UNIVERSITÉ PARIS VII

THÈSE

présentée par

Pierre Nicodème

pour obtenir le titre de

Docteur en Informatique
de l'Université de Paris VII

Arrêtés ministériels du 5 juillet 1984
et du 30 mars 1992

Alignement avec des Familles de Séquences Protéiques

Soutenue le 4 septembre 1997 devant la commission d'examen

MM.	Maurice NIVAT	Président
	Maxime CROCHEMORE	Rapporteur
	Philippe DESSEN	Rapporteur
Mme.	Mireille REGNIER	
MM.	Daniel KAHN	
	Jean-Marc STEYAERT	

Thèse préparée au sein du Projet *Algo* de l'*Inria* en collaboration avec le *Lix*, Laboratoire d'Informatique de l'*École polytechnique*, et le Laboratoire de Biologie Moléculaire de l'*Inra* de Toulouse.

Remerciements

Maurice Nivat a accepté de présider mon jury de thèse et je ne pouvais trouver président plus prestigieux. Je le remercie d'avoir accepté cette responsabilité. Maxime Crochemore a été mon professeur lors de mon DEA, et j'ai pu apprécier la grande clarté de ses exposés. J'ai collaboré pendant mon stage de DEA avec Philippe Dessen qui m'a fait bénéficier de son expérience de la biologie. Tous deux ont accepté la lourde tâche d'être rapporteurs de ma thèse et je les en remercie vivement.

Les circonstances de la vie ont voulu qu'après un bref passage dans la recherche, je passe de nombreuses années à travailler dans l'industrie informatique. C'est à la confiance accordée par François Michel et Jean-Yves Caleca, qui ont encouragé un projet de collaboration entre la Société Copernique et l'INRIA que je dois d'avoir retrouvé le milieu de la recherche. Je leur en suis très reconnaissant.

Cette collaboration a été possible grâce à l'accueil qui m'a été fait au projet Algo de l'INRIA par Philippe Flajolet. Sans lui, il est certain que je n'aurais pas pu revenir au métier de chercheur. C'est grâce au temps qu'il m'a donné que j'ai pu acquérir les connaissances qui m'ont permis de participer pleinement aux activités du Projet Algo. Il va également sans dire que travailler dans son entourage est une source permanente de découvertes et d'enrichissement intellectuel.

En même temps qu'à Philippe Flajolet, c'est à tout les membres du projet Algo que je dois ma reconnaissance. J'ai bénéficié de la culture scientifique sans défaut de Bruno Salvy, des conseils très pédagogiques de Philippe Dumas avec qui j'ai partagé un bureau pendant plusieurs années, des compétences en bio-informatique de Mireille Régnier, de la qualité des enseignements de Philippe Robert, des encouragements de Xavier Gourdon et de François Morain, et de l'aide administrative de Virginie Collette. Frédéric Chyzak m'a aidé à surmonter les problèmes d'environnement et de défaillances matérielles avec beaucoup de patience et de gentillesse. Si j'ajoute que l'ambiance profondément chaleureuse du Projet ALGO m'a aidé à surmonter les moments de découragements que connaissent bien des thésards, c'est pour dire combien j'exprime à tous ma reconnaissance.

À l'INRIA, je dois aussi des remerciements à l'équipe de Jean-Jacques Codani, qui m'a aidé grâce à son expérience à effectuer des calculs lourds. Eric Glemet m'a permis de sortir de debugging délicats et Jean-Paul Comet m'a apporté d'utiles références bibliographiques.

Je dois à Jean-Marc Steyaert une grande reconnaissance. Outre ses responsabilités à l'École Polytechnique, il a su diriger cette thèse en me donnant, au sein du LIX, des conseils constructifs et opportuns qui m'ont permis d'avancer dans mon travail. Mes passages au LIX m'ont aussi permis de recueillir de précieux conseils de Fabrice Lefebvre. Evelyne Rayssac m'a apporté l'aide de son secrétariat. Je les remercie et je remercie également tous les membres du LIX qui m'ont aidé et que je n'ai pas nommés explicitement.

Cette thèse a été effectuée en collaboration soutenue avec l'INRA de Toulouse, et tout particulièrement avec Daniel Kahn, qui ajoute à ses grandes qualités de biologiste l'art de sentir quand des développements logiciels sont trop ambitieux, qualité nécessaire dans la recherche informatique. Cette collaboration fructueuse a aussi bénéficié des apports de Florence Corpet et Jérôme Gouzy, et j'en ai tiré de grands bénéfices. Je les en remercie. Je ne saurais oublier non plus de remercier Michel et Christiane Rouffet qui m'ont logé *avec la plus grande amitié lors de mes passages à Toulouse tout en me donnant leurs encouragements.*

Chapitre 1

Introduction

Les cinquante dernières années ont vu se dérouler le développement foudroyant de deux sciences, la biologie moléculaire et l'informatique. Elles ont vu aussi des progrès considérables des branches des mathématiques déjà existantes, et la création de nouvelles branches telles que le calcul formel, la démonstration automatique ou les théories de la complexité.

Au début des années 50, alors que la biologie cellulaire est déjà une science ancienne, la biologie moléculaire au contraire en est encore à ses débuts. En 1953, Watson et Crick découvrent la structure en double hélice de l'ADN. Cette découverte fondamentale permet de comprendre le mode de réplication de l'ADN lors de la reproduction des cellules. Ce succès permet en particulier de comprendre des fonctions génétiques en termes moléculaires. Cette même année, Sanger détermine la séquence en acides aminés de l'insuline. Cette découverte montre pour la première fois qu'une protéine a une structure précisément définie en acides aminés.

La découverte de Watson et Crick, qui est une des découvertes majeures du XX^e^ siècle, et celle de Sanger marquent le vrai début de la biologie moléculaire. Après ces découvertes, d'autres vont suivre à un rythme accéléré. En 1958, Crick postule l'existence des ARN de transfert. Jacob et Monod formulent en 1961 le concept d'ARN messager, autre maillon essentiel à la synthèse des protéines. En 1961 également, Crick et Brenner établissent les caractéristiques principales du code génétique, où à chaque triplet de bases de l'ADN est associé un acide aminé. En 1966, après une étude des ARN de transfert et des ribosomes, qui sont de gros complexes parcourant l'ARN lors de la traduction, Khorana déchiffre complètement le code génétique. En 1977, plusieurs chercheurs découvrent que plusieurs gènes sont discontinus, formés d'exons, qui sont des parties de gènes effectivement codantes, et d'introns, qui ne sont pas exprimés dans la traduction. Ces introns sont excisés lors de la transcription de l'ARN par le mécanisme de l'épissage. La découverte ultérieure de l'épissage alternatif montre que, suivant le type de cellule dans laquelle a lieu la transcription, la machinerie moléculaire modifie l'ordre dans laquelle sont générés des fragments de protéines, donnant à la nature la capacité de produire par combinatoire un nombre immense de protéines différentes. Parallèlement à ces découvertes mettant en jeu la compréhension de la machinerie moléculaire, les techniques de manipulations moléculaires se développent. On peut citer en particulier la technique de recombinaison d'ADN, qui date du début des années 1970. Cette technique a fourni des moyens puissants pour analyser ou modifier des gènes ou des protéines. La présence dans l'ADN de nombreux sites de restrictions permet de couper l'ADN et de le traiter en fragments que l'on peut déplacer d'une molécule d'ADN à l'autre. Cette technique permet, entre autres, d'effectuer le séquençage de gènes. Les méthodes de clonage, jointes à ces méthodes de recombinaison, permettent de produire artificiellement l'expression de gènes, ce qui a conduit au développement du génie génétique.

L'histoire des mathématiques remonte à l'antiquité, mais on a pu dire que le nombre de mathématiciens vivant aujourd'hui est supérieur à celui de tous les mathématiciens morts à ce jour. C'est dire combien cette science a progressé au cours des dernières années. La démonstration toute récente du théorème de Fermat par Wiles a bien sûr enthousiasmé les esprits. Elle fait intervenir des branches des mathématiques qui n'ont rien à voir avec la théorie classique des nombres dont ce théorème fait partie, et qui ont fait récemment l'objet d'avancées considérables. Ce résultat phare ne doit pas cacher d'autres résultats moins connus du grand public qui sont aussi d'une grande importance. En 1963, Cohen démontra que l'hypothèse du continu de Cantor ne pouvait se déduire des axiomes de Zermelo et Fraenkel. Ce résultat, combiné avec le fameux théorème d'incomplétude de Gödel, prouvait que l'hypothèse du continu était indécidable dans ce système d'axiomes. Le résultat de Cohen allait servir de base à la théorie de la calculabilité qui est l'un des fondements de l'informatique théorique moderne. En 1967, la théorie des fractales reçoit un apport décisif avec le travail de Mandelbrot, et cette théorie est appliquée aujourd'hui dans des domaines aussi variés que la compression d'images ou la modélisation du système alvéolaire des poumons. En 1967, Matjasevitch répond par la négative au 10ème problème posé par Hilbert lors du congrés extraordinaire des mathématiciens en 1900; il n'existe pas d'algorithme permettant de résoudre Matjoute la classe des équations Diophantiennes, ou équations polynomiales en nombres entiers. Ce résultat achève de ruiner le rêve de Hilbert d'axiomatiser toutes les mathématiques, rêve déjà bien ébranlé par Gödel. À la fin des années 60, Knuth effectue des applications remarquables des probabilités à l'algorithmique, montrant que le comportement en moyenne des algorithmes est en général considérablement meilleur que le comportement dans le cas le pire. Des algorithmes comme la transformée de Fourier rapide, développée également dans les années 60, permettent de gagner des ordres de grandeur dans la vitesse de nombreux calculs. Dans les mêmes années, la théorie de la calculabilité commence à s'édifier, qui va permettre de classifier les problèmes selon leur difficulté, mettant en particulier en évidence la classe des problèmes dits NP-complets, pour lesquels il est nécessaire de rechercher des solutions approchées. En 1976, le théorème des quatre couleurs est démontré. Toute carte planaire peut être coloriée, en attribuant une couleur à chaque face, ou pays, de façon à ce que deux pays ayant une frontière commune n'aient jamais la même couleur. La grande nouveauté de ce théorème tient dans le fait que sa démonstration, après plus de cent ans d'efforts "à la main", est démontrée par ordinateur, la démonstration nécessitant 1200 heures de calcul. Depuis cette période, des disciplines charnières entre les mathématiques et l'informatique ont permis de démontrer automatiquement un grand nombre de théorèmes, dans des domaines tels que l'asymptotique ou la combinatoire.

L'informatique se développe également très rapidement au cours des cinquante dernières années, l'informatique théorique étant une branche des sciences mathématiques, tandis que l'on considère plutôt comme une technique l'informatique dite dure, objet d'applications à des systèmes matériels et logiciels dans la puissante industrie informatique. Parallèlement à des progrès considérables dans les théories des langages et des compilateurs, les années 70 voient les débuts du système d'exploitation d'ordinateur UNIX et du langage C, langage de programmation souple et puissant. La puissance de calcul des ordinateurs augmente régulièrement, rendant les stations de travail d'une génération aussi performantes que les grands calculateurs de la génération précédente. Alors que la quantité des données stockées sur ordinateur croît exponentiellement, les progrès des bases de données et des algorithmes d'indexation permettent de structurer ces données et de les accéder rapidement. La maîtrise des réseaux est acquise au cours des années 80, donnant naissance à l'extraordinaire outil de communication des connaissances qu'est Internet.

La bioinformatique, de son côté, accompagne le développement de la biologie, et bénéficie des progrès des mathématiques et de l'informatique. La frontière entre biomathématique et bioinformatique est parfois difficile à cerner. Dans ce contexte multidisciplinaire, on peut citer comme travail pionnier le travail de probabilité effectué par Hardy en 1908 pour analyser le modèle génétique de Mendel. En 1970, les premiers résultats de séquençage d'ADN et de protéines sont stockés sur ordinateur, et apparaît le premier algorithme de comparaison de séquences, dû à Needleman et Wunsch. Cet algorithme sera par la suite amélioré par Smith et Waterman. En 1983, la recherche d'homologies par ordinateur, liée à la comparaison de séquences, connaît un premier grand succès quand Doolittle montre, par recherche dans une base de séquences, qu'une protéine codée par un oncogène nommé v-*sis*, (un oncogène est un gène muté cause de cancer), est similaire à un facteur de croissance nommé PDGF. Ceci met en évidence, pour la première fois, la fonction physiologique d'un oncogène. Parmi les autres apports de la bioinformatique, au cours des dernières années, on peut citer les méthodes prédictives de repliement des ARN, les méthodes de détection d'introns-exons, la classification phylogénétique automatique, la cartographie du génome humain et la prédiction de structures en trois dimensions de protéines. Les grandes bases de données biologiques, comme la base de gènes GENBANK, la base de protéines SWISSPROT, la base de structures protéiques tridimensionnelles PDB et la base de domaines protéiques PRODOM constituent également des outils d'intérêt considérables pour les biologistes. Ces bases de données et ces outils, dont nous n'avons pas dressé une liste exhaustive, combinés avec la puissance d'utilisation temps réel d'Internet constituent aujourd'hui une partie essentielle du cadre de travail du biologiste.

Nous avons décrit dans ce paragraphe quelques unes des découvertes les plus marquantes du demi-siècle qui se termine dans les domaines de la biologie, des mathématiques et de l'informatique, en étant bien loin d'être exhaustif. Nous décrivons dans le prochain les caractéristiques principales des protéines. Nous décrivons enfin dans le dernier paragraphe de ce chapitre l'apport de notre thèse au travail de recherche effectué dans les applications des mathématiques et de l'informatique à la biologie, ainsi que le plan de cette thèse.

1.1 Caractéristiques principales des protéines

Nous décrivons dans ce paragraphe les caractéristiques et fonctions principales des protéines, qui sont des objets d'intérêt majeur pour les biologistes. Nous mentionnerons en particulier.

1. *La catalyse enzymatique.* Dans leur quasi totalité, les réactions chimiques dans les systèmes biologiques sont catalysées par des macromolécules spécifiques appelées enzymes. Certaines de ces réactions, comme l'hydratation du dyoxide de carbone, sont assez simples. D'autres, comme la réplication d'un chromosome entier sont très compliquées. Les enzymes font preuve d'énormes capacités catalytiques. Elles augmentent souvent les vitesses de réaction par des facteurs qui dépassent le million. Il est remarquable que presque toutes les enzymes connues soient des protéines, ce qui donne aux protéines un rôle unique dans le processus des transformations chimiques dans les systèmes biologiques.

2. *Transport et stockage.* De nombreuses petites molécules et des ions sont transportés par des protéines spécifiques. Par exemple, l'hémoglobine transporte l'oxygène dans les érythrocytes, ou globules rouges, tandis que la myoglobine, une protéine apparentée, stocke l'oxygène dans les muscles. Le fer, qui est un constituant essentiel de l'hémoglobine et de la myoglobine, est transporté dans le plasma du sang par la transferrine, et est stocké dans le foie comme un complexe avec la ferritine, une autre protéine.

3. *La coordination des mouvements.* Les protéines forment le composant majeur des muscles. La contraction musculaire est accomplie par le glissement de deux sortes de filaments protéiques.

4. *Le support mécanique.* La haute force de tension de la peau et des os est due à la présence de collagène, une protéine fibreuse.

5. *La protection immunitaire.* Les anticorps sont des protéines hautement spécifiques qui reconnaissent et se combinent avec des substances étrangères telles que les virus, les bactéries, et les cellules d'autres organismes.

6. *La génération et la transmission d'impulsions nerveuses.* Les réponses des cellules nerveuses à des stimuli spécifiques sont effectuées par l'intermédiaire de protéines réceptrices.

7. *Le contrôle de la croissance et de la différenciation.* L'expression séquentielle et contrôlée de l'information génétique est essentielle pour la croissance ordonnée et la différenciation des cellules. Seule une petite fraction du génome d'une cellule est exprimée à un moment donné. Dans les bactéries, des protéines répresseurs sont d'importants éléments de contrôle qui inactivent l'expression de segments spécifiques de l'ADN d'une cellule. Dans les organismes supérieurs, la croissance et la différenciation sont contrôlés par des protéines qui régulent la croissance. Par exemple, la protéine facteur de croissance des nerfs guide la croissance du système nerveux.

Les protéines sont construites sur un répertoire de vingt acides aminés. Elles sont constituées d'un squelette carboné auquel viennent se rattacher des chaînes latérales, qui varient en taille, forme, charge, capacité de liaison hydrogène et réactivité chimique. Une classification utile des acides aminés est établie selon qu'ils sont habituellement présents *à l'intérieur* d'une protéine repliée, *à sa surface*, ou dans les deux positions.

- Les acides aminés à groupements hydrophobes sont la Valine, la Leucine, l'Isoleucine, la Phénylalanine et la Méthionine.
- Les acides aminés à groupement hydrophiles sont l'Acide aspartique, L'Acide glutamique, l'Asparagine, la Glutamine, la Lysine, l'Arginine et l'Histidine.
- Les acides aminés présents à la surface et à l'intérieur des protéines sont la Glycine, l'Alanine, la Cystéine, la Sérine, la Thréonine, la Tyrosine, la Proline et le Tryptophane.

D'autres caractéristiques importantes pour les acides aminés sont la présence de groupements aromatiques dans la Phénylalanine, la Tyrosine et le Tryptophane, et la présence de soufre dans la Cystéine et la Méthionine. L'Histidine peut ne pas être chargée, ou être chargée positivement suivant l'environnement local.

On distingue quatre niveaux de structures principaux dans l'architecture des protéines.

- La *structure primaire* donne la séquence linéaire des acides aminés et l'emplacement des disulfides, s'il y en a.
- La *structure secondaire* se rapporte à la conformation spatiale des résidus d'acides aminés qui sont proches les uns des autres dans la séquence linéaire, comme le repliement de la chaîne. On trouve comme éléments des structures secondaires les hélices α, les feuillets β et la triple hélice de type collagène.
- La *structure tertiaire* se rapporte à des conformations spatiales de résidus d'acides aminés qui sont éloignés dans la séquence linéaire.
- La *structure quaternaire* concerne des protéines formées de plus d'une chaîne polypeptidique. Chaque chaîne polypeptidique est appelée une sous-unité. La structure quaternaire se rapporte à la conformation spatiale de ces sous-unités et à la nature de leurs contacts. L'hémoglobine est un exemple typique de protéine ayant une structure quaternaire formée de quatre sous-unités.

Les études sur la conformation, la fonction et l'évolution des protéines ont révélé l'importance de deux niveaux additionnels de structure secondaire. Les *structures super-secondaires* se rapportent à des groupements de structures secondaires. D'autre part, de nombreuses chaînes polypeptidiques se replient en deux ou plusieurs régions compactes qui sont reliées par un segment flexible de chaîne polypeptidique. Ces unités globulaires compactes sont appelées *domaines*, et leur taille varie de 60 à 400 résidus. Les cellules des espèces eucaryotes sont caractérisées, à la différence de celles des procaryotes, par la présence d'un noyau. Un principe important a été dégagé de l'analyse des gènes et des protéines chez les eucaryotes: les domaines de protéines y sont fréquemment encodées par des parties distinctes de gènes, les exons. La base ProDom constitue un exemple de base de domaines de protéines.

Note bibliographique. Parmi les ouvrages introduisant à la biologie, nous mentionnons l'ouvrage de biologie générale de Purves et *al.*, "Le monde du vivant" [68], le tiré à part des premiers chapitres du livre de biochimie de Stryer, "Molecular design of life" [80], et l'ouvrage de Watson et *al.* décrivant les principales techniques de manipulation de l'ADN, "Recombinant DNA" [87].

1.2 Objectif et plan de cet ouvrage

Une grande partie de notre travail trouvera son application dans la recherche de similarités entre séquences protéiques. Nous étudierons en particulier la recherche de similarités entre des séquences de protéines et des alignements multiples de protéines, tels que ceux qui sont contenus dans ProDom. Le domaine théorique de l'alignement de séquences a été partiellement exploré, et des modèles simplifiés comme le modèle d'alignement sans trous sont bien décrits mathématiquement. Nous présenterons ce modèle mathématique qui s'appuie sur des résultats de marche aléatoire. Nous présenterons ensuite le logiciel d'alignement Blast et un nouvel algorithme qui permet d'étendre ce logiciel à l'alignement de multi-alignement et de séquences. Nous montrerons ensuite comment appliquer la méthode des applications Poissonniennes à ce dernier algorithme. Nous traiterons enfin dans un dernier chapitre un sujet différent, le problème du multiplexage de la réaction de polymérisation en chaîne, qui trouve, entre autres, des applications dans le domaine du diagnostic médical.

Le plan de notre thèse s'énonce comme suit.

- Chapitre 1. *Introduction.*
- Chapitre 2. *Le modèle de Karlin.*
  Ce modèle forme la description mathématiques de l'alignement de séquences sans trous.
- Chapitre 3. *Compléments mathématiques.*
  Ce chapitre donne des compléments mathématiques concernant les marches aléatoires et le modèle de Karlin.
- Chapitre 4. *Le logiciel d'alignement local BLAST.*
  Nous décrivons les principes sur lesquels est construit ce logiciel.
- Chapitre 5. *BlastMultAl: une extension de Blast pour la recherche de similarités avec des multi-alignements.*
  Nous développons une extension de Blast basée sur l'utilisation de graphes d'alignement.
- Chapitre 6. *Les approximations Poissonniennes.*
  Nous décrivons la méthode des approximations Poissonniennes qui permet de calibrer rapidement les multi-alignements de ProDom pour l'algorithme BlastMultAl.

- Chapitre 7. *Le multiplexage de PCR.*
  Nous donnons un modèle mathématique pour le multiplexage de PCR, étudions sa complexité, et proposons un algorithme.

Les chapitres sont dans une large mesure indépendants les uns des autres, à l'exception des chapitres 2 et 3.

Chapitre 2

Le modèle mathématique de Karlin pour l'alignement de séquences sans gaps

2.1 Introduction

Ce chapitre reprend un ensemble de preuves techniques dispersées dans la littérature et en donne une présentation unifiée. Il donne par ailleurs une présentation plus moderne et plus concise de la démonstration dans le cas de variables discrètes qui est celui des applications biologiques (démonstration originale de la note 12). Les preuves des théorèmes 9 et 19 sont originales.

Nous allons considérer le modèle mathématique sous-jacent à l'alignement local de deux séquences, dans le cas où les alignements $1-1$ considérés sont sans trous, et où l'on utilise un schéma de scores additif. Étant donné deux séquences $s_1 \dots s_{n_1}$ et $t_1 \dots t_{n_2}$ sur un même alphabet Σ, un alignement local $\mathcal{A}$ de longueur l est une suite de paires $\{(s_{j_1}, t_{j_2}), (s_{j_1+1}, t_{j_2+1}), \dots, (s_{j_1+l}, t_{j_2+l})\}$, avec $1 \leq j_1 \leq n_1$, $1 \leq j_2 \leq n_2$, $j_1 + l \leq n_1$, et $j_2 + l \leq n_2$. Étant donné d'autre part une matrice $\mathcal{M}$ associant à toute paire $\{a_i, a_j\} \in \Sigma \times \Sigma$ un score, le score de l'alignement est égal à la somme du score des paires qui le constituent. Dans les cas d'alignement de séquences biologiques qui nous intéressent, et plus particulièrement de séquences protéiques, nous considérerons des matrices de similarité entre acides aminés qui associent des scores positifs à des paires d'acides aminés de caractéristiques physico-chimiques voisines, et des scores négatifs à celles qui ont des caractéristiques différentes. Nous nous intéresserons alors aux alignements de scores élevés et tout particulièrement à l'alignement optimal, qui est l'alignement de meilleur score. En négligeant les effets de bord, l'espace de recherche a comme dimension le produit $n_1 \times n_2$ de la taille des deux séquences.

En considérant d'autre part une distribution de probabilité sur l'alphabet Σ, et en considérant que les tirages successifs de lettres sont indépendants, nous en déduisons immédiatement une distribution de probabilité sur l'alphabet $\Sigma \times \Sigma$ sur lequel nous avons défini les scores. Nous avons ainsi défini une variable aléatoire X de $\Sigma \times \Sigma$ sur $\mathbb{R}$, avec

$$X(a_1, a_2) = \mathcal{M}(a_1, a_2), \quad \{(a_1, a_2) \in \Sigma \times \Sigma\},$$

Nous sommes donc amenés à étudier le problème général suivant.
Énoncé probabiliste associé à une recherche d'alignement optimal. Soit X_i une suite de variables aléatoires i.i.d, de distribution X. Soit S_k les sommes partielles, $S_k = \sum_{i=1}^{k} X_i$. On cherche à connaître la distribution de $M(n) = \max_{1 \leq j \leq k \leq n}(S_k - S_j)$.

La variable $M(n)$ correspond au score de l'alignement optimal de deux séquences de longueur n_1 et n_2 telles que $n_1 \times n_2 = n$.

Nous démontrerons que

$$Pr\left(M(n) > \frac{log(n)}{\lambda} + x\right) = 1 - e^{-Ke^{-\lambda x}}. \tag{2.1}$$

où λ peut être calculé comme racine d'une équation, et K possède un développement en série de convergence exponentielle. Asymptotiquement, quand $n \to \infty$, cette recherche n'a de sens que si $E(X) < 0$; dans le cas contraire, $M(n) \to \infty$ avec probabilité 1 par la loi des grands nombres.

Nous allons considérer la marche aléatoire définie dans le demi-plan d'abscisses positives par la suite des points de coordonnées $\{i, S_i\}$, $i \in \mathbb{N}$, que nous noterons par la suite (S_i). Nous rechercherons alors la section $S_{[k,l]} = S_l - S_k$ de valeur maximale. La démonstration repose principalement sur des propriétés de la marche aléatoire (S_i). Elle utilise également des résultats de la théorie du renouvellement (renewal theory). La première démonstration de l'équation 2.1, représentative d'une loi des valeurs extrêmes, et des formules donnant les paramètres K et λ, dans le cas où X est une variable continue, est due à Iglehart (1972). Cette démonstration a été généralisée par Karlin et Dembo au cas discret (1992) et appliquée par Altschul au cas de la recherche de similarités entre séquences biologiques (protéines ou ADN) dans le logiciel d'alignement de séquences BLAST [3].

Nous ferons fréquemment référence aux articles de Iglehart [44], et de Karlin et Dembo [50], ainsi qu'aux livres de Chung [15] et Durrett [26] et au cours de DEA de Philippe Robert sur les Files d'Attente (Paris VI, Laboratoire de Probabilités, 1993-1994). (On trouvera également de nombreux résultats relatifs aux marches aléatoires dans Gut [38]). Nous avons approfondi certaines démonstrations, et nous avons donné la preuve d'un résultat concernant le paramètre K qui était annoncé par Karlin sans démonstration. On pourra aussi consulter le tutorial de Theilhaber [82] qui a une approche différente de celle que nous avons adoptée.

Nous donnons à présent le schéma général de la démonstration, que nous expliciterons de manière plus formelle au prochain paragraphe. (On pourra se reporter au tableau 2.1 pour les notations).

La marche aléatoire ($S_i = \sum_{j=1}^{i} X_j$) est décomposée en sections en utilisant des *points d'échelle* négatifs; ces points correspondent à des temps de record négatif successifs. La figure 2.1 donne un exemple de cette décomposition.

Si q_i est le ième point d'échelle, la marche aléatoire y a la valeur S_{q_i}. On définit récursivement le point d'échelle q_{i+1} par $\{q_{i+1} = \min(y;\ \forall x,\ q_i \leq x < y;\ S_y < S_x), q_0 = 0\}$.

Les sections de la marche aléatoire $S_{[q_i,q_{i+1}]}$, situées entre les temps d'échelle, sont alors des répliques probabilistes de la première section $S_{[q_0,q_1]}$, qui est la portion de la marche aléatoire partant de l'origine et s'arrêtant au temps d'arrêt ζ^- de première atteinte

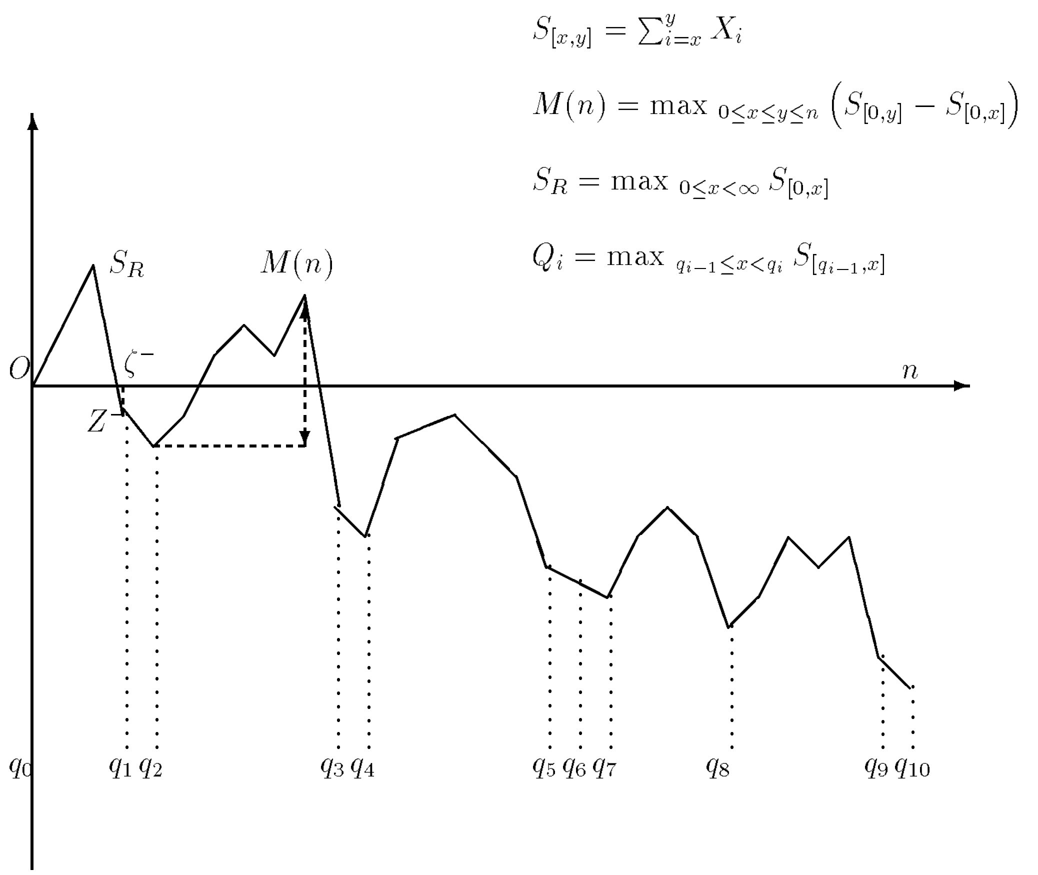


Figure 2.1: Décomposition de la marche aléatoire pour la démonstration de la formule de Karlin

$$\mathcal{L}(Z^+) \Longrightarrow \mathcal{Q}(S_R) \Longrightarrow \mathcal{Q}(Q_1) \Longrightarrow \mathcal{Q}(M(q_i)) \Longrightarrow \mathcal{Q}(M(n)),$$

avec $\mathcal{L}(X)$ et $\mathcal{Q}(X)$ respectivement loi et queue de la distribution de X

Figure 2.2: Schéma de la démonstration d'Iglehart-Karlin:

X	l'incrément de la marche aléatoire, avec $E(X) < 0$ et $P(X > 0) > 0$
$F(y)$	la distribution de X: $F(y) = P(X < y)$
λ	la solution en t de l'équation $E(e^{tX}) = 1$
S_R	la valeur record positive absolue de la marche aléatoire (S_i) $S_R = \max_{i\geq 1} S_{[0,i]}$
t_R	le temps minimum auquel le record absolu S_R est atteint $t_R = \{\min i; S_i = S_R\}$
$R(y)$	la distribution de S_R $R(y) = P(S_R < y)$
$M(n)$	le maximum obtenu par une section de la marche aléatoire dans l'intervalle $[1, n]$ $M(n) = \max S_{[k,j]} = \max\ _{1\leq j\leq k\leq n}(S_k - S_j)$.
ζ^+	le temps d'arrêt correspondant au premier passage de la marche aléatoire dans le demi-plan d'ordonnées strictement positives $\zeta^+ = \{\min i, S_i > 0\}$
Z^+	la valeur de la somme partielle pour le temps d'arrêt ζ^+ $Z^+ = S_{\zeta^+}$
ζ^-	le temps d'arrêt correspondant au premier passage de la marche aléatoire dans le demi-plan d'ordonnées négatives $\zeta^- = \{\min i, S_i \leq 0\}$
Z^-	la valeur de la somme partielle pour le temps d'arrêt ζ^- $Z^- = S_{\zeta^-}$
$L(y)$	la distribution de la variable Z^+ $L(y) = P(0 < Z^+ = S_{\zeta^+} < y)$
q_i	les temps d'arrêt correspondant aux records négatifs successifs (variables à échelle) $q_0 = 0$, $q_i = \{\min j, S_j < S_{q_{i-1}}\}$
Q_i	le maximum des sommes partielles commençant en q_{i-1} et finissant avant q_i: $Q_i = \{\max S_{[q_{i-1},k]},\ q_{i-1} \leq k < q_i\}$
$Q(y)$	la distribution de la variable Q_1 $Q(y) = P(0 \leq Q_1 < y)$

Tableau 2.1: Notations utilisées.

du demi-plan négatif. On remarquera que la section $S_{[x,y]}$ de valeur maximale commence certainement en l'un des points d'échelle q_i, et qu'elle est même le record Q_i pour la section commençant à ce point d'échelle. La variable $M(q_m)$ correspondra au maximum des variables Q_i, $i \in [1, m]$, des m premières sections, et la remarque précédente montre que tous les Q_i ont même distribution que Q_1.

L'identité de Spitzer [78] donne, pour une marche aléatoire correspondant à la somme d'une variable aléatoire X de moyenne strictement négative $E(X) < 0$, la loi des sommes $Z^+ = S_{\zeta^+}$ (resp. $Z^- = S_{\zeta^-}$) obtenus au temps d'arrêt ζ^+ (resp. ζ^-) du premier passage dans le demi-plan d'abscisse positive (resp. négative).

Partant de ces identités pour la loi de Z^+, en décomposant la marche aléatoire infinie en la partie précédant ζ^+, et le reste de la marche aléatoire, on reliera la queue de distribution du record absolu S_R de la marche aléatoire à la loi de Z^+.

En effectuant ensuite une décomposition "verticale" de la marche aléatoire, on trouvera une décomposition asymptotique faisant intervenir $q_1 = \zeta^-$, et on reliera la queue de distribution de Q_1 à celle de S_R.

2.1.1 Plan de la démonstration

1. *Calcul de $R(y)$. (Record absolu).* Des résultats de marche aléatoire (identité de Spitzer) donnent la transformée de Laplace de la distribution de Z^+. Cette identité est obtenu en utilisant la factorisation de Wiener-Hopf [15, 27, 69].

   On décompose la marche aléatoire infinie en:

   - une section s'arrêtant en ζ^+;
   - le reste de la marche aléatoire.

   Le maximum de cette deuxième section donne lieu à une convolution, et on obtient pour $R(y)$ une équation de renouvellement dépendant de $L(y)$.

   Le théorème du renouvellement nous donne la solution asymptotique de cette équation. On obtient
   $$1 - R(y) \to C_1 e^{-\lambda y}, \text{ quand } y \to \infty.$$

2. *Calcul de $Q(y)$. (Record de la première section).* On relie la distribution de Q_1 à celle de S_R. Ceci se fait en décomposant la marche aléatoire par rapport à un seuil y et en comparant Q_1 à ce seuil. On obtient
   $$Pr[S_R > y] = Pr[Q_1 \geq y] + Pr[S_{q_1} = \eta] Pr[S_{R[q_1 \ldots \infty]} > y - \eta, Q_1 < y],$$
   où $S_{R[q_1 \ldots \infty]}$ est le record absolu d'une marche aléatoire commençant en q_1. La deuxième partie de l'équation introduit une intégrale de convolution qu'on étudiera asymptotiquement, quand $y \to \infty$.

   On en déduit que
   $$1 - Q(y) \to C_2 e^{-\lambda y}.$$
   La distribution de la variable Z^- intervient dans le calcul de C_2.

3. *Calcul de la queue de distribution de $M(n)$.* On remarquera que $M(q_m)$, ou variation positive maximum sur $[1 \ldots q_m]$ est le maximum des $Q_j, j \in [1, m]$; en effet toute excursion de variation maximum commence nécessairement à un point d'échelle des records décroissants (voir figure 2.1).

   On a donc $Pr[M(q_m) < y] = Q^m(y)$. Ceci donne la distribution de $M(n)$ pour les valeurs de n égales aux q_i. L'application à la limite ($i \to \infty$) du théorème de Blackwell permet d'obtenir $M(n)$ pour tout n.

2.2 Identité de Spitzer et factorisation de Wiener-Hopf

Nous considérons la marche aléatoire associée aux sommes partielles $S_n = \sum_{i=1}^{n} X_n$ où les variables X_n sont indépendantes et distribuées comme une variable donnée X. Nous allons chercher à obtenir la loi de la variable $Z^- = S_{\zeta^-}$, où ζ^- est le temps d'arrêt d'atteinte du demi-plan négatif par la marche aléatoire.

Cette loi est obtenue grâce à une double décomposition de Wiener-Hopf du quasi-inverse de $E\left(e^{-\xi X}\right)$.

On peut trouver cette démonstration dans Karlin & Taylor [53] qui utilise les fonctions caractéristiques; nous suivrons cependant Neveu [61] et Robert [71] pour la démonstration de la décomposition de Wiener-Hopf que nous ferons au paragraphe 2.2.2, et Chung [15] pour son application au calcul de la loi de Z^-.

La factorisation de Wiener-Hopf associe à la quasi-inverse (marquée par une variable u, avec $|u| < 1$) de la transformée de Laplace de la distribution de X deux fonctions ϕ_+ et ϕ_-:

$$\frac{1}{1 - uE(e^{-\xi X})} = \phi_+(u, \xi)\phi_-(u, \xi); \tag{2.2}$$

les fonctions ϕ_+ et ϕ_- sont analytiques et respectivement bornées dans les demi-plans $\Re(\xi) \geq 0$ et $\Re(\xi) \leq 0$; cette factorisation est unique.

La double décomposition mentionnée précédemment est,

- d'une part un développement exp-log du quasi-inverse connu sous le nom de formule de Spitzer.
- et d'autre part une décomposition faisant intervenir les temps d'arrêt ζ^- et ζ^+,

Compte tenu de l'unicité des fonctions ϕ^+ et ϕ^-, cette double décomposition de Wiener-Hopf nous donnera deux identités; nous tirerons de l'une d'elles la loi de Z^-.

2.2.1 Décomposition de Wiener-Hopf en exp-log

Nous appliquons la décomposition de Wiener-Hopf à la marche aléatoire (S_n) en utilisant un développement exp-log du quasi-inverse:

$$\frac{1}{1-x} = \exp\left(\log\left\{\frac{1}{1-x}\right\}\right) = \exp\left(\sum_{i=1}^{\infty} \frac{x^i}{i}\right);$$

nous obtenons:

$$\frac{1}{1 - uE(e^{-\xi X})} = \phi_+\phi_- = \exp\left(\sum_{i=1}^{\infty} \frac{u^i}{i}\left[E(e^{-\xi X})\right]^i\right); \tag{2.3}$$

Les X_i étant indépendants, nous avons $E(e^{-\xi S_n}) = \left[E(e^{-\xi X})\right]^n$, et donc

$$\frac{1}{1-uE(e^{-\xi X})} = \phi_+\phi_- = \exp\left(\sum_{i=1}^{\infty}\frac{u^i}{i}E(e^{-\xi S_i})\right), \tag{2.4}$$

que nous pouvons réécrire:

$$\frac{1}{1-uE(e^{-\xi X})} = \prod_{i=1}^{\infty}\exp\left(\frac{u^i}{i}E\left(e^{-\xi S_i}\right)\right); \tag{2.5}$$

mais nous avons

$$E(e^{-\xi S_i}) = E(e^{-\xi S_i}; S_i > 0) + E(e^{-\xi S_i}; S_i \leq 0),$$

et donc

$$\frac{1}{1-uE(e^{-\xi X})} = \phi_+\phi_- = \prod_{i=1}^{\infty}\exp\left(\frac{u^i}{i}E\left(e^{-\xi S_i}; S_i > 0\right)\right)\prod_{i=1}^{\infty}\exp\left(\frac{u^i}{i}E\left(e^{-\xi S_i}; S_i \leq 0\right)\right). \tag{2.6}$$

Dans le premier produit infini, on a $\left|e^{-\xi S_n}\right| \leq 1$, puisque $S_n > 0$ et $\Re\xi \geq 0$; nous en déduisons

$$\left|\prod_{i=1}^{\infty}\exp\left(\frac{u^i}{i}E\left(e^{-\xi S_i}; S_i > 0\right)\right)\right| < \prod_{i=1}^{\infty}\exp\left(\frac{|u|^i}{i}\right) = \exp\left(\sum_{i=1}^{\infty}\frac{|u|^i}{i}\right) = \frac{1}{1-|u|},$$

et comme $|u| < 1$, ce produit infini est borné. De la même manière, le produit infini conditionné par $\{S_i \leq 0\}$ est borné pour $\Re\xi \leq 0$. La décomposition de Wiener-Hopf étant unique, on obtient

$$\phi_+ = \prod_{i=1}^{\infty}\exp\left(\frac{u^i}{i}E\left(e^{-\xi S_i}; S_i > 0\right)\right) \tag{2.7}$$

$$\phi_- = \prod_{i=1}^{\infty}\exp\left(\frac{u^i}{i}E\left(e^{-\xi S_i}; S_i \leq 0\right)\right) \tag{2.8}$$

2.2.2 Factorisation de Wiener-Hopf suivant les temps d'arrêt ζ^+ et ζ^-

Nous allons maintenant donner une deuxième décomposition de Wiener-Hopf et la preuve de l'unicité de la décomposition.

Nous avons besoin au préalable d'un lemme disant que la marche aléatoire prise après un temps d'arrêt est une réplique probabiliste de la marche aléatoire initiale. On trouve ce lemme dans Chung [15], Robert [71] et Durrett [26] (p.177).

Lemme 1. *Soient X_n une suite de variables aléatoires i.i.d. et $\mathcal{F}_n$ la tribu engendrée par $X_1, \ldots, X_n$. Soit α un temps d'arrêt $\Omega \to \{\mathbb{N} \bigcup \infty\}$, relativement à la filtration $(\mathcal{F}_n)_n \geq 0$; conditionnellement à l'événement $\{\alpha < \infty\}$, la suite $(S_{\alpha+n} - S_\alpha)$ est indépendante du couple (S_α, α), et a même loi que la marche aléatoire initiale (S_n).*

Preuve. Soient $p \in \mathbb{N}$ et f, g deux fonctions Boréliennes positives, on a

$$E\left(f(S_{n+\alpha} - S_\alpha) g(S_\alpha) 1_{\{\alpha=p\}}\right) = E\left(f(S_{n+p} - S_p) g(S_p) 1_{\{\alpha=p\}}\right).$$

En remarquant que $g(S_p)1_{\{\alpha=p\}}$ est $\mathcal{F}_n$ mesurable et que $f(S_{n+p} - S_p)$ ne dépend que des $(X_k)_{k\geq p+1}$, l'indépendance des (X_n) nous donne

$$\begin{aligned} E\left(f((S_{n+\alpha} - S_\alpha) g(S_\alpha) 1_{\{\alpha=p\}}\right) &= E\left(f(S_{n+p} - S_p)\right) E\left(g(S_p) 1_{\{\alpha=p\}}\right) \\ &= E\left(f(S_n)\right) E\left(g(S_\alpha) 1_{\{\alpha=p\}}\right), \end{aligned}$$

et donc $(S_{\alpha+n} - S_\alpha)$ a même loi que S_n et est indépendant du couple (S_α, α). □

Le théorème donnant la décomposition de Wiener-Hopf est alors:

Théorème 2. *Soit X une variable aléatoire. Pour tout u, tel que $|u| < 1$, il existe deux fonctions $\phi_+(u, .)$, $\phi_-(u, .)$ vérifiant*

1.

$$\frac{1}{1 - uE(e^{-\xi X})} = \phi_+(u, \xi)\phi_-(u, \xi). \tag{2.9}$$

2. La fonction $\phi_-(u, .)$ est bornée ainsi que son inverse dans le demi-plan $\{\Re\xi \leq 0\}$; la fonction $\phi_+(u, .)$ est bornée ainsi que son inverse dans le demi-plan $\{\Re\xi \geq 0\}$ et

$$\lim_{\Re\xi\to\infty} \phi_+(u, \xi) = 1.$$

Une telle décomposition est unique.

Nous considérons un temps d'arrêt α (tel que ζ^+ ou ζ^-), bien défini sur la filtration $\mathcal{F}_n$ construite sur la suite X_i. Nous allons décomposer la marche aléatoire à partir de ce temps d'arrêt. Considérons la série génératrice associée à la transformée de Laplace de la distribution S_n; nous avons

$$E\left(\sum_{n=0}^{\infty} u^n e^{-\xi S_n}\right) = E\left(\sum_{n=0}^{\infty} \left[ue^{-\xi X}\right]^n\right) = \frac{1}{1 - uE(e^{-\xi X})}. \tag{2.10}$$

Nous décomposons le premier terme de l'équation 2.10 en

$$E\left(\sum_{n=0}^{\infty} u^n e^{-\xi S_n}\right) = E\left(\sum_{n=0}^{\alpha-1} u^n e^{-\xi S_n}\right) + E\left(\sum_{n=\alpha}^{\infty} u^n e^{-\xi S_n}\right), \tag{2.11}$$

étant entendu que sur l'ensemble $\{\alpha = \infty\}$ la première somme est $\sum_{n=0}^{\infty}$ et la deuxième est vide et égale à zéro. Le dernier terme de l'équation 2.11 peut être réécrit comme

$$E\left(\sum_{n=\alpha}^{\infty} u^n e^{-\xi S_n}\right) = E\left(\sum_{n=0}^{\infty} u^{\alpha+n} e^{-\xi S_{\alpha+n}}\right) = E\left(u^\alpha e^{-\xi S_\alpha} \sum_{n=0}^{\infty} u^n e^{-\xi(S_{\alpha+n} - S_\alpha)}\right). \tag{2.12}$$

Le lemme 1 nous dit que $S_{\alpha+n} - S_\alpha$ a la même loi que S_n et est indépendant de S_α et de α. Le membre de droite de 2.12 est donc égal à

$$E\left(u^\alpha e^{-\xi S_\alpha}\right) E\left(\sum_{n=0}^{\infty} u^n e^{-\xi S_n}\right) = E\left(u^\alpha e^{-\xi S_\alpha}\right) \frac{1}{1 - uE(e^{-\xi X})}.$$

Nous pouvons donc réécrire 2.10 sous la forme

$$\frac{1}{1 - uE(e^{-\xi X})}\left[1 - E\left(u^\alpha e^{-\xi S_\alpha}\right)\right] = E\left(\sum_{n=0}^{\alpha-1} u^n e^{-\xi S_n}\right). \tag{2.13}$$

Nous avons donc, en prenant $\alpha = \zeta^-$, temps d'atteinte du demi-plan d'ordonnées négatives,

$$\frac{1}{1 - uE(e^{-\xi X})} = \frac{E\left(\sum_{n=0}^{\zeta^- -1} u^n e^{-\xi S_n}\right)}{1 - E\left(u^{\zeta^-} e^{-\xi S_{\zeta^-}}\right)}. \tag{2.14}$$

Notons

$$\phi_-(u, \xi) = \frac{1}{1 - E\left(u^{\zeta^-} e^{-\xi S_{\zeta^-}}\right)}. \tag{2.15}$$

Comme $S_{\zeta^-} \leq 0$ et $|u| < 1$, sur $\Re\xi \leq 0$, nous avons

$$E\left(u^{\zeta^-} e^{-\xi S_{\zeta^-}}\right) < |u|,$$

et donc

$$1 - |u| \leq |1 - E\left(u^{\zeta^-} e^{-\xi S_{\zeta^-}}\right)| \leq 1 + |u|;$$

donc ϕ_- et son inverse sont bornées sur $\{\Re\xi \leq 0\}$.

Notons à présent

$$\phi_+(u, \xi) = E\left(\sum_{0 \leq n < \zeta^-} u^n e^{-\xi S_n}\right). \tag{2.16}$$

Nous devons montrer que ϕ_+ possède aussi les propriétés énoncées dans le théorème 2.

Nous allons utiliser à cette fin un argument probabiliste en suivant Neveu [61] et Robert [71]. Pour $n < \zeta^-$, nous avons $S_1 > 0, \ldots, S_{n-1} > 0, S_n > 0$; nous pouvons donc écrire

$$\phi_+(u, \xi) = \sum_{n \geq 0} E\left(u^n e^{-\xi S_n} 1_{\{S_1 > 0, \ldots, S_{n-1} > 0, S_n > 0\}}\right).$$

D'après l'indépendance des X_i, le vecteur $(X_1, X_2, \ldots, X_n)$ a même loi que $(X_n, X_{n-1}, \ldots, X_1)$; donc $(S_n, 1_{\{S_1>0,\ldots,S_{n-1},S_n>0\}})$ et $(S_n, 1_{\{S_n > S_{n-1}, S_n > S_{n-2}, \ldots, S_n > 0\}})$ ont aussi même loi. On peut donc écrire la fonction ϕ_+ comme

$$\phi_+(u, \xi) = \sum_{n \geq 0} E\left(u^n e^{-\xi S_n} 1_{\{S_n > S_{n-1}, S_n > S_{n-2}, \ldots, S_n > 0\}}\right). \tag{2.17}$$

Notons ζ^+ le premier temps "record" (en considérant une échelle croissante de la marche aléatoire (S_n)); c'est aussi le temps d'atteinte du demi-plan strictement positif, puisque la valeur atteinte est strictement plus grande que la valeur 0 à l'origine. Nous avons

$$\zeta^+ = \inf\{k > 0 / S_k > 0\} = \inf\{k > 0 / S_k > S_{k-1}, \ldots, S_k > S_1, S_k > 0\}.$$

Les termes de la somme infinie de l'équation 2.17 valent 1 pour $n = 0$ et 0 pour $n < \zeta^+$. Nous pouvons donc écrire

$$\phi_+(u, \xi) = 1 + E\left(u^{\zeta^+} e^{-\xi S_{\zeta^+}} \sum_{n \geq \zeta^+} u^{n-\zeta^+} e^{-\xi(S_n - S_{\zeta^+})} 1_{\{S_n > S_{n-1}, S_n > S_{n-2}, \ldots, S_n > 0\}}\right). \tag{2.18}$$

La somme dans l'équation 2.18 porte sur tous les instants records de la marche aléatoire $(S_{n+\zeta^+} - S_{\zeta^+}; n \geq 0)$; d'autre part, les instants de records de la marche aléatoire $(S_{n+\zeta^+} - S_{\zeta^+})$ sont ceux de la marche aléatoire (S_n) après l'instant ζ^+. Comme par définition de ζ^+, premier temps record strictement positif, nous avons $\{S_n > S_{\zeta^+}\} \Rightarrow \{S_n > S_i, \forall i < \zeta^+\}$, nous avons

$$\begin{aligned}
&1_{\{n \geq \zeta^+, S_n > S_{n-1}, S_n > S_{n-2}, \ldots, S_n > 0\}} \\
&\quad = 1_{\{n \geq \zeta^+, S_n > S_{n-1}, S_n > S_{n-2}, \ldots, S_n > S_{\zeta^+}\}} 1_{\{n \geq \zeta^+, S_n \geq S_{\zeta^+ - 1}, S_n > S_{\zeta^+ - 2}, \ldots, S_n > 0\}} \\
&\quad = 1_{\{n \geq \zeta^+, S_n > S_{n-1}, S_n > S_{n-2}, \ldots, S_n > S_{\zeta^+}\}} \\
&\quad = 1_{\{n \geq \zeta^+, S_n - S_{\zeta^+} > S_{n-1} - S_{\zeta^+}, S_n - S_{\zeta^+} > S_{n-2} - S_{\zeta^+}, \ldots, S_n - S_{\zeta^+} > 0\}}.
\end{aligned}$$

Le lemme 1 nous dit que la loi de $S_n - S_{\zeta^+}$ est identique à celle de $S_{n-\zeta^+}$ $(n \geq \zeta^+)$. Nous pouvons donc réécrire l'espérance de la somme de l'équation 2.18 sous la forme

$$\begin{aligned}
&E\left(\sum_{n \geq \zeta^+} u^{n-\zeta^+} e^{-\xi(S_n - S_{\zeta^+})} 1_{\{n \geq \zeta^+, S_n - S_{\zeta^+} > S_{n-1} - S_{\zeta^+}, S_n - S_{\zeta^+} > S_{n-2} - S_{\zeta^+}, \ldots, S_n - S_{\zeta^+} > 0\}}\right) \\
&\quad = E\left(\sum_{n \geq \zeta^+} u^{n-\zeta^+} e^{-\xi(S_n - S_{\zeta^+})} 1_{\{S_{n-\zeta^+} > S_{n-\zeta^+ - 1}, S_{n-\zeta^+} > S_{n-\zeta^+ - 2}, \ldots, S_{n-\zeta^+} > S_0\}}\right);
\end{aligned}$$

par translation de l'indice n, nous obtenons alors

$$\begin{aligned}
&E\left(\sum_{n \geq \zeta^+} u^{n-\zeta^+} e^{-\xi(S_n - S_{\zeta^+})} 1_{\{S_n > S_{n-1}, S_n > S_{n-2}, \ldots, S_n > 0\}}\right) \\
&\quad = \sum_{n \geq 0} E\left(u^n e^{-\xi S_n} 1_{\{S_1 > 0, \ldots, S_{n-1} > 0, S_n > 0\}}\right) = \phi_+(u, \xi).
\end{aligned}$$

Nous avons obtenu une représentation de $\phi_+(u, \xi)$ analogue à celle de $\phi_-(u, \xi)$,

$$\phi_+(u, \xi) = \frac{1}{1 - E\left(u^{\zeta^+} e^{-\xi S_{\zeta^+}}\right)}.$$

De même que $\phi_-(u, \xi)$ était bornée ainsi que son inverse sur $\Re\xi \leq 0$, $\phi_+(u, \xi)$ est bornée ainsi que son inverse sur $\Re\xi \geq 0$; de plus, comme $S_{\zeta^+} > 0$,

$$\lim_{\Re\xi \to +\infty} \phi_+(u, \xi) = 1.$$

Nous démontrons au chapitre 3 (paragraphe 3.1.1) l'analycité de ϕ_- et ϕ_+, pour $|u| < 1$.

Nous avons montré l'existence de deux fonctions vérifiant les conditions 1. et 2. de notre théorème. Il nous reste à montrer l'unicité de la décomposition.

Soit deux autres fonctions ψ_+ et ψ_- vérifiant les propriétés (*1.*) et (*2.*). La relation 2.9 nous donne

$$\frac{1}{1 - uE(e^{-\xi X})} = \phi_+(u,\xi)\phi_-(u,\xi) = \psi_+(u,\xi)\psi_-(u,\xi), \qquad \text{ou}$$

$$\frac{\psi_+(u,\xi)}{\phi_+(u,\xi)} = \frac{\phi_-(u,\xi)}{\psi_-(u,\xi)}, \quad |u| < 1, \Re\xi = 0. \tag{2.19}$$

La fonction $H(\xi)$ définie par

$$H(\xi) = \frac{\psi_+(u,\xi)}{\phi_+(u,\xi)}, \quad \Re\xi \geq 0,$$
$$H(\xi) = \frac{\phi_-(u,\xi)}{\psi_-(u,\xi)}, \quad \Re\xi \leq 0,$$

est analytique sur $\mathbb{C} - \{\Re\xi = 0\}$ et continue sur $\mathbb{C}$; elle est donc analytique sur $\mathbb{C}$ tout entier (on démontrera ces propriétés au chapitre 3, théorème 3.1). Cette fonction est de plus bornée, l'étant dans chacun des demi-plans $\Re(z) \leq 0$ et $\Re(z) \geq 0$; nous savons donc par le théorème de Liouville qu'elle est constante; or

$$\lim_{\Re\xi \to +\infty} H(\xi) = \lim_{\Re\xi \to +\infty} \frac{\psi_+(u,\xi)}{\phi_+(u,\xi)} = 1,$$

donc $H \equiv 1$, $\psi_+ = \phi_+$, $\psi_- = \phi_-$ et notre décomposition est bien unique.

2.2.3 Loi de ζ^- (resp. ζ^+) et de S_{ζ^-} (resp. S_{ζ^+}).

Nous faisons dans ce qui suit usage de la propriété Taubérienne suivante, (voir chapitre 3 théorème 40):

Propriété 3. *Soit une suite de nombres réels $c_n \geq 0$, telle que $\sum_{n=0}^{\infty} c_n r^n$ converge pour $0 \leq r < 1$, alors*

$$\lim_{r \to 1^-} \sum_{n=0}^{\infty} c_n r^n = \sum_{n=0}^{\infty} c_n,$$

où $\sum_{n=0}^{\infty} c_n$ peut être fini ou infini.

Nous allons démontrer le théorème suivant concernant le temps d'arrêt ζ^- (resp. ζ^+).

Théorème 4. *La fonction génératrice du temps d'arrêt ζ^- est donnée par*

$$E\left(u^{\zeta^-}\right) = 1 - \exp\left(-\sum_{i=1}^{\infty} \frac{u^i}{i} P\left[S_i \leq 0\right]\right), \qquad et \tag{2.20}$$

$$P(\zeta^- < \infty) = 1 \quad \textit{si et seulement si} \quad \sum_{i=1}^{\infty} \frac{1}{i} P\left[S_i \leq 0\right] = \infty, \tag{2.21}$$

auquel cas

$$E(\zeta^-) = \exp\left(\sum_{i=1}^{\infty} \frac{1}{i} P\left[S_i > 0\right]\right), \tag{2.22}$$

et

$$1 - E\left(e^{-\xi S_{\zeta^-}}\right) = \exp\left(-\sum_{i=1}^{\infty} \frac{1}{i} E\left(e^{-\xi S_i}; S_i \leq 0\right)\right) \tag{2.23}$$

Preuve. En appliquant l'unicité de la factorisation de Wiener-Hopf aux décompositions obtenues dans les équations 2.7 et 2.8, nous obtenons les deux identités

$$\phi_-(u,\xi) = \frac{1}{1 - E\left(u^{\zeta^-} e^{-\xi S_{\zeta^-}}\right)} = \prod_{i=1}^{\infty} \exp\left(\frac{u^i}{i} E\left(e^{-\xi S_i}; S_i \leq 0\right)\right) \quad \text{et}, \tag{2.24}$$

$$\phi_+(u,\xi) = E\left(\sum_{0 \leq n < \zeta^-} u^n e^{-\xi S_n}\right) = \prod_{i=1}^{\infty} \exp\left(\frac{u^i}{i} E\left(e^{-\xi S_i}; S_i > 0\right)\right). \tag{2.25}$$

En remarquant que $u^{\zeta^-} = 0$ pour $|u| < 1$ et $\zeta^- = \infty$, nous pouvons réécrire l'équation 2.24 en

$$1 - E\left(u^{\zeta^-} e^{-\xi S_{\zeta^-}}\right) = 1 - E\left(u^{\zeta^-} e^{-\xi S_{\zeta^-}}; \zeta^- < \infty\right) = \exp\left(-\sum_{i=1}^{\infty} \frac{u^i}{i} E\left(e^{-\xi S_i}; S_i \leq 0\right)\right); \tag{2.26}$$

En faisant $\xi = 0$ dans 2.26, comme $E\left(1; S_i \leq 0\right) = P(S_i \leq 0)$, nous obtenons l'équation 2.20.

Pour $|u| < 1$, en prenant une détermination convenable du logarithme, nous pouvons écrire l'équation 2.24 sous la forme

$$\sum_{i=1}^{\infty} \frac{u^i}{i} E\left(e^{-\xi S_i}; S_i \leq 0\right) = \log(\phi_-(u,\xi)). \tag{2.27}$$

L'analycité de ϕ_- et ϕ_+ entraîne que la série du premier terme de l'équation 2.27 converge pour $|u| < 1$. Nous pouvons donc appliquer la propriété 3 en faisant $u = 1$ dans l'équation 2.27, ce qui nous donne l'équation 2.23.

Pour la même raison, la limite de $E(u^{\zeta^-})$ s'obtient en faisant $u = 1$ dans la série du membre droit de l'équation 2.20. Nous notons à nouveau que $u^{\zeta^-} = 0$ pour $\zeta^- = \infty$, et comme, d'autre part, nous avons

$$\begin{aligned}\lim_{u\to 1^-} E(u^{\zeta^-}) &= \lim_{u\to 1^-} E(u^{\zeta^-}; \zeta^- < \infty) \\ &= \lim_{u\to 1^-} \sum_{i=1}^{\infty} P(\zeta^- = i)u^i = \sum_{i=1}^{\infty} P(\zeta^- = i) = P(\zeta^- < \infty),\end{aligned}$$

nous obtenons que $\{P(\zeta^- < \infty) = 1\}$ si et seulement si la somme $\sum_1^\infty \frac{1}{i} P(S_i \leq 0)$ est infinie. Enfin, en faisant $\xi = 0$ dans l'équation 2.25, nous avons

$$E\left(\sum_{n=1}^{\zeta^- -1} u^n\right) = \exp\left(\sum_{i=1}^{\infty} \frac{u^i}{i} P\left[S_i > 0\right]\right); \tag{2.28}$$

$$\begin{aligned}\text{mais} \quad E\left(\sum_{n=0}^{\zeta^- -1} u^n\right) &= \sum_{k=0}^{\infty} P(\zeta^- = k) \sum_{n=0}^{k-1} u^n = \sum_{k=0}^{\infty}\sum_{n=0}^{\infty} u^n P(\zeta^- = k) \\ &= \sum_{n=0}^{\infty}\sum_{k>n}^{\infty} u^n P(\zeta^- = k) = \sum_{n=0}^{\infty} u^n P(\zeta^- > k),\end{aligned}$$

par interversion des deux sommations; nous pouvons donc réécrire l'équation 2.28, en utilisant la propriété 3 lorsque u tend vers 1,

$$\lim_{u\to 1^-} \sum_{n=0}^{\infty} u^n P(\zeta^- > n) = \sum_{n=0}^{\infty} P(\zeta^- > n) = E(\zeta^-) = \exp\left(\sum_{i=1}^{\infty} \frac{1}{i} P\left(S_i > 0\right)\right). \tag{2.29}$$

□

2.3 Démonstration de la formule des valeurs extrêmes d'Iglehart-Karlin et calcul des constantes

Nous revenons à la démonstration proprement dite des formules d'Iglehart-Karlin que nous énonçons complètement:

Théorème 5. *Soit une variable aléatoire X telle que $E(X) < 0$ et $\{P(X > 0) > 0\}$ et X_n une suite de variables aléatoires indépendantes et distribuées comme X. Soit les sommes partielles $S_k = \sum_{i=1}^{k} X_k$ et $M(n)$ la section de valeur maximum de la marche aléatoire (S_n), telle que $M(n) = \max_{1\leq i\leq j\leq n}\{S_j - S_i\}$.*

La queue de distribution de $M(n)$ est donnée par la formule (loi de Gumbel)

$$\Pr\left(M(n) > \frac{\log(n)}{\lambda} + x\right) = 1 - e^{-Ke^{-\lambda x}}, \tag{2.30}$$

où la constante λ est la solution positive de l'équation en ξ

$$E[e^{\xi X}] = 1, \tag{2.31}$$

et la constante K est donné par la formule

$$K = \frac{\exp\left\{-2\sum_{k=1}^{\infty}\frac{1}{k}\left(E(e^{\lambda S_k}; S_k < 0) + \Pr(S_k \geq 0)\right)\right\}}{\lambda E\left(X e^{\lambda X}\right)} > 0. \tag{2.32}$$

Preuve. Nous suivons le plan de la démonstration donné au paragraphe 2.1.1. Nous traitons dans le paragraphe 2.3.1 le cas où X est une variable continue, et dans le paragraphe 2.3.2 le cas où X est une variable discrète. Nous supposerons de plus que X est à support compact.

Les définitions sont celles du tableau 2.1.

2.3.1 Cas d'une variable X continue

2.3.1.1 Calcul de la queue de distribution de la somme partielle record ($R(y)$)

Considérons le temps d'arrêt ζ^+ correspondant à la première somme partielle strictement positive, de valeur $Z^+ = S_{\zeta^+}$, qui est une variable aléatoire à défaut de loi $L_{Z^+}(y) = P(0 < Z^+ \leq y)$. La transposition de l'équation 2.23 du cas ζ^- au cas ζ^+ nous donne

$$1 - E\left(e^{\xi Z^+}; \zeta^+ < \infty\right) = \exp\left(-\sum_{i=1}^{\infty}\frac{1}{i}E\left(e^{\xi S_i}; S_i > 0\right)\right), \tag{2.33}$$

que nous pouvons réécrire

$$\begin{aligned} 1 - E\left(e^{\xi Z^+}\right) &= \exp\left(-\sum_{i=1}^{\infty}\frac{1}{i}\left[E(e^{\xi S_i}) - E\left(e^{\xi S_i}; S_i \leq 0\right)\right]\right) \\ &= \exp\left(-\sum_{i=1}^{\infty}\frac{1}{i}\left(E^{\xi X}\right)^i\right)\exp\left(\sum_{i=1}^{\infty}\frac{1}{i}E\left(e^{\xi S_i}; S_i \leq 0\right)\right) \\ &= \left(1 - E(e^{\xi X})\right)\exp\left(\sum_{i=1}^{\infty}\frac{1}{i}E\left(e^{\xi S_i}; S_i \leq 0\right)\right). \end{aligned} \tag{2.34}$$

En choisissant λ tel que $E(e^{\lambda X}) = 1$, nous obtenons aussi $E(e^{\lambda Z^+}; \zeta^+ < \infty) = 1$. Nous montrons dans le lemme suivant qu'il existe toujours un tel λ.

Lemme 6. *Soit une variable aléatoire X telle que $E(X) < 0$ et $\{P(X > 0) > 0\}$; alors il existe un réel λ tel que $E(e^{\lambda X} = 1)$. On a alors $E(Xe^{\lambda X}) > 0$.*

Preuve. Considérons la fonction $\theta(t) = E(e^{tX})$. Nous avons $\theta(0) = 1$ et $\theta'(0) = E(X) < 0$; de plus, $\{P(X > 0) > 0\}$ implique qu'il existe ϵ et α positifs tels que $\{P(X > \epsilon) > \alpha\}$. Nous avons alors

$$E(e^{tX}) > E(e^{tX}; X > \epsilon) > e^{\epsilon t}P(X > \epsilon) = e^{\epsilon t}\alpha \to \infty \quad \text{quand} \quad t \to \infty;$$

ceci entraîne par continuité qu'il existe $\lambda \in [0, \infty]$ tel que $\theta(\lambda) = 1$.

D'autre part, la fonction $\theta(t)$ est convexe ($\theta''(t) = E(X^2 e^{tX}) > 0$); de plus, $\theta(0) = 1$ et $\theta'(0) = E(X) < 0$. On en déduit $\theta'(\lambda) = E(Xe^{\lambda X}) > 0$. □

En faisant $\xi = 0$ dans l'équation 2.33, nous obtenons une version plus fine de 2.21

$$1 - P(\zeta^+ < \infty) = 1 - L_{Z^+}(\infty) = \exp\left(\sum_{i=1}^{\infty} -\frac{1}{i} P(S_i > 0)\right) \tag{2.35}$$

Nous considérons à présent le record absolu R de la marche aléatoire sur $[0, \infty]$ de distribution $R(y) = P(R \leq y)$. Nous avons l'équation de convolution

$$\begin{aligned} P(R \leq y) &= P(\zeta^+ = \infty) + \int_0^y P(R_{[\zeta^+,\infty]} < y - t) P(Z^+ = t) P(\zeta^+ < \infty) dt \\ &= P(\zeta^+ = \infty) + \int_0^y P(R_{[\zeta^+,\infty]} < y - t) P(Z^+ = t) dt, \end{aligned} \tag{2.36}$$

où $R_{[\zeta^+,\infty]}$ représente le record absolu de la marche aléatoire commençant en ζ^+; la deuxième équation s'obtient par l'identité $1_{\{Z^+>0\}} = 1_{\{\zeta^+<\infty\}}$.

Nous pouvons écrire l'équation 2.36 comme

$$R(y) = 1 - L_{Z^+}(\infty) + \int_0^y R(y-t) dL_{Z^+}(t), \tag{2.37}$$

mais dans cette équation la distribution dL_{Z^+} est une distribution à défaut ($L_{Z^+}(\infty) < 1$). Nous allons faire la substitution $R(y) = 1 - V(y)e^{-\lambda y}$; ceci nous donne

$$1 - V(y)e^{-\lambda y} = 1 - L_{Z^+}(\infty) + \int_0^y \left(1 - V(y-t)e^{-\lambda(y-t)}\right) dL_{Z^+}(t), \tag{2.38}$$

ou

$$V(y) = (L_{Z^+}(\infty) - L_{Z^+}(y))e^{\lambda y} + \int_0^y V(y-t)e^{\lambda t} dL_{Z^+}(t); \tag{2.39}$$

dans cette dernière équation, la distribution $\Lambda_{Z^+}(t) = e^{\lambda t} dL_{Z^+}(t)$ est une distribution sans défaut ($\Lambda_{Z^+}(\infty) = E(e^{\lambda Z^+}) = 1$); nous pouvons donc utiliser les théorèmes standard de la théorie du renouvellement. Le théorème du renouvellement est une conséquence du théorème de Blackwell (voir Feller [27] ou Durrett [26] pour ces deux théorèmes); ces deux théorèmes ont une formulation non dégénérée quand la distribution sur laquelle est construite le processus de renouvellement $U_i > 0, T_n = \sum_{i=1}^n U_i$ est sans défaut ($\{P(U_i) < \infty\} = 1$).

Le théorème du renouvellement s'énonce comme suit:

Théorème 7. *Soit F la fonction de répartition d'une variable aléatoire positive de moyenne μ. Supposons que h soit une fonction directement Riemann intégrable et que la fonction H soit la solution de l'équation du renouvellement*

$$H(t) = h(t) + \int_0^t H(t-x) dF(x). \tag{2.40}$$

Si F n'est pas arithmétique, alors

$$\lim_{t\to\infty} H(t) = \frac{1}{\mu} \int_0^\infty h(x) dx, \quad si \quad \mu < \infty.$$

Nous démontrons au chapitre 3 que, si X est absolument continue et à support compact, la fonction $(L_{Z^+}(\infty) - L_{Z^+}(y))e^{\lambda y}$ est directement Riemann intégrable (paragraphe 3.2.1). Le théorème 7, appliqué à l'équation 2.39, nous donne donc:

$$\lim_{y\to\infty} V(y) = \lim_{y\to\infty} e^{\lambda y}(1 - R(y)) = \frac{\int_0^\infty e^{\lambda y}(L_{Z^+}(\infty) - L_{Z^+}(y))dy}{\int_0^\infty y d\Lambda_{Z^+}(y)} = C_1. \tag{2.41}$$

L'intégration par partie du numérateur de la dernière expression nous donne $\frac{1}{\lambda}(1 - L_{Z^+}(\infty))$; le dénominateur s'écrit

$$\int_0^\infty y d\Lambda_{Z^+}(y) = \int_0^\infty y e^{\lambda y} dL_{Z^+}(y) = E(Z^+ e^{\lambda Z^+}; \zeta^+ < \infty) \quad \text{et, donc}$$

$$C_1 = \frac{1 - L_{Z^+}(\infty)}{\lambda E(Z^+ e^{\lambda Z^+}; \zeta^+ < \infty)}, \quad \text{et} \quad R(y) \to 1 - C_1 e^{-\lambda y}. \tag{2.42}$$

Nous pouvons calculer le dénominateur en dérivant en ξ le premier et le dernier terme de l'équation 2.34, ce qui est licite compte tenu de l'analycité de la fonction ϕ^+ dans la factorisation de Wiener-Hopf, et en faisant $\xi = \lambda$, ce qui annule le terme contenant $(1 - E(e^{\xi X}))$. Ceci nous donne

$$E(Z^+ e^{\lambda Z^+}) = E(X e^{\lambda X}) \exp\left(\sum_{i=1}^{\infty} \frac{1}{i} E\left(e^{\lambda S_i}; S_i \le 0\right)\right). \tag{2.43}$$

2.3.1.2 Calcul de la queue de distribution de la variable Q_1, maximum de la marche aléatoire avant le premier passage dans le demi-plan négatif

Nous suivons à nouveau Iglehart [44]. Nous considérons la variable $Q = Q_1$ correspondant au maximum atteint par la marche aléatoire avant d'atteindre pour la première fois le demi-plan d'ordonnées négatives, de distribution $Q(y)$

Soit le temps d'arrêt $a_y = a_{[0,y]} = \inf\{i; S_i \notin [0, y]\}$, qui est le temps de sortie de l'intervalle $[0, y]$.

Nous remarquons que si $Q < y$, comme Q est le maximum atteint avant le premier passage dans le demi-plan négatif, la sortie de l'intervalle $[0, y]$ se fait par une valeur négative t ($t = S_{a_y} < 0$), et nous noterons

$$A^-_{(y)} = 1_{\{Q \le y\}} S_{a_y} \quad \text{et} \quad A^-_{(y)}(t) = P(S_{a_y} \le t, Q \le y)$$

Nous décomposons les probabilités suivant que le record Q est supérieur ou inférieur à y. Nous écrivons donc

$$P(S_R > y) = P(S_R > y, Q \ge y) + P(S_R > y, Q < y); \tag{2.44}$$

d'autre part, les inégalités $Q < y \le S_R$ indiquent que le record absolu a lieu après le temps a_y; il est équivalent de dire que le record absolu de la marche aléatoire partant de l'origine vaut y, ou que le record absolu de la marche aléatoire partant du temps a_y et du point d'abscisse négative t vaut $y-t$. En appliquant à nouveau le lemme 1, ceci s'écrit sous la forme de l'équation de convolution

$$\begin{aligned} P(S_R > y, A^-_{(y)} = t, Q < y) &= P(S_{R[a_y,\infty]} > y-t, A^-_{(y)} = t, Q < y)) \\ &= P(S_{R[a_y,\infty]} > y-t) P(A^-_{(y)} = t, Q < y); \end{aligned}$$

Nous avons d'autre part,

$$\{Q > y \Rightarrow S_R > y\}, \quad \text{et donc} \quad P(S_R > y, Q > y) = P(Q > y);$$

nous pouvons donc réécrire l'équation 2.44, en utilisant le lemme 1 (répliques probabilistes) pour la distribution de $S_{R[a_y,\infty]}$, sous la forme

$$P(S_R > y) = P(Q > y) + \int_{-\infty}^{0^-} P(S_R > y-t) d_t P(S_{a_y} < t, Q < y), \tag{2.45}$$

ou encore, sous la forme,

$$1 - R(y) = 1 - Q(y) + \int_{-\infty}^{0^-} (1 - R(y-t)) dA^-_{(y)}(t). \tag{2.46}$$

Quand $y \to \infty$, nous avons $dA^-_{(y)}(t) \to dZ^-(t)$, presque sûrement, avec Z^- la valeur de la marche aléatoire lors du premier record négatif (la sortie de l'intervalle des ordonnées $[0, y]$ se fait au point d'abscisse ζ^-); ceci découle du fait que $\mu = E(X) < 0$, et que, par conséquent, presque sûrement, $Q < \infty$.

En prenant la limite à l'infini, nous pouvons écrire l'équation 2.46 comme

$$\lim_{y\to\infty} (1 - R(y)) = \lim_{y\to\infty} (1 - Q(y)) + \lim_{y\to\infty} \left\{ \int_{-\infty}^{0^-} (1 - R(y-t)) dZ^-(t) \right\};$$

en multipliant par $e^\lambda y$, et en utilisant le fait (équation 2.41) que

$$\lim_{y\to\infty} (1 - R(y)) e^{\lambda y} = \lim_{y\to\infty} (1 - R(y-t)) e^{\lambda(y-t)} = C_1,$$

nous obtenons à la limite

$$C_1 = \lim_{y\to\infty} (1 - Q(y)) e^{\lambda y} + \int_{-\infty}^{0^-} C_1 e^{\lambda t} dZ^-(t),$$

ce qui donne la queue de distribution de Q,

$$\lim_{y\to\infty} (1 - Q(y)) = C_1 (1 - E(e^{\lambda Z^-})) e^{-\lambda y} = C_2 e^{-\lambda y}. \tag{2.47}$$

2.3.1.3 Queue de distribution de $M(n)$

Le lemme 1 nous permet d'affirmer que les marches aléatoires commençant après les instants d'échelle de records négatifs successifs q_i sont toutes des répliques de la marche aléatoire initiale; par conséquent les records $Q_1, Q_2, \ldots, Q_i, \ldots$ obtenus dans les intervalles de temps $[q_0, q_1[, [q_1, q_2[, \ldots, [q_{i-1}, q_i[, \ldots$ sont indépendants et distribués identiquement à $Q_1 = Q$, dont nous avons calculé la queue de distribution $Q(y)$ au paragraphe précédant. Considérons les m premières excursions de la marche aléatoire délimitées par les temps q_i et $M(q_m) = \max\{Q_1, Q_2, \ldots, Q_m\}$. Nous avons alors

$$P\left(M(q_m) \leq \frac{\log m}{\lambda} + x\right) = \left[Q\left(\frac{\log m}{\lambda} + x\right)\right]^m;$$

en utilisant la limite obtenue dans l'équation 2.47, nous obtenons

$$\begin{aligned}\lim_{m\to\infty} P\left(M(q_m) \leq \frac{\log m}{\lambda} + x\right) &= \lim_{m\to\infty}\left(1 - C_2 e^{-\log m} e^{-\lambda x}\right)^m \\ &= \lim_{m\to\infty}\left(1 - \frac{1}{m} C_2 e^{-\lambda x}\right)^m = e^{-C_2 e^{-\lambda x}}. \qquad (2.48)\end{aligned}$$

Nous avons calculé la queue de distribution de $M(n)$ pour les grandes valeurs $n = q_m$; il nous reste à la calculer pour toutes les grandes valeurs de n.

Nous utilisons le théorème limite suivant de la théorie du renouvellement (voir le théorème 34 au chapitre 3):

Théorème 8. *Soit $U_1, U_2, \ldots$ une suite de variables positives indépendantes et identiquement distribuées, et $T_i = U_1 + \cdots + U_i$ des temps de renouvellement; soit $N_t = \sup\{n; T_n \leq t\}$. Alors, si $E(U_1) = \mu \leq \infty$, quand $t \to \infty$, on a $N_t/t \to 1/\mu$ presque sûrement.*

Nous utiliserons ce théorème avec $U_i = q_i$, en notant $E(q_i) = E(q_1) = E(\zeta^-) = \chi$. Remarquons que $N_{q_i} = \sup\{n; T_n \leq q_i\} = i$.

Soit $m' = \left\lfloor \frac{n}{E(q)} \right\rfloor = \left\lfloor \frac{n}{\chi} \right\rfloor$ où $\lfloor h \rfloor$ représente la partie entière de h; le théorème 8 nous donne

$$\lim_{q_{m'}\to\infty} N_{q_{m'}}/q_{m'} \to 1/\chi;$$

En tenant compte de notre dernière remarque, qui nous donne $N_{q_{m'}} = m'$, nous obtenons $\lim_{m'\to\infty} q_{m'}/m' = \chi$, et donc,

$$\frac{q_{\lfloor \frac{n}{\chi} \rfloor}}{n} = \frac{q_{m'}}{n} \to \frac{m'\chi}{n} = \frac{\left\lfloor \frac{n}{\chi} \right\rfloor}{\frac{n}{\chi}} \to 1, \quad \text{presque sûrement, quand } n \to \infty. \qquad (2.49)$$

Sachant que $q_1 = q = \zeta^-$, nous pouvons écrire l'équation 2.48 sous la forme

$$\lim_{n\to\infty} P\left(M(q_{\lfloor \frac{n}{\chi} \rfloor}) \leq \frac{\log \left\lfloor \frac{n}{\chi} \right\rfloor}{\lambda} + x\right) = e^{-C_2 e^{-\lambda x}}, \qquad (2.50)$$

et, donc, en utilisant 2.49,

$$\lim_{n\to\infty} P\left(M(n) \leq \frac{\log n}{\lambda} + x\right) = e^{-\frac{C_2}{\chi} e^{-\lambda x}} = e^{-\frac{C_2}{E(\zeta^-)} e^{-\lambda x}}. \qquad (2.51)$$

En tenant compte du fait, d'une part que $Z^+ = S_{\zeta^+}$ et $Z^- = S_{\zeta^-}$, et d'autre part des équations 2.22, 2.23, 2.35, 2.42, 2.43, 2.47 et 2.50, nous avons donc démontré dans le cas continu la formule 2.1

$$\Pr\left(M(n) > \frac{\log(n)}{\lambda} + x\right) = 1 - e^{-Ke^{-\lambda x}},$$

avec λ solution positive de l'équation $E(e^{tX}) = 1$.

K est donné par

$$K = \frac{C_2}{E(\zeta^-)} = \frac{C_1(1 - E(e^{\lambda Z^-}))}{E(\zeta^-)} = \frac{(1 - L_{Z^+}(\infty))(1 - E(e^{\lambda Z^-}))}{\lambda E(Z^+ e^{\lambda Z^+}; \zeta^+ < \infty) E(\zeta^-)};$$

ce qui nous donne la formule 2.32,

$$K = \frac{\exp\left\{-2\sum_{k=1}^{\infty}\frac{1}{k}\left(E(e^{\lambda S_k}; S_k < 0) + \Pr(S_k \geq 0)\right)\right\}}{\lambda E\left(X e^{\lambda X}\right)}.$$

Par le lemme 6 on a $K > 0$. □

2.3.2 Cas d'une variable X discrète

La démonstration de l'unicité de la factorisation de Wiener-Hopf (paragraphe 2.2.2), et celle de l'identité de Spitzer 2.23 qui en découle font usage de l'analycité de la fonction $g(\xi) = E(u^{\zeta^+} e^{-\xi S_{\zeta^+}})$ de la marche aléatoire d'incrément X. Nous utilisons comme hypothèses pour la démonstration de l'analycité de $g(\xi)$ (voir chapitre 3 paragraphe 3.1) le fait que X est à support compact. La démonstration de la décomposition de Wiener-Hopf est donc valide dans le cas où X est une variable discrète.

Pour utiliser le théorème du renouvellement (théorème 7) dans le cas continu, nous supposons (voir chapitre 3 paragraphe 3.2) que la distribution de X est absolument continue. Nous démontrons dans ce paragraphe une version discrète du théorème du renouvellement, qui complétera la démonstration dans le cas d'un modèle discret.

Nous supposerons X arithmétique de pas δ et $|X| < A$ pour un A positif. Dans ces conditions, $Z^+ = S_{\zeta^+}$ et S_R sont des variables arithmétiques bornées positives de pas δ, et nous avons $0 < Z^+ < A$.

L'équation de convolution 2.36 s'écrit à présent,

$$P(R \leq y) = P(\zeta^+ = \infty) + \sum_{t=0}^{y} P\{S_{R[\zeta^+,\infty]} \leq y - t\} P(Z^+ = t), \tag{2.52}$$

ou, en notant $R_n = P(S_R \leq n)$, $r_n = P(S_R = n)$, $L_n = P(0 < Z^+ \leq n)$, et $l_n = P(Z^+ = n)$,

$$R_n = 1 - L_\infty + \sum_{k=0}^{n} R_{n-k} l_k. \tag{2.53}$$

La variable aléatoire X est supposée avoir un support discret de pas δ, et donc, nous avons $P\{X = u\} = 0$ si $u \neq z\delta$, avec $z \in \mathbb{Z}$, et de même, $r_u = l_u = 0$ si $u \neq z\delta$, avec $z \in \mathbb{Z}$.

Nous transformons l'équation 2.53 comme nous l'avons fait pour l'équation 2.37 pour obtenir une distribution sans défaut, par la transformation $R_n = 1 - V_n e^{-n\lambda}$, ce qui nous donne une équation parallèle à l'équation 2.39,

$$V_n = (L_\infty - L_n)e^{n\lambda} + \sum_{k=0}^{n} V_{n-k} e^{k\lambda} l_k. \tag{2.54}$$

Notons que, de même que dans le cas continu, nous avons

$$E(e^{\lambda Z^+}) = E(e^{\lambda X}) = 1, \quad \text{ou encore} \quad \sum_{k=0}^{\infty} e^{k\lambda} l_k = 1.$$

Nous allons à présent utiliser le théorème du renouvellement discret que nous démontrerons dans les conditions où nous l'utilisons, à savoir X à support compact, ce qui correspond dans le théorème que nous énonçons à des suites f_n et b_n n'ayant qu'un nombre fini de termes non nuls.

Théorème 9 (Théorème du renouvellement discret.). *Soient b_n et f_n deux suites à termes positifs telles que*

$$\begin{gathered} f_n \geq 0, \quad , \sum_{k=0}^{\infty} f_k = 1 \quad b_n \geq 0, \quad B = \sum_{k=0}^{\infty} b_k < \infty, \\ \textit{et telles que} \quad 0 \neq \mu = \sum n f_n < \infty. \end{gathered} \tag{2.55}$$

Soit d'autre part l'équation de renouvellement

$$v_n = b_n + f_0 v_n + f_1 v_{n-1} + \cdots + f_n v_0. \tag{2.56}$$

Alors, si f_n n'est pas périodique,

$$v_n \to B\mu^{-1} \quad \textit{quand} \quad n \to \infty,$$

et, si f_n est périodique de pas δ,

$$v_{n\delta} \to B\delta\mu^{-1} \quad \textit{quand} \quad n \to \infty.$$

Preuve. Feller [27] (Volume 1) démontre ce théorème par des moyens arithmétiques et des arguments de limites de suites. Il effectue la démonstration dans le cas non périodique, puis l'étend au cas périodique.

Avec nos hypothèses de compacité sur X, on peut faire la démonstration suivante qui utilise l'intégrale de Cauchy. (On peut consulter Churchill [16] ou Flajolet et Sedgewick [28] pour cette approche).

Considérons, dans le cas non périodique, les séries génératrices associées à nos suites

$$F(z) = \sum_{n=0}^{\infty} f_n z^n, \quad B(z) = \sum_{n=0}^{\infty} b_n z^n, \quad \text{et} \quad V(z) = \sum_{n=0}^{\infty} v_n z^n \quad (z \in \mathbb{C}).$$

Comme nous l'avons mentionné précédemment, les sommes donnant $F(z)$ et $B(z)$ ne comportent qu'un nombre fini de termes; $F(z)$ et $B(z)$ sont donc des polynômes. D'autre part, nous obtenons immédiatement à partir de l'équation de renouvellement

$$V(z) = B(z) + V(z)F(z) = \frac{B(z)}{1-F(z)}. \tag{2.57}$$

Nous allons montrer que le pôle $z = 1$ de $V(z)$ est le pôle de plus petit module, et qu'il est un pôle simple.

$F'(1) = \mu \neq 0$, donc $z = 1$ est un pôle simple de $V(z)$. Montrons que c'est le pôle de plus petit module.

Pour $\{|z| < 1, z \in \mathbb{C}\}$, on a $|F(z)| \leq \sum f_n |z|^n < \sum f_n = 1$.

D'autre part, pour $\{|z| = 1, z \neq 1\}$, supposons qu'il existe un terme non nul de $F(z)$ d'exposant k tel que $z^k = e^{i\theta} \neq 1$ $(0 < \theta < 2\pi)$. Nous avons

$$\Re F(z) = \sum_{i \neq k} f_i \Re z^i + f_k \cos\theta < \sum f_n = 1,$$

puisque θ est strictement compris entre 0 et 2π; donc, pour que $F(z)$ soit égal à 1, il faut que pour tous les termes non nuls de F(z) d'exposants k, on ait $z^k = 1$.

Soit alors n_0 l'exposant du plus petit terme non nul dans la série $F(z)$. On a $z^{n_0} = 1$, et donc $z = e^{2i\pi/n_0}$. La série F(z) n'étant pas périodique, il existe un terme non nul d'exposant n_1 tel que $n_1 \neq kn_0$ $(k \in \mathbb{N})$, ou encore $n_1 = n_0 q + r$, avec $r < n_0, r \in \mathbb{N}$. Nous avons alors $z^{n_1} = e^{(r/n_0)2i\pi} \neq 1$, et donc $F(z) \neq 1$.

Le polynôme $1 - F(z)$ a donc une seule racine, en $z = 1$, dans le disque $|z| \leq 1$. Les autres racines de ce polynôme sont de module strictement plus grand que 1; il existe donc $\epsilon > 0$ tel que toutes ces racines soient de module supérieur à $1 + \epsilon$.

Nous avons d'autre part $\mu = \sum n f_n = F'(1) < \infty$. En développant $B(z)$ et $1 - F(z)$ au voisinage de 1, nous obtenons

$$V(z)(1-z) = \frac{B(1) + (1-z)B'(1) + \frac{1}{2}(1-z)^2 B''(1) + \dots}{F'(1) + \frac{1}{2}(1-z)F''(1) + \dots};$$

le numérateur comme le dénominateur de la dernière expression sont des polynômes, et le résidu de $V(z) = \frac{B(z)}{1-F(z)}$ au point $z = 1$ vaut $B(1)/F'(1)$.

La fonction $h(z) = V(z) - \frac{B(1)}{(1-z)F'(1)}$ est analytique et donc bornée dans le disque $|z| \leq 1 + \epsilon$; nous pouvons maintenant appliquer la formule intégrale de Cauchy

$$\begin{aligned} v_n = [z^n]V(z) &= \int_{|z|=1+\epsilon} \frac{V(z)}{z^n} = \int_{|z|=1+\epsilon} \frac{B(1)}{z^n(1-z)F'(1)} + \int_{|z|=1+\epsilon} \frac{h(z)}{z^n} \\ &= Res_{\{z=1\}} \frac{B(1)}{z^n(1-z)F'(1)} + \mathcal{O}(1+\epsilon)^{-n} = \frac{B(1)}{F'(1)} + \mathcal{O}(1+\epsilon)^{-n} \to \frac{B(1)}{F'(1)} = B\mu^{-1}. \end{aligned}$$

Dans le cas où f_n et b_n sont des suites périodiques de période δ, nous avons

$$F(z) = \sum f_{n\delta} z^{n\delta} = F_*(z^\delta), \quad \text{avec } F_*(z) = \sum f_{n\delta} z^n \quad \text{et}$$
$$B(z) = \sum b_{n\delta} z^{n\delta} = B_*(z^\delta), \quad \text{avec } B_*(z) = \sum b_{n\delta} z^n.$$

La série génératrice $V(z)$, fonction de $F(z)$ et de $B(z)$, est une fonction de z^δ et donc v_n est périodique de période δ. D'autre part, la suite $f_{*n} = f_{n\delta}$ est non périodique, et sa série génératrice est $F_*(z)$. Nous pouvons donc lui appliquer le résultat obtenu pour les suites non périodiques, et, en remarquant que $F_*(z) = F(z^{1/\delta})$, nous obtenons

$$v_{*n} = v_{n\delta} \to \frac{B_*(1)}{F'_*(1)} = \frac{B(1)\delta}{F'(1)} = B\delta\mu^{-1}.$$

□

Nous allons utiliser, dans le cas discret, le petit lemme suivant, relatif à la sommation par parties:

Lemme 10. *Soit u_i et v_i deux suites de nombres, de sommes partielles $U_n = \sum_{i=0}^n u_i$ et $V_n = \sum_{i=0}^n v_i$, et Δ l'opérateur aux différences, tel que $\Delta(G(n)) = G(n) - G(n-1)$; alors, on a:*

$$\sum_{i=1}^n U_i \Delta V_i = U_n V_n - U_0 V_0 - \sum_{i=1}^n V_{i-1} \Delta U_i. \tag{2.58}$$

Preuve. En effet

$$\Delta(U_i V_i) = U_i V_i - U_{i-1} V_{i-1} = \sum_{j=1}^i u_j \sum_{j=1}^i v_j - \sum_{j=1}^{i-1} u_j \sum_{j=1}^{i-1} v_j \tag{2.59}$$

$$= U_i v_i + v_i V_i - u_i v_i = U_i \Delta V_i + V_{i-1} \Delta U_i. \tag{2.60}$$

En sommant de 1 à n le premier et le dernier terme de l'équation 2.59, nous obtenons:

$$\sum_{i=1}^n \Delta(U_i V_i) = U_n V_n - U_0 V_0 = \sum_{i=1}^n U_i \Delta V_i + \sum_{i=1}^n V_{i-1} \Delta U_i,$$

ce qui nous donne l'équation 2.58. □

Dans le cas discret, nous pouvons alors énoncer:

Théorème 11. *Si X est une variable discrète de pas δ, la queue de distribution du record absolu S_R de la marche aléatoire (S_i) est donnée par*

$$R(y) \to 1 - C_1 e^{-\lambda y},$$

$$\textit{avec} \quad C_1 = \frac{(1 - L_{Z^+}(\infty))\delta}{(e^{\lambda\delta} - 1) E(Z^+ e^{\lambda Z^+}; \zeta^+ < \infty)}. \tag{2.61}$$

Nous allons appliquer le théorème 9 à l'équation 2.54

$$V_n = (L_\infty - L_n) e^{n\lambda} + \sum_{k=0}^n V_{n-k} e^{k\lambda} l_k, \tag{2.62}$$

$$\text{avec} \quad f_k = e^{k\lambda} l_k, \quad v_k = V_k, \quad \text{et} \quad b_n = (L_\infty - L_n)e^{n\lambda}.$$

Remarquons que, X étant à support discret de pas δ, nous avons

$$\{f_i = 0, \quad \forall i \neq n\delta\},$$

$$\{R_{n\delta+i} = P(S_R \leq n\delta + i) = R_{n\delta}, \quad \forall i < \delta\}.$$

Nous pouvons donc nous limiter à ne considérer que les termes de rang $i\delta$ dans l'équation 2.62, ce qui nous donne une équation de la forme

$$V_{n\delta} = (L_\infty - L_{n\delta})e^{n\delta\lambda} + \sum_{k=0}^{n} V_{(n-k)\delta} e^{k\delta\lambda} l_{k\delta}, \tag{2.63}$$

Le théorème du renouvellement discret nous donne:

$$V_{n\delta} \to \sum_{i=0}^{\infty} (L_\infty - L_{n\delta})e^{n\delta\lambda}\delta\mu^{-1}. \tag{2.64}$$

Nous sommons par partie cette dernière somme, en utilisant le lemme 10, avec

$$u_0 = L_\infty, \quad u_i = -l_{n\delta} \ (i \geq 1), \quad v_i = e^{i\delta\lambda}, \quad \text{et} \quad V_i = \frac{e^{(i+1)\delta\lambda} - 1}{e^{\delta\lambda} - 1}$$

En remarquant d'une part que $L_{n\delta} = L_\infty$ pour $n\delta > M$, avec M borne supérieure du support compact de X, et d'autre part que $\sum_{i=1}^{\infty} l_{i\delta} = L_\infty$, que $\sum_{i=1}^{\infty} l_{i\delta} e^{i\lambda\delta} = 1$, et que $L_0 = l_0 = P(Z^+ = 0) = 0$, nous avons:

$$\begin{aligned}
\sum_{i=0}^{\infty} (L_\infty - L_{n\delta})e^{n\delta\lambda} &= (L_\infty - L_0) + \sum_{i=0}^{\infty} (L_\infty - L_{n\delta})e^{n\delta\lambda} \\
&= L_\infty - \left[(L_\infty - L_{n\delta})\frac{e^{(n+1)\lambda\delta} - 1}{e^{\lambda\delta} - 1}\right]_0^\infty + \sum_{i=0}^{\infty} \frac{e^{i\lambda\delta} - 1}{e^{\lambda\delta} - 1} l_{i\delta} \\
&= L_\infty - L_\infty + \frac{1 - L_\infty}{e^{\lambda\delta} - 1}.
\end{aligned}$$

Nous obtenons ainsi 2.61. Nous remarquons qu'en faisant tendre δ vers 0 dans l'équation 2.61, nous retrouvons l'équation 2.42

Remarque. Les calculs donnant les queues de distribution des variables Q et $M(n)$ se déduisent immédiatement, dans le cas discret, en remplaçant les intégrales par des sommes, de ceux effectués dans le cas continu; ils nous redonnent les formules 2.47 et 2.51.

Note 12 (Une démonstration directe du théorème 11.).

Nous avons démontré le théorème 11 en suivant l'approche d'Iglehart et de Karlin, c'est à dire en faisant le changement de variable $V_n = 1 - R_n e^{-\lambda n}$, puis en considérant la solution asymptotique d'une équation de renouvellement sur une distribution sans défaut. On peut faire une démonstration plus directe qui est peut-être plus intuitive.

On part à nouveau de l'équation 2.53,

$$R_n = 1 - L_\infty + \sum_{k=0}^{n} R_{n-k} l_k.$$

En passant aux séries génératrices, on obtient, dans le cas non périodique

$$\sum R_n z^n = (1 - L_\infty) \sum z^n + \sum l_k z^k \sum R_n z^n,$$

ou, encore

$$R(z) = (1 - L_\infty)\frac{1}{1-z} + l(z)R(z) = \frac{1 - L_\infty}{(1-z)(1-l(z))}. \tag{2.65}$$

On remarque alors que

$$l(e^\lambda) = \sum_{i=0}^{\infty} l_i e^{\lambda i} = E(e^{\lambda Z^+}) = 1,$$

et par un raisonnement analogue au raisonnement fait pour la démonstration du théorème du renouvellement discret, on montre que e^λ est, pour $R(z)$, en dehors du pôle $z = 1$, le pôle de plus petit module; c'est un pôle simple puisque $l'(e^\lambda) = E(Z^+ e^{\lambda Z^+}) \neq 0$. Le résidu de $R(z)$ en 1 vaut $(1 - L_\infty)/(1 - l(1)) = 1$. Le résidu en e^λ vaut $\frac{1-L_\infty}{(1-e^\lambda)l'(e^\lambda)}$, et on obtient donc directement par le théorème de Cauchy

$$R_n = 1 - \frac{1 - L_\infty}{(e^\lambda - 1)E(Z^+ e^{\lambda Z^+})} e^{-\lambda n} + o(e^{-\lambda n}).$$

Dans le cas discret de pas δ, on fait apparaître le pôle $e^{\lambda\delta}$ et le facteur δ apparaît dans le calcul de $l'(1)$.
Remarque. Les calculs donnant les queues de distribution des variables Q et $M(n)$ se déduisent immédiatement, dans le cas discret, en remplaçant les intégrales par des sommes, de ceux effectués dans le cas continu; ils nous redonnent les formules 2.47 et 2.51.

2.3.3 Deux autres résultats asymptotiques

Karlin et Dembo donnent deux autres résultats asymptotiques intéressants.

- Un résultat sur la longueur des segments de score maximum.
- Un résultat utilisant une variable U_i couplée à la variable X_i qu'on peut appliquer pour trouver la distribution limite des lettres dans le segment de score maximum.

Ces résultats sont basés sur des propriétés des martingales de Wald simple ou double et sur le théorème d'échantillonnage optionnel.

Nous définissons les martingales.

Définition 13 (Martingales.). *Un processus stochastique* $\{X_n; n = 0, 1, \dots\}$ *est une martingale si* $\forall n$,

1. $E(|X_n|) < \infty$, *et*
2. $E(X_{n+1}|X_0, \dots, X_n) = X_n$.

Avec $\phi(t) = E(e^{tX})$ et $S_m = \sum_1^m X_i$, la martingale simple de Wald s'écrit

$$P_m = \frac{e^{tS_m}}{[\phi(t)]^m}. \tag{2.66}$$

En notant $W_k = \sum_{i=1}^k U_i$, et $\psi(t,\xi) = E(e^{tX+\xi U})$, la martingale de Wald double s'écrit

$$H_m = \frac{e^{tS_m+\xi W_m}}{[\psi(t,\xi)]^m}. \tag{2.67}$$

On montre facilement que P_m est une martingale; en effet, X_{m+1} est indépendant de $\{X_1,\dots,X_m\}$, et donc

$$\begin{aligned} E(P_{m+1}) &= \frac{E(e^{t(X_1+\cdots+X_m+X_{m+1})}|X_1,\dots X_m)}{E(e^{tX})^{m+1}} \\ &= \frac{E(e^{t(X_1+\cdots+X_m)}|X_1,\dots X_m)E(e^{tX_{m+1}})}{E(e^{tX})^{m+1}} \\ &= \frac{e^{t(X_1+\cdots+X_m)}}{E(e^{tX})^m} = P_m. \end{aligned}$$

nous allons utiliser la variante suivante du théorème d'échantillonnage optionnel qui s'applique aux martingales à variations bornées.

Théorème 14 (Théorème d'échantillonnage optionnel.). *Soit P_n une martingale définie sur une suite X_n (les X_i sont i.i.d, et P_n ne dépend que de $X_1,\dots,X_n$); si T est un temps d'arrêt relatif aux X_n, et si*

1. *$E(T) < \infty$,*
2. *et il existe une constante A telle que*

$$E(|P_{n+1} - P_n| \text{ si } X_0,\dots,X_n) \le A \qquad \text{pour } n < T,$$

alors, $E(P_T) = E(X_0)$.

Preuve. On peut consulter Karlin et Taylor [52] p. 260 pour la démonstration. □

Nous allons considérer le temps d'arrêt L correspondant à la première sortie de l' intervalle $[0,y]$, et nous ferons tendre y vers l'infini. Nous avons

$$L(y) = \{\min k,\ S_k > y \text{ ou } S_k < 0\}.$$

Nous définissons également un indicateur $I(y)$ disant si l'on sort de l'intervalle $[0,y]$ par le haut ($S_L > y$) ou par le bas ($S_L < 0$),

$$\begin{aligned} I(y) = 1 &\text{ si } S_L > y; \\ I(y) = 0 &\text{ si } S_L < 0; \end{aligned}$$

Nous allons considérer le temps d'atteinte ρ du record absolu S_R, et utiliser l'inégalité de Chebyshev.

Cette inégalité s'écrit, pour une fonction $f(x)$ positive et strictement croissante sur $[0, \infty]$,

$$\Pr(|X| \geq u) \leq \frac{E[f(X)]}{f(u)}.$$

Nous allons appliquer cette inégalité à la fonction $f(x) = e^{tx}, t \geq 0$, qui vérifie les conditions posées, et qui vérifie de plus $f(0) = 1$.

Nous avons alors, pour ρ,

$$\Pr(\rho = k) \leq \Pr(S_k > 0) \leq E(e^{tS_k}) = E(e^{tX})^k.$$

La fonction $E(e^{tX})$ a pour dérivée seconde $E(X^2 e^{tX}) > 0$; elle est donc convexe, et vaut 1 pour $t = 0$ et pour $t = \lambda$. Nous avons donc $E(e^{tX}) < 1$ pour $0 < t < \lambda$, et donc

$$\Pr(\rho = k) < e^{-bk}$$

pour un $b > 0$. Nous en déduisons que la queue de probabilité de ρ décroît exponentiellement vite.

Revenons à notre variable $L(y)$; nous avons

$$\rho > y \Rightarrow L > y,$$

ce qui s'écrit sous forme de probabilités

$$\Pr(L > y) \leq \Pr(\rho > y) = \mathcal{O}(e^{-y}).$$

Nous en déduisons immédiatement

$$E(L) = \int \Pr(L > x)dx < \infty.$$

En prenant comme temps d'arrêt le temps L, nous avons donc vérifié la première condition du théorème d'arrêt optionnel 14. La deuxième condition se déduit immédiatement du fait que nos variables X sont à support bornés. En appliquant ce théorème , nous obtenons donc

$$E(e^{tS_L - L\log(E[e^{tX}])}) \equiv E(e^{tX - \log(E[e^{tX}])}) = 1. \tag{2.68}$$

Nous énonçons alors le premier résultat asymptotique annoncé dans ce paragraphe.

Théorème 15. *Soit $I_k(y)$ l'indicateur $I(y)$ défini ci-dessus et relatif à la section de marche aléatoire partant du kième temps d'échelle décroissante q_k. Si $T_k(y)$ est le premier temps dans la succession des sections tel que $I_k(y) = 1$, alors*

$$\frac{L_k(y)}{y} \rightarrow \frac{1}{\omega} = \frac{1}{E(Xe^{\lambda X})},$$

où λ est la seule racine positive de l'équation $E(e^{tX}) = 1$.

Preuve. (Esquisse). On se référera à Karlin et Dembo [49] (lemmes 2 et 3) pour une démonstration détaillée.

Le lemme 2 de [49] donne $\Pr(I(y) = 1)e^{\lambda y} \geq \delta > 0$.

Dans le lemme 3 de [49], Karlin et Dembo utilisent les moments de la martingale en dérivant quatre fois l'équation 2.68, et en faisant $t = \lambda$. La première dérivation donne

$$E[(S_L - \omega L)e^{\lambda S_L}] = 0, \quad \omega = E(Xe^{\lambda X}).$$

On obtient, en décomposant suivant les valeurs de l'indicateur $I(y)$,

$$E(Le^{\lambda S_L}) = \mathcal{O}(y);$$

puis, on obtient une inégalité quadratique, qui donne

$$E(L^2 e^{\lambda S_L}) = \mathcal{O}(y^2).$$

On obtient ensuite

$$E\left[(S_L - \omega L)^4 e^{\lambda S_L}\right] = \mathcal{O}(y^2).$$

On en déduit, après utilisation du lemme 2, et en tenant compte du fait que $S_L - y$ est borné, conditionnellement à $I(y) = 1$,

$$E\left[\left(\frac{L(y)}{y} - \frac{1}{\omega}\right)^4 \mid I(y) = 1\right] = O\left(\frac{1}{y^2}\right) \qquad (y \to \infty).$$

□

En considérant à présent la martingale double H_m définie en 2.67, Karlin et Dembo obtiennent le second théorème annoncé.

Théorème 16. *Soit U_i une suite de variables aléatoires bornées et i.i.d telles que U_i peut dépendre de X_i, mais est indépendant de X_j, $j \neq i$. Soit $W_j = \sum_{i=1}^{j} U_i$ et*

$$W_k(y) = W_{T_k(y)} - W_{q_{k-1}},$$

où $T_k(y)$ est le temps de la première sortie de l'intervalle $[0, y]$ par le haut, obtenu lors de la kième excursion relative au temps d'échelle décroissante. Alors, en excluant un ensemble de mesure nulle,

$$\frac{W_k(y)}{L_k(y)} \to v \qquad p.s. \text{ quand } y \to \infty, \tag{2.69}$$

où $v = E(Ue^{\lambda X})$.

Preuve. La démonstration suit le déroulement de la démonstration faite pour le théorème 15, en dérivant l'identité

$$E(e^{tS_L + \xi W_L - L\log(E[e^{tX+\xi W}])}) \equiv 1,$$

et en faisant après chaque étape de dérivation $t = \lambda$ et $\xi = 0$. □

On peut appliquer ce résultat au cas où l'on considère un ensemble de lettres $\{a_i\}$, avec un score s_i et une probabilité p_i associés.

On considérera la lettre a_i, et on fera $U_j = 1$ si la lettre à la position j est a_i et $U_j = 0$ dans le cas contraire. La variable W_k du théorème 16 compte alors le nombre d'occurrence de la lettre a_i dans le segment $[q_k, T_k(y)]$; asymptotiquement, le théorème 16 donne la fréquence d'apparition de la lettre a_i dans le segment de score maximum, qu'on peut exprimer par le corollaire suivant.

Corollaire 17. *Soit un alphabet $a_1, \ldots, a_r$, et un score s_i et une probabilité p_i associés à chaque lettre a_i. Asymptotiquement, la fréquence d'apparition f_i de la lettre a_i dans le segment de score maximal vaut $f_i = p_i e^{\lambda s_i}$.*

Ce corollaire s'applique immédiatement au cas de l'alignement de deux séquences, où les lettres r_i d'une requête et c_j d'une cible ont les probabilités d'apparition respectives q_i et p_j, et où le score d'appariement de deux lettres r_i et c_j vaut s_{ij}. Nous obtenons alors la distribution limite $p_j^\star$ dans le segment de la cible correspondant à l'alignement optimal comme

$$p_j^\star = \sum_i q_i p_j e^{\lambda s_{ij}},$$

où λ est donné par $\sum_{i,j} q_i p_j e^{\lambda s_{ij}} = 1$.

En particulier, dans le cas où la séquence requête est composée d'une seule lettre i_0, $(q_{i_0} = 1;\ q_i = 0, i \neq i_0)$, nous obtenons

$$p_j^\star = p_j e^{\lambda s_{i_0 j}}. \tag{2.70}$$

2.3.4 Rapidité de la convergence de la série intervenant dans le calcul de la constante K

La constante K de Karlin est donnée par la formule 2.32

$$K = \frac{\exp\left\{-2\sum_{k=1}^\infty \frac{1}{k}\left(E(e^{\lambda S_k}; S_k < 0) + \Pr(S_k \geq 0)\right)\right\}}{\lambda E\left(X e^{\lambda X}\right)}.$$

Nous allons nous intéresser à la rapidité de convergence de la série apparaissant comme exposant au numérateur de cette expression, qui peut s'écrire

$$S = \sum_{k=1}^\infty \frac{1}{k}\left(E(e^{\lambda S_k}; S_k < 0) + \Pr(S_k \geq 0)\right) = \sum_{k=1}^\infty \frac{1}{k} E(e^{\lambda S_k^-}), \qquad \text{avec } S_k^- = \min(S_k, 0). \tag{2.71}$$

Nous allons montrer que cette série converge exponentiellement (ce résultat de vitesse de convergence est annoncé sans démonstration par Karlin et Dembo [50]).

Nous utiliserons le théorème sur les grandes déviations suivant (voir Durrett p.70-73)

Théorème 18 (Théorème des grandes déviations.). *Soit $X_1, X_2, \ldots$ une suite de variables aléatoires indépendantes et identiquement distribuées, telle qu'il existe un $\theta > 0$ pour lequel*

$$\phi(\theta) = E(e^{\theta X_i}) < \infty. \tag{2.72}$$

Alors, si $\kappa(t) = \log \phi(t)$, $S_n = \sum_{k=1}^{n} X_k$, $\mu = E(X_i)$, *et si* a *est une constante strictement supérieure à* μ, $a > \mu$, *on a, pour* n *assez grand,*

$$P(S_n \geq na) \leq e^{-n\{at-\kappa(t)\}}, \tag{2.73}$$

et on a, pour $t > 0$ *petit,* $at - \kappa(t) > 0$.

On peut réécrire l'équation 2.73 de ce théorème en disant qu'il existe $\epsilon > 0$ tel que

$$P(S_n \geq na) \leq e^{-n\epsilon}. \tag{2.74}$$

Nous allons alors démontrer:

Théorème 19. *La série permettant de calculer la constante* K *de Karlin converge exponentiellement vite.*

Preuve. Considérons le terme $E(e^{\lambda S_k^-})$ de l'équation 2.71; nous pouvons l'écrire sous la forme

$$E(e^{\lambda S_k^-}) = \int_{-\infty}^{\alpha} e^{\lambda S_k^-} d\mathcal{P} + \int_{\alpha}^{0} e^{\lambda S_k^-} d\mathcal{P} + \int_{0}^{-\infty} e^{\lambda S_k^-} d\mathcal{P}, \tag{2.75}$$

où $\mathcal{P}$ est la probabilité induite par la suite X_i pour la variable aléatoire S_k, et où $\alpha = -i\log(k)/\lambda < 0$ est solution en x de l'équation $e^{\lambda x} = 1/k^i$, $\quad i \in \mathbb{N}^+$. Remarquons que

- pour $S_k^- \in]-\infty, \alpha]$, on a $e^{\lambda S_k^-} < e^{\lambda\alpha} = \frac{1}{k^i}$;
- pour $S_k^- \in [\alpha, 0]$, on a $e^{\lambda S_k^-} < 1$;
- et pour $S_k^- \geq 0$, on a $S_k^- = 0$, et donc $e^{\lambda S_k^-} = 1$;

ceci nous donne

$$E(e^{\lambda S_k^-}) < \int_{-\infty}^{\alpha} \frac{1}{k^i} d\mathcal{P} + \int_{\alpha}^{\infty} 1 d\mathcal{P} < \frac{1}{k^i} + P(S_k^- \geq \alpha) = \frac{1}{k^i} + P(S_k \geq \alpha),$$

puisque l'on a $\alpha < 0$. C'est pour borner la probabilité $P(S_k \geq \alpha)$ que nous allons utiliser le théorème des grandes déviations.

Remarquons que le théorème des grandes déviations ne fait pas d'hypothèses sur le signe de l'espérance μ de X_i; l'espérance de X_i étant négative pour toute nos démonstrations concernant les résultats de Karlin, nous avons $\mu < 0$, et la fonction $g(x) = \frac{\mu}{2}x + \frac{i\log x}{\lambda}$ décroît jusqu'à $-\infty$ quand x croît de $-2i/\lambda\mu$ à $+\infty$; il existe donc $x_0 > 0$ tel que $\forall x > x_0$ on ait $g(x) < 0$, et, de manière équivalente, tel que, $\forall k > x_0$, on ait $k\frac{\mu}{2} < -\frac{i\log k}{\lambda}$.

Nous avons donc, pour $k > x_0$,

$$P(S_k \geq \alpha) = P\left(S_k \geq -\frac{i\log k}{\lambda}\right) < P\left(S_k > k\frac{\mu}{2}\right) < e^{-k\epsilon},$$

en appliquant le théorème des grandes déviations (théorème 18) avec $\mu < a = \mu/2 < 0$, et sous la forme de l'équation 2.74. Nous avons obtenu

$$E(e^{\lambda S_k^-}) < \frac{1}{k^i} + e^{-k\epsilon}. \tag{2.76}$$

Cette dernière inégalité est vérifiée pour chaque valeur de i pour k assez grand. La convergence de la série est donc exponentielle.

$\square$

2.3.5 Probabilité d'obtenir plusieurs alignements de scores supérieurs à un score donné

Nous évaluons dans ce paragraphe la probabilité d'obtenir i alignements de score supérieur à un score y. Nous suivons pour ceci Karlin et Altschul [47] pour obtenir une formule qui est une généralisation de la formule 2.30 . Avec λ et K donnés par les formules 2.31 et 2.32, nous normalisons les scores en écrivant

$$S' = \lambda S - \log(Kn).$$

La formule 2.30 s'écrit alors

$$P(S' \geq x) \approx 1 - e^{-e^{-x}}. \tag{2.77}$$

Nous avons alors.

Lemme 20. *La probabilité d'obtenir r sections de la marche aléatoire d'incrément X de scores normalisés respectifs $S'_1, S'_2, \ldots, S'_r$, tels que $S'_1 \geq S'_2 \geq \cdots \geq S'_r \geq x$ est bien approchée par l'équation*

$$\Pr(S'_r \geq x) \approx 1 - e^{-e^{-x}} \sum_{k=0}^{r-1} \frac{e^{-kx}}{k!}. \tag{2.78}$$

Preuve. La densité $f(x)$ de la variable aléatoire du score S' est approchée pour S' assez grand par $f(x) \approx \exp(e^{-x})e^{-x} = \exp(e^{-x} - x)$. La densité conjointe des $S'_1, S'_2, \ldots, S'_r$ est alors approchée par

$$f(x_1, \ldots, x_r) \approx \exp\left(-e^{-x_r} - \sum_{k=1}^{r} x_k\right),$$

où on a $x_1 \geq x_2 \geq \ldots x_r$. On peut alors évaluer par intégration la queue de distribution de S'_r. En effectuant une intégration par partie, nous écrivons

$$\begin{aligned}
\Pr(S'_r \geq x) &= \int_{x_1 \geq x_2 \geq \cdots \geq x_r \geq x} f(x_1, \ldots x_r) dx_1 dx_2 \ldots dx_r & (2.79)\\
&\approx \int_x^\infty dx_r \int_{x_r}^\infty dx_{r-1} \cdots \int_{x_2}^\infty \exp\left(-e^{-x_r} - \sum_{k=1}^{r} x_k\right) dx_1 & (2.80)\\
&= \int_x^\infty \frac{e^{-e^{-t}} e^{-rt}}{(r-1)!} dt & (2.81)\\
&= \left[e^{-e^{-t}} \frac{e^{-(r-1)t}}{(r-1)!}\right]_x^\infty - \int_x^\infty \frac{e^{-e^{-t}} e^{-(r-1)t}}{(r-2)!} & (2.82)\\
&= 1 - e^{-e^{-x}} \sum_{k=0}^{r-1} \frac{e^{-kx}}{k!}. & (2.83)
\end{aligned}$$

Ceci nous donne le résultat annoncé. □

Chapitre 3

Compléments mathématiques pour le modèle de Karlin

Ce chapitre peut être sauté en première lecture. Il est placé ici pour la complétude de la partie analytique. Les démonstrations des lemmes 22 et 23 et celle du théorème 30 sont originales.

3.1 Analycité de ϕ_+, ϕ_- et H dans la décomposition de Wiener-Hopf

Nous allons démontrer l'analycité des fonctions ϕ^+ et ϕ^-, puis celle de H, fonctions utilisées dans la décomposition de Wiener-Hopf.

3.1.1 Continuité et analycité de ϕ_+ et de ϕ_-.

Nous allons démontrer l'analycité de ϕ_+; la démonstration de l'analycité de $\phi-$ est identique. Nous pouvons écrire

$$\phi_+(u, \xi) = \frac{1}{1 - g(u, \xi)},$$

avec $g(\xi) = E\left(u^{\zeta^+} e^{-\xi S_{\zeta+}}\right)$; la continuité et l'analycité de $g(\xi)$ entraîneront celles de ϕ_+. Nous démontrerons la continuité et l'analycité de $g(\xi)$ en deux lemmes séparés; nous ferons usage pour la démonstration de l'analycité de $g(\xi)$ d'un théorème de Khoan [54] relatif à la dérivation sous le signe intégral que nous énonçons.

Théorème 21. *Soit μ une mesure sur un espace topologique séparé et localement compact Ω, f une fonction μ intégrable, et $F = \int_\Omega f(x, \xi) d\mu(x)$. Supposons que Ξ soit un ouvert de $\mathbb{R}$ ou de $\mathbb{C}$ et que les conditions suivantes soient réalisées:*

1. *pour μ-presque tout x, la fonction $\xi \to f(x, \xi)$ est continûment dérivable sur Ξ;*

2. *pour tout $\xi \in \Xi$, la fonction $x \to \frac{\partial f}{\partial \xi}(x, \xi)$ est μ-mesurable;*

3. *il existe une fonction μ-intégrable, h, telle que*

$$\left|\frac{\partial f}{\partial \xi}(x,\xi)\right| \leq h(x)$$

pour μ-presque tout $x \in \Omega$.

Alors, F est continûment dérivable sur Ξ et

$$\frac{\partial F}{\partial \xi}(\xi) = \int_\Omega \frac{\partial f}{\partial \xi}(x,\xi) d\mu(x).$$

Preuve. La démonstration utilise le théorème des accroissements finis (voir Gourdon [32]) et le théorème de la convergence dominée de Lebesgue (voir Deheuvels [23] ou Durrett [26] et Dieudonné, Élements d'Analyse [25]); On peut aussi consulter Whittaker et Watson [88] (p.92) qui ne fait pas intervenir la théorie de la mesure, mais il faut que f soit une fonction continue de x. □

Remarque. Dans les cas que nous considérons, nous avons $\Omega = \mathbb{R}$, ou $\Omega = \mathbb{Z}$, qui sont séparés et localement compacts.

Nous avons donc à démontrer le lemme suivant

Lemme 22. *Dans les conditions énoncées au théorème 2 la fonction $g(\xi) = E\left(u^{\zeta^+} e^{-\xi S_{\zeta^+}}\right)$ est continue.*

Preuve. Nous avons

$$\begin{aligned} |g(\xi + d\xi) - g(\xi)| &= \left|E\left(u^{\zeta^+} e^{-(\xi+d\xi)S_{\zeta^+}}\right) - E\left(u^{\zeta^+} e^{-\xi S_{\zeta^+}}\right)\right| \\ &= \left|E\left(u^{\zeta^+} e^{-\xi S_{\zeta^+}}\left(e^{-d\xi S_{\zeta^+}} - 1\right)\right)\right| \leq E\left|u^{\zeta^+} e^{-\xi S_{\zeta^+}}\left(e^{-d\xi S_{\zeta^+}} - 1\right)\right|. \end{aligned} \tag{3.1}$$

Nous pouvons appliquer l'inégalité de Cauchy-Schwartz $E|XY| \leq \sqrt{E|X|^2}\sqrt{E(|Y|^2}$ à l'équation 3.1, ce qui nous donne

$$|g(\xi + d\xi) - g(\xi)| \leq \sqrt{E\left|u^{\zeta^+} e^{-\xi S_{\zeta^+}}\right|^2}\sqrt{E\left|e^{-d\xi S_{\zeta^+}} - 1\right|^2} \leq \sqrt{E\left(\left|e^{-d\xi S_{\zeta^+}} - 1\right|^2\right)},$$

puisque $|u| \leq 1$, $\Re\xi \geq 0$ et $S_{\zeta^+} \geq 0$. Pour z complexe, nous avons

$$|e^z - 1| = \left|\sum_{n=1}^{\infty} \frac{z^n}{n!}\right| \leq \sum_{n=1}^{\infty} \frac{|z|^n}{n!} = e^{|z|} - 1,$$

et d'autre part la fonction de x réel $t(x) = 2x - e^x + 1$ est croissante sur $[0, \log(2)]$, et vaut 0 pour $x = 0$; donc, pour $|z| < \log 2$, nous avons $|e^z - 1| \leq e^{|z|} - 1 < 2|z|$; comme nous avons supposé X à support compact, il existe une constante $A > 1$ telle que $|S_{\zeta^+}| < \max|X| < A$; par conséquent, pour $|d\xi| < \frac{\log(2)}{A}$, nous pouvons majorer $|g(\xi + d\xi) - g(\xi)|$ par

$$|g(\xi + d\xi) - g(\xi)| \leq \sqrt{E\left(2|d\xi S_{\zeta^+}|\right)^2} \leq 2A|d\xi|;$$

La continuité de $g(\xi)$ est démontrée. □

Lemme 23. *Dans les conditions énoncées au théorème 2 la fonction* $g(\xi) = E\left(u^{\zeta^+} e^{-\xi S_{\zeta^+}}\right)$ *est analytique.*

Preuve. Nous voulons démontrer que

$$\phi_+(u,\xi) = \frac{1}{1 - E\left(u^{\zeta^+} e^{-\xi S_{\zeta^+}}\right)}.$$

est analytique.

Une condition nécessaire et suffisante d'analycité d'une fonction est qu'elle soit continûment dérivable (voir Churchill [16]). C'est ce que nous allons démontrer pour la fonction $g(\xi) = E\left(u^{\zeta^+} e^{-\xi S_{\zeta^+}}\right)$ en montrant qu'elle vérifie les conditions du théorème 3.1 , ce qui entraînera l'analycité de la fonction ψ_+. Nous suivons en partie Feller [27] pour cette démonstration.

Considérons la distribution conjointe de ζ^+ et de S_{ζ^+}

$$P\{\zeta^+ = n, S_{\zeta^+} \in I\} = H_n\{I\},$$

pour tout I Borélien de $\mathbb{R}^+$; nous avons

$$H_n\{I\} = P\{S_1 \leq 0, \ldots, S_{n-1} \leq 0, S_n \in I\}, \tag{3.2}$$

avec $H_0\{I\} = 0$. Nous pouvons écrire $g(\xi)$ sous la forme

$$g(\xi) = \sum_{n=1}^{\infty} u^n \int_0^{\infty} e^{-\xi x} H_n(dx) = \sum_{n=1}^{\infty} \int_0^{\infty} e^{-\xi x} u^n H_n(dx); \tag{3.3}$$

l'intégrande étant positive, nous pouvons intervertir les signes somme et intégrale, ce qui nous donne

$$g(\xi) = \int_0^{\infty} \sum_{n=1}^{\infty} e^{-\xi x} u^n H_n(dx) = \int_0^{\infty} e^{-\xi x} \sum_{n=1}^{\infty} u^n H_n(dx);$$

sur $\mathbb{R}^+$ la mesure $\mu(dx) = \sum_{n=1}^{\infty} u^n H_n(dx)$ vérifie

$$\mu[0, +\infty] = \sum u^n P(\zeta^+ = n) < \frac{P(\zeta^+ < \infty)}{1-u},$$

et donc $\mu[0, +\infty] < \infty$ pour $u < 1$; nous devons à présent vérifier les conditions du théorème 21 avec $f(x,\xi) = e^{-x\xi}$.

- Condition 1: elle est immédiatement vérifiée.
- Condition 2: considérons l'intégrale

$$T(\xi) = \int_0^{\infty} \frac{\partial f}{\partial \xi}(x,\xi)\mu(dx) = \int_0^{\infty} x e^{-\xi x} \sum_{n=1}^{\infty} u^n H_n(dx);$$

nous pouvons majorer $T(\xi)$ par

$$T(\xi) \leq \int_0^{\infty} x e^{-\xi x} \frac{1}{1-u} dx = \frac{1}{\xi^2} \frac{1}{1-u}.$$

L'intégrande est positive et $T(\xi)$ est bornée; la fonction $x \to \frac{\partial f}{\partial \xi}(x,\xi)$ est donc μ-mesurable;

- Condition 3: elle est immédiatement vérifiée, puisque $\frac{\partial f}{\partial \xi}(x,\xi)$ est une fonction à valeurs partout positives ou nulles, et qu'on peut donc utiliser le résultat obtenu pour la démonstration de la condition 2.

Toutes les conditions d'application du théorème 21 sont vérifiées; la fonction $g(\xi)$ est analytique pour $u < 1$, et il s'en suit que ϕ_+ l'est aussi, puisque $|g(u,\xi)| < 1$. $\square$

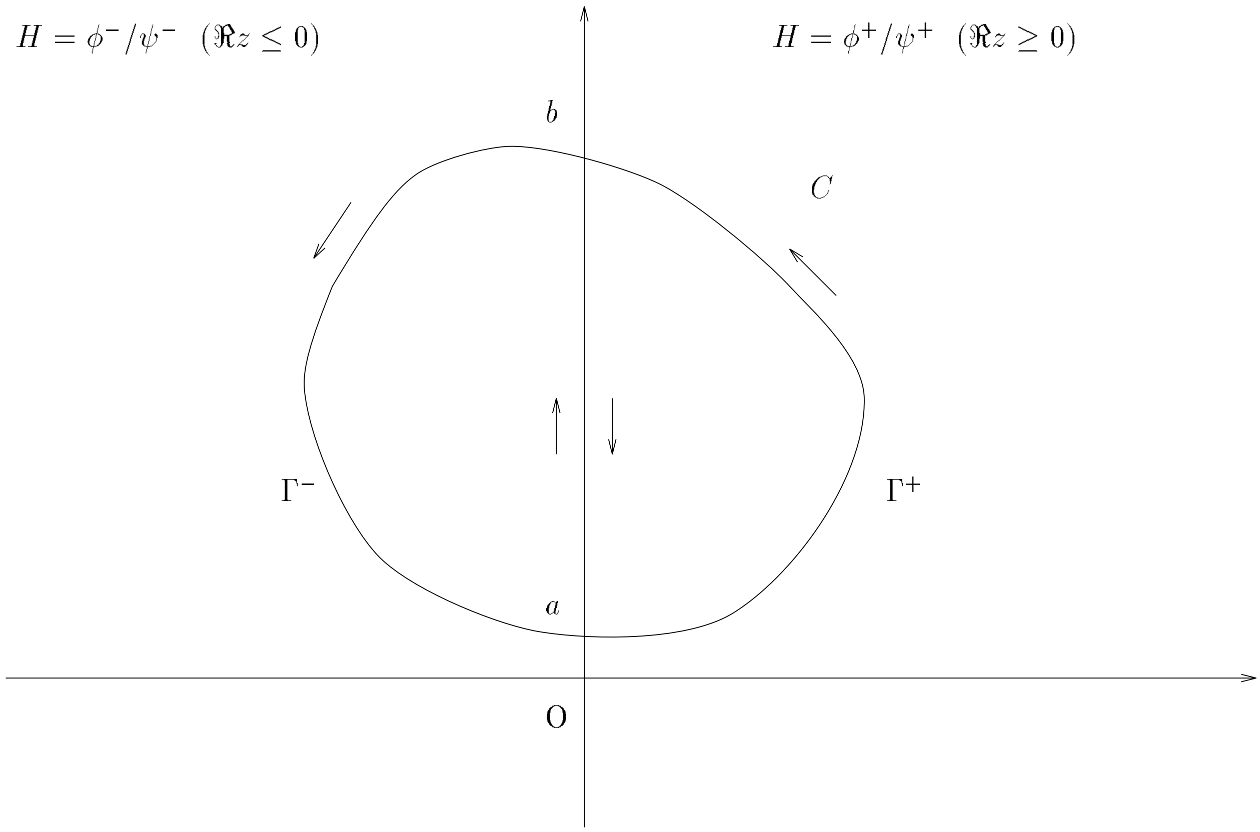


Figure 3.1: La fonction H, et les contours d'intégration C, Γ^- et Γ^+

3.1.2 Continuité et analycité de H.

L'équation 2.19 s'écrit

$$H(\xi) = \frac{\psi_+(u,\xi)}{\phi_+(u,\xi)} = \frac{\phi_-(u,\xi)}{\psi_-(u,\xi)}, \quad |u| < 1, \Re\xi = 0.$$

La continuité de H découle immédiatement de celles de ϕ_+ et ψ_+ d'une part, et de ϕ_- et ψ_- d'autre part, dans leurs demi-plans respectifs ($\{\Re\xi \leq 0\}, \{\Re\xi \geq 0\}$) et de cette dernière équation pour $\{\Re\xi = 0\}$.

Pour démontrer l'analycité de H, nous allons utiliser le théorème de Morera (voir Bak et Newman [8] ou Churchill [16]), que nous énonçons.

Théorème 24 (Morera). *Si une fonction F est continue dans un domaine simplement connexe D et si, pour tout contour fermé C intérieur à D, on a*

$$\int_C f(z)dz = 0, \tag{3.4}$$

alors f est analytique sur D tout entier.

Nous avons alors:

Lemme 25. *La fonction H est analytique sur $\mathbb{C}$ tout entier.*

Preuve. Nous savons que H est continue sur $\mathbb{C}$ tout entier, et qu'elle est analytique d'une part sur $\{\Re\xi \leq 0\}$, puisque les fonctions ϕ_- le sont dans ce domaine, et d'autre part sur $\{\Re\xi \geq 0\}$, puisque les fonctions ϕ_- le sont dans cet autre domaine. Soit alors un contour quelconque C du plan complexe; nous avons les deux cas suivants:

1. C ne coupe pas l'axe imaginaire, et dans ce cas, C est entièrement contenu dans l'un des deux demi-plans $\{\Re\xi \leq 0\}$ et $\{\Re\xi \geq 0\}$; dans ce cas, H étant analytique dans chacun de ces demi-plans, l'intégrale $\int_C H(z)dz$ vaut 0, par application du théorème de Cauchy.

2. C coupe l'axe imaginaire en deux points a et b (on supposera que l'ordonnée de b est supérieure à celle de a), séparant C en deux courbes C^- et C^+ (voir la figure 3.1, $C^- = C \bigcap \{\Re\xi \leq 0\}$, $C^+ = C \bigcap \{\Re\xi \geq 0\}$); les contours fermés Γ^- obtenus en complétant C^- parcouru dans le sens trigonométrique par l'arc $\vec{ab}$, et Γ^+ obtenus en complétant C^+ parcouru dans le même sens par l'arc $\vec{ba}$ sont chacun contenus dans un des demi-plans $\{\Re\xi \leq 0\}$ et $\{\Re\xi \geq 0\}$. Nous avons donc

$$\int_C H(z)dz = \int_{\Gamma^-} H(z)dz + \int_{\Gamma^+} H(z)dz = 0,$$

   en appliquant le théorème de Cauchy à la fonction $f(z)$ sur chacun des contours Γ^- et Γ^+.

Par conséquent, sur tout contour fermé C contenu dans $\mathbb{C}$, l'intégrale de $H(z)$ vaut zéro; nous pouvons appliquer le théorème de Morera qui nous dit que $H(z)$ est partout analytique sur $\mathbb{C}$. $\square$

3.2 Propriétés de la loi de la somme $Z^+ = S_{\zeta^+}$

3.2.1 Continuité

Nous reprenons les conditions posées pour la démonstration des formules de Karlin-Iglehart. L'incrément X de notre marche aléatoire est tel que $E(X) < 0$, et que $\{P(X > 0) > 0\}$, et d'autre part X est à support compact. Nous effectuons la démonstration dans le cas où la distribution de X est absolument continue, c'est à dire qu'il existe une fonction F telle que $F(x) = \int_{-\infty}^{x} f(t)dt$. Nous supposerons en outre que f est continue par morceaux, ce qui entraîne, puisque X est à support borné, qu'il existe $M < \infty$ tel que, $\{\forall x, F(x+dx) - F(x) < Mdx\}$ (F est Lipschitzienne - voir Gourdon [32] ou Titchmarsh [83]).

Nous noterons $\phi(t) = E(e^{itX}) = \int_{-\infty}^{+\infty} e^{itx} f(x)dx$ la fonction caractéristique de X.

Soit $L_{Z^+}(x) = P(Z^+ < x)$ la loi de Z^+; nous allons démontrer que L_{Z^+}est Lipschitzienne et donc continue. La démonstration utilise un théorème local limite de convergence vers la distribution normale $\mathcal{N}$.

Nous aurons besoin des théorèmes ou lemmes généraux suivants concernant les fonctions caractéristiques.

Théorème 26 (Riemann-Lebesgue). *Si $f(x)$ est intégrable et $\phi(t) = \int_{-\infty}^{+\infty} e^{itx} f(x)dx$, alors $\phi(t) \to 0$ quand $t \to \pm\infty$.*

Preuve. On peut trouver ce théorème dans Feller [27] □

Lemme 27. *Il existe seulement trois possibilités pour la fonction caractéristique:*

1. *$|\phi(t)| < 1$ pout tout $t \neq 0$.*
2. *$|\phi(\tau)| = 1$ et $|\phi(t)| < 1$ pour $0 < t < \tau$. Dans ce cas $|\phi|$ a période τ et il existe un nombre réel b tel que $F(x+b)$ est arithmétique de pas $h = 2\pi/\lambda$.*
3. *$|\phi(t)| = 1$ pout tout t. Dans ce cas $\phi(t) = e^{ibt}$ et F est concentrée au point b.*

Preuve. (Esquisse.) Nous suivons Feller. Supposons $\phi(\tau) = \int_{\infty}^{\infty} e^{i\tau x} f(x)dx = 1$; dans ce cas l'espérance de la fonction non négative $1 - \cos \tau x$ vaut 0; ceci implique que pour tout intervalle $[a, b]$ ne contenant pas un nombre mh avec $h = \frac{2\pi}{\tau}$, et $m \in \mathbb{Z}$, nous avons $\int_a^b f(x)dx = 0$. La distribution F est donc concentrée aux points mh.

Dans le cas où $|\phi(\tau)| = 1$, il existe b réel tel que $\phi(\tau) = e^{itb}$. La variable $X - b$ a pour fonction caractéristique $\phi(t)e^{-itb}$ qui vaut 1 en $t = \tau$, et le résultat précédant s'applique à la variable translatée. □

Il faut remarquer que cette démonstration ne dit rien sur $\limsup |\phi(t)|$ quand $t \to \infty$.

Durrett [26] (p. 134) donne l'exemple suivant de distribution définie sur $\mathbb{Q}$ qui incite à la prudence en ce qui concerne le comportement limite de la fonction caractéristique:

Exemple 28. *Soit $q_1, q_2, \ldots$ une énumération des rationnels positifs tels que $q_n \leq n$; nous considérons la distribution $P(X = q_n) = P(X = -q_n) = 1/2^{n+1}$; nous avons $E(X) = 0$, $E(X^2) < \sum n^2/2^n < \infty$, mais $\phi(t) = E(e^{itX}) = \sum_{n=1}^{\infty} \cos q_n/2^n$ et $\limsup_{t\to\infty} \phi(t) = 1$.*

Nous allons utiliser également un théorème local limite de convergence (voir Durrett p. 134), de la famille des théorèmes de Berry-Esseen (le théorème de Berry-Esseen est un théorème de vitesse de convergence faible, en loi ou fonction de répartition et non en densité).

Théorème 29. *Soit $X_1, X_2, \ldots$ une suite de variables aléatoires indépendantes et identiquement distribuées, avec $EX_i = 0$ et $0 < EX_i^2 = \sigma^2 < \infty$, les X_i ayant une fonction caractéristique commune telle que $|\phi(t)| < 1$ pour tout $t \neq 0$. Soit $S_n = X_1 + \cdots + X_n$ et $\mathbf{n}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-x^2/2\sigma^2}$;*

alors, si $x_n/\sqrt{n} \to x$ et $a < b$, on a

$$\sqrt{n} P(S_n \in [x_n + a, x_n + b]) \to (b - a)\mathbf{n}(x)$$

Compte tenu de nos hypothèses sur la variable X, de densité continue par morceaux et à support compact, nous pouvons appliquer le théorème de Riemann-Lebesgue et le lemme 27, qui nous donnent les hypothèses du théorème 29 et en particulier $|\phi(t)| < 1, \ \forall t \neq 0$.

Nous arrivons donc à un théorème de continuité pour la loi de Z^+:

Théorème 30. *Dans les conditions énoncées au début du paragraphe (X est à support compact et a une densité continue par morceaux), la loi L_{Z^+} de Z^+ est Lipschitzienne, et donc continue.*

Preuve. Nous allons étudier la variation de la loi $L_{Z^+}(y)$ pour de petites variations de y. Nous avons

$$P(Z^+ \in [y, y + dy]) \leq \sum_{n=1}^{\infty} P(\zeta^+ = n) P(S_n \in [y, y + dy]). \tag{3.5}$$

Nous allons utiliser la convergence locale limite de la loi des S_n vers la loi normale. Le théorème 29 s'écrit:

$$\forall \epsilon, \text{ quand } x_n/\sqrt{n} \to x, \exists N, \forall n > N, \quad |\sqrt{n} P(S_n \in [x_n + a, x_n + b]) - (b - a)\mathbf{n}(x)| < \epsilon.$$

Choisissons $\epsilon = \alpha dy$ avec $\alpha < 1$, $a = -2dy$, et $b = 2dy$, et faisons tendre x_n vers $x\sqrt{n} = y$ de sorte que $|x_n - x\sqrt{n}| = |x_n - y| < dy$; nous avons alors

$$x_n + a = x_n - 2dy < y - dy < x_n < y + dy < x_n + 2dy = x_n + b, \quad \text{et}$$

$$\sqrt{n} P(S_n \in [x_n + a, x_n + b]) < (b - a)\mathbf{n}(x) + \epsilon < (b - a)(\frac{1}{\sigma\sqrt{2\pi}} + \alpha) < A_1 dy,$$

avec $A_1 < \infty$. En appliquant le théorème 29, nous avons donc, pour $n > N$

$$\sqrt{n} P(S_n \in [y, y + dy]) < \sqrt{n} P(S_n \in [x_n + a, x_n + b] < A_1 dy, \tag{3.6}$$

ou encore

$$P(S_n \in [y, y + dy]) < \frac{A_1}{\sqrt{n}} dy, \tag{3.7}$$

Nous pouvons réécrire l'équation 3.5, avec N tel que l'équation 3.7 soit valide, sous la forme

$$P(Z^+ \in [y, y+dy]) \leq \sum_{n=1}^{N} P(\zeta^+ = n) P(S_n \in [y, y+dy]) + \sum_{n=N}^{\infty} P(\zeta^+ = n) P(S_n \in [y, y+dy]) \tag{3.8}$$

$$< \sum_{n=1}^{N} P(S_n \in [y, y+dy]) + \sum_{n=N}^{\infty} \frac{E(\zeta^+)}{n} P(S_n \in [y, y+dy]), \tag{3.9}$$

le dernier terme étant obtenu en appliquant l'inégalité de Markov à la variable positive ζ^+.

Nous avons supposé que la densité f de X était continue par morceaux, et à support compact. Par conséquent f est bornée; soit $M < \infty$ bornant $f(x)$.

Soit m le minimum de X; nous pouvons faire la translation $X = m + X'$, et X' est à support compact positif. Soit $S'_n = \sum_{j=1}^{n} X' = S_n + nm$; les X'_j comme les X_j étant indépendants, nous avons

$$\begin{aligned}
P(S_n \in [y, y+dy]) &= P(S'_n = S_n + nm \in [y' = y + nm, y' + dy' = y + nm + dy]) \\
&= dP(S_n < y') \\
&= d\left(\int_{x_1=0}^{y'} f(x_1)dx_1 \int_{x_2=0}^{y'-x_1} f(x_2)dx_2 \cdots \int_{x_n=0}^{y'-x_1\cdots-x_{n-1}} f(x_n)dx_n\right) \\
&< M^n d\left(\int_{x_1=0}^{y'} dx_1 \int_{x_2=0}^{y'-x_1} dx_2 \cdots \int_{x_n=0}^{y'-x_1\cdots-x_{n-1}} dx_n\right) \\
&= M^n d(\frac{y'^n}{n!}) = \frac{M^n y'^{n-1}}{(n-1)!} dy' = \frac{M^n (y+mn)^{n-1}}{(n-1)!} dy;
\end{aligned}$$

nous en déduisons que, pour $n \leq N$, X étant bornée, de même que S_n et S'_n, il existe $A_2 < \infty$ tel que $P(S_n \in [y, y+dy]) < A_2 dy$, pour tout y.

Nous avons donc obtenue une double majoration de la décomposition de l'équation 3.8, et donc nous pouvons écrire

$$P(Z^+ \in [y, y+dy]) < NA_2 dy + \sum_{n=N}^{\infty} \frac{A_1 E(\zeta^+) dy}{n\sqrt{n}} = A_3 dy \quad \text{avec} \quad A_3 < \infty.$$

Nous en concluons que la loi (au sens de la fonction de répartition) de Z^+ est lipschitzienne, et donc continue. □

3.2.2 Intégrabilité Riemannienne directe

La démonstration de la formule de Karlin-Iglehart nous conduit à rechercher la valeur asymptotique à l'infini d'une fonction inconnue H donnée par une équation du renouveau du type $H = h + H * F$, où $*$ indique l'opération de convolution de deux distributions. Ce résultat se nomme théorème du renouvellement. Nous allons maintenant définir ce qu'est l'intégrabilité Riemannienne directe, qui est utilisée dans les conditions de ce théorème .

Définition 31. *Étant donné un pas δ, soient les sommes de Riemann supérieures et inférieures approximant l'intégrale d'une fonction f définies comme*

$$I^\delta = \sum_{i=0}^{\infty} \delta \ \sup\{f(x) : x \in [i\delta, (i+1)\delta)\} \quad et$$
$$I_\delta = \sum_{i=0}^{\infty} \delta \ \inf\{f(x) : x \in [i\delta, (i+1)\delta)\};$$

la fonction f est alors dite directement Riemann intégrable si I^δ et I_δ approchent la même limite finie quand $\delta \to 0$.

On peut consulter Durrett et Feller pour cette définition rendue nécessaire pour les intégrations sur $[0, \infty]$ par le fait qu'il existe des fonctions continues et Riemann intégrables qui ont des variations fortes, et peuvent même tendre à ne pas être bornées (voir Feller); pour de telles fonctions h, il n'existe pas de solutions à l'équation de renouvellement $H = h + H * U$ dont nous parlerons au prochain paragraphe.

Dans le cas où l'intégration porte sur un intervalle borné, l'intégrabilité Riemannienne directe est équivalente à l'intégrabilité Riemannienne classique, et, en particulier, les fonctions continues par morceaux, qui sont Riemann intégrables, sont aussi directement Riemann intégrables. Le nom d'intégrabilité directe provient du fait que l'on considère directement des sommes sur $[0, +\infty]$, et non la limite d'une intégrale définie sur un segment fini $[0, a]$, quand on fait tendre a vers $+\infty$. Nous avons démontré dans les conditions du théorème 30 que la loi de Z^+ est continue; elle est à support compact, puisque le support de Z^+ est contenu dans celui de X qui a été supposé à support compact. La fonction $z(y) = L_{Z^+}(\infty) - L_{Z^+}(y))e^{\lambda y}$ définie sur $[0, \infty]$ a même support compact que Z^+, et elle est continue comme $L_{Z^+}(y)$. Nous avons donc:

Théorème 32. *Dans les conditions du théorème 30 (X de densité continue par morceaux et à support compact, et $E(\zeta^+) < \infty$), la loi L_{Z^+}de Z^+ et la fonction $z(y) = L_{Z^+}(\infty) - L_{Z^+}(y))e^{\lambda y}$ sont directement Riemann intégrables.*

Le lemme suivant donne une condition nécessaire et suffisante d'intégrabilité Riemannienne directe pour les fonctions positives; nous le donnons pour compléter ce paragraphe sur l'intégrabilité Riemannienne directe.

Lemme 33. *Si $h \geq 0$ est continue, h est directement Riemann intégrable si et seulement si la somme de Riemann supérieure I^δ est bornée pour un δ_0.*

Preuve. Si la fonction h est directement Riemann intégrable, il existe par définition un δ_0 tel que $I_{\delta_0} < \infty$; la condition est donc nécessaire. Elle est suffisante; en effet, supposons que, pour δ_0, nous ayons

$$I^{\delta_0} = \sum_{i=0}^{\infty} \delta \ \sup\{f(x) : x \in [i\delta, (i+1)\delta[\} < \infty.$$

La série donnant I^{δ_0} est bornée et à terme positif; pour tout ϵ, il existe donc un N tel que

$$\sum_{n>N} \delta_0 \ \sup\{h(x); x \in [i\delta_0, (i+1)\delta_0[\} < \epsilon \ ;$$

d'autre part, h étant positive, nous avons

$$\begin{aligned} I^{\delta_0} - I_{\delta_0} &= \sum_{i=0}^{\infty} \{(\sup h(x) - \inf h(x)); \quad x \in [i\delta_0, (i+1)\delta_0[\} \\ &\leq \sum_{i=0}^{N} \{(\sup h(x) - \inf h(x)); \quad x \in [i\delta_0, (i+1)\delta_0[\} \\ &\quad + \sum_{n>N} \delta_0 \, \{\sup h(x) : \quad x \in [i\delta_0, (i+1)\delta_0[\}. \end{aligned} \tag{3.10}$$

Sur $[0, N\delta_0]$, la fonction h est continue et à support compact; elle est donc uniformément continue; nous pouvons donc trouver un entier $k > 0$ tel que

$$\forall j, \quad 0 < j < N2^k, \quad \{\sup h(x) - \inf h(x); \quad x \in [j\delta_0/2^k, (j+1)\delta_0/2^k]\} < \epsilon;$$

enfin, la dernière somme du terme de droite de l'inégalité 3.10 décroît quand le pas diminue, et donc quand on passe du pas δ_0 au pas $\delta_0/2^k$; nous avons donc obtenu la majoration

$$I^{\delta_0/2^k} - I_{\delta_0/2^k} \leq N2^k\epsilon + \epsilon;$$

la continuité absolue de h sur $[0, N\delta_0]$ étend ce dernier résultat à tout $\delta < \delta_0/2^k$, ce qui complète notre démonstration. □

Remarque: ce lemme permet de montrer d'une autre manière que L_{Z^+} et la fonction $z(y)$ du théorème 32, fonctions continues et à support compact, sont directement Riemann intégrable.

3.2.3 Le résultat asymptotique donné par le théorème du renouvellement

Dans la section 2.3, nous avions à résoudre une équation de la forme

$$H = h + H * F, \tag{3.11}$$

où $*$ représente l'opération de convolution (définie sur le demi-axe positif), F est la distribution d'une variable positive X, h est une fonction connue définie sur l'intervalle $[0, \infty]$, et H est la fonction inconnue. Une telle équation s'appelle équation du renouvellement. La recherche de solution à une telle équation fait partie de la théorie du renouvellement; nous suivons la ligne générale de présentation de Feller et de Durrett, sans donner les démonstrations. On pourra aussi consulter pour la théorie du renouvellement Dacunha-Castelle [21] et Ross [72].

L'équation 3.11 peut aussi s'écrire

$$H(t) = h(t) + \int_{s=0}^{t} H(t-s)F(ds). \tag{3.12}$$

Le théorème du renouvellement donne la valeur asymptotique de H à l'infini en fonction de l'intégrale sur $[0, \infty]$ de h et de l'espérance μ de X.

Nous considérons une suite de variables positives indépendantes et identiquement distribuée (X_n) de distribution commune F et de moyenne finie $(E(X_1) = \mu < \infty)$, et des temps de renouvellement $T_i = X_1 + \cdots + X_i$; (dans un modèle de client-serveur, les T_i peuvent être considérés comme les temps d'arrivée des clients).

Nous allons faire usage de la mesure $U = \sum_{i=1}^{\infty} F^{*i}$, où F^{*i} est la convolée i-ème de F avec elle-même; cette mesure est appelée mesure du renouvellement. On peut aussi définir de manière équivalente la mesure U sur tout sous-ensemble A de $\mathbb{R}^+$ comme

$$U(A) = \sum_{i=0}^{\infty} P(T_i \in A) \quad \text{avec } T_0 = 0.$$

On obtient une première propriété remarquable en considérant le temps d'arrêt N_t tel que $N_t = \inf\{k : T_k > t\}$: Considérons $U(t) = U([0,t])$; nous avons

$$\{T_i \in [0,t]\} \Longleftrightarrow \{N_t > i\}; \quad \text{et donc}$$

$$U(t) = \sum_{i=0}^{\infty} P(T_i \in [0,t]) = \sum_{i=0}^{\infty} P(N_t > i) = E(N_t).$$

Nous avons alors les deux théorèmes équivalents suivants:

Théorème 34. *(Théorème limite pour le temps d'arrêt). Quand* $t \to \infty$, $\frac{N_t}{t} \to \frac{1}{\mu}$ *presque sûrement, avec* $0 < \mu = EX_i \leq \infty$, *et* $1/\infty = 0$*;*

et, dans les mêmes conditions,

Théorème 35. *Quand* $t \to \infty$, $\frac{U(t)}{t} \to \frac{1}{\mu}$.

Preuve. (En suivant Durrett). On démontre d'abord que $E(N_t) < \infty$, résultat qui nous sera utile par la suite. En effet, nous avons $E(X_i) = \mu > 0$; il existe donc $\delta > 0$ tel que $P(X_i > \delta) = \epsilon > 0$. Choisissons alors K tel que $K\delta \geq t$. Puisque une séquence de K fois X_i consécutifs qui sont supérieurs à δ rendent $T_{mK} > K\delta > t$, et qu'il y a m séquences de K fois X_i dans T_{mK}, nous avons

$$\begin{aligned}
&P(K \text{ consécutifs } X_i \text{ tels que } X_i > \delta) = \epsilon^K,\\
&P(\text{Somme } \Sigma \text{ de } K \text{ consécutifs } X_i \text{ telle que } \Sigma < n\delta)\\
&\qquad < 1 - P(K \text{ consécutifs } X_i \text{ tels que } X_i > \delta) = 1 - \epsilon^K,\\
&P(m \text{ fois } \Sigma < n\delta) < P(\Sigma < n\delta)^m = (1-\epsilon^K)^m.
\end{aligned}$$

Ou encore, plus formellement,

$$\begin{aligned}
&P(X_i > \delta, i = j, \ldots, (j+K)) = \epsilon^K,\\
&P(\sum_{i=j}^{j+K} X_i < n\delta) < 1 - P(X_i > \delta, i = j, \ldots, (j+K)) = 1 - \epsilon^K,\\
&P(\sum_{i=j}^{j+K} X_i < n\delta, \ \ \forall j = l + sK, \ 0 \leq s \leq m) < P(\sum_{i=j}^{j+K} X_i < n\delta)^m = (1-\epsilon^K)^m,
\end{aligned}$$

ce qui nous donne

$$P(N_t > mK) = P(T_{mK} < t) < P(T_{mK} < K\delta) \leq (1 - \epsilon^K)^m.$$

Nous en déduisons

$$U(t) = E(N_t) = \sum_{i=0}^{\infty} P(N_t > i) \leq K \sum_{m=0}^{\infty} P(N_t > mK) \leq K \sum_{m=0}^{\infty} (1 - \epsilon^K)^m < \infty. \tag{3.13}$$

On utilise ensuite l'équation de Wald (voir Durrett)

$$E(S_{N_t}) = E(X)E(N_t), \quad (N_t \text{ temps d'arrêt}, \ S_n = \sum_{i=1}^{n} X_i), \tag{3.14}$$

en appliquant cette équation au temps d'arrêt N_t et à la somme T_{N_t} $(\geq t)$; ceci nous donne

$$\mu E(N_t) = \mu U(t) = E(T_{N_t}) \geq t;$$

$$\text{et, donc } \frac{U(t)}{t} \geq \frac{1}{\mu}. \tag{3.15}$$

D'autre part, si X est borné ($P(X \leq c) = 1$), étant donné que

$$N_t - 1 < N_t \Rightarrow S_{N_t - 1} < t, \qquad \text{par définition, et que,}$$

$$E(S_{N_t}) = E(S_{N_t - 1} + X) = E(S_{N_t - 1}) + E(X),$$

on obtient

$$\mu E N_t = \mu U(t) = E(S_{N_t}) \leq t + c; \tag{3.16}$$

en combinant les équations 3.15 et 3.16, on obtient

$$\frac{U(t)}{t} \to \frac{t}{\mu} \quad \text{quand} \quad t \to \infty.$$

On trouvera dans Durrett un complément de démonstration dans le cas où X n'est pas bornée. □

La solution de l'équation de renouvellement 3.11 s'obtient de manière équivalente comme celle de l'équation

$$H(t) = \int_0^t h(t - s) dU(s); \tag{3.17}$$

Ceci est immédiat en itérant le développement de H dans l'équation 3.11.

Le théorème de Blackwell qui suit décrit le comportement à l'infini de la mesure de renouvellement U; son application permettra d'arriver au théorème du renouvellement qui nous donne la valeur asymptotique de H à l'infini.

Théorème 36 (Blackwell). *Si F est non-arithmétique, la mesure de renouvellement U vérifie $U([t, t+h]) \to \frac{h}{\mu}$ quand $t \to \infty$.*

Dans le cas arithmétique, on a:

Théorème 37. *Si F est arithmétique de pas τ, $U([t, t+h]) \to \frac{h}{\mu}$ quand $t \to \infty$ et quand h est un multiple du pas τ.*

On peut consulter Feller dans le cas arithmétique et, dans le cas continu, Durrett, qui utilise une méthode de couplage avec un processus de renouvellement stationnaire.

Nous avons besoin du lemme suivant concernant le mesure de renouvellement U:

Lemme 38. *Pour tout s et t positifs, on a $U([t, t+s]) \leq U([0, s]) = U(s)$.*

Preuve. Remarquons tout d'abord que N_t est le nombre de renouvellements dans l'espace de temps $[0, t]$, en comptant le renouvellement $T_0 = 0$.

Nous noterons $N_{[t,s]}$ (resp. $N_{]t,s]}$) avec $t < s$ le nombre de renouvellements dans l'espace de temps fermé $[t, s]$ (resp. semi-ouvert $]t, s]$). Intuitivement, l'espérance $U([t, t+s]$ du nombre de renouvellements dans $[t, t+s]$ est au plus $U(t)$, puisque le cas le pire correspond à un renouvellement au temps t.

Plus formellement, nous allons considérer les cas où $\{T_{N_t} = t\}$ (renouvellement en t) et où $\{T_{N_t} \neq t\} = \{T_{N_t} < t\}$ (pas de renouvellement en t). Nous avons

$$E(N_{[t,t+s]}) = E(N_{[t,t+s]} 1_{\{T_{N_t}=t\}}) + E(N_{[t,t+s]} 1_{\{T_{N_t}<t\}}); \tag{3.18}$$

Compte tenu de l'indépendance des incréments X_i, nous avons d'une part

$$E(N_{[t,t+s]} 1_{\{T_{N_t}=t\}}) = E(N_{[0,s]}) P(T_{N_t} = t),$$

et d'autre part

$$\begin{aligned} E(N_{[t,t+s]} 1_{\{T_{N_t}\neq t\}}) &= \int_{t^+}^{t+s} P(T_{N_t+1} = u) E((1 + N_{]t+u,t+s]}) 1_{\{T_{N_t+1}=u\}} 1_{\{T_{N_t}\neq t\}}) \, du \\ &= \int_{t^+}^{t+s} P(T_{N_t+1} = u) E((1 + N_{]u,t+s]}) 1_{\{T_{N_t+1}=u\}}) P(T_{N_t} < t) \, du, \end{aligned}$$

compte tenu de l'indépendance des événements $\{(1 + N_{]u,t+s]}) 1_{\{T_{N_t+1}=u\}}; u > t\}$ et $\{T_{N_t} < t\}$.

Mais nous avons aussi

$$E(1 + N_{]t+u,t+s]} 1_{\{T_{N_t+1}=u\}}; 0 < u \leq s) = E(N_{[0,s-u]}) \leq E(N_{[0,s]}),$$

puisque l'espérance du nombre de renouvellements dans $[t+u, t+s]$, sachant qu'il y a un renouvellement en $t+u$, est égale au nombre de renouvellement dans $[0, s-u]$, en comptant le renouvellement au temps zéro.

Nous en déduisons que

$$E(N_{[t,t+s]}) \leq E(N_{[0,s]}) P(T_{N_t} = t) + E(N_{[0,s]}) P(T_{N_t} < t) = E(N_{[0,s]}),$$

ou encore, de manière équivalente, que

$$\forall t > 0, \ \forall s > 0 \qquad U([t, t+s]) \leq U([0, s]). \tag{3.19}$$

□

Nous arrivons maintenant au théorème du renouvellement lui-même que nous allons démontrer en suivant Durrett.

Théorème 39 (Théorème du renouvellement). *Si H est donné par l'équation de renouvellement $H = h + H * F$, où F est non-arithmétique et h est directement Riemann intégrable, alors la valeur asymptotique de H à l'infini est donnée par*

$$H(t) \to \frac{1}{\mu}\int_0^\infty h(s)ds \quad \textit{quand} \quad t \to \infty, \tag{3.20}$$

où μ est la moyenne de la variable de loi F.

(Voir la définition 31 pour l'intégrabilité Riemannienne directe.)

Preuve. Nous allons chercher à résoudre l'équation intégrale $H(t) = \int_0^t h(t-s)dU(s)$ équivalente à l'équation du renouvellement; nous effectuons la démonstration dans le cas général où le support de h est $[0, \infty]$.

Considérons d'abord une fonction h en escalier

$$h(s) = \sum_{k=0}^{\infty} a_k 1_{[k\delta,(k+1)\delta[}(s),$$

où $\sum_{k=0}^\infty |a_k| < \infty$. Les équations 3.13 et 3.19 nous donnent $U([t, t+\delta]) \leq U([0, \delta]) < \infty$.

Choisissons K tel que

$$\sum_{k \geq K} |a_k| \leq \frac{\epsilon}{2U([0,\delta])};$$

choisissons de plus T tel que, pour $t \geq T$ et $0 \leq k < K$, on ait

$$|a_k|.\,|U([t-(k+1)\delta, t-k\delta]) - \delta/\mu| \leq \frac{\epsilon}{2K},$$

ce qui possible en raison du théorème de Blackwell ($U([t, t+h]) \to h/\mu$, quand $t \to \infty$).

En considérant $U[a,b] = 0$ quand $a < b \leq 0$ et $U[a,b] = U[0,b]$ quand $a < 0 \leq b$, nous avons

$$\int_0^t h(t-s)dU(s) = \int_0^t h(t)dU(t-s) = \sum_{k=0}^{\infty} a_k U([t-(k+1)\delta, t-k\delta]),$$

et donc

$$\left| \int_0^t h(t-s)dU(s) - \frac{1}{\mu}\sum_{k=0}^{K} a_k\delta \right| < \epsilon/2 + \epsilon/2 = \epsilon,$$

dont nous déduisons en faisant tendre K et t vers l'infini

$$\int_0^t h(t-s)dU(s) \to \frac{1}{\mu}\sum_{k=0}^{\infty} a_k\delta.$$

En remarquant que si h_1 est dominée par h_2, la fonction $H_1(x) = \int_0^t h_1(t-s)dU(s)$ est dominée par la fonction $H_2(x) = \int_0^t h_2(t-s)dU(s)$, nous pouvons appliquer le dernier résultat, pour une fonction h quelconque, et pour un $\delta > 0$ quelconque, aux fonctions en escalier

$$\overline{h}_\delta(x) = \{\sup h(x);\quad x \in [k\delta, (k+1)\delta[\ \} \quad \text{et} \quad \underline{h}_\delta(x) = \{\inf h(x);\quad x \in [k\delta, (k+1)\delta[\ \};$$

ce qui nous donne

$$\begin{aligned}\frac{I_\delta}{\mu} = \lim_{t\to\infty} \int_0^t \underline{h}_\delta(t-s)dU(s) &\leq \liminf_{t\to\infty} \int_0^t h(t-s)dU(s) \\ &\leq \limsup_{t\to\infty} \int_0^t h(t-s)dU(s) \leq \lim_{t\to\infty} \int_0^t \overline{h}_\delta(t-s)dU(s) = \frac{I^\delta}{\mu},\end{aligned}$$

où I_δ et I^δ sont les sommes de Riemann directes supérieures et inférieures définies par la subdivision de pas δ de $[0, \infty]$.

Si h est directement Riemann intégrable, I_δ et I^δ approchent la même limite finie I quand $\delta \to 0$, et nous avons alors

$$H(t) = \int_0^t h(t-s)dU(s) \to \frac{I}{\mu} = \frac{1}{\mu}\int_0^\infty h(s)ds \quad \text{quand} \quad t \to \infty.$$

□

3.3 Théorèmes Taubériens

Nous avons utilisé dans le paragraphe 2.2.3 le théorème Taubérien suivant.

Théorème 40. *Soit une suite de nombres réels $c_n \geq 0$, telle que $\sum_{n=0}^\infty c_n r^n$ converge pour $0 \leq r < 1$, alors*

$$\lim_{r\to 1^-} \sum_{n=0}^\infty c_n r^n = \sum_{n=0}^\infty c_n,$$

où $\sum_{n=0}^\infty c_n$ peut être fini ou infini.

On peut consulter Titchmarsh [83] ou Gourdon [32] pour les théorèmes Taubériens dans le cas où c_n est une suite de valeurs complexes.

Nous avons besoin d'hypothèses moins fortes (suite réelle positive) pour lesquelles Rudin [73] nous donne le théorème suivant (p. 74) dans le cas où $\sum c_n$ converge.

Théorème 41. *Supposons que, pour la série à termes réels c_n, $\sum c_n$ converge. Soit $f(x) = \sum_{n=0}^\infty c_n x^n$. Alors,*

$$\lim_{x\to 1} f(x) = \sum c_n.$$

Dans le cas où $\sum c_n = \infty$, en considérant une série c_n à termes positifs, pour tout $A > 0$, il existe un entier N tel que $\sum_{n=0}^N c_n > A$.

Nous avons alors $f(x) > \sum_{n=0}^N c_n x^n > x^N \sum_{n=0}^N c_n > x^N A$, et donc $f(x) \to \infty$ quand $x \to 1$.

Les deux cas, somme finie ou infinie du théorème 40 sont donc traités.

Chapitre 4

Le logiciel d'alignement local de séquences BLAST

4.1 Introduction

Quand on découvre un nouveau gène, ou une nouvelle protéine, les méthodes d'homologie permettent la détection de similarités avec d'autres gènes, ou d'autres protéines dont les propriétés sont connues. La découverte de telles similarités d'une protéine et de protéines contenues dans une base de protéines telle que SWISSPROT donne souvent d'importants aperçus sur les propriétés fonctionnelles, métaboliques et structurales de la nouvelle protéine. Les outils de recherche de similarités sont par conséquent précieux pour les biologistes.

4.1.1 Alignement de deux séquences

Waterman traite dans son livre "Computational Biology" [85] de nombreux aspects relatifs à l'alignement de deux séquences. Nous reformulons les définitions principales, dans le cas de la comparaison de deux séquences protéiques.

Soient deux séquences

$$\begin{aligned} & u = x_1 x_2 \ldots x_{n-1} x_n, \\ \text{et} \quad & v = y_1 y_2 \ldots y_{n-1} y_n, \end{aligned}$$

où x_i et y_j représentent des acides aminés ($x_i, y_j \in \Sigma$), et Σ est l'alphabet représentatif des acides aminés. Nous considérons alors une matrice de similarité $\mathcal{M}$ donnant pour chaque paire d'acides aminés (a_i, a_j) un score, avec la notation $s(a_i, a_j) = \mathcal{M}_{ij}$; ce score est élevé pour des acides aminés de caractéristiques biologiques ou chimiques voisines, et bas quand les caractéristiques sont éloignées. En associant une distribution de scores p_i à l'alphabet $\Sigma = \{a_i\}$, on définit une variable aléatoire $\{X, \Sigma \times \Sigma \to \mathbb{R}\}$. Deux conditions techniques essentielles à la recherche de similarité entre deux séquences sont que l'espérance de X soit négative, et qu'il existe une probabilité non nulle de scores positifs:

$$E(X) = \sum_{i,j} p_i p_j s(a_i, a_j) < 0, \quad P(X > 0) > 0.$$

Ceci induit pour l'alignement optimal de deux séquences de longueur n et m un score et une longueur en $\mathcal{O}(\log(n))$. On peut éventuellement se ramener à une situation où ces deux conditions sont vérifiées par "recalibration" des scores. Les méthodes permettant des insertions et suppressions (pour lesquels on utilise le raccourci "indels", de l'anglais insertions et deletions), supposent une pénalité de trou γ, où un trou (nous utiliserons désormais le mot anglais gap) correspond à un saut dans l'alignement. De nombreuses méthodes supposent deux pénalités différentes:

- une pénalité d'ouverture de gap γ_1,
- et une pénalité d'extension de gap γ_2.

Un choix de $\gamma_1 = \gamma_2 = \infty$ équivaut à interdire les indels. On peut considérer un alignement $\mathcal{A}$ de deux séquences x_i et y_j comme une suite de paires d'indices (i_t, j_t), où les i_t et les j_t forment des suites croissantes qui peuvent être localement stationnaires. L'approche classique utilise alors un schéma de scores additifs qui permet de calculer des scores d'alignements par programmation dynamique.

$$S(i_{t+1}, j_{t+1}) = \max \begin{cases} 0, \\ S(i_t, j_t) + s(x_{i_t+1}, y_{i_t+1}), \\ S(i_t + 1, j_t) + \gamma, \\ S(i_t, j_t + 1) + \gamma. \end{cases} \tag{4.1}$$

Si l'on place les scores obtenus dans un tableau à deux entrées correspondants aux indices dans les deux séquences, on peut considérer les cases d'indice (i_m, j_m) telles que la valeur de la case (son score S_{i_m,j_m}) soit maximum. En procédant en marche arrière, on peut chercher la ou les cases antécédentes de valeur maximum; ce sont les cases d'indice (i_{m-1}, j_{m-1}), telles que $\{i_{m-1} = i_m - 1 \text{ ou } i_m\}$, $\{j_{m-1} = j_m - 1 \text{ ou } j_m\}$, et $\{S_{i_{m-1},j_{m-1}} = \max(S_{i_m-1,j_m-1}, S_{i_m-1,j_m}, S_{i_m,j_m-1})\}$. Ce processus s'arrête lorsque l'on arrive à une case de score nul. Les suites de paires d'indices obtenues correspondent à des alignements optimaux (il peut y en avoir plusieurs).

La méthode d'alignement de séquences avec indels a été inventée par Needleman et Wunsch (1970) [60]. L'amélioration majeure de cette méthode vient de Pearson et Lipman dans FASTA (1988) [66], qui accélère les recherches au moyen d'une table de hachage des mots courts de l'une des deux séquences. La programmation dynamique est alors seulement utilisée pour étendre des alignements primaires ayant un score élevé.

Les méthodes d'alignement global de séquences optimisent l'alignement des deux séquences dans toute leur longueur. La première contrainte de l'équation 4.1, $S(i_{t+1}, j_{t+1}) \geq 0$, est relaxée, et il s'en suit que les scores obtenus dans la construction d'alignements globaux peuvent être négatifs (on pourra consulter le chapitre 6 pour plus de détails sur la construction de ces alignements). Ces méthodes d'alignement global provoquent souvent l'inclusion dans l'alignement résultant de longs morceaux de faible similarité. Au contraire, les méthodes d'alignement local cherchent les alignements de sous-séquences des séquences initiales qui sont bien conservées. Ceci induit que l'on peut trouver plusieurs alignements locaux lors de l'alignement de deux séquences, correspondant à des alignements globaux de différentes sous-séquences.

Quand on interdit les insertions-suppressions, et que l'on recherche des alignements locaux maximaux, la méthode la plus populaire est certainement la méthode d'alignement local Blast [3] (Basic Local Alignment Search Tool) que nous décrivons dans ce chapitre. Un avantage majeur de Blast est le cadre probabiliste solide donnant la pertinence d'un alignement, cadre que nous avons décrit au chapitre 2.

Nous verrons expérimentalement au chapitre 6 que ce cadre probabiliste s'applique bien à l'extension de Blast que nous avons faite pour traiter les alignements multiples, extension que nous décrivons au chapitre 5.

4.2 Description générale de l'algorithme Blast

Blast [3] a été développé en 1990 par Altschul, Gish, Miller, Myers et Lipman peu après que les travaux de Karlin, Dembo, Kawabata et Altschul [49, 51, 46] aient permis de calculer de manière rigoureuse la pertinence probabiliste des scores des alignements sans gaps de paires de séquences.

Le tableau 4.1 donne la liste des acides aminés et leur probabilité d'apparition dans les protéines.

Acide aminé	Abréviation	Probabilité d'apparition
Alanine	A	0.081
Arginine	R	0.057
Asparagine	N	0.045
Acide aspartique	D	0.054
Cystéine	C	0.015
Glutamine	Q	0.039
Acide Glutamique	E	0.061
Glycine	G	0.068
Histidine	H	0.022
Isoleucine	I	0.057
Leucine	L	0.093
Lysine	K	0.056
Méthionine	M	0.025
Phénylalanine	F	0.040
Proline	P	0.049
Sérine	S	0.068
Thréonine	T	0.058
Tryptophane	W	0.013
Tyrosine	Y	0.032
Valine	V	0.067

Tableau 4.1: Les acides aminés, constituants des protéines. On remarquera que le Tryptophane est un acide aminé rare, tandis que l'Alanine et la Leucine sont fréquemment présents dans les protéines.

```
#  Matrix made by matblas from blosum62.iij
#  * column uses minimum score
#  BLOSUM Clustered Scoring Matrix in 1/2 Bit Units
#  Blocks Database = /data/blocks_5.0/blocks.dat
#  Cluster Percentage: >= 62
#  Entropy =   0.6979, Expected =  -0.5209
   A  R  N  D  C  Q  E  G  H  I  L  K  M  F  P  S  T  W  Y  V  B  Z  X  *
A  4 -1 -2 -2  0 -1 -1  0 -2 -1 -1 -1 -1 -2 -1  1  0 -3 -2  0 -2 -1  0 -4
R -1  5  0 -2 -3  1  0 -2  0 -3 -2  2 -1 -3 -2 -1 -1 -3 -2 -3 -1  0 -1 -4
N -2  0  6  1 -3  0  0  0  1 -3 -3  0 -2 -3 -2  1  0 -4 -2 -3  3  0 -1 -4
D -2 -2  1  6 -3  0  2 -1 -1 -3 -4 -1 -3 -3 -1  0 -1 -4 -3 -3  4  1 -1 -4
C  0 -3 -3 -3  9 -3 -4 -3 -3 -1 -1 -3 -1 -2 -3 -1 -1 -2 -2 -1 -3 -3 -2 -4
Q -1  1  0  0 -3  5  2 -2  0 -3 -2  1  0 -3 -1  0 -1 -2 -1 -2  0  3 -1 -4
E -1  0  0  2 -4  2  5 -2  0 -3 -3  1 -2 -3 -1  0 -1 -3 -2 -2  1  4 -1 -4
G  0 -2  0 -1 -3 -2 -2  6 -2 -4 -4 -2 -3 -3 -2  0 -2 -2 -3 -3 -1 -2 -1 -4
H -2  0  1 -1 -3  0  0 -2  8 -3 -3 -1 -2 -1 -2 -1 -2 -2  2 -3  0  0 -1 -4
I -1 -3 -3 -3 -1 -3 -3 -4 -3  4  2 -3  1  0 -3 -2 -1 -3 -1  3 -3 -3 -1 -4
L -1 -2 -3 -4 -1 -2 -3 -4 -3  2  4 -2  2  0 -3 -2 -1 -2 -1  1 -4 -3 -1 -4
K -1  2  0 -1 -3  1  1 -2 -1 -3 -2  5 -1 -3 -1  0 -1 -3 -2 -2  0  1 -1 -4
M -1 -1 -2 -3 -1  0 -2 -3 -2  1  2 -1  5  0 -2 -1 -1 -1 -1  1 -3 -1 -1 -4
F -2 -3 -3 -3 -2 -3 -3 -3 -1  0  0 -3  0  6 -4 -2 -2  1  3 -1 -3 -3 -1 -4
P -1 -2 -2 -1 -3 -1 -1 -2 -2 -3 -3 -1 -2 -4  7 -1 -1 -4 -3 -2 -2 -1 -2 -4
S  1 -1  1  0 -1  0  0  0 -1 -2 -2  0 -1 -2 -1  4  1 -3 -2 -2  0  0  0 -4
T  0 -1  0 -1 -1 -1 -1 -2 -2 -1 -1 -1 -1 -2 -1  1  5 -2 -2  0 -1 -1  0 -4
W -3 -3 -4 -4 -2 -2 -3 -2 -2 -3 -2 -3 -1  1 -4 -3 -2 11  2 -3 -4 -3 -2 -4
Y -2 -2 -2 -3 -2 -1 -2 -3  2 -1 -1 -2 -1  3 -3 -2 -2  2  7 -1 -3 -2 -1 -4
V  0 -3 -3 -3 -1 -2 -2 -3 -3  3  1 -2  1 -1 -2 -2  0 -3 -1  4 -3 -2 -1 -4
B -2 -1  3  4 -3  0  1 -1  0 -3 -4  0 -3 -3 -2  0 -1 -4 -3 -3  4  1 -1 -4
Z -1  0  0  1 -3  3  4 -2  0 -3 -3  1 -1 -3 -1  0 -1 -3 -2 -2  1  4 -1 -4
X  0 -1 -1 -1 -2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -2  0  0 -2 -1 -1 -1 -1 -1 -4
* -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4 -4  1
```

Figure 4.1: La matrice de score BLOSUM62 sur l'alphabet des acides aminés. Les lettres B, Z et X correspondent à un alphabet dégénéré; B veut dire D ou N, Z veut dire E ou Q, et X est une lettre passe-partout pouvant être remplacée par n'importe lequel des 20 acides aminés.

Les matrices de similarité entre acides aminés les plus utilisées sont les matrices PAM [22] et BLOSUM [41]. On trouvera une description de la construction de ces matrices dans Lerman et *al.* [58]. La figure 4.1 représente la matrice BLOSUM62, qui est la plus fréquemment utilisée parmi les matrices BLOSUM. L'indice (ici 62) correspond à un nombre d'itérations dans un processus de type Markovien. Les paramètres de ce processus sont évalués à partir des fréquences de mutations ponctuelles dans des familles de protéines homologues. Si le score de l'alignement de deux acides aminés a_i et a_j dans une telle matrice est $s(a_i, a_j)$ le score de l'alignement local $\{x_{j_1}, \ldots, x_{j_1+l}/y_{j_2}, \ldots, y_{j_2+l}\}$ de deux séquences $\{x_1, \ldots\}$ et $\{y_1, \ldots\}$ sera $S = \sum_{i=0}^{l-1} s(x_{j_1+i}, y_{j_2+i})$. L'objectif de Blast est de déduire de la donnée des deux séquences initiales les alignements de score élevé, et plus généralement d'effectuer cette opération lors des alignements d'une séquence requête et de N séquences cibles.

Pour une séquence requête de taille n et une séquence cible de taille m, l'espace de recherche a une dimension voisine de nm. Blast utilise une approche algorithmique robuste et très efficace qui lui permet de ne parcourir qu'une partie de cet espace de recherche. Blast peut être utilisé pour faire l'alignement de séquence nucléiques d'ADN, l'alignement de séquences protéiques, et l'alignement de séquences d'ADN et de séquences protéiques. Dans ce dernier cas, Blast génère les séquences protéiques correspondant à l'ADN, puis traite à nouveau la comparaison de séquences protéiques. Les mécanismes algorithmiques principaux sont les mêmes dans ces différents cas, qui différent par des conditionnements préalables des séquences. Nous décrirons donc uniquement le cas de l'alignement de séquences protéiques effectué par le logiciel `Blastp` que nous avons étendu pour effectuer de l'alignement avec des multi-alignements.

Blast se présente sous la forme de deux composantes logicielles, un preprocessing off-line `setdb`, effectué une fois pour toutes sur la base de séquences cibles sur laquelle on veut travailler. Le deuxième composant, `blastp` est lancé chaque fois que l'on veut effectuer une recherche d'alignement d'une séquence requête et des séquences cibles. Ce deuxième composant intègre l'essentiel des méthodes algorithmiques et probabilistes de Blast. Nous décrivons dans le prochain paragraphe ces deux composantes.

```
>CCHU (PIR) Cytochrome c - Human
GDVEKGKKIFIMKCSQCHTVEKGGKHKTGPNLHGLFGRKTGQAPGYSYTAANKNKGIIWG
EDTLMEYLENPKKYIPGTKMIFVGIKKKEERADLIAYLKKATNE
>CCCZ (PIR) Cytochrome c - Chimpanzee
GDVEKGKKIFIMKCSQCHTVEKGGKHKTGPNLHGLFGRKTGQAPGYSYTAANKNKGIIWG
EDTLMEYLENPKKYIPGTKMIFVGIKKKEERADLIAYLKKATNE
>CCMQR (PIR) Cytochrome c - Rhesus macaque
GDVEKGKKIFIMKCSQCHTVEKGGKHKTGPNLHGLFGRKTGQAPGYSYTAANKNKGITWG
EDTLMEYLENPKKYIPGTKMIFVGIKKKEERADLIAYLKKATNE
```

Figure 4.2: Séquences cibles input de `setdb` au format FASTA

4.3 Description détaillée de Blast

4.3.1 Le preprocessing off-line `setdb`

Le preprocessing de Blast est une mise en forme des données des séquences cibles visant la meilleure efficacité du traitement on-line.

Comme exemple de ce preprocessing, considérons la base cible constituée des trois séquences de la figure 4.2 au format FASTA. Les lignes commençant par le caractère “>” sont des lignes header donnant l'identificateur de la séquence protéique et une description brève de la protéine correspondante. Les autres lignes donnent la séquence elle-même.

Le preprocessing `setdb` génère à partir de la base input 3 fichiers, un fichier contenant les séquences, un fichier contenant les headers, et un fichier d'index pointant à la fois sur les débuts de séquences et sur les débuts de headers.

- *Fichier des séquences.* Ce fichier est une concaténation des séquences, séparées par des octets nuls. Cependant, pour accélérer le traitement principal, les acides aminés sont codés par un nombre binaire allant de 1 à 20, et correspondant à l'indice de l'acide aminé dans les matrices de similarités PAM et BLOSUM. Ce codage permet, lors du traitement principal, quand un acide aminé de la requête est aligné avec un acide aminé de la cible, d'accéder directement à l'entrée correspondante de la matrice de similarité utilisée.

- *Fichier des headers.* Ce fichier est formé de la concaténation des headers séparés par des octets nuls. Il ne sera utilisé par Blast que lors de l'édition de l'output du traitement principal.

- *Fichier index.* Ce fichier est une table. À chaque séquence de la base cible est associé un numéro correspondant à son numéro d'ordre séquentiel dans la base. L'entrée n de la table comprend un pointeur sur le début de la séquence n dans le fichier des séquences, et un pointeur sur le début du header de cette séquence dans le fichier des headers.

4.3.2 Le traitement principal `blastp`

4.3.2.1 Principes de l'algorithme

Les résultats de Karlin et *al.* permettent de calculer un score seuil S_M tel que les alignements de score supérieur à ce score aient une chance faible d'être obtenus par hasard, c'est à dire par alignement de séquences aléatoires ayant même longueur que les séquences considérées. L'équation 2.30 s'écrit pour deux séquences de longueur m et n

$$\Pr\left(S > \frac{\log(mn)}{\lambda} + x\right) = 1 - e^{-Ke^{-\lambda x}},$$

où S est le score de l'alignement local optimal des deux séquences. Si l'on veut avoir

$$\Pr\left(S > \frac{\log(mn)}{\lambda} + x\right) \approx Ke^{-\lambda x} < \epsilon,$$

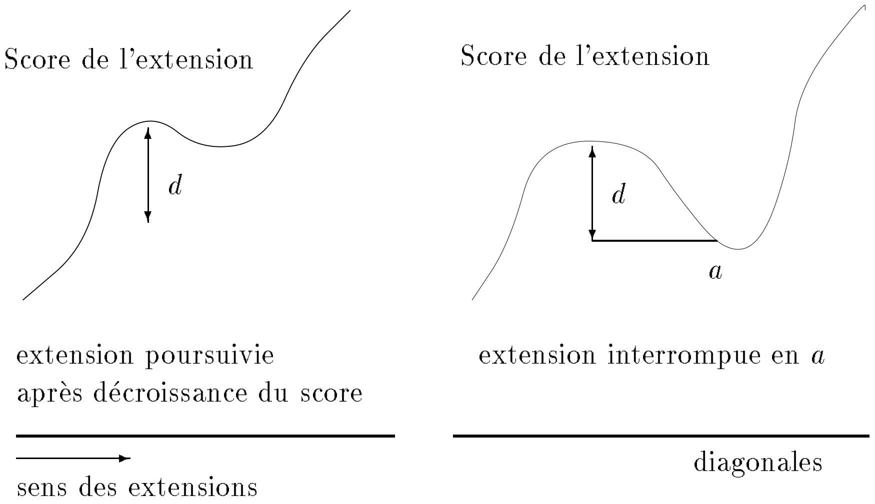


Figure 4.3: L'heuristique d'interruption de certaines extensions

il suffit de choisir

$$S_M = \frac{1}{\lambda} \log \left(\frac{Kmn}{\epsilon} \right).$$

La stratégie de Blast consiste à minimiser le temps passé à calculer des alignements ayant une faible probabilité d'atteindre cette valeur seuil S_M, pour un ϵ donné. Pour réaliser cette stratégie, Blast utilise successivement plusieurs heuristiques, l'appariement des mots courts, l'interruption de certaines extensions ayant de trop fortes portions de score décroissant, et le marquage de l'espace de recherche déjà parcouru. Nous décrivons ces trois heuristiques.

- *L'appariement des mots courts.* Blast utilise des appariements de mots courts d'une longueur donnée c. L'heuristique considère que si un alignement ne contient pas au moins un mot court c de score au moins égal à un seuil T, alors cet alignement peut être éliminé. Ceci revient à dire qu'au prix de la perte d'un faible pourcentage d'alignements de bonne qualité, on peut éliminer très rapidement les alignements de mauvaise qualité. Un parcours de la séquence requête s_r permet de générer l'ensemble $M_c = \{m_c\}$ de tous les mots s'appariant suffisamment avec l'un des mots de longueur c de la requête;

  $$m_c \in M_c \ (|m_c| = c) \quad \text{si } \exists r_c \subset s_r, |r_c| = c, \ S(m_c, r_c) \geq T.$$

  Le parcours de la séquence cible permet alors de détecter les mots appartenant à M_c. Si l'on a gardé des pointeurs vers les mots courts de la requête s'appariant au mot trouvé dans M_c, on peut étendre cet appariement local appelé *hit* en un alignement plus long, d'une part à gauche du *hit*,. On appellera improprement *diagonales* le support de ces extensions. Une diagonale est un ensemble de paires (i, j), où i est une position sur la séquence requête, et j une position sur la séquence cible, telles que l'on ait $i - j = k$, pour une valeur constante k. L'extension d'un *hit* se fait, à gauche, en progressant le long de la diagonale du *hit*, par valeurs de positions décroissantes, et, de la même manière, à droite, en progressant sur la diagonale par valeurs de positions croissantes; dans les deux cas, on ajoute le score des acides aminés situés aux positions i et j considérées au score déjà obtenu.

- *L'interruption de certaines extensions.* La deuxième heuristique consiste à abandonner l'extension lorsque le score de celle-ci décroît plus qu'une valeur seuil d. Si on observe une telle décroissance du score, on revient à la dernière position à laquelle le score a augmenté, et on arrête l'extension considérée, à gauche, ou à droite (voir figure 4.3). On interrompt une extension, soit parce que l'on a atteint l'extrémité d'une des séquences, soit parce que le score décroît trop fortement. Lorsqu'une extension est interrompue, on mémorise la position où le score maximum a été atteint. Cette opération est effectuée pour l'extension à droite et pour l'extension à gauche. Si le score total des deux extensions est inférieur à un score seuil S_2, l'extension du *hit* est rejetée. Si, au contraire, l'alignement a un score au moins égal à S_2, toute la diagonale est balayée, sans tenir compte du score maximum de décroissance d. Ce sont les alignements obtenus après balayage complet d'une diagonale qui sont mémorisés.

- *Marquage de l'espace de recherche.* On considère deux séquences de longueur respective m et n. On peut modéliser l'espace de recherche de l'alignement de ces deux séquences par un rectangle quadrillé de cotés de longueur respectives m et n (les cotés des carrés du quadrillage ont une longueur unité). On choisit le point en haut et à gauche du rectangle comme point origine. Un alignement correspond dans ce modèle à un sous-ensemble connexe d'une diagonale. Les extensions droites et gauche d'un *hit* se font toujours sur une diagonale. Blast parcourt la séquence cible dans le sens des positions croissantes. De même, les mots courts des *hits*, pour une position donnée sur la cible sont fournis ordonnés selon des positions croissantes sur la requête. Par cette construction, si l'on a parcouru lors d'une extension (éventuellement interrompue) une diagonale depuis une position de *hit* x_H jusqu'à une position x_f, et si, ultérieurement, on trouve sur cette diagonale un autre *hit* en position $y \leq x_f$, on est sûr que l'extension faite depuis ce *hit* a déjà été faite, qu'elle ait été acceptée ou rejetée. Il suffit donc de mémoriser, pour chaque diagonale, la plus grande position atteinte. Si un *hit*, sur une diagonale, a une position inférieure à cette position, il est inutile de l'étendre.

Nous décrivons dans le prochain paragraphe l'implémentation de Blast basée sur les principes que nous avons décrit.

4.3.2.2 Implémentation de Blast

L'implémentation de Blast est composée des phases et étapes principales suivantes.

1. *Phase-1.* La génération des mots courts voisins des mots courts de la requête et la création d'un automate *multi-string* de type Aho-Corasick contenant ces mots courts.

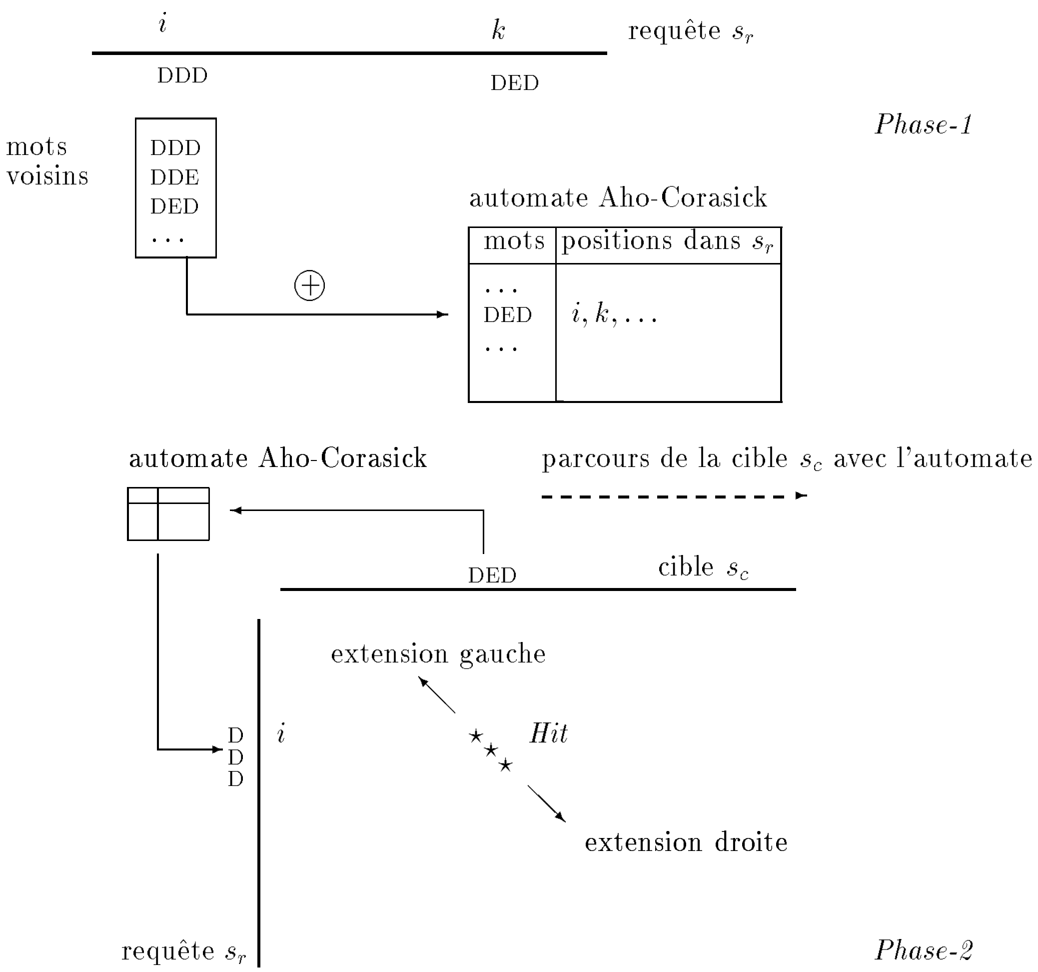


Figure 4.4: Vue générale de l'implémentation de Blast

2. *Phase-2.*

   (a) *Étape 2.a.* Le parcours des séquences cibles, avec détection des *hit*s.

   (b) *Étape 2.b.* L'extension des *hit*s et la mémorisation des alignements de score suffisant.

   (c) *Étape 2.c.* Le traitement probabiliste des alignements stockés, suivi d'un tri et d'un filtrage des alignements de pertinence probabiliste insuffisante.

La figure 4.4 donne une vue générale des phases 1 et 2.

Pour construire l'automate Aho-Corasick de la phase-1, automate que nous décrivons au paragraphe 4.3.2.2, la séquence requête est parcourue. Pour chaque position, on considère le mot court m_r de longueur c commençant à cette position et on génère tous les mots courts m_x tels que $s(m_r, m_x) \geq T$, où T est une valeur seuil. Des valeurs efficaces de c et T pour les protéines sont $c = 3$ et $T = 12$ dans le contexte de BLOSUM62.

Génération des mots courts. La génération des mots courts se fait au moyen d'une astuce algorithmique qui permet de ne pas générer la plupart des mots courts de score inférieurs à T. Lors de la lecture de la matrice de similarité, un tableau $\mathbf{T}_{max}$ est généré, donnant pour chaque acide aminé le score maximum obtenu avec un quelconque des acides aminés; en général, ce score est le score de l'acide aminé avec lui-même. Une matrice d'ordre $\mathbf{O}$ des acides aminés est également construite, qui donne pour chaque acide aminé a la liste des acides aminés dans l'ordre décroissant des scores obtenus avec a. Étant donné un mot court $x_1x_2\ldots x_c$, on déduit du tableau $\mathbf{T}$ le score minimum n_i qu'il faut avoir atteint à la position $i < c$ pour pouvoir espérer atteindre le score T à la position c. L'énumération des lettres correspondant aux acides aminés se fait alors de manière récursive, mais en faisant à la position i le parcours de l'alphabet dans l'ordre donné par la matrice $\mathbf{O}$ pour la lettre x_i.

On peut conceptualiser cette énumération par le parcours d'un *trie* planaire correspondant à un mot court de la requête.

- Chaque nœud du *trie* a un nombre de fils égal à la taille de l'alphabet, chaque nœud (à l'exception de la racine) est étiqueté par une lettre de l'alphabet, et l'étiquetage des fils d'un nœud de profondeur i (la racine est de profondeur 1) se fait de gauche à droite dans l'ordre de la matrice $\mathbf{O}$ pour la lettre x_i.

- L'arbre est parcouru en préordre, et le score d'un nœud tel que les étiquettes de la racine à ce nœud soient $y_1y_2\ldots y_i$ est $s(x_1x_2\ldots x_i/y_1y_2\ldots y_i)$.

- On raccourcit le parcours en préordre, dès que le score d'un nœud est inférieur à n_i. Dans ce cas, on ne parcourt pas les sous-arbres enracinés en ce nœud, ni les sous-arbres enracinés dans ses frères de droite.

Pour les valeurs $c = 3$ et $T = 12$, cette méthode d'énumération permet de se limiter à l'énumération d'une vingtaine de mots courts pour chaque mot court de la requête, sur les $20^3 = 8000$ mots courts possibles.

Génération de l'automate d'Aho-Corasick. Chaque mot court généré par l'algorithme d'énumération décrit çi-dessus est inséré dans un automate *multi-string* d'Aho-Corasick, que nous décrivons au paragraphe 4.3.2.3. Cet automate est légèrement modifié en forme de transducteur. Les états terminaux de l'automate génèrent la liste des positions de la requête démarrant un mot court de score suffisant avec le mot court détecté par l'automate.

Extension des Hits. Le passage sur un état terminal de l'automate Aho-Corasick déclenche un *hit*. Si le *hit* ne se situe pas sur une position de l'espace de recherche qui a déjà été parcourue, le *hit* est étendu. Les *hits* sont étendus le long de la diagonale correspondante, d'abord à gauche, puis à droite. La figure 4.5 donne l'algorithme d'extension. Son principe consiste à marquer les positions successives correspondant à un record pour le score. Entre ces records, l'extension peut passer sur des parties de l'alignement où le score décroît. Si cette décroissance est trop forte, l'extension est interrompue, et on en reste au dernier record. La fin d'extension se produit aussi naturellement lorsqu'on atteint le début ou la fin d'une des deux séquences, marqués par des octets nuls (le score de ces octets avec un

```
q_Hit → beginning of the Hit in the Query
t_Hit → beginning of the Hit in the Target

Blast_extension(q_Hit, t_Hit)
{
    q    pointer to the Query
    t    pointer to the Target
    d    maximum decrease of score allowed during extension
    M    similarity matrix

    if (Hit ∈ already searched portion of corresponding diagonal)    return

    /* extend to the left */
    score = sum = 0;    q = q_Hit;  t = t_Hit
    x = max (−score_of_Hit,−d)
    do {
        q−−;  t−−;    sum += M(*q,*t)

        if ( sum > 0 ) {
            score += sum;   sum = 0
            x = max (−score, −d)
        }
    } while (sum ≥ x) /* fails on null prefix of either sequence */

    /* extend to the right */
    sum = 0
    q = q_Hit;  t = t_Hit
    do {
        sum += M(*q,*t);    q++;  t++

        if ( sum > 0 ) {
            score += sum;   sum = 0
            x = max(−score, −d)
        }
    } while (sum ≥ x) /* fails on null terminator of either sequence */

    if (score < S2)
        { Mark searched portion of diagonal;    return }

    /* full scan of the diagonal of the Hit */
}
```

Figure 4.5: Extension gauche et droite d'un *hit*

```
q_diag(Hit) → beginning of the diagonal corresponding to the Hit in the Query
t_diag(Hit) → beginning of the diagonal corresponding to the Hit in the Target

Blast_fulldiagonal_scan(Hit)
{
        q = q_diag            pointer to the Query
        t = t_diag            pointer to the Target
        M                     similarity matrix

        score = sum = 0;  q_beg = q

        for (;; ) {
                sum += M(*q,*t);     q++;  t++
                if (sum ≤ 0) {
                        if (score > 0) {
                                q = q_end;  t = t_end
                                if (score > S2) {
                                        /* Record the Alignment */
                                        /* Check possibly overlapping Alignments */
                                        check_overlap(q_beg, q_end)
                                }
                        }
                        if (end_of_diagonal)    break

                        q_beg = q_end = q
                        score = sum = 0
                }
                else
                        if (sum > score) {     /* new increase of score ... */
                                score = sum    /* ... before return to zero */
                                q_end = q
                        }
        }
}
```

Figure 4.6: Parcours complet de la diagonale d'un *Hit*

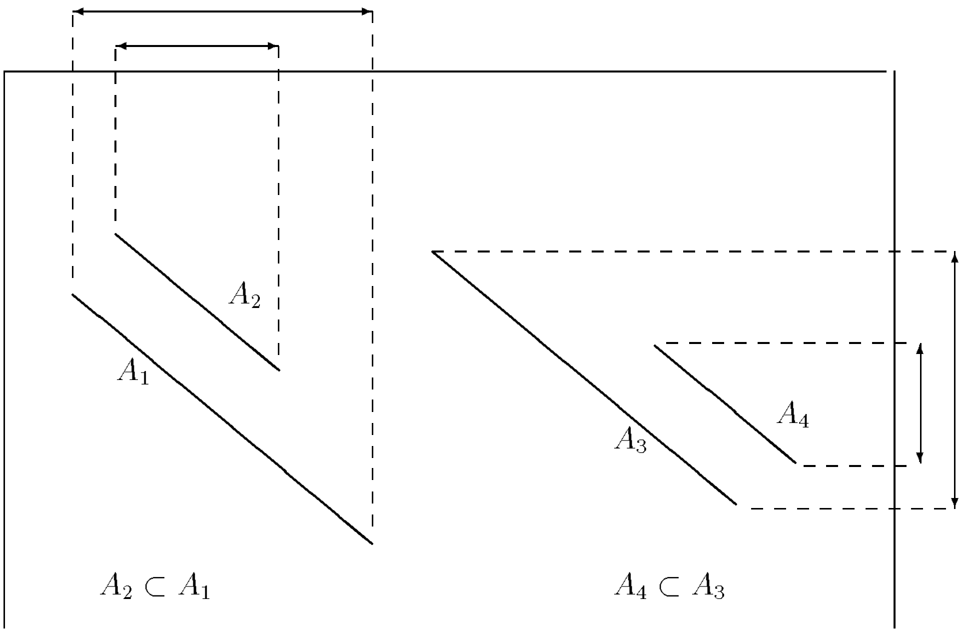


Figure 4.7: Gestion des recouvrements d'alignements (overlaps)

acide aminé quelconque a été, par construction, lors de l'initialisation de la matrice de scores, rendu très fortement négatif; le passage sur un octet nul bloque donc l'algorithme d'extension).

Un score seuil S_2 est choisi lors de la phase d'initialisation de Blast. Si, lorsqu'on a atteint les limites de l'extension, l'alignement a un score inférieur à S_2, l'alignement est rejeté; dans le cas contraire, toute la diagonale contenant contenant le *hit* est examinée (voir la figure 4.6). La diagonale est alors marquée comme ayant été entièrement parcourue.

Gestion des recouvrements. Lors du parcours complet d'une diagonale, les alignements de score supérieur au score seuil S_2 sont comparés aux alignements déjà enregistrés pour éviter des recouvrements (overlap en anglais). La figure 4.7 donne l'exemple de deux paires d'alignements se recouvrant. Blast considère que deux alignements se recouvrent si, lorsqu'on représente les alignements par des segments de diagonales, et qu'on considère les côtés du rectangle formant l'espace de recherche,

- soit la projection de l'un des alignements sur le côté correspondant à la requête est incluse dans la projection de l'autre alignement sur le même côté,

- soit la projection de l'un des alignements sur le côté correspondant à la cible est incluse dans la projection de l'autre alignement sur le même côté.

Lorsque deux alignements se recouvrent, l'alignement ayant le score le plus faible est éliminé. Pour effectuer cette gestion des alignements, Blast maintient, lors de la recherche d'alignement de la requête et d'une séquence cible, une liste chaînée des alignements retenus. Cette liste est mise à jour en cas de recouvrement de deux alignements.

Traitement probabiliste des alignements. Les alignements sont calibrés statistiquement de la manière suivante. Nous considérons une séquence requête de longueur l_r, et un ensemble de séquences cibles constituant une base de données de longueur totale L. Soit, pour une séquence donnée de longueur l_c de cette base, des alignements de scores normalisés respectifs $S'_1 \geq S'_2 \cdots \geq S'_i$, avec $S' = \lambda S - \log(K l_r l_c)$. Blast calcule la probabilité de Poisson que i alignements aient un score supérieur à s en utilisant la formule 2.78

$$p = \Pr(S'_i \geq s) \approx 1 - e^{-e^{-s}} \sum_{k=0}^{i-1} \frac{e^{-ks}}{k!}.$$

Blast calcule ensuite, pour la base cible totale, une estimation du nombre de séquences pouvant fournir un tel résultat (nombre i d'alignements de score supérieur à S_i entre une requête de longueur l_r et une cible de longueur l_c dans une base de longueur L), estimation donnée par l'équation

$$Expect = p\frac{L}{l_c}.$$

Blast donne enfin une probabilité approchée notée $P(i)$ pour les scores obtenus

$$P(i) = 1 - e^{-Expect}.$$

Blast calcule un seuil P_2 à partir d'un nombre estimé *a priori* d'alignements significatifs affichés en sortie. Cette valeur seuil va servir à l'élimination de certains alignements. L'élimination des alignements de pertinence probabiliste insuffisante se fait en deux étapes.

- Au cours de la phase recherche de l'algorithme (phase-2), les alignements de score inférieur au score seuil S_2, comme nous l'avons vu, ne sont pas retenus.
- À la fin de la recherche d'alignement avec une séquence cible, Blast calcule les probabilités $P(i)$ pour les i alignements de score supérieurs à S_2. Si, lors de ce calcul, il n'y a pas au moins un $P(i)$ inférieur à P_2, Blast élimine les alignements obtenus avec cette séquence cible.

Remarque. Les paramètres λ et K intervenant dans les calculs probabilistes sont évalués de la manière suivante.

Le paramètre λ, solution de l'équation 2.31, est calculé par recherche dichotomique.

Le paramètre K est donné par l'équation 2.32. Nous avons vu (voir paragraphe 2.3.4) que la série intervenant dans cette équation converge exponentiellement vite. Une vingtaine d'itérations donnant les premiers termes de cette série fournissent une bonne approximation pour K.

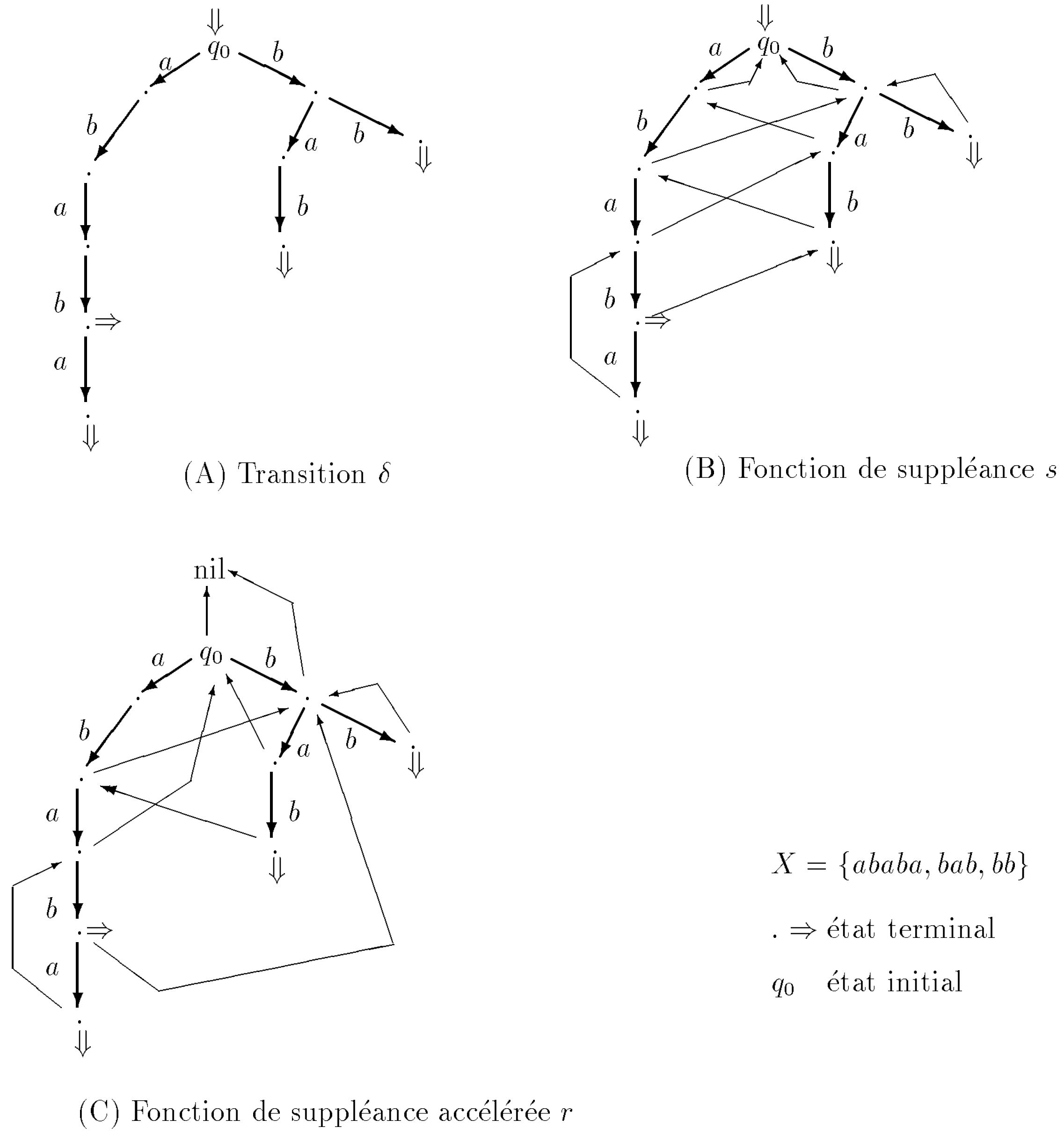


Figure 4.8: L'automate d'Aho-Corasick. (Cours de DEA, M. Crochemore). L'ensemble des mots cherchés est $X = \{ababa, bab, bb\}$.

4.3.2.3 L'algorithme d'Aho-Corasick.

Nous décrivons dans ce paragraphe l'algorithme d'Aho-Corasick utilisé dans la phase-1 de Blast. Nous suivrons la présentation de Crochemore [19], renvoyant à Crochemore et Rytter [20] pour une description approfondie de cet algorithme.

On peut poser le problème de la manière suivante. Nous considérons un alphabet A, et un texte $T \in A^\star$. Nous considérons d'autre part un ensemble fini de mots $X \subset \{A^\star\}$. Nous voulons trouver toutes les occurrences de X dans t, d'une manière efficace.

la démarche d'Aho-Corasick se décompose en plusieurs étapes.

1. Construction d'un automate en arbre T représentant l'ensemble X.

2. Construction d'un automate S ayant les mêmes états que T, et contenant à la fois les transitions de T et des transitions définies en cas d'échec.

3. Parcours du texte t en utilisant l'automate S.

Construction de l'automate en arbre T. On construit, pour l'ensemble X des mots cherchés, l'arbre $\mathcal{T}$ de la manière suivante. Les branches de $\mathcal{T}$ sont étiquetées par des éléments de X. Les nœuds de $\mathcal{T}$ sont identifiés par les préfixes des mots de X. La racine est le mot vide ϵ. Le père d'un nœud associé au préfixe non vide xa est le nœud associé au préfixe x. Les nœuds de $\mathcal{T}$ sont considérés comme les états de l'automate en arbre, dont nous appelons les transitions δ. La figure 4.8 (A) donne la fonction δ pour l'ensemble de mots $X = \{ababa, bab, bb\}$. Nous avons $\mathcal{T} = (Q, A, q_0, T, \delta)$, où Q est l'ensemble des états que nous avons défini pour l'automate en arbre, A est l'alphabet, q_0 est l'état initial, T est un ensemble provisoire d'états terminaux formés des états feuilles de l'arbre, et δ est la fonction de transition.

Utilisation d'une fonction de suppléance s. La fonction de transition δ n'est pas définie lorsque l'on "sort" d'un des mots de X. Le principe de l'algorithme d'Aho-Corasick, comme celui des algorithmes de type Knuth-Morris-Pratt [56], ou Boyer-Moore [13], consiste à tirer partie de la connaissance du texte déjà parcouru pour éviter des transitions inutiles, ou, pire, des retours arrière dans le texte. Cette connaissance du texte déjà lu et de l'ensemble X s'utilise de la manière suivante. Considérons que, si l'on a déjà trouvé un préfixe $u \in A^+$, d'un mot de X, on est dans l'automate $\mathcal{T}$ à l'état q_u. Si on lit dans le texte la lettre a, et si la transition $\delta(q_u, a)$ n'est pas définie, on peut espérer qu'un suffixe de u soit préfixe d'un mot de X. Soit $s(u)$ le plus long suffixe de u qui soit préfixe d'un mot de X. On peut alors recommencer notre recherche en l'état $q_{s(u)}$, en testant si $\delta(q_{s(u)}, a)$ est défini. Dans le cas où la réponse est positive, on continue la progression dans $\mathcal{T}$ à partir de $q_{s(u)}$. Dans le contraire, on réitère l'opération de suppléance en considérant $s(s(u)), s(s(s(u))), \ldots$, et les états associés, ceci jusqu'à ce que, soit on ait trouvé une transition δ définie pour l'état et la lettre a, soit on soit revenu à l'état initial q_0. La fonction de suppléance pour l'ensemble de mots $X = \{ababa, bab, bb\}$ est donnée dans la figure 4.8 (B). L'algorithme de balayage du texte est donné à la figure 4.9.

```
A       alphabet,    t ∈ A*,    X ⊂ {A*} (X set of words to search in text t)
q0      initial state
T       set of terminal states
δ       transition function associated to X
s       failure function

Aho-Corasick(t)
{
        q = q0
        for ( p = first to last position of t) {
                a letter at position p in t
                while (      q ≠ q0
                        &&   δ(q, a) = nil )      { q = s(q) }

                if ( δ(q, a) =  nil )    q = q0
                else                     q = δ(q, a)
                if (q ∈ T)               a word of X belongs to t, with end at position p
        }
}
```

Figure 4.9: Aho-Corasick. Le balayage du texte.

```
𝒯 = (Q, A, q0, T, δ),  tree automaton for X.
L     List of states Q ordered by increasing level

Build-Failure-Function()
{
        s(q0) = nil;     δ(nil, A) = q0
        for (q first to last element of L) {
                p'= father-of(q)
                a letter such that δ(p', a) = q
                p = s(p')
                while (δ(p, a) undefined)    p = s(p)
                s(q) = δ(p, a)
                if (δ(p, a) ∈ T)      add q to T
        }
}
```

Figure 4.10: Aho-Corasick. Construction de la fonction de suppléance s (failure function).

Construction de la fonction de suppléance s. La construction de la fonction de suppléance s se fait en temps linéaire. L'arbre correspondant à $\mathcal{T}$ est parcouru en largeur d'abord, ce qui induit que l'on accède les états suivant un ordre de liste où ils sont rangés selon une profondeur allant croissant dans l'arbre. Le calcul des suppléants amène, pour un état q associé à un mot u, à considérer un état $s(q)$ de profondeur strictement inférieure dans l'arbre. L'algorithme consiste alors à considérer le père $f(q)$ du nœud q. Si le nœud q est accédé par la lettre a, ce qui équivaut à $(\delta(f(q), a) = q)$, le suppléant de q est le nœud p tel que $\delta(p, a)$ soit défini, et que p corresponde au plus long suffixe de u qui soit préfixe d'un mot de X. On calcule p en itérant sur les suffixes de u qui sont préfixes de X. On trouvera dans la figure 4.10 l'algorithme de construction de la fonction de suppléance. On y remarquera qu'au cours de la construction de la fonction de suppléance, on complète l'ensemble des états terminaux T. Ceci est rendu nécessaire par le fait que certains états terminaux ne sont pas sur les feuilles de l'arbre $\mathcal{T}$ (par exemple l'état terminal correspondant au mot bab, et associé au mot $abab$ dans la figure 4.8 (B)). L'algorithme de construction de la fonction de suppléance est décrit dans la figure 4.10.

Accélération de la fonction de suppléance. On peut définir une nouvelle fonction de suppléance r plus rapide que la fonction s. On appellera contexte d'un état l'ensemble des lettres pour lesquelles une transition sortant de cet état est défini. On notera $C(q)$ le contexte de l'état q, et on aura donc

$$C(q) = \{a \in A / \delta(q, a) \text{ défini.}\}$$

Si on échoue à trouver une transition pour une lettre a depuis un état q, et si le suppléant $s(q)$ a même contexte que q, on échouera en cherchant une transition pour la même lettre depuis $s(q)$. Il en est de même si le contexte de $s(q)$ est inclus dans celui de q. La définition de la fonction r se fait alors de la manière suivante

$$r(q) = \begin{cases} s(q) & \begin{cases} si & C(s(q)) \not\subset C(q) \\ ou & C(s(q)) = \emptyset \end{cases} \\ \\ r(s(q)) & sinon \end{cases}$$

On trouvera la fonction de suppléance accélérée pour l'ensemble de mots $X = \{ababa, bab, bb\}$ dans la figure 4.8 (C).

Complexité. (Voir Crochemore et Rytter [20]). La complexité de l'algorithme de Aho-Corasick est:

- en espace, $O(\sum_{x \in X} |x|)$;
- en temps,
  - en préparation, $O(\sum_{x \in X} |x|)$,
  - en recherche, $O(|t|)$.

Chapitre 5

BlastMultAl. Une extension de Blast pour la recherche de similarités avec des multi-alignements.

5.1 Introduction

Nous décrivons dans ce chapitre entièrement original une extension de Blast permettant de calculer les alignements d'une séquence requête et d'un multi-alignement. Cette extension est basée sur la construction de graphes simples déduits des multi-alignements, que nous appelons graphes d'alignement. Le mécanisme d'extension de Blast est alors complété par une recherche de meilleur chemin, au sens du score correspondant dans le graphe d'alignement. Ce nouvel algorithme appelé BlastMultAl est appliqué à la recherche d'alignements avec la base de multi-alignements ProDom. Des comparaisons avec des méthodes classiques d'alignement sont faites pour la base ProDom28, qui est une base dont les multi-alignements ne contiennent pas de *gaps*. La calibration probabiliste des multi-alignements de la version 28 de ProDom a été faite par simulation directe. Nous montrerons au chapitre 6 comment effectuer la calibration probabiliste des multi-alignements de ProDom par la méthode des approximations Poissonniennes.

5.1.1 Les alignements multiples de séquences

Notre problème consiste à chercher les similarités d'un alignement multiple avec un ensemble de séquences tel que SWISSPROT, ou d'une séquence avec un ensemble d'alignements tels que ceux de ProDom.

Un alignement multiple, ou multi-alignement, de n séquences de longueurs inférieures à une longueur l peut être vu comme un tableau à deux dimensions $A_{i,k}$, avec $i = 1 \ldots l$ et $k = 1 \ldots n$. Certains éléments du tableau peuvent être des caractères de saut correspondant à des *gaps* dans l'alignement. Par construction, les alignements multiples regroupent des séquences similaires ou homologues, c'est à dire ayant un ancêtre commun dans l'histoire de l'évolution. Les acides aminés présents dans une colonne du tableau sont donc souvent assez similaires.

Nous nous limitons aux questions survenant quand les alignements multiples sont déjà obtenus, et ne considérons donc pas les différentes méthodes utilisées pour construire ces alignements.

Deux méthodes sont couramment utilisées pour faire des recherches avec des alignements multiples, les *Profiles*, et les *blocks*.

Un *profile* [35] [37] considère les acides aminés présents dans chaque colonne d'un multi-alignement. La version la plus simple des *profiles* donne le score de l'acide aminé a_j à la colonne i du *profile* en moyennant les scores de a_j avec tous les acides aminés présents à la colonne i.

Une méthode de programmation dynamique est utilisée pour aligner le *profile* et la séquence requête, avec le même principe de scores additifs que celui utilisé dans le cas de la comparaison de deux séquences.

La méthode des *blocks* de protéines de Henikoff [40] vient en filiation de celle des *profiles*. Un block est un *profile* court sans *gaps*. À une famille de protéines homologues correspond un ensemble ordonné de *blocks* séparés par des régions non alignés. Partant des familles de protéines connues, Steven et Jorga Henikoff ont construit leur base de *blocks*.

5.1.2 Rechercher des similarités avec les familles de la base ProDom

ProDom constitue l'une des plus grosses bases de données constituée d'alignements multiples. Une difficulté importante pour traiter de manière unifiée cette base tient aux variations importantes dans les dimensions des alignements; sans considérer les alignements dégénérés ne contenant qu'une séquence, ProDom contient des alignements très courts d'une vingtaine de positions comme des alignements longs de plusieurs milliers de positions, des alignements constitués de 2 séquences et d'autres constitués de plusieurs centaines de séquences. On trouve dans ProDom toutes les configurations intermédiaires, tant en nombre de séquences, qu'en longueur des domaines.

Cette diversité et la longueur assez grande des domaines (en moyenne 100 positions) rendent difficile l'utilisation d'une approche de type “block”.

Nous avons par conséquent essayé dans un premier temps d'utiliser une approche de type “*profile*”, en utilisant les *profiles* les plus simples, sans *gaps* et sans pondération. Les résultats obtenus, que nous détaillerons ultérieurement dans ce chapitre, n'ont pas donné un avantage évident à cette approche, comparée à la recherche de similarité avec les séquences consensus des domaines. Nous avons donc été conduit à utiliser une approche nouvelle, qui utilise des graphes d'alignement simples. Nous décrivons ici cette approche, ainsi que les résultats des comparaisons faites avec les consensus pondérés et non pondérés, et avec les *profiles* pondérés et non pondérés, ainsi que les comparaisons faites avec la méthode naïve des séquences “à plat”.

Dans une première étape, les *gaps* n'ont pas été pris en compte, et les résultats de comparaisons faits dans ce chapitre concernent des méthodes sans *gaps*. Nous avons cependant récemment étendu notre approche au cas des *gaps*.

L'intuition biologique sous-tendant notre approche revient à dire qu'un alignement multiple est constitué d'un squelette formé des parties très bien conservées et très contraintes de l'alignement, pour lesquelles le consensus constitue une bonne représentation, autour desquelles les parties variables et moins contraintes fournissent des choix de sous-séquences possibles; on est alors conduit à rechercher la combinaison optimum de ces choix pour modéliser la recherche de similarité.

Cette approche conduit à la construction de graphes. De même que les *profiles*, ces graphes d'alignement ont deux applications possibles:

1. chercher les similarités d'une séquence requête et de multi-alignements appartenant à une base de données de multi-alignements;

2. chercher les similarités d'un alignement multiple et de séquences appartenant à une base de données de séquences.

On pourrait aussi utiliser les graphes d'alignement pour étudier une troisième application que nous n'avons pas abordé, chercher les similarités d'un multi-alignement et de multi-alignements appartenant à une base de données d'alignements.

Nous avons développé suivant cette approche de graphes d'alignement une extension de Blast appelée BlastMultAl qui répond aux objectifs des deux premières applications énoncées.

5.2 Graphes d'alignement et méthode du meilleur chemin

5.2.1 Présentation générale de la méthode

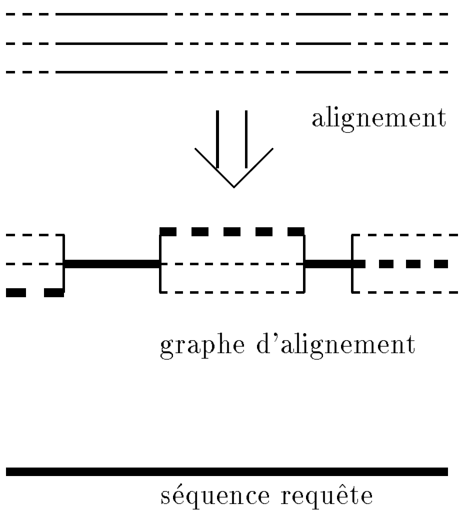
Comme le fait l'approche de type "blocks", nous considérons que les régions bien conservées sont jointes par des régions mal conservées, ou plutôt moins bien conservées, et moins contraintes. Nous effectuons à ce point une approche complémentaire de l'approche "blocks". Nous prenons le consensus des séquences pour représenter les régions conservées, mais nous représentons les régions faiblement conservées par un ensemble de sous-séquences parallèles, ou branches. Quand nous cherchons à étendre un alignement avec une séquence requête, et que nous traversons une région faiblement conservée, nous essayons successivement les branches, et nous conservons, pour l'alignement considéré, celle qui s'aligne le mieux avec la séquence requête.

Nous décrivons au paragraphe 5.2.2.1 comment nous distinguons les positions conservées des positions faiblement conservées dans un alignement. Nous y discutons également les choix heuristiques effectués pour garder la méthode calculatoirement efficace.

Nous définissons à présent, de manière non formelle, le graphe d'alignement d'un alignement multiple de la manière suivante.

- Dans une première étape, nous définissons les régions conservées et nous prenons le *consensus* de ces régions. Nous parlerons de *régions consensus* pour ces régions. Nous détaillerons au paragraphe 5.2.2 la construction de ces régions qui fait appel aux matrices de similarité.

- Nous appelons *branche* une sous-séquence d'une séquence s'étendant d'un bout à l'autre d'une région faiblement conservée. Nous appellerons *régions de branches* de telles régions.




```
BBBBNNNNNNBBBBBBBBEEEEBBBBBBB
FFFFNNNNNNFFFFFFFFZZZZFFFFFFF
CCCCNNNNNNCCCCCCCCEEEECCCCCCC

BBBB      BBBBBBBB    BBBBBBB
FFFFNNNNNNFFFFFFFFEEEEFFFFFFF
CCCC      CCCCCCCC    CCCCCCC

CCCCNNNNNNBBBBBBBBEEEEFFFFFFF
```

Figure 5.1: Un *Graphe d'alignement* et un *Meilleur-Chemin*. Dans les régions conservées, le graphe d'alignement est constitué du consensus des séquences du multi-alignement, (représenté en traits pleins); des arêtes parallèles, ou branches, ont été construites dans les régions moins bien conservées (et représentées en pointillé). La recherche d'alignement avec ce graphe est équivalente à une recherche d'alignement avec une longue séquence constituée en autorisant toutes les combinaisons des branches, autour des consensus. Le meilleur chemin est représenté en gras dans la partie gauche de la figure. (Remarque. La partie droite de la figure correspond à un exemple jouet avec un modèle de match exact alors que nous effectuons pratiquement dans nos applications des matchs approchés, comme Blast.)

- Pour la construction de notre graphe, nous considérons que les extrémités des branches sont reliées aux extrémités des séquences consensus qui leur sont contiguës, en respectant l'ordre du multi-alignement.

- En considérant un multi-alignement sans *gap*, un *chemin admissible* est formé de la concaténation d'acides aminés à des positions contiguës, en respectant la condition qu'à l'intérieur d'une région conservée, une fois une branche choisie, tous les acides aminés choisis pour cette région appartiennent à la même branche.

La figure 5.1 montre sur sa partie droite un multi-alignement, et sur sa partie gauche, le schéma du graphe d'alignement correspondant.

Une fois défini le graphe d'alignement, en utilisant une matrice de scores, $\mathcal{M}$, et un schéma de scores additifs, on peut définir le score de l'alignement d'un chemin admissible et d'une séquence requête. Le choix du *meilleur chemin* consiste à choisir localement le chemin admissible donnant le meilleur alignement, suivant un principe de programmation dynamique. Ce meilleur chemin est celui qui aura, localement, le score le plus élevé.

5.2.2 Détermination des régions conservées et des régions faiblement conservées.

5.2.2.1 Définition d'un coefficient de conservation pour une position de multi-alignement

Nous décrivons à présent comment, au niveau d'une position, nous faisons la distinction entre position conservée et position non-conservée.

Définition 42. *Considérons un multi-alignement de n séquences de longueur l.*

- *Soit a_{ij} l'acide aminé à la position i dans la séquence j.*
- *Soit $s(a_k, a_l)$ le score de similarité de deux acides aminés a_k et a_l dans une matrice de similarité $\mathcal{M}$.*
- *Soit c_i l'acide aminé consensus à la position i, tel que $\Sigma_{k=1}^{n} s(a_{ik}, c_i)$ est maximal.*
- *Soit $\overline{c_i}$ l'acide aminé anti-consensus à la position i, tel que $\Sigma_{k=1}^{n} s(a_{ik}, \overline{c_i})$ est minimum.*

nous définissons alors le coefficient de conservation τ_i pour la position i comme

$$\tau_i = \frac{1}{n} \times \frac{\Sigma_{k=1}^{n}[s(a_{ik}, c_i) - s(\overline{c_i}, c_i)]}{s(c_i, c_i) - s(\overline{c_i}, c_i)} \tag{5.1}$$

Dans cette dernière définition, le dénominateur de la fraction définissant τ_i vaut n fois la variation maximum de score de l'acide aminé dans la matrice de similarité. Par construction, le coefficient τ est positif et prend des valeurs inférieures ou égales à 1, prenant la valeur 1 pour une position parfaitement conservée.

Nous choisissons à présent un niveau seuil $\tilde{\tau}$, constant pour le multi-alignement. Suivant que $\tau > \tilde{\tau}$ ou non, nous dirons que la position est conservée ou non. Différentes stratégies peuvent être adoptées quand on considère un ensemble de multi-alignements. On peut choisir une valeur unique $\tilde{\tau}$ pour tous les multi-alignements, ou, au contraire, choisir un seuil $\tilde{\tau}$ dépendant des classes de multi-alignements, obtenues en regroupant dans une même classe les multi-alignements contenant le même nombre de séquences.

Les résultats statistiques du prochain paragraphe montrent que, si on veut définir un τ constant pour tous les multi-alignements de ProDom28 avec la matrice de similarité BLOSUM62, le choix empirique de $\tilde{\tau} = 95\%$ est raisonnable pour notre application.

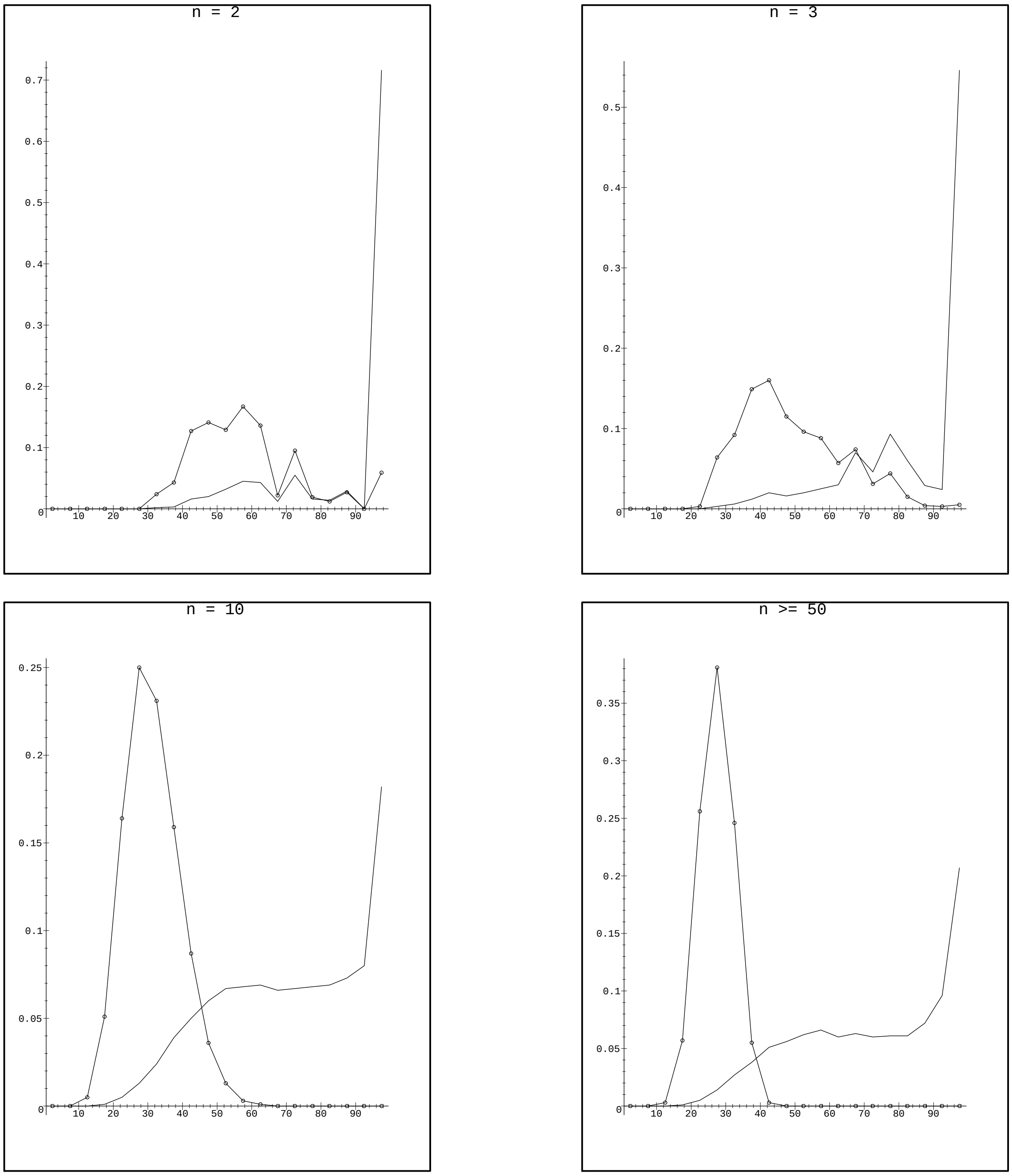


Figure 5.2: Distribution de τ pour les multi-alignements de ProDom28 (courbes non marquées) et pour des multi-alignements aléatoires (courbes marquées par des points). Les calculs sont faits avec BLOSUM62 comme matrice de score. Les multi-alignements constitués de n séquences sont regroupés dans une même classe. La figure donne les classes $n = 2, 3, 10$ et $n \geq 50$.

5.2.2.2 Analyse statistique du coefficient τ pour ProDom28

Nous considérons les multi-alignements de ProDom28, et nous utilisons la matrice BLOSUM62 comme matrice de similarité $\mathcal{M}$. Nous étudions dans ce paragraphe les résultats statistiques obtenus avec ces données pour le coefficient de conservation τ donné par l'équation 5.1.

Nous avons calculé la distribution de ce paramètre dans les cas où les multi-alignements sont composés de n séquences, pour les valeurs $n = 2, 3, 10$, et $n \geq 50$ (les multi-alignements de plus de 50 séquences sont regroupés pour ce travail dans la même analyse). Nous avons aussi calculé la distribution de τ pour des multi-alignements générés aléatoirement, pour les mêmes valeurs de n, et comparé les résultats obtenus dans ces deux cas, multi-alignements réels, et multi-alignements aléatoires. La figure 5.2 donne ces distributions. On y remarquera que les valeurs de τ pour $n = 2$ (resp. $n = 3$) sont fortement corrélées à la distribution des scores de la matrice BLOSUM62. Ceci explique le pic obtenu pour $\tau = 72\%$ (resp. $\tau = 78\%$). Les pics obtenus pour $\tau \geq 95\%$ dans tous le cas des multi-alignements de ProDom28 reflètent le fait que les acides aminés d'une colonne d'un alignement sont assez largement similaires, et qu'ils sont très similaires pour les colonnes d'une position bien conservée. Cependant, comme on peut s'y attendre, quand le nombre de séquences des multi-alignements de ProDom28 croît, la probabilité d'avoir un τ élevé diminue. Pour ProDom28, nous avons fait le choix empirique, à partir de cette analyse statistique, de considérer comme conservées (resp. mal conservées) les positions pour lesquelles on a $\tau > \tilde{\tau} = 95\%$ (resp. $\tau \leq \tilde{\tau}$).

5.2.3 Cadre probabiliste et choix heuristiques

Nous avons développé au chapitre 2 la théorie mathématique de l'alignement local optimal, qui est utilisée par Blast. Si nous considérons une suite $\{X_i\}$ de variables aléatoires i.i.d, la section maximum de la marche aléatoire sur $[0, l]$ vérifiera l'équation 2.1 que nous rappelons

$$Pr\{\mathcal{S}(l) > \frac{\log(l)}{\lambda} + x\} = 1 - e^{-Ke^{-\lambda x}}, \tag{5.2}$$

où on a

$$S_k = \sum_{i=1}^{k} X_i, \quad \text{et} \quad \mathcal{S}(l) = Max_{1 \leq j < k \leq l}(S_k - S_j).$$

Ces résultats sont valables à condition que la variable incrément X vérifie $E(X) < 0$ et $\{\Pr(X > 0) > 0\}$. D'autre part, λ est la solution positive de l'équation en t

$$E(e^{tX}) = 1, \tag{5.3}$$

et K est donné par une formule calculable (formule 2.32). Il est indispensable de tenir compte de ces conditions, et en particulier de la condition $\{E(X) < 0\}$ pour notre algorithme de meilleur chemin. En simplifiant, nous pouvons utiliser le modèle suivant. Soit X la variable aléatoire élémentaire, utilisée dans l'alignement de deux séquences. Nous allons nous intéresser au score $B_k = \sum_{i=1}^{k} X_i$ d'une branche de longueur k pour un alignement de la séquence requête et du multi-alignement cible, et au score R_{hk} obtenu en prenant la meilleure branche dans une région formée de h branches. Nous souhaitons que l'espérance de ce dernier score R_{hk} soit négative. Soit μ l'espérance $E(X)$ et σ^2 la variance de X. Nous avons $E(B_k) = k\mu$ et $\mathrm{Var}(B_k) = k\sigma^2$. En prenant des approximations gaussiennes, qui sont très vite vérifiées, pour les sommes partielles des variables X, nous obtenons une approximation de la densité de la variable B_k

$$P(B_k = x) \approx \phi_k(x) = \frac{1}{\sigma\sqrt{2k\pi}} e^{\frac{(x-k\mu)^2}{k\sigma^2}},$$

et sa fonction de répartition

$$P(B_k < x) \approx \Phi_k(x) = \int_{-\infty}^{x} \frac{1}{\sigma\sqrt{2k\pi}} e^{\frac{(t-k\mu)^2}{k\sigma^2}} dt.$$

En supposant, ce qui est une supposition très pessimiste, que les variables B_k sur les branches sont distribuées de manière indépendante, la fonction de répartition $\Theta_{hk}(x)$ de R_{hk} vaut

$$\Theta_{hk}(x) \approx \Phi_k(x)^h,$$

et sa densité $\theta(x)$ vaut

$$\theta_{hk}(x) \approx h\Phi_k(x)^{h-1}\phi_k(x).$$

Nous en déduisons

$$\chi(k,h) = E(R_{hk}) \approx \int_{-\infty}^{\infty} hx\Phi_k(x)^{h-1}\phi_k(x)dx.$$

Le tableau suivant donne les valeurs de $\chi(k,h)$ pour $1 \leq k \leq 8$ et $1 \leq h \leq 7$.

$$\chi(k,l) = E(\max_{i=1}^{h} S_k) = E(\max_{i=1}^{h} \sum_{j=1}^{k} U_j)$$

		k							
		8	7	6	5	4	3	2	1
h	1	-7.44	-6.51	-5.58	-4.65	-3.72	-2.79	-1.86	-0.93
	2	-4.13	-3.41	-2.71	-2.03	-1.38	-0.76	-0.20	0.70
	3	-2.47	-1.86	-1.28	-0.72	-0.21	0.25	0.62	0.83
	4	-1.40	-0.86	-0.35	0.13	0.55	0.91	1.16	1.21
	5	-0.61	-0.12	0.33	0.75	1.11	1.39	1.55	1.61
	6	0.11	0.45	0.86	1.23	1.54	1.77	1.80	1.85
	7	0.50	0.92	1.30	1.63	1.89	2.07	2.11	2.16

$$U = \mathcal{N}(\mu, \sigma^2)$$
$$\mu = E(X) = -0.93, \quad \sigma^2 = E(X^2) - E(X)^2 = 4.41$$

Fréquence de fond des acides aminés: SWISSPROT28
Matrice de Similarité: BLOSUM62

Dans ce tableau, pour un k donné, χ croît avec h, et pour un h donné, χ croît quand k décroît.

Pour éviter d'obtenir une trop grande dérive positive de l'espérance des variables aléatoires R dans les régions de branche (dérive correspondant à des χ positifs), nous avons fait deux choix empiriques:

1. ne pas prendre de branches de longueur inférieure à 3 ($k \geq 3$);

2. limiter le nombre de branches parallèles à 5 ($h \leq 5$).

Ces choix correspondent dans le tableau donné précedemment à une zone d'espérance positive, mais, compte tenu des corrélations fortes entre séquences dans un multi-alignement, ces choix sont raisonnables. La dernière contrainte ($h \leq 5$) nous obligera à utiliser une méthode de réduction du nombre de branches quand le nombre de séquences dans le multi-alignement est supérieur à 5; ceci se fera en regroupant les branches les plus similaires en une seule branche par un algorithme que nous détaillerons au paragraphe 5.3. En résumant nos choix heuristiques, avec BLOSUM62 comme matrice de score, une région de branches s'étendra sur au moins 3 positions contiguës ayant chacune un coefficient de conservation $\tau > 95\%$, et sera constituée d'au plus 5 branches. Le tableau 5.1 donne un aperçu du résultat du pré-traitement sur la base ProDom28.

Les expérimentations pratiques montrent que, en faisant ces choix heuristiques, le modèle de Karlin est bien applicable. Par contre, les paramètres K et λ intervenant dans l'équation 5.2 devront être calculés pour chaque multi-alignement (voir paragraphe 5.4.2.2), ce qui correspond à la phase de calibration des *blocks* de Henikoff-Henikoff.

nombre de séquences dans les multi-alignements	nombre de multi-alignements	nombre de régions de de branches	nombre de positions des régions de branches	nombre total de positions	pourcentage de positions dans les régions de branches
1	15074			2595435	0.0
2	3046	16031	68697	547066	12.56
3	1370	8682	45549	243514	18.70
4	836	5444	30566	168549	18.13
5	506	3906	24325	99597	24.42
6	336	3028	21679	72094	30.07
7	274	2705	21197	61398	34.52
8	183	1998	17098	43195	39.58
9	161	1706	14323	34913	41.02
≥ 10	1319	12991	147659	260516	56.68
≥ 2	8031			983776	

Tableau 5.1: Vue d'ensemble de ProDom28 et des résultats de pré-traitement avec le seuil $\tilde{\tau} = 95\%$ (les branches de longueur inférieure à 3 ont été éliminées).

5.3 Implémentation

Nous présentons à présent les aspects algorithmiques de notre approche. L'architecture générale est semblable à celle de Blast. Elle est donnée par le tableau suivant.

1-	Prétraitement - Génération des graphes d'alignement
2-	Traitement *on-line* *a*) Parcours de la requête et génération des mots courts de score élevé *b*) Parcours du graphe d'alignement et extension des *hits*

Le pré-traitement "`SetdbMultAl`" est effectué sur la base, qui dans notre cas est une base de multi-alignements, alors que dans le cas de Blast, le pré-traitement "`Setdb`" est effectué sur une base de séquences. Cependant, alors que le pré-traitement de Blast est surtout une mise en forme des données, sans véritable conséquences algorithmiques, notre pré-traitement définira les positions conservées et celles qui sont dans les régions de branches, et traitera la réduction éventuelle du nombre de branches; il représente donc plus qu'une simple mise en forme des données.

```
---------------------------------------------------------------------------
>1284 (10) DEHYDROGENASE (IMP INOSINE-5'-MONOPHOSPHATE DEHYDROGENASE)

GUAC_ASCLU 105  331   ..NGYSEVFVDFIRRVREQFPTHTIFAGNVVTGEMVEELILSGADVVKVGIG..
GUAC_ECOLI 91   327   ..NGYSEHFVQFVAKAREAWPTKTICAGNVVTGEMCEELILSGADIVKVGIG..
IMP1_HUMAN 30   514   ..QGNSVYQIAMVHYIKQKYPHLQVIGGNVVTAAQAKNLIDAGVDGLRVGMG..
IMP2_HUMAN 30   514   ..QGNSIFQINMIKYIKDKYPNLQVIGGNVVTAAQAKNLIDAGVDALRVGMG..
IMP_ACICA  9    455   ..HGHSAGVIERVRWVKQNFPQVQVIGGNIATGDAALALLDAGADAVKVGIG..
IMP_BACSU  11   455   ..HGHSQGVLNTVTKIRETYPELNIIAGNVATAEATRALIEAGADVVKVGIG..
IMP_ECOLI  32   477   ..HGHSEGVLQRIRETRAKYPDLQIIGGNVATAAGARALAEAGCSAVKVGIG..
IMP_LEIDO  28   510   ..QGNTIYQIAFIKWVKSTYPHLEVVAGNVVTQDQAKNLIDAGADGIRIGMG..
IMP_MESAU  30   514   ..QGNSIFQINMIKYMKEKYPNLQVIGGNVVTAAQAKNLIDAGVDALRVGMG..
IMP_MOUSE  30   514   ..QGNSIFQINMIKYIKEKYPSLQVIGGNVVTAAQAKNLIDAGVDALRVGMG..
----------------------------------------------------------------------------
conserved positions 1 .. C                C      CCC C      C   C     CC C..
----------------------------------------------------------------------------
conserved positions 2 .. C                C      CCCCC      C   C     CCCC..
----------------------------------------------------------------------------
consensus sequence    ..HGHSVYQINMIKYVKEKYPNLQVIGGNVVTAEQAKNLIDAGADAIRVGMG..
----------------------------------------------------------------------------
                      ..N YSEVFVDFIRRVREQF THTIFA     GEMVEE ILS ADVVK    ..
                      ..Q NSVYQIAMVHYIKQKY TKTICA     AAQAKN IDA VDGLR    ..
Alignment-Graph       ..HGHSAGVIERVRWVKQNFPHLQVIGGNVVTGDAALALLDAGVDALRVGMG..
                      ..H HSQGVLNTVTKIRETY QVQVIG     AEATRA IEA CSAVK    ..
                      ..H HSEGVLQRIRETRAKY ELNIIA     AAGARA AEA ADGIR    ..
---------------------------------------------------------------------------
```

Figure 5.3: Une partie (positions 270 à 319) de l'alignement 1284 de ProDom28, et la sortie correspondante après le pré-traitement; les branches trop courtes de l'étape 1 ont été remplacées par le consensus dans l'étape 2. Les positions considérées comme conservées ont un coefficient de conservation τ tel que $\tau > \tilde{\tau} = 95\%$. Après l'étape 1, 21.4% des positions sont considérées comme conservées, et 30.0% après l'étape 2.

L'architecture du traitement principal de notre méthode (“`BlastMultAl`”) est elle aussi largement inspirée de celle du traitement principal de Blast (“`Blastp`”), mais l'extension des points d'ancrage obtenus par les mots courts de scores élevés est sensiblement différente de celle faite dans Blast. Nous détaillons dans les deux paragraphes suivants le pré-traitement et le traitement principal.

5.3.1 Étape de pré-traitement. Construction des Graphes d'alignement

Cette étape a deux fonctions.

1. La définition des positions bien conservées et de celles qui ne le sont pas. Ceci est très simple, une fois la valeur de $\tilde{\tau}$ choisie. La détection des branches trop courtes et leur remplacement par le consensus sont aussi faciles.

2. La réduction éventuelle du nombre de branches parallèles.

Nous décrivons cette dernière opération.

Nous voulons réduire le nombre de branches d'une région de branches en regroupant les branches similaires dans des classes, et en prenant pour chaque classe le consensus de la classe. Nous détaillons l'algorithme permettant de réduire le nombre de classes de k à $k-1$.

Nous considérons une partition de notre ensemble de n branches b_i en k classes $S_j, j = 1 \ldots k$. Nous avons

étape de réduction:	
1-	prendre le consensus C_j de chaque classe S_j;
2-	pour chaque paire l, m, avec $1 \leq l < m \leq k$, calculer le score $S(C_l C_m)$, somme des scores des positions individuelles; choisir la paire l', m' qui a le score le plus élevé;
3-	fusionner les classes $S_{l'}$ et $S_{m'}$

L'initialisation du précédant algorithme se fait en considérant une partition dans laquelle chaque classe S_j contient une branche et une seule. Nous itérons ensuite la réduction du nombre de classes jusqu'à obtenir 5 classes, le nombre choisi de manière heuristique dans la précédente section.

Exemple de pré-traitement. La figure 5.3 fournit un exemple typique de pré-traitement, appliqué à une section du multi-alignement 1284 de ProDom28, multi-alignement constitué de 10 séquences. On voit dans le bas de la figure le graphe d'alignement associé à ce multi-alignement.

5.3.2 Traitement principal. Recherche d'un alignement d'une séquence requête et d'un graphe d'alignement

Ce traitement est constitué de deux phases. La deuxième phase est elle-même constituée de trois étapes. La première phase et la troisième étape de la deuxième phase sont identiques aux phases et étapes correspondantes de Blast (voir le paragraphe 4.3.2.2).

1. *Phase 1.* Un automate multi-chaînes de type Aho-Corasick est construit sur tous les mots courts m_a ayant un score élevé (au dessus d'un seuil T) avec un des mots courts m_r de la requête. À chaque mot court m_a est associé une liste de pointeurs vers les mots courts m_r de la requête ayant avec lui un score élevé.

2. *Phase 2.*

   (a) *Étape 2.a.* Détection des hits dans les graphes d'alignement.

   Un hit est un alignement de score supérieur au seuil T d'un mot court de la requête avec un mot court du graphe d'alignement. Pour des raisons de simplicité et d'utilisation efficace de l'automate d'Aho-Corasick, nous considérons des séquences de travail, en nombre égal au nombre de branches dans les régions de branches. Ces séquences de travail sont constituées en considérant, pour chaque région de branches, la branche de même numéro (dans une représentation de l'alignement comme celle de la figure 5.3, ces branches sont situées sur une même ligne). Nous concaténons alors, dans l'ordre de traversée du graphe une branche pour chaque région de branche et le consensus pour les régions consensus. C'est sur ces séquences de travail que l'on passe l'automate Aho-Corasick pour chercher les *hits*.

   (b) *Étape 2.b.* Extension des hits.

   Il s'agit, une fois trouvé un hit, de voir si on peut l'étendre à un appariement de score élevé HSP (de l'anglais High Scoring Pair). Les HSPs sont définis par quatre paramètres. (1) La position relative , (qui peut avoir une valeur négative), de l'extrémité gauche de la requête par rapport à l'extrémité gauche du graphe d'alignement. (2) La position du début du HSP. (3) La position de fin du HSP. (Ces deux dernières positions peuvent être choisies indifféremment dans la séquence requête, ou dans le graphe d'alignement). (4) Le score du HSP.

   (c) *Étape 2.c.* Cette dernière étape consiste à trier les HSPs, et à éliminer ceux qui ont une faible pertinence probabiliste.

Le principe de l'extension effectuée à l'étape 2.b est simple. Celle-ci est d'abord effectuée par une extension à droite. Chaque fois que l'on traverse une région de branches, on essaye toutes les branches, et on ne conserve que la meilleure branche. La réalisation de ces extensions demande de mémoriser cependant plusieurs paramètres: on peut atteindre un record sur une branche qui a ensuite un mauvais score. La branche ayant le meilleur score de traversée n'est donc pas nécessairement celle sur laquelle le record a été obtenu. Il faut mémoriser ces deux événements pour d'une part pouvoir poursuivre l'extension avec le meilleur score de traversée de la région, et d'autre part pouvoir revenir au record si la suite de l'extension échoue.

L'implémentation demande aussi de différencier l'initialisation de l'extension, selon que le mot court (supposé de longueur 3) dont on part est entièrement compris dans une région consensus, fait partie d'une région consensus et d'une région de branches, ou est à cheval sur deux régions de branches et une région de consensus de longueur 1.

```
Mark beginning and end of consensus regions with a sentinel byte.
Assign a working string of the length of the region to each branch region.
Put a sentinel byte at beginning and end of these strings.

BlastMultAl_Diagonal_Extension(diag)
diag;                   /* Diagonal of a successfull Hit Extension */
{
    /* Process the Alignment-Graph from left to right according to */
    /* the diagonal positions shift */

    ... score += ...        /* Blast_Extension along the consensus. */
                            /* Increment variable ``score'' */
                            /* If any successfull alignment */
                            /* memorize record score and */
                            /* corresponding alignment position */
    if (sentinel byte) {    /* beginning of branch region */
        r_score = -∞        /* 'r_score' will be the best score at end of */
                            /* the scanned branch region;  this may be */
                            /* different from the record for the region */
        foreach branch of the region {
            Copy branch to working string
            b_score = score         /* 'b_score' is the local variable */
                                    /* for extension in branch regions */
            Blast_Extension(working string) {   /* fails on sentinel byte */
                                    /* If any successfull alignment */
                                    /* memorize record score and */
             ... b_score += ...     /* corresponding alignment position */
            }
            if (b_score > r_score)     r_score = b_score
        }
        if (r_score > 0)     score = r_score
        else                 score = 0
    }
    /* Continue Blast_Extension on consensus segment */
}
```

Figure 5.4: L'algorithme BlastMultAl. Extension le long de la diagonale d'un Hit.

Nous détaillons dans la figure 5.4 l'algorithme d'extension de BlastMultAl le long d'une diagonale. On pourra consulter la figure 4.6 décrivant l'algorithme d'extension de Blast le long d'une diagonale mentionné comme "Blast_Extension" dans la figure 5.4. La boucle externe permettant d'itérer suivant le nombre des régions de branches a été déroulée. L'extension d'un Hit à droite est identique à l'extension le long d'une diagonale, à ceci près que cette extension est interrompue en cas de trop forte décroissance du score, comme dans le cas de Blast. L'extension d'un Hit à gauche est symétrique de l'extension à droite.

5.4 Applications

Comme nous l'avons mentionné précédemment, on peut tirer deux applications principales de notre algorithme de meilleur chemin, qui sont aussi les deux applications des *Profiles*.

- La première application concerne la recherche de similarité avec une famille donnée. On connaît une famille de protéines qui partagent des similarités suffisantes pour former un multi-alignement. On veut chercher dans une base de séquences des séquences appartenant aussi à la famille, en cherchant les séquences qui ont une bonne similarité avec le multi-alignement. Nous donnons dans le paragraphe 5.4.1 les résultats d'une comparaison de six méthodes différentes appliquées à une recherche de similarités avec un sous-ensemble de la famille des globines.

- La seconde application prend une perspective opposée. Elle consiste à chercher les similarités entre une séquence et une base de données de multi-alignements. Nous exposons l'application de cette approche à la base de données ProDom28 dans le paragraphe 5.4.2.

5.4.1 Recherche de similarités d'un multi-alignement et d'un ensemble de séquences (analyse par famille). Comparaisons de différentes méthodes

Nous restreignons la comparaison faite dans ce paragraphe au cas de méthodes de recherche de similarité sans *gaps*.

Les figures 5.5, 5.6, 5.7, 5.8, 5.9 et 5.10 donnent les résultats obtenus avec les six méthodes suivantes:

- (A) Méthode de Meilleur Chemin avec des Graphes d'alignement (BlastMultAl).

- (B) Consensus Non-pondéré.

- (C) Consensus pondéré.

- (D) Profile Non-pondéré (score moyenné sans pondération à chaque position).

- (E) Profile Pondérés (score moyenné avec pondération à chaque position; la pondération est indépendante de la position).

- (F) Séquences à Plat. En considérant n séquences S_m dans le multi-alignement, qui est la requête, pour chaque séquence cible testée S_t, on sélectionne la séquence S_m ayant le meilleur score, et on mémorise ce score.

5.4.1.1 Schémas de pondération

Les schémas de pondération utilisés pour le calcul des consensus du multi-alignement de globines utilisé et des scores de *profiles* associés à ce multi-alignement ont été obtenus en utilisant ClustAlv [42] et TreeWgt [30].

ClustAlv produit un arbre binaire de type phylogénique pour les séquences d'un multi-alignement. À chaque séquence est associée un nœud de l'arbre, et la topologie de l'arbre donne le degré de similarité de deux séquences (les séquences topologiquement les plus proches sont les plus similaires).

En considérant un tel arbre topologique en entrée, TreeWgt donne des poids pour chacune des séquences, et ces poids sont utilisés pour calculer les consensus pondérés et les scores de *profiles* pondérés.

5.4.1.2 Description des tests

Chaque séquence de rang impair dans le multi-alignement #1 de ProDom28 (famille des globines) a été sélectionné pour construire un multi-alignement, la Famille-Échantillon. Les séquences de test sont:

- les séquences globines de la base de protéines SwissProt32 qui n'appartiennent pas au domaine #1 de ProDom28 (66 séquences);
- des séquences non-globines choisies au hasard dans SwissProt32 (1875 séquences).

Pour chaque méthode, des scores normalisés, ou Z'-scores ont été calculés comme suit.

1. Pour chaque séquence de test, l'alignement avec la Famille-Échantillon produisant le meilleur score S a été conservé.
2. L'espérance $E(S)$ et l'écart-type $V(S)$ de ces scores pour les séquences non-globines a été calculé.
3. Un Z-score a été calculé, comme
$$Z = \frac{S - E(S)}{V(S)}.$$
4. Le modèle de Karlin nous dit qu'une régression linéaire appliquée au nuage de points de coordonnées $\{\log(L), Z\}$ où L est la longueur des séquences test, est une droite de pente $\frac{1}{\lambda}$. Nous pouvons calculer λ par cette régression linéaire. Nous définissons alors le Z'-score comme
$$Z' = Z - \frac{1}{\lambda}\log(L),$$
pour éliminer les déviations dues aux différences de longueur entre les séquences.

5.4.1.3 Discussion des graphiques

Nous discutons les figures 5.5 à 5.10. Les 66 séquences globines de test sont représentées par des cercles noirs, et les 1875 séquences non globines de test par des croix grises.

La plupart des séquences globines de test ont une longueur proche de 150, ce qui donne de nombreux cercles au voisinage de la ligne verticale $\log(L) = \log(150) \approx 5$.

Les six méthodes séparent bien sept globines fortement similaires des globines de la Famille-Échantillon. Pour BlastMultAl, deux de ces sept globines ont même longueur et même score ($Z' = 23.4$), ce qui donne lieu à une superposition des cercles sur la figure 5.5 (voir le tableau 5.4 donnant les Z'-scores).

Pour toutes les méthodes, il y a une partie des globines de test qui ne sont pas reconnues comme similaires à la Famille-Échantillon; elles correspondent aux cercles qui restent profondément à l'intérieur du nuage de fond représentant les protéines choisies aléatoirement.

Il est à noter également que BlastMultAl donne des scores moins élevés aux protéines très fortement similaires de notre Famille-Échantillon. Ceci n'est pas un inconvénient majeur, car ces similarités sont les plus faciles à détecter.

5.4.1.4 Comparaisons par ordre de meilleur rang

La figure 5.11 montre combien de non-globines sont trouvées dans les 30 scores les plus élevés avec les six méthodes considérées au paragraphe 5.4.1. C'est BlastMultAl qui fournit le moins de non-globines dans les 30 meilleurs scores, ce qui est un bon résultat pour cette approche. C'est au contraire la méthode simpliste des séquences à plat qui en fournit le plus. Pour chaque protéine correspondant à un rectangle blanc dans cette figure, cette table donne aussi la méthode qui fournit le meilleur Z'-score. Aussi bien pour les consensus que pour les *profiles*, cette figure montre que la pondération des séquences améliore les résultats.

Les *profiles* pondérés donnent le plus souvent le meilleur Z'-score pour les protéines de la figure 5.11. Cependant, le logiciel BlastMultAl détecte mieux les similarités assez distantes entre la Famille-Échantillon et plusieurs protéines (lettres -A dans la première colonne de la figure 5.11. BlastMultAl est donc une méthode bien complémentaire de la méthode des *profiles* pondérés qui apparaît comme la plus sensible.

5.4.1.5 Comparaisons globales des résultats de Z'-score

Le tableau 5.2 compare pour les six méthodes considérées les proportions de Z'-scores dans les intervalles de valeurs $[0, 2]$, $[2, 3]$, and $[3, \infty]$.

Le nombre de vrais positifs au-dessus de $Z' = 3$ est voisin pour BlastMultAl (0.364), et pour les méthodes de *profiles* (0.364 et 0.348), mais c'est BlastMultAl qui a le plus petit nombre de faux positifs, avec un $Z' > 3$ (0.007). Dans l'intervalle des valeurs de Z'-scores moyen ($[2, 3]$), les résultats de BlastMultAl ne sont pas excellents. Quand on considère $Z' \geq 2$, les *profiles* trouvent plus de vrais positifs (0.546 et 0.560) que BlastMultAl (0.485). La méthode des séquences à plat a de très mauvais résultats pour les vrais positifs (0.182).

Ces comparaisons mettent en évidence les gains obtenus en sophistiquant un peu les méthodes.

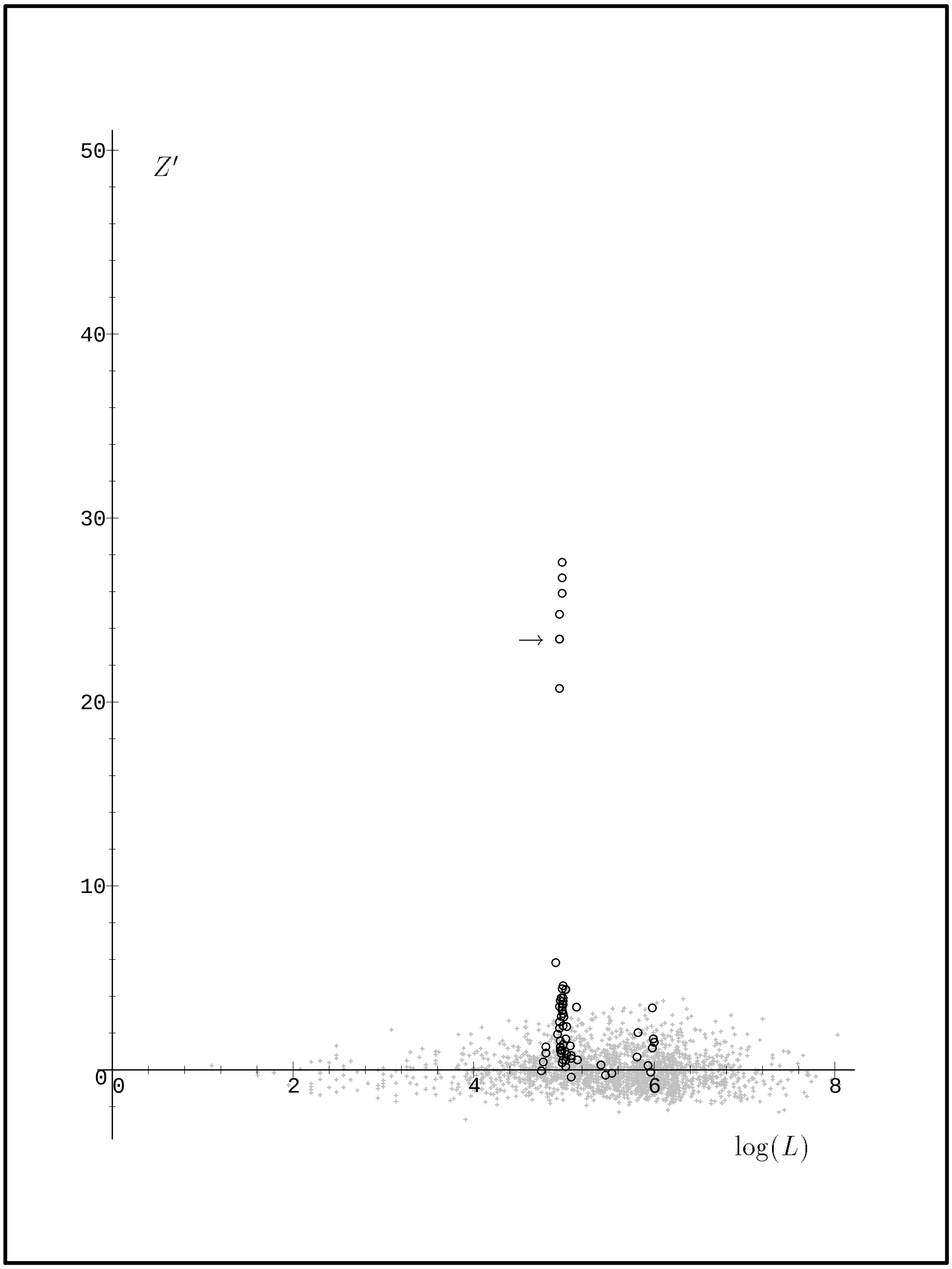


Figure 5.5: Z'-scores pour BlastMultAl (A). (Cercles noirs pour les globines, croix grises pour les non-globines). La flèche marque la superposition de deux cercles.

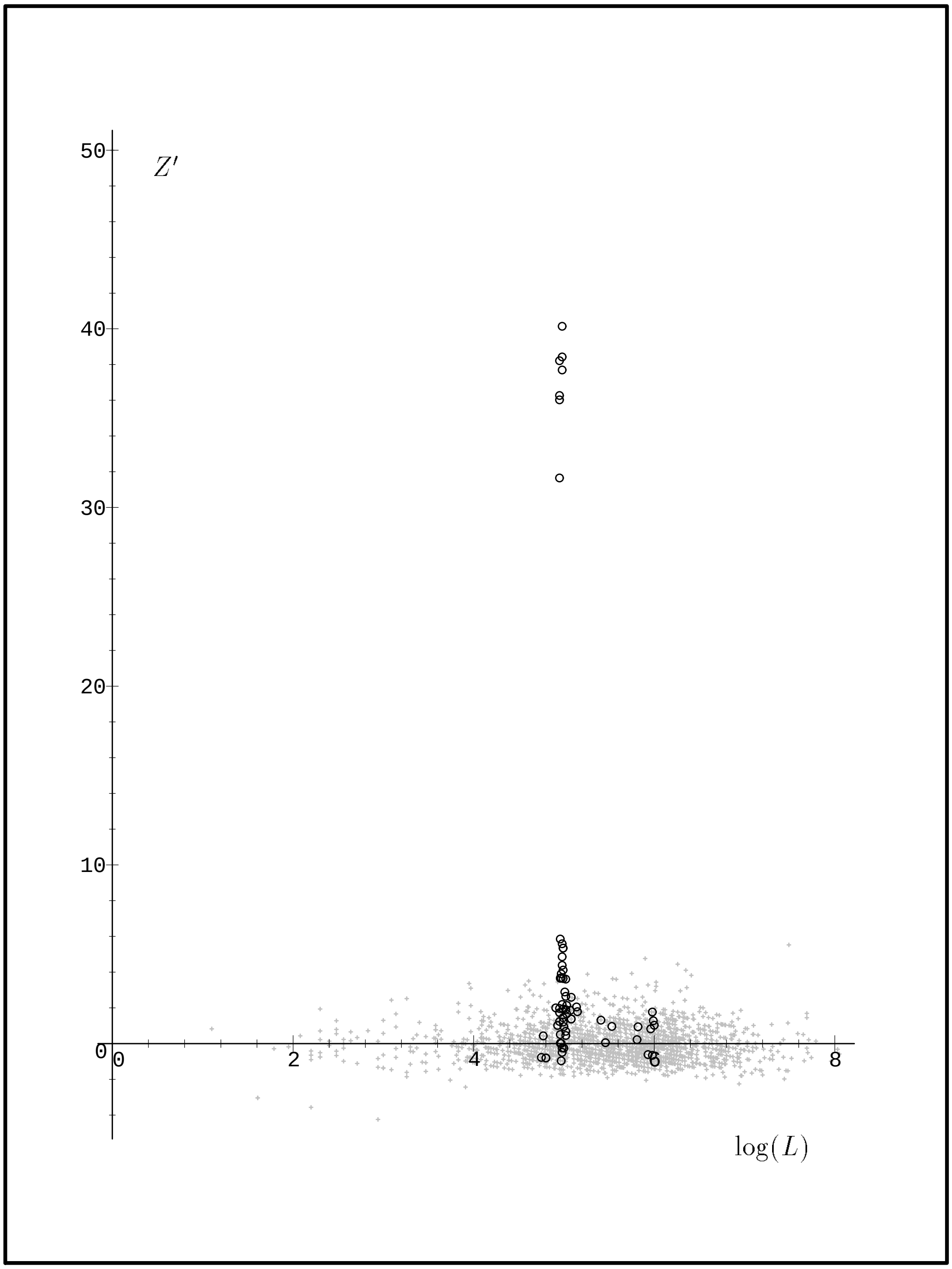


Figure 5.6: Z'-scores pour la méthode des Consensus Non-pondérés (B). (Cercles noirs pour les globines, croix grises pour les non-globines).

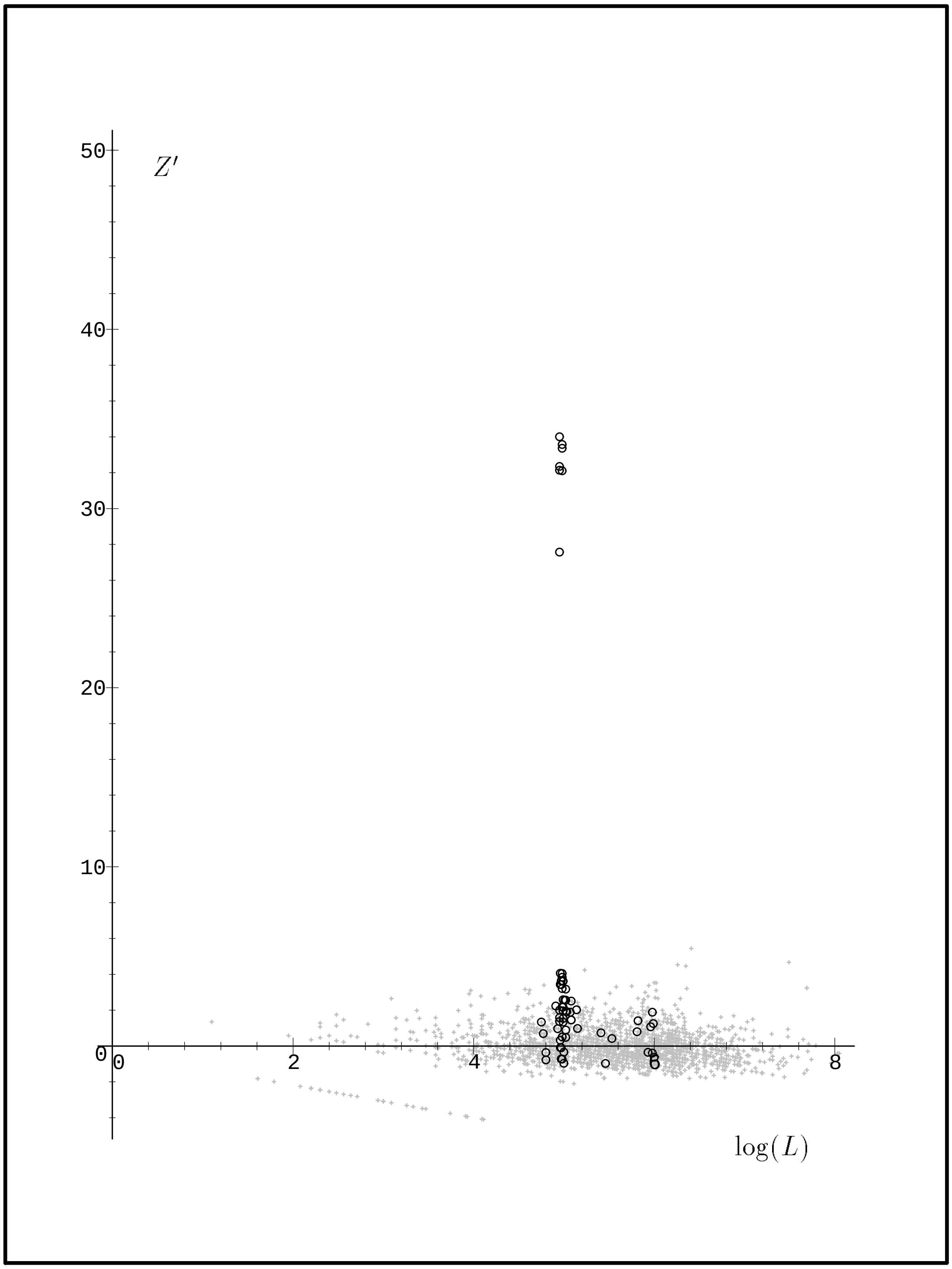


Figure 5.7: Z'-scores pour la méthode de Consensus Pondérés (C). (Cercles noirs pour les globines, croix grises pour les non-globines).

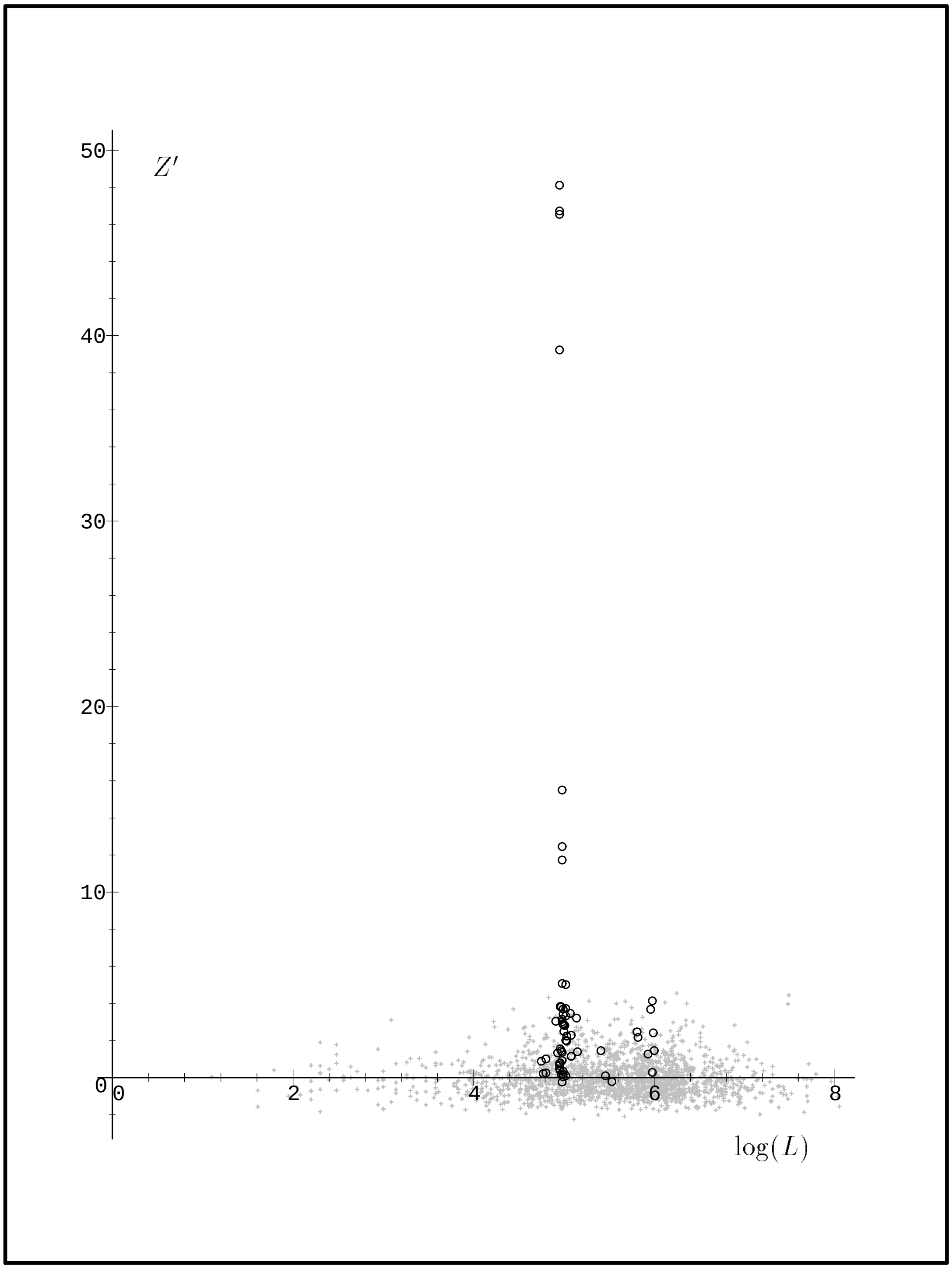


Figure 5.8: Z'-scores pour les *Profiles* Non-pondérés (D). (Cercles noirs pour les globines, croix grises pour les non-globines).

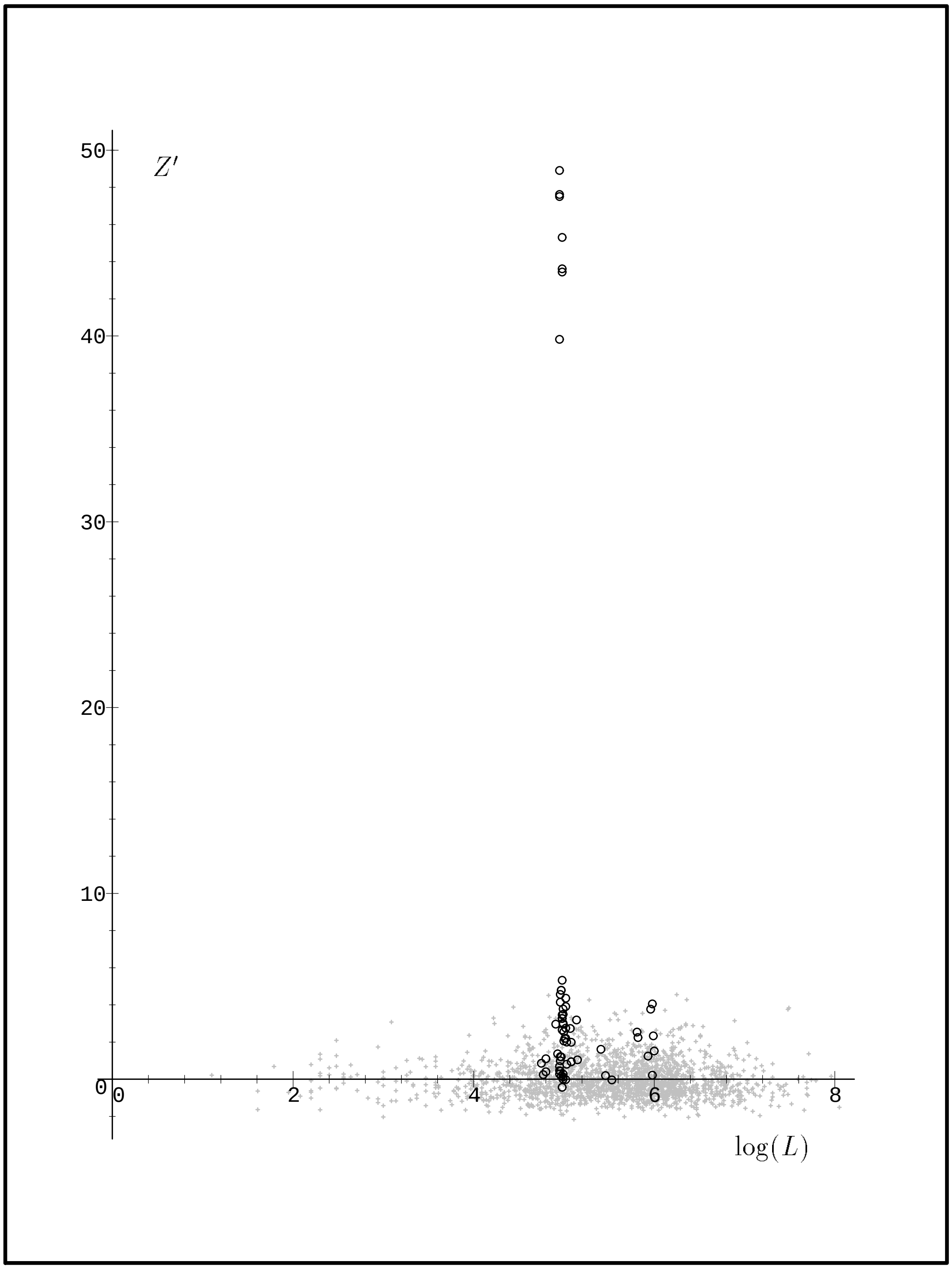


Figure 5.9: Z'-scores pour la méthode de *Profiles* pondérés (E). (Cercles noirs pour les globines, croix grises pour les non-globines).

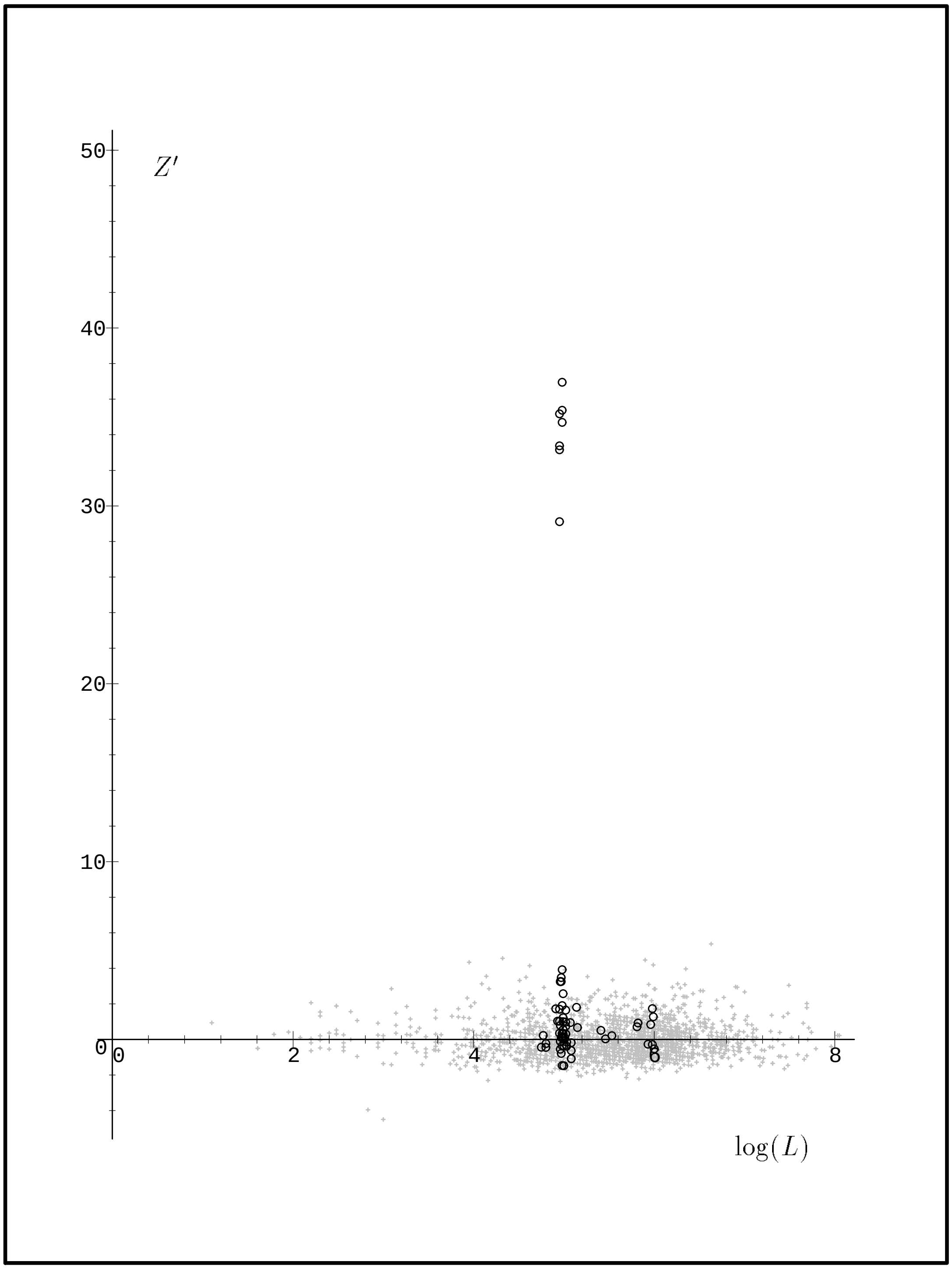


Figure 5.10: Z'-scores pour la méthode des séquences à plat (F). (Cercles noirs pour les globines, croix grises pour les non-globines).

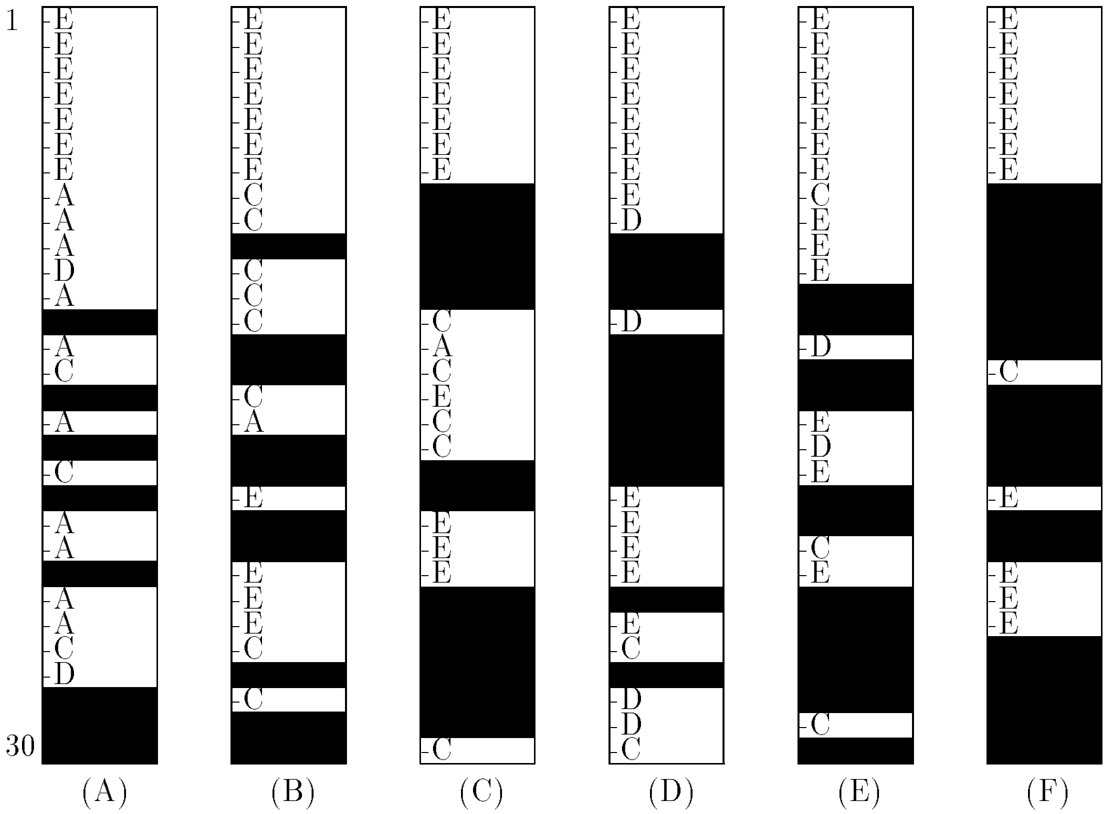


Figure 5.11: les 30 meilleurs Z'-scores obtenus avec une Famille-Échantillon de globines et un ensemble cible de 66 globines et 1875 non-globines. (A) BlastMultAl. (B) Consensus Non-pondéré. (C) Consensus Pondéré. (D) Profile Non-pondéré. (E) Profile Pondéré. (F) Séquences à plat. Les non-globines sont représentées par des rectangles noirs. Les lettres -A, ... ,-F indiquent la méthode donnant le meilleur Z'-score pour la protéine correspondante.

	(A)	(B)	(C)	(D)	(E)	(F)
GLOBINES						
$3 \leq Z'$	0.364	0.303	0.273	0.364	0.348	0.182
$2 \leq Z' < 3$	0.121	0.106	0.136	0.182	0.212	0.015
$Z' < 2$	0.515	0.591	0.591	0.455	0.439	0.803
NON-GLOBINES						
$3 \leq Z'$	0.007	0.011	0.010	0.013	0.011	0.010
$2 \leq Z' < 3$	0.038	0.024	0.022	0.031	0.032	0.033
$Z' < 2$	0.954	0.965	0.968	0.956	0.957	0.958

Tableau 5.2: Comparaisons des Z'-scores pour les six méthodes étudiées.

Rangs obtenus par recherche de similarité entre d'une part la Famille-Échantillon de ProDom28 et d'autre part 66 globines et 1875 non-globines							
Nom		Blast-MultAl	Consensus Non-Pond.	Cons. Pond.	Profiles Non-Pond.	Prof. Pond.	Séquences à plat
HBB_MICGA		1	1	2	7	6	1
HBB_LYNLY		2	4	3	5	5	4
HBB_APTFO		3	2	6	6	4	2
HBA2_ARCGA		4	3	1	1	1	3
HBA1_ARCGA		5	5	4	3	3	5
HBA_LYNLY		6	6	5	2	2	6
HBA_MICGA		7	7	7	4	7	7
GLB2_CALSO		8	90	65	45	47	134
LGB3_SESRO		9	98	94	53	49	794
GLB2_TYLHE		10	1296	518	1065	1238	1174
GLB4_LUMTE		11	458	530	9	14	313
GLBT_CHITH		12	579	348	785	867	632
RL22_HALMA	•	13	938	681	148	249	1718
GLB1_CALSO		14	878	1016	202	261	788
LGB1_MEDTR		15	11	17	30	29	627
MCP2_ECOLI	•	16	551	978	124	109	1648
GLB1_LUMTE		17	879	630	604	573	970
YFH5_YEAST	•	18	1530	1564	683	559	646
LGB4_MEDSA		19	12	18	26	22	485
GLYA_SALTY	•	20	1275	934	214	285	1680
GLB2_LUMTE		21	1077	1567	39	67	1885
LGB2_SESRO		22	97	93	52	48	793
CYS1_DICDI	•	23	466	1342	869	772	1432
GLB1_PHESE		24	133	149	403	554	606
GLB3_LUMTE		25	86	84	37	41	119
LGB2_MEDTR		26	13	30	350	36	620
HMPA_ECOLI		27	127	108	13	18	129
SSO1_YEAST	•	28	298	1035	169	536	301
COAT_MISV	•	29	711	572	766	755	1090
THYL_RAT	•	30	27	365	929	937	21
Les protéines non-globines sont marquées par un •							

Tableau 5.3: Les similarités détectées par BlastMultAl entre la Famille-Échantillon de globines et les globines GLB2_TYLHE, GLBT_CHITH, GLB1_CALSO, GLB1_LUMTE et GLB1_PHESE ne sont pas détectées par les autres méthodes. Les similarités entre la Famille-Échantillon et GLB2_CALSO et LGB3_SESRO sont sensiblement mieux détectées par BlastMultAl que par les autres méthodes.

Z'-scores obtenus par recherche de similarité entre d'une part la Famille-Échantillon de ProDom28 et d'autre part 66 globines et 1875 non-globines							
Nom		Blast-MultAl	Consensus Non-Pond.	Cons. Pond.	Profiles Non-Pond.	Prof. Pond.	Séquences à plat
HBB_MICGA		27.59	40.14	33.57	11.73	43.44	36.95
HBB_LYNLY		26.75	37.70	33.36	15.50	43.62	34.70
HBB_APTFO		25.91	38.43	32.11	12.46	45.30	35.38
HBA2_ARCGA		24.77	38.22	34.01	48.11	48.91	35.17
HBA1_ARCGA		23.42	36.27	32.35	46.54	47.50	33.38
HBA_LYNLY		23.42	36.03	32.14	46.72	47.61	33.15
HBA_MICGA		20.74	31.65	27.57	39.23	39.82	29.11
GLB2_CALSO		5.83	2.00	2.24	3.04	2.96	1.72
LGB3_SESRO		4.57	1.93	1.96	2.87	2.96	0.10
GLB2_TYLHE		4.41	-0.49	0.51	-0.24	-0.44	-0.35
GLB4_LUMTE		4.37	0.69	0.48	5.01	4.36	0.98
GLBT_CHITH		4.37	0.45	0.90	0.10	-0.02	0.31
RL22_HALMA	•	4.35	-0.05	0.25	1.76	1.24	-1.05
GLB1_CALSO		3.92	0.01	-0.10	1.44	1.20	0.11
LGB1_MEDTR		3.90	5.34	3.62	3.41	3.49	0.32
MCP2_ECOLI	•	3.86	0.50	-0.08	1.96	2.13	-0.95
GLB1_LUMTE		3.77	0.02	0.33	0.37	0.43	-0.10
YFH5_YEAST	•	3.75	-0.77	-0.72	0.27	0.45	0.30
LGB4_MEDSA		3.73	5.34	3.62	3.70	3.78	0.55
GLYA_SALTY	•	3.65	-0.47	-0.03	1.38	1.10	-1.00
GLB2_LUMTE		3.58	-0.24	-0.73	3.15	2.65	-1.47
LGB2_SESRO		3.56	1.93	1.96	2.87	2.96	0.10
CYS1_DICDI	•	3.52	0.68	-0.48	-0.02	0.12	-0.65
GLB1_PHESE		3.44	1.73	1.58	0.76	0.46	0.35
GLB3_LUMTE		3.41	2.05	2.03	3.21	3.19	1.81
LGB2_MEDTR		3.40	4.86	3.21	0.92	3.27	0.33
HMPA_ECOLI		3.37	1.77	1.89	4.14	4.05	1.73
SSO1_YEAST	•	3.36	1.08	-0.12	1.63	0.52	1.02
COAT_MISV	•	3.32	0.22	0.42	0.14	0.15	-0.24
THYL_RAT	•	3.32	3.63	0.83	-0.08	-0.09	3.35
Les protéines non-globines sont marquées par un •							

Tableau 5.4: Les Z'-scores correspondant au précédant tableau.

5.4.1.6 Recherche de similarités distantes avec des graphes d'alignement

Les tableaux 5.3 et 5.4 montrent comment les 30 séquences qui ont le meilleur score avec BlastMultAl réagissent avec les autres méthodes. Le tableau 5.3 donne les rangs obtenus par les 30 séquences ayant les meilleurs scores avec BlastMultAl, pour chacune des six méthodes. Le tableau 5.4 donne les Z'-scores correspondant. Ces tableaux montrent que plusieurs similarités distantes sont détectées par BlastMultAl, et non par les autres méthodes.

- Les similarités de la Famille-Échantillon de globines et de GLB2_CALSO, une globine de mollusque, et LGB3_SESRO, une léghemoglobine, sont beaucoup mieux détectées par BlastMultAl que par les autres méthodes (les léghemoglobines sont des hémoprotéines présentes dans les nodules des racines des plantes légumineuses; leur fonction consiste à fournir de l'oxygène aux bactéroïdes).

- La similarité de la Famille-Échantillon et de GLB4_LUMTE, une sous-unité de globine de ver, est détectée par BlastMultAl et par les *profiles*, mais ne l'est pas par les autres méthodes. La similarité de la Famille-Échantillon et d'une autre sous-unité de globine de ver, GLB1_LUMTE, n'est détectée que par BlastMultAl.

- Autres résultats intéressant, la détection de la similarité de la Famille-Échantillon et de la globine d'insecte GLBT_CHITH n'est faite que par BlastMultAl, et BlastMultAl détecte également la similarité de la Famille-Échantillon et de l'hémoglobine géante de ver GLB1_PHESE, même si 5 non-globines ont un meilleur score que cette dernière.

5.4.2 Recherche de similarités d'une séquence et de multi-alignements appartenant à une base de données de multi-alignements

Nous considérons à présent la seconde application de notre méthode. Nous partons d'une séquence requête, et nous voulons savoir, parmi un ensemble de multi-alignements, lesquels sont similaires de notre requête. Cette approche s'applique en particulier aux recherches de similarité avec les multi-alignements de la base ProDom.

5.4.2.1 Méthode de construction des bases de multi-alignements ProDom

Les bases ProDom[1] [76] sont générées à partir des bases de protéines SWISSPROT[2] de la manière suivante.

- Les comparaisons par paires de séquences de SWISSPROT, effectuées avec Blast, produisent un ensemble de HSPs (paires de séquence de scores élevés, en anglais High Scoring Pair).

[1] URL: http: //protein.toulouse.inra.fr/prodom.html

[2] Un numéro de version d'une base ProDom correspond au numéro de version de la base SwissProt à partir de laquelle cette base a été générée.

- Quand deux HSPs se chevauchent, on construit un HSS (ensemble de score élevé, en anglais High Scoring Set) en regroupant les parties communes à ces deux HSPs dans un même HSS. De la même manière, on peut regrouper un HSP et un HSS, ou deux HSS en un seul HSS. Cette opération se poursuit tant que la longueur des HSS construits est supérieure à une longueur minimum. En tenant compte des HSPs dont ils proviennent, on construit un graphe de HSSs.
- Les chemins du graphe de HSSs qui contiennent le plus de séquences fournissent les multi-alignements finaux. Les multi-alignements obtenus sont alors entièrement recalculés au moyen du logiciel MultAlin [18].

On peut se référer au tableau 5.1 pour avoir une vue d'ensemble de la base ProDom28.

5.4.2.2 Calibration probabiliste des multi-alignements de ProDom28

Comme nous l'avons précédemment mentionné, les multi-alignements de ProDom28 sont d'allure très inhomogène. Si nous voulons comparer les scores obtenus entre une requête et deux multi-alignements, nous devons tenir compte de la différence de structure de ces multi-alignements, qui se traduit par des comportements probabilistes différents. En d'autres termes, nous devons calculer les constantes λ et K de Karlin pour chaque multi-alignement, et nous pourrons ensuite comparer les pertinences probabilistes des scores de deux alignements obtenus avec deux multi-alignements différents. Chaque multi-alignement de ProDom28 a été calibré individuellement en le comparant au moyen de BlastMultAl à 10000 séquences aléatoires. Il s'agit donc dans ce cas d'une simulation directe. Nous verrons au prochain chapitre que, pour calibrer les multi-alignements de ProDom32, nous avons utilisé une méthode beaucoup plus rapide, celle des approximations Poissonniennes.

En utilisant l'approximation de l'exponentielle pour les petites valeurs de l'exposant, un changement de variable dans l'équation 5.2 nous donne

$$\mathcal{F}(S(l_q l_r)) \approx Prob(S(l_q l_r) > y) \approx K l_q l_r e^{-\lambda y}, \tag{5.4}$$

où $\mathcal{F}(S)$ est la fonction de répartition des scores obtenus par simulation, l_q est la longueur de l'objet requête (dans notre cas des multi-alignements), et l_r la longueur des séquences simulées, (longueur que nous avons choisies égales à 300 pour nos calibrations). Pour chaque recherche d'alignement de la requête et d'une séquence simulée, nous avons conservé le meilleur score. Nous avons ensuite représenté le logarithme de la fonction de répartition $\mathcal{F}(S)$ en fonction du score S. Les paramètres K et λ ont alors été directement calculés par régression linéaire, pour chaque multi-alignement. Les calculs numériques effectués montrent la très bonne qualité de la régression linéaire et l'influence négligeable de l'approximation faite pour le calcul de l'exponentielle. Ces calculs numériques montrent aussi que le modèle de Karlin s'applique bien aux statistiques d'alignements effectués avec des graphes d'alignement.

Nous développons à présent une autre caractérisation de nos graphes d'alignement. La recherche de similarité avec un graphe d'alignement est en gros équivalente à une recherche de similarité avec une séquence plus longue que le graphe, séquence qui contiendrait les mêmes alignements potentiels. Nous appelons *longueur-de-complexité* d'un multi-alignement la longueur de cette séquence étendue, correspondant à ce multi-alignement. On peut définir un ordre de grandeur de cette longueur-de-complexité en fonction des principaux paramètres du multi-alignement après le pré-traitement. Si L est la longueur du multi-alignement, r le nombre de régions de branches, et b le nombre de branches dans les régions de branches, on a comme ordre de grandeur de la longueur-de-complexité $\mathcal{L}^c$

$$\mathcal{L}^c = \mathcal{O}(L \times r^b),$$

ou,

$$\log(\mathcal{L}^c) = \mathcal{O}(\log(L)) + \mathcal{O}(b \log(r)).$$

La figure 5.12 représente les valeurs de λ calculées par simulation en fonction de la valeur de $\mathcal{L}^c$ pour les multi-alignements de ProDom28 dans le cas où $b = 2$ (multi-alignements formés de deux séquences) et dans le cas où $b = 5$ (multi-alignements formés d'au moins 5 séquences).

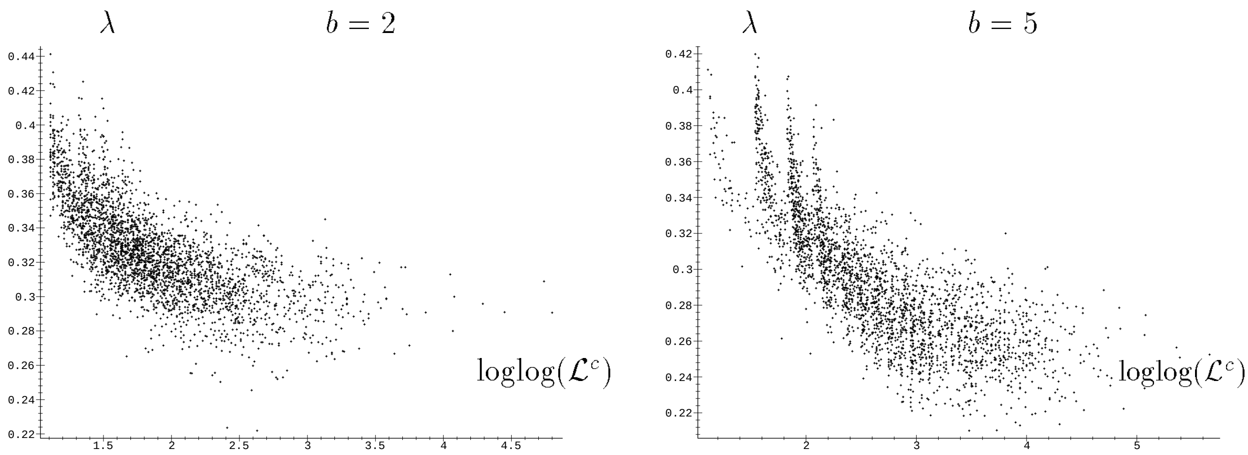


Figure 5.12: Les valeurs de λ pour les multi-alignements de ProDom28 contenant $b = 2$ et $b = 5$ branches dans les régions de branches.

5.4.2.3 Interrogation de ProDom28 avec BlastMultAl

Les simulations décrites dans le paragraphe précédant permettent une calibration probabiliste des multi-alignements de ProDom28 pour la méthode BlastMultAl. Nous donnons dans ce paragraphe des résultats de requêtes effectuées d'une part avec BlastMultAl sur les multi-alignements de ProDom28 ainsi calibrés, et d'autre part avec Blast sur les consensus de ces multi-alignements. Les fonctions de calcul probabilistes et les fonctions d'édition de Blast ont été réutilisées telles quelles. Nous avons choisi comme séquence requête la protéine respiratoire d'insecte GLBT_CHITH, et les résultats des recherches de similarité effectuées avec les deux logiciels considérés sont donnés respectivement dans les figures 5.13 et 5.14 (voir le paragraphe 4.3.2.2 pour le calcul de $P(n)$). Dans la figure 5.13, nous avons seulement donné les similarités avec de vrais alignements, contenant au moins deux séquences. Dans ces deux figures, le nombre de séquences dans les multi-alignements est donné entre parenthèses. BlastMultAl détecte deux fois plus de similarités que Blastp, mais ne détecte pas les similarités avec les multi-alignements 470, 177, 377 et 4429. Pour ces multi-alignements les valeurs de λ qui ont été calculées par simulation sont voisines de 0.25, à comparer avec $\lambda_p \approx 0.32$ pour Blastp. Le paramètre λ intervient de manière exponentielle dans les calculs de pertinence probabiliste associés aux scores, et les faibles valeurs de λ pour les multi-alignements mentionnés expliquent que BlastMultAl manque les similarités de ces multi-alignements et de GLBT_CHITH.

```
--------------------------------------------------------------------------------
                                                                   Smallest
                                                                   Poisson
                                                            High  Probability
Sequences producing High-scoring Segment Pairs:            Score  P(N)      N

554   (21)   GLOBIN PRECURSOR. PRECURSOR (ERYTHROCRUORIN  ...    94  0.0057    1
2647   (5)   GLOBIN, COMPONENT POLYMERIC MAJOR MONOMERIC P...    50  0.23      1
1740   (7)   HEMOLYSIN PROTEIN). (CLYII-A II) (CYTOLYSIN A...    62  0.28      2
2121   (6)   . PRIMASE DNA Q05271 P07362 P30103 P02923 P33...    65  0.88      1
3140   (4)   (CLONE PRECURSOR ABUNDANT EMBRYONIC USP92). P...    52  0.93      1
1284  (10)   DEHYDROGENASE (IMP INOSINE-5'-MONOPHOSPHATE ....    76  0.93      1
1261  (10)   ENZYME SYSTEM, (PHOSPHOTRANSFERASE PHOSPHOTR....    70  0.98      1
1158  (11)   EXTRACELLULAR GLOBIN (ERYTHROCRUORIN). (GLOB....    63  0.98      1
982   (13)   HEAVY MYOSIN CHAIN, (MHC CHAIN SKELETAL MUSC....    73  0.9997    1
1051  (12)   CHEMOTAXIS METHYL-ACCEPTING CHEMORECEPTOR PR....    63  0.9999    1
1579   (8)   RIBOSOMAL 30S S15. MITOCHONDRIAL P21771 P0637...    60  0.99995   2
--------------------------------------------------------------------------------
```

Figure 5.13: Résultat d'une requête BlastMultAl sur ProDom28. La séquence requête est la protéine respiratoire GLBT_CHITH.

```
-------------------------------------------------------------------------------
                                                                   Smallest
                                                                   Poisson
                                                            High  Probability
Sequences producing High-scoring Segment Pairs:            Score  P(N)      N

554  (21)   GLB3(2) GLBW(1) GLBV(1)... GLOBIN PRECURSOR. ...    87  9.7e-07   1
470  (24)   LGB3(3) LGBA(2) HBPL(2)... LEGHEMOGLOBIN I. H...    39  0.88      2
177  (46)   TETO(3) TETM(2) RF3(1)... FACTOR ELONGATION (...    41  0.98      2
377  (28)   NUCM(4) PHNL(3) NUOD(1)... LARGE HYDROGENASE ...    53  0.98      1
4429  (3)   DIKINASE PYRUVATE,ORTHOPHOSPHATE PRECURSOR P2...    41  0.993     2
3140  (4)   EA87(1) EA30(1) (CLONE PRECURSOR ABUNDANT EMB...    52  0.995     1
-------------------------------------------------------------------------------
```

Figure 5.14: Résultat d'une requête Blastp sur les consensus non-pondérés de ProDom28. La séquence requête est la protéine respiratoire GLBT_CHITH.

Parmi les similarités relevées avec BlastMultAl, et qui ne sont pas détectées par le requête Blastp sur les consensus, on peut remarquer la similarité de GLBT_CHITH et du multi-alignement 1158 de ProDom28. Alors que GLBT_CHITH est une globine d'insecte, ce multi-alignement est composé de globines de différents vers: des vers de terre (LUMTE et PHESE), un ver marin (TYLHE), un ver tube géant de fond des océans (LAMSP) , et un ver de la boue (TUBTU). La détection de ces similarités distantes n'est pas possible avec le consensus seul.

Les comparaisons faites avec différentes requêtes montrent que les deux méthodes sont largement complémentaires. BlastMultAl permet, comme cela avait été mis en évidence dans le cas de la recherche de similarité avec une famille, de mettre en évidence des similarités distantes qui ne sont pas détectées avec le consensus.

5.4.3 Les biais des multi-alignements courts

La figure 5.12 montre un biais sensible des λ pour les alignements courts (disons de moins de 60 positions). Les valeurs de λ calculées par simulation sont manifestement surévalués pour ces multi-alignements courts. Ce biais apparaît aussi quand on cherche à calculer λ par simulation pour des multi-alignements composés d'une seule séquence. Il est nécessaire de tenir compte du fait que le modèle probabiliste de Karlin est un modèle asymptotique, et il n'est donc pas étonnant que les résultats pour des objets de faible longueur soient décevants. À titre d'exemple, le multi-alignement 2647 de ProDom28 qui apparaît dans le résultat de la requête de similarité de GLBT_CHITH faite avec BlastMultAl (figure 5.13) a une longueur $L = 26$ et le paramètre λ calculé par simulation pour ce multi-alignement vaut $\lambda \approx 0.37$. Il serait possible de corriger partiellement ces biais par des méthodes de régression sur les nuages de points obtenus dans la figure 5.12.

5.4.4 Performances

Le logiciel BlastMultAl effectue un traitement sensiblement plus lourd que celui effectué par Blastp; il est plus lent que ce dernier logiciel. Une recherche de similarité avec BlastMultAl demande environ six fois plus de temps que la même recherche effectuée avec Blastp.

Chapitre 6

Les approximations Poissonniennes et leurs applications à l'alignement de séquences

6.1 Introduction

Nous décrivons dans ce chapitre la méthode des approximations Poissonniennes. L'application qui en est faite à l'algorithme BlastMultAl et à la base de protéines ProDom est originale (paragraphe 6.5).

Il est bien connu que dans une suite de tirages aléatoires indépendants et identiquement distribués d'une pièce biaisée tels que la probabilité d'obtenir pile vaille χ/n, le nombre de fois X où l'on obtient pile sur n tirages suit asymptotiquement une loi de Poisson de paramètre χ, telle que

$$\Pr(X = k) = e^{-\chi}\frac{\chi^k}{k!}.$$

Ce résultat d'approximation Poissonnienne reste vrai quand les tirages ne sont plus identiquement distribués, mais que la somme des probabilités d'obtenir pile est en $\mathcal{O}(1)$. Hodges et Le Cam [43] ont majoré la distance de la loi donnant le nombre de piles et de la loi de Poisson correspondante. Chen [14] a utilisé une identité mathématique remarquable pour montrer que la convergence vers une loi de Poisson peut être établie dans de nombreux cas concernant des variables dépendantes. Sa méthode est une adaptation de la méthode différentielle utilisée par Stein [79] pour borner l'erreur de l'approximation normale pour la somme de variables dépendantes. Pour cette raison, cette méthode d'approximations Poissonniennes est appelée méthode de Chen-Stein.

Aldous [2] donne de nombreux exemples d'applications des approximations Poissonniennes dans son livre sur l'heuristique de "Poisson clumping", qu'on peut traduire par "groupes de Poisson". Waterman et Vingron ont appliqué cette heuristique à l'alignement local de deux séquences. Ils ont montré expérimentalement que la formule 2.1

$$Pr\left(\mathcal{S}(n,m) > \frac{\log(nm)}{\lambda} + x\right) = 1 - e^{-Ke^{-\lambda x}}, \tag{6.1}$$

obtenue pour l'alignement sans gaps de deux séquences continuait à s'appliquer dans le cas où l'on introduisait les gaps. Dans cette formule, que nous avons démontrée au chapitre 2, $\mathcal{S}(n.m)$ représente le score de l'alignement optimal de deux séquences de longueur respectives n et m. Cependant les valeurs des paramètres λ et K dépendent dans ce cas des pénalités d'ouverture et d'extension de gaps. Waterman et Vingron ont montré qu'on pouvait évaluer ces paramètres de manière très efficace au moyen des approximations de Poisson, en comptant le nombre de groupes d'alignements optimaux et sous-optimaux. Cette méthode est environ cent fois plus rapide que celle qui consiste à évaluer les paramètres par simulation directe, en ne comptant que les alignements optimaux. Nous appliquerons cette méthode très efficace au calibrage probabiliste des multi-alignements de la base ProDom32 pour l'algorithme de recherche de similarités BlastMultAl.

Nous ferons au paragraphe 6.3 l'exposé de la méthode de Chen-Stein, dans le cas où l'on considère la convergence de la distribution du nombre d'occurrences d'événements rares mais non indépendants vers une distribution de Poisson. Nous montrerons ensuite au paragraphe 6.4 comment ce résultat a été appliqué par Waterman et Vingron à l'alignement de séquences.

Nous mentionnons enfin un résultat concernant une autre méthode de recherche de similarité avec des multialignements, les profiles. Goldstein et Waterman [31] ont utilisé les approximations Poissonniennes pour obtenir la distribution asymptotique des scores des Profiles.

6.2 La convergence Poissonnienne dans le cas Bernoulli indépendant

D'une manière un peu simplifiée, les lois limites de somme de variables aléatoires $\{0,1\}$ appartiennent à deux grandes familles, celles dont les lois limites sont normales, ou Gaussiennes, et celles dont les lois limites sont Poissonniennes. Quand les probabilités d'apparition des événements $\{0\}$ et $\{1\}$ ne sont pas petites, c'est à dire quand aucun de ces deux événements n'est rare, la loi limite est Gaussienne. Au contraire, quand la probabilité d'un de ces deux événements est rare, la loi limite est Poissonnienne.

Notation. Nous noterons dans ce chapitre par P_χ la distribution de Poisson de paramètre χ sur $\mathbb{N}$ ($P_\chi(X=k)=e^{-\chi}\frac{\chi^k}{k!}$).

Le cas le plus simple d'approximation Poissonnienne se produit quand on compte le nombre d'occurrences de l'événement $\{1\}$ parmi n tirages indépendants $\{0,1\}$, lorsque la probabilité d'obtenir 1 pour un tirage vaut $p=\chi/n$.

Soit la variable élémentaire X, définie comme $\{\Pr(X=1)=p=1-\Pr(X=0\}$. Nous considérons la variable aléatoire $U=\sum_{i=1}^n X_i$. En prenant la transformée de Laplace de U, nous avons

$$E(e^{tU})=[E(e^{tX})]^n=[1-p(1-e^t)]^n=[1-\frac{\chi}{n}(1-e^t)]^n\to e^{\chi(e^t-1)}. \tag{6.2}$$

Or la transformée de Laplace d'une variable Z vérifiant une loi de Poisson de paramètre

χ est

$$E(e^{tZ}) = \sum e^{tk} e^{-\chi} \frac{\chi^k}{k!} = e^{\chi(e^t-1)}. \tag{6.3}$$

Une distribution est déterminée de manière unique par sa transformée de Laplace (voir Feller [27]), et donc U tend vers une loi de Poisson de paramètre χ, quand n tend vers l'infini.

Avant de généraliser un peu ce résultat, nous énonçons un théorème de convergence utilisant les fonctions caractéristiques (voir Durrett [26], p. 99 ou Feller, Tome 2, p. 508).

Théorème 43 (Continuité). *Soient f_n, $1 \le n \le \infty$ des distributions de probabilités de fonctions caractéristiques $\phi_n = \int e^{itx} df_n(x)$.*

1. *Si $f_n \to f_\infty$, alors $\phi_n(t) \to \phi_\infty(t)$ pour tout t.*

2. *Si ϕ_n converge simplement vers une limite ϕ continue en 0, alors la suite de distributions f_n converge en loi (faiblement) vers la distribution f de fonction caractéristique ϕ.*

Rappellons que la fonction caractéristique d'une variable X est par définition $\phi(t) = E(e^{itX})$.

Nous avons alors.

Théorème 44 (Tirages de Bernoulli indépendants avec probabilités variables). *Pour tout n, soient $X_{n,m}$, $1 \le m \le n$ des variables aléatoires indépendantes, telles que $\Pr(X_{n,m} = 1) = p_{n,m} = 1 - \Pr(X_{n,m} = 0)$. Supposons*

1. *$\sum_{m=1}^{n} p_{n,m} \to \chi \in]0, \infty[$, et*

2. *$\max_{1 \le m \le n} p_{n,m} \to 0$.*

Si $S_n = X_{n,1} + \cdots + X_{n,n}$, alors S_n tend en loi vers P_χ, loi de Poisson de paramètre χ.

Preuve. Nous suivons Durrett pour la démonstration. La fonction caractéristique

$$E(e^{itP_\chi}) = e^{\chi(e^{it}-1)}$$

s'obtient en remplaçant t par it dans l'équation 6.3.

Considérons

$$\phi_{n,m}(t) = E\left(e^{itX_{n,m}}\right) = (1 - p_{n,m}) + p_{n,m} e^{it}.$$

Soit $S_n = X_{n,1} + \cdots + X_{n,n}$. Nous avons alors

$$E\left(e^{itS_n}\right) = \prod \left(1 + p_{n,m}(e^{it} - 1)\right). \tag{6.4}$$

Soit $0 \le p \le 1$. Nous avons $-2 \le \Re(e^{it} - 1) \le 0$, et donc

$$\left|e^{p(e^{it}-1)}\right| = e^{p\Re(e^{it}-1)} \le 1. \tag{6.5}$$

D'autre part, le point de valeur complexe $1 + p(e^{it} - 1) = (1 - p) + pe^{it}$ est le barycentre des points 1 et e^{it} affectés des poids respectifs $(1 - p)$ et p. Ce point est donc à l'intérieur du cercle unité, et donc

$$\left|1 + p\left(e^{it} - 1\right)\right| \leq 1. \tag{6.6}$$

Considérons d'autre part une suite de nombres complexes $z_1, \ldots, z_n$ et $w_1, \ldots, w_n$ de module inférieur ou égal à 1. Nous avons

$$\begin{aligned} \left|\prod_{m=1}^{n} z_m - \prod_{m=1}^{n} w_m\right| &\leq \left|z_1 \prod_{m=2}^{n} z_m - z_1 \prod_{m=2}^{n} w_m\right| + \left|z_1 \prod_{m=2}^{n} w_m - w_1 \prod_{m=2}^{n} w_m\right| \\ &\leq \left|\prod_{m=2}^{n} z_m - \prod_{m=2}^{n} w_m\right| + |z_1 - w_1| \\ &\leq \sum_{m=1}^{n} |z_m - w_m|, \end{aligned} \tag{6.7}$$

la dernière équation étant obtenue en développant la précédente pour les m croissant jusqu'à n.

D'autre part, si z est un nombre complexe de module inférieur ou égal à 1, nous avons

$$|e^z - (1 + z)| = \left|\frac{z^2}{2!} + \frac{z^3}{3!} + \ldots\right| \leq \frac{|z^2|}{2}(1 + \frac{1}{2} + \frac{1}{2^2} + \ldots) = |z|^2. \tag{6.8}$$

Nous pouvons à présent borner la distance de $E(e^{itS_n})$ (équation 6.4) à $e^{\chi(e^{it}-1)}$.

$$\begin{aligned} |D(t)| &= \left|\exp(\sum_{m=1}^{n} p_{n,m}(e^{it} - 1)) - E(e^{itS_n})\right| && (6.9) \\ &= \left|\exp(\sum_{m=1}^{n} p_{n,m}(e^{it} - 1)) - \prod_{m=1}^{n} \left(1 + p_{n,m}(e^{it} - 1)\right)\right| && (6.10) \\ &= \left|\prod_{m=1}^{n} e^{p_{n,m}(e^{it}-1)} - \prod_{m=1}^{n} \left(1 + p_{n,m}(e^{it} - 1)\right)\right| && (6.11) \\ &\leq \sum_{m=1}^{n} \left|e^{p_{n,m}(e^{it}-1)} - \left(1 + p_{n,m}(e^{it} - 1)\right)\right| && (6.12) \\ &\leq \sum_{m=1}^{n} p_{n,m}^2 \left|e^{it} - 1\right|^2. && (6.13) \end{aligned}$$

L'équation 6.12 s'obtient en utilisant l'équation 6.7 avec $z_m = e^{p_{n,m}(e^{it}-1)}$ et $w_m = (1 + p_{n,m}(e^{it} - 1))$ (les équations 6.5 et 6.6 montrent que les conditions $|z_m| \leq 1$ et $|w_m| \leq 1$ sont vérifiées). Si d'autre part on a $\max_m p_{n,m} \leq 1/2$, comme $|e^{it} - 1| \leq 2$, on en déduit $|p_{n,m}(e^{it} - 1)| \leq 1$; on peut donc appliquer l'équation 6.8 avec $z = p_{n,m}(e^{it} - 1)$, ce qui donne l'équation 6.13.

En utilisant à nouveau $|e^{it} - 1| \leq 2$, nous obtenons

$$D(t) \leq 4 \sum_{m=1}^{n} p_{n,m}^2 \leq 4 \left(\max_{1 \leq m \leq n} p_{n,m}\right) \sum_{m=1}^{n} p_{n,m} \to 0,$$

compte tenu de nos hypothèses. En utilisant $\sum_{m=1}^{n} p_{n,m} \to \chi$, nous obtenons

$$E(e^{itS_n}) \to e^{\chi(e^{it}-1)}.$$

Cette dernière fonction est continue en zéro. Nous pouvons donc appliquer le théorème 43, qui nous dit que S_n converge en loi vers la loi de Poisson P_χ. □

On peut donner une autre méthode de démonstration du théorème 44, qui utilise des convolutions de mesures; c'est la méthode utilisée par Hodges et le Cam (1960) [43], qui donne de plus une vitesse de convergence par la loi de Poisson. Nous obtiendrons ce résultat comme application de la méthode de Chen-Stein, que nous développons au prochain paragraphe.

6.3 Approximations Poissonniennes et équation de Stein

Nous allons montrer dans ce paragraphe comment l'équation de Stein permet de traiter de manière très efficace les suites de variables de Bernoulli faiblement dépendantes, nous donnant pour les sommes de telles suite la convergence et la vitesse de convergence vers la loi de Poisson. La méthode s'applique aussi au cas plus simple de variables de Bernoulli indépendantes. Arratia et *al.* [5] donnent plusieurs versions des approximations de Poisson, en particulier une version de processus, qui correspond à la convergence d'un vecteur multidimensionnel vers un vecteur Poissonnien. Nous n'aurons besoin que de la version de dimension 1 de leur approche, qui correspond à leur théorème 1.

Notations. Nous noterons P_χ la distribution de Poisson de paramètre χ, $P_\chi\{i\} = \frac{\chi^i}{i!}e^{-\chi}$; nous noterons, pour $i \in \mathbb{N}$, $Q\{i\} = P(X = i)$ si Q est une distribution de probabilité sur $\mathbb{N}$, et si X est une variable aléatoire ayant cette distribution; si f est une fonction bornée, nous noterons $P_\chi(f) = \sum_0^{+\infty} f(i)\frac{\chi^i}{i!}e^{-\chi}$ et nous noterons la norme "max" par $||f|| = \sup_n\{|f(n)|\}$. Nous remarquons que $P_\chi()$ est une fonctionnelle linéaire sur l'espace vectoriel des fonctions bornées.

L'équation de Stein que nous définirons au paragraphe 6.3.1 est une identité d'énoncé simple. On en déduit par une méthode analytique qui, elle, n'est pas facile, des bornes en valeur et en variation pour la fonction g "transformée" de Stein d'une fonction bornée f (paragraphe 6.3.2).

6.3.1 L'équation de Stein

L'identité de Stein s'énonce par le lemme suivant.

Lemme 45. *Si f est une fonction bornée de $\mathbb{N}$ sur $\mathbb{N}$, alors on a*

$$\sum_{x\geq 0}(\chi f(x+1) - xf(x))P_\chi\{x\} = 0. \tag{6.14}$$

Réciproquement, si Q est une distribution de probabilité sur $\mathbb{N}$ telle que

$$\sum_{x\geq 0}(\chi g(x+1) - xg(x))Q\{x\} = 0, \tag{6.15}$$

pour toute fonction g bornée, alors $Q = P_\chi$.

Preuve. L'assertion 6.14 s'obtient immédiatement par télescopage des termes de la somme.

Supposons que, f étant une fonction bornée, il existe g bornée telle que

$$\chi g(x+1) - xg(x) = f(x) - P_\chi(f). \tag{6.16}$$

Nous avons, (en nous souvenant que $P_\chi(f) = \sum_0^{+\infty} \frac{\chi^i}{i!} f(i)e^{-\chi} = Cste \leq ||f|| < \infty$),

$$\int(\chi g(x+1) - xg(x))Q(dx) = \int f(x)Q(dx) - P_\chi(f)\int Q(dx) = Q(f) - P_\chi(f).$$

Nous en déduisons que $Q(f) = P_\chi(f)$ et que Q est une Poisson de paramètre χ.

Nous allons maintenant montrer l'existence d'une solution à l'équation 6.16. En prenant $h(n) = f(n) - P_\chi(f)$, considérons

$$g(n) = \frac{(n-1)!}{\chi^n}\sum_0^{n-1} h(i)\frac{\chi^i}{i!},\ n > 0, \quad \text{et} \quad g(0) = 0. \tag{6.17}$$

Nous avons

$$\chi g(n+1) - ng(n) = h(n) = f(n) - P_\chi(f),$$

et, donc, g est solution de l'équation 6.16.

De plus, en remarquant que

$$P_\chi(h) = \sum_0^\infty h(i)\frac{\chi^i}{i!} = P_\chi(f - P_\chi(f)) = P_\chi(f) - P_\chi(f) = 0,$$

on obtient que, pout tout $n > 0$,

$$g(n) = -\frac{(n-1)!}{\chi^n}\sum_n^{+\infty} h(i)\frac{\chi^i}{i!}. \tag{6.18}$$

Nous avons

$$|P_\chi(f)| = \sum_{i=0}^{+\infty} f(i)e^{-\chi}\frac{\chi^i}{i!} \leq |\max_n f(n)|\sum_{i=0}^{+\infty} e^{-\chi}\frac{\chi^i}{i!} = ||f||,$$

et donc $|h(n)| < \max f(n) + ||f|| = 2||f||$. Nous en déduisons

$$|g(n)| \leq 2||f||\sum_{i=0}^{+\infty}\frac{\chi^i}{n(n+1)\dots(n+i)} \leq 2e^\chi||f||. \tag{6.19}$$

$\square$

6.3.2 Des bornes uniformes pour la transformée de Stein d'une fonction f

Nous avons bien démontré au paragraphe précédant que g était bornée, mais, pour de grandes valeurs de χ, cette borne est exponentiellement grande. Le théorème suivant, dû à Barbour et Eagleson, donne un résultat très fort pour la fonction g, quand χ est grand. Ce résultat est indépendant de la fonction f choisie; il est donc uniforme.

Théorème 46. *Si f est une fonction positive majorée par 1, si g est solution de l'équation*

$$\chi g(x+1) - xg(x) = f(x) - P_\chi(f),$$

et si $\Delta g(n) = g(n+1) - g(n)$, on a les estimations suivantes.

$$||g|| \leq 1 \wedge \frac{1}{\sqrt{\chi}} \quad et \quad ||\Delta g|| \leq 1 \wedge \frac{1}{\chi}.$$

(Nous utilisons ici la notation classique en probabilité $a \wedge b = \min(a, b)$).

Preuve. Nous renvoyons au livre de Barbour, Holst et Janson [9] (lemme 1.1.1) pour la démonstration de ce théorème . La borne $||g|| \leq 2\min(1, \chi^{-1/2})$ s'obtient assez facilement. Barbour et *al.* [9] améliorent cette borne jusqu'à $||g|| \leq \min(1, \sqrt{2/e\chi} \leq \min(1, \chi^{-1/2})$. □

6.3.3 Application à des tirages de Bernoulli

Nous allons appliquer les bornes du théorème 46 au cas des tirages de Bernoulli indépendants, ce qui nous redonnera le résultat de Le Cam, puis aux tirages faiblement dépendants. Nous appliquerons ensuite ce dernier résultat aux alignements de séquences (paragraphe 6.4).

Notations. Dans ce qui suit, on notera $(I_i)_{1\leq i\leq n}$ une suite de variables aléatoires de Bernoulli de paramètres respectifs $(p_i)_{1\leq i\leq n}$, $W = \sum_1^n I_i$, et $\chi = E(W) = \sum_1^n p_i$.

Nous allons nous intéresser à la loi de la variable W.

En considérant l'équation 6.16 appliquée à la variable W, et en prenant les espérances sur W, nous obtenons

$$\begin{aligned} |E(f(W)) - P_\chi(f)| &= |E(\chi g(W+1) - Wg(W))| \\ &= \left| E\left(\sum_1^n (p_i g(W+1) - I_i g(W))\right)\right|. \end{aligned} \tag{6.20}$$

L'estimation de ce dernier terme nous donnera donc une estimation de la distance en variation totale (voir la définition ci-dessous) de la loi de W et de celle d'une loi de Poisson de paramètre χ.

Définition 47. *Étant donné deux variables aléatoires U et V sur $\mathbb{R}$, la distance en variation totale de ces deux variables est définie comme*

$$||\mathcal{L}(U) - \mathcal{L}(V)|| = 2\sup_A |\Pr(U \in A) - \Pr(V \in A)|,$$

où le sup est pris sur tous les sous-ensembles A des réels.

En prenant $f = I_A$, l'indicatrice de l'ensemble A, nous voyons que l'équation 6.20, vraie pour tout A, puisqu'elle est vraie pour tout f, définit bien la distance en variation totale de W à P_χ. Nous appliquons maintenant l'équation 6.20 au cas des variables indépendantes.

6.3.3.1 Cas des variables Bernoulli indépendantes

Nous utilisons l'indépendance des I_i dans l'équation 6.20. Nous pouvons écrire

$$E(I_i g(W)) = E(I_i g(W_i + I_i)) = E(I_i g(W_i + 1)),$$

où $W_i = \sum_{j \neq i} I_j$. L'équation 6.20 nous donne alors

$$|E(f(W)) - P_\chi(f)| = \left| \sum_1^n p_i E[g(W+1) - g(W_i + 1)] \right|.$$

En remarquant que $g(W+1) - g(W_i+1) = 0$ si $I_i = 0$, nous avons

$$\{g(W+1)-g(W_i+1)\} = \{g(W+1)-g(W_i+1)|I_i = 1\} = \{g(W_i+2)-g(W_i+1)|I_i = 1\}.$$

Et donc, nous obtenons, les variables I_i et $g(W_i+2) - g(W_i+1)$ étant indépendantes,

$$|E(f(W)) - P_\chi(f)| = \left| \sum_1^n p_i E(I_i) E[g(W_i + 2) - g(W_i + 1)] \right|$$

Ceci nous donne la majoration

$$|E(f(W)) - P_\chi(f)| \leq ||\Delta g|| \left| \sum_1^n p_i^2 \right|.$$

Le théorème 46 nous donne $||\Delta g|| \leq 1 \wedge \frac{1}{\chi}$, où $\chi = \sum p_i$; nous obtenons donc,

$$||\mathcal{L}(W) - P_\chi|| \leq ||\Delta g|| \max_{1 \leq i \leq n} p_i \sum_1^n p_i \leq \max_{1 \leq i \leq n} p_i.$$

C'est le résultat obtenu par Hodges et Le Cam.

6.3.3.2 Cas des variables Bernoulli faiblement dépendantes

Nous allons considérer maintenant le cas de dépendances "locales" entre les variables Bernoulli, cas que nous pourrons appliquer à l'alignement local de séquences, dans le cas simplifié de recherche d'un plus grand mot commun à ces deux séquences.

Notations. Pour $1 \leq i \leq n$, on considère $V_i \subset U_i = [\{1, \ldots, n\} - \{i\}]$ et $V_i^c = U_i - V_i$, complémentaire de V_i dans l'ensemble U_i. On note $Z_i = \sum_{k \in V_i} I_k$, et $Y_i = \sum_{j \in V_i^c} I_j$. De la sorte, nous avons $W = \sum_{j=1}^n I_j = Y_i + Z_i + I_i$. L'ensemble V_i^c est l'ensemble des indices j pour lesquels I_j est "presque"indépendant de I_i, c'est à dire pour lesquels $E(I_i I_j) \approx p_i p_j$. Ceci correspond à la notion de variables mélangeantes (voir Durrett, "mixing variables").

Nous pouvons alors énoncer le théorème suivant.

Théorème 48. *Avec les notations utilisées, soit W le nombre d'occurrences d'événements dépendants, et P_χ une variable aléatoire de Poisson de moyenne $E(P_\chi) = E(W) = \chi$. Nous avons alors*

$$||\mathcal{L}(W) - P_\chi|| \le \left(1 \wedge \frac{1}{\chi}\right)\left(\sum_1^n p_i^2 + p_i E(Z_i) + E(I_i Z_i)\right) + \left(1 \wedge \frac{1}{\sqrt{\chi}}\right)\sum_1^n \eta_i, \tag{6.21}$$

où $\eta_i = |E(E_i/I_j, j \in V_i^c) - p_i|$.

Preuve. En notant toujours $W_i = \sum_{j\neq i} I_j, 1 \le i \le n$, l'équation 6.20 s'écrit à présent

$$\begin{aligned}&|E(f(W)) - P_\chi(f)| = \\ &\quad\left|\sum_1^n p_i E[g(W+1) - g(W_i+1)] + \sum_1^n [p_i E(g(W_i+1)) - E(I_i g(W))]\right|.\end{aligned} \tag{6.22}$$

En remarquant que $E(g(W+1) - g(W_i+1)) \le p_i \Delta g(W_i+1)$, ceci nous donne

$$|E(f(W)) - P_\chi(f)| \le \sum_1^n p_i^2 ||\Delta g|| + \left|\sum_1^n E[p_i g(Y_i + Z_i + 1)) - I_i g(Y_i + Z_i + 1)]\right|.$$

Nous pouvons écrire

$$g(Y_i + Z_i + 1) = g(Y_i+1) + \sum_{j=1}^{Z_i}[g(Y_i + j + 1) - g(Y_i + j)] = g(Y_i + 1) + D_i,$$

en notant $D_i = \sum_1^{Z_i}[g(Y_i + j + 1) - g(Y_i + j)]$. Nous obtenons

$$E[p_i g(Y_i + Z_i + 1)) - I_i g(Y_i + Z_i + 1)] = p_i E(g(Y_i+1)) - E(I_i g(Y_i+1)) + E(p_i D_i - I_i D_i). \tag{6.23}$$

Nous avons d'autre part

$$D_i = \left|\sum_1^{Z_i}[g(Y_i + j + 1) - g(Y_i + j)]\right| \le ||\Delta g|| Z_i.$$

Nous tirons donc de l'équation 6.22

$$\begin{aligned}|E(f(W)) - P_\chi(f)| \le\ & ||\Delta g|| \sum_1^n \left[p_i^2 + p_i E(Z_i) + E(I_i Z_i)\right] \\ & + \left|\sum_1^n [p_i E(g(Y_i+1)) - E(I_i g(Y_i+1))]\right|.\end{aligned} \tag{6.24}$$

Considérons l'expression

$$A_i = E(g(Y_i+1)(p_i - I_i)).$$

Pour deux variables aléatoires quelconques X et Y, et une fonction g de carré intégrable, nous avons (voir Dacunha-Castelle [21], T.1, proposition 1.4.3)

$$E(g(X)Y) = E(g(X)E(Y/X)).$$

En appliquant cette proposition à l'expression A_i, notre fonction g étant bornée, et donc de carré intégrable sur $\mathbb{N} \cap [0, n]$, nous avons pour le dernier terme de l'équation 6.24 la majoration

$$\left| \sum_1^n [p_i E(g(Y_i + 1)) - E(I_i g(Y_i + 1))] \right| \leq \sum_1^n E\left(|g(Y_i + 1)| \times |E(p_i - I_i/I_j, j \in V_i^c)|\right).$$

Si nous notons $\eta_i = |E(I_i/I_j, j \in V_i^c) - p_i|$, η_i mesure la "faible dépendance"de I_i et des variables I_j pour $j \in V_i^c$. En récapitulant, l'équation 6.24 nous donne

$$|E(f(W)) - P_\chi(f)| \leq ||\Delta(g)|| \left(\sum_1^n p_i^2 + p_i E(Z_i) + E(I_i Z_i) \right) + ||g|| \sum_1^n \eta_i. \tag{6.25}$$

Compte tenu des bornes du théorème 46 pour g, nous obtenons

$$||\mathcal{L}(W) - P_\chi|| \leq \left(1 \wedge \frac{1}{\chi}\right) \left(\sum_1^n p_i^2 + p_i E(Z_i) + E(I_i Z_i) \right) + \left(1 \wedge \frac{1}{\sqrt{\chi}}\right) \sum_1^n \eta_i. \tag{6.26}$$

□

Exemple 49 (Record de tirages piles).

Waterman [85] donne dans son livre le résultat suivant pour le nombre de record R_n de suite de piles d'une variable de Bernoulli X "pile-face", sur n tirages indépendants, où pile compte un et face zéro, et $P(X = 1) = p = 1 - P(X = 0) = 1 - q$:

$$E(R_n) \approx \log_{1/p}(n) + \log_{1/p}(q) + \gamma/\chi - 1/2, \tag{6.27}$$

où γ est la constante d'Euler, et, comme précédemment, on a $\chi = np$. Ce résultat est donné sans démonstration. Ce que nous cherchons, pour pouvoir l'appliquer à l'aligneent de séquences biologiques, est une mesure probabiliste pour les suites de piles de longueur dépassant $E(R_n)$. Nous voulons donc une estimation fiable de $\Pr(R_n \geq t)$, la probabilité d'obtenir une suite de piles de longueur au moins égale à t. Nous nous intéresserons plus particulièrement au comportement asymptotique de R_n quand $n \to +\infty$.

On peut remarquer que les longues suites de piles sont des événements rares, de faible probabilité, et que, pour $t \ll n$, il y a de nombreux endroits où ces événements rares peuvent arriver. On s'attend donc à obtenir à la limite une loi de Poisson. Cependant, on ne peut pas appliquer directement l'approximation de Poisson à l'événement "une suite de piles commence en i", parce que ces événements arrivent en "groupes", ("clumps"en anglais). Si un tel événement (une suite de pile de longueur t) arrive à la position i, avec une probabilité p, une autre suite de piles de longueur t commencera en $i + 1$. En moyenne, si on prend une suite de tirages équilibrés, $p = 1/2$, de telles suites commencent toutes les 2^t positions, mais l'effet de groupes fait que la distance moyenne séparant deux groupes est très supérieure à 2^t. Comme nous nous intéressons à $P(R_n > t)$, nous pouvons ne considérer que la première suite de longueur t dans un groupe, et, de cette manière, "dégrouper" le modèle. L'heuristique de groupes ("clumping heuristic") formulée par Aldous [2] dit que le nombre de groupes est Poisson à la limite. L'idée est que les positions des groupes suivent une loi de Poisson, et que la longueur des groupes vérifie une autre loi, indépendante de la première. Dans le cas d'une suite de piles, la loi de la longueur des groupes est une loi géométrique où la valeur k s'obtient avec probabilité $(1-p)p^k(1-p) = p^k(1-p)^2$. Nous allons utiliser la méthode de Chen-Stein, et plus particulièrement le théorème 48, pour trouver la limite de R_n, ainsi que des bornes d'erreur.

Notations. X_i est une suite de variables de Bernoulli indépendantes de paramètre p. Nous notons R_n le nombre maximum de 1 consécutifs pour cette suite.

Pour $t \leq n$ et $i \leq n$, nous posons

$$\begin{aligned} &I_i = 1_{\{X_i=0, X_{i+1}=1,\ldots,X_{i+t}=1\}}, \quad \text{si} \quad 1 < i \leq n-t, \\ &I_1 = 1_{\{X_1=1,X_2=1,\ldots,X_t=1\}}, \quad \text{et} \\ &I_i = 0, \quad \forall i > n-t. \end{aligned}$$

Nous avons l'égalité

$$\{R_n < t\} = \{W = 0\},$$

où W désigne comme dans le théorème précédant un nombre d'occurrences d'événements dépendants, ici le nombre d'occurrences de groupes de piles de longueur supérieure ou égale à t. D'autre part, les variables I_i et I_j sont indépendantes si $|i-j| > t$.

En reprenant les notations du théorème 48, notons $V_i = \{j/\ 0 < |i-j| \leq t\}$. Les termes η_i du théorème 48 sont alors tous nuls, compte tenu de l'indépendance des $I_j \in V_i^c$ et des I_i, pour tout i.

Nous avons les égalités

$$\begin{aligned} &p_1 = P(I_1) = p^t, \\ &p_i = P(I_i) = (1-p)p^t, \ \text{si} \ \ 1 < i \leq n-t, \ \ \text{et} \\ &\chi = E(W) = (n-t-1)p^t(1-p) + p^t. \end{aligned}$$

Nous avons aussi $E(I_iZ_i) = 0$. En effet on a $E(I_iZ_i) = E(I_i \sum_{k\in V_i} I_k) = \sum_{k\in V_i}(I_iI_k)$; de plus, si $I_i = 1, (i > 1)$, on a $X_i = 0$, et donc $I_j = 0$, pour tout j tel que $0 \vee (i-t) \leq j < i$, et si $I_j = 1$, pour tout j tel que $i < j \leq (i+t) \wedge (n-t)$, on a $X_j = 0$.

Nous avons donc, d'après le théorème 48 (Chen-Stein)

$$||\mathcal{L}(W) - P_\chi|| \leq \left(1 \wedge \frac{1}{\chi}\right)\left(p_1^2 + E(I_1Z_1) + \sum_{i=2}^{n-t+1}[p_i^2 + E(I_iZ_i)]\right) = \left(1 \wedge \frac{1}{\chi}\right) b(n,t,p). \tag{6.28}$$

Le terme $b(n,t,p)$ est borné de la manière suivante.

$$b(n,t,p) < p^t[2t(1-p)p^t + p^t] + (n-t)(2t+1)[(1-p)p^t]^2 \tag{6.29}$$

$$< (2t+1)[(1-p)p^t]^2\left\{n - t + \frac{1}{1-p} + \frac{p}{(1-p)^2(2t+1)}\right\}. \tag{6.30}$$

Nous avons en particulier

$$|\Pr(R_n < t) - e^{-\chi}| = |\Pr(W=0) - e^{-\chi}| \leq ||\mathcal{L}(W) - P_\chi|| \leq \left(1 \wedge \frac{1}{\chi}\right) b(n,t,p).$$

Les probabilités qui nous intéressent correspondent au cas où χ n'est ni très petit (≈ 0), ni trop grand ($\approx +\infty$). Nous avons $\chi = E(W) = (n-t-1)p^t(1-p)+p^t$, et donc, le terme dominant pour χ est np^t. Nous voulons que np^t reste borné, ou, de manière équivalente, que $\log(np^t) = t - \log_{1/p}(n)$ reste borné. Nous choisirons plus précisément une variable x telle que

$$np^t(1-p) = p^x,$$

ou, de manière équivalente,

$$t = \log_{1/p}(n(1-p)) + x.$$

Dans ces conditions, nous avons

$$b(n,t,p) < \frac{2t+1}{n}p^{2x}\left(1 - \frac{t}{n} + \frac{1}{n(1-p)} + \frac{p}{n(1-p)^2(2t+1)}\right) \tag{6.31}$$

$$= \mathcal{O}\left(\frac{\log(n)}{n}\right). \tag{6.32}$$

Nous obtenons donc

$$|\{\Pr(R_n < \log_{1/p}[n(1-p)] + x)\} - e^{-\chi}| = \mathcal{O}\left(\frac{\log(n)}{n}\right). \tag{6.33}$$

Quelques manipulations algébriques nous montrent que nous avons trouvé une formule asymptotique identique à celle d'Iglehart-Karlin 2.1, dans un cas plus simple (les records de piles consécutifs), mais avec une borne sur la convergence. En effet, posons $\lambda = \log(1/p)$. En faisant apparaître les compléments à 1 des membres du premier terme de l'équation 6.33, nous obtenons

$$\left|\left\{\Pr(R_n \geq \log_{1/p}(n(1-p)) + x)\right\} - \{1 - e^{-\chi}\}\right| = \mathcal{O}\left(\frac{\log(n)}{n}\right).$$

Ceci peut s'écrire, en posant $\lambda = \log(1/p)$,

$$\lim_{n\to+\infty} \Pr\left(R_n \geq \frac{\log(n)}{\lambda} + x + \frac{\log(1-p)}{\lambda}\right) = 1 - e^{-\chi} = 1 - e^{-p^x}.$$

Nous avons $p = e^{-\lambda}$; en faisant le changement de variable $y = x + \frac{1}{\lambda}\log(1-p)$, nous obtenons

$$\Pr\left(R_n \geq \frac{\log n}{\lambda} + y\right) \to 1 - e^{-Ke^{-\lambda y}},$$

pour un K constant.

Arratia et *al.* étendent ce résultat au match exact de deux séquences [6] (on peut consulter le livre de Waterman [85] pour cette extension).

6.4 Approximations Poissonniennes et alignement de séquences

Nous avons abordé le cas de l'alignement de séquences sans gaps au paragraphe 2.1 du chapitre 2 Nous allons à présent redéfinir ce qu'est l'alignement de séquences dans un cadre beaucoup plus général, celui qui considère également les alignements avec gaps. Waterman et Vingron [86] ont appliqué la puissante méthode de Chen-Stein développée dans ce chapitre, ainsi que l'heuristique de clumping-declumping d'Aldous (voir paragraphe précédant), pour calibrer statistiquement les méthodes d'alignement avec gaps, en fonction des pénalités des gaps. Nous utiliserons cette approche remarquablement efficace pour calibrer l'algorithme BlastMultAl (chapitre 5) pour les multi-alignements de ProDom. Nous suivrons fréquemment Waterman et Vingron [86] dans ce paragraphe.

6.4.1 Le modèle des alignements de séquences

Nous considérons deux séquences

$$\begin{aligned} \mathbf{x} &= x_1 x_2 \dots x_n, \quad \text{et} \\ \mathbf{y} &= y_1 y_2 \dots y_m, \end{aligned}$$

prises sur un alphabet fini. Pour l'ADN, cet alphabet aura quatre lettres; il aura vingt lettres pour les protéines. Dans un premier temps, nous considérons également que le choix des lettres est déterministe, puis nous passerons au cas des lettres choisies de manière aléatoire. Nous nous donnons une fonction de score $s(x, y)$ pour l'alignement de deux lettres x et y. Ceci ne concerne le changement de lettres (x se "substitue" à y), et les insertions ou substitutions (indels).

Par exemple, soit

$$s(x, y) = \begin{cases} +1, & x = y, \\ -\mu, & x \neq y \end{cases} \tag{6.34}$$

et soit $-\delta$ le score des indels. L'alignement

A T A G C
A A G C C

a pour score $S = 2 - 3\mu$. Suivant les insertions suppression effectuées, on peut changer le score. L'alignement

A T A G C –
A – A G C C

a pour score $S = 4 - 2\delta$. Le résultat de l'alignement dépend donc fortement du choix de la paire (μ, δ).

6.4.1.1 Alignements globaux

Un alignement global est un alignement dans lequel toutes les lettres des deux séquences sont représentées. Un tel alignement est parfaitement défini une fois déterminées les lettres alignées. Les autres lettres sont alignées avec un caractère de gap. Le score de l'alignement peut, du même coup être calculé une fois connues les lettres alignées. S'il y a k lettres alignées, l'alignement est donc défini par les suites

$$\mathcal{I}(\mathbf{x}) = \{1 \leq i_1 < i_2 < \cdots < i_k \leq n\},$$
$$\mathcal{I}(\mathbf{y}) = \{1 \leq j_1 < j_2 < \cdots < j_k \leq m\}.$$

Pour calculer le nombre d'alignements globaux possibles $N_{n,m}$, on utilise la convolution de Vandermonde. Celle-ci s'obtient en développant les termes de l'identité $(1+x)^n(1+x)^m = (1+x)^{n+m}$, et en identifiant les coefficients de x^{r+s}. Ceci donne l'identité

$$\sum_k \binom{n}{r+k}\binom{m}{s-k} = \binom{n+m}{r+s}$$

	–	**T**	**A**	**T**	**C**	**T**	**T**	**A**	**A**	**C**	**G**	**C**	**C**
–	0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-10	-11	-12
G	-1	-1	-2	-3	-4	-5	-6	-7	-8	-9	-8	-9	-10
A	-2	-2	0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-10
T	-3	-1	-1	1	0	-1	-2	-3	-4	-5	-6	-7	-8
C	-4	-2	-2	0	2	1	0	-1	-2	-3	-4	-5	-6
A	-5	-3	-1	-1	1	1	0	1	0	-1	-2	-3	-4
A	-6	-4	-2	-2	0	0	0	1	2	1	0	-1	-2
T	-7	-5	-3	-1	-1	1	1	0	1	1	0	-1	-2
T	-8	-6	-4	-2	-2	0	2	1	0	0	0	-1	-2
C	-9	-7	-5	-3	-1	-1	1	1	0	1	0	1	0
G	-10	-8	-6	-4	-2	-2	0	0	0	0	2	1	0
C	-11	-9	-7	-5	-3	-3	-1	-1	-1	1	1	3	2
A	-12	-10	-8	-6	-4	-4	-2	0	0	0	0	2	2

Tableau 6.1: Un exemple d'alignement global (tiré de Waterman et Vingron)

Cette dernière identité donne une nouvelle identité sur les sommes de binomiaux (voir Graham, Knuth et Patashnik [33])

$$\sum_k \binom{n}{r+k}\binom{m}{s+k} = \binom{m+n}{n-r+s} \tag{6.35}$$

Le nombre d'alignements $N_{n,m}$ est calculé en sommant sur k le nombre de paires alignées, ce qui nous donne

$$N_{n,m} = \sum_k \binom{n}{k}\binom{m}{k} = \binom{n+m}{n}$$

La dernière équation est tirée de l'identité 6.35 en y faisant $r = s = 0$.

Nous cherchons le meilleur alignement de $\mathbf{x}$ et $\mathbf{y}$, tel que

$$\mathbf{S}(\mathbf{x}, \mathbf{y}) = \max \mathcal{S}(\mathcal{I}(\mathbf{x}), \mathcal{I}(\mathbf{y})),$$

où $\mathcal{S}$ est calculé comme la somme des scores des lettres alignés et des pénalités d'alignement d'une lettre et d'un caractère gap.

Au lieu de chercher l'alignement optimal (celui de meilleur score) parmi le nombre exponentiel d'alignements, on peut calculer directement cet alignement en $\mathcal{O}(nm)$ étapes par la méthode de programmation dynamique. Étant donné les séquences $x_1 \dots x_n$ et $y_1 \dots y_m$, l'alignement

$$\mathcal{A} = \left\{ \begin{array}{l} x_{i_1} \dots x_{i_u} \dots x_{i_k} \\ y_{j_1} \dots y_{j_u} \dots y_{j_l} \end{array} \right\}$$

a comme score, sachant que deux caractères de gaps ne sont jamais alignés,

$$\mathcal{S}(\mathcal{A}) = \sum_{\forall i_u, j_v, x_{i_u} \neq -, y_{j_v} \neq -} s(x_{i_u}, y_{j_v}) + \delta g.$$

Dans cette dernière équation , la variable g compte le nombre de caractères gaps dans l'alignement. En considérant le cas possible d'un gap aligné avec le premier caractère d'une des deux séquences, initialisons les valeurs $S_{0,j}$ et $S_{i,0}$, comme

$$S_{0,j} = -\delta j \quad \text{et} \quad S_{i,0} = -\delta i.$$

La définition récursive de l'algorithme permettant de calculer S_{ij} utilise la programmation dynamique, comme suit.

$$S_{i,j} = \max \left\{ \begin{array}{l} S_{i-1,j} - \delta, \\ S_{i,j-1} - \delta, \\ S_{i-1,j-1} + s(x_i, y_j) \end{array} \right. \tag{6.36}$$

On a $S_{n,m} = \mathbf{S}(\mathbf{x}, \mathbf{y})$. Cet algorithme part de l'alignement optimal $S_{i-1,j-1}$ pour construire l'alignement optimal $S_{i,j}$ en considérant les trois manières dont cet alignement peut se terminer, $(x_i, -), (-, y_j), (x_i, y_j)$.

Le tableau 6.1 donne un exemple d'alignement avec

$$s(x, y) = \left\{ \begin{array}{lll} +1 & \text{si} & x = y, \\ -1 & \text{si} & x \neq y, \end{array} \right. \qquad \text{et} \quad \delta = 1.$$

Les deux alignements optimaux sont représentés par des valeurs encadrées dans la matrice du tableau 6.1. Ces deux alignements sont

$$\begin{array}{l} \mathrm{G\ A\ T\ C} - - \mathrm{A\ A\ T\ T\ C\ G\ C\ A} \\ \mathrm{T\ A\ T\ C\ T\ T\ A\ A} - - \mathrm{C\ G\ C\ C} \end{array}$$

et

$$\begin{array}{l} \mathrm{G\ A\ T\ C\ A\ A\ T\ T} - - \mathrm{C\ G\ C\ A} \\ \mathrm{T\ A\ T\ C} - - \mathrm{T\ T\ A\ A\ C\ G\ C\ C} \end{array}$$

6.4.1.2 Alignements locaux

Nous passons à présent aux alignements locaux. Un alignement local de deux séquences $\mathbf{x}$ et $\mathbf{y}$ correspond à l'alignement global d'un intervalle de $\mathbf{x}$ et d'un intervalle de $\mathbf{y}$. Dans une séquence de longueur n, il y a $\binom{n}{2}$ intervalles $[i, j]$ tels que $1 \leq i < j \leq n$; il y a n intervalles de la forme $[i, i]$, et si l'on tient compte de l'intervalle vide, on obtient un total de $\binom{n}{2} + n + 1 = \binom{n+1}{2} + 1$ intervalles.

Si l'on considère que les longueurs respectives de $\mathbf{x}$ et $\mathbf{y}$ sont n et m, le problème de l'alignement local de ces séquences correspond donc à

$$\left(\binom{n+1}{2} + 1\right)\left(\binom{m+1}{2} + 1\right)$$

problèmes d'alignements globaux.

Nous noterons $\mathbf{H}(\mathbf{x}, \mathbf{y})$ la fonction de score que nous cherchons à maximiser, qui est à présent.

$$\begin{aligned}
\mathbf{H}(\mathbf{x}, \mathbf{y}) &= \max \begin{cases} 0 \\ \mathbf{S}(I, J); \quad I \subset \mathbf{x}, J \subset \mathbf{y} \end{cases} \\
&= \max \begin{cases} 0 \\ \mathbf{S}([x_k, x_{k+1}, \ldots, x_i], [y_l, y_{l+1}, \ldots, y_j]) \\ \qquad 1 \leq k \leq i \leq n,\ 1 \leq l \leq j \leq m \end{cases}
\end{aligned}$$

Nous avons à nouveau une formule récursive pour la fonction de score. Posons

$$H_{i,j} = \max \begin{cases} 0 \\ \mathbf{S}([x_k, \ldots, x_i], [y_l, \ldots, y_j]) \\ \qquad 1 \leq k \leq i,\ 1 \leq l \leq j \end{cases}$$

$H_{i,j}$ est le score du meilleur alignement se terminant en x_i et y_j, ou 0, si aucun alignement n'a un score positif. L'algorithme est initialisé avec $H_{i,j} = 0$, si $i = 0$ ou si $j = 0$. On a alors

$$H_{i,j} = \max \begin{cases} H_{i-1,j} - \delta, \\ H_{i,j-1} - \delta, \\ H_{i-1,j-1} + s(i, j), \\ 0 \end{cases} \tag{6.37}$$

Nous avons donc

$$\mathbf{H}(\mathbf{x}, \mathbf{y}) = \max_{1 \leq i \leq n; 1 \leq j \leq m} H_{i,j}.$$

Cet algorithme est connu sous le nom d'algorithme de Smith et Waterman [75].

Le tableau 6.2 donne un exemple d'application de cet algorithme avec

$$\left\{ \begin{array}{ll} s(x, y) = 1, & \text{si } x = y, \\ s(x, y) = -1, & \text{si } x \neq y, \end{array} \right| \qquad \text{et} \qquad \delta = 1.$$

(a) - $H_{i,j}$

	–	**T**	**A**	**T**	**C**	**T**	**T**	**A**	**A**	**C**	**G**	**C**	**C**
–	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0	0	0	0	1	0	0
A	0	0	$\boxed{1}$	0	0	0	0	1	1	0	0	0	0
T	0	1	0	$\boxed{2}$	1	1	1	0	0	0	0	0	0
C	0	0	0	1	$\boxed{3}$	$\boxed{2}$	$\boxed{1}$	0	0	1	0	1	1
A	0	0	1	0	$\boxed{2}$	2	1	$\boxed{2}$	1	0	0	0	0
A	0	0	1	0	$\boxed{1}$	1	1	2	$\boxed{3}$	2	1	0	0
T	0	1	0	2	1	$\boxed{2}$	2	1	$\boxed{2}$	2	1	0	0
T	0	1	0	1	1	2	$\boxed{3}$	$\boxed{2}$	$\boxed{1}$	1	1	0	0
C	0	0	0	0	2	1	2	2	1	$\boxed{2}$	1	2	1
G	0	0	0	0	1	1	1	1	1	1	$\boxed{3}$	2	1
C	0	0	0	0	1	0	0	0	0	2	2	$\boxed{4}$	3
A	0	0	1	0	0	0	0	1	1	1	1	3	3

(b) - $H'_{i,j}$

	–	**T**	**A**	**T**	**C**	**T**	**T**	**A**	**A**	**C**	**G**	**C**	**C**
–	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0	0	0	0	1	0	0
A	0	0	$\underline{0}$	0	0	0	0	1	1	0	0	0	0
T	0	1	0	$\underline{0}$	1	1	1	0	0	0	0	0	0
C	0	0	0	1	$\underline{0}$	$\underline{0}$	$\underline{0}$	0	0	1	0	1	1
A	0	0	1	0	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	1	0	0	0	0
A	0	0	1	0	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	2	1	0	0
T	0	1	0	2	1	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	2	1	0	0
T	0	1	0	1	1	2	$\underline{0}$	$\underline{0}$	$\underline{0}$	1	1	0	0
C	0	0	0	0	2	1	2	2	1	$\underline{0}$	1	2	1
G	0	0	0	0	1	1	1	1	1	1	$\underline{0}$	2	1
C	0	0	0	0	1	0	0	0	0	2	2	$\underline{0}$	3
A	0	0	1	0	0	0	0	1	1	1	1	3	3

(c) - $H''_{i,j}$

	–	**T**	**A**	**T**	**C**	**T**	**T**	**A**	**A**	**C**	**G**	**C**	**C**
–	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0	0	0	0	1	0	0
A	0	0	$\underline{0}$	0	0	0	0	1	1	0	0	0	0
T	0	1	0	$\underline{0}$	$\underline{0}$	1	1	0	0	0	0	0	0
C	0	0	0	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	0	0	1	0	1	1
A	0	0	1	0	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	1	0	0	0	0
A	0	0	$\boxed{1}$	0	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{1}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	0	0
T	0	1	0	$\boxed{2}$	1	$\underline{1}$	$\underline{1}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	0	0
T	0	1	0	1	1	2	$\underline{2}$	$\underline{1}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	0	0
C	0	0	0	0	2	1	$\underline{1}$	$\underline{1}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{1}$	1
G	0	0	0	0	1	1	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$
C	0	0	0	0	1	0	0	0	0	$\underline{1}$	$\underline{0}$	$\underline{0}$	$\underline{1}$
A	0	0	1	0	0	0	0	1	1	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$

Tableau 6.2: Un exemple d'alignement local (tiré de Waterman et Vingron). (a) Alignements optimaux. (b) Matrice transitoire où le clump de score optimal a été mis à zéro. (c) Matrice déclumpée et recalculée, et alignement sous-optimal.

Les alignements optimaux sont

$$\begin{array}{ccccccccccc} A & T & C & - & - & A & A & T & T & C & G & C \\ A & T & C & T & T & A & A & - & - & C & G & C \end{array}$$

et

$$\begin{array}{ccccccccccc} A & T & C & A & A & T & T & - & - & C & G & C \\ A & T & C & - & - & T & T & A & A & C & G & C \end{array}$$

Nous avons abordé les événements arrivant en "clumps" au paragraphe 6.3.3.2, lorsque nous avons étudié les records de piles dans des tirages de pièces de monnaie. L'apparition de plusieurs alignements optimaux pour notre exemple est un nouvel exemple d'événements groupés, arrivant en clumps. Nous allons appliquer la même méthode que pour les records de piles, en ne cherchant à compter que des clumps séparés. Pour ce faire, il faut déclumper, ce que Waterman et Vingron font de la manière suivante.
Declumping pour l'algorithme de Smith et Waterman. Dans un premier temps, on détermine le clump, en remontant du point de score $H_{i,j}$ maximum. Si une position d'indices (j,k) est dans le clump, on mettra dans le clump les positions d'indices (u,v) telles que l'on ait $(u,v) \in \{(k-1,l-1),(k-1,l),(k,l-1)\}$, $S_{u,v} > 0$ et $S_{u,v} = \max(S_{k-1,l-1}, S_{k-1,l}, S_{k,l-1})$. Le clump est "encadré" par au plus deux alignements optimaux. Toute la région comprise entre ces deux alignements optimaux est remise à zéro, comme on le voit sur la partie (b) du tableau 6.2, où les valeurs remises à zéro ont été soulignées.

Ceci définit une matrice $H'_{i,j}$. Dans une deuxième étape, on recalcule la matrice, en repartant du coin le plus en haut et le plus à gauche du clump. Soient (k,l) les coordonnées de ce point. Le tableau $H'_{i,j}$ est progressivement mis à jour en partant de ce point, et en progressant

- ligne par ligne, en incrémentant l'indice i de la ligne à partir de k
- le long d'une ligne, en incrémentant l'indice j de la colonne.

L'algorithme s'écrit donc

$$\left\{ \begin{array}{l} H'_{i,j} = H_{i,j}, \quad \text{si} \quad i < k \text{ ou } j < l, \\ H'_{r,s} = \max \left\{ \begin{array}{l} 0, \\ H'_{r-1,s} - \delta, \\ H'_{r,s-1} - \delta, \\ H'_{r,s} + s(x_r, y_s) \end{array} \right. \quad , \quad s \geq k, r \geq l \end{array} \right. \tag{6.38}$$

On peut faire les remarques suivantes sur cette mise à jour.

- Les valeurs des éléments de la matrice $H'_{i,j}$, après remise à zéro des éléments du clump, sont inférieures ou égales à celles de la matrice $H_{i,j}$. Supposons que après avoir mis à jour la ligne $i-1$, κ_{i-1} soit un indice tel que $H'_{i-1,j} = H_{i-1,j}$, $\forall j \geq \kappa_{i-1}$. Ceci qui veut dire qu'après le rang κ_{i-1}, tous les éléments de la ligne $i-1$ de la matrice $H'_{i,j}$ sont égaux à ceux de la matrice $H_{i,j}$. Quand on met à jour la ligne i, si on a $r \geq \kappa_{i-1}$, et $H'_{i,r} = H_{i,r}$, alors tous les éléments de la ligne i de rang supérieur à r de la matrice H' seront égaux aux éléments correspondants de la matrice H. On peut donc arrêter la progression de l'algorithme sur une ligne dès que l'on a trouvé une égalité entre un élément de H et l'élément correspondant de H'.

- L'algorithme de mise à jour étant symétrique en terme de lignes et de colonnes, on peut faire la même remarque en ce qui concerne la progression sur une colonne.

Dans la matrice H'' (tableau 6.2 (c)), les positions qui ont été mises à jour par l'algorithme sont représentées par des valeurs soulignées.

Ces deux remarques permettent à Waterman et Vingron d'avoir un algorithme de mise à jour de la matrice qui est linéaire et non quadratique en la taille des séquences.

Cette méthode de "declumping" permet d'obtenir itérativement le deuxième, troisième, *etc.* meilleurs clumps, de scores respectifs $H_{(1)}, H_{(2)}, H_{(3)}, \ldots$, et on a

$$H_{(1)} \geq H_{(2)} \geq H_{(3)} \ldots$$

Dans le tableau 6.2 (c), nous avons encadré le premier alignement sous-optimal obtenu après declumping du meilleur alignement. Il s'agit de l'alignement

$$\begin{array}{c} AT \\ AT. \end{array}$$

Il existe en général plusieurs alignements sous-optimaux à ce stade, ce qui nécessite autant d'opérations de declumping.

Cette méthode peut être généralisée au cas classique en biologie où l'on considère une pénalité d'ouverture de gap α et une pénalité d'extension de gap β pour un coût total de gap $g(k) = \alpha + k\beta$ pour un indel de k lettres.

6.4.1.3 Résultats en distribution

Nous allons considérer à présent des séquences aléatoires $\mathbf{X} = X_1X_2 \ldots X_n, \mathbf{Y} = Y_1Y_2 \ldots Y_n$, où les X_i d'une part, et les Y_i d'autre part, sont des variables indépendantes et identiquement distribuées. Un modèle utile à considérer est le modèle à deux paramètres

$$s(X_i, Y_i) = \begin{cases} 1 & \text{si } X_i = Y_i, \\ -\mu & \text{si } X_i \neq Y_i \quad (\mu \geq 0) \end{cases}$$

$$\text{et} \quad \delta \geq 0, \qquad \text{pénalité d'ouverture ou d'extension de gap.}$$

Les quantités qui nous intéressent sont les scores $G_n = \mathbf{G}(\mathbf{X}, \mathbf{Y})$ de l'alignement global optimal et $H_n = \mathbf{H}(\mathbf{X}, \mathbf{Y})$ de l'alignement local optimal. Par construction, on a

$$G_n \leq H_n.$$

En écrivant

$$G_n = G(X_1 \ldots X_n, Y_1 \ldots Y_n),$$

on peut observer que

$$G_{n+m} \geq G_n + G(X_{n+1} \ldots X_{n+m}, Y_{n+1} \ldots Y_{n+m}).$$

D'autre part, $G(X_{n+1} \ldots X_{n+m}, Y_{n+1} \ldots Y_{n+m})$ a même distribution que G_m. Le théorème sous-additif ergodique de Kingman [55] (voir Durrett, p. 361) s'applique alors, qui nous donne

$$\lim_{n \to \infty} \frac{G_n}{n} = \rho_{\mu,\delta} \quad \text{presque sûrement,} \tag{6.39}$$

où $\rho_{\mu,\delta}$ est une fonction de μ et δ. Cependant, Si nous prenons le cas

$$\begin{cases} \mu = \infty \quad \text{et} \\ \delta = 0, \end{cases}$$

le score de l'alignement global optimal G_n est égal à la longueur de la plus longue sous-séquence commune (les indels ne sont pas pénalisés, mais les substitutions sont interdites). La constante $\rho_{\infty,0}$ est alors connue sous le nom de constante de Chvátal-Sankoff [17]. Malheureusement, même dans le cas d'un alphabet de deux lettres $\{0, 1\}$ pour les deux séquences $\mathbf{X}$ et $\mathbf{Y}$, tel que $\Pr(X_i = 1) = 1 - \Pr(X_i = 0) = \Pr(Y_i = 1) = 1 - \Pr(Y_i = 0)$, cette constante est inconnue.

Quand $\rho_{\mu,\delta}$ est positif, l'inégalité $G_n \leq H_n \leq n$ permet de montrer que

$$\Pr\left(\lim_{n\to\infty} \frac{H_n}{n} = \rho_{\mu,\delta}\right) \to 1.$$

En d'autre termes, le rapport H_n/n tend vers $\rho_{\mu,\delta}$ en probabilité (voir Waterman [85] p. 296). Waterman et Arratia [7] montrent que, quand $\rho_{\mu,\delta}$ est négatif, il existe une constante $\beta_{\mu,\delta}$, fonction de μ et δ, telle que

$$\beta_{\mu,\delta} \leq \lim_{n\to\infty} \frac{H_n}{\log(n)} \leq 2b.$$

La constante b, fonction de μ et de δ s'obtient de la manière suivante. Soit

$$r(q) = \lim_{n\to\infty} \frac{-\log(\Pr(S_n \geq qn))}{n} = \inf\left(-\frac{\log(\Pr(S_n \geq qn))}{n}\right).$$

Nous avons alors

$$b = b(\mu, \delta) = \max_{q\geq 0} \frac{q}{r(q)}.$$

Récemment, Zhang [89] a montré que, dans le cas $\rho_{\mu,\delta} < 0$, on a

$$\Pr\left(\frac{H_n}{\log(n)} \to 2b\right) = 1.$$

Dans le cas

$$\begin{cases} \mu = \infty \quad \text{et} \\ \delta = \infty, \end{cases}$$

nous cherchons le plus long mot commun à deux sous-séquences, problème de formalisation très voisine du problème de record de piles (exemple 49), et qui donne de manière identique une limite pour H_n en $\mathcal{O}(\log(n))$.

On voit donc apparaître deux comportements limites, en fonction des valeurs de μ et de δ. Arratia et Waterman [7] ont étudié ce phénomène, mettant en évidence une transition de phase. Celle-ci dépend du signe de $\rho_{\mu,\delta}$, et est définie dans le cadran $\{\mu \geq 0, \delta \geq 0\}$ par une courbe de type hyperbolique et d'asymptotes parallèles aux axes (voir la figure 6.1). Cette courbe délimite la région linéaire, qui contient l'origine, et la région logarithmique, qui

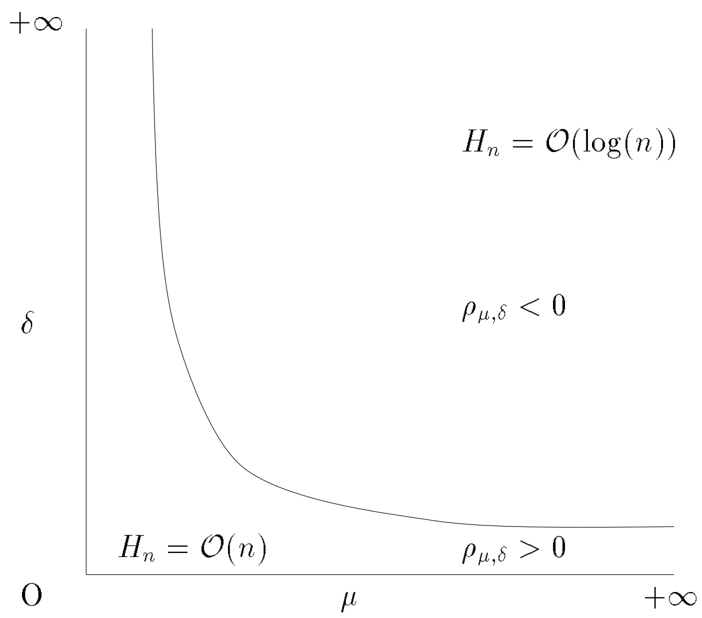


Figure 6.1: Allure générale de la courbe de transition de phase pour l'alignement de séquences.

contient le point $[+\infty, +\infty]$ (voir Waterman [85], Arratia et Waterman [7], ou Waterman et Vingron [84]).

Nous avons développé au chapitre 2 le cas de l'alignement de séquences sans gaps, qui correspond au cas $\delta = \infty$, et à un schéma de score $s(x, y)$ beaucoup plus général que le schéma de scores utilisé précédemment dans ce paragraphe ($s(x, y) = 1$, si $x = y$, et $s(x, y) = -\mu$ si $x \neq y$). Dans ce cas plus général, sous la seule condition que $E(x, y) < 0$, Karlin, Dembo et Kawabata [51] ont montré la loi limite

$$\lim \frac{H_n}{\log_{1/p}(n \times n)} \to 1 \quad \text{avec probabilité 1.}$$

Dans cette équation, p est défini par $\log(1/p) = \lambda$, ($p = e^{-\lambda} \in [0, 1]$), et λ est la racine positive de l'équation

$$E(e^{ts(X,Y)}) = 1.$$

L'équation définie par Karlin et Altschul pour calibrer statistiquement l'alignement de deux séquences $\mathbf{X}$ et $\mathbf{Y}$ de longueur respectives n et m s'écrit (voir le paragraphe 2.3 et l'exemple 49),

$$\Pr\Big(\mathbf{H}(\mathbf{X}, \mathbf{Y}) > \log_{1/p}(nm) + x\Big) = 1 - e^{-Kp^x}. \tag{6.40}$$

Cette équation peut se réécrire d'une manière qui met plus en évidence l'approximation de Poisson effectuée, en posant

$$t = \log_{1/p}(nm) + x,$$

qui donne $p^x = mnp^t$. L'équation 6.40 devient alors

$$\Pr(\mathbf{H}(\mathbf{X}, \mathbf{Y}) > t) \approx 1 - e^{-Knmp^t}. \tag{6.41}$$

Comme nous l'avons mentionné précédemment, les équations 6.40 et 6.41 ont une grande importance quand on effectue les recherches de similarité d'une séquence requête et de séquences d'une base de données. La calibration probabiliste des scores tient compte des longueurs des séquences cibles de la base de données, qui peuvent être très variables d'une séquence à l'autre, ce qui induit une variabilité importante des scores des alignements optimaux.

On peut noter à ce point que, si le modèle d'alignements sans gaps avec des systèmes de scores généraux a été analysé complètement, il n'en est rien des modèles d'alignements avec gaps, dans le cadre d'utilisation de matrices de similarités quelconques. Nous montrons au prochain paragraphe comment Waterman et Vingron ont utilisé l'approximation de Poisson et l'heuristique de clumping-declumping d'Aldous pour l'étude de ces alignements avec gaps. Nous utiliserons cette même méthode pour calibrer l'algorithme BlastMultAl pour les multi-alignements de ProDom, avec le même gain considérable en temps pour les calibrations probabilistes que celui obtenu par Waterman et Vingron.

6.4.2 Du modèle d'alignements sans gaps au modèle avec gaps

Waterman et Vingron [86] montrent que la formule 6.41 est une bonne approximation pour toute la région logarithmique, sans qu'il soit possible d'en donner une démonstration rigoureuse. Ils utilisent le modèle heuristique de clumping-declumping d'Aldous, qui prend en compte à la fois les alignements optimaux et sous-optimaux. Cette prise en compte des alignements sous-optimaux permet d'évaluer les paramètres K et p de l'équation 6.41 très rapidement, alors que la méthode de simulation directe qui ne prend en compte que les alignements optimaux est beaucoup plus lente.

6.4.2.1 Le modèle

L'heuristique de clumping d'Aldous suppose que les clumps d'alignements de score élevé sont placés par un processus de Poisson, et que la taille des clumps est définie par une deuxième variable aléatoire. Si l'on considère la valeur moyenne c du meilleur score, et une valeur test $t = c + x$, le nombre de clumps de score supérieur à t aura une distribution de Poisson de moyenne $\chi_{n,m}(t)$. On peut écrire ceci sous la forme

$$\begin{aligned}\Pr(\text{au moins un score dépasse } t) &= 1 - \Pr(\text{aucun score ne dépasse } t)\\ &= 1 - e^{-\chi_{m,n}(t)} = 1 - e^{-\chi}\end{aligned}$$

Notons $W(t)$ le nombre de clumps d'alignements de score supérieur ou égal à t. Waterman et Vingron calculent $W(t)$ en utilisant l'algorithme de declumping décrit au paragraphe 6.4.1.2. Ils itèrent le declumping jusqu'à ce qu'ils obtiennent un i-ème alignement sous-optimal de score $H_{(i)} < t$. La supposition que la formule 6.41 est valide dans toute la région logarithmique repose sur les hypothèses suivantes.

(H1) $W(t)$ est approximativement distribué comme une Poisson de paramètre $\chi_{n,m}(t) = E(W(t))$.
(H2) $E(W(t)) = \widehat{K}p^t$.
(H3) $\widehat{K} = Kmn$.

Ces trois hypothèses sont suffisantes pour approcher la pertinence des alignements avec gaps.

- L'hypothèse (H1) est utilisée pour estimer la pertinence des alignements optimaux et sous-optimaux.

- L'hypothèse (H2) permet d'effectuer une interpolation pour la moyenne de la Poisson $E(W(t))$, pour des valeurs élevées de t, dans des régions où les simulations donnent un petit nombre d'alignements, insuffisant pour une bonne estimation de cette moyenne.

- Les hypothèses (H1) et (H2) regroupées signifient que la distribution empirique des scores des alignements optimaux et sous-optimaux est bien approchée par $e^{-\chi} = e^{-\widehat{K}p^t}$.

- L'hypothèse (H3) dit que le paramètre $\widehat{K}$ est une fonction linéaire des longueurs n et m. Elle permet de normaliser les scores en connaissant seulement les valeurs de K et de p.

Waterman et Vingron [86] vérifient numériquement sur des séquences simulées que les hypothèses (H1), (H2) et (H3) sont bien vérifiées, ce qui valide le modèle utilisé. Comme nous l'avons mentionné précédemment, il n'existe pas de théorie mathématique parfaitement rigoureuse pour ce modèle.

6.4.2.2 Évaluation des paramètres

L'intérêt de la méthode est évidemment, une fois calculés par simulation (utilisation de séquences simulées) les paramètres p et K pour un système de score et des pénalités de gaps (ouverture et extension), d'utiliser ces paramètres pour calibrer des alignements de séquences "de la vie réelle", c'est à dire des séquences prises dans des bases de données.

Nous comparons les deux moyens d'effectuer ces simulations, l'estimation directe et l'estimation par declumping.

Estimation directe.
C'est le moyen est le plus simple. Il s'agit d'utiliser l'équation

$$y(t) = \Pr(H_n < t) \approx e^{-Kmnp^t}. \tag{6.42}$$

Cette équation s'applique aux alignements optimaux (de score H_n). Chaque fois que l'on aligne deux séquences simulées, on considère l'alignement optimal et on effectue une statistique des scores correspondant. Après avoir effectué un nombre suffisant de simulations (en pratique de l'ordre de 1000 à 10000 comparaisons), on obtient les données $y(t)$ en fonction de t. L'équation 6.42 peut s'écrire

$$z(t) = \log\{-\log(y(t))\} \approx \log(Kmn) + t\log p = \alpha + \beta t.$$

On peut alors faire une régression linéaire de $z(t)$ en fonction de t, qui nous donne les valeurs α et β. Les valeurs m et n sont les longueurs des séquences simulées et donc connues. On obtient donc $K = e^{\alpha}/mn$, et $p = e^{\beta}$. Cette méthode directe a l'inconvénient d'être très coûteuse en temps.

Estimation par declumping.
C'est avec cette méthode d'estimation qu'apparaît la puissance de l'approche utilisant l'heuristique de declumping et l'approximation de Poisson. Au lieu de considérer les alignements optimaux seulement, Waterman et Vingron considèrent à la fois les alignements optimaux et sous-optimaux au-dessus du score seuil variable t. L'observation fondamentale pour cette approche est que la moyenne χ de la Poisson vers laquelle le modèle converge peut être estimée par estimation du nombre d'alignements optimaux et sous-optimaux de scores supérieurs à t. L'algorithme de declumping (voir l'équation 6.38) permet de calculer itérativement les alignements sous-optimaux

$$H_{(1)} \geq H_{(2)} \geq \cdots \geq H_{(N)}.$$

Pour chaque valeur seuil t on a alors un indice $u(t)$ tel que

$$H_{(i)} > t,\ \forall i \leq u(t) \quad \text{et} \quad H_{u(t)+1} \leq t.$$

Les hypothèses (H2) et (H3) du précédant paragraphe nous donnent alors

$$E(W(t)) = E(u(t)) = Kmnp^t. \tag{6.43}$$

L'intérêt de cette approche tient dans le fait que, en comptant les alignements sous-optimaux, un petit nombre de comparaisons suffit pour avoir assez de données pour évaluer les paramètres p et K. Avec des séquences simulées de taille 900, Waterman et Vingron se contentent de la comparaison de 10 paires de séquences avec la méthode de declumping, pour 1000 comparaisons avec la méthode directe. La méthode de declumping, dans ce cas précis, leur fournit environ 300 alignements sous-optimaux pour chaque paire. Le calcul des paramètres p et K s'effectue alors par régression linéaire, d'une manière identique à celle utilisée dans le cas de l'estimation directe. On voit que Waterman et Vingron ont gagné un facteur 100 dans le nombre de comparaisons effectuées en utilisant la méthode de declumping. Nous allons montrer dans le prochain paragraphe comment nous avons utilisé cette même méthode pour calibrer les multi-alignements de la base ProDom relativement à l'algorithme BlastMultAl.

6.5 La calibration des multi-alignements de ProDom pour l'algorithme BlastMultAl

Nous allons à présent appliquer la méthode de declumping et l'approximation de Poisson à la base de multi-alignements protéiques ProDom. Nous avons donné au paragraphe 5.4.2.1

une description de ProDom28. Les bases d'indice supérieurs sont construites d'une manière globalement identique, à partir de la base SWISSPROT ayant l'indice correspondant, avec cette différence que les gaps sont pris en compte dans les dernières versions de ProDom, et en particulier dans la version ProDom33 sur laquelle nous avons travaillé.

Les alignements multiples de ProDom sont très inhomogènes en terme de longueur et de nombre de séquences contenues, et les graphes d'alignement que nous construisons sur ces alignements multiples (voir le paragraphe 5.2) sont d'allures très différentes. Nous avons donc besoin d'une calibration probabiliste propre à chaque multi-alignement. Cette calibration probabiliste a été effectuée pour ProDom28 par simulation directe, avec l'aide de Jean-Jacques Codani. Nous avons lancé pendant les périodes de faible utilisation des machines ces simulations directes sur un réseau d'une cinquantaine de machines. Ces simulations, effectuées sur des machines de puissance assez variables, ont généré une charge d'environ 110 jours de temps CPU. L'utilisation du declumping va réduire considérablement cette charge de simulation.

Nous décrivons dans ce paragraphe la définition des clumps pour BlastMultAl et la validation du modèle d'approximations Poissonniennes pour deux exemples caractéristiques choisis dans ProDom33, les multi-alignements 437 et 14669.

- Le multi-alignement 437 contient 34 séquences et a une longueur de 570 positions. Dans le graphe d'alignement obtenu par preprocessing, 53% des positions sont dans des régions de branches, et il y a 37 régions de branches. Les régions de branches ont une longueur moyenne de 5 positions. Ceci correspond dans le tableau de la page 84 à une espérance positive, mais, comme nous l'avons déjà mentionné, ce tableau correspond à des séquences indépendantes les unes des autres, et il est donc très pessimiste; le modèle de Karlin reste donc applicable.

- Le multi-alignement 14669 est un multi-alignement "dégénéré", puisqu'il ne contient qu'une séquence. Il a une longueur de 373 positions. Ce multi-alignement nous permet de nous placer dans des conditions voisines de celles testées par Waterman et Vingron.

Nous avons volontairement choisi deux multi-alignements de tailles voisines, en termes de nombre de positions, pour que les effets de bord dans l'exécution de l'algorithme d'alignement soient similaires, et de plus de dimensions suffisantes pour que ces effets de bord soient faibles.

6.5.1 Les clumps pour BlastMultAl

La manière avec laquelle BlastMultAl traite les gaps est pour l'instant une manière simplifiée. Les gaps d'un multi-alignement sont considérés relativement au consensus qui correspond à une séquence de longueur maximum pour l'alignement. Ceci induit un biais de l'algorithme en faveur des séquences les plus longues du multi-alignement. L'extension des *Hits* est toujours effectuée le long d'une diagonale (voir le paragraphe 4.3.2.1). Quand on parcourt une diagonale avec BlastMultAl, si on passe sur une région de branches, on essaye les différentes branches pour conserver la meilleure. Il est possible que ces branches contiennent des gaps, qui sont considérés par l'algorithme comme un caractère

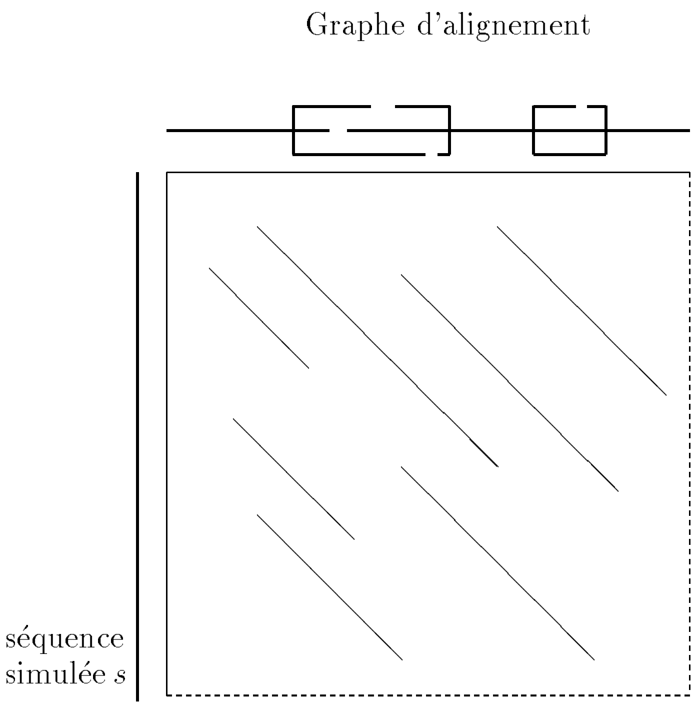


Figure 6.2: Clumps pour BlastMultAl

supplémentaire dans l'alphabet des acides aminés. Nous rappelons les principes algorithmiques décrits au paragraphes 4.3.2 et 5.3.2. De même que Blast, quand BlastMultAl détecte un alignement de score suffisant, toute la diagonale correspondante est examinée. Si à un moment donné, un score positif et suffisant (au-dessus d'un seuil donné) est atteint, l'alignement est étendu tant que le score reste positif; après retour du score à zéro ou à des valeurs négatives, les bornes de l'alignement sont définies, et l'algorithme est réinitialisé à la position atteinte pour la recherche d'un nouvel alignement.

Cet algorithme d'extension entraîne par construction, dans le cas de Blast (correspondant à des multi-alignements contenant une seule séquence), que deux alignements de scores élevés des séquences considérées ne peuvent se couper; dans ce cas, le nombre d'alignements retenus est égal au nombre de clumps.

La situation est différente dans le cas de BlastMultAl. Quand on effectue l'extension le long d'une diagonale, le passage sur les différentes branches (voir figure 6.2) peut conduire à des superpositions d'alignements. On peut définir les clumps sur une diagonale de la manière suivante. On considère les positions sur une diagonale et on affecte un score de marquage, initialisé à zéro, à chaque position. Quand on trouve un alignement de score suffisant (supérieur au seuil minimum S_2), on marque toutes les positions incluses dans l'alignement avec le score de l'alignement, si ce score est supérieur au score de marquage déjà enregistré pour la position. Il peut y avoir ainsi superposition d'alignements, auquel cas le score de marquage d'une position peut prendre successivement plusieurs valeurs positives croissantes. Une sentinelle avec un score à zéro est ajoutée en fin de diagonale. Avec cette construction, le nombre de clumps sur une diagonale sera égal au nombre de retour à zéro du score de marquage lors du parcours de la diagonale. Les clumps de

BlastMultAl, comme ceux de Blast, sont alors constitués de morceaux des diagonales, et le declumping est très simple.

6.5.2 Vérification du modèle

Nous allons vérifier expérimentalement, pour les deux multi-alignements choisis comme échantillon, et pour l'algorithme BlastMultAl, l'adéquation du modèle de declumping au modèle Poissonnien. De même que Waterman et Vingron pour leur approche de l'alignement avec gaps, nous ne pouvons donner de justification théorique rigoureuse de cette vérification. Nous vérifions successivement les hypothèses H1, H2 et H3 du paragraphe 6.4.2.1 pour les deux multi-alignements considérés.

- *Vérification de l'hypothèse H1.*

  Nous effectuons les alignements du multi-alignement considéré avec 5000 séquences simulées. La longueur n des séquences simulées est de $n = 868$ positions pour les séquences simulées dans le cas du multi-alignement 14669 et de $n = 570$ positions dans le cas du multi-alignement 437. Ceci nous donne un produit mn équivalent dans les deux cas, avec m la longueur des multi-alignements considérés. L'hypothèse H1 stipule que la distribution du nombre $W(t)$ de clumps de score supérieur à un score seuil t est Poisson de moyenne $\chi = E(W(t))$.

  1. Multi-alignement 14669. (Figure 6.3). Nous sommes dans un cas simplifié du modèle Waterman-Vingron (alignement de deux séquences sans indels). Nous obtenons des résultats similaires à ceux de Waterman et Vingron [86]. Plus le score seuil est élevé, et meilleure est l'approximation Poissonnienne. Cette approximation est parfaite pour $t = 40$, excellente pour $t = 30$ et encore très bonne pour $t = 20$. Elle est moins bonne pour $t = 13$.
  2. Multi-alignement 437. (Figure 6.4). Nous sommes dans un cas typique d'utilisation de BlastMultAl. Le multi-alignement 437 contient un nombre assez grand (34) de séquences. Les mêmes constatations que dans le cas du multi-alignement 14669 s'imposent. Plus le score seuil t est élevé, et meilleure est l'approximation Poissonnienne. Cette approximation est très bien vérifiée pour $t = 40$, un peu moins bien pour $t = 30$ et sensiblement moins bien pour $t = 20$ et $t = 13$.

  BlastMultAl fournit un nombre nettement plus grand d'alignements potentiels avec les séquences simulées pour le multi-alignement 437 (graphe d'alignement), que le multi-alignement 14669 (une seule séquence). On a constaté avec le multi-alignement 14669 que la qualité de l'approximation Poissonnienne diminuait quand le nombre de clumps augmentait (seuil décroissant). On peut donc s'attendre, pour des seuils identiques, à ce que l'approximation Poissonnienne soit moins bien vérifiée, à niveau de seuil t identique, par BlastMultAl pour le multi-alignement 437 que pour le multi-alignement 14669; c'est ce qu'on observe effectivement. On peut faire également la remarque expérimentale suivante. On peut bien approximer les distributions obtenues dans les cas où χ est grand en utilisant des Poissons décalées $P_{\chi,\sigma}$, où l'on a

$$\Pr(X_{P_{\chi,\sigma}} = i) = e^{-(\chi+\sigma)} \frac{(\chi+\sigma)^{i+\sigma}}{(i+\sigma)!}. \tag{6.44}$$

L'espérance et la variance d'une Poisson de paramètre χ décalée de σ sont respectivement χ et $\chi + \sigma$. L'interprétation de ces décalages est néanmoins difficile. On peut aussi comparer le résultat expérimental obtenu pour les deux multi-alignements étudiés et le résultat théorique obtenu avec les suites de piles (exemple 49). Dans ce dernier cas, les équations 6.28 et 6.29 nous donnent

$$||\mathcal{L}(W) - P_\chi|| \leq \left(1 \wedge \frac{1}{\chi}\right) b(n,t,p),$$

avec

$$\chi = (n-t-1)p^t(1-p) + p^t \quad \text{et} \quad b(n,t,p) < (2t+1)[(1-p)p^t]^2\{n - t + \ldots\}.$$

On voit alors que, si p est voisin de 1, ce qui correspond à chercher des événements fréquents, on a $b(n,t,p) = \mathcal{O}(\chi)$, et la qualité de la convergence vers la loi de Poisson n'est plus assurée.

- *Vérification de l'hypothèse H2.*

  Nous vérifions l'hypothèse $E(W(t)) = \widehat{K}p^t$, ou

  $$\log\{E(W(t))\} = \log(\widehat{K} + t\log(p).$$

  Nous calculons à cet effet les alignements des multi-alignements 14669 et 437 et de 5000 séquences. La figure 6.5 donne les valeurs de $\log\{E(W(t))\}$ en fonction de t pour ces simulations. L'hypothèse de linéarité est très bien vérifiée pour les deux multi-alignements. Cette hypothèse est particulièrement importante, car c'est elle qui permet de calculer par régression linéaire, pour chaque alignement, le paramètre λ intervenant de manière exponentielle dans la formule de Karlin (équation 2.30).

- *Vérification de l'hypothèse H3.*

  Nous vérifions l'hypothèse $\widehat{K}/m = Kn$. Nous effectuons d'une part pour le multi-alignement 14669, et d'autre part pour le multi-alignement 437, les alignements de ces multi-alignements avec 60 séquences simulées. Nous effectuons cette opération avec des séquences simulées de longueur n, et nous calculons la valeur de $\widehat{K}/m = Kn$ au moyen de l'équation 6.43 que nous écrivons sous la forme

  $$\log\{E(W(t)\} = \log(Kmn) + t\log p = \alpha(n) + \beta t.$$

  Nous calculons $\alpha(n)$ par régression linéaire de $\log\{E(W(t)\}$ par rapport à t. Nous répétons cette opération pour les valeurs de n allant de 200 à 2000 par pas de 200. La figure 6.6 donne les résultats de ces tests. La croissance linéaire est bien vérifiée pour les deux multi-alignements; elle l'est très bien pour le multi-alignement 437.

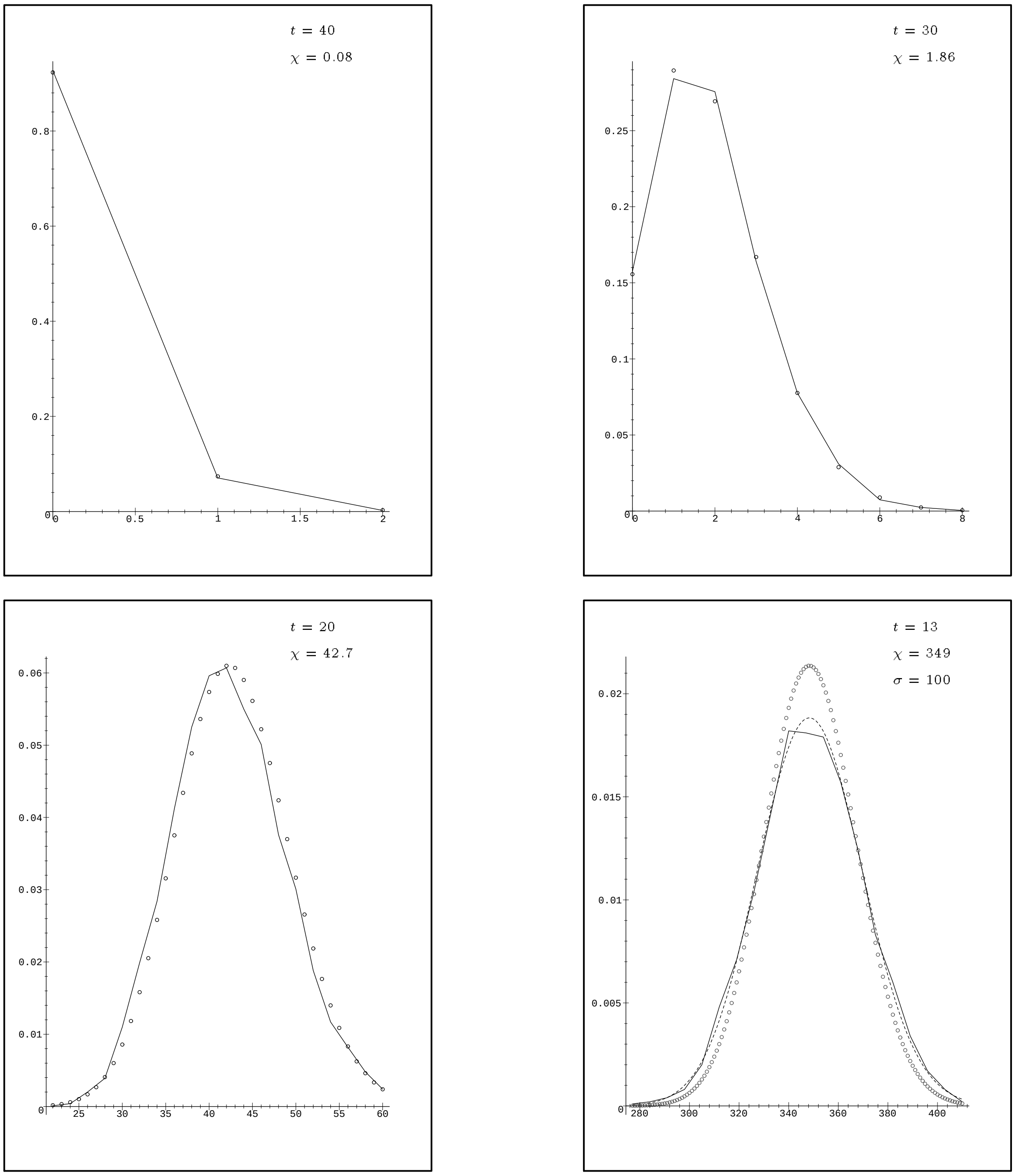


Figure 6.3: Comparaison du modèle de clumps avec un modèle Poissonnien pour le multi-alignement 14669. Le nombre de séquences simulées est 5000. Les courbes en trait pleins donnent la distribution du nombre de clumps au dessus du score t. Les cercles correspondent à une Poisson P_χ de paramètre $\chi = E(W(t))$. La courbe en pointillé correspond à une Poisson décalée de σ , $P_{\chi,\sigma}(i) = e^{-(\chi+\sigma)}(\chi+\sigma)^{i+\sigma}/(i+\sigma)!$.

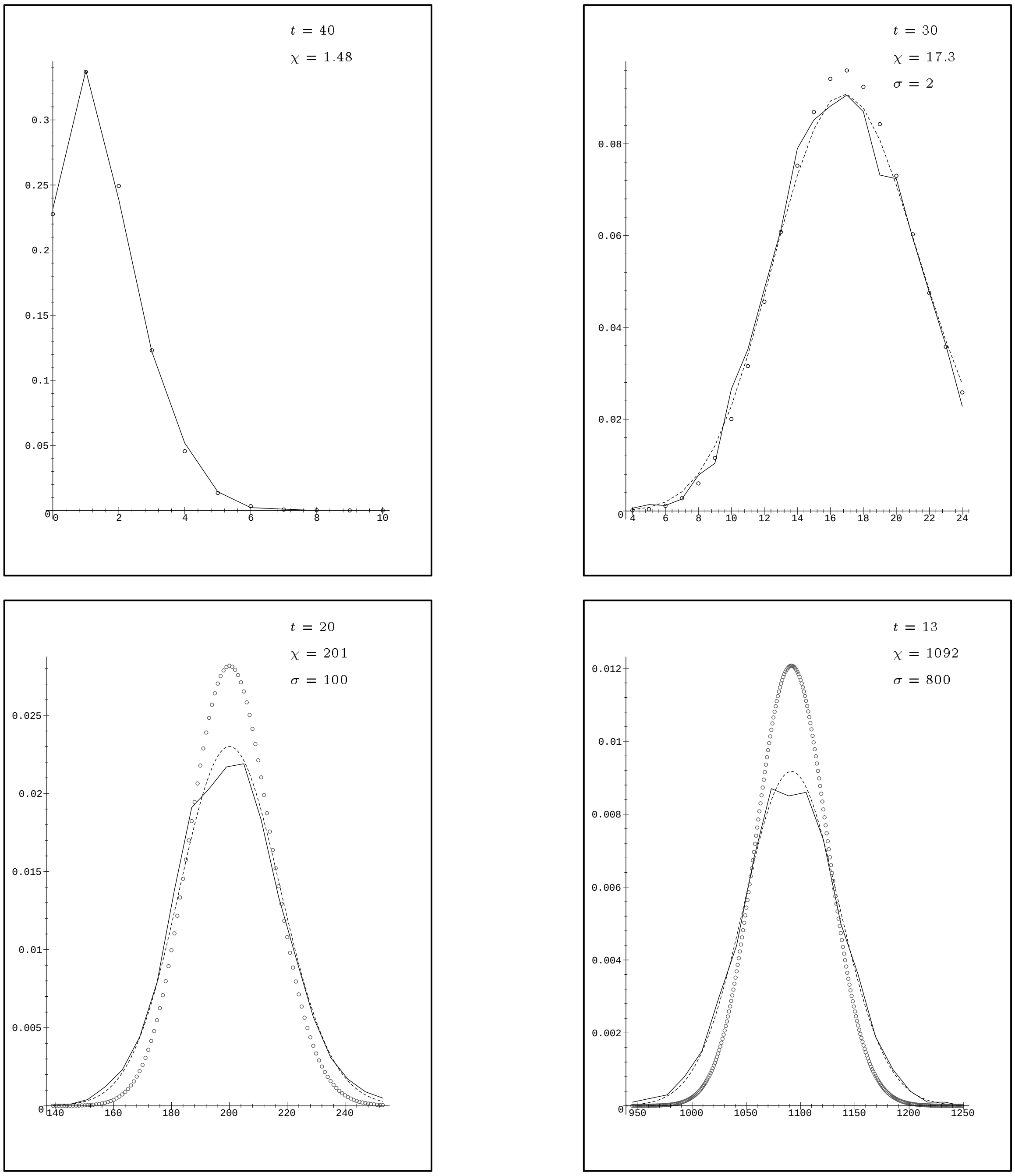
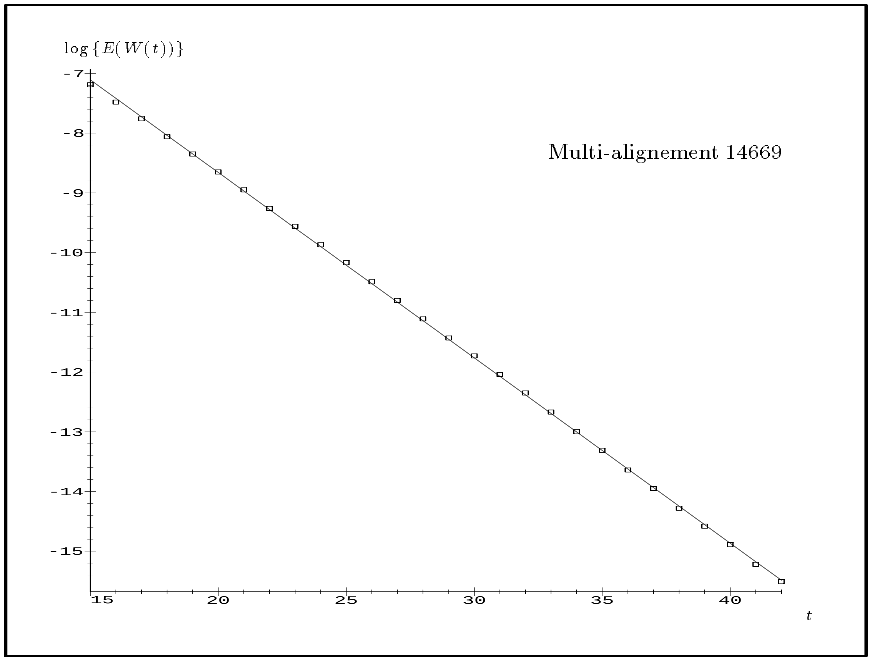


Figure 6.4: Comparaison du modèle de clumps avec un modèle Poissonnien pour le multi-alignement 437. Le nombre de séquences simulées est 5000. Les courbes en trait pleins donnent la distribution du nombre de clumps au dessus du score t. Les cercles correspondent à une Poisson P_χ de paramètre $\chi = E(W(t))$. La courbe en pointillé correspond à une Poisson décalée de σ, $P_{\chi,\sigma}(i) = e^{-(\chi+\sigma)}(\chi+\sigma)^{i+\sigma}/(i+\sigma)!$.



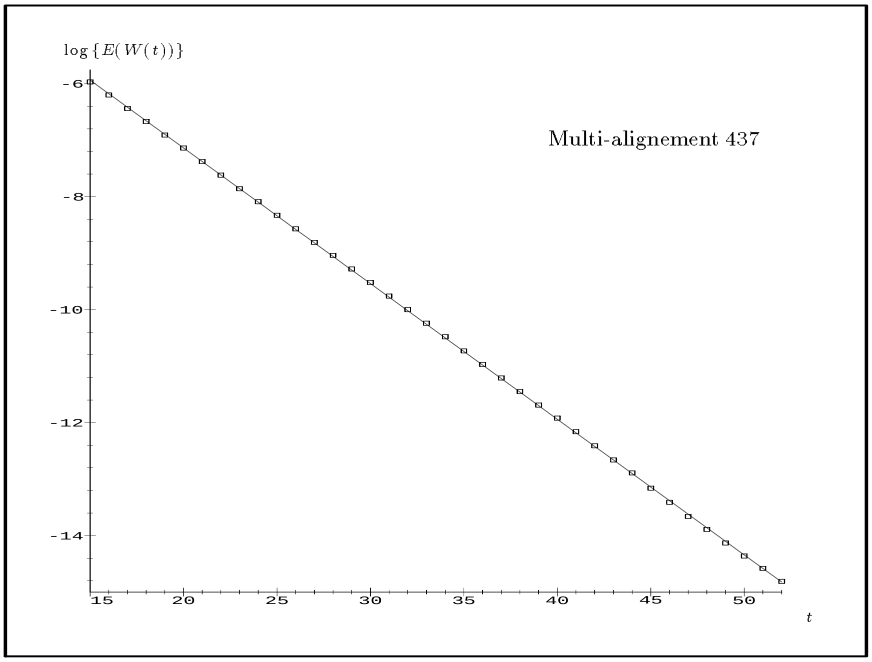
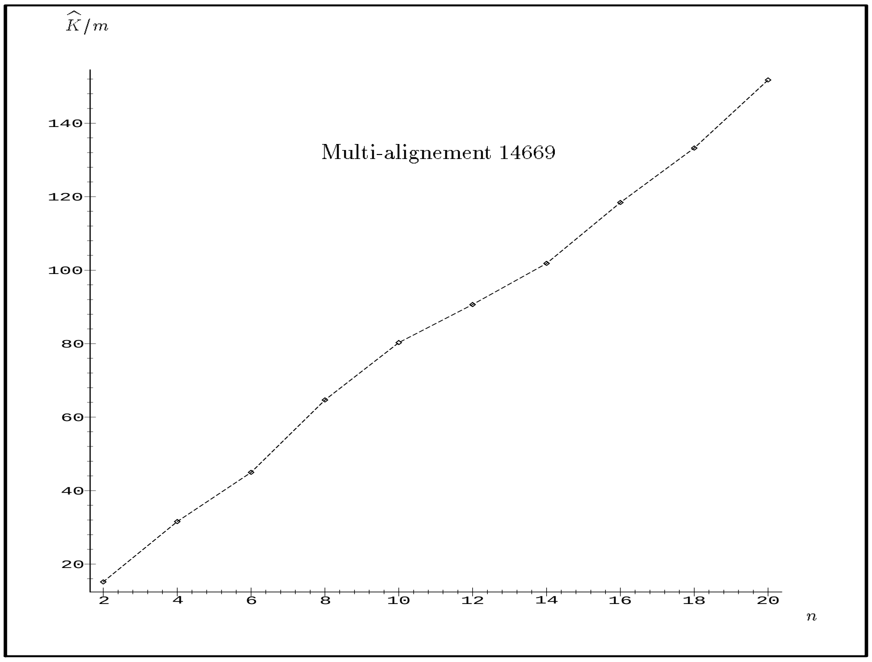


Figure 6.5: Vérification de l'hypothèse H2 (linéarité de $\chi = E(W(t))$ en fonction de t.)



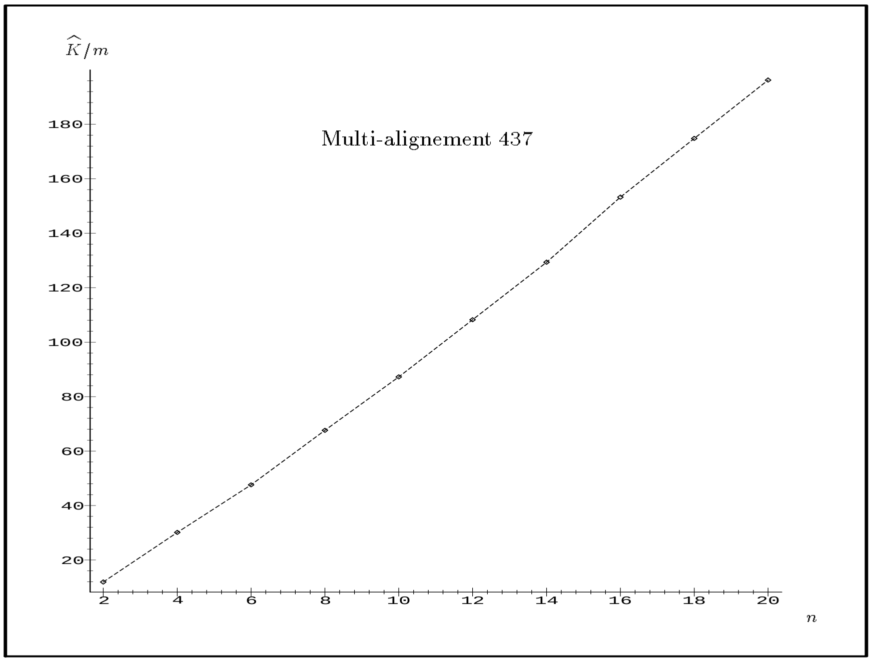


Figure 6.6: Vérification de l'hypothèse H3 (linéarité de $\widehat{K}/m$ en fonction de n, longueur des séquences simulées). L'unité d'échelle pour n est la centaine de positions.

Les tests effectués montrent que les hypothèses H1, H2 et H3 sont suffisamment bien vérifiées pour que l'on puisse appliquer le modèle de declumping avec approximations de Poisson à la recherche d'alignements effectuée avec BlastMultAl sur les multi-alignements de ProDom.

6.5.3 Calibration des multi-alignements de ProDom33

Nous pouvons à présent appliquer les résultats précédents pour la calibration de ProDom33. Nous sommes cependant confrontés à nouveau à une difficulté inhérente aux bases de séquences ou de multi-alignements, la très grande diversité de dimensions des éléments de ces bases. Les multi-alignements de ProDom33 peuvent contenir de plus de 600 séquences pour les plus gros multi-alignements, comme ils peuvent n'en contenir qu'une (multi-alignements dégénérés). Les longueur des multi-alignements peuvent varier entre 20 positions et 3500 positions. L'un des grands intérêts des formules de Karlin est de permettre de comparer les alignements obtenus avec des séquences de longueur différentes, le logarithme des longueurs de ces séquences intervenant dans le calcul de la pertinence probabiliste de ces alignements. Cependant, la formule de Karlin est un résultat asymptotique qui ne peut qu'être approché dans des conditions d'application à des bases de données. Des difficultés semblables surgissent quand nous voulons appliquer la méthode de declumping pour calibrer des multi-alignements de ProDom.

Le théorème 15 nous dit qu'asymptotiquement, un alignement de score t a pour longueur t/ω, où $\omega = E(Xe^{\lambda X})$, X étant la variable construite avec une matrice de score sur une distribution donnée de l'alphabet. Numériquement, avec BLOSUM62 comme matrice de score, et la distribution de SWISSPROT pour l'alphabet des protéines, on a $\omega \approx 1.14$. Nous avons vu que la qualité de l'approximation Poissonnienne était d'autant plus grande que le score t était grand. Cependant, en considérant que $L(t) \approx t/\omega$, en moyenne, avec $L(t)$ la longueur des alignements de score t, on voit qu'avec un score $t = 40$, on a $L(40) \approx 35$, et on aura un résultat fortement biaisé pour la calibration des multi-alignements courts par rapport à la calibration des multi-alignements longs. Compte tenu de l'excellente vérification de l'hypothèse H2, nous avons choisi un seuil très bas pour la calibration, $t = 13$. Ceci nous permet de calibrer tous les multi-alignements de ProDom en limitant le biais lié à la longueur des multi-alignements, et en acceptant que l'hypothèse H1 soit moins bien vérifiée. Une solution alternative consisterait à prendre un seuil élevé (par exemple $t = 40$), et à ne calibrer que les multi-alignements assez longs (par exemple de longueur supérieure à 100 positions), puis à prendre comme valeur λ pour tous les multi-alignements non calibrés la moyenne des valeurs λ des multi-alignements calibrés. Ce traitement des multi-alignements courts est un point délicat que de futures recherches devraient améliorer.

Comparativement aux 110 jours de simulation directe que nous avons mentionnés au début de ce paragraphe, la calibration des multi-alignements de ProDom33 par declumping se fait en 3 jours de temps CPU, sur une DEC ALPHA. Ceci permet d'intégrer la calibration probabiliste des multi-alignements dans la chaîne de production de ProDom, et de suivre ainsi la génération des versions successives de cette base.

Chapitre 7

Un modèle mathématique et informatique pour le génotypage par PCR multiplex

7.1 Introduction

Nous considérons dans ce chapitre entièrement original une manipulation familière aux biologistes, la Réaction de Polymérisation en Chaîne, en abrégé PCR, de l'anglais Polymerase Chain Reaction. Des opérations de recherche comme des diagnostics de routines mettent souvent en jeu simultanément plusieurs de ces réactions, ce que nous dénommerons multiplexage. Ces multiplexages sont des manipulations biologiques délicates qui échouent assez souvent, mais ils ont le gros avantage de permettre de sérieuses économies dans les manipulations effectuées, ainsi que des gains de temps. Nous développerons et étudierons un modèle mathématique et un algorithme pour optimiser le choix des amorces, qui sont des composants des manipulations effectuées, avec l'objectif d'amplifier le plus grand nombre de sites d'un ADN, en effectuant le minimum d'opérations de multiplexage.

Introduite au milieu des années 1980, la Réaction de Polymérisation en Chaîne est devenue un outil d'utilisation quasi-routinière pour le biologiste. Par sa capacité à amplifier des millions des segments d'ADN, elle peut être considérée comme un puissant microscope biologique. Elle est appliquée à la surveillance des thérapies du cancer, à la détection d'infections bactériennes, à la prédiction du sexe de cellules pré-natales, et à de nombreuses autres applications. Elle permet des gains de temps considérables par rapport à d'autres méthodes. En raison de sa sensibilité, (même l'ADN d'une seule cellule peut être amplifiée), la méthode PCR est particulièrement adaptée à la détection de séquences très faiblement représentées dans un échantillon biologique (particule virale, etc.). Des machines [65] ont été construites, qui permettent des amplifications *in vitro* d'ADN au moyen de PCR. La contrepartie de ces progrès est le coût relativement élevé d'une amplification. De même, l'amplification de chaque fragment cible d'ADN demande une expérience de PCR séparée, avec les manipulations correspondantes, et l'immobilisation d'un automate. Pour ces raisons, la possibilité d'effectuer des amplifications PCR multiplex (ou amplifications PCR simultanées) augmente beaucoup le potentiel de la méthode, en divisant les coûts d'expérimentation. Nous présentons au paragraphe 7.2 la méthode PCR, puis au paragraphe 7.3 les conditions permettant le multiplexage. Nous décrivons ensuite au paragraphe 7.4 une modélisation du problème de multiplexage par un modèle d'urnes, puis au paragraphe 7.5 un algorithme découlant de ce modèle, et au paragraphe 7.6 les résultats expérimentaux obtenus. Au paragraphe 7.7, nous montrons que le problème du multiplexage est *NP*-complet, et nous donnons une analyse probabiliste de ce problème au paragraphe 7.8.

7.2 La réaction de polymérisation en chaîne

7.2.1 Qu'est ce que la PCR?

Nous décrivons un peu plus en détail la PCR, renvoyant au livre de J. D. Watson *et al.*, *Recombinant DNA* [87] pour une introduction détaillée à ce sujet.

Nous rappelons que l'ADN est constitué de quatre nucléotides ou bases, l'adénine, la cytosine, la guanine, et la thymine, représentés en raccourci respectivement par les lettres A, C, G et T. La structure bien connue de l'ADN en double hélice est construite sur des appariements des nucléotides, A $\Leftrightarrow$ T, C $\Leftrightarrow$ G, correspondant à une parfaite complémentarité des deux brins d'ADN.

Un gène est une portion d'ADN ayant des capacités fonctionnelles de haut niveau. Les gènes peuvent s'étendre sur des longueurs de plusieurs centaines de milliers de bases. Un problème majeur dans l'analyse de gènes est leur nombre qui, dans les génomes complexes tels que ceux des mammifères, peut atteindre 100.000.

Pour comprendre le progrès apporté par la PCR, dans l'analyse d'un grand nombre de gènes, il est intéressant de considérer la méthode qui existait avant l'introduction de la PCR, le clonage par plasmides. Un plasmide est une sorte de mini-chromosome, qui possède, comme le chromosome, des capacités élevées d'auto-réplication à l'intérieur d'une cellule. Cloner un ADN consiste à laisser se recombiner un morceau d'ADN auquel on s'intéresse avec l'ADN du plasmide, et à laisser ce dernier se répliquer. Cette méthode a deux inconvénients. D'une part, la vitesse de réplication des plasmides reste relativement faible. D'autre part, les recombinaisons ne peuvent se faire qu'autour de sites de coupure de l'ADN déterminés par des enzymes de restriction. Une enzyme de restriction typique comme *EcoRI* n'a que six bases de longueur. À une enzyme correspond donc en général un grand nombre de sites de coupures, ce qui complique l'analyse. Le clonage par plasmides est donc un procédé qui manque parfois de précision.

La PCR, au contraire, va permettre de produire un nombre énorme de copies d'un site parfaitement ciblé, sans que l'on ait besoin d'effectuer des clonages. La PCR exploite certaines caractéristiques de la réplication de l'ADN. L'ADN simple brin est utilisé comme matrice pour la synthèse d'un nouveau brin complémentaire. Ces brins simples matriciels sont produits facilement par simple chauffage de l'ADN double-brin à une température voisine de l'ébullition. Il suffit d'une petite section d'ADN double-brin pour initialiser ("amorcer") la synthèse (figure 7.1).

Le point de départ de la synthèse d'ADN peut être spécifié en fournissant une amorce nucléotidique qui vient s'apparier avec le brin matriciel à ce point. Les deux brins d'ADN peuvent servir de brin matriciel, à condition qu'une amorce soit fournie pour chaque brin.

Un cycle de PCR, comme décrit par la figure 7.1, s'effectue comme suit.

On sépare d'abord les deux brins par chauffage. On baisse ensuite la température pour permettre aux amorces de s'apparier aux séquences complémentaires dans les molécules d'ADN. On laisse ensuite le processus de synthèse se poursuivre, dans les directions 5' $\Rightarrow$ 3'. À la fin d'un cycle, pratiquement 100% des copies du fragment d'ADN compris entre les positions des amorces ont été dupliqué. Le tableau suivant montre la vitesse de réplication du fragment sous amplification.

cycle	1	1 segment	=>	2	segments
	2	2 segments	=>	4	segments
	3	4 segments	=>	8	segments
...		...			...
...	10	...	=>	1024	segments
...		...			...
...	20	...	=>	1048576	segments
...		...			...
cycle	n	2^{n-1} segments	=>	2^n	segments

La longueur des amorces, une pour chaque brin d'ADN, fait environ 20 nucléotides. Si on suppose une distribution parfaitement aléatoire des bases à chaque position, la probabilité de trouver une amorce à une position quelconque d'un génome est $1/4^{20} = O(10^{-24})$, une valeur à comparer à la taille des génomes (environ 6.10^9 pour le génome humain). Ceci montre que les amorces définissent avec une très grande précision la portion d'ADN à amplifier.

7.2.2 Les conditions de la réaction de polymérisation en chaîne

Les amorces ne peuvent pas être choisies à volonté à l'intérieur d'un locus (d'une portion) dans un gène. Elles doivent respecter les conditions permettant une correcte amplification par PCR.

La température d'hybridation à laquelle la polymèrase synthétise les nouveaux brins d'ADN est une de ces conditions. Cette température dépend de la composition des amorces, et plus particulièrement du pourcentage respectif des bases A et T, par rapport aux bases G et C. Une méthode plus précise lie la température d'hybridisation à des mesures expérimentales de l'énergie d'appariement des bases. Quand on choisit une paire d'amorces, il faut donc que les températures d'hybridisation des deux amorces soient voisines.

Une autre condition tient à l'homologie entre les deux amorces, et à l'auto-homologie interne à une amorce. De telles homologies empêchent très souvent une amplification correcte, les amorces s'hybridant l'une l'autre, ou des copies d'une amorce auto-homologue s'hybridant entre elles.

```
(a) 5' ..CTGACACAACTGTGTTCACTAGCAA......AAGGTGAACGTGGATGAAGTTGGTG.. 3'
            --------------------
       3' ..GACTGTGTTGACACAAGTGATCGTT......TTCCACTTGCACCTACTTCAACCAC.. 5'
                                          --------------------

                                     ↓ Chaleur

(b) 5' ..CTGACACAACTGTGTTCACTAGCAA......AAGGTGAACGTGGATGAAGTTGGTG.. 3'



       3' ..GACTGTGTTGACACAAGTGATCGTT......TTCCACTTGCACCTACTTCAACCAC.. 5'

                                       ↓

(c) 5' ..CTGACACAACTGTGTTCACTAGCAA......AAGGTGAACGTGGATGAAGTTGGTG.. 3'
                                        3'CCACTTGCACCTACTTCAAC 5'


          5'ACACAACTGTGTTCACTAGC 3'
       3' ..GACTGTGTTGACACAAGTGATCGTT......TTCCACTTGCACCTACTTCAACCAC.. 5'

                                       ↓

(d) 5' ..CTGACACAACTGTGTTCACTAGCAA......AAGGTGAACGTGGATGAAGTTGGTG.. 3'
                                   3'<<-TTCCACTTGCACCTACTTCAAC 5'
                                        amorce arrière
             amorce avant
          5'ACACAACTGTGTTCACTAGCAA->> 3'
       3' ..GACTGTGTTGACACAAGTGATCGTT......TTCCACTTGCACCTACTTCAACCAC.. 5'
```

Figure 7.1: Mécanisme de la PCR. (a) La cible de l'amplification est délimitée (les nucléotides soulignés sont utilisés comme amorces nucléotidiques pour la PCR). (b) Quand on chauffe l'ADN, les brins se séparent. (c) Les amorces nucléotidiques s'hybrident spécifiquement à leur séquences complémentaires, à l'extrémité 3' de chaque brin du fragment cible. (d) L'ADN polymérase utilise ces amorces pour commencer la synthèse des nouveaux brins complémentaires des séquences d'ADN, synthèse qui s'effectue dans la direction 5'-vers-3'. (Exemple tiré de *"Recombinant DNA"*, J. D. Watson, et *al.*)

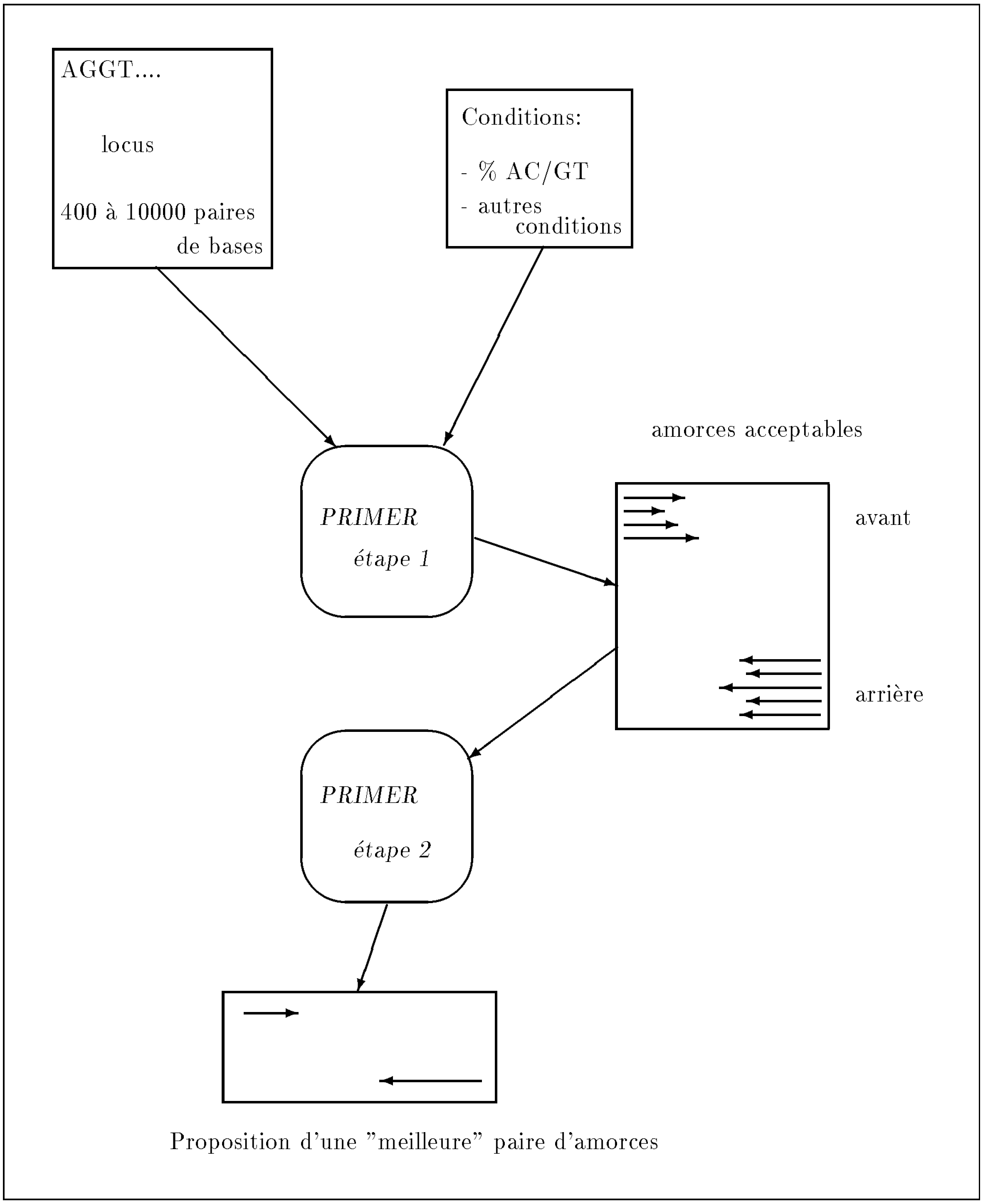


Figure 7.2: Vue d'ensemble du programme *PRIMER*

```
*seq: HUMDYSGP
         GAATTCTAATTCACGTTATTTTCTTTTTTTCCAATTTCTCTCCATGCCTATGTGTTGTCC
         CTTCGCAGCTATAAAGCCATGGCCGATTCATGGGTGCTTTTGTTAAGGCGTTCAGCAGTC
         ACGTTTGTAGATTTTTGAATGGGACTTAGAGCCCTTTTTTGTTCTTTATGTATTTCTCTA
         TTTCTCAGCAAAGGAAATGCAGACATGCAAGAAATAGTGATCAAATGTCCTGTGTACTAT
         TGTGGGTGTCATTAATGGTATAGGGAGAAATAGAAAATAGTTGCAAAGATGCATTTAACA
         AATAAACGAGGTCTTGAGATTCACCATGAATGTGGCCCCTTCTATGAAAAGTAGTTAACA
         TCCAACTGCAAAGTTGTACTGGATCAGTTTGACTTTAACCTTTAGCTAATATGAAAATAT
         GGAATTGTGTGGTGGTGCTCACAAAAAAGAAAATTCATTTTTCTTAATTATCATCAATTA
         ACATGTACTGACTACCCATGAGGGAAAGTTAATTTGCTCTTGAGTGGAACCAGTTATTTG
         CCCTATTATTTCTCCCTTGCTTATTCCCCTCTCCCTCCCTCCTCTCTTTCCATTCAACAA
         AGAAAAATAGATAAAGCAATTTCTGATTAGCCAGTGAAAGCCTCTAACATAAAATTTCCA
         AAGATGTGCCATAAATTATCCACAAAATGTAAAACTTTTCAATTTTGGTTTGCATTTTCT
         TTTTTCTTATTATAAAGGTAATAAGTGCTCATTATAGAATTTGAAAAATATAGGAAGTTG
         CACGGAAGACGAATAAAATCAGCCATAATCCTACAAACCTATTGACACTTGTACATATGT
         TTGTTATCTCTAATGCATTCATTATGATAATGCATCTTTTCAACCAATAGAGTAATCACT
         GGTGACTCAAATTTGTGTTTCTAATTTTTCTTTTTCTTCTTTTTTCCTTTTTGCAAAAAC
         CCAAAATATTTTAGCTCCTACTCAGACTGTTACTCTGGTGACACAACCTGTGGTTACTAA
         GGAAACTGCCATCTCCAAACTAGAAATGCCATCTTCCTTGATGTTGGAGGTACCTGCTCT
         GGCAGATTTCAACCGGGCTTGGACAGAACTTACCGACTGGCTTTCTCTGCTTGATCAAGT
         TATAAAATCACAGAGGGTGATGGTGGGTGACCTTGAGGATATCAACGAGATGATCATCAA
         GCAGAAGGTATGAGAAAAAATGATAAAAGTTGGCAGAAGTTTTTCTTTAAAATGAAGATT
         TTCCACCAATCACTTTACTCTCCTAGACCATTTCCCACCAGTTCTTAGGCAACTGTTTCT
         CTCTCAGCAAACACATTACTCTCACTATTCAGCCTAAGTATAATCAAGGATATAAATTAA
         TGCAAATAACAAAAGTAGCCATACATTAAAAAGGAAATATACAAAAAAAAAAAAAAAAAA
         AAGCAGAAACCTTACAAGAATAGTTGTCTCAGTTAAATTTACTAAACAACCTGGTATTTT
         AAAAATCTATTTTATACCAAATAAGTCACTCAACTGAGCTATTTACATTTAAACTGTTTG
         TTTTGGACTACGCAGCCCAACATATTGCAGAATCAAATATAATAGTCTGGGAATTGATTA
         TTATCCACTCTTCTAAGTTGTCTGTGCCAATTTGCCTTCTCCAATGATAAGGATAATTGA
         AAGAGAGCTATAACTTAAAAAGAGAAAAGTAACAAAACATAAGATATTTAAAATTACCCT
         AGATCTTAAAGTTGGCATTTATGCAATGCCATGTTCAAATGAACATGTTTTTAATACAAA
         TAGTGCATTTTTCAGCCTCAGTGTAATCCATTTGGTAAAATTATGACATCAACTAGAAAC
         ATTAGAATACATTGATGTAAATATGGTTTACCTAGCTAGATCAAATATACTATATATCTT
         TTATATTTGTGAATGATTAAGAAAAATAATGTTGGAATTGTTATACATTAAAGTTTTTTC
         ACTTGTAACAGCTTTCAAGCCTTTCTAAAGAAATACAAAGTTGTGCTGAAGGTATTTAGG
         TATTAAAGTACTACCTTTTGAAAAAACAAGAAGTGAGGCAGACAGAGTAAGGGGAATTTC
         TTTGTAAAATAAACTTCACCAATTCCATAGGAATAAAAGTAATTTGATAGTAAACAACCT
         GCATTTAAAGGCCTTGAGCTTGAATACAGAAGACCTGAATTC
```

taille	amorces avant acceptables	amorces arrière acceptables	nombre de paires d'amorces
2202 bp.	343	339	116277
nb. de paires candidates = $O(20000)$, avec 150 bp $\leq d \leq$ 450 bp d est la distance entre une amorce avant et une amorce arrière dans une paire candidate			

Figure 7.3: Un locus du gène de la dystrophine

- Optimal primer length [20]:
- Minimum primer length [18]:
- Maximum primer length [22]:
- Optimal primer annealing temperature[1] [60.0]:
- Minimum acceptable primer annealing temperature [57.0]:
- Maximum acceptable primer annealing temperature [63.0]:
- Minimum acceptable primer GC% [20]:
- Maximum acceptable primer GC% [80]:
- Salt concentration (mM) [50.0]:
- DNA concentration (nM) [50.0]:
- Maximum number of unknown bases (Ns) allowed in a primer [0]:
- Maximum acceptable primer self-complementarity (number of bases) [12]:
- Maximum acceptable 3' end primer self-complementarity (number of bases) [8]:
- GC clamp how many 3' bases [0]:
- Restriction sites which flank region of interest []:
- Product length ranges [100-150 150-250 250-400]:
- Repeat sequences:

Figure 7.4: Les conditions d'entrée du programme *PRIMER*. Le programme demande à l'utilisateur les valeurs qu'il veut affecter, pour chaque condition. Les valeurs entre crochets sont les valeurs par défaut.

7.2.3 Un logiciel de choix d'amorces: *PRIMER*

Il existe différents logiciels, qui permettent la prédiction des amorces à choisir à l'intérieur d'un locus. Nous avons utilisé le logiciel *PRIMER*, de S.E. Lincoln, M.J. Daly, and E.S. Lander [59] comme logiciel de sélection d'amorces. Ce logiciel parcourt une séquence d'ADN et y choisit des amorces avant et arrières qui respectent les conditions données au paragraphe 7.2.2. Il propose aussi une meilleure paire d'amorces pour l'amplification du locus considéré par PCR. La figure 7.2 donne une vue d'ensemble du programme *PRIMER*.

Le logiciel *PRIMER* s'exécute en deux étapes.

- L'*étape-1* sélectionne des candidats avant et arrière.
- L'*étape-2* choisit une meilleure paire (une amorce avant et une amorce arrière) parmi l'ensemble des paires d'amorces candidates.

MULTIPCR prend comme entrée la sortie de *PRIMER étape-1*, et choisit pour chaque locus une amorce avant et une amorce arrière compatibles avec les amorces déjà choisies pour les autres *loci*, chaque fois que cela est possible.

La figure 7.4 donne les conditions traitées par le programme *PRIMER*. Nous faisons les commentaires suivants sur ce tableau.

Les concentrations en sel et en ADN influent sur le choix des amorces.

Le terme "*annealing temperature*" fait référence à la température d'appariement des amorces et de l'ADN.

Il peut y avoir dans les suites de bases d'un gène, dans une base de données, des bases inconnues, pour des raisons de manipulations imparfaites lors du séquençage. Le programme *PRIMER* peut accepter dans les amorces quelques positions pour lesquelles les bases sont inconnues.

La condition de composition en GC à l'extrémité 3' de l'amorce ("GC clamp...") donne le nombre de bases à cette extrémité qui sont obligatoirement soit G, soit C.

L'utilisateur peut définir une région d'intérêt dans laquelle on recherche les amorces, région plus courte que le gène donné en entrée, en précisant des sites de restriction qui encadreront cette région d'intérêt. Ceci est utile quand le gène donné en entrée au programme est "brut de séquençage".

Le produit des PCR est souvent ensuite soumis à une électrophorèse, manipulation par migration où la longueur des fragments est déterminante. Pour cette raison, *PRIMER* autorise à choisir, pour le fragment qui sera amplifié, parmi différents intervalles de longueur.

Il existe souvent dans les gènes des morceaux de séquences très répétés. À titre d'exemple, il existe dans le génome humain une région très répétée appelée région *Alu*. *PRIMER* s'interdit de choisir des amorces dans de telles régions, car cela perturberait les amplifications.

Nous renvoyons à la documentation *on line* du logiciel *PRIMER* pour une description détaillée des conditions d'entrées et des contrôles effectués sur les amorces candidates pour un locus.

7.3 Le multiplexage de la réaction de polymérisation en chaîne

7.3.1 Objectifs

Comme nous l'avons fait remarquer précédemment, la PCR est une opération relativement onéreuse et coûteuse en temps. Outre l'immobilisation d'une machine, elle multiplie les manipulations, et donc les risques d'erreur humaines. Pour ces raisons, chaque fois que cela est possible, les biologistes souhaitent pouvoir regrouper plusieurs PCR dans une seule manipulation.

Les conditions mentionnées pour l'amplification d'un seul fragment doivent bien entendu rester vérifiées quand on cherche à amplifier simultanément plusieurs fragments.

Le multiplexage de PCR consiste à appliquer la méthode à n loci et n paires d'amorces. Nous voulons alors que les conditions énoncées pour une paire d'amorces, restent vérifiées pour les $\binom{2n}{2}$ combinaisons de deux parmi les $2n$ amorces.

Nous détaillons dans la section 7.3.2 un modèle de compatibilité entre amorces proposé par Gilles Thomas et les conditions correspondantes.

7.3.2 Les conditions pour le multiplexage de PCR

Nous donnons un certain nombre de définitions que nous utiliserons pour la description du problème de multiplexage. Nous parlerons d'amplification d'un *locus* quand nous considérerons l'amplification d'un seul segment d'ADN; une seule amplification est autorisée à l'intérieur d'un locus; à chaque amplification de locus correspond une amorce *avant* et une amorce *arrière*. Nous définissons une *sous-amorce* comme une sous-séquence de longueur σ d'une amorce, et nous considérons dans la suite que σ a la même valeur pour toutes les sous-amorces d'une expérience de multiplexage. Dans les expérimentations pratiques, σ aura les valeurs 4 ou 5. Nous définissons une 3'-*sous-amorce* comme la sous-amorce terminant l'extrémité 3' d'une amorce (les amorces sont toujours lues dans la direction 5' $\Rightarrow$ 3').

Les conditions pour le multiplexage de la PCR sont les suivantes.

1. Conditions d'amplification d'un *locus*.
   (a) La distance entre une amorce avant et une amorce arrière doit être comprise entre 150 et 450 bases (ces valeurs minimum et maximum données en paramètres correspondent au "product range size" pris en entrée par *PRIMER*).
   (b) Les amorces satisfont aux conditions de non-palindromicité qui empêchent l'auto-homologie.
   (c) Les 3'-sous-amorces ne sont complémentaires renversées avec aucune sous-amorce de même longueur (par exemple, A et T d'une part, G et C d'autre part constituant les bases complémentaires, la 3'-sous-amorce GAAT est complémentaire renversée et donc incompatible avec la sous-amorce ATTC).
2. Amplification d'un *Multi-locus*, ou conditions d'une *expérience*.
   (a) Aucune 3'-sous-amorce de locus d'une expérience n'est complémentaire renversée avec une sous-amorce quelconque d'une amorce quelconque de l'expérience; ceci causerait une hybridisation des amorces entre elles. Notons que des sous-amorces, y compris des 3'-sous-amorces peuvent être identiques entre différents loci, ou à l'intérieur d'un même locus.
   (b) Les températures de dénaturation (de séparation des brins) des loci doivent être voisines, ou, condition équivalente, les pourcentages AT/GC dans les amorces d'une PCR multi-locus doivent appartenir à un intervalle limité de valeurs (par exemple $48\% - 52\%$).
   (c) Distance d'électrophorèse. La différence des longueurs entre deux segments amplifiés dans la même PCR multi-locus doit être plus grande qu'une valeur seuil notée δ (bases). Ceci est nécessaire pour pouvoir différencier correctement les différents segments après électrophorèse. Dans le cas de loci non polymorphes, on peut espérer $\delta \leq 5$. Au contraire, dans le cas de loci polymorphes, on a $\delta \geq 50$.

Remarque. L'électrophorèse est une méthode de migration qui permet à des segments courts de se déplacer plus vite que les segments plus longs; cette méthode a néanmoins une précision limitée (paramètre δ).

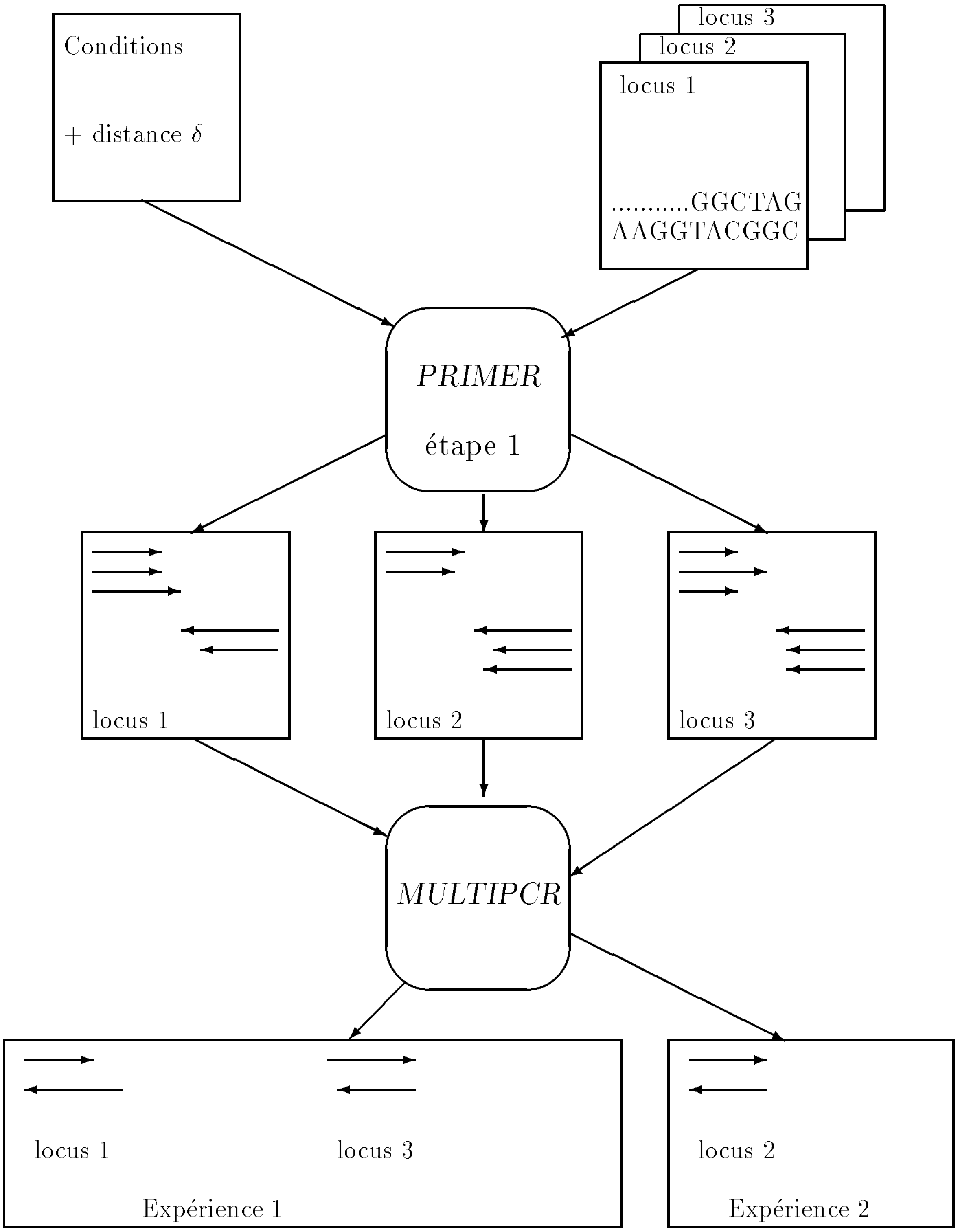


Figure 7.5: Vue d'ensemble du programme *MULTIPCR*

7.4 Un modèle d'urnes pour résoudre le problème de compatibilité entre amorces

Les conditions que nous avons énoncées au paragraphe 7.3.2 peuvent être vues comme un problème de mots et de facteurs dans des mots (séquences et sous-amorces dans la terminologie biologique). Nous faisons au prochain paragraphe un bref détour par la combinatoire des mots, pour en donner un éclairage probabiliste, avant de décrire le modèle d'urnes que nous utiliserons pour résoudre notre problème.

7.4.1 La combinatoire des sous-amorces dans les loci

Des procédés combinatoires standards (voir Flajolet-Sedgewick [74], chap. 7) donnent la série génératrice $S_k(z)$ du nombre de mots ne contenant pas un motif $m_0 m_1 \dots m_{k-1}$, en fonction du polynôme d'auto-corrélation $c(z)$ du motif. Pour un alphabet de quatre lettres, nous avons

$$S_k(z) = \sum s_{n,k} z^n = \frac{c(z)}{z^k + (1-4z)c(z)}. \tag{7.1}$$

Dans cette équation, $s_{n,k}$ est le nombre de mots de taille n ne contenant pas le motif (de longueur k). Le polynôme d'auto-corrélation est défini en fonction du motif, comme $c_k(z) = \sum_{i=0}^{k-1} c_i z^i$ où c_i vaut 1 si le préfixe de longueur $k - i$ du motif et le suffixe de même longueur du motif sont identiques ($a_0 = 1$ pour tous les motifs). Dans le cas le pire, ou toutes les lettres du motif sont identiques, le polynôme d'auto-corrélation vaut $1+z+\cdots+z^{k-1}$. Si ρ_k est le pôle dominant de $S_k(z)$ (la plus petite racine du dénominateur de $S_k(z)$), la probabilité $P_{n,k}$ qu'un mot de taille n ne contienne pas le motif, (sur un alphabet de quatre lettres que nous supposons équidistribuées), vaut

$$P_{n,k} \approx \frac{c_k(1/4)}{\rho_k^n 4^n}. \tag{7.2}$$

Le cas où cette probabilité est la plus faible est le cas où toutes les positions du motif contiennent la même lettre. Dans ce cas, $c_4(z) = 1+z+z^2+z^3$, et $c_5(z) = 1+z+\cdots+z^4$. En utilisant le logiciel Maple, nous trouvons, pour un mot de longueur 2000, correspondant à la taille moyenne de nos *loci*,

$$P_{2000,5} \approx 0.3061, \qquad \text{et} \qquad P_{2000,4} \approx 0.0035.$$

Il y a donc 7 chances sur dix, au moins, de trouver n'importe quelle sous-amorce de longueur 5 dans un *locus* de longueur 2000, et une probabilité presque 1 de trouver n'importe quelle sous-amorce de longueur 4. Ceci conduit à l'idée d'utiliser un modèle d'urnes, un modèle classique pour les techniques de "hachage".

7.4.2 Le modèle d'urnes pour le multiplexage de PCR

Nous donnons ici un exemple constructif de notre modèle, en faisant le codage suivant pour augmenter son efficacité.

- Nous associons à chaque base un nombre entre 0 et 3. Nous choisirons le codage ϕ

$$\mathrm{A} \Rightarrow 0,\ \mathrm{C} \Rightarrow 1,\ \mathrm{G} \Rightarrow 2,\ \mathrm{T} \Rightarrow 3,$$

  pour l'alphabet $\Sigma = \{A, C, G, T\}$.

- Nous effectuons l'application Φ de $\Sigma^\star \Rightarrow \mathbb{N}$ définie par $\Phi(x_n x_{n-1} \dots x_1 x_0) = \sum_{i=0}^{n} \phi(x_i) 4^i$, $(x_i \in \Sigma)$.

  L'application Φ associe à chaque sous-amorce de longueur $\sigma = 4$ un nombre entre 0 et $4^4 = 256$, et à chaque sous-amorce de longueur $\sigma = 5$ un nombre entre 0 et $4^5 = 1024$.

  À titre d'exemple, nous avons

$$\mathrm{TTA} \Rightarrow 330_4 \Rightarrow 60_{10}$$

Nous considérons alors un modèle de 256 urnes, quand $\sigma = 4$, ou un modèle de 1024 urnes, quand $\sigma = 5$. On jette une boule noire dans l'urne i s'il existe une sous-amorce α telle que $\Phi(\alpha) = i$. On jette une boule blanche dans l'urne j s'il existe une 3'-sous-amorce renversée β telle que $\Phi(\beta) = j$. La condition de compatibilité entre sous-amorces et 3'-sous-amorces renversées entraîne alors, pour un modèle donné, qu'aucune urne ne contient simultanément des boules noires et des boules blanches.

La figure 7.6 donne un exemple d'utilisation du modèle d'urnes.

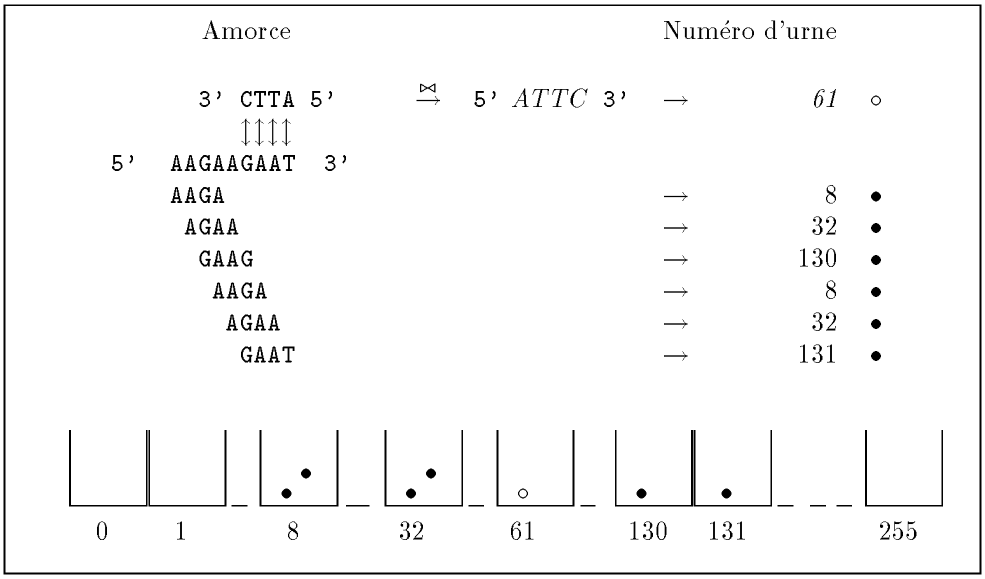


Figure 7.6: Un exemple de modèle de 256 urnes pour contrôler la compatibilité de sous-amorces.

$\mathbf{T_1}$

n loci	
2-mers	
3'-sous-amorces complém. renversées	sous-amorces
0	0
1	0
0	1
0	0
0	0
0	0
0	0
0	1
1	0
0	1
0	1
0	0
0	0
0	0
0	0
0	1
score 2	score 5

OU

$\mathbf{T_2}$

1 locus (2 amorces)	
2-mers	
3'-sous-amorces complém. renversées	sous-amorces
1	0
0	0
0	1
0	0
0	0
0	1
0	0
0	0
1	0
0	0
0	1
0	0
0	0
0	0
0	0
0	1

$\Rightarrow$

$\mathbf{T_3}$

$n+1$ loci (essai)	
2-mers	
3'-sous-amorces complém. renversées	sous-amorces
1	0
1	0
0	1
0	0
0	0
0	1
0	0
0	1
1	0
0	1
0	1
0	0
0	0
0	0
0	0
0	1
score 3	score 6

Figure 7.7: Tables de compatibilité pour la sélection de paires d'amorces ($\sigma = 2$). Le score correspond, pour une colonne, au nombre de bits à 1 dans la colonne.

7.5 Un algorithme dérivant du modèle d'urnes

7.5.1 Principe de l'algorithme

Nous proposons dans ce paragraphe un algorithme d'approximation à grande efficacité, qui se décrit comme suit. Nous trions notre ensemble de loci en ordre croissant, suivant le nombre de paires d'amorces contenues dans chaque locus. Nous traitons notre ensemble ordonné de loci, locus après locus. Pour chaque locus, nous essayons chaque paire d'amorces compatible avec les conditions énoncées, y compris la condition de distance (condition 1(a)).

Pour chaque paire d'amorces, nous jetons les boules blanches et noires correspondantes dans les urnes, selon le modèle décrit ci-dessus. Nous éliminons les paires qui causent des collisions "blanc et noir". Parmi les paires d'amorces acceptables, nous choisissons la paire d'amorces qui minimise, dans l'ordre suivant.

1. Le nombre d'urnes contenant des boules blanches.

2. Le nombre d'urnes contenant des boules noires, chaque fois que le nombre d' "urnes blanches" est identique pour deux paires.

Les boules blanches et noires correspondant aux paires d'amorces déjà sélectionnées *restent dans les urnes* quand on traite un nouveau locus.

Les loci ne possédant pas de paires compatibles avec les paires des loci déjà choisis pour l'expérience en cours sont laissés de coté et traités dans une expérience ultérieure.

7.5.2 Implémentation

Nous remarquons que, dans le modèle décrit au paragraphe précédent, nous n'avons pas besoin de savoir combien de boules blanches ou noires sont présentes dans les urnes. La seule question importante est la question booléenne. "Y a t-il des boules blanches/noires dans l'urne i? " Nous pouvons donc simplifier notre modèle en représentant chaque condition booléenne par un bit, et en utilisant des tables de 4^4 bits, ou de 4^5 bits suivant les longueurs considérées pour les sous-amorces. L'heuristique que nous avons décrit dans le paragraphe précédant se traduit dans cette implémentation en un choix d'amorces qui minimise le nombre de bits à 1 (ou score) dans ces tables. Un "OU" exclusif d'une table $\mathbf{T_1}$ décrivant le système d'amorces déjà acceptées avec une table de travail $\mathbf{T_2}$ correspondant à la paire d'amorces essayées donnera une table $\mathbf{T_3}$ sur laquelle on vérifie que la condition de compatibilité est respectée. On conserve ensuite la paire d'amorces qui minimise les scores de la table $\mathbf{T_3}$.

La figure 7.7 donne un exemple de cette méthode.

7.5.3 Entrées-Sorties

Nous donnons dans le tableau 7.1 un résumé des conditions testées en entrée par *MULTIPCR*, et des sorties de l'algorithme. (Le tableau 7.4 donne les conditions d'entrée du programme *PRIMER*).

Entrées
• Liste des *loci* à amplifier. Les *loci* correspondent à des segments disjoints et indépendants d'ADN. Ils peuvent s'étendre sur différents gènes, comme sur des régions inter-gèniques.
• Intervalles de températures, ou de pourcentages GC/AT.
• Intervalles des longueur des fragments amplifiés; pratiquement, entre 150 et 450 bases.
• Distance d'électrophorèse δ; différence minimum entre les longueurs de deux fragments amplifiés.
• Longueur de contrôle σ (4 ou 5 bases); longueur sur laquelle l'égalité complémentaire inverse d'une 3'-sous-amorce et d'une sous-amorce quelconque est interdite.
Sorties
• La liste des *loci* sélectionnés pour chaque expérience, avec les paires d'amorces correspondantes.
• Pour chaque expérience, un contrôle croisé des incompatibilités au niveau $\sigma - 1$ est affiché.
• La liste des *loci* qui ne peuvent pas être amplifiés est affichée.

Tableau 7.1: Entrées-Sorties pour MULTIPCR

longueur de contrôle σ	distance d'électro-phorèse δ	numéro d'expérience	nombre de loci amplifiés	scores des 3'-sous-amorces	scores des sous-amorces
4	1	1	56	25	230
		2	6	10	137
		3	1	2	26
	2	1	56	27	227
		2	6	12	136
		3	1	2	26
	3	1	51	36	214
		2	10	15	163
		3	2	4	51
	4	1	52	29	222
		2	9	16	140
		3	2	4	55
	5	1	46	35	214
		2	14	20	184
		3	3	6	77
pas d'amplification possible			13		
nombre de loci traités			76		

longueur de contrôle σ	distance d'électro-phorèse δ	numéro d'expérience	nombre de loci amplifiés	scores des 3'-sous-amorces	scores des sous-amorces
5	1	1	61	78	704
		2	3	6	86
	2	1	59	80	694
		2	4	8	123
		3	1	2	29
	3	1	60	85	702
		2	4	8	123
	4	1	60	84	693
		2	4	8	115
	5	1	49	72	693
		2	15	27	344
pas d'amplification possible			12		
nombre de loci traités			76		

Tableau 7.2: Traitement de 76 *loci* (gène de la dystrophine) par MULTIPCR (minimisation des scores des 3'-sous-amorces)

longueur de contrôle σ	distance d'électro-phorèse δ	numéro d'expérience	nombre de loci amplifiés	scores des 3'-sous-amorces	scores des sous-amorces
4	1	1	54	32	219
		2	8	15	157
		3	1	2	26
	2	1	55	31	223
		2	7	12	147
		3	1	2	26
	3	1	51	36	214
		2	10	17	165
		3	2	4	51
	4	1	50	31	221
		2	11	18	159
		3	2	4	55
	5	1	48	28	218
		2	12	20	169
		3	3	6	77
pas d'amplification possible			13		
nombre de loci traités			76		

longueur de contrôle σ	distance d'électro-phorèse δ	numéro d'expérience	nombre de loci amplifiés	scores des 3'-sous-amorces	scores des sous-amorces
5	1	1	61	95	704
		2	3	6	86
	2	1	59	89	669
		2	4	8	123
		3	1	2	29
	3	1	60	91	698
		2	4	8	123
	4	1	60	80	711
		2	4	8	115
	5	1	50	74	694
		2	14	28	321
pas d'amplification possible			12		
nombre de loci traités			76		

Tableau 7.3: Traitement de 76 *loci* (gène de la dystrophine) par MULTIPCR (minimisation des scores des sous-amorces)

longueur de contrôle σ	distance d'électro-phorèse δ	numéro d'expérience	nombre de loci amplifiés	scores des 3'-sous-amorces	scores des sous-amorces
4	1	1	214	34	222
		2	32	27	223
		3	2	4	48
	2	1	133	34	222
		2	113	26	230
		3	2	4	51
	3	1	84	31	224
		2	81	30	226
		3	73	34	222
		4	10	17	168
	4	1	61	33	222
		2	60	30	223
		3	61	32	222
		4	59	31	218
		5	7	10	128
	5	1	48	31	223
		2	45	29	223
		3	49	32	218
		4	49	30	219
		5	47	31	219
		6	10	19	171
pas d'amplification possible			0		
nombre de loci traités			248		

Tableau 7.4: Traitement de 248 loci par *MULTIPCR*, avec contrôle de compatibilité sur 4 bases (modèle of 256 urnes)

longueur de contrôle σ	distance d'électro-phorèse δ	numéro d'expérience	nombre de loci amplifiés	scores des 3'-sous-amorces	scores des sous-amorces
5	1	1	245	134	870
		2	3	4	61
	2	1	127	131	827
		2	121	117	837
	3	1	82	98	775
		2	85	105	786
		3	80	106	790
		4	1	2	29
	4	1	63	96	720
		2	61	86	726
		3	63	90	741
		4	61	81	731
	5	1	51	78	668
		2	49	76	638
		3	50	72	678
		4	50	75	719
		5	48	75	665
pas d'amplification possible			0		
nombre de loci traités			248		

Tableau 7.5: Traitement de 248 loci par *MULTIPCR*, avec contrôle de compatibilité sur 5 bases (modèle of 1024 urnes)

7.6 Résultats expérimentaux

Les résultats décrits dans ce paragraphe ont été obtenus en faisant fonctionner le programme *MULTIPCR* sur des données tirées de la base de données d'ADN "*GENBANK*".

Une première série de calculs a été faite sur un groupe de 76 *loci* contenant 50 *loci* du gène de la dystrophine pris au hasard, responsable de la dystrophie musculaire de Baker et Duchêne (tableaux 7.2 et 7.3). Une deuxième série de calculs a été faite sur 248 *loci* de primates pris au hasard, mais en ne conservant que les *loci* assez long (tableaux 7.4 et 7.5). La taille moyenne des loci pour ces deux derniers tableaux est 2000 pb., avec un nombre moyen de 20000 paires d'amorces admissibles.

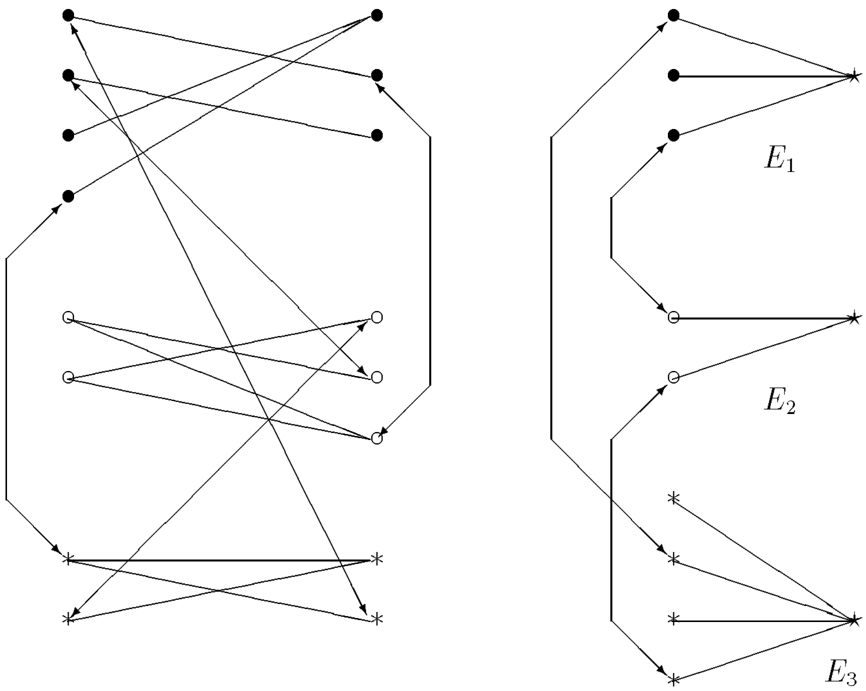
Nous faisons les remarques suivantes sur ces résultats expérimentaux "*in silico*" (obtenus par calcul et sans vérification expérimentale par des manipulations biologiques).

- Dans le cas des amplifications des *loci* de dystrophine, (tableaux 7.2 et 7.3), le nombre de *loci* qui ne peut pas être amplifié, dans le cas où la longueur de contrôle σ vaut 4, est proche de celui obtenu quand la longueur de contrôle σ vaut 5. Ceci est dû au fait que les *loci* sont courts et qu'ils fournissent un nombre assez faible de paires d'amorces candidates.

- Quand on prend une longueur de contrôle σ égale à 4, (système d'urnes de taille $4^4 = 256$), et une distance d'électrophorèse δ égale à 1, (distance qui cause la contrainte la plus faible), si l'on considère l'expérience numéro 1 dans les tableaux 7.4 et 7.5, et si on ajoute, pour chacun de ces tableaux, le score des 3'-sous-amorces, égal au nombres d'urnes remplies par des boules blanches, (resp. 25 et 32), et le score des sous-amorces, égal au nombres d'urnes remplies par des boules noires, (resp. 230 et 219), on constate que le nombre d'urnes pleines, égal à la somme de ces scores (resp. 255 et 251), est presque égal au nombre total d'urnes (256). De plus, on constate expérimentalement, dans le cas du tableau 7.2 ($\sigma = 4$) que le nombre de *loci* amplifiés dans la première expérience décroît lentement (56, 56, 51, 52 et 46), quand la distance d'électrophorèse δ augmente de 1 à 5. On peut effectuer des considérations voisines pour les tableaux correspondants à une longueur de contrôle $\sigma = 5$, mais, dans ce dernier cas, les système d'urnes sont sensiblement moins bien remplis (score total de 704+78=782, pour un système de 1024 urnes dans le tableau 7.2, pour l'expérience 1, avec $\delta = 1$).

- Il n'est pas réellement surprenant de voir dans le tableau 7.5 qu'il est théoriquement possible d'amplifier simultanément 245 *loci*, même si c'est certainement invraisemblable au niveau biologique, compte tenu des techniques connues aujourd'hui. Les *loci* de primates considérés ont été volontairement choisis avec des longueurs comprises entre 4000 et 10000 bases, ce qui donne un nombre élevé de paires candidates. Quand la taille des *loci* augmente, la combinatoire augmente aussi, et on s'attend à pouvoir amplifier simultanément un plus grand nombre de *loci*. Ceci explique les très bons résultats des tableaux 7.4 et 7.5

- On peut faire une autre remarque au sujet des résultats obtenus avec une distance d'électrophorèse δ égale à 4 ou 5, dans le tableau 7.5. Le système d'urnes n'est manifestement pas saturé, et c'est la distance d'électrophorèse qui impose la contrainte la plus forte. Nous rappelons que cette distance δ est la différence minimum de longueur de deux segments amplifiés. Si une paire d'amorces amplifie un fragment de longueur l, on ne peut amplifier dans la même expérience aucun fragment de longueur l', telle que $l - \delta + 1 \leq l' \leq l + \delta - 1$. On peut modéliser simplement la contrainte liée à cette distance d'électrophorèse en considérant un segment de longueur L et en jetant sur ce segment de manière aléatoire et indépendante des segments courts de longueur δ. La distance séparant deux segments courts sera une variable aléatoire discrète équidistribuée de support $[\delta, 2\delta + 1]$, et de moyenne $l = \frac{3\delta - 1}{2}$. Nous avons considéré l'intervalle $[150 - 450]$ pour les longueurs de fragments, ce qui correspond à une longueur $L = 300$. Le nombre attendu $N = L/l$ de *loci* amplifiés simultanément est $N \approx 43$ pour $\delta = 4$, et $N \approx 54$ pour $\delta = 5$. Ces nombres sous-estiment un peu les résultats donnés par la table 7.5. Ceci est dû à une optimisation de notre algorithme qui consiste à essayer les paires d'amorces en commençant par celles qui amplifient les segments les plus courts, puis à essayer itérativement celles qui amplifient des fragments de plus en plus grands.



(a) Une représentation du problème d'amorces compatibles

(b) Une transformation du problème de choix multiples compatibles

Figure 7.8: Transformation d'un graphe quelconque en un ensemble de sous-graphes bipartites modélisant le problème des amorces compatibles

7.7 Le multiplexage de PCR, un problème *NP*-complet

Nous montrons dans ce paragraphe que la détermination des paires d'amorces qui maximisent le nombre de loci contenus dans une expérience est un problème *NP*-complet, ce qui justifie l'utilisation d'un algorithme d'approximation.

Nous mentionnons ici que Gabriel Robins *et al.* ont montré le caractère *NP*-complet d'un problème assez voisin du nôtre, à savoir l'optimisation de la PCR sur des familles multigènes [71]. Leur problème est un problème de minimisation pour le choix d'amorces sur ces familles. Ils réduisent ce problème (un problème de super-chaîne approché, ou avec erreurs) au problème de *Couverture Minimum d'un Ensemble.*

Nous modélisons notre problème comme un ensemble de sous-graphes bipartites avec des arêtes additionnelles joignant des sommets appartenant à des sous-graphes différents (Figure 2 (a)). Dans ce graphe, chaque amorce est représentée par un sommet. L'ensemble des sommets est partitionné par locus, chaque locus correspondant à un sous-graphe bipartite. Dans notre exemple, des sommets appartenant au même locus sont représentés par le même caractère ($\bullet$ pour le locus 1, $\circ$ pour le locus 2, $*$ pour le locus 3); les amorces avant sont représentées sur la partie gauche de la Figure 2 (a), et les amorces arrières sur la partie droite.

Nous définissons deux types d'arêtes.

- Les *arêtes d'acceptation*, à l'intérieur d'un sous-graphe bipartite limité à un seul locus. Ces arêtes non-fléchées indiquent que les amorces avant et arrière qu'elles joignent sont compatibles.

- Les *arêtes d'incompatibilité*, qui joignent le sommet d'un locus à un sommet d'un autre locus. Ces arêtes fléchées indiquent que les amorces qu'elles joignent ne sont pas compatibles.

Nous étudions le "*Problème d'Amorces Compatibles*", en abrégé *PAC*, qui a la description suivante.

Instance du problème. Un graphe composé d'un ensemble de graphes bipartites $B_1, B_2, ..., B_J$. Les arêtes de ces graphes constituent un ensemble d'arêtes d'acceptation A. Un ensemble d'arêtes d'incompatibilité, ces arêtes joignant des paires de sommets qui n'appartiennent pas au même graphe bipartite. Un entier K.

Question. Est-il possible de choisir un sous-ensemble des arêtes d'acceptations $A' \subseteq A$ avec $|A'| \geq K$ telles que A' contienne au plus une arête de chaque B_i, $1 \leq i \leq J$, et tel qu'il n'y ait jamais deux sommets extrémités de ces arêtes qui soient également extrémités d'une arête d'incompatibilité.

Remarque. Le problème réel, tel qu'il est exposé dans la section 2.2 contient en outre des arêtes d'incompatibilité entre sommets d'un même sous-graphe (amorces avant et arrière d'un même locus). Il est donc de complexité supérieure au problème *PAC* de la présente section.

Nous utiliserons dans notre démonstration le *Problème des Choix Multiples Compatibles*, en abrégé *PCMC* (en anglais *Multiple Choice Matching Problem*), qui est connu pour être *NP*−complet [29]. Ce problème a la définition suivante.

Instance du problème. Un graphe $G = (V, E)$, une partition de E en ensembles disjoints $E_1, E_2, ..., E_J$, un entier positif K.

Question. Y a t-il un sous-ensemble $E' \subseteq E$ avec $|E'| \geq K$ tel qu'il n'y ait jamais deux arêtes dans E' qui aient un sommet commun et tel que E' contienne au plus une arête de chaque E_i, $1 \leq i \leq J$?

Commentaire. Le problème reste *NP*-complet même si G est bipartite.

Un graphe typique de *PAC* est présenté dans la Figure 2(a). Chaque arête d'un graphe *"Problème de Choix Multiples Compatibles"*, est transformée en une arête d'un graphe *PAC*. Des sommets vides (figurés par un $\star$) sont ajoutés, un pour chaque sous-ensemble de sommets de *PCMC*. La Figure 2(b) représente une telle transformation d'un graphe *PCMC* (non représenté sur notre figure) en un graphe *PAC*. Par conséquent, résoudre *PAC* en temps polynomial reviendrait à résoudre aussi *PCMC* en temps polynomial, ce qui contredirait le caractère *NP*-complet de *PCMC*.

Théorème 50. *Le problème d'amorces compatibles PAC est NP-complet.*

Preuve. Nous allons transformer en temps polynomial un graphe quelconque en un graphe de la forme *Problème de Compatibilité d'Amorces* (Figure 7.8).

Considérons une instance $I(PCMC)$ du *Problème des Choix Multiples Compatibles*. Nous transformons toute arête a_{ij} d'un ensemble E_i de $I(PCMC)$ en un sommet f_{ij} d'une instance $I(PAC)$ du *Problème des Amorces Compatibles*.

Nous transformons maintenant tout sommet $s_{ij,kl}$, avec $i \neq k$, extrémité commune de deux arêtes a_{ij} et a_{kl} de sous-ensembles différents E_i et E_k de $I(MCMP)$, en une *arête d'incompatibilité* $i_{ij,kl}$ joignant les sommets f_{ij} et f_{kl} dans $I(PAC)$. Nous associons à chaque ensemble E_i de $I(PCMC)$ un sommet vide v_i dans $I(PAC)$, et nous joignons ce sommet à chaque sommet f_{ij} par une *arête de compatibilité* c_{ij} (les notations indicielles ij et kl réfèrent ici aux arêtes des ensembles E_i et E_k d'indices respectifs j et k, et ne réfèrent pas à une notation indicielle des noeuds; des arêtes e_{ij} et e_{kl} peuvent donc se rencontrer). Nous ne transformons pas les arêtes de $I(PCMC)$ qui sont intersections d'arêtes d'un même sous-ensemble E_j, parce que ces intersections n'ont aucun effet dans le *Problème des Choix Multiples Compatibles*.

La Figure 7.8 (b) illustre une telle transformation, avec trois sous-ensembles E_1, E_2, E_3, et les sommets vides représentés par un $\star$.

Nous avons donc transformé tout graphe-*PCMC* en un graphe-*PAC*; la transformation elle-même est polynomiale.

Nous voyons donc que résoudre la question du problème *PAC* avec l'entier K résoudrait aussi le problème *PCMC* avec le même entier K; en effet, chaque solution du premier problème fournirait un ensemble A' d'arêtes de compatibilité, avec $|A'| \geq K$, et l'ensemble des sommets non vides de ces arêtes correspondrait clairement à un ensemble E' d'arêtes du graphe initial, avec $\mid E' \mid = \mid A' \mid \geq K$. Par Conséquent, le *Problème des Amorces Compatibles* est *NP*-complet. □

7.8 Analyse de l'algorithme MULTIPCR

7.8.1 Évaluation de la probabilité limite de rejet d'un locus

Cette section présente une analyse simplifiée de l'algorithme; elle donne une borne supérieure sur son optimalité.

Quand nous considérons le tableau 7.4, nous voyons qu'avec une distance d'électrophorèse $\delta = 5$, le nombre d'urnes remplies par des boules blanches ou par des boules noires (scores des 3'-sous-amorces et des sous-amorces respectivement), est presque égal au nombre total U d'urnes ($U = 256$), avec moins de 50 loci dans l'expérience. En revenant au cas $\delta = 1$, ceci signifie que nous atteignons déjà après 50 loci un système "stationnaire" où il n'y a plus d'urnes vides, et dont la composition en b urnes "blanches" et en n urnes "noires" restera constant jusqu'à la fin de l'expérience. Si s est le nombre de sous-amorces des amorces (en pratique, si $\sigma = 4$, on a $s = 17$ pour une longueur d'amorces de 20), en notant $\pi_{1,b,n}$ la probabilité d'acceptation d'une amorce par un système de U urnes soit blanches soit noires, nous avons

$$\pi_{1,b,n} = \frac{b}{U}\left(1 - \frac{b}{U}\right)^s \quad et \quad \pi_{1,30,226} \approx 0.014. \tag{7.3}$$

La probabilité π_{11} de compatibilité de deux amorces entre elles (jetées dans un système d'urnes vides), est

$$\pi_{11} = \frac{1}{U}\left(1 - \frac{1}{U}\right)^{2s} + \left(1 - \frac{1}{U}\right)\left(1 - \frac{2}{U}\right)^{2s} \approx 0.766. \tag{7.4}$$

L'algorithme *MULTIPCR* considère un nombre V d'amorces avant pour un locus, et, pour chaque amorce avant, un nombre R d'amorces arrières à une distance acceptable (entre 150 et 450 bp) de cette amorce avant; suivant la longueur des loci, V sera entre 100 et 500, tandis que R restera voisin de 50. Quand nous considérons un locus, la distribution du nombre d'amorces avant acceptées v et celle des amorces arrières acceptées r sont toutes deux binomiales. La faible valeur de $\pi_{1,30,226}$ nous autorise à utiliser l'approximation de Poisson pour ces deux distributions, avec des paramètres respectifs $\nu = V\pi_{1,b,n}$ et $\rho = R\pi_{11}\pi_{1,b,n}$. La probabilité Π de rejet d'un locus, à l'état stationnaire, est alors

$$\Pi(\nu(V), \rho(R)) \approx \sum_{i=0}^{\infty} \left(\Pr\{v = i\} \times (\Pr\{r = 0\})^i\right) = e^{-\nu + \nu e^{-\rho}}, \tag{7.5}$$

probabilité dont on trouve quelques valeurs pour $R = 50$ dans

V	250	300	350	400	450
$\Pi(\nu(V), \rho(50))$	0.230	0.171	0.128	0.095	0.071

7.8.2 Complexité de l'algorithme MULTIPCR

La distribution du nombres de boules dans des urnes est un problème bien connu, de manière exacte, comme de manière asymptotique, dans plusieurs modèles. Dans le cas d'un modèle fini et équidistribué, on peut consulter Johnson et Kotz [45] (p.110) pour une approche probabiliste directe, ou Flajolet et Sedgewick [74] (pp.428-429) pour une approche combinatoire. Ces deux approches nous donnent la distribution du nombre Y d'urnes occupées quand on jette n boules dans m urnes,

$$\Pr(Y = k) = \binom{m}{k} m^{-n} \left\{ {n \atop k} \right\}, \tag{7.6}$$

où le nombre de Stirling de deuxième espèce $\left\{ {n \atop k} \right\}$ représente le nombre de manières de partitionner n éléments en k sous-ensembles non-vides. Les nombres de Stirling de deuxième espèce sont aussi donnés par la formule

$$\left\{ {n \atop k} \right\} = \frac{n!}{k!}[z^n](e^z - 1)^k = [z^n]\frac{z^k}{(1-z)(1-2z)\dots(1-kz)}.$$

La difficulté pour étendre ces résultats à une analyse de notre algorithme vient de l'optimisation consistant à choisir la meilleure paire d'amorces. Nous n'avons pas, pour l'instant trouvé de modèle mathématique satisfaisant qui nous permette une analyse vraiment précise de l'algorithme MULTIPCR.

Nous donnons donc un ordre de grandeur approximatif de la complexité de cet algorithme.

La complexité de l'algorithme MULTIPCR est en $\mathcal{O}(Knl)$ avec $K \approx E\tau_v k_r \approx 6E$, où E est le nombre d'expériences, n est le nombre de *loci*, l est la longueur moyenne d'un *locus*, $\tau_v \approx 1/7$ est la probabilité de commencer une amorce avant sur une position, et $k_r \approx 40$ est le nombre approximatif d'amorces arrières pour une amorce avant. (Voir le paragraphe 7.6 pour les valeurs numériques de τ_v et k_r).

Compte tenu des résultats obtenus sur 248 loci (tableaux 7.4 et 7.5) avec un nombre moyen de 300 à 350 amorces avant par locus, nous constatons expérimentalement que l'algorithme MULTIPCR est quasi-optimal.

7.8.3 Dimension de l'espace de recherche

Il est intéressant d'avoir une idée de la dimension de l'*espace* dans lequel nous cherchons des *solutions* à notre problème.

Étant donnés $n = m \times k$ *loci*, où m est le nombre d'expériences et K le nombre moyen de *loci* dans une expérience, soit $p_{n,k}$ le nombre de partitions de n éléments en ensembles de taille k. (Un ensemble de taille k correspond dans ce modèle à une expérience de PCR-multiplex). Il y a un grand nombre de telles partitions. Nous donnons ici une idée de la valeur de $p_{n,k}$ pour les valeurs de n et de k correspondant à des applications biologiques.

Soit $P(z) = \sum \frac{p_{n,k}}{n!} z^n$ la fonction génératrice exponentielle des $p_{n,k}$. Nous comptons le nombre de partitions en considérant les ensembles $\mathcal{S}_k$ de k éléments, et les ensembles de tels ensembles. En notant $S_k(z)$ la fonction génératrice exponentielle des $\mathcal{S}_k$, des méthodes de calcul symbolique standards (voir Flajolet et Sedgewick [74]) nous donnent

$$S_k(z) = \frac{z^k}{k!} \quad \text{et} \quad P(z) = e^{S_k(z)} = e^{\frac{z^k}{k!}}. \tag{7.7}$$

Nous en déduisons

$$p_{mk,k} = \frac{(mk)!}{m!(k!)^m}.$$

Pour des applications réelles en biologie, on ne peut espérer amplifier simultanément plus de 20 *loci*. Considérons à titre d'exemple le cas $n = 200$ *loci*, $m = 10$ expériences, et $k = 20$ *loci* par expérience. Nous obtenons

$$p_{200,20} = \frac{200!}{10!(20!)^{10}} \approx 3 \times 10^{185}.$$

Il est clair que notre algorithme d'approximation réduit radicalement la taille de l'espace dans lequel nous recherchons effectivement des solutions.

7.9 Conclusion

Nous avons exploré un modèle de compatibilité entre amorces qui permette le multiplexage des réactions de polymérisation en chaîne. Ce modèle considère qu'il est essentiel d'éviter la complémentarité renversée de l'extrémité d'une amorce avec des sous-séquences d'une amorce quelconque dans le mélange utilisé pour l'expérience multi-PCR. Les résultats expérimentaux de nos calculs montrent que quand les loci sont suffisamment longs, il devrait être possible d'amplifier simultanément un nombre élevé de loci. En l'état actuel, notre algorithme MULTIPCR a été bâti en utilisant la méthode de PCR traditionnelle, où la longueur des fragments amplifiés est limitée à environ 500 bp.. Ceci induit d'importantes limitations à notre approche quand nous voulons l'appliquer à des *loci* très polymorphes. La composition de ces *loci* polymorphes varie fortement d'un individu à l'autre, ce qui cause d'importantes variation de longueur des fragments amplifiés avec une même paire d'amorces. Il est courant pour de tels *loci* que les variations de longueur de fragments amplifiés puissent atteindre 50 bp., et ceci limite à 4 ou 5 le nombre de *loci* que l'on peut discriminer par électrophorèse. Notre méthode n'est pas non plus applicable efficacement dans le cas où l'on part d'un dictionnaire d'amorces, qui fournit une paire d'amorces pour chaque *locus*. Cependant, l'apparition récente d'une nouvelle méthode de PCR, la *Long Accurate PCR* [11, 10], ouvre de nouvelles perspectives. Cette méthode permet l'amplification de très longs fragments (Jusqu'à 35-kb). Elle devrait offrir des champs nouveaux d'application aux méthodes de multiplexage et aux approches algorithmiques telles que la notre et celle de Robins et *al.* [67],

7.10 Appendice

Nous donnons dans cet appendice des exemples d'entrée et de sortie du programme *MULTIPCR*, qui donnent une illustration concrète de notre algorithme. Nous avons pris comme exemple le multiplexage des 76 *loci* du gène de la dystrophine.

L'entrée du programme MULTIPCR (figure 7.10) est une légère transformation de la sortie auxiliaire du programme PRIMER (figure 7.9).

```
fichier HSDMD17.for   (sélection des lignes ok seulement)
      a                    b                    c     d     e f
>    30   TCCCTTGCTATTTCAGTGAACC   60.5 45% | 7 5 || ok
>    31   CCCTTGCTATTTCAGTGAACC    58.7 48% | 7 5 || ok
                    ....
                    ....

fichier HSDMD17.rev   (sélection des lignes ok seulement)

<   173   GACTGCTTGGAGTGGCAAAG     61.0 55% | 4 3 || ok
<   174   AGACTGCTTGGAGTGGCAAA     61.0 50% | 4 2 || ok
<   175   AAGACTGCTTGGAGTGGCAA     61.0 50% | 4 3 || ok
                    ....
                    ....

a - position de l'amorce dans le locus
b - amorce
c - température de fusion
d - pourcentage en GC
e - degré d'auto-complémentarité, toutes sous-amorces
f - degré d'auto-complémentarité, 3'-sous-amorce
```

Figure 7.9: Édition de sortie du programme PRIMER.

```
HSDMD17  for   30   TCCCTTGCTATTTCAGTGAACC   60.5 45% | 7 5 || ok
HSDMD17  for   31   CCCTTGCTATTTCAGTGAACC    58.7 48% | 7 5 || ok
                         ....
                         ....

HSDMD17  rev  173   GACTGCTTGGAGTGGCAAAG     61.0 55% | 4 3 || ok
HSDMD17  rev  174   AGACTGCTTGGAGTGGCAAA     61.0 50% | 4 2 || ok
HSDMD17  rev  175   AAGACTGCTTGGAGTGGCAA     61.0 50% | 4 3 || ok
                         ....
                         ....
```

Figure 7.10: L'entrée du programme MULTIPCR.

```
----------------------------------------------------------------------------
PCR MULTIPLEX: CHECKING ON 5 BASES LONG
               LENGTHS OF SEGMENTS AMPLIFIED DIFFER BY MORE THAN 4 BASES
----------------------------------------------------------------------------

 HUMDYSTR       number of pairs of primers =       0
 HUMDYST21      number of pairs of primers =       9
 HUMDYST14      number of pairs of primers =      35
 HSDYSTITH      number of pairs of primers =     225
                   .............
 HUMDYST08      number of pairs of primers =   11877
 HSDMD45        number of pairs of primers =   12138
                   .............
 HSATP          number of pairs of primers =  2162028
 HSAHNAKA       number of pairs of primers =  2178455
---------------------------------------------------------------------------
 MULTIPLEX RESULTS: number of loci = 76       number of experiments = 2
---------------------------------------------------------------------------
     EXPERIMENT-01: number of loci = 49
---------------------------------------------------------------------------
exp-01 HSA1GLY2           for  2549 ATGGGACCGTCTCCAGATAC
exp-01 HSA1GLY2           rev  2916 GAAGTTGATAGTTGGGGTAGTG
exp-01 HSA1GLY2           len = 367  (including the primers)

exp-01 HSA1MBG1           for  2454 TTGAGAATACCCTTCCCACC
exp-01 HSA1MBG1           rev  2868 GTAGAACTCTCTCTAACGGG
exp-01 HSA1MBG1           len = 414
                   .............

exp-01 HUMDYSVB           for   258 GCGCATCCTAAATCAGGAGT
exp-01 HUMDYSVB           rev   596 CTAATGTCAAGTGTTCCGTGAG      v
exp-01 HUMDYSVB           len = 338
---------------------------------------------------------------------------
exp-01    3'-subprimers score = 72    subprimers score = 693
---------------------------------------------------------------------------

---------------------------------------------------------------------------
     EXPERIMENT-02: number of loci = 15
---------------------------------------------------------------------------
exp-02 HSA2M              for  2874 CCCGAGCTTCTGTCTCAGTT
exp-02 HSA2M              rev  3066 AGTGAGATAGCCAATGGCCT
exp-02 HSA2M              len = 192
                   .............

exp-02 HUMDYST18          for   144 TCAACAACTCCTTCCCTAGTTC
exp-02 HUMDYST18          rev   310 AATCTTTTCACCATGGACCC
exp-02 HUMDYST18          len = 166
---------------------------------------------------------------------------
exp-02    3'-subprimers score = 27    subprimers score = 344
---------------------------------------------------------------------------

---------------------------------------------------------------------------
LOCI WITH NO POSSIBLE PCR AMPLIFICATION WITHIN THE GIVEN CONDITIONS: 12 LOCI
---------------------------------------------------------------------------
HSDYSTIA
.....
HUMDYSVA
---------------------------------------------------------------------------
```

Figure 7.11: Un exemple de sortie du programme MULTIPCR

La figure 7.11 donne une partie de la sortie de MULTIPCR obtenue avec les 76 *loci* du gène responsable de la dystrophine. La longueur de contrôle est $\sigma = 5$, et la distance d'électrophorèse $\delta = 5$. (On peut consulter la figure 7.3 pour les scores obtenus pour ces *loci*).

Dans cette figure, le format d'une ligne pour une amorce est donné par l'exemple suivant (`for` et `rev` pour l'anglais *forward* et *reverse*, respectivement *avant* et *arrière* en français).

expérience	nom du locus	for/rev	position dans le locus	amorce
exp-02	HUMDYST18	rev	310	AATCTTTTCACCATGGACCC

Chapitre 8

Conclusion

Nous cherchons ici à conclure brièvement et à proposer quelques axes de recherches.

Le travail effectué sur la recherche de similarité avec des multi-alignements nous a conduit à étudier et développer l'algorithme BlastMultAl. Nous sommes conscients des limites de l'approche que nous avons effectuée. L'utilisation de chaînes de Markov cachées rencontre actuellement beaucoup d'intérêt dans la communauté bioinformatique. Il faut cependant noter que cette approche se prête mal à l'étude d'alignements multiples contenant peu de séquences, ce qui est le cas le plus fréquent pour la base ProDom. Blast, d'autre part a été considérablement amélioré par Gish, et prend en compte désormais le traitement des gaps. Enfin, de nombreux chercheurs préconisent actuellement l'utilisation de matrices de scores dépendant des positions.

On pourrait utiliser ces nouvelles techniques en considérant des alphabets étendus, où une lettre de l'alphabet correspondrait à une distribution d'acides aminés sur une colonne d'alignement multiple. En supposant que l'on sache définir cette alphabet, et effectuer une projection de l'ensemble des distributions sur cet alphabet, on pourrait utiliser avec un coût d'adaptation minime la version de Blast traitant les gaps. Ceci amènerait à construire une matrice pouvant dépendre des positions et affectant un score à l'appariement d'une lettre de l'alphabet étendu et d'une lettre de l'alphabet des acides aminés. Le pré-traitement d'un tel algorithme serait plus coûteux, mais on pourrait espérer obtenir, avec les mêmes performance que Blast, une plus grande sensibilité dans la recherche de similarité. Cette approche partiellement apparentée à celle des profiles permettrait également d'effectuer un lien entre le modèle mathématique de Karlin utilisé dans le cas de l'alignement $1-1$ et le modèle mathématique des profiles.

Sur le plan mathématique, on peut remarquer que le résultat de Karlin fournit une queue de distribution pour les alignements sans gaps. Si cela est dans l'ensemble satisfaisant pour les applications, le problème de la distribution asymptotique complète du score de l'alignement optimal reste posé. De même, dans le cas où l'on a des gaps, on ne connaît que la valeur asymptotique de l'espérance du score de l'alignement optimal, et on aimerait avoir des formules analytiques pour la queue de distribution étudiée numériquement par Waterman et Vingron.

Le multiplexage de PCR est toujours un problème technique d'importance majeure pour les biologistes. Depuis le moment où nous avons effectué notre travail, la découverte de la possibilité d'amplifier de très longs fragments d'ADN a modifié les données du problème. Il serait certainement indispensable de définir en collaboration avec des biologistes le modèle de compatibilité permettant des amplifications multiples de très longs fragments. La définition d'un tel modèle et l'extension des travaux effectués permettrait certainement d'améliorer beaucoup l'efficacité des multiplexages de PCR.

Nous espérons que l'avenir nous permettra de contribuer aux avancées de la recherche dans ces différents domaines.

Bibliographie

[1] Aho, A. V., and Corasick, M. J. Efficient String Matching: An Aid to Bibliographic Search. *Communications of the ACM 18* (1975), 333–340.

[2] Aldous, D. *Probability Approximations via the Poisson Clumping Heuristic.* Springer Verlag U.T.M., 1989.

[3] Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. Basic Local Alignment Search Tool. *J. Mol. Biol. 215* (1990), 403–410.

[4] Arnold, B. C., Balakrishnan, N., and Nagaraja, H. N. *A First Course in Order Statistics.* John Wiley, 1992.

[5] Arratia, R., Goldstein, L., and Gordon, L. Two moments suffice for Poisson approximations: the Chen-Stein method. *The Annals of Probability 17* (1989), 9–25.

[6] Arratia, R., Gordon, L., and Waterman, M. S. The Erdös-Rényi law in distribution, for coin tossing and sequence matching. *The Annals of Statistics 18* (1990), 539–570.

[7] Arratia, R., and Waterman, M. A phase transition for the score in matching random sequences allowing deletions. *The Annals of Applied Probability 4* (1994), 200–225.

[8] Bak, J., and Newman, D. J. *Complex Analysis.* Springer Verlag U.T.M., 1997.

[9] Barbour, A., Holst, L., and Janson, S. *Poisson Approximation.* Oxford Science Publications, 1992.

[10] Barnes, W. M. Effective amplification of long targets from cloned inserts and human genomic DNA. *Proc. Natl. Acad. Sci. USA 91* (June 1994), 5695–5699.

[11] Barnes, W. M. PCR amplification of up to 35-kb DNA with high fidelity and high yield from λ bacteriophage templates. *Proc. Natl. Acad. Sci. USA 91* (March 1994), 2216–2220.

[12] Billingsley, P. *Probability and Measure.* John Wiley, 1979.

[13] Boyer, R. S., and Moore, J. S. A fast string-searching algorithm. *Communication of the ACM 20*, 10 (1977), 762–772.

[14] CHEN, L. H. Y. Poisson approximation for dependant trials. *The Annals of Probability 3* (1975), 534–545.

[15] CHUNG, K. L. *A Course in Probability Theory.* Academic Press, 1974. Second edition.

[16] CHURCHILL, R. V. *Complex Variables and Applications.* Mac Graw-Hill, 1960.

[17] CHVÁTAL, V., AND SANKOFF, D. Longuest common subsequence of two random sequences. *The Annals of Probability 12* (1975), 306–315.

[18] CORPET, F. Multiple sequence alignment with hierarchical clustering. *Nucl. Acids Res., 16* (1988), 10881–10890.

[19] CROCHEMORE, M. Algorithmes sur les mots. Cours de DEA d'informatique fondamentale, Université Paris-VII, 1991.

[20] CROCHEMORE, M., AND RYTTER, W. *Text Algorithms.* Oxford University Press, 1994.

[21] DACUNHA-CASTELLE, D., AND DUFLO, M. *Probabilité et Statistiques.* Masson, 1983.

[22] DAYHOFF, M. O., SCHWARTZ, R. M., AND ORCUTT, B. C. Atlas of protein sequence and structure. *Atlas of Protein Sequence and structure 5, suppl. 3* (1978), 345–352. Nat. Biomed. res. Found. Washington. DC.

[23] DEHEUVELS, P. *L'Intégrale.* P.U.F., 1980.

[24] DEVLIN, K. *Mathématiques - Un nouvel âge d'or.* Masson, 1992. Traduit de l'anglais - Mathematics: the New Golden Age.

[25] DIEUDONNÉ, J. *Éléments d'Analyse.* Gauthier-Villars, 1968.

[26] DURRETT, R. *Probability: Theory and Examples.* Duxburry Press, 1996. Second edition.

[27] FELLER, W. *An Introduction to Probability theory and Its Applications.* John Wiley, 1966. Second edition.

[28] FLAJOLET, P., AND SEDGEWICK, R. The Average Case Analysis of Algorithms: Complex Asymptotics and Generating Functions. Research Report 2026, Institut National de Recherche en Informatique et en Automatique, Sept. 1993.

[29] GAREY, M. R., AND JOHNSON, D. S. *Computers and Intractability, A Guide to the Theory of NP-completeness.* W. H. Freeman and Company, 1979. Appendix.

[30] GERSTEIN, M., SONNHAMER, E., AND CHOTHIA, C. Volume changes in protein evolution. *J. Mol. Biol. 236* (1994), 1067–1078. Appendix.

[31] GOLDSTEIN, L., AND WATERMAN, M. S. Approximations to Profile Score Distributions. *Journal of Computational Biology 1*, 2 (1994), 93–104.

[32] GOURDON, X. *Les Maths en tête - Analyse.* Ellipses, 1994.

[33] GRAHAM, R., KNUTH, D., AND PATASHNIK, O. *Concrete Mathematics.* Addison-Wesley, 1989.

[34] GREBINSKI, V., AND KUCHEROV, G. Reconstructing a Hamiltonian circuit by querying the graph: Application to DNA physical mapping. INRIA-Lorraine and CRIN/CNRS, Campus Scientifique, 615 rue du Jardin Botanique, BP 101, 54602 Villers-lès-Nancy, France, 1996.

[35] GRIBSKOV, M. Profile analysis. *Methods in Molecular Biology 25* (1994), 247–266.

[36] GRIBSKOV, M., LÜTHY, R., AND EISENBERG, D. Profile analysis. *Methods in Enzymology 183* (1990), 146–159.

[37] GRIBSKOV, M., MACLACHLAN, A., AND EISENBERG, D. Profile analysis: detection of distantly related proteins. *Proc. Natl. Acad Sci. USA 84* (1987), 4355–4358.

[38] GUT, A. *Stopped Random Walks: limit theorems and applications.* Applied Probability, 1988.

[39] HARRISON, J. K., PEARSON, W. R., AND LYNCH, K. R. Molecular characterization of alpha-1 and alpha-2 adrenoreceptors. *Trends Pharm. Sci. 12* (1991), 62–67.

[40] HENIKOFF, S., AND HENIKOFF, J. G. Automated assembly of proteins blocks for database searching. *Nucleic Acids Research 19*, 23 (1991), 6565–6572.

[41] HENIKOFF, S., AND HENIKOFF, J. G. Amino acid substitution matrices from proteins blocks. *Proc. Natl. Acad Sci. USA 89* (1992), 10915–10919.

[42] HIGGINS, D., FUCHS, R., AND BLEASBY, A. J. CLUSTAL V: improved software for multiple sequence alignment. *CABIOS 8* (1992), 189–191.

[43] HODGES, J., AND CAM, L. L. The Poisson approximation to the binomial distribution. *Ann. Math. Statis. 31* (1960), 737–740.

[44] IGLEHART, D. L. Extremes values in the GI/G/1 queues. *The annals of Mathematical Statistics 43*, 2 (1972), 627–635.

[45] JOHNSON, N., AND KOTZ, S. *Urn Models and Their Application.* Wiley Series in Probability and Mathematical Statistics, 1977.

[46] KARLIN, S., AND ALTSCHUL, S. F. Methods for assessing the statistical significance of molecular sequences features by using general scoring schemes. *Proc. Natl. Acad Sci. USA 87* (March 1990), 2264–2268.

[47] Karlin, S., and Altschul, S. F. Applications and statistics for multiple high-scoring segments in molecular sequences. *Proc. Natl. Acad Sci. USA 90* (June 1993), 5873–5877.

[48] Karlin, S., and Brendel, V. Chance and statistical significance in protein and DNA sequence analysis. *Science 257* (July 1992), 39–49.

[49] Karlin, S., and Dembo, A. Strong limit theorems of empirical functionals for large excedances of partial sums of i.i.d. variables. *The Annals of Probability 19*, 4 (1991), 1737–1755.

[50] Karlin, S., and Dembo, A. Limit distribution of maximal segmental score along Markov-dependant partial sums. *Adv. Appl. Prob. 24* (1992), 113–140.

[51] Karlin, S., Dembo, A., and Kawabata, T. Statistical composition of high-scoring segments from molecular sequences. *The Annals of Statistics 19*, 2 (1990), 571–581.

[52] Karlin, S., and Taylor, H. *A first course in Stochastic Processes.* Academic Press, 1975. Second Edition.

[53] Karlin, S., and Taylor, H. *A second course in Stochastic Processes.* Academic Press, 1975. Second Edition.

[54] Khoan, V.-K. *Distributions, Analyse de Fourier, Opérateurs aux dérivées partielles.* Vuibert, 1972. tome 1.

[55] Kingman, J. Subadditive ergodic theory. *The Annals of Probability 1* (1973), 883–909.

[56] Knuth, D. E., Morris, J. H., and Pratt, V. R. Fast pattern matching in strings. *SIAM Journal on Computing 6*, 2 (1977), 323–350.

[57] Laha, R., and Rohatgi, V. *Probability Theory.* John Wiley, 1979.

[58] Lerman, I. C., Peter, P., and Risler, J. L. Matrices AVL pour la classification et l'alignement de séquences protéiques. Research Report 2466, INRIA, Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 Le Chesnay Cedex, France, Sep. 1994.

[59] Lincoln, S. E., Daly, M. J., and Lander, E. S. *PRIMER: A Computer Program for Automatically Selecting PCR Primers.* MIT Center for Genome Research and Whitehead Institute for Biomedical Research, Nine Cambridge Center, Cambridge, Massachusetts 02142, 1991.

[60] Needelman, S. B., and Wunsch, C. D. A general method applicable to the search of similarities in the amino acid sequences of two proteins. *J. Mol. Biol. 48* (1970), 443–453.

[61] Neveu, J. Cours de Troisième Cycle. Laboratoire de Probabilités, Université Paris VI, 1983-1984.

[62] NICODÈME, P. A computer support for genotyping by multiplex PCR. Research Report LIX/RR/93/09, LIX, École Polytechnique, 91128, Palaiseau Cedex, France, 1993.

[63] NICODÈME, P. BlastMultAl, a Blast extension for similarity searching with alignment graphs. Research Report 2911, INRIA, Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay, France, 1996.

[64] NICODÈME, P., AND STEYAERT, J. M. Selecting optimal oligonucleotide primers for multiplex PCR. In *Fifth International Conference on Intelligent Systems for Molecular Biology* (1997), AAAI Press, pp. 210–213.

[65] OLSCHWANG, S., DELAITRE, O., MELOT, T., PETER, M., SCHMITT, A., FRELAT, G., AND THOMAS, G. Description and use of a simple laboratory-made automat for *in vitro* DNA amplification. *Methods in Molecular and Cellular Biology 1*, 3 (May/June 1989), 121–127.

[66] PEARSON, W. R., AND LIPMAN, D. J. Improved tools for biological sequences comparisons. *Proc. Natl. Acad Sci. USA 85* (1988), 2444–2448.

[67] PEARSON, W. R., ROBINS, G., WREGE, D. E., AND ZHANG, T. A new approach to primer selection in Polymerase Chain Reaction experiments. In *Third International Conference on Intelligent Systems for Molecular Biology* (1995), AAAI Press, pp. 285–291.

[68] PURVES, W. K., ORIANS, G. H., AND HELLER, H. C. *Le Monde du Vivant.* Flammarion, 1994. Traduction française de - Life, the Science of Biology.

[69] ROBERT, P. Cours de DEA - Files d'attente. Laboratoire de Probabilités, Université Paris VI, 1993-1994.

[70] ROBERT, P. Cours de DEA de Probabilité - Approximations Poissonniennes. Laboratoire de Probabilités, Université Paris VI, 1995-1996.

[71] ROBINS, G., WREGE, D. E., ZHANG, T., AND PEARSON, W. R. On the primer selection problem in polymerase chain reaction experiments. Research Report CS-93-68, Department of Computer Science, University of Virginia, Charlottesville, Virginia 22903, U.S.A., 1993.

[72] ROSS, S. *Stochastic Processes.* John Wiley, 1983.

[73] RUDIN, W. *Principles of Mathematical Analysis.* Mac Graw-Hill, 1976.

[74] SEDGEWICK, R., AND FLAJOLET, P. *An Introduction to the Analysis of Algorithms.* Addison-Wesley, 1996.

[75] SMITH, T., AND WATERMAN, M. S. Identification of common molecular subsequences. *Journal of Molecular Biology 147* (1981), 195–197.

[76] Sonnhamer, E. L., and Kahn, D. The modular arrangement of proteins as inferred from analysis of homology. *Protein Science 3* (1994), 482–492.

[77] Sorokin, A., Lapidus, A., Capuano, V., Galleron, N., Putjic, P., and Ehrlich, S. D. A new approach using multiplex long accurate PCR and yeast artificial chromosomes for bacterial mapping and sequencing. *Genome Research 6* (1996), 448–453.

[78] Spitzer, F. *Principles of Random Walk.* Van Nostrand, 1964.

[79] Stein, C. M. A bound for the error in the normal approximation to the distribution of a sum of dependant random variables. vol. 3, Proc. Sixth Berkeley Symp. Math. Statist. Probab., Univ. California Press, pp. 583–602.

[80] Stryer, L. *Molecular Design of Life.* W. H. Freeman and Company, 1988.

[81] Tatusov, R., Altschul, S. F., and Koonin, E. V. Detection of conserved segments in proteins: Iterative scanning of sequences databases with alignments blocks. *Proc. Natl. Acad Sci. USA 91* (1994), 12091–12095.

[82] Theilhaber, J. A Self-Contained Tutorial on the Random Walk Theory behind Maximal Segment Pair Statistics in Biological Sequence Comparisons. Bioinformatics Group. IBM Corporation, 404 Wyman Street, Waltham, MA 02254, USA, Feb. 1997.

[83] Titchmarsh, E. *The theory of functions.* Oxford Science Publications, 1939. Second edition.

[84] Vingron, M., and Waterman, M. S. Sequence alignment and penalty choice. Review of concepts, case studies and implications. *Journal of Molecular Biology 235* (1994), 1–12.

[85] Waterman, M. S. *Introduction to Computational Biology.* Chapman & Hall, 1995.

[86] Waterman, M. S., and Vingron, M. Sequence comparison significance and Poisson approximation. *Statistical Science 3* (1994), 367–381.

[87] Watson, J. D., Witkowski, J., Gilman, M., and Zoller, M. *Recombinant DNA.* Scientific American Books, 1992. Second edition.

[88] Whittaker, E., and Watson, G. *A Course of Modern Analysis.* Cambridge University Press, 1965.

[89] Zhang, Y. A limit theorem for matching random sequences allowing deletions. *The Annals of Applied Probability 4* (1995), 1236–1240.

Liste des figures

Liste des tableaux

Table des matières