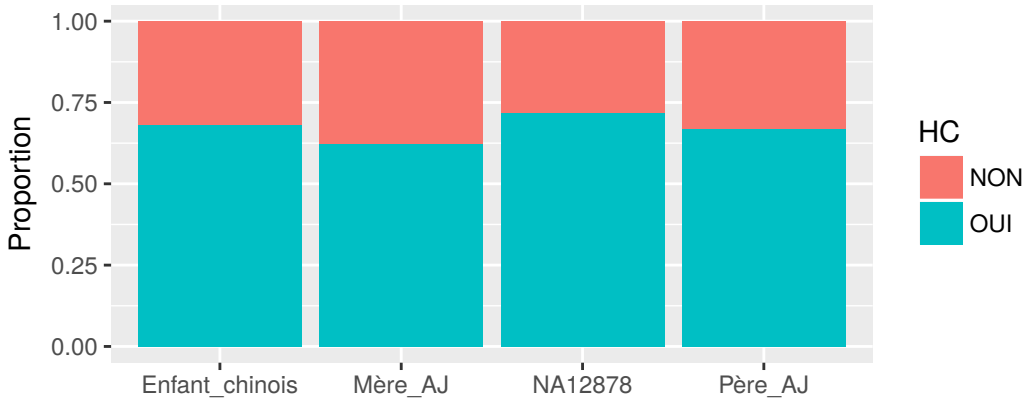


FIGURE 6.11 – Proportion de variants détectés par Freebayes et validés dans le jeu de données HC.

le cadre d'un séquençage ciblé en diagnostic, nous avons préparé trois jeux de données supplémentaires, en plus de l'exome complet, correspondants à trois panels de gènes différents (*design*). Les trois panels font respectivement 1.1 Mb, 643 Kb et 67 Kb et sont actuellement utilisés en routine à l'hôpital pour le diagnostic moléculaire. La recherche de motifs est donc faite sur ces trois jeux de données pour évaluer la sensibilité de la détection des motifs avec des petits jeux de données correspondant aux cas d'utilisation potentielle (Table 6.9). Comme pour les expériences sur les données synthétiques, $DiNAMO$ montre une meilleure sensibilité de détection des motifs. En effet $DiNAMO$ est le seul outil qui détecte des motifs dans le plus petit jeu de données.

Nous avons également évalué la pertinence des motifs identifiés par chaque outil en les utilisant pour filtrer les variants dus aux SSE. De même, nous avons à l'aide des variants HC, calculé pour chaque outil, la sensibilité et la spécificité de la classification des variants entre SSE et vrais variants. La figure 6.12 montre que $DiNAMO$ a la meilleure spécificité dans l'annotation de variants, faisant de $DiNAMO$ l'outil le plus adapté pour la recherche de motifs dans le cadre d'un séquençage ciblé avec un nombre limité de gènes séquencés. La spécificité devient problématique pour les 3 outils pour une taille de données limitée, comme c'est le cas pour le *design 3* de 67 Kb. Quant à la sensibilité, celle-ci reste stable (~0.99) pour les 3 outils.

	DiNAMO	**DREME**	**Discrover**
Exome	AAAT	AAAT	DGYA
	AGGG	ACCC	GSHG
	BGGG	AGGG	GSWG
	CCMW	ARTT	NGCD
	GRCC	BCCC	SWGC
	NAAA	BGGG	VDGC
	NCCC	CCMW	
	TTTK	CCTT	
	VBTT	GAAA	
	WTGG	GGGA	
		GGKT	
		GRCC	
		HAAA	
		TCWT	
		TTTA	
		TTTK	
		VTTT	
		WTGG	
Design 1 (1.1Mb)	HCCC	MCCC	AAAA
	KGGR	TTTK	
	WAAA	WAAA	
	WWTK		
Design 2 (643Kb)	AAAW	AAAW	
	BCSC	SCCC	
	BTTD	TTTT	
Design 3 (67Kb)	YKTT		

TABLE 6.9 – Les motifs trouvés par chaque outil ($DiNAMO$, DREME, Discrover) dans les 4 jeux de données de différentes tailles.

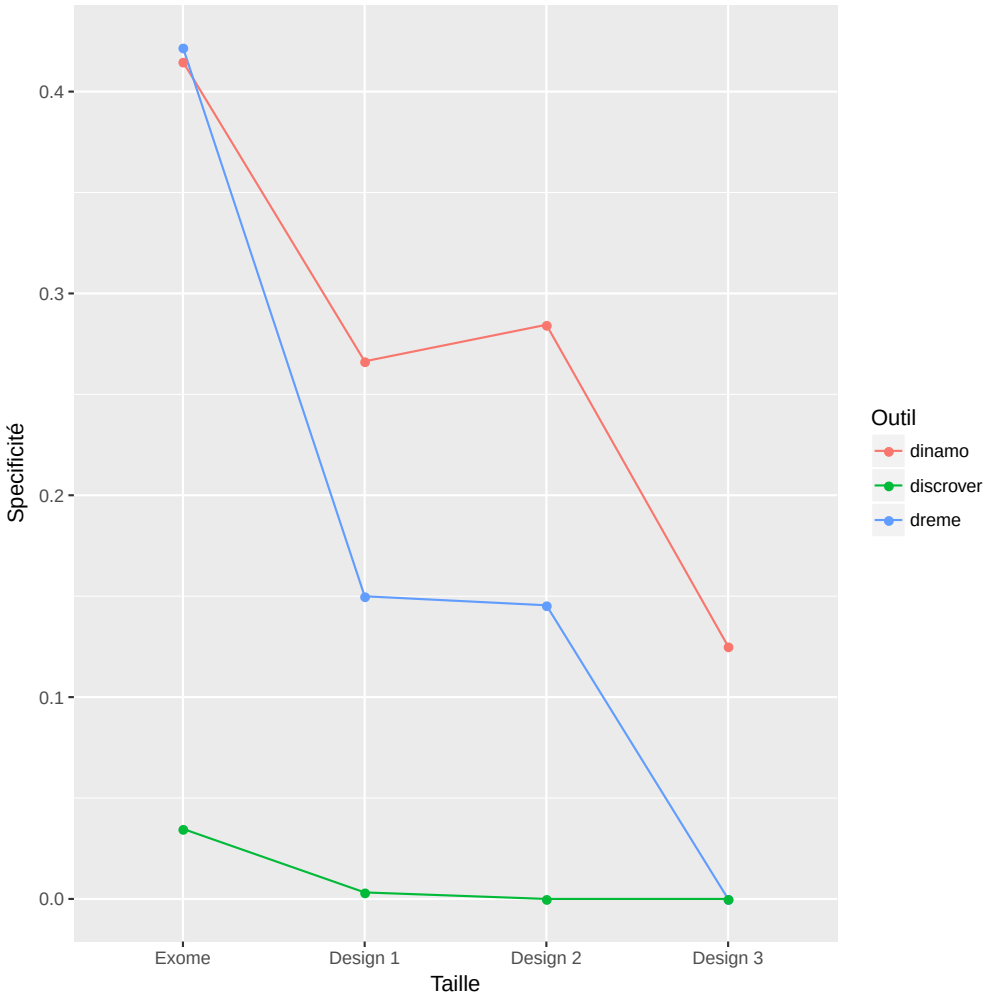


FIGURE 6.12 – Spécificité de la classification des variants pour chacun des outils de recherche de motifs.

6.3 Recherche de motifs liés à des mutations naturelles

Jusqu'à présent les méthodes d'analyse du contexte nucléotidique, qui ont pour objectif la séparation des SSE et des vrais variants, se sont exclusivement focalisées sur les motifs qui favorisent l'apparition des erreurs de séquençage. Nous formulons l'hypothèse que l'outil $DiNAMO$ peut également permettre d'identifier les régions sujettes aux modifications ou à la non réparation des anomalies de l'ADN. De telles régions ou motifs favoriseraient l'apparition de mutations à certaines positions du génome.

Pour tester cette hypothèse, nous avons analysé le contexte nucléotidique d'un jeu de données de variants génomiques en utilisant $DiNAMO$. Il s'agit de la base de données publique ESP (*Exome Sequencing Project*) qui référence des variants (SNP et InDels) obtenus par le séquençage de 6 500 exomes humains.

Cette base de données contient 1 871 001 SNPs, pour lesquels nous avons extrait le contexte nucléotidique (taille 42), afin de les analyser avec $DiNAMO$. La méthode d'extraction du contexte nucléotidique des variants ESP diffère légèrement de celle des SSE décrite dans la section 6.1.1.1. En effet, il s'agit ici de tester s'il y a des motifs liés à l'apparition de mutations naturelles indépendamment du séquençage. L'identification des positions nucléotidiques avec un vrai variant ne permet pas d'utiliser la technique du biais de représentation d'un brin par rapport à l'autre pour identifier le contexte nucléotidique favorisant la mutation. Les deux contextes nucléotidiques, sens et antisens du génome de référence, sont donc extraits (Figure 6.14). Le traitement de ces séquences extraites est ensuite similaire à celui des SSE (cf. section 6.1.1.1) : séparation des séquences de 42 nucléotides en 2 séquences pour construire le jeu de données $\mathscr{P}$ et $\mathscr{N}$ suivi de la recherche de motifs (motifs de taille 2, à la position des variants).

$DiNAMO$ identifie ainsi deux motifs sur-représentés qui sont 'CG' et 'NC', montrant un lien entre les mutations de la base ESP et ces contextes nucléotidiques. L'augmentation de la taille des motifs recherchés ne modifie pas les résultats car cela ne fait qu'ajouter des N en amont. Ces résultats montrent que le processus naturel de mutation des bases n'est pas complètement aléatoire, et qu'il dépend du contexte nucléotidique.

Au vu des ces résultats, une recherche bibliographique a été effectuée. En effet, plu-

sieurs études montrent une hyper-mutabilité du di-nucléotide CG, généralement noté CpG, en TG [18, 136, 138, 210, 213]. Ces mutations sont dues à la méthylation de l'ADN sur la cytosine qui est transformée en 5-Méthylcytosine (5mc) [27]. Les 5mC représentent une position facilement mutable car elle peuvent subir une désamination spontanée en thymine [87], ce qui génère un mesappariement *mismatch* T–G au niveau de la séquence double brin. Il existe un mécanisme de réparation qui reconnaît ce type de mutations (Figure 6.13), mais il n'est pas efficace à 100%, ce qui permet occasionnellement la mutation d'un C en T au cours de la réplication de l'ADN.

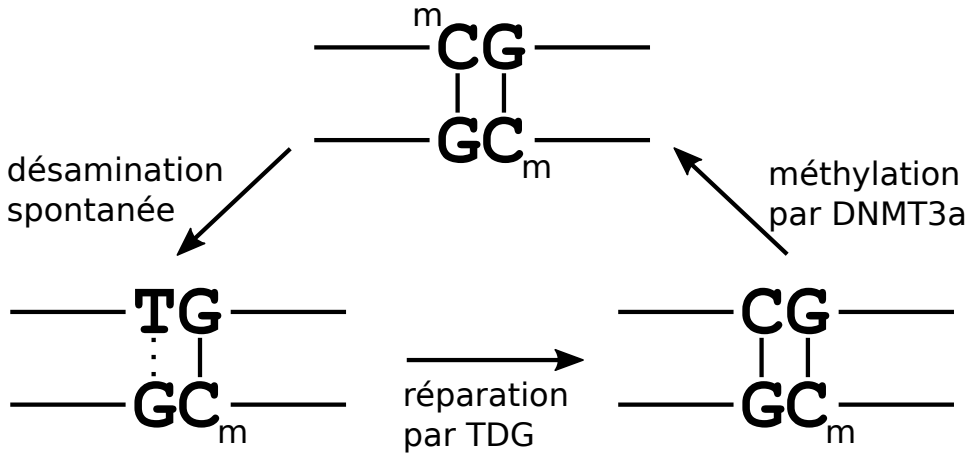


FIGURE 6.13 – Mutation des CpG méthylés en TpG par désamination spontanée. La thymine DNA glycosylase ou G/T mismatch-specific thymine DNA glycosylase (TDG) est une enzyme qui reconnaît et répare les mesappariement T–G. DNMT3a : (pour « DNA methyl-transferase ») est une enzyme qui catalyse la méthylation des CpG.

Cette hyper-mutablité des CpG est probablement la raison pour laquelle les motifs 'CG' et 'NC' sont sur-représentés dans la base de données des variants. En effet, comme la figure 6.14 le montre, la mutation du C dans un di-nucléotide CpG peut induire une sur-représentation du motif NC (à cause d'un niveau de mutabilité élevé de NC → NT). Étant donné que le contexte nucléotidique est extrait sur les 2 brins du génome de référence, le motif CG sera aussi sur-représenté en corrélation avec le motif NC sur l'autre brin (Figure 6.14). Ces motifs correspondent parfaitement aux deux motifs détectés par *DiNAMO*. Pour valider ce lien entre les motifs et l'hyper-mutabilité des CpG, nous avons réalisé une analyse supplémentaire en cherchant des motifs de taille 6, mais centrés autour de la position du variant (au lieu de chercher des motifs à la position des variants). Cette dernière analyse a mis en évidence les motifs NNCGNN et NNNCGN, montrant que c'est bien le di-nucléotide CG qui mute, renforçant l'hypothèse que la

sur-représentation de ces deux motifs est liée au phénomène de l'hyper-mutablité des CpG.

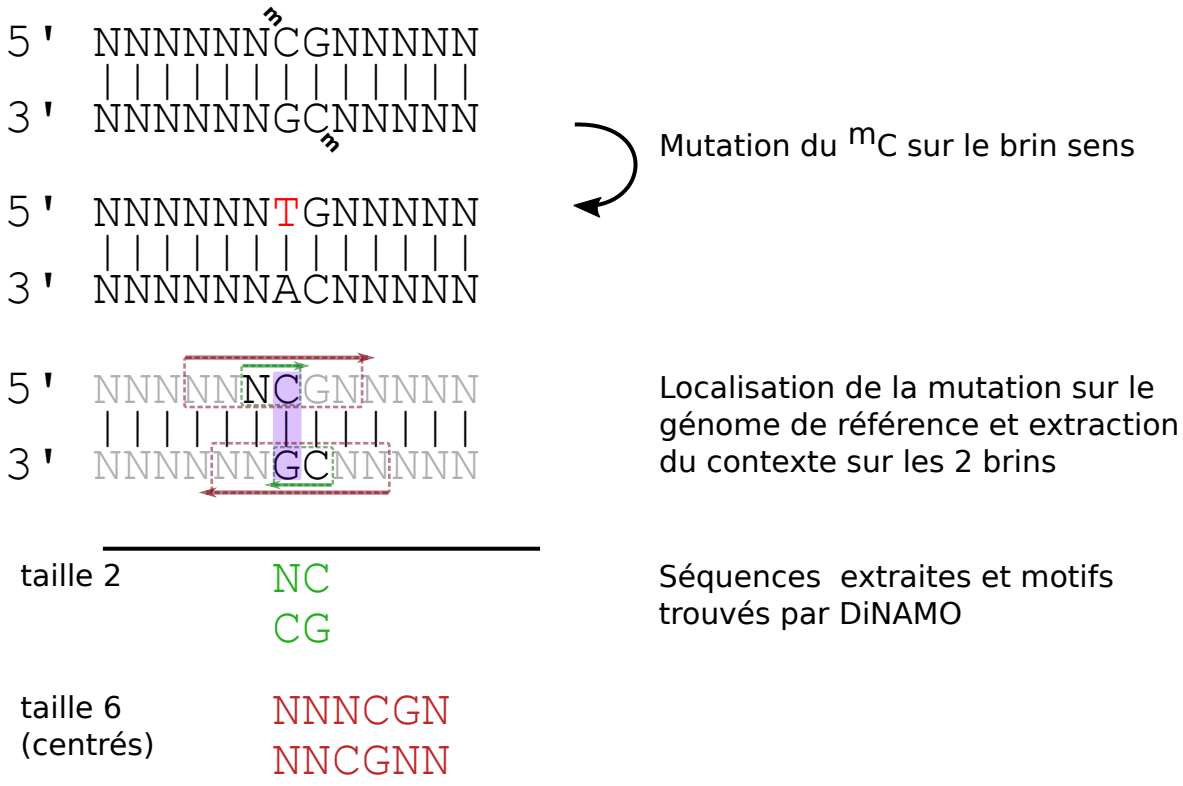


FIGURE 6.14 – Lien entre l'hyper-mutabilité CpG et les motifs CG et NC. Dans le cas d'un dinucléotide CpG, les 2 bases peuvent muter.

CHAPITRE 7

CONCLUSIONS ET PERSPECTIVES

7.1 Conclusions

Dans le chapitre 3, nous avons présenté la problématique liée au taux d'erreur pour les séquenceurs de seconde génération ainsi que les différentes méthodes permettant de filtrer ces erreurs de séquençage et qui sont utilisées par les outils de recherche de variants (cf. section 3.1). Ces méthodes sont basées principalement sur le critère de comptage des *reads* et sont efficaces pour filtrer les erreurs aléatoires.

Les erreurs de séquençage non-alétoires (SSE) restent néanmoins problématiques et difficilement détectables par les approches classiques de filtrage des erreurs, et ce malgré l'utilisation de filtres et de modèles statistiques sophistiqués (cf. section 3.1). Ces approches sont basées sur la recherche de motifs d'ADN, potentiellement responsables de l'apparition des erreurs de séquençage, mais limitées à la recherche de motifs relativement simples (petites tailles ou bien faiblement dégénérés) et souvent utilisables uniquement pour un type de données spécifiques (type de séquenceurs, profondeur de séquençage,...). Or, les études montrent que les SSE peuvent être spécifiques d'une chimie ou d'un *run* de séquençage. Il est donc nécessaire d'avoir une méthode sensible, qui ne nécessite pas des jeux de données d'une taille importante, et rapide pour pouvoir l'appliquer à chaque *run* de séquençage.

Dans ce contexte, les objectifs de la thèse étaient de concevoir et d'implémenter un algorithme énumératif et exact pour la recherche de motifs d'ADN (objectif 1), de proposer une fonction de score pour les variants selon leur contexte nucléotidique (objectif 2) et de les appliquer sur un jeu de données de variants, afin de les évaluer (objectif 3).

En réponse à ces objectifs, le logiciel $DiNAMO$ a été développé dans l'optique d'être sensible pour permettre la détection des motifs rares, et rapide pour pouvoir être exécuté à chaque *run* de séquençage. Pour construire $DiNAMO$, le processus de recherche de motifs a été découpé en deux étapes. Dans un premier temps, l'outil génère tous les motifs dégénérés possibles à partir des motifs exacts qui existent dans le jeu de données d'entrée. Ceci évite de devoir explorer tout l'espace de tous les motifs d'une taille l, dont le nombre augmente de façon exponentielle avec leur taille ($n = 15^l$). La deuxième étape consiste à déterminer les motifs ayant le plus grand pouvoir discriminant (les motifs qui maximisent l'information mutuelle). Cet outil répond à l'**objectif 1** de la thèse.

Les chapitres 5 et 6 montrent les résultats d'évaluation de l'outil $DiNAMO$. L'outil a été évalué avec différents jeux de données. Un jeu de données de séquences synthétiques pour l'évaluation et de la sensibilité de l'outil et deux jeux de données empiriques correspondant à deux applications différentes, recherche de TFBS et identification des SSE. L'évaluation réalisée sur les données synthétiques a montré que $DiNAMO$ a une meilleure sensibilité et spécificité, tout en étant rapide (cf. section 5.1). Sur les deux jeux de données empiriques, $DiNAMO$ montre aussi une meilleure sensibilité en détectant les motifs déjà connus et validés par des études antérieures, ainsi que des nouveaux motifs correspondants à des potentiels co-facteurs de transcription ou des motifs induisant l'apparition des SSE (cf. sections 5.2 et 6.1).

Le chapitre 6 présente une fonction de score (cf. section 6.2.1) et un protocole d'analyse de données en réponse à l'**objectif 2** de la thèse. La fonction de score classe le variant entre vrai variant ou SSE, en comparant le nombre d'occurrences d'un variant à une position donnée à ceux attendus étant donné le motif en amont .

L'outil $DiNAMO$ a ensuite été appliqué à un jeu de données construit à partir de données de séquençage de haute qualité, générées dans le cadre du projet GIAB et disponibles publiquement en ligne (chapitre 6). Cette application répond à l'**objectif 3** de la thèse. $DiNAMO$ a permis la détection de centaines de SSE qui ont été considérées

comme des variants par l'outil de détection de variants *Freebayes*. Les variants et leur classement en SSE ont été comparés à un ensemble de variants préalablement validés (HC), montrant que l'intégration de l'information des motifs, détectés par $DiNAMO$, durant le processus d'appel des variants, améliore les résultats et réduit le taux de faux positifs.

7.2 Perspectives

Dans le chapitre 6, nous avons montré l'impact de l'utilisation des motifs identifiés par $DiNAMO$ sur la qualité des résultats d'appel de variants. En effet, les résultats dans la section 6.2.3.2 montrent une forte corrélation entre le classement des variants par notre fonction de score et leur statut de validation parmi les variants préalablement validés.

À court terme, d'autres fonctions de score pourraient être testées en intégrant plus de variables. Par exemple, on peut intégrer l'information du score qualité des variants ou bien de leurs bases flanquantes. Une autre information importante qui pourrait être intégrée dans l'information de score est le type de la mutation. Par exemple, les deux substitutions C>T et C>A après un même motif m ont, pour le moment, le même score. On pourrait prendre en compte les bases de référence et les bases alternatives dans la fonction de score. L'ajout de ces variables dans notre fonction de score améliorerait probablement le classement des variants.

Il serait possible de prendre en compte l'information complémentaire sur le motif CG favorisant la mutation naturelle du C quand il est méthylé. Ce motif donnent une certaine fiabilité au variant qui a plus de chance d'être un vrai variant, ce qui pourrait être considéré dans le score final. Dans la même optique, il est également possible d'ajouter l'information sur les motifs défavorisant l'apparition des erreurs de séquençage. Il s'agit des motifs sous-représentés en amont des SSE, peuvent être obtenus avec $DiNAMO$ en inversant tout simplement les deux fichiers d'entrée ($\mathscr{P}$ et $\mathscr{N}$). Le taux d'erreur attendu derrière ces motifs peut être pris en compte par la fonction de score de la même façon que les motifs sur-représentés.

Une fonctionnalité qui pourrait également simplifier l'utilisation de $DiNAMO$ est la détection automatique de la longueur optimale du motif. Dans la version actuelle, c'est

l'utilisateur qui choisit la taille de motifs à rechercher (paramètre l). La détermination de la longueur optimale des motifs pourrait être réalisée en comparant des motifs de tailles différentes au sein du même treillis. Par exemple, le motif AAAA serait comparé directement au motif AAAAA dans le treillis. Cela impliquerait une autre méthode pour le parcours des fichiers de séquences afin de compter et construire le treillis à partir de motifs de tailles différentes.

Lors de nos analyses sur les données GIAB, les InDels ont été écartées à cause de la difficulté de les compter. En effet, il est difficile de connaître exactement la position d'un InDel dans les régions de faible complexité. De ce fait, une même InDel peut être représentée de plusieurs façons différentes [191], ce qui par conséquence, fausse les comptages des variants utilisés par la fonction de score. Pour pallier ce problème, une étape de normalisation avec des logiciels dédiés sera indispensable, ainsi qu'une réflexion sur une méthode de comptage qui sera adaptée aux InDels dans les régions de faible complexité.

Le but final est d'utiliser $DiNAMO$ pour analyser les données de chaque *run* de séquençage. L'analyse se déroulera en deux étapes. Premièrement, $DiNAMO$ sera lancé sur l'ensemble des fichiers BAM (les *reads* alignés) générés par le *run* de séquençage pour identifier les motifs induisant l'apparition des SSE. Ces motifs seront utilisés par la suite, pour analyser les variants détectés par le logiciel d'appel de variants (les fichiers VCF), à l'aide de la fonction de score, afin d'identifier et filtrer les SSE (comme montré dans le chapitre 6). Cela pourrait permettre de réduire les seuils de profondeur utilisés pour filtrer les variants, permettant ainsi la détection des variants avec un faible ratio allélique (VAF) dans le cadre des analyses somatiques, tout en gardant une bonne spécificité en éliminant les erreurs de séquençage non aléatoires (SSE).

ANNEXE A

ANNEXE

A.1 Les motifs implantés dans les données synthétiques

	IUPAC content	6	8	10	12	14
Number of implanted motifs						
1		GCCGAT	CTGVCG	CNWCGG	YBATNT	HWWHCB
		AAGAGT	AWACRA	SCWSKT	SNTTKK	GRMNHS
		CCTCAG	RACCTS	KMTCYS	VCAHAH	YVBDTM
		AAAAGC	RTGCGS	TKSRKA	DSAYRY	BRCNYW
		CTTTGC	GMRTCT	RKCTDA	NSCGKS	TVSDNA
2		GATTTC,TGAGGG	MGTKAC,RCGCGK	GYTGWB,MTWASW	RNSTCW,TRRWHR	RHMYDK,GYBMNR
		AGCCTT,ACGTCG	CTVTAG,AYMTCA	BKTRTG,KCTYYW	WYRHTM,SCSDBG	TNHSKK,KHHHAS
		TTAGGT,TTTTTT	CDCCTA,MTGGCS	HAYTMA,RTBGMA	DKYADG,GHRRBT	BNCRRY,CNGSHH
		CAGCTC,GCAGCG	ACMTCR,DGTACT	HGAYTS,KVSCAT	SAVWRR,CYTDBM	RYRRVB,CSWMVN
		ATATTT,TTCAGT	ATYCRT,CCMTYC	AKYTWM,BKWTAT	MSADRY,MVRWTS	DDDDTG,DYVDCY
3		GGTGCG,AGTGGC,TTAGGT	TTRCCK,TCTGBC,GAGGSW	ACCRBY,ARKCRS,GKRKRC	CVSCGN,BSTGBR,NWWATS	MVKSTN,MADDMH,BYWHRK
		TTCCGT,CGATGT,GGGCAC	GCMTSG,TCTCSS,AACWWC	BCSTCM,TWCSMK,CRCBKG	DRCKDG,ATTNWD,RHTAHS	BGKYNS,SNWRSM,HBRMGD
		ATGGTG,CCGTGT,ACTGTT	KACCAM,TAGWRG,AWWGCT	AKKWWC,TKASKK,ASWWCS	DSSVCC,RBKTKS,BKCARB	YDMVSY,HBVAWK,BMRMDS
		GGATTT,GAAAGT,TTACTA	ASCKAC,TDTTCC,RCCCSG	AYBAYC,RYSCMG,TGYCSH	MTHKWK,HCGBMR,HNCRCC	DSHSVC,BHHARY,AHSWHH
		TGAACT,TGTACC,TCTCCT	CGTSAS,AGGAHC,CYATSA	YTYAYR,VMRCTG,CCYMVG	YWACHB,WSBMYA,MMDMKT	HKWBBT,SVTSNK,WCSNDY
4		CCGCTG,CGGTTC,TTCGCC,AAACGC	TTGMCW,YAAAST,AVCACA,RCTCTK	HMGWAG,MCCDYA,SRYRTC,AWRHCA	GWHWYK,YBWDCT,TBMKYK,AVACWN	GABDVH,KKHWRH,WANYHM,MMBVKK
		CTAAGA,CATGCG,AGTGGC,GCACAG	AGGKST,MACGYG,GKWTCT,GCAMAK	CBTMAY,RAWHTT,VSATAM,CHHGGT	HRMSWA,YYYYRK,SHKWGW,WSHKGK	HBCKBW,KSTDKN,KWBWYB,HBKGMB
		GCCGCT,GCCCCC,CATTGT,TGTTGA	ACRGAS,TWTSGC,AMTAGY,SWGCGC	YTBTYC,RAMAKK,TTSWKK,CGTTWN	TDWSRW,SGYVVG,SGVKWY,SRBRGW	MAHDDM,MVHWDT,YMABVB,THDVSW
		ATTGCT,GCAGCT,GGAACC,CCCAAG	AMYGCC,TKTGGS,TGAMTW,WGRCTT	MKRYCG,KGAWKM,YYSCTS,RCAKHC	HSDCSC,YWKSDG,STDBCR,NYCGWK	HSDRYS,DMMSRB,WMKCNV,HWDHGY
		ACCAAA,GATAGG,CAGTTC,CTCGTC	TCKWGC,MCSTCG,AATYAR,TKWCCA	VRGCCS,MCMMKC,BYGKGG,WYAAMM	CTSRRN,NWAGCD,RGVMRS,YMWMTH	BVSBGW,TWSHMN,VWTYVD,NRCDBT

TABLE A.1 – Les ensembles de motifs IUPAC générés aléatoirement et implantés dans les séquences du jeu de données synthétiques.

A.2 Les co-facteurs détectés et leur statut de validation

	DINAMO	MEMECHIP	DISCROVER	HOMER
GATA1	STAT1	STAT1	KLF1	MYC
	OTX1	KLF1	STAT1	KLF4
	MAX	RUNX1	RUNX1	SPI1
	DUXA	OTX2		STAT1
	SP1	Ahr::Arnt		ID2
	Myog	Myog		OTX2
	TAL1::TCF3	KLF5		TGIF1
	Hmx3	FOXP2		MTF1
	PROX1	FLI1		HOXB13
	YY1	YY1		
	FOSL1	FOSL1		
	Ahr::Arnt			
	ELK4			
	MTF1			
	XBP1			
	MYC			
	SREBF1			
	CENPB			
SOX2	POU3F4	POU3F4	POU3F4	Pou2f3
	KLF5	POU4F2	ZIC1	E2F4
	mix-a	KLF5	KLF5	EGR1
	Nr5a2	ZIC1	POU4F2	ELK4
	SCRT2	ESRRB		FOXO3
	USF2	EWSR1-FLI1		mix-a
	PROP1	TEF		ESRRB
	BHLHE22	Klf1		Tcf7
	TEAD3	mix-a		Pparg::Rxra
	CDX2	SCRT1		HES7
	STAT1	E2F4		
	POU3F2	TEAD3		
	FOXO3	POU2F2		
	NRF1	SOX8		
	NR1H2::RXRA	ETV2		
	BCL6B	LIN54		
	ELF4			
KLF1	GATA1/2	GATA1/2	GATA1/2	SP3
	Tcf12	KLF13	KLF16	GATA1/2
	Dmbx1			OTX2
	E2F7			TFAP4
	AR			FOXH1

	DINAMO	MEMECHIP	DISCROVER	HOMER
OCT4	SPIC	POU3F4	Sox6	POU2F2
	POU3F4	POU4F2	Pou2f3	Sox17
	KLF5	Sox6	KLF5	POU3F4
	NR4A2	KLF5	SP1	ELK1
	MAFG::NFE2L1	SPIC		GATA1::TAL1
	ETV6	ESRRB		SP1
	ZIC1	ELK4		ESRRB
	FOXG1	TCF4		Klf1
	MEF2C	LIN54		SP4
	MAFK	BCL6B		Creb5
	GSC			TEAD3
	ESRRB			ID2
	RELA			
STAT3	NR4A2	KLF1	GATA2	SP3
	Ascl2	GATA2	KLF1	GATA2
	KLF4	KLF13		KLF5
	ESRRG			OTX2
	SP1			TFAP4
	GSC			FOXH1
	ID2			NFAT5
	BCL6B			
	POU6F2			
	FOSL1			
	POU4F1			
	TBX15			
	ELK1			
	GLIS1			
	TCF7L2			
	SRF			
	ELF3			

TABLE A.2 – Comparaison des motifs détectés par les 3 différents logiciels. Les motifs sont triés selon leur score donné par chaque programme. L'annotation est effectuée par l'outil TOMTOM [21] et la base de données Jaspar [129]. Pour les motifs qui correspondent à plusieurs annotations dans la base de données Jaspar, nous rapportons dans ce tableau et nous testons dans *Ingenuity Pathway Analysis* uniquement la première. Les cofacteurs prédits sont colorés en vert s'il existe une interaction «protéine-protéine/ADN» avec la FT principal, décrite dans les bases de données IPA de haute confiance ("*Ingenuity expert findings*" et "*Experimentally observed*"). En jaune, les interactions trouvés dans toutes les bases de données IPA (interactions prédites ou non validés expérimentalement). Version du logiciel IPA : Build : 463341M, content : 42012434.

BIBLIOGRAPHIE

12. ABOUELHODA, M. & GHANEM, M. in *Scientific Data Mining and Knowledge Discovery* 207–247 (Springer, 2009).

14. AGOSTINI, F., CIRILLO, D., PONTI, R. D. & TARTAGLIA, G. G. SeAMotE: a method for high-throughput motif discovery in nucleic acid sequences. *BMC genomics* **15,** 925 (2014).

15. ALKAN, C., COE, B. P. & EICHLER, E. E. Genome structural variation discovery and genotyping. *Nature Reviews Genetics* **12,** 363 (2011).

16. ALLHOFF, M. *et al. Discovering motifs that induce sequencing errors* in *BMC bioinformatics* **14** (2013), S1.

17. AMEZIANE, N., BOGARD, M. & LAMORIL, J. *Principes de biologie moléculaire en biologie clinique* ISBN : 9782842996857. `https://books.google.fr/books?id=cwxYehzAoz8C` (Elsevier, 2005).

18. ANTONARAKIS, S. E. CpG dinucleotides and human disorders. *eLS* (2006).

19. BAEZA-YATES, R. & NAVARRO, G. Faster approximate string matching. *Algorithmica* **23,** 127–158 (1999).

20. BAILEY, T. L. DREME: motif discovery in transcription factor ChIP-seq data. *Bioinformatics* **27,** 1653–1659 (2011).

21. BAILEY, T. L. *et al.* MEME SUITE: tools for motif discovery and searching. *Nucleic acids research,* gkp335 (2009).

22. BATS, A., CELLIER, C., SAMAHA, E., LAURENT-PUIG, P. & LECURU, F. Lynch syndrome: Towards a multidisciplinary management of tumour screening. *Gynecologie, obstetrique & fertilite* **39,** 272 (2011).

23. BAUM, L. E., PETRIE, T., SOULES, G. & WEISS, N. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The annals of mathematical statistics* **41,** 164–171 (1970).

24. BECK, J., URNOVITZ, H. B., MITCHELL, W. M. & SCHÜTZ, E. Next generation sequencing of serum circulating nucleic acids from patients with invasive ductal breast cancer reveals differences to healthy and nonmalignant controls. *Molecular cancer research* **8,** 335–342 (2010).

25. BERRIDGE, M. J. Module 4: Sensors and Effectors. *Cell Signalling Biology* **6.** ISSN : 1749-7787. doi :10.1042/csb0001004. http://csb.portlandpresspublishing.com/content/6/csb0001004 (2014).

26. BEYENS, M., BOECKX, N., VAN CAMP, G., de BEECK, K. O. & VANDEWEYER, G. pyAmpli: an amplicon-based variant filter pipeline for targeted resequencing data. *BMC bioinformatics* **18,** 554 (2017).

27. BIRD, A. P. DNA methylation and the frequency of CpG in animal DNA. *Nucleic Acids Research* **8,** 1499–1504 (1980).

28. BOYER, R. S. & MOORE, J. S. A fast string searching algorithm. *Communications of the ACM* **20,** 762–772 (1977).

29. BRENT, R. P. *Algorithms for minimization without derivatives* (Courier Corporation, 2013).

30. BURSET, M. & GUIGO, R. Evaluation of gene structure prediction programs. *genomics* **34,** 353–367 (1996).

31. CARROT-ZHANG, J. & MAJEWSKI, J. LoLoPicker: Detecting Low-Fraction Variants in Low-Quality Cancer Samples from Whole-exome Sequencing Data. *bioRxiv,* 043612 (2016).

32. CHEN, Q. K., HERTZ, G. Z. & STORMO, G. D. MATRIX SEARCH 1.0: a computer program that scans DNA sequences for transcriptional elements using a database of weight matrices. *Bioinformatics* **11,** 563–566 (1995).

33. CHEN, X. *et al.* Integration of external signaling pathways with the core transcriptional network in embryonic stem cells. *Cell* **133,** 1106–1117 (2008).

34. CHENG, A. Y., TEO, Y.-Y. & ONG, R. T.-H. Assessing single nucleotide variant detection and genotype calling on whole-genome sequenced individuals. *Bioinformatics* **30,** 1707–1713 (2014).

35. CHENG, L., LOPEZ-BELTRAN, A., MASSARI, F., MACLENNAN, G. T. & MONTIRONI, R. Molecular testing for BRAF mutations to inform melanoma treatment decisions: a move toward precision medicine. *Modern Pathology* **31,** 24 (2018).

36. CHENG, Y. *et al.* Erythroid GATA1 function revealed by genome-wide analysis of transcription factor occupancy, histone modifications, and mRNA expression. *Genome research* **19,** 2172–2184 (2009).

37. CHIU, R. W. *et al.* Non-invasive prenatal assessment of trisomy 21 by multiplexed maternal plasma DNA sequencing: large scale validity study. *Bmj* **342,** c7401 (2011).

38. CHOMSKY, N. Syntactic Structures, Mouton, La Haye. *Trad. fr.: Le Seuil, Paris* (1957).

39. CIBULSKIS, K. *et al.* Sensitive detection of somatic point mutations in impure and heterogeneous cancer samples. *Nature biotechnology* **31,** 213 (2013).

40. CLANCY, S. Genetic mutation. *Nature Education* **1,** 187 (2008).

41. CLEVENGER, J., CHAVARRO, C., PEARL, S. A., OZIAS-AKINS, P. & JACKSON, S. A. Single nucleotide polymorphism identification in polyploids: a review, example, and recommendations. *Molecular plant* **8,** 831–846 (2015).

42. CONSORTIUM, 1. G. P. *et al.* A map of human genome variation from population-scale sequencing. *Nature* **467,** 1061 (2010).

43. CONSORTIUM, I. H. G. S. *et al.* Initial sequencing and analysis of the human genome. *Nature* **409,** 860 (2001).

44. COPPIN, L. *et al.* VHL mosaicism can be detected by clinical next-generation sequencing and is not restricted to patients with a mild phenotype. *European Journal of Human Genetics* **22,** 1149 (2014).

45. CROCHEMORE, M. & RYTTER, W. Text algorithms: Oxford University Press. *Inc. New York, NY, USA* (1994).

46. CROCHEMORE, M., HANCART, C. & LECROQ, T. *Algorithmique du texte* (Vuibert Paris, 2001).

47. DAS, M. K. & DAI, H.-K. A survey of DNA motif finding algorithms. *BMC bioinformatics* **8,** S21 (2007).

48. DEAKIN, C. T. *et al.* Impact of next-generation sequencing error on analysis of barcoded plasmid libraries of known complexity and sequence. *Nucleic acids research* **42,** e129–e129 (2014).

49. DEBARRI, H. *et al.* IDH1/2 but not DNMT3A mutations are suitable targets for minimal residual disease monitoring in acute myeloid leukemia patients: a study by the Acute Leukemia French Association. *Oncotarget* **6,** 42345 (2015).

50. DEFRANCE, M., SAND, O., VAN HELDEN, J. *et al.* Using RSAT oligo-analysis and dyad-analysis tools to discover regulatory signals in nucleic sequences. *Nature protocols* **3,** 1589 (2008).

51. DEMPSTER, A. P., LAIRD, N. M. & RUBIN, D. B. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological),* 1–38 (1977).

52. DENISSENKO, M. F., PAO, A., TANG, M.-s. & PFEIFER, G. P. Preferential formation of benzo [a] pyrene adducts at lung cancer mutational hotspots in P53. *Science* **274,** 430–432 (1996).

53. DEPRISTO, M. A. *et al.* A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nature genetics* **43,** 491 (2011).

54. DEWEY, F. E., PAN, S., WHEELER, M. T., QUAKE, S. R. & ASHLEY, E. A. DNA sequencing: clinical applications of new DNA sequencing technologies. *Circulation* **125,** 931–944 (2012).

55. D'HAESELEER, P. How does DNA sequence motif discovery work? *Nature biotechnology* **24,** 959 (2006).

56. DOWTY, J. G. *et al.* Cancer risks for MLH1 and MSH2 mutation carriers. *Human mutation* **34,** 490–497 (2013).

57. DURBIN, R., EDDY, S. R., KROGH, A. & MITCHISON, G. *Biological sequence analysis: probabilistic models of proteins and nucleic acids* (Cambridge university press, 1998).

58. EDDY, S. R. Accelerated profile HMM searches. *PLoS computational biology* **7,** e1002195 (2011).

59. EDDY, S. R. Hidden markov models. *Current opinion in structural biology* **6,** 361–365 (1996).

60. EDDY, S. R. Profile hidden Markov models. *Bioinformatics (Oxford, England)* **14,** 755–763 (1998).

61. EDGAR, R., DOMRACHEV, M. & LASH, A. E. Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic acids research* **30,** 207–210 (2002).

62. ELEMENTO, O., SLONIM, N. & TAVAZOIE, S. A universal framework for regulatory element discovery across all genomes and data types. *Molecular cell* **28,** 337–350 (2007).

63. FAN, H. C., BLUMENFELD, Y. J., CHITKARA, U., HUDGINS, L. & QUAKE, S. R. Noninvasive diagnosis of fetal aneuploidy by shotgun sequencing DNA from maternal blood. *Proceedings of the National Academy of Sciences* **105,** 16266–16271 (2008).

64. FERRAGINA, P. & MANZINI, G. Indexing compressed text. *Journal of the ACM (JACM)* **52,** 552–581 (2005).

65. FEUK, L., CARSON, A. R. & SCHERER, S. W. Structural variation in the human genome. *Nature Reviews Genetics* **7,** 85 (2006).

66. FISHER, R. A. On the interpretation of χ 2 from contingency tables, and the calculation of P. *Journal of the Royal Statistical Society* **85,** 87–94 (1922).

67. FOULKES, W. & REAL, F. Many mosaic mutations. *Current Oncology* **20,** 85 (2013).

68. FOX, E. J., REID-BAYLISS, K. S., EMOND, M. J. & LOEB, L. A. Accuracy of next generation sequencing platforms. *Next generation, sequencing & applications* **1** (2014).

69. FRAENKEL, Y. M., MANDEL, Y., FRIEDBERG, D. & MARGALIT, H. Identification of common motifs in unaligned DNA sequences: application to Escherichia coli Lrp regulon. *Bioinformatics* **11,** 379–387 (1995).

70. FREBOURG, T., MAUILLON, J., THOMAS, G. & OLSCHWANG, S. Hereditary nonpolyposis colorectal cancer. Definition, genetics, diagnosis, and medical surveillance. *Gastroenterologie clinique et biologique* **27,** 708 (2003).

71. GARRISON, E. & MARTH, G. Haplotype-based variant detection from short-read sequencing. *arXiv preprint arXiv:1207.3907* (2012).

72. GATES, K. S. An overview of chemical processes that damage cellular DNA: spontaneous hydrolysis, alkylation, and reactions with radicals. *Chemical research in toxicology* **22,** 1747–1760 (2009).

73. GERSTUNG, M. *et al.* Reliable detection of subclonal single-nucleotide variants in tumour cell populations. *Nature communications* **3,** 811 (2012).

74. GISSELBRECHT, S. Oncogènes et leucémies: historique et perspectives. *médecine/sciences* **19,** 201–210 (2003).

75. GLENN, T. C. Field guide to next-generation DNA sequencers. *Molecular ecology resources* **11,** 759–769 (2011).

76. GOEBEL, B., DAWY, Z., HAGENAUER, J. & MUELLER, J. C. *An approximation to the distribution of finite sample size mutual information estimates* in *Communications, 2005. ICC 2005. 2005 IEEE International Conference on* **2** (2005), 1102–1106.

77. GOODWIN, S., MCPHERSON, J. D. & MCCOMBIE, W. R. Coming of age: ten years of next-generation sequencing technologies. *Nature Reviews Genetics* **17,** 333 (2016).

78. GRIFFITHS, A., MILLER, J., SUZUKI, D., LEWONTIN, R. & GELBART, W. Spontaneous mutations. *An Introduction to Genetic Analysis* (2000).

79. GRIFFITHS, A. *Introduction à l'analyse génétique* ISBN : 9782744500978. `https://books.google.fr/books?id=jIatswEACAAJ` (De Boeck Université, 2002).

80. GRUNDY, W. N., BAILEY, T. L., ELKAN, C. P. & BAKER, M. E. Meta-MEME: motif-based hidden Markov models of protein families. *Bioinformatics* **13,** 397–406 (1997).

81. HADIGOL, M. & KHIABANIAN, H. MERIT reveals the impact of genomic context on sequencing error rate in ultra-deep applications. *BMC Bioinformatics* **19,** 219 (2018).

82. HAKEM, R. DNA-damage repair; the good, the bad, and the ugly. *The EMBO journal* **27,** 589–605 (2008).

83. HARBISON, C. T. *et al.* Transcriptional regulatory code of a eukaryotic genome. *Nature* **431,** 99 (2004).

84. HEINZ, S. *et al.* Simple combinations of lineage-determining transcription factors prime cis-regulatory elements required for macrophage and B cell identities. *Molecular cell* **38,** 576–589 (2010).

85. HELDEN, J. v., RIOS, A., COLLADO-VIDES, J. *et al.* Discovering regulatory elements in non-coding sequences by analysis of spaced dyads. *Nucleic acids research* **28,** 1808–1818 (2000).

86. HENRY, V. J., BANDROWSKI, A. E., PEPIN, A.-S., GONZALEZ, B. J. & DESFEUX, A. OMICtools: an informative directory for multi-omic data analysis. *Database* **2014** (2014).

87. HOLLIDAY, R. & GRIGG, G. DNA methylation and mutation. *Mutation Research / Fundamental and Molecular Mechanisms of Mutagenesis* **285,** 61–67 (1993).

88. HOLM, S. A Simple Sequentially Rejective Multiple Test Procedure. *Scandinavian Journal of Statistics* **6,** 65–70 (1979).

89. HUGHES, J. D., ESTEP, P. W., TAVAZOIE, S. & CHURCH, G. M. Computational identification of cis-regulatory elements associated with groups of functionally related genes in Saccharomyces cerevisiae. *Journal of molecular biology* **296,** 1205–1214 (2000).

90. HUTTER, M. *Distribution of mutual information* in *Advances in neural information processing systems* (2002), 399–406.

91. HWANG, S., KIM, E., LEE, I. & MARCOTTE, E. M. Systematic comparison of variant calling pipelines using gold standard personal exome variants. *Scientific reports* **5,** 17875 (2015).

92. IQBAL, Z., CACCAMO, M., TURNER, I., FLICEK, P. & MCVEAN, G. De novo assembly and genotyping of variants using colored de Bruijn graphs. *Nature genetics* **44,** 226 (2012).

93. JÄRVINEN, H. J. *et al.* Controlled 15-year trial on screening for colorectal cancer in families with hereditary nonpolyposis colorectal cancer. *Gastroenterology* **118,** 829–834 (2000).

94. JONASSEN, I. Efficient discovery of conserved patterns using a pattern graph. *Bioinformatics* **13,** 509–522 (1997).

95. KANAGAWA, T. Bias and artifacts in multitemplate polymerase chain reactions (PCR). *Journal of bioscience and bioengineering* **96,** 317–323 (2003).

96. KARP, G., ISAWA, J. & MARSHALL, W. *Biologie cellulaire et moléculaire* ISBN : 9782807308015. https://books.google.fr/books?id=nR9aDwAAQBAJ (De Boeck supérieur, 2018).

97. KAWALIA, A. *Addressing NGS Data Challenges: Efficient High Throughput Processing and Sequencing Error Detection* thèse de doct. (Universität zu Köln, 2015).

98. KELLY JR, T. J. & SMITH, H. O. A restriction enzyme from Hemophilus influenzae: II. Base sequence of the recognition site. *Journal of molecular biology* **51,** 393–409 (1970).

99. KIELY, A. P. *et al.* α-Synucleinopathy associated with G51D SNCA mutation: A link between Parkinson's disease and multiple system atrophy? *Acta neuropathologica* **125,** 753–769 (2013).

100. KISSELEVA, T., BHATTACHARYA, S., BRAUNSTEIN, J. & SCHINDLER, C. Signaling through the JAK/STAT pathway, recent advances and future challenges. *Gene* **285,** 1–24 (2002).

101. KNÜSEL, L. & BABLOK, B. Computation of the noncentral gamma distribution. *SIAM Journal on Scientific Computing* **17,** 1224–1231 (1996).

102. KNUTH, D. E., MORRIS Jr, J. H. & PRATT, V. R. Fast pattern matching in strings. *SIAM journal on computing* **6,** 323–350 (1977).

103. KO, L. & ENGEL, J. DNA-binding specificities of the GATA transcription factor family. *Molecular and cellular biology* **13,** 4011–4022 (1993).

104. KOBOLDT, D. C. *et al.* VarScan 2: somatic mutation and copy number alteration discovery in cancer by exome sequencing. *Genome research* **22,** 568–576 (2012).

105. KOHLMANN, A. *et al.* The Interlaboratory RObustness of Next-generation sequencing (IRON) study: a deep sequencing investigation of TET2, CBL and KRAS mutations by an international consortium involving 10 laboratories. *Leukemia* **25,** 1840 (2011).

106. KORANNE, S. in *Handbook of Open Source Tools* 127–143 (Springer, Boston, MA, 2011).

107. KROGH, A., BROWN, M., MIAN, I. S., SJÖLANDER, K. & HAUSSLER, D. Hidden Markov models in computational biology: Applications to protein modeling. *Journal of molecular biology* **235,** 1501–1531 (1994).

108. KURTZ, S. *Approximate string searching under weighted edit distance* in *Proc. WSP* **96** (1996), 156–170.

109. LANDAU, G. M. & VISHKIN, U. Fast parallel and serial approximate string matching. *Journal of algorithms* **10,** 157–169 (1989).

110. LANGMEAD, B. & SALZBERG, S. L. Fast gapped-read alignment with Bowtie 2. *Nature methods* **9,** 357 (2012).

111. LATRUFFE, N., BLEICHER-BARDELETTI, F., DUCLOS, B. & VAMECQ, J. *Biochimie - Tout le cours en fiches - 2e éd: 200 fiches de cours, 155 QCM, sujets de synthèse et ressources en ligne* ISBN : 9782100765959. `https://books.google.fr/books?id=ggUqDwAAQBAJ` (Dunod, 2017).

112. LAWRENCE, C. E. *et al.* Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *SCIENCE-NEW YORK THEN WASHINGTON-* **262,** 208–208 (1993).

113. LECROQ, B. *et al.* Ultra-deep sequencing of foraminiferal microbarcodes unveils hidden richness of early monothalamous lineages in deep-sea sediments. *Proceedings of the National Academy of Sciences* **108,** 13177–13182 (2011).

114. LEINONEN, R., SUGAWARA, H., SHUMWAY, M. & COLLABORATION, I. N. S. D. The sequence read archive. *Nucleic acids research* **39,** D19–D21 (2010).

115. LI, H. Toward better understanding of artifacts in variant calling from high-coverage samples. *Bioinformatics* **30,** 2843–2851 (2014).

116. LI, H. & DURBIN, R. Fast and accurate short read alignment with Burrows–Wheeler transform. *Bioinformatics* **25,** 1754–1760 (2009).

117. LIU, J. S., NEUWALD, A. F. & LAWRENCE, C. E. Bayesian models for multiple local sequence alignment and Gibbs sampling strategies. *Journal of the American Statistical Association* **90,** 1156–1170 (1995).

118. LIU, X., HAN, S., WANG, Z., GELERNTER, J. & YANG, B.-Z. Variant callers for next-generation sequencing data: a comparison study. *PloS one* **8,** e75619 (2013).

119. LIU, X., BRUTLAG, D. L., LIU, J. S. *et al. BioProspector: discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes.* in *Pacific symposium on biocomputing* **6** (2001), 127–138.

120. LIU, X., WU, J., GU, F., WANG, J. & HE, Z. Discriminative pattern mining and its applications in bioinformatics. *Briefings in bioinformatics* **16,** 884–900 (2014).

121. LOMAN, N. J. *et al.* Performance comparison of benchtop high-throughput sequencing platforms. *Nature biotechnology* **30,** 434 (2012).

122. MAASKOLA, J. *Discriminative learning for probabilistic sequence analysis* thèse de doct. (Dept. of Computational Molecular Biology (Head: Martin Vingron), Max Planck Institute for Molecular Genetics, Max Planck Society, 2015).

123. MAASKOLA, J. & RAJEWSKY, N. Binding site discovery from nucleic acid sequences by discriminative learning of Hidden Markov Models. *Nucleic acids research* **42,** 12995–13011 (2014).

124. MACHANICK, P. & BAILEY, T. L. MEME-ChIP: motif analysis of large DNA datasets. *Bioinformatics* **27,** 1696–1697 (2011).

125. MACKELPRANG, R. *et al.* Metagenomic analysis of a permafrost microbial community reveals a rapid response to thaw. *Nature* **480,** 368 (2011).

126. MANBER, U. & MYERS, G. Suffix arrays: a new method for on-line string searches. *siam Journal on Computing* **22,** 935–948 (1993).

127. MANCHERON, A. *Extraction de Motifs Communs dans un Ensemble de Séquences. Application à l'identification de sites de liaison aux protéines dans les séquences primaires d'ADN.* thèse de doct. (Université de Nantes, 2006).

128. MARSCHALL, T. & RAHMANN, S. Efficient exact motif discovery. *Bioinformatics* **25,** i356–i364 (2009).

129. MATHELIER, A. *et al.* JASPAR 2016: a major expansion and update of the open-access database of transcription factor binding profiles. *Nucleic acids research* **44,** D110–D115 (2016).

130. MCCREIGHT, E. M. A space-economical suffix tree construction algorithm. *Journal of the ACM (JACM)* **23,** 262–272 (1976).

131. MEACHAM, F. *et al.* Identification and correction of systematic error in high-throughput sequence data. *BMC bioinformatics* **12,** 451 (2011).

132. MEDINA-RIVERA, A. *et al.* RSAT 2015: regulatory sequence analysis tools. *Nucleic acids research,* gkv362 (2015).

134. MILLER, C. A. *et al.* SciClone: inferring clonal architecture and tracking the spatial and temporal patterns of tumor evolution. *PLoS computational biology* **10,** e1003665 (2014).

135. MIRONOV, A., KOONIN, E., ROYTBERG, M. & GELFAND, M. Computer analysis of transcription regulatory patterns in completely sequenced bacterial genomes. *Nucleic Acids Research* **27,** 2981–2989 (1999).

136. MISAWA, K. A codon substitution model that incorporates the effect of the GC contents, the gene density and the density of CpG islands of human chromosomes. *BMC genomics* **12,** 397 (2011).

137. MORRIS JR, J. & PRATT, V. *A linear pattern-matching algorithm* (1970).

138. MUGAL, C. F., ARNDT, P. F., HOLM, L. & ELLEGREN, H. Evolutionary consequences of DNA methylation on the GC content in vertebrate genomes. *G3: Genes, Genomes, Genetics* **5,** 441–447 (2015).

139. MULLER, E. *et al.* OutLyzer: software for extracting low-allele-frequency tumor mutations from sequencing background noise in clinical practice. *Oncotarget* **7,** 79485 (2016).

140. MYERS, G. A fast bit-vector algorithm for approximate string matching based on dynamic programming. *Journal of the ACM (JACM)* **46,** 395–415 (1999).

141. NAKAMURA, K. *et al.* Sequence-specific error profile of Illumina sequencers. *Nucleic acids research* **39,** e90–e90 (2011).

142. NAVARRO, G. A guided tour to approximate string matching. *ACM computing surveys (CSUR)* **33,** 31–88 (2001).

143. NAVARRO, G. NR-grep: a fast and flexible pattern-matching tool. *Software: Practice and Experience* **31,** 1265–1312 (2001).

144. NEUWALD, A. F., LIU, J. S., LIPMAN, D. J. & LAWRENCE, C. E. Extracting protein alignment models from the sequence database. *Nucleic Acids Research* **25,** 1665–1677 (1997).

145. NG, H.-H. & SURANI, M. A. The transcriptional and signalling networks of pluripotency. *Nature cell biology* **13,** 490 (2011).

146. NIK-ZAINAL, S. *et al.* The life history of 21 breast cancers. *Cell* **149,** 994–1007 (2012).

147. NISHIDA, K., FRITH, M. C. & NAKAI, K. Pseudocounts for transcription factor binding sites. *Nucleic acids research* **37,** 939–944 (2008).

148. ORPHANIDES, G., LAGRANGE, T. & REINBERG, D. The general transcription factors of RNA polymerase II. *Genes & development* **10,** 2657–2683 (1996).

149. PARK, P. J. ChIP–seq: advantages and challenges of a maturing technology. *Nature Reviews Genetics* **10,** 669 (2009).

150. PAVESI, G., MAURI, G. & PESOLE, G. An algorithm for finding signals of unknown length in DNA sequences. *Bioinformatics* **17 Suppl 1:S207-14** (2001).

151. PAVESI, G., MEREGHETTI, P., MAURI, G. & PESOLE, G. Weeder Web: discovery of transcription factor binding sites in a set of sequences from co-regulated genes. *Nucleic acids research* **32,** W199–W203 (2004).

152. PEARSON, C. E., EDAMURA, K. N. & CLEARY, J. D. Repeat instability: mechanisms of dynamic mutations. *Nature Reviews Genetics* **6,** 729 (2005).

153. PEVZNER, P. A., SZE, S.-H. *et al. Combinatorial approaches to finding subtle signals in DNA sequences.* in *ISMB* **8** (2000), 269–278.

155. PRAY, L. DNA replication and causes of mutation. *Nature education* **1,** 214 (2008).

156. QIN, W. *et al.* Analysis of TSC cortical tubers by deep sequencing of TSC1, TSC2 and KRAS demonstrates that small second-hit mutations in these genes are rare events. *Brain pathology* **20,** 1096–1105 (2010).

157. QIN, W. *et al.* Ultra deep sequencing detects a low rate of mosaic mutations in tuberous sclerosis complex. *Human genetics* **127,** 573–582 (2010).

158. QUAIL, M. A. *et al.* A tale of three next generation sequencing platforms: comparison of Ion Torrent, Pacific Biosciences and Illumina MiSeq sequencers. *BMC genomics* **13,** 341 (2012).

159. QUEEN, C., WEGMAN, M. N. & KORN, L. J. Improvements to a program for DNA analysin: a procedure to find homologies among many sequences. *Nucleic acids research* **10,** 449–456 (1982).

161. REDON, R. *et al.* Global variation in copy number in the human genome. *nature* **444,** 444 (2006).

162. ROMBAUTS, S. *et al.* Computational approaches to identify promoters and cis-regulatory elements in plant genomes. *Plant physiology* **132,** 1162–1176 (2003).

163. SAAD, C. *et al.* DiNAMO: highly sensitive DNA motif discovery in high-throughput sequencing data. *BMC bioinformatics* **19,** 223 (2018).

164. SAGOT, M.-F. & VIARI, A. *A double combinatorial approach to discovering patterns in biological sequences* in *Annual Symposium on Combinatorial Pattern Matching* (1996), 186–208.

166. SANDMANN, S. *et al.* Evaluating variant calling tools for non-matched next-generation sequencing data. *Scientific reports* **7,** 43169 (2017).

167. SANDVE, G. K. F. *Potentials and limitations of motif-based binding site prediction in DNA* thèse de doct. (Norwegian University of Science et Technology, 2008).

168. SANDVE, G. K. & DRABLØS, F. A survey of motif discovery methods in an integrated framework. *Biology direct* **1,** 11 (2006).

169. SANGER, F., NICKLEN, S. & COULSON, A. R. DNA sequencing with chain-terminating inhibitors. *Proceedings of the national academy of sciences* **74,** 5463–5467 (1977).

170. SCHBATH, S. Compound Poisson approximation of word counts in DNA sequences. *ESAIM: probability and statistics* **1,** 1–16 (1997).

171. SCHIRMER, M. *et al.* Insight into biases and sequencing errors for amplicon sequencing with the Illumina MiSeq platform. *Nucleic acids research* **43,** e37–e37 (2015).

172. SCHNEIDER, T. D. & STEPHENS, R. Sequence logos: a new way to display consensus sequences. *Nucleic Acids Research* **18,** 6097–6100 (1990).

174. SEARLS, D. B. Linguistic approaches to biological sequences. *Bioinformatics* **13,** 333–344 (1997).

175. SEARLS, D. B. String variable grammar: A logic grammar formalism for the biological language of DNA. *The Journal of Logic Programming* **24,** 73–102 (1995).

176. SEARLS, D. B. The computational linguistics of biological sequences. *Artificial intelligence and molecular biology* **2,** 47–120 (1993).

177. SEARLS, D. B. The linguistics of DNA. *American Scientist* **80,** 579–591 (1992).

178. SEIDL, H. *et al.* Ultraviolet exposure as the main initiator of p53 mutations in basal cell carcinomas from psoralen and ultraviolet A-treated patients with psoriasis. *Journal of Investigative Dermatology* **117,** 365–370 (2001).

179. SELLERS, P. H. The theory and computation of evolutionary distances: pattern recognition. *Journal of algorithms* **1,** 359–373 (1980).

180. SHANAHAN, K. L. A systematic error in mass flow calorimetry demonstrated. *Thermochimica acta* **387,** 95–100 (2002).

181. SHERRY, S. T. *et al.* dbSNP: the NCBI database of genetic variation. *Nucleic acids research* **29,** 308–311 (2001).

182. SHI, J. *et al.* AMD, an automated motif discovery tool using stepwise refinement of gapped consensuses. *PloS one* **6,** e24576 (2011).

183. SHIN, S. & PARK, J. Characterization of sequence-specific errors in various next-generation sequencing systems. *Molecular BioSystems* **12,** 914–922 (2016).

184. SHIRAISHI, Y. *et al.* An empirical Bayesian framework for somatic mutation detection from cancer genome sequencing data. *Nucleic acids research* **41,** e89–e89 (2013).

185. SINHA, S. On counting position weight matrix matches in a sequence, with application to discriminative motif finding. *Bioinformatics* **22,** e454–e463 (2006).

186. SINHA, S. & TOMPA, M. *A statistical method for finding transcription factor binding sites.* in *ISMB* **8** (2000), 344–354.

187. SINHA, S. & TOMPA, M. YMF: a program for discovery of novel transcription factor binding sites by statistical overrepresentation. *Nucleic acids research* **31,** 3586–3588 (2003).

188. STORMO, G. D., SCHNEIDER, T. D., GOLD, L. & EHRENFEUCHT, A. Use of the 'Perceptron' algorithm to distinguish translational initiation sites in E. coli. *Nucleic acids research* **10,** 2997–3011 (1982).

189. SUDMANT, P. H. *et al.* An integrated map of structural variation in 2,504 human genomes. *Nature* **526,** 75 (2015).

190. TALLACK, M. R. *et al.* A global role for KLF1 in erythropoiesis revealed by ChIP-seq in primary erythroid cells. *Genome research* **20,** 1052–1063 (2010).

191. TAN, A., ABECASIS, G. R. & KANG, H. M. Unified representation of genetic variants. *Bioinformatics* **31,** 2202–2204 (2015).

192. TATTINI, L., D'AURIZIO, R. & MAGI, A. Detection of genomic structural variants from next-generation sequencing data. *Frontiers in bioengineering and biotechnology* **3,** 92 (2015).

193. THIJS, G. *et al.* A Gibbs sampling method to detect overrepresented motifs in the upstream regions of coexpressed genes. *Journal of Computational Biology* **9,** 447–464 (2002).

194. THOMAS, J. A. & COVER, T. M. *Elements of information theory* (John Wiley & Sons, 2006).

195. THOMAS-CHOLLIER, M. *et al.* RSAT peak-motifs: motif analysis in full-size ChIP-seq datasets. *Nucleic acids research* **40,** e31–e31 (2011).

196. THORVALDSDÓTTIR, H., ROBINSON, J. T. & MESIROV, J. P. Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. *Briefings in bioinformatics* **14,** 178–192 (2013).

197. TOMLINSON, I. P., NOVELLI, M. & BODMER, W. The mutation rate and cancer. *Proceedings of the National Academy of Sciences* **93,** 14800–14803 (1996).

198. TOMPA, M. *An exact method for finding short motifs in sequences, with application to the ribosome binding site problem.* in *ISMB* **99** (1999), 262–271.

199. TOMPA, M. *et al.* Assessing computational tools for the discovery of transcription factor binding sites. *Nature biotechnology* **23,** 137–144 (2005).

200. Van HELDEN, J., ANDRÉ, B. & COLLADO-VIDES, J. Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies. *Journal of molecular biology* **281,** 827–842 (1998).

201. VANET, A., MARSAN, L. & SAGOT, M.-F. Promoter sequences and algorithmical methods for identifying them. *Research in Microbiology* **150,** 779–799 (1999).

202. WALL, J. D. *et al.* Estimating genotype error rates from high-coverage next-generation sequence data. *Genome research* **24,** 1734–1739 (2014).

203. WATSON, J. D., CRICK, F. H. *et al.* Molecular structure of nucleic acids. *Nature* **171,** 737–738 (1953).

204. WEINER, P. *Linear pattern matching algorithms* in *Switching and Automata Theory, 1973. SWAT'08. IEEE Conference Record of 14th Annual Symposium on* (1973), 1–11.

205. WILEY, G. B., KELLY, J. A. & GAFFNEY, P. M. Use of next-generation DNA sequencing to analyze genetic variants in rheumatic disease. *Arthritis research & therapy* **16,** 490 (2014).

206. WILM, A. *et al.* LoFreq: a sequence-quality aware, ultra-sensitive variant caller for uncovering cell-population heterogeneity from high-throughput sequencing datasets. *Nucleic acids research* **40,** 11189–11201 (2012).

207. XIE, X. *et al.* Systematic discovery of regulatory motifs in human promoters and 3' UTRs by comparison of several mammals. *Nature* **434,** 338 (2005).

208. XU, C. A review of somatic single nucleotide variant calling algorithms for next-generation sequencing data. *Computational and structural biotechnology journal* (2018).

209. YANG, X., CHOCKALINGAM, S. P. & ALURU, S. A survey of error-correction methods for next-generation sequencing. *Briefings in bioinformatics* **14,** 56–66 (2012).

210. YING, H. & HUTTLEY, G. Exploiting CpG hypermutability to identify phenotypically significant variation within human protein-coding genes. *Genome biology and evolution* **3,** 938–949 (2011).

211. YOHE, S. & THYAGARAJAN, B. Review of clinical next-generation sequencing. *Archives of pathology & laboratory medicine* **141,** 1544–1557 (2017).

212. ZABELL, S. L. On Student's 1908 Article "The Probable Error of a Mean". *Journal of the American Statistical Association* **103,** 1–7 (2008).

213. ZHAO, Z. & ZHANG, F. Sequence context analysis of 8.2 million single nucleotide polymorphisms in the human genome. *Gene* **366,** 316–324 (2006).

214. ZOOK, J. M., SAMAROV, D., MCDANIEL, J., SEN, S. K. & SALIT, M. Synthetic spike-in standards improve run-specific systematic error analysis for DNA and RNA sequencing. *PloS one* **7,** e41356 (2012).

215. ZOOK, J. M. *et al.* Extensive sequencing of seven human genomes to characterize benchmark reference materials. *Scientific data* **3,** 160025 (2016).

216. ZOOK, J. M. *et al.* Integrating human sequence data sets provides a resource of benchmark SNP and indel genotype calls. *Nature biotechnology* **32,** 246 (2014).

Références web pour les figures

1. https://opentextbc.ca/biology/chapter/9-1-the-structure-of-dna (2018).
2. https://fr.wikipedia.org/wiki/Code_g%C3%A9n%C3%A9tique (2018).
3. https://www.smileisbac.com/terminale/annale-genetique-et-evolution-polynesie-biologie-s-225 (2018).
4. https://commons.wikimedia.org/wiki/File:Point_CAA.png (2018).
5. https://nanoporetech.com/resource-centre/using-long-nanopore-reads-delineate-structural-variants-svs-human-genome (2018).
6. https://www.pathwayz.org/Tree/Plain/GAMETIC+VS.+SOMATIC+MUTATIONS (2018).
7. https://www.khanacademy.org/science/high-school-biology/hs-molecular-genetics/hs-biotechnology/a/dna-sequencing (2018).
8. https://www.atdbio.com/content/58/Next-generation-sequencing (2018).
9. http://jaspar.genereg.net/matrix/MA0035.2 (2018).
10. https://software.broadinstitute.org/gatk/documentation/article?id=11081 (2018).
11. ftp://ftp-trace.ncbi.nlm.nih.gov/giab/ftp (2018).

13. A.F.A., S., R., H. & P., G. *RepeatMasker* http://repeatmasker.org.

133. Miles, A. *pysamstats* https://github.com/alimanfoo/pysamstats.

154. Popovitch, G. *sparsepp* Accessed: 2017-01-16. https://github.com/greg7mdp/sparsepp.

160. R Core Team. *R: A Language and Environment for Statistical Computing* R Foundation for Statistical Computing. https://www.R-project.org.

165. SAMTOOLS. *hts-specs: Specifications of SAM/BAM and related high-throughput sequencing file formats* https://samtools.github.io/hts-specs/VCFv4.2.pdf.

173. SCIENTIFIC, T. F. *Torrent Suite* https://www.thermofisher.com/fr/fr/home/life-science/sequencing/next-generation-sequencing/ion-torrent-next-generation-sequencing-workflow/ion-torrent-next-generation-sequencing-data-analysis-workflow/ion-torrent-suite-software.html.