Indexation et comparaison d'une grande quantité de données génomiques à l'aide d'algorithmes pour le traitement d'images

Jocelyn de Goër de Herve

UNIVERSITÉ Clermont Auvergne

École Doctorale
des Sciences pour l'Ingénieur (EDSPI)

UNIVERSITÉ CLERMONT AUVERGNE

UMR 6158 CNRS LIMOS

Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes

THÈSE DE DOCTORAT

Présentée par

M. Jocelyn DE GOËR DE HERVE

pour obtenir le grade de

Docteur d'Université

Spécialité : **Informatique**

Indexation et comparaison d'une grande quantité de données génomiques à l'aide d'algorithmes pour le traitement d'images

---

Soutenue publiquement le 09 avril 2019, devant un jury composé de :

Rapporteurs :	**Mme. Karine BENNIS ZEITOUNI**	Pr. à l'UVSQ, Vélizy
	M. Vladimir MAKARENKOV	Pr. à l'UQAM, Montréal (Québec)
Examinateurs :	**M. Mohand-Said HACID**	Pr. à l'UCB, Lyon
	M. Laurent BRÉHÉLIN	CR. au LIRMM, Montpellier
Directeurs de thèse :	**M. Engelbert MEPHU-NGUIFO**	Pr. à l'UCA, Clermont-Ferrand
	Mme. Myoung-Ah KANG	MCF. à l'UCA, Clermont-Ferrand

« Les plus grandes innovations du XXI^{ème} siècle seront selon moi,

au carrefour de la biologie et de la technologie »

S. Paul JOBS

REMERCIEMENTS

Je tiens tout d'abord, à adresser mes plus sincères remerciements à M. Engelbert MEPHU-NGUIFO (Professeur à l'UCA) et à Mme. Myoung-Ah KANG (Maître de conférence à l'UCA) pour m'avoir accueilli au sein du LIMOS et accompagné en tant que Directeurs de Thèse, durant ce projet doctoral. Leurs judicieux conseils, leurs critiques constructives et surtout leur bienveillance ont été pour moi un apport inestimable.

Mes remerciements les plus sincères vont aussi à la Direction de mon Unité INRA, à M. Xavier BAILLY (Ingénieur de Recherche INRA et Directeur de l'UMR EPIA), à Mme. Karine. CHALVET-MONFRAY (Professeur à VetAgroSup et Directrice Adjointe de l'UMR EPIA), ainsi qu'à Mme. Gwenaël VOURC'H (Directrice de Recherche INRA et Directrice de l'UMR EPIA) et à M. Christian DUCROT (Directeur de Recherche à l'INRA Directeur de l'UMR EPIA et Chef de Département adjoint du département de Santé Animale de l'INRA), qui m'ont accordé leur confiance pour la construction de ce projet de thèse et m'ont permis d'intégrer le LIMOS durant quelques jours par semaine pour sa réalisation.

Ma reconnaissance et mes remerciements vont aussi à M. Farouk TOUMANI (Professeur à l'UCA et Directeur du LIMOS) pour m'avoir accueilli au sein de son laboratoire et m'avoir mis un bureau à disposition. Je remercie aussi l'ensemble des différents personnels techniques et administratifs du LIMOS pour leur sympathique accueil et pour les différentes aides qu'ils ont pu m'apporter.

J'exprime ma gratitude à Mme. Karine BENNIS-ZEITOUNI (Professeure à l'UVSQ), à M. Vladimir MAKARENKOV (Professeur à l'UQAM), à M. Mohand-Said HACID (Professeur à l'UCB) et à M.

Laurent BRÉHÉLIN (Chargé de Recherche au LIRMM) pour avoir accepté d'évaluer ce travail de thèse.

Je tiens aussi à remercier les différentes personnes qui ont, de près ou de loin contribuées à cette thèse, que ce soit par leurs encouragements (« t'en es où ? »), pour leurs aides techniques (Hayfa, Marina, David, Louis et Emmanuel) pour les conseils judicieux en termes de bibliographie (A. Elbakyan) ou pour les travaux de relecture de ce présent manuscrit (Emmeke et Samuel).

Enfin, mes remerciements vont aussi aux différentes personnes qui ont partagé mon quotidien durant ces années de Thèse. Qu'elles aient été directement impliquées ou non, elles m'ont permis, grâce aux bons moments passés avec elles, en tant que co-bureaux ou autour d'un café, d'égayer mon quotidien et parfois de me remonter le moral. Pour moi, la Recherche est aussi une question de rencontre, d'échange et de convivialité. Ainsi, j'ai une pensée toute particulière pour Philippe, Jinhua, Dan, David, Myriam, Émilie, Françoise, Nelly, Jean, Ismaella, Samuel, Haizou, Marc, Nestor, Libo et Yanik, ainsi que tous les membres de l'UMR EPIA. Sans oublier mes deux compères, Vanel et Suan, avec qui nous avons pu vérifier la théorie selon laquelle l'informaticien est une machine à transformer la caféine en code informatique et en productions scientifiques.

Je n'oublie pas non plus ma petite famille (Emmeke, Lizon et Maëlle) qui ont supporté ma thèse (dans tous les sens du terme), au court de ces différentes années.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

RÉSUMÉ

L'accroissement constant des capacités de séquençage de l'ADN entraîne l'émergence de nouveaux questionnements biologiques. Le stockage et le traitement de cette masse d'informations restent des enjeux majeurs pour les années à venir. Durant le processus d'analyse des données génomiques, la recherche de séquences exactes ou proches, au travers de bases de données de génomes de références, est une tâche incontournable. Elle est notamment nécessaire dans les phases d'assemblage, d'alignement de séquences et plus généralement pour identifier la séquence de référence la plus proche d'une séquence requête. Ces tâches sont notamment essentielles dans le cadre d'étude en Biologie Évolutive, en Phylogénie ou en Métagénomique.

Traditionnellement, une grande majorité des techniques servant à réaliser ces différentes tâches, sont issues de méthodes en algorithmique du texte. L'objectif de cette thèse, est d'évaluer la possibilité d'utiliser des algorithmes issus du domaine de la comparaison des images numériques. En effet, les méthodes de production des images numériques ont connu une importante augmentation depuis ces 40 dernières années, entrainant des problèmes de recherche et de comparaison, qui par certains aspects, peuvent être considérés comme étant similaires aux traitements nécessaires à l'analyse des données génomiques.

Au cours de cette thèse, nous nous sommes plus particulièrement intéressés au concept de hachage perceptuel, utilisé habituellement pour indexer et comparer des images numériques, afin de déterminer si de telles méthodes sont pertinentes pour comparer des séquences exactes ou approchées au sein de bases de données de séquences de références. Ainsi, nous proposons deux contributions. La première est une fonction de hachage perceptuel,

permettant l'indexation de séquences ADN/ARN. Outre une diminution importante des données indexées par rapport aux séquences fournies en entrée, cette fonction de hachage a la particularité de conserver la propriété de comparabilité entre deux clés de hachage. Deux séquences ADN/ARN proches, auront des clés de hachage également proches et ainsi comparables. La seconde contribution, est l'adaptation d'une méthode permettant de faire ressortir les zones communes entre deux images, à la problématique de la comparaison de séquences ADN.

Ces travaux se placent dans un contexte d'accroissement des volumes de données génomique, où l'enjeu est de concevoir des algorithmes permettant d'identifier rapidement les génomes de référence les plus proches d'une séquence requête. Le but étant d'effectuer un prétraitement rapide, permettant de ne conserver que des séquences pertinentes et par la suite d'utiliser des méthodes plus classiques en bio-informatique.

ABSTRACT

The constant improvement of DNA sequencing techniques is leading to the emergence of new biological questions. The storage and processing of this large amount of data remains a major challenge for the coming years. During the genomic data analysis process, the search for exact or close sequences, through reference genome databases, is an essential task. It is particularly necessary for genome assembly, sequence alignment and more generally to identify the reference sequence closest to a request sequence. These tasks are particularly essential in the context of studies in Evolutionary Biology, Phylogeny or Metagenomics.

Traditionally, techniques used to perform these different tasks have been developed using text algorithmic methods. The objective of this PhD thesis is to evaluate the possibility of using algorithms coming from the domain of digital image comparison. Indeed, methods of producing digital images have increased significantly over the past 40 years, leading to research and comparison problems, which in some respects can be considered similar to the processing required for the analysis of genomic data.

In this PhD thesis, we focused on the concept of perceptual hashing, usually used to index and compare digital images, to determine whether such methods are relevant for comparing exact or approximated DNA/RNA sequences within reference sequence databases. Thus, we propose two contributions. The first one is a perceptual hash function, allowing DNA/RNA sequence indexing. In addition to a significant decrease in indexed data compared to the sequences provided as input, this hash function has the particularity of maintaining the comparability property between two hash keys. Two close DNA/RNA sequences will have hash keys that are also close and therefore comparable. The second contribution is the adaptation

of a method to bring out the common areas between two images, to the problem of DNA sequence comparison.

This work is taking place in a context of increasing volumes of genomic data, where the challenge is to design algorithms in order to identify quickly the reference genomes closest to a request sequence. The aim is to perform rapid pre-processing, allowing only relevant sequences to be stored and then to use more traditional bioinformatics methods.

PUBLICATIONS ET COMMUNICATIONS

Jocelyn De Goër, Myoung-Ah Kang, Xavier Bailly, Engelbert Mephu-Nguifo – *PSH-DB, un système clé- valeur permettant l'indexation et la recherche de séquences ADN – Conférence BDA 2017 – Nancy (France)*

Jocelyn De Goër, Myoung-Ah Kang, Xavier Bailly and Engelbert Mephu-Nguifo – *PSH-DB, a key-value system to index and retrieve biological DNA sequences – Lightning Talks - XLDB2017, 10th Extremely Large Databases Conference – Octobre 2017 – Clermont-Ferrand (France)*

Jocelyn DE GOËR DE HERVE, Myoung-Ah KANG, Xavier BAILLY, Engelbert MEPHU NGUIFO – *Une nouvelle approche de comparaison de séquences ADN à l'aide d'une fonction de hachage perceptuel - Session poster – JOBIM16 – Juin 2016 – Lyon (France)*

Jocelyn DE GOËR DE HERVE, Myoung-Ah KANG, Xavier BAILLY, Engelbert MEPHU NGUIFO – *Indexation et comparaison de séquences ADN à partir d'une fonction de hachage perceptuel Session poster – JOBIM15 – Juillet 2015 – Clermont-Ferrand (France)*

Jocelyn DE GOËR DE HERVE, Myoung-Ah KANG, Xavier BAILLY, Engelbert MEPHU NGUIFO – *A perceptual hash algorithm for indexing and similarity search in a database of DNA sequences Poster Session – ECCB15 – Septembre 2014 – Strasbourg (France)*

INTRODUCTION GÉNÉRALE

Le développement constant des nouvelles technologies du numérique entraîne depuis une quarantaine d'années, la production de données avec un rythme exponentiel. L'avènement d'Internet, a généralisé l'échange d'informations numérisées à travers les réseaux informatiques. Le monde est ainsi devenu en quelques années de plus en plus connecté. On estime aujourd'hui, que l'humanité produit en quelques jours autant de données qu'elle ne l'a fait en deux millions d'années. Cette production de données numériques est présente dans tous les domaines tels que dans l'industrie, la médecine, l'économie, les médias (photo, audio, vidéo), les réseaux sociaux… À cela s'ajoute le développement des capteurs numériques servant à connecter des outils, des machines ou les objets du quotidien à Internet. Cette folle expansion, devrait atteindre selon les prévisions, un volume de 163 zettaoctets en 2025, soit 8 fois plus qu'actuellement. On estime par exemple que 29 téraoctets sont publiés sur Internet chaque seconde.

De nos jours, le séquençage de l'ADN est devenu un outil incontournable et a révolutionné des domaines tels que la biologie et la médecine. En l'espace de quatre décennies, nous avons assisté à une évolution constante des techniques de séquençage de l'ADN. Ces techniques permettent des débits de lecture de plus en plus rapides, tout en permettant la lecture de fragments d'ADN de plus en plus long. Depuis 2005, date à laquelle sont apparus sur le marché les premiers séquenceurs à haut débit, nous assistons à une production de données génomiques de plus en plus massive, pour un coût de production qui a été divisé par cent mille. Les dernières générations de séquenceurs, miniaturisés, modulables et portables [1], vont conduire à une démocratisation certaine de ces outils. Les champs d'application possibles sont considérables et peuvent être par exemple, le suivi de maladies en temps réel sur le

terrain, l'étude du microbiote intestinal, la détection de cancers ou plus généralement, la médecine personnalisée.

Outre l'émergence de ces nouveaux questionnements biologiques [2], le stockage et les besoins d'analyse rapide de cette masse d'informations reste un enjeu majeur pour les années à venir. En informatique, ces dernières années, l'accélération des calculs a été rendue possible grâce à la parallélisation des algorithmes, la multiplication des unités de calculs des processeurs (cœurs) ou le calcul générique à base de processeurs graphiques (GPU). Cependant, la production exponentielle des données, évolue plus rapidement que les capacités de calcul et d'analyse de celles-ci. En bio-informatique, l'un des enjeux est donc de créer des algorithmes permettant d'identifier rapidement les génomes de référence les plus proches, d'une séquence nouvellement séquencée. Ceci afin d'effectuer un rapide prétraitement permettant de conserver uniquement des séquences de références pertinentes pour qu'elles soient par la suite analysées par des outils plus spécifiques aux questionnements biologiques.

Parallèlement, avec le développement d'Internet, l'identification de documents (audio et image) au travers de bases de données de référence est un domaine qui a aussi connu une explosion des volumes de données à analyser et qui fait l'objet de nombreuses recherches depuis plusieurs années [3]. Les principales méthodes se basent sur des concepts de calculs d'empreintes, et plus particulièrement de hachage perceptuel [4]. À partir d'un document (audio ou image), plusieurs clés sont calculées à l'aide d'une fonction de hachage perceptuel. Les clés de hachage sont alors uniques, ont une taille très inférieure aux documents d'origine mais restent néanmoins représentatives des données fournies en entrée. Ces clés peuvent alors être comparées entre elles. Ceci permet de constituer des bases de données de références, les clés de hachage servant d'indexes, pour permettre la recherche et l'identification de document requête.

Contributions de cette thèse

La littérature propose un nombre important de méthodes permettant de comparer des séquences nucléotidiques. Ces méthodes utilisent des techniques de comparaison de textes dérivées en majorité de l'algorithme de Boyer–Moore [5], d'alignement local [6], d'indexation de k-mers [7], d'utilisation d'arbres des suffixes [8] comme structure de données. Bien que ces méthodes soient reconnues comme étant très efficaces, leur utilisation peut être limitée de par leurs complexités algorithmiques, ce qui peut avoir pour conséquence de rendre difficile le changement d'échelle en termes de taille des données dans les années à venir. La littérature décrit aussi l'utilisation de table de hachage [9], permettant la recherche exacte et l'alignement de séquences ADN. Le processus d'indexation, nécessite le découpage en sous-séquences (k-mer), des séquences à indexer et pour chacun d'entre eux de conserver leurs positions sur la séquence dont ils sont issus. Ceci peut donc s'avérer coûteux en espace de stockage.

Une autre méthode est aussi apparue en 2012. MC Saldías *et al*. [10], ont pu démontrer un réel potentiel concernant l'utilisation de méthodes issues de la comparaison d'images, pour aligner des séquences ADN. L'article décrit le processus d'encodage des séquences ADN sous forme d'images 2D puis l'utilisation d'une méthode de corrélation de phase d'une Transformée de Fourier Discrète (TFD). Bien que les temps de réalisation des processus d'alignement apparaissent nettement plus lents que l'outil de référence en bio-informatique BLAST, cet article démontre un certain intérêt quant à l'utilisation de méthodes permettant d'effectuer de la comparaison d'images, afin de comparer des séquences nucléotidiques ADN ou ARN. L'approche méthodologique utilisée se révèle être proche des méthodes de hachage perceptuel, car elle se base sur une fonction de traitement du signal telle que la TFD, qui permet de déterminer les fréquences significatives d'une image et ainsi de les utiliser comme identifiants discriminants.

Au cours de cette thèse, nous nous sommes plus particulièrement intéressés au concept de hachage perceptuel, utilisé habituellement pour indexer et comparer des images numériques, afin de déterminer si de telles méthodes sont pertinentes pour comparer des séquences exactes ou approchées au sein de base de données de séquences de références. Ainsi, nous

proposons deux contributions. La première est une fonction de hachage perceptuel, permettant l'indexation de séquences ADN/ARN. Outre une diminution importante des données indexées par rapport aux séquences fournies en entrée, cette fonction de hachage a la particularité de conserver la propriété de comparabilité entre deux clés de hachages. Deux séquences ADN/ARN proches, auront des clés de hachage également proches et ainsi comparables. La seconde contribution, est l'adaptation d'une méthode permettant de faire ressortir les zones communes entre deux images, à la problématique de la comparaison de séquences ADN.

Ces travaux se placent dans un contexte d'accroissement des volumes de données génomiques, où l'enjeu est de concevoir des algorithmes permettant d'identifier rapidement les génomes de référence les plus proches d'une séquence requête. Le but étant d'effectuer un prétraitement rapide, permettant de ne conserver que des séquences pertinentes et par la suite d'utiliser des méthodes plus spécifiques à la bio-informatique. C'est donc dans ce cadre que vont se focaliser les travaux décrits dans ce présent manuscrit de thèse.

Dans une première partie, nous présenterons le contexte inhérent à ce travail de thèse, ainsi que divers éléments de bibliographie dans les domaines du traitement des données génomiques, de la comparaison des images numériques ainsi que des fonctions de hachage.

Au cours de la seconde partie de ce manuscrit, nous présenterons et décrirons deux contributions :

- La fonction de hachage perceptuel PSH (Perceptual Sequence Hashing), qui permet le calcul de clés de hachage à partir de séquences ADN. Les clés de hachage ont une taille inférieure aux séquences fournies en entrée, mais conservent toutefois la propriété de pouvoir être comparées, permettant ainsi d'obtenir un indice de similarité entre les séquences ADN dont elles sont issues

- La méthode de comparaison de séquence PSC (Perceptual Sequence Correlation), qui permet de comparer deux séquences ADN de tailles différentes en faisant ressortir les zones communes entre celles-ci.

PREMIÈRE PARTIE – ÉTAT DE L'ART

Cette première partie a pour objet de présenter le contexte général ayant conduit aux travaux menés durant cette Thèse. Elle abordera tout d'abord le sujet de l'évolution des données issues du séquençage de l'ADN, puis les problématiques sous-jacentes telles que la comparaison de séquences et leur stockage au sein d'une base de données. Par la suite, un parallèle sera établi avec l'évolution des données d'images numériques et les problématiques d'indexation, de comparaison et de recherche d'images, communes aux données en génomique. Enfin, nous nous focaliserons plus particulièrement sur les méthodes d'indexation basées sur l'utilisation de fonctions de hachage et l'utilisation de structure de données telles que les systèmes de type NoSQL Clé-Valeur.

Cette première partie qui présente différents éléments de bibliographie n'a pas pour vocation d'être exhaustive. Les différents concepts, méthodes et algorithmes abordés durant cette première partie ont pour objectif de présenter au lecteur, le panorama de concepts, d'outils et de techniques ayant menés aux développements méthodologiques, qui seront décrits au cours de la seconde partie de ce présent manuscrit. Le lecteur pourra, s'il souhaite approfondir de façon détaillée tout ou partie de ces aspects, se référer aux publications citées tout au long de ce chapitre.

Le chapitre 1, présentera le contexte biologique de cette thèse. Il abordera tout d'abord, quelques notions biologiques autour de la notion de génome puis de l'ADN et de sa structure, et enfin, il présentera l'évolution des technologies de séquençage, génératrices d'un volume de données de plus en plus conséquent.

Le chapitre 2, abordera les problématiques en bio-informatique en termes de méthodologie d'analyse et des usages biologiques. Il présentera la problématique autour du stockage des données génomiques, puis se focalisera plus particulièrement sur l'état de l'art autour des méthodes permettant la comparaison et l'alignement de séquences ADN et par extension l'identification de séquences non caractérisées au sein de base de données de référence.

Le chapitre 3, présentera tout d'abord quelques notions en imagerie numérique, puis il présentera différentes méthodes permettant d'indexer puis de comparer des images au sein de système de recherche d'image par le contenu.

Le chapitre 4, présentera la notion de fonction de hachage. Il introduira tout d'abord une définition générale des fonctions de hachage ainsi que les différentes propriétés qui les caractérisent. Par la suite, il décrira les principaux types de fonctions de hachage et enfin il présentera leurs différents usages et applications.

Chapitre 1 **- CONTEXTE BIOLOGIQUE**

RÉSUMÉ : Ce premier chapitre introduit le contexte biologique, qui est le point de départ des travaux menés durant cette thèse. Après un bref rappel de quelques dates historiques sur la découverte de l'ADN et de ses premiers usages, il présente succinctement quelques concepts biologiques tels que l'ADN, l'ARN puis la notion de génome. Dans une seconde partie, il introduit la notion de taille des génomes, puis les différentes générations de séquenceurs ADN, afin d'illustrer l'évolution de la production des données génomiques.

SOMMAIRE :

Introduction

La découverte de l'ADN (Acide, DésoxyriboNucléique) a été une révolution sans précédent pour les sciences du vivant. Elle est l'aboutissement de nombreux travaux et découvertes menés au sein de plusieurs disciplines, telles que la biologie, la médecine, la physique et la chimie. En l'espace de 150 ans, il est devenu envisageable puis réalisable, de pouvoir étudier le fonctionnement des organismes vivants à l'échelle de leurs génomes, puis à l'échelle de leurs populations. Ceci a été rendu possible grâce au développement des techniques de séquençage qui, encore aujourd'hui, ne cessent d'évoluer. Parallèlement, le développement de l'informatique a aussi joué un rôle important, en tant que support, pour le développement et la mise en œuvre d'algorithmes de traitement et d'analyse en bio-statistiques et bio-informatique et aussi en tant que support de stockage des données produites par le séquençage et les différentes phases d'analyse.

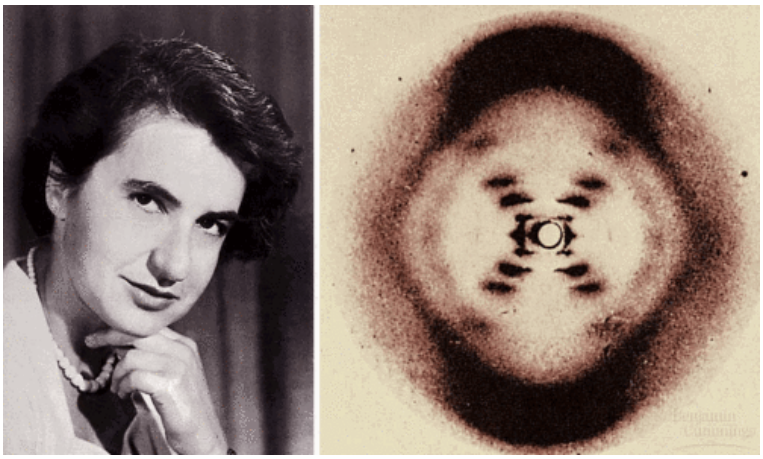
Ce premier chapitre, introduira le contexte biologique, relatif et nécessaire à la compréhension des travaux menés durant cette thèse. Ainsi, dans un premier temps, il présentera, quelques notions de base en biologie, notamment en ce qui concerne l'ADN et l'ARN ainsi que les différentes applications et utilisation possible. Par la suite il abordera les problématiques inhérentes à l'évolution des données en génomique induite par l'évolution des techniques de séquençages de l'ADN. Puis, il présentera brièvement les différentes techniques de séquençage.

1. Quelques dates importantes

La découverte de l'ADN puis le développement des techniques nécessaires à son séquençage et à son analyse sont relativement récents par rapport à l'histoire de la biologie. Nous en présentons ici les principales étapes sous forme d'une chronologie.

- **1869** : le biologiste Friedrich Miescher isole pour la première fois l'ADN, qu'il décrit comme étant une substance riche en phosphore, située à l'intérieur des cellules.

- **1898** : Richard Altmann (pathologiste et histologiste), parvient à isoler pour la première fois, l'acide nucléique
- **1896** : Albrecht Kossel (médecin) découvre les 4 bases azotés constituantes de l'ADN.
- **1928** : Les biochimistes Phoebus Levene et Walter A. Jacobs identifient les désoxyriboses. C'est à partir de cette date que l'on commence à parler d'acide désoxyribonucléique.
- **1944** : Le docteur Oswald Avery, met en évidence le caractère héréditaire de l'ADN et le mécanisme de mutation génétique des bactéries.
- **1951** : Rosalind Franklin (physico-chimiste) parvient à produire le premier cliché de l'ADN en utilisant une technique de diffraction des rayons X. Ce cliché sera capital pour la compréhension de la structure à double hélice de l'ADN.



(a) (b)

Figure 1.1 – *(a) Dr. Rosalind Franklin et (b) son cliché de l'ADN élaboré à partir d'une technique de diffraction des rayons X*

- **1953** : À partir des travaux de Rosalind Franklin, obtenus de façon détournée par l'intermédiaire de Maurice Wilkins ; James Dewey Watson (biochimiste) et Francis Crick (biologiste), établissent la structure en double hélice de l'ADN et reçurent avec M. Wilkins, le prix Nobel de physique et de médecine, le 31 octobre 1962.
- **1968** : Marshall Nirenberg, Har Gobind Khorana et Robert Holley reçurent le prix Nobel en Physiologie / médecine, pour avoir mis en évidence les mécanismes de production des protéines à partir du code génétique regroupés en 3 nucléotides, appelés codons.

- **1970** : Le terme « bioinformatique » fût employé pour la première fois dans un article de B. Hesper et Hogeweg P. [11].
- **1970** : Publication du premier algorithme d'alignement de séquences ADN ou protéiques [12].
- **1977** : Apparition des premières techniques de séquençage de l'ADN développées par A.M Maxam et W. Gilbert [13] et F. Sanger et A.R. Coulson [14].
- **1978** : Séquençage du premier organisme vivant (virus phiX174) par les équipes du Pr. F. Sanger [15]. Le virus phiX174 est un organisme dont l'ADN est constitué d'un seul brin circulaire d'une longueur de 5386 nucléotides organisées en 11 gènes.
- **1986** : K. Mullis [16] présenta une technique permettant d'amplifier in-vitro un fragment d'ADN, permettant d'en obtenir une grande quantité de répliques. Cette technique est connue sous le nom de PCR (Polymerase Chain Reaction)
- **1990** : Lancement du projet Human Genome Project piloté par le National Institutes of Health (NIH) et avait pour objectif la réalisation du séquençage complet du génome humain ainsi que d'organismes modèles [17].
- **2001** : le projet Human Genome Project, publie la première séquence complète brute du génome humain.
- **2005** : apparition des premiers séquenceurs ADN de seconde génération, connus sous le nom de « séquençage haut-débit ».
- **2006** : première étude en métagénomique {Citation}
- **2012** : 3ème génération de séquenceurs caractérisée par leurs capacités à pouvoir séquencer une seule molécule d'ADN
- **2015** : Commercialisation du premier séquenceur ADN portatif
- **2017** : Premier séquençage de l'ADN dans l'espace à bord de l'ISS

2. Généralité sur les génomes

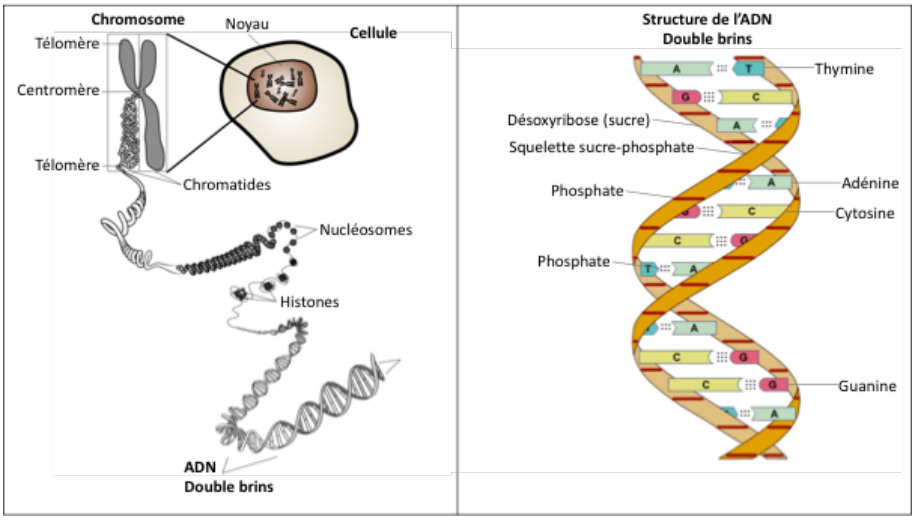
Le mot « génome » est un mot composé à partir des mots « gène » et « chromosome ». Il représente l'ensemble des informations génétiques et héréditaires d'un organisme biologique. Il est constitué d'ADN et dans certains cas, il peut aussi être constitué d'ARN, lorsque celui-ci est le matériel génétique de base d'un organisme, comme pour certains virus.

Au sein du génome, l'ADN est organisé en différentes régions, dite codantes et régions non codantes.

Les régions codantes correspondent aux gènes. Les gènes sont des séquences d'ADN définies, qui composent les chromosomes. Ils sont localisés à des endroits précis (locus) et portent l'information génétique spécifique servant à coder des protéines particulières par l'intermédiaire de l'ARN.

Les régions non codantes sont des segments intergéniques ou des introns lorsqu'elles sont situées à l'intérieur des gènes. Elles représentent la grande majorité de l'information génétique contenu dans un génome. Longtemps qualifié « d'ADN poubelle », leurs rôles est encore aujourd'hui à déterminer. Ainsi, les zones non codantes sont constituées d'éléments transposables (transposons), qui sont des séquences capables de se répliquer et de se répandre au sein du génome et ceci de façon autonome. Les zones non codantes peuvent aussi être constituées de séquences répétées.

3. Acide DésoxyriboNucléique (ADN)



(a) (b)

Figure 1.2 – *Schéma général, de la cellule (a) à la structure de l'ADN (b) (Wikipedia CC BY-SA 3.0)*

L'acide désoxyribonucléique ou ADN, est une macromolécule de plusieurs centimètres de long, qui est présente au sein des cellules de tous les organismes multicellulaires terrestre (Figure 1.2). L'ADN est constituée d'une succession d'éléments de base appelés nucléotides. Les nucléotides sont reliés les uns aux autres par l'intermédiaire d'un sucre (le désoxyribose), qui se lie à ses voisins par une liaison phosphodiester et ils forment deux brins complémentaires, se faisant face, enroulés en double hélice. Les deux brins d'ADN sont orientés dans le sens 5' vers 3' en raison de la conformation chimique des désoxyriboses. Les deux brins complémentaires sont liés entre eux de façon inversée, l'extrémité 5' du premier, étant en contact direct avec l'extrémité 3' du second. Le caractère antiparallèle des deux brins complémentaires, entraine la formation d'un grand sillon, profond et large et d'un petit sillon, étroit et peu profond. Ceci permet d'avoir accès à la séquence d'enchainement des nucléotides, sans devoir séparer les brins.

Il existe quatre types de nucléotides différents qui constituent l'ADN, l'Adénine (A), la Cytosine (C), la Guanine (G) et la Thymine (T). Au sein de la double hélice de l'ADN, les nucléotides s'appairent en fonction de leur type, par des liaisons hydrogène. Ainsi, il existe deux liaisons hydrogène entre l'Adénine et la Thymine et trois liaisons entre la Cytosine et la Guanine.

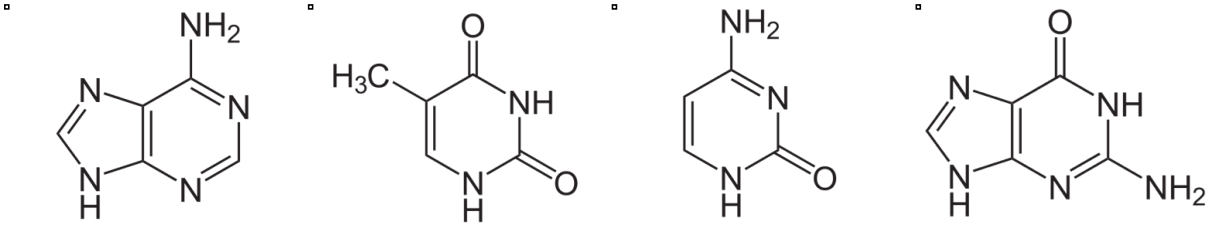


Figure 1.3 – *Structure chimique des quatre bases azotées, composante de base de l'ADN*

Cette succession de nucléotides peut paraître, à première vue, comme étant aléatoire, cependant, elle se révèle au contraire, avoir une organisation structurée et organisée en différentes unités appelées gènes. Les différents gènes composent le génome et sont porteurs de l'information génétique de tous les êtres vivants. L'ADN détermine la synthèse des protéines, par l'intermédiaire de l'ARN.

Au sein des organismes procaryotes, qui sont généralement des organismes monocellulaires tels que les bactéries, l'ADN se situe dans le cytoplasme de la cellule. Chez les eucaryotes (les animaux, champignons, plantes ou protozoaires), l'ADN, se trouve au sein du noyau, sous forme linéaire et scindé en plusieurs unités, qui forment les chromosomes. Une petite partie de l'ADN se trouve aussi dans la matrice des mitochondries ainsi que dans les chloroplastes. Il peut apparaître sous différentes formes et peut ainsi être linéaire, ramifié ou circulaire.
L'ADN est le support du patrimoine génétique de tout être vivant. L'ensemble des informations génétiques constitue le génotype qui, lors de son expression constitue le phénotype, c'est à dire, l'ensemble des caractéristiques physiques d'un individu, mais aussi plus généralement de son espèce.

Malgré les liaisons chimiques fortes et la complémentarité des nucléotides, l'ADN n'est pas une molécule dotée d'une stabilité à toute épreuve. En effet, même si ces mécanismes permettent d'assurer la stabilité du code génétique, durant le processus de réplication des cellules, des modifications accidentelles peuvent altérer l'information génétique et créer différentes formes d'un même gène (allèles). On parle ainsi de mutation si les modifications se font sur quelques nucléotides. Les mutations peuvent survenir de façon spontanée au cours des phases de réplication, via des erreurs d'appariement, ce qui est le mécanisme de base des processus d'évolution des espèces. Mais elles peuvent aussi apparaître à la suite d'événements, tels que l'échange avec l'ADN d'un autre organisme, on parle alors de recombinaison génétique. La recombinaison génétique peut se faire par transmission génétique entre bactéries, mais aussi via des techniques de génie génétique (techniques d'OGM, CRISPR-CAS9). L'ADN peut aussi être modifiée par des évènements ou des conditions extérieures aux organismes, tels l'exposition à la radioactivité, à certains types d'ultraviolets ou à certains produits chimiques. Il existe 3 types de modification possibles de l'ADN :

La substitution, lorsqu'une ou plusieurs paires de bases sont remplacées par d'autre paires de base d'un autre type.

Figure 1.4 – *Exemple de substitutions de 5 nucléotides au sein d'une séquence d'ADN*

L'insertion, lorsqu'une ou plusieurs paires de bases sont insérées au sein d'une séquence d'ADN.

Figure 1.5 – *Exemple d'insertion de 6 nucléotides au sein d'une séquence d'ADN*

La suppression, lorsqu'une ou plusieurs paires de bases sont supprimées au sein d'une séquence.

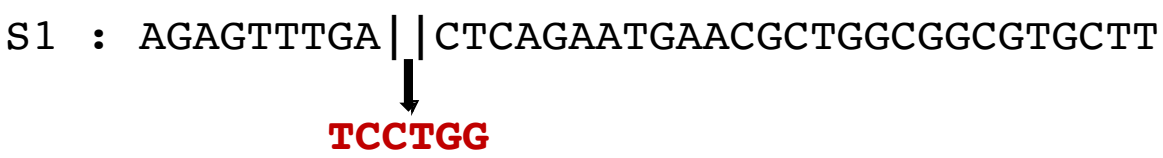


Figure 1.6 – *Exemple de suppression de 6 nucléotides au sein d'une séquence d'ADN*

4. Acide RiboNucléique (ARN)

L'Acide RiboNucléique (ARN) est un polymère linéaire qui possède une structure chimique proche de l'ADN. Ainsi, tout comme l'ADN, l'ARN est constitué d'un enchaînement de

nucléotides, qui sont reliées par des liaisons phosphodiesters et dont l'ordre d'enchaînement est directement dicté par l'ordre d'enchainement des nucléotides de l'ADN. Contrairement à l'ADN qui possède généralement une structure en double hélices complémentaires, l'ARN est dit « simple brin », à l'exception de certains virus qui peuvent être « double brins ». Il peut adopter différents types de conformation qui sont directement liés à sa fonction. Les brins sont constitués de 4 types de nucléotides, cependant à la différence de l'ADN, l'Uracile remplace la Thymine.

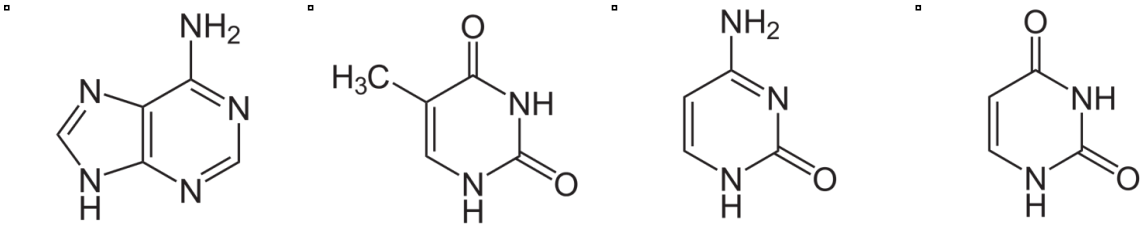


Figure 1.7 – *Structure chimique des quatre bases azotées, composantes de base de l'ARN*

L'ARN est issu de l'ADN par un complexe enzymatique appelé l'ARN polymérase. Celui-ci génère l'ARN par copie de certaines régions codantes de l'ADN. L'ARN intervient dans différents processus chimiques au sein de la cellule. Ainsi, l'ARN Messager est utilisé en tant que support intermédiaire par les cellules afin de synthétiser les protéines. Ce processus appelé « transcription » permet, à partir de triplets de nucléotides appelés « codons », de coder des acides aminés constituant les protéines. L'ARN Ribosomique permet de transformer les ARN Messagers en protéine grâce à des complexes enzymatiques appelés ribosomes. Les ARN de Transfert sont de petites séquences d'ARN qui interviennent durant le processus de synthèse des protéines.

Une hypothèse émise [18] permet de penser qu'au niveau de l'échelle de l'évolution des espèces, l'ARN serait plus ancien que l'ADN. Elle se base sur le nombre de fonctions exercées par l'ARN comparativement à l'ADN et part du principe que l'ADN ne serait apparu que dans un second temps, en tant que support de stockage, étant donné sa constitution redondante en double hélice.

5. Taille des génomes

Le concept de taille des génomes, a été utilisé pour la première fois en 1969 [19]. Bien que la notion de génome ait un caractère universel chez tous les êtres vivants, la quantité d'informations génétiques qu'il contient, varie en fonction des espèces. Même si actuellement il n'est pas possible de déterminer précisément le rôle exact de chaque portion d'un génome, la complexité d'un organisme n'est pas corrélée à la taille de son génome. La taille d'un génome peut être estimée, soit par sa masse, exprimée en picogramme (pg), soit en fonction du nombre de nucléotides qui le compose. L'unité qui permet d'estimer le nombre de nucléotides composant un génome ou une séquence ADN/ARN, est la paire de bases (pb). Les paires de base AT et CG, ayant des masses moléculaires proches, il est possible d'établir une relation entre les deux modes de calcul, 1 picogramme étant équivalant à 978 paires de base.

Génome :	**Taille :**	**Supports de stockage :**	**Capacité :**
Grippe	0,013 Mpb	Disquette 3,5'	1,4 Mo
Escherichia coli	4,64 Mpb	Compact Disk (CD)	700 Mo
Humain	3,2 Gpb	Digital Versatil Disk (DVD)	4,7 Go
Blé	17 Gpb	Disque dur :	500 Go à 8 To
Paris Japonica	150 Gpb		
Polychaos Dubium	675 Gpb		

Tableau 1.1 – *Exemples de tailles de génomes face aux principaux supports de stockage informatique, avec comme convention d'encodage : 1 paire de base (pb) = 1 octet*

6. Séquençage de l'ADN

Le séquençage de l'ADN est un ensemble de techniques visant à déterminer l'ordre d'enchaînement des quatre types de nucléotides d'une molécule d'ADN. L'objectif est de pouvoir convertir et de stocker les résultats de cette lecture au format numérique, afin de les traiter et de les analyser à l'aide de méthodes statistiques et d'algorithmes informatiques. Développée en 1977, par l'équipe du Pr. F. Sanger [15], soit 24 ans après la découverte de la structure de l'ADN par Watson et Crick, la première technique de séquençage réalisée en

laboratoire biologique a incontestablement révolutionné l'étude des êtres vivants, en permettant de travailler à l'échelle de leurs génomes. Par la suite, en 2005 [20], une seconde génération de techniques de séquençage est apparue sur le marché. Ces techniques se sont révélées beaucoup plus rapides mais aussi de fait, beaucoup moins onéreuses que les précédentes. Elles ont ouvert la voie au séquençage dit à haut débit ou « next-generation sequencing ». Ainsi, les successions de générations de séquenceurs à haut débit n'ont cessées d'accélérer les débits de séquençage avec des taux de précisions de plus en plus hauts et des coûts de plus en plus bas. À partir de 2012, une 3ème génération a commencé à émerger. Cette 3ème génération, dont les techniques sont encore en cours de développement, tente de réunir les avantages des deux précédentes, à savoir, la capacité de lire sans amplification des fragments de molécule d'ADN de plus en plus longs, dans un temps de séquençage raccourci, tout en ayant une haute résolution et précision de lecture. Cette 3ème génération a aussi vu apparaître des séquenceurs ADN plus modulables et portatifs, de la taille d'un disque dur externe et pouvant être utilisés sur le terrain. Cette évolution exponentielle des technologies de séquençage [21] qui se révèlent toujours plus performantes, moins chère et plus rapides, suit un modèle comparable, bien que plus rapide, au développement de l'informatique, communément représentée par la loi de Gordon E. Moore [22]

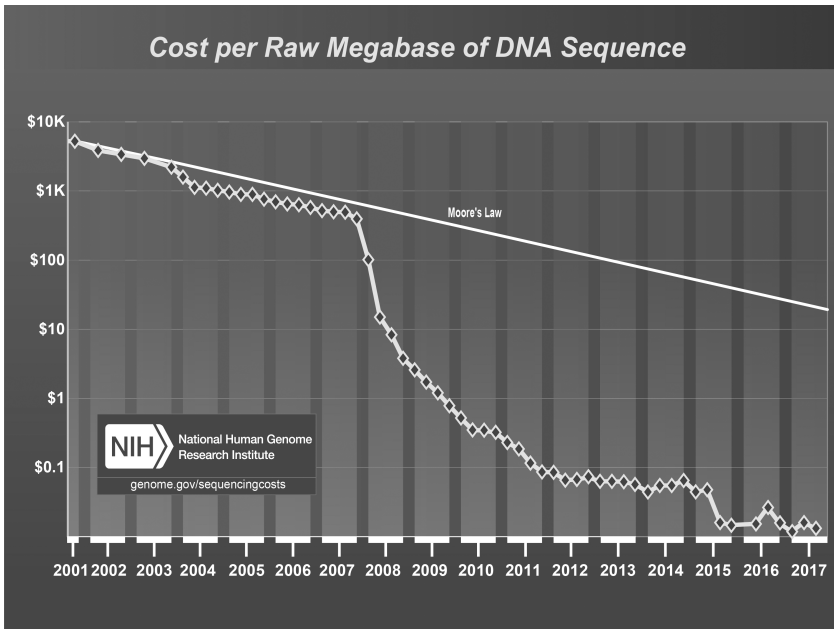


Figure 1.8 – *Évolution du coût du séquençage, pour 1 million de paires de bases durant la période 2001 – 2017*

(Source NIH - https://www.genome.gov/27541954/dna-sequencing-costs-data/)

Le séquençage de l'ADN a été une révolution sans précédent pour les sciences du vivant. Les nombreuses méthodes et techniques en perpétuelle évolution ont permis le développement de tout un champ d'applications dans les domaines de la recherche, mais aussi en médecine. La possibilité d'observer les organismes vivants à une échelle moléculaire a ainsi ouvert des perspectives infinies. Ainsi, il a donc été possible d'étudier les molécules d'ADN en tant que support de l'hérédité génétique, puis d'identifier et de cartographier l'ensemble des gènes d'un individu et leurs mécanismes d'expression. Ceci permet donc d'étudier les mécanismes de maladies génétiques, d'établir des arbres phylogénétiques permettant d'étudier l'évolution des espèces, ou de déterminer la diversité biologique d'un échantillon.

6.1. 1ère génération de séquenceurs ADN

Les premières méthodes de séquençage de l'ADN ont été développé à la fin des années 70. Deux méthodes totalement indépendantes furent créées parallèlement. La méthode de Sanger, était basée sur une approche de synthèse enzymatique, tandis que celle de Maxam et Gilbert était basée sur des techniques de dégradations chimiques. Pour ces travaux concernant ces deux techniques, Sanger, Maxam et Gilbert obtinrent le Prix Nobel de Chimie en 1980. Cependant, la méthode de Sanger, devint rapidement la plus utilisée. Elle se démarqua rapidement par rapport à sa concurrente car celle-ci nécessitait l'utilisation de produits toxiques et radioactifs et qu'il lui était difficile d'effectuer un passage à une échelle de production industrielle. Le premier séquenceur, l'AB370A a été lancé en 1986 par la société Applied Biosystems. Il pouvait séquencer 96 échantillons en parallèle, pour un débit de 500 kilo bases par jour et générer des reads de 600 paires de base.

En l'espace de quelques années, l'automatisation des techniques a permis d'augmenter considérablement les vitesses de séquençage, notamment en remplaçant l'utilisation de gel de polyacrylamide par des nanotubes capillaires. Ces évolutions ont fait émerger la première génération de séquenceurs et permis d'envisager la réalisation du premier séquençage de l'ADN d'un être humain avec un coût de 3 milliards de dollar et 10 années de travail.

6.1.1. Méthode de Sanger

Développée en 1977, la méthode de Sanger, aussi connue en tant que méthode de « séquençage par terminaison de chaîne » ou « séquençage dideoxy » resta la principale méthode de séquençage durant plus d'une vingtaine d'années. Elle se base sur des mécanismes de synthèse enzymatique de l'ADN permettant l'amplification des échantillons par clonage. Ainsi, l'ADN polymérase à la faculté de pouvoir synthétiser un brin complémentaire à partir d'un brin matrice en présence de dNTPs (Deoxynucleotide Triphosphates) et il est ainsi possible de révéler et de lire le contenu d'une séquence sur un gel de polyacrylamide, à l'aide de marqueurs spécifiques, par fluorescence.

La méthode de Sanger permit le premier séquençage du génome complet d'un organisme vivant, en l'occurrence, le virus phage phiX174. Ce virus avait été sélectionné pour la simplicité de son génome, composé d'une molécule d'ADN simple brin circulaire, avec seulement 11 gènes et une taille de 5386 nucléotides. En démontrant, qu'il était possible de lire les brins d'ADN, ce travail, réalisé en quelque mois, fût l'accomplissement de plusieurs décennies de recherche, depuis la découverte de l'ADN et de sa structure. Ces travaux ont donc posé les bases méthodologiques, permettant d'étudier les organismes vivants à l'échelle de leurs génomes. Ceci a donc ouvert la voie à de nombreuses applications dans tous les domaines des sciences du vivant.

6.2. 2ème génération de séquenceurs ADN

La seconde génération de séquenceurs a été introduite sur le marché à partir de l'année 2005. Par rapport à la génération précédente, elle permit d'augmenter considérablement les capacités de séquençage, via la miniaturisation et l'automatisation de nombreux procédés et le traitement des échantillons en parallèle, permettant ainsi le traitement de plusieurs dizaines de milliers de séquences simultanément. Les débits apportés par cette nouvelle génération de séquenceurs ont été si importants, que cette génération a été qualifiée de « séquenceur à haut débit ». Parmi cette seconde génération de séquenceurs, il existe deux

technologies majoritaires, la technologie « Sequencing by Synthesis » (SBS) de la société Illumina et la technologie « Sequencing By Ligation » (SBL).

La sortie, sur un marché dynamique et concurrentiel, de séquenceurs toujours plus performants a permis entre 2001 et 2017 de diminuer les coûts de séquençage par cent mille. Parmi les références du secteur nous pouvons citer les entreprises Roche, Illumina, ou Life Technologies.

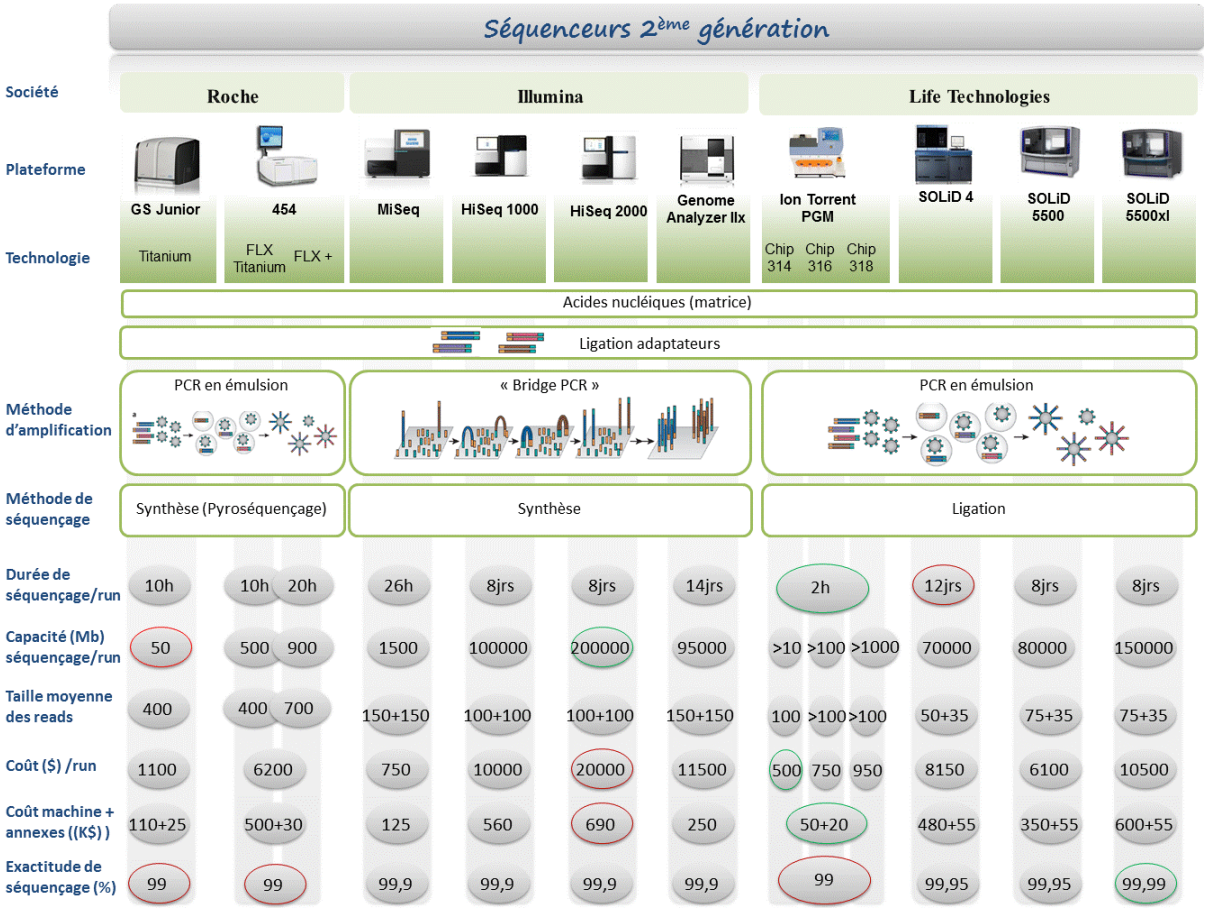


Figure 1.9 – *Comparaison des principales plateformes de séquençage de 2nde génération (http://biochimej.univ-angers.fr/Page2/COURS/9ModulGenFoncVeg/5MethEtudGenFonc/1MethodeSEQUENGAGE/1SEQUENGAGE.htm)*

Ces entreprises ont développé différentes techniques qui néanmoins nécessitent toutefois l'amplification de chaque brin d'ADN à séquencer, en parallèle dans des milieux réactionnels

séparés (amplification clonale), via des méthodes de « polymerase chain reaction » (PCR en émulsion ou Bridge PCR) et des alternances de phase de lecture, via différentes méthodes de séquençage (pyroséquençage, ligation) et de nettoyage des chambres de réactions (Scan and Wash).

En termes d'analyse, cette seconde génération de séquenceurs d'ADN a permis de travailler, non plus à l'échelle des populations d'individus mais à l'échelle de l'individu au sein de sa population. Elle a aussi permis le développement de la Métagénomique, qui consiste à séquencer l'ensemble des individus d'un écosystème ou d'un environnement. Ceci peut-être par exemple l'étude de la diversité microbienne d'un échantillon d'eau de mer ou d'un prélèvement d'un échantillon intestinal.

6.3. 3ème génération de séquenceurs ADN

La 3ème génération de séquenceurs, a été introduite sur le marché à partir de l'année 2008. Elle est connue sous l'acronyme SMS (Single Molécule Sequencing) car, par rapport aux précédentes générations, elle se distingue principalement par le fait que le séquençage d'une seule molécule d'ADN peut désormais être réalisé, sans étape de pré-amplification et que les reads sont plus longs (plusieurs dizaines de paires de base par run). De plus, par rapport aux séquenceurs des génération précédentes, la lecture des séquences se fait en temps réel, c'est à dire que les étapes de lecture des bases nucléiques et la constitution des séquences sont réalisées simultanément. Ce qui distingue aussi cette 3ème génération des précédentes est l'émergence d'une grande diversité des méthodes, permettant de réaliser du séquençage SMS. Parmi ces méthodes de séquençage, nous pouvons citer la technologie Single Molecule Real-Time (SMRT) développée par la société Pacific Biosystems (PacBio) [23] ou la technologie de séquençage par Nanopore développée par Oxford Nanopore Techniques (ONT) [24].

Les séquenceurs de 3ème génération, n'ont toutefois pas encore supplanté leurs homologues des génération précédentes. Ils connaissent encore des évolutions, car les taux d'erreurs de lecture sont parfois encore élevés (>10%). Ceci peut par conséquent entrainer des difficultés au niveau des étapes d'analyse, notamment durant les processus d'assemblage, de

transcriptomiques ou les phases de comparaison avec des génomes de référence. Il est donc encore courant de combiner l'utilisation de techniques de séquençage de 2ème et de 3ème génération afin de tirer parti des avantages de chacune d'entre elles.

6.3.1. Séquençage par synthèse de l'ADN polymérase :

Le séquençage par synthèse de l'ADN polymérase est issu des méthodes traditionnelles de première génération dont il reprend les principes avec l'utilisation de désoxynucléotides fluorescents et d'ADN polymérases. Le brin d'ADN à séquencer, est au préalablement découpé en différents fragments, à l'aide d'enzymes de restriction, puis une queue poly A est ajoutée à chaque fragment. Par la suite, les différents brins sont hybridés à la surface d'une *Flow Cell* qui dispose de milliards de chaînes poly T sur sa surface. Le séquençage des brins se fait par ajouts successifs des bases A, T, G, C fluorescentes, à partir du site de capture poly T. La fluorescence est ensuite révélée par un laser qui balaye la surface de la Flow Cell et enregistre la localisation de chaque molécule fluorescente incorporée via l'utilisation d'un capteur CCD.

6.3.2. Séquençage via l'utilisation de Nanopores :

Développé à partir de 1998 [Church, et al., 1998] [25], le séquençage à base de nanopores est une technologie de séquençage d'ADN qui est apparue sur le marché à partir de l'année 2012. Le principe de cette technique est de faire passer un brin d'ADN au travers d'un nanopore traversé par un courant électrique ayant une très faible tension. Un nanopore est un puits, semi-conducteur en silicium, qui a une taille d'une dizaine de nanomètres et qui est recouvert par une membrane plongée dans une solution de chlorure de potassium. Ainsi, la tension créée par le déplacement des ions, induite par le passage d'une base nucléique au travers de la membrane, va générer une intensité différente et quantifiable en fonction du type de celle-ci.

Par rapport aux autres technologies de séquençage, le séquençage par Nanopore se distingue par le fait qu'elle ne nécessite qu'une quantité très faible d'échantillons biologiques, demande peu de préparation des échantillons à séquencer, limite l'utilisation de réactifs chimiques et a

des coûts d'exploitation limités. De ce fait, l'utilisation de séquenceurs exploitants cette technologie n'a pas besoin d'être réalisé spécifiquement au sein d'un laboratoire biologique. De plus, cette technologie peut être miniaturisée. La société Oxforde Nanopore Technologies commercialise parmi ses références produits, des séquenceurs ADN portatifs, de la taille d'un disque dur externe et projette de commercialiser un séquenceur connectable à un smartphone. Cette miniaturisation ouvre d'énormes perspectives, quant à l'utilisation d'une telle technologie qui a déjà permis le premier séquençage ADN dans l'espace, au sein de l'ISS [26] ou le suivi en temps réel de l'évolution du virus Ébola en Afrique de l'Ouest [27]. En janvier 2018, M. Jain et al. [28] ont réalisé le premier séquençage et l'assemblage complet du génome humain, à partir d'un séquenceur nanopore avec un taux de couverture de 30x et des reads ayant des tailles comprises entre 100kb et 880kb.

Figure 1.10 – *Séquenceur MinION commercialisé par la société Oxford Nanopore Technologies*

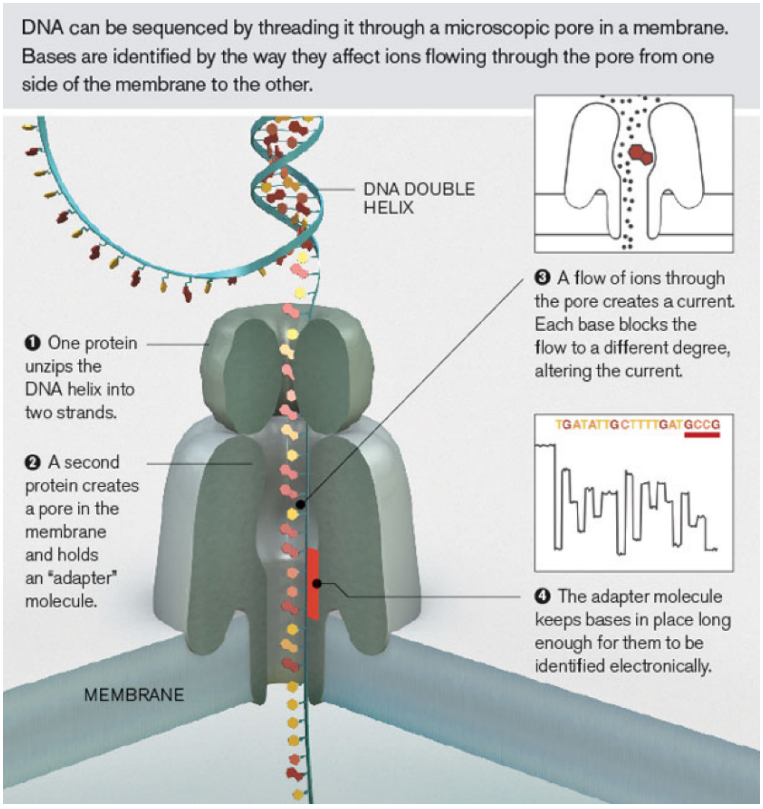


Figure 1.11 – *Principe de fonctionnement de la technologie de séquençage de l'ADN par Nanopore*

6.3.3. Une génération de séquenceurs toujours en développement

Les techniques actuelles de séquençage de l'ADN ont connu une progression significative depuis une dizaine d'années. Elles ont permis d'augmenter considérablement la rapidité de séquençage, de séquencer des fragments d'ADN de plus en plus longs, pour un coût de production de plus en plus faible. Cependant, cette 3ème génération poursuit toujours son développement à un rythme soutenu. Ainsi, de nombreuses sociétés telles que Stratos Genomics, NobleGen ou Halcyon Molecular ont vu le jour afin de développer de nouvelles technologies de séquençage ou d'améliorer des techniques existantes. Ainsi, la société Stratos Genomics, travaille sur une évolution du séquençage par nanopore appelée Sequencing By eXpansion (SBX), où les molécules d'ADN sont converties en molécules plus grosses, facilitant ainsi la lecture via nanopore.

Avec les évolutions récentes en microscopie, le séquençage de l'ADN à l'aide de microscope électronique est une technique qui poursuit aussi son développement. Son principe général est de pouvoir observer directement les molécules d'ADN, à l'aide d'un microscope électronique en transmission (MET). Ainsi, chaque base nucléique du brin d'ADN est marquée à l'aide d'atomes lourds rendant de ce fait observable chaque nucléotide.

Les enjeux autours de ces technologies sont colossaux, notamment concernant les applications autours de la santé. La miniaturisation des séquenceurs pourrait permettre de mettre les analyses génétiques à la portée d'un grand nombre de professionnels de la santé. Mais l'utilisation du séquençage de l'ADN pourrait aussi avoir une importance capitale en informatique, les molécules d'ADN pouvant servir à stocker efficacement des données [29], avec des densités de stockage estimées à 215 Po par gramme d'ADN.

Conclusion

En l'espace de quatre décennies, le séquençage de l'ADN est devenu un outil incontournable et a révolutionné l'étude des organismes vivants. Les dernières générations de séquenceurs de plus en plus miniaturisés, modulables et portables, vont conduire à une démocratisation certaine de ces outils.

Ce premier chapitre a abordé le contexte biologique inhérent aux travaux menés durant cette thèse. Il a présenté succinctement, quelques notions biologiques autour de l'ADN et de sa structure, ainsi que les principales étapes ayant conduits à sa découverte par la communauté scientifique. Dans une seconde partie, il a introduit la notion de taille des génomes en l'illustrant avec différentes tailles de génomes représentatives d'organismes vivants. Enfin, dans une dernière partie, il a présenté la notion de séquençage de l'ADN, puis les principales techniques de séquençage et les différentes générations de séquenceurs.

L'évolution des techniques de séquençage de l'ADN a ouverte de nombreuses perspectives quant à l'identification et l'étude des organismes vivants. Les possibilités d'analyse des génomes entraînent encore aujourd'hui de nouveaux questionnements biologiques et

promettent de nombreuses perspectives en médecine. Cependant, la gestion de cette masse de données produite avec un rythme de plus en plus soutenu, reste un enjeu majeur pour les années à venir. La gestion de cette masse de données entraîne toujours de nombreuses problématiques, notamment au niveau du son stockage, de son partage, mais aussi de son analyse.

POINTS CLÉS :

- La découverte de l'ADN, puis la possibilité de son séquençage ont permis l'étude des êtres vivants ou des populations à une échelle moléculaire.
- Le séquençage de l'ADN a entraîné de nouveaux questionnements biologiques.
- L'évolution toujours constante des techniques de séquençage va conduire à une démocratisation de ces outils.
- Compte tenu de la taille des génomes des organismes vivants, les sciences du numérique vont devoir s'adapter afin de pouvoir répondre à l'arrivée d'une masse de données de plus en plus importante en termes de stockage et d'analyse.

Chapitre 2 - PROBLÉMATIQUES AUTOUR DE L'ANALYSE DES DONNÉES GÉNOMIQUES

RÉSUMÉ : Ce second chapitre, aborde la problématique de l'analyse des données génomiques sous différents aspects. Dans un premier temps, il introduira les problématiques biologiques dans le domaine de la génomique, puis il traitera des questions autour de l'analyse des données en bio-informatique telles que l'évolution de la taille des données générées, le stockage des données sous forme de bases de données, les différents formats de fichiers, permettant le stockage et l'échange des données et les méthodes de compression permettant de diminuer la taille des données. Enfin, dans une seconde partie, il présentera les problématiques générales de la comparaison de séquences ADN et par extension, la recherche de séquences proches au sein de bases de données. Puis il dressera plus particulièrement un panorama des méthodes de comparaison de séquences ADN.

SOMMAIRE :

Introduction

L'évolution constante des techniques de séquençage de l'ADN, qui est toujours en plein essor depuis une quinzaine d'années, a eu pour conséquence de provoquer l'émergence de nouveaux questionnements biologiques. La gestion et l'analyse de cette masse de données en constante évolution, reste un enjeu majeur pour les années à venir. Ainsi, la production des données génomiques dont les coûts ont, en l'espace d'une quinzaine d'années été divisé par cent mille, reste un challenge majeur. Il est tout d'abord nécessaire de stocker des volumes de données de plus en plus importants, avec comme contraintes que ces données doivent être organisées de façon à pouvoir être rapidement accessibles. Durant le processus d'analyse des données génomiques, la recherche de sous-séquences, au travers de bases de données de génomes de référence, est une tâche incontournable. Au cours du processus d'analyse des données issues du séquençage de l'ADN, l'une des premières étapes, est de caractériser les séquences nouvellement séquencées, ce qui permet de déterminer les génomes de référence les plus proches de celles-ci. Cette phase d'identification nécessite généralement l'utilisation de bases de données constituées de séquences de référence. Elle est notamment nécessaire, au cours des phases d'assemblage permettant de reconstituer une séquence consensus à partir de plus petites, durant les phases d'annotation (pour déterminer les fonctions biologiques d'un chromosome), pour détecter des zones de mutations sur un génome, pour effectuer de la classification d'espèce (phylogénie) ou pour identifier les différentes espèces biologiques issues d'un échantillon (métagénomique).

1. Analyse des données génomiques

Le développement des techniques de séquençage de l'ADN a incontestablement été une importante révolution et a initié une nouvelle discipline appelée génomique. La génomique est apparue en 1972 avec le premier séquençage d'un organisme vivant, mais elle ne pris son essors qu'à partir des années 2000, avec la publication du premier génome humain. Elle a pour objet d'étudier les organismes des êtres vivants à une échelle moléculaire. Elle permet d'établir la cartographie des différents gènes (génomique structurelle) et de décrypter leurs mécanismes d'expression et d'hérédité (génomique fonctionnelle). L'études des différents

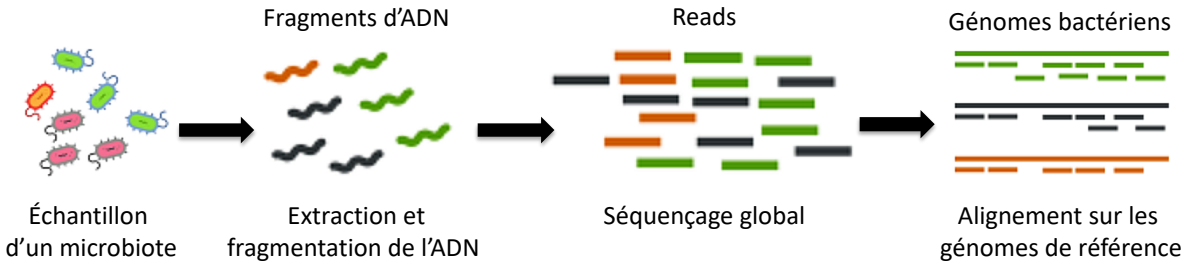
mécanismes liés aux génomes a permis de nombreuses applications. Ainsi, dans le domaine de la médecine la génomique permet l'étude et le diagnostic de nombreuses maladies génétiques, ou de mieux comprendre les mécanismes de mutation de l'ADN conduisant à la cancérisation d'une cellule.

La génomique permet aussi d'étudier les relations de parenté entre des groupes d'organismes (espèce ou population), pour en déduire les mécanismes d'évolution des espèces (phylogénie), afin de définir l'identification et la dénomination des organismes (taxonomie). La phylogénie permet de définir des degrés de parenté entre différentes espèces ou entre plusieurs individus d'une même espèce et ceci avec une notion de temporalité. L'objectif est de créer des arbres phylogénétiques permettant de représenter l'évolution d'un groupe d'individu ou d'espèces à travers le temps afin de reconstituer l'histoire de l'évolution taxonomique. L'élaboration des études en phylogénie nécessite l'utilisation d'algorithmes de parcimonie, de vraisemblance maximale ou d'inférence bayésienne fondée. Actuellement les méthodes phylogéniques se basent sur la comparaison de certaines portions du génome des espèces telles que l'ARN 16S. L'ARN 16S est une séquence d'ARN ribosomique. Présente chez la plupart des espèces, elle est utilisée comme marqueur phylogénique, car elle permet de les caractériser. De plus elle a une vitesse d'évolution lente, ce qui permet d'établir des divergences génétiques entre espèces.

Outre les études en génomique menées à l'échelle des individus, l'évolution des techniques de séquençage a permis l'émergence d'étude d'un organisme au sein de son écosystème. Ainsi, la métagénomique, vise à étudier la diversité génétique d'un échantillon biologique. Contrairement à la génomique, qui base son approche sur l'isolement et la culture en laboratoire de l'organisme étudié, la méta-génomique vise à séquencer l'ADN de tous les organismes vivants, présents dans un échantillon, pour en recenser l'intégralité des organismes vivants qui le composent. Ceci représente une avancée importante, puisque 99% des bactéries ne peuvent pas être cultivées en laboratoire [30].

Lors des études en métagénomique, deux approches sont principalement utilisées. La première appelée métagénomique ciblée ou métabarcoding, consiste à amplifier la séquence partielle ou intégrale d'une portion de l'ADN ou de l'ARN d'un individu, ayant une grande

variabilité génétique, représentative de l'espèce, car quasiment identique chez des individus appartenant à la même espèce. Ainsi, la cytochrome oxydase (COI), gène mitochondrial ou l'ARN ribosomique 16S sont utilisés comme marqueurs d'identification. La seconde approche dite de métagénomique globale est basée sur le séquençage total de tous les brins d'ADN présents au sein d'un échantillon biologique. Avec la diminution des coûts liés au séquençage de l'ADN, cette approche va sans doute devenir la norme, car elle permet de mettre en évidence l'intégralité du contenu des gènes d'une population microbienne et ainsi d'étudier de nombreuses fonctions génétiques présentes au cœur d'un écosystème.



Source : http://dridk.me/metagenomique.html

Figure 2.1 – *Principales étapes d'une étude en métagénomique*

2. Évolution de la taille des bases de données

L'apparition des premières techniques de séquençage de l'ADN, a été une véritable révolution et a ouvert la voie à d'énormes perspectives en terme de connaissance du vivant. Le séquençage permit tout d'abord la lecture de séquences courtes, puis de plus en plus longues, grâce à l'amélioration des techniques. Conjointement avec le développement de l'informatique, qui allait servir de support de stockage et d'analyse, ces évolutions ont eu pour conséquence de faire émerger une nouvelle discipline : la bio-informatique des séquences. En termes de production de données, l'évolution des techniques de séquençages de l'ADN, a eu pour conséquence directe de produire de plus en plus de données génomiques. Cette production de données génomiques qui a évoluée de façon exponentielle, devrait encore s'accélérer durant les années à venir, compte tenu de la démocratisation du séquençage et

des futures perspectives offertes notamment en métagénomique et plus généralement en termes de médecine personnalisée.

Les premières bases de données génétiques furent crées dans les années 70. L'une des toutes premières bases de données fut la base de données du Los Alamos National Laboratory (LANL). Par la suite au début des années 80, la communauté scientifique souhaita organiser le partage des données de séquences ADN ou d'ARN à travers le monde et créa l'International Nucleotide Sequence Database Collaboration (INSDC). Aux États-Unis, un consortium composé du LANL, du NIH et de sociétés privées, donna naissance en 1982 à GenBank, la base de données de séquences ADN la plus connue à travers le monde. Désormais sous la tutelle du National Center for Biotechnology Information (NCBI) depuis 1988, GenBank a, en l'espace de 20 ans, vu le nombre de séquences qu'elle stocke être multiplié par un million.

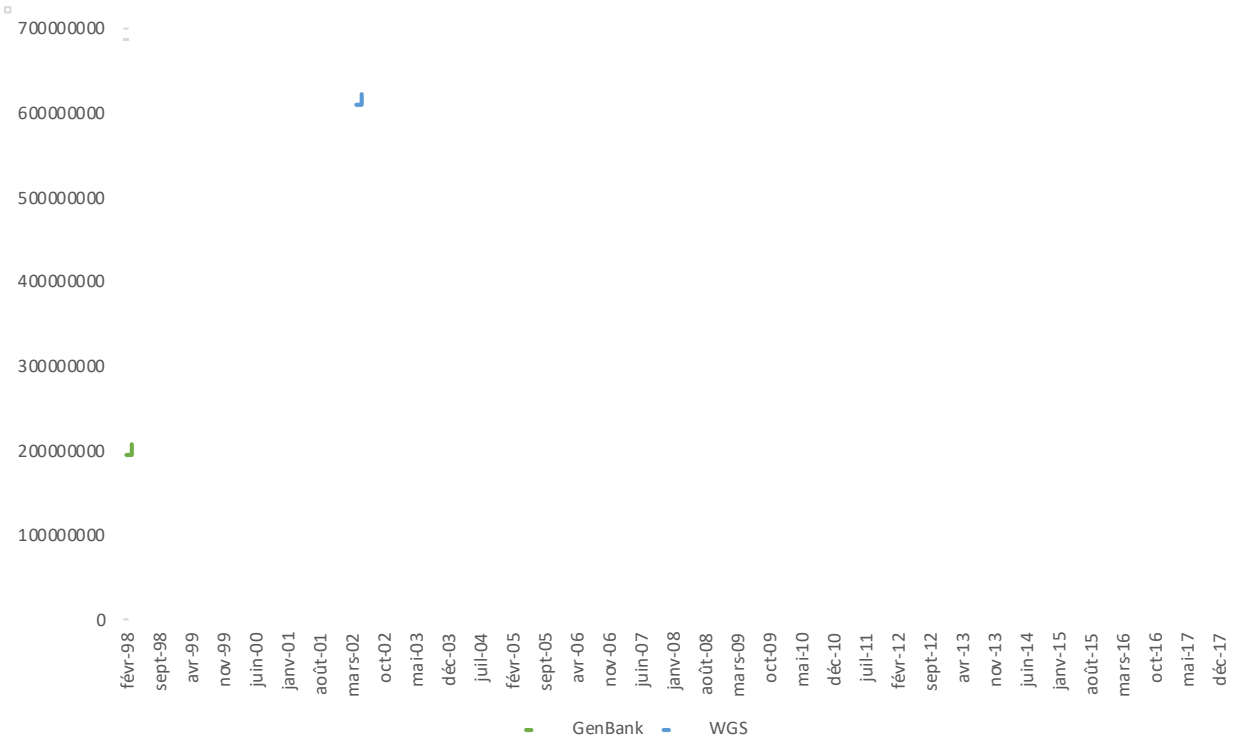


Figure 2.2 – *Évolution du nombre de séquences au sein des bases de données Genbank et WGS durant la période 1998-2017*

De façon générale, le nombre de base de données biologiques produites à travers le monde n'a cessé d'augmenter. Ainsi, d'après D. J. Rigden Al. [31], leur nombre est passé de 1 000 en 2008 à plus de 1700 en 2018, avec 273 bases de données d'ADN/ARN dédiées à la communauté scientifique. Alimentées par la communauté des biologistes à travers le monde, les bases de données à caractère scientifique ont pour caractéristique d'être librement consultables et téléchargeables. Leurs systèmes d'interrogation permettent de pouvoir rechercher des séquences d'ADN ou d'ARN en utilisant des identifiants taxonomiques et via des systèmes de recherche de séquences proches, à partir de séquences requête. Les systèmes de recherche de séquences proches se basent principalement sur des algorithmes d'alignement de séquences en ligne. Le téléchargement s'effectue sous forme de fichiers pouvant avoir plusieurs formats pour le stockage des données extraites.

Nom :	Organisme :	Description :
BioSample	European Molecular Biology Laboratory, European Bioinformatics Institute (EMBL-EBI)	Biological samples used as sources of sequence, structure or expression data
DDBJ	Center for Information Biology and DNA Data Bank of Japan	All known nucleotide and protein sequences
EBI patent sequences	EMBL-EBI	Non-redundant databases of patent DNA and protein sequences
European Genome-phenome Archive (EGA)	EMBL-EBI	Permanent repository for all types of potentially identifiable genetic and phenotypic data from biomedical research projects.
European Nucleotide Archive	EMBL-EBI	Repository for public nucleotide sequence data
GenBank	NCBI	database that contains publicly available nucleotide sequences
BioSample/BioProject	NCBI	BioProject and BioSample databases at NCBI
The Sequence Read Archive (SRA)	NCBI EBI DDJP	Sequence Read Archive

Tableau 2.1 – *Tableau des principales bases de données publiques de séquences ADN/ARN*

3. Stockage numérique des données génomiques

Concernant le stockage sur support numérique des données issues du séquençage de l'ADN, par convention, seul le brin d'ADN 5' est stocké, le brin 3' pouvant être retrouvé par

complémentarité. Afin de faciliter la gestion et l'analyse de ces données, de nombreux formats de stockages ont été développés. En fonction des formats de stockage, les nucléotides peuvent être encodées de différentes manières, occupant plus ou moins d'espace mémoire.

Nom :	Extension :	Description :
SFF (Standard flowgram format)	sff	Format de fichier binaire utilisé par Roche, Whitehead Institute for Biomedical Research et l'Institut Sanger pour stocker les résultats de pyroséquençage
AB1 (Applied Biosystems)	ab1	Format de fichier utilisé par Applied Biosystems pour le stockage des données de chromatogramme
EMBL	embl	Format utilisé par l'EMBL pour le stockage des séquences ADN et protéiques
FASTA	fasta, fas, fap, nt, aa, fna	Format de fichier provenant de la suite d'outils FASTA, utilisé pour le stockage de données de séquences ADN et protéiques au format texte
FASTQ	fq, fastq	Format de fichier utilisé pour le stockage de données de séquences ADN ainsi que les données de qualités correspondantes
SAM (Sequence Alignment Map)	sam	Format texte permettant le stockage de séquences ADN alignées
BAM	bam	Version binaire compressée du format SAM
GenBank	gbk, gb	Format de fichier texte permettant le stockage de séquences ADN et peptidiques, utilisé par GenBank et RefSeq
SRA (Sequence Read Archives)	sra	Format d'archivage de séquences ADN brutes, développé par le NCBI.

Tableau 2.2 – *Tableau des principaux formats de stockage pour les données génomiques*

3.1. Format de fichier texte

La majorité des formats existants sont au format texte. Cela permet de faciliter leur lecture et les échanges puisque leur structure est assez simple, facilement interprétables et ne nécessite pas d'outils spécifiques pour la consultation ou l'édition. Cependant, en termes d'espace de stockage utilisé, les formats texte, ne sont pas très optimaux, puisqu'ils stockent chaque nucléotide sous la forme d'un caractère encodé par la majorité des systèmes d'exploitation, sous 8 bits (1 octet), alors que les quatre états induits par les quatre nucléotides peuvent être encodés avec seulement deux bits de données. De plus, la faible variabilité de l'alphabet à

quatre lettres composant les séquences ADN, ne permet pas une compression optimale avec des méthodes de compression utilisées habituellement pour compresser du texte. Les formats texte, les plus utilisés par la communauté bio-informatique, sont les formats FASTA et FASTQ. Issu de la suite d'outils FASTA [32], le format FASTA est un format de fichier texte, qui permet de stocker chaque nucléotide sous forme d'un caractère (1 octet), de telle sorte que chaque type de nucléotide est représenté par sa lettre initiale. Le format FASTA est accepté par une grande majorité des outils d'analyse. Développé par l'Institut Sanger, le format FASTQ [33] avait pour objectif de lier au sein d'un même fichier, les données de séquençage avec les scores de qualité de lecture, fournis par les séquenceurs ADN.

3.2. Format binaire

Bien qu'une grande majorité des formats de stockage soient au format texte, il existe cependant des formats encodant les données génomiques sous une forme binaire. De structures plus complexes, les fichiers binaires nécessitent des outils spécifiques à chaque format, pour leur consultation, leur édition ou leur conversion. La principale motivation pour l'utilisation de tels formats est de pouvoir utiliser des mécanismes de compression des données. Ainsi, le format BAM (Binary Alignment Map) fait partie des formats de fichier binaires les plus utilisés à l'heure actuelle. Le format BAM, est l'équivalent binaire du format de fichier SAM [34]. Il permet de stocker des données de séquences ADN ainsi que les données d'alignements de ces séquences. Afin de réduire la taille des données, le format BAM utilise l'algorithme de compression BGZF (Blocked GNU Zip Format). BGZF est un algorithme de compression développé spécifiquement pour les données génomiques. Il utilise un mécanisme de découpage des données par blocs, où chacun des blocs est compressé via l'algorithme de compression GZIP. Le découpage par bloc des données, puis leur compression ont pour objectifs d'assurer un bon niveau de compression tout en permettant un accès aléatoire aux données.

4. Généralités sur la compression des données génomiques

En informatique, la notion de compression des données regroupe un ensemble de méthodes et de techniques, permettant de réduire l'espace mémoire occupé par des données via un encodage spécifique de celles-ci ou la gestion des zones redondantes. La compression fait donc référence, à la réduction de la quantité de données utilisées pour représenter n'importe quel type de données informatiques : fichier binaire, document, image son ou vidéo. Afin de pouvoir consulter des données compressées, il est tout d'abord nécessaire de passer par une étape de décompression. En fonction de l'algorithme de compression utilisé, les données pourront, une fois décompressées n'avoir subi aucune perte (compression sans perte) ou avoir subi des dégradations perceptibles ou non (compression avec perte).

Avec l'évolution de la production des données numériques inhérentes à la montée en puissance des ordinateurs, les besoins en termes de stockage sur supports physiques sont devenus de plus en plus importants. Ces besoins ne pouvant être entièrement comblés par l'évolution matérielle des machines, il a dont été nécessaire d'étudier puis de mettre en place des techniques de compression de plus en plus sophistiquées. Ainsi, les techniques de compression ont commencé à voir le jour au début de l'ère de la micro-informatique, les premières machines n'étant que faiblement dotées en mémoire vive ou support de stockage. De nos jours, les techniques de compression sont omniprésentes, avec des applications dans tous les domaines. Nous pouvons citer des algorithmes de compression sans perte, adaptés à la compression de texte, tel que l'algorithme Deflate [35], qui est notamment utilisé par le format de fichier ZIP ou le format d'image PNG. Il combine les algorithmes de compression LZ77 [36] basé sur une méthode de compression via l'élaboration d'un dictionnaire et le codage de Huffman [37], permettant d'encoder de façon optimale des caractères en fonction de leur probabilité de fréquence d'apparition au sein d'un texte. Deflate est notamment utilisé au sein d'outils de compression modernes tels que GZIP ou au sein du format d'image PNG. Il permet de diviser la taille d'un fichier avec des taux pouvant aller jusqu'à 70%. Les algorithmes avec perte, sont majoritairement utilisés pour compresser des données de type image, son ou vidéo. Ainsi, le format MP3, ou plus précisément MPEG-1/2 Audio Layer III, est composé d'une méthode de compression audio, dont le principe est de supprimer du spectre fréquentiel, les fréquences non perçues par l'oreille humaine, provoquant ainsi une réduction significative des

données. Cette notion d'encodage perceptuel est aussi mise en œuvre au sein du format d'image JPEG. En effet, la méthode de compression du format JPEG, s'appuie sur la suppression des hautes fréquences obtenues à partir d'une Transformée en Cosinus Discrète et qui sont porteuses des données relatives aux zones homogènes de l'image.

Les domaines des sciences du vivant, sont aussi concernés par les problématiques liées à l'augmentation des données produites. Ainsi, les données en microscopie, en imagerie médicale et en génomique, ont connues une fulgurante augmentation au cours de ces 40 dernières années. Dès lors, l'utilisation d'algorithmes génériques de compression peut être envisagée, notamment pour les données de type multimédia, cependant, ces algorithmes s'avèrent être moins efficaces concernant le traitement des données génomiques. En effet, l'alphabet servant à représenter les 4 types de nucléotides étant lui-même composé de 4 lettres (*A, T, C, G*), il est difficile de pouvoir appliquer efficacement des algorithmes de compression tel que Deflate. Les algorithmes génériques étant la plupart du temps optimisés pour compresser des données textuelles utilisant, dont les caractères sont encodés sur 1 octet, ils s'avèrent être moins performants pour traiter des données ayant un alphabet avec 4 états distincts et pouvant être encodé uniquement sur 2 bits de données. Il a donc été nécessaire de développer des méthodes spécifiques permettant de prendre en charge efficacement ce type de données avec leurs particularités.

La compression de séquences ADN est encore aujourd'hui considérée comme étant un important challenge et cette thématique de recherche a fait l'objet de nombreuses publications depuis une vingtaine d'années. La plupart des algorithmes ont été développés en utilisant les caractéristiques des séquences d'ADN tels que les mutations ou le complément inverse. Ainsi, nous pouvons tout d'abord citer les algorithmes BioCompress [38] et BioCompress2 [39], dont les approches sont issues de l'algorithme de compression LZW. Ces deux méthodes se basent sur la recherche de zones répétées et de séquences palindromiques, afin de les remplacer par des liens pointant sur les occurrences précédentes.

En 1997 E. Rivals *et al.* [40], publièrent l'algorithme CFACT, dont l'approche générale était semblable à BioCompress avec cependant l'utilisation d'un arbre des suffixes pour stocker les zones identiques les plus longues. À la différence avec BioCompress, le processus de

compression de CFACT est réalisé en deux temps. Lors de la première étape, l'arbre des suffixes contenant les zones identiques les plus longues est construit. Durant la seconde phase, les zones non répétitives, donc considérées par CFACT comme non compressibles, sont encodées trivialement en utilisant 2 bits de données par nucléotide. En ce qui concerne les zones répétées, les secondes occurrences sont remplacées par un pointeur composé de la position de la première occurrence et de sa longueur, les occurrences suivantes sont remplacées par l'identifiant de la première.

À partir de 2001, les méthodes de compression décrites ont commencé à travailler à partir de séquences approchées. Ainsi, X. Chen *et al.* [41], présentèrent GenCompress, un algorithme de compression basé sur la recherche de séquence approximative. Cette méthode permit d'augmenter le ratio de compression à 1,74 bits par nucléotide, comparée aux méthodes précédentes. GenCompress introduisit aussi, pour la première fois, la possibilité d'estimer la proximité entre deux séquences, à partir de leurs formes compressées.

L'algorithme de compression GenomeCompress [42], permis par la suite la compression des zones répétitives et non répétitives via l'utilisation d'un système d'encodage de paquets de 4 nucléotides, sur 5 bits de données et ainsi d'avoir un ratio de 1,25 bits / nucléotides.

En 2010, l'algorithme HuffBit Compress [43], dont l'approche était basée sur l'utilisation d'un arbre binaire étendu, arriva à obtenir un ratio de compression de 1,006 bits / nucléotide.
En 2015, un nouveau type d'approche a été décrit [44] permettant la compression de séquences ADN, via la transformation de celles-ci en image binaire à deux dimensions (2D) et en y appliquant une méthode de compression basée sur la recherche de zones homogènes, ne contenant que des données binaires.

5. Comparaison de séquences ADN

La comparaison de séquences ADN sert à déterminer dans quelle mesure deux séquences S_1 et S_2 se ressemblent, permettant ainsi d'en déduire que ces deux séquences partagent une origine évolutive commune. Ainsi, lorsque deux séquences S_1 et S_2 ont un fort taux de

similarité, on parle de séquences homologues. La recherche d'homologie est donc une tâche incontournable pour toute étude en génomique. Elle est notamment utilisée pour les tâches d'assemblage de plusieurs séquences à partir d'une séquence de référence, pour étudier les mutations ou pour déterminer une sous-séquence commune entre plusieurs séquences. Durant l'étude fonctionnelle des génomes, la comparaison de séquences permet aussi d'effectuer de l'inférence de connaissances sur une séquence à partir des connaissances déjà acquises sur une séquence proche. Ainsi, deux séquences très similaires, dont l'une aurait une fonction codante bien déterminée, permettrait d'émettre l'hypothèse que la seconde aurait aussi une fonction codante proche de la première.

Afin de répondre à ces problématiques, de nombreuses méthodes en bio-informatique ont été proposées. Ainsi, les premières méthodes de comparaison qui ont été utilisées, étaient basées sur des concepts d'algorithmique du texte tel que la Distance de Levenstein [45]. Par la suite, sont apparues des méthodes plus spécifiques en bio-informatique, connues sous le nom d'algorithmes d'alignement de séquences. Puis, d'autres familles de méthodes sont arrivées telles que des méthodes utilisant des algorithmes de programmation dynamique, des algorithmes utilisant des arbres des suffixes ou des algorithmes d'indexation, basés sur des tables de hachage. Les algorithmes de programmation dynamique procèdent en réduisant le problème en plusieurs sous-instances, elles-mêmes pouvant être résolues par décomposition. Les algorithmes basés sur des arbres de suffixes utilisent des arbres comme support d'indexation. Ils ont pour avantage de permettre une recherche rapide. Une autre technique d'indexation consiste à utiliser des tables de hachage comme support. Les tables de hachage permettent un accès rapide aux données qu'elles contiennent. Leurs systèmes d'indexation reposent sur l'utilisation de fonctions de hachage.

5.1. Alignement de séquences

L'alignement de séquence regroupe un ensemble de méthodes et de techniques de bio-informatique, dont l'objectif est de représenter les séquences à aligner, les unes sur les autres, de façon à pouvoir faire ressortir les régions similaires ou homologues. Il existe deux types d'alignement : l'alignement global et l'alignement local.

L'alignement global est utilisé lorsque l'on souhaite trouver le meilleur alignement possible, entre deux séquences S_1 et S_2. Afin de maximiser le score d'alignement, tous les éléments des séquences sont considérés. De plus, lorsque S_1 et S_2 ont des tailles différentes, des insertions (gaps) sont ajoutées à la séquence la plus petite pour qu'elle puisse avoir une taille identique à la séquence la plus longue. Afin de maximiser le score d'alignement, il est aussi possible de supprimer des éléments d'une séquence. En revanche, si les tailles des séquences sont trop différentes, le meilleur score d'alignement est recherché en tentant d'aligner directement la courte sur la plus longue. C'est cette approche qui a donné lieu aux méthodes d'alignement local.

```
S1 :  ATAGATAGTTCGCCTATGTTGCAGTCGGTCTCGGT
        ||       |||||||  ||||||   |||||||
S2 :  TTA---GCATCGCCTAGGTTGCAAGGGGTCTCGAG
```

Figure 2.3 – *Exemple d'alignement global entre deux séquences ADN S_1 et S_2*

L'alignement local est utilisé lorsque l'on souhaite rechercher des zones proches c'est à dire ayant une forte probabilité d'homologie entre deux séquences S_1 et S_2. Ceci permet de pouvoir comparer des séquences ayant des longueurs différentes. Les algorithmes d'alignement local, sont notamment utilisés pour effectuer de la recherche de séquences proches au sein des bases de données biologiques.

```
S1 :  ACTAGTTCGCCTGT--CGTCGATAATCGAAGCCAG
            ||||||||  |||||||||
S2 :  TTAGCATCGCCTGTGTCGTCGATAACATTGCATCT
```

Figure 2.4 – *Exemple d'alignement local entre deux séquences ADN S_1 et S_2*

5.1.1. Algorithme de Needleman-Wunsch

Développé en 1970, l'algorithme de Needleman-Wunsch [12] est un algorithme permettant d'effectuer un alignement global entre deux chaînes de caractères. Dérivé de l'algorithme de Levenstein, il fût le premier algorithme en bio-informatique à utiliser des concepts de programmation dynamique. Ainsi, il utilise une matrice de similarité permettant de déterminer les sous-chemins optimaux, puis par incrémentation, le chemin optimal représentant le meilleur alignement possible entre deux séquences. Bien que très efficace pour calculer le score d'alignement optimal, il est cependant limité par une complexité algorithmique d'*O(nm)*.

5.1.2. L'algorithme Smith-Waterman

Développé en 1981, l'algorithme de Smith-Waterman [46] est un algorithme d'alignement local de séquences biologiques. Il est basé sur l'algorithme de Needleman-Wunsch dont il est une extension. Par conséquent, il reprend les concepts de programmation dynamique, ainsi que l'utilisation d'une matrice de similarité. Cependant, en plus d'un alignement optimal basé sur l'intégralité des séquences qu'il essaie d'aligner, il recherche aussi des régions ou sous-séquences ayant un fort taux d'alignement. Il garantit le meilleur taux d'alignement possible, cependant, en termes de complexité algorithmique, Smith-Waterman a tout comme Needleman-Wunsch une complexité en *O(nm)*.

5.1.3. FASTA

Développé en 1988, FASTA [32] est un algorithme d'alignement local permettant de rechercher les séquences similaires contenues au sein d'une base de données, à partir d'une séquence requête. Il s'appuie sur des méthodes heuristiques lui permettant de trouver rapidement des sous-séquences exactes appelées k-tuples. Pour chaque zone d'alignement, des statistiques sont calculées de façon à fournir des informations permettant de déterminer si elles ont une signification biologique ou s'il s'agit d'alignements statistiquement aléatoires.

Il est à noter que le format utilisé par l'outils FASTA pour stocker les séquences ADN et protéiques est devenu le format standard de référence en bio-informatique.

5.1.4. BLAST

Développé en 1990, BLAST (Basic Local Alignement Search Tool) [6] est un algorithme d'alignement local, permettant de rechercher des zones similaires entre deux ou plusieurs séquences. Basé sur des méthodes de recherche approximées de type heuristique, il ne permet pas de garantir un résultat optimal mais toutefois statistiquement mesuré. Ceci permet d'accélérer considérablement les tâches d'alignement des séquences. De ce fait, Il se révèle significativement plus rapide que ses homologues Needleman-Wunsch et Smith-Waterman, dont il est toutefois dérivé. Par conséquent, il est rapidement devenu une référence incontournable en recherche biologique et en médecine. Les publications scientifiques décrivant le fonctionnement de son algorithme sont aussi très vite devenues, parmi celles les plus cités au monde. En 1997, une seconde version, NCBI BLAST est développée avec pour améliorations principales, la prise en compte des insertions et des délétions ce qui est une avancée majeure et permet une meilleure sensibilité de détection des zones d'homologie.

5.1.5. Indexation des séquences

L'indexation des données regroupe un ensemble de techniques visant à déterminer un mode d'organisation ou de classement particulier, permettant d'en faciliter l'exploitation. Les méthodes d'indexation permettent d'optimiser les temps d'accès aux données, en tirant parti des ressources en mémoire cache des processeurs des machines ou en utilisant des mécanismes de arallélisation. Ces différentes méthodes sont utilisées conjointement avec différents types de structure de données tels que des tables de hachage ou des arbres des suffixes. Elles sont utilisées dans de nombreux domaines telles que la fouille de textes, la recherche d'images ou la recherche de séquences ADN/ARN en bio-informatique.

Dans le domaine de la comparaison de séquences ADN, les principaux algorithmes d'indexation, utilisent des structures de données arborescentes ou des structures de données basées sur des tables de hachage. L'objectif est de diminuer les temps de recherche, notamment de séquences exactes, au sein de volumes de données en constante évolution. Ainsi, lors d'une phase de recherche, l'indexation permet d'effectuer des recherches au sein d'une base de données à partir d'une requête, sans devoir comparer la requête avec toute la base.

La stratégie d'indexation des séquences nucléotidiques ou protéiques a été formalisée par Dumas *et al.* [47]. Elle se base sur l'indexation de sous-séquences courtes et de taille fixe appelées k-tuple ou k-mer. Ainsi, la phase de recherche débute sur la recherche des différents k-tuples au sein de la structure d'indexation. Cette stratégie est notamment utilisée par BLAST, qui recherche des correspondances exactes à partir de k-tuples de 11 nucléotides.

5.1.6. Arbre des suffixes

L'arbre des suffixes est une structure de données, qui permet d'indexer des chaînes de caractères telles que du texte ou des séquences ADN. Ce type de structure stocke l'information sous la forme d'un arbre contenant tous les suffixes possibles de la chaîne à indexer. De cette façon, il permet d'effectuer une recherche très rapide d'un motif ou d'une séquence en parcourant à partir de la racine, les différents nœuds, contenant des suffixes en descendant jusqu'aux feuilles. Si la phase de création de l'arbre, peut s'avérer longue, étant donné le processus de création des différents suffixes, qui est en relation directe avec la longueur des séquences à indexer ; le temps nécessaire à la phase d'interrogation est quant à lui directement proportionnel à la taille de la séquence recherchée. Ce type de structure s'avère donc très efficace lors de la recherche de sous-séquences exactes. Cependant, la principale limitation, est que dans la majorité des cas, elles doivent être maintenues en mémoire vive pour des raisons de performance de lecture au sein des arbres. Ceci peut très rapidement entraîner des limitations concernant la taille des données à indexer, étant donné l'importante redondance nécessaire à la construction d'un arbre optimale. La majorité des algorithmes de construction d'arbre des suffixes produisent des arbres étant 10x plus

volumineux que les données d'origines, ce qui s'avère toujours être une limitation pour l'utilisation de cette structure [48]. Cependant, avec la démocratisation des disques durs de type SSD (Solid State Drive), de nouvelles méthodes, permettant de tirer parti des caractéristiques de ces nouveaux supports de stockage, ont été décrites telle que STS [49].

5.1.7. MUMmer

MUMmer [50] est un algorithme d'alignement global, permettant de rechercher des zones communes entre deux séquences. Il utilise un arbre des suffixes généralisé pour indexer les séquences à comparer. Cette structure permet de rechercher rapidement des zones strictement communes entre les deux séquences, appelées MUM (Maximal Unique Match). Un MUM est une sous-séquence unique, présente au sein des deux séquences à aligner. Par la suite l'algorithme tente d'effectuer des alignements approximatifs au niveau des régions situées entre les MUMs, en utilisant des techniques de programmation dynamique telles que celles employées par l'algorithme Smith-Waterman.

5.2. Table des suffixes

D'autres approches dérivées des arbres des suffixes ont été décrites. Elles se basent sur une version simplifiée appelée table des suffixes [49]. Une table des suffixes contient l'ensemble des suffixes possibles d'une chaîne de caractère. Leur principal avantage est qu'elles permettent de réduire efficacement l'espaces mémoire. *Abouelhoda et al.* [51], ont pu démontrer que tous les algorithmes basés sur l'utilisation d'arbres des suffixes peuvent utiliser avantageusement une structure de type table des suffixes.

Les tables des suffixes sont notamment utilisées comme structure de données par l'algorithme BWT (Burrows-Wheeler Transform) [52] qui permet de réorganiser efficacement les données afin de pouvoir y appliquer des algorithmes de compression. BWT est notamment à la base de l'algorithme de compression BZIP2 et il est aussi utilisé en bio-informatique depuis une dizaine d'années. Ainsi, plusieurs utilitaires tels que BWA (Burrows-Wheeler Aligner) [52],

SOAP2 [53] ou BOWTIE [54] sont basés sur l'utilisation de BWT, ainsi qu'une table des suffixes. L'algorithme BWT, créée un indexe à partir des séquences de références et utilise ensuite des mécanismes de parallélisation pour effectuer les phases d'alignement. Il s'avère très performant pour aligner des séquences ayant un taux élevé de similarité, en permettant de réduire considérablement les temps d'exécution et l'utilisation de la mémoire vive des machines, toutefois, au prix d'une légère perte de sensibilité, par rapport aux algorithmes basés sur le hachage.

5.3. Table de hachage

5.3.1. SSAHA

Développé en 2001, SSAHA (Sequence Search and Alignment by Hashing Algorithm) [55], est un algorithme de recherche de séquences proches au sein d'une base de données, à partir d'une séquence requête. Il utilise une structure de données de type table de hachage pour stocker les séquences indexées. Le processus d'indexation consiste à découper les séquences à indexer en sous-séquences non chevauchantes appelées k-mers. Les k-mers ont une taille *k* et sont stockées au sein de la table de hachage qui répertorie aussi la position qu'ils occupent sur la séquence à indexer. Le processus d'interrogation consiste à découper les séquences requête en sous-séquences, puis de compter le nombre d'occurrences (hits) rencontrées dans la base.

5.3.2. RAMDB

RAMdb (Rapid Access Motif database) [56], est un algorithme de recherche de séquences nucléiques ou protéiques exactes, au sein d'une base de données. Il travaille à partir de courtes séquences d'environs 10pb pour les séquences nucléiques et 4 protéines pour les séquences protéiques. Lors du processus d'indexation, les séquences sont découpées en k-mer, puis stockées au sein d'une table de hachage. La table de hachage associe à chaque k-mer le numéro d'identification de la séquence dont il est issu. La structure de données de type

table de hachage permet des temps de recherche très rapides pour des séquences de même taille ou de tailles supérieures lorsqu'il y a des chevauchements. En revanche, le principal inconvénient de cette structure de données est que sa taille est 2 fois plus importante que la taille des données avant le processus d'indexation.

5.3.3. BLAT

BLAT (BLAST-Like Alignment Tool) [57] est un outil d'alignement local qui a une approche similaire à BLAST, à la différence qu'il n'indexe pas la séquence requête, mais les séquences contenues au sein de la base de données. Lors du processus d'indexation, BLAT extrait et indexe des k-tuples non chevauchants. Ceci a pour conséquence de permettre une réduction des données d'environs 70% par rapport aux séquences d'origine. Cette importante réduction des données permet en outre de pouvoir tenir dans la mémoire vive des machines, réduisant ainsi considérablement les temps d'accès aux données. Par conséquent, les temps de traitement s'avèrent, d'après les auteurs être 500x plus rapides que la majorité des autres outils d'alignement.

5.4. Algorithmes de traitement du signal

Une autre famille de méthodes d'alignement de séquences basée sur l'utilisation d'algorithme de traitement du signal est décrite dans la littérature. L'idée générale, est de considérer les séquences ADN en tant que signal discret, où chaque nucléotide est représenté par un point défini au sein du signal [58], [59], avec des valeurs numériques attribuées de façon arbitraire correspondantes à chaque type de nucléotide. Contrairement aux familles de méthodes précédentes, ces méthodes sont majoritairement utilisées pour la réalisation de travaux d'alignement multiple. Les techniques d'alignement de séquences multiples permettent de déterminer les zones d'homologie entre plusieurs séquences. Elles sont notamment utilisées pour déduire des similarités de structure entre plusieurs gènes ou en phylogénie pour inférer des relations d'évolution entre espèces. La littérature décrit des méthodes basées sur la théorie des ondelettes [60] ou à partir de Transformée de Fourier Discrète telle que MAFFT

[61]. Des méthodes utilisant des algorithmes de DTW (Dynamic Time Warping) [62]. Rakthanmanon et Al, [63] ont aussi été explorées afin de comparer des séquences ADN avec pour objectif de les classifier. D'autre part des méthodes utilisant des transformées en Ondelettes ont aussi été décrites pour effectuer de la détection de motifs récurrents au sein de séquences, permettant ainsi de détecter des régions codantes, [64].

Cependant, bien que l'utilisation d'algorithmes de traitement du signal pour réaliser l'alignement de deux séquences soit une problématique très peu abordée par la littérature, quelques méthodes prouvant l'efficacité de ces approches ont néanmoins été développées. Ainsi en 1982, Felsenstein et Al [65], démontrèrent l'efficacité de l'utilisation d'une FFT pour aligner des séquences ADN, en présentant une méthode d'encodage des séquences au sein d'une matrice binaire, permettant d'exploiter les caractéristiques de regroupement fréquentiel d'une FFT, avec une complexité algorithmique d'ordre logarithmique. Cependant cette méthode avait comme inconvénient, le fait qu'il soit nécessaire de calculer 9 FFT par alignements, une pour chaque type de base nucléique et pour les deux séquences à comparer et une FFT inverse. De plus, sa faible tolérance aux indels ne rendait cette méthode exploitable que dans certains cas précis. C'est probablement la faible tolérance aux indels qui limita l'utilisation des méthodes issues du traitement du signal. Néanmoins, la littérature décrit par la suite quelques méthodes utilisant des FFT [66], [67], [68]. Cependant, ces méthodes avaient toujours pour principale limitation leur intolérance aux *indels*.

Par la suite en 2005, une nouvelle méthode basée sur une corrélation croisée de FFT a été décrite [69], permettant la détection de séquences similaires. L'idée est d'encoder les deux séquences à comparer en assignant un nombre complexe à chaque type de nucléotide, les bases complémentaires ayant des signes opposés. Par la suite, un algorithme de corrélation croisée est appliqué, permettant de déterminer les zones communes entre les séquences comparées. Cette méthode démontra un intérêt certain pour ce type d'approche, qui apporta une meilleure prise en charge des indels.

En 2012, Curilem Saldías *et al.* [10], publièrent une nouvelle méthode d'alignement de séquences basée sur une technique de corrélation des images numériques calculée à partir de fonctions FFT, permettant de faire ressortir les zones communes entre deux images. Les

séquences ADN sont tout d'abord encodées sous la forme d'une matrice de pixels, où pour chaque type de nucléotide, une valeur d'intensité lumineuse est attribuée.

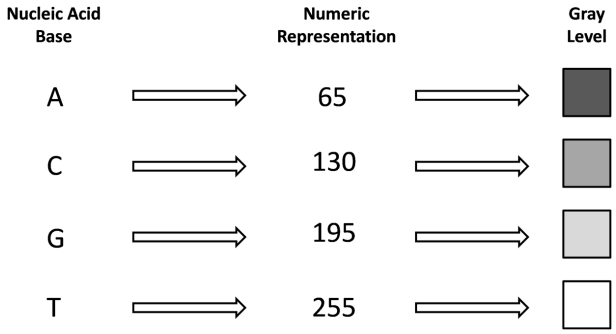


Figure 2.5 – *Chaque base d'acide nucléique est représentée par un nombre spécifique qui représente le niveau de gris d'un pixel*

Le processus de corrélation d'images est appliqué entre les deux images, ce qui a pour effet de faire ressortir les zones communes, c'est à dire les zones ayant un fort taux d'homologie entre les deux séquences.

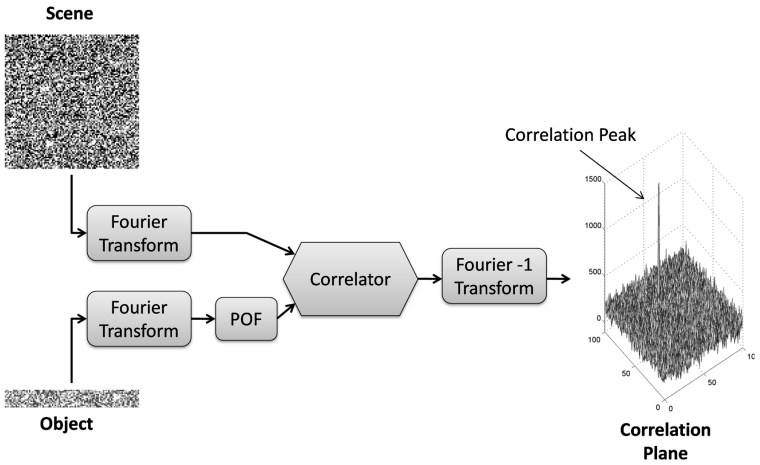


Figure 2.6 – *Les images de scène et d'objets sont transformées par Fourier. L'objet est filtré. Les deux transformées de Fourier sont corrélées. Le pic indique la correspondance*

Bien que les temps de calcul nécessaires à l'exécution de cette méthode d'alignement soient très lents, 100x plus lent que BLAST, les auteurs relatent des niveaux de sensibilité comparables et même supérieurs à BLAST, lorsque les taux de mutation augmentent.

Au-delà des résultats obtenus, cet article a surtout démontré pour la première fois, l'intérêt de l'utilisation des techniques de comparaison d'images numériques, basées sur la représentation des séquences ADN sous la forme de matrices de pixels 2D puis en leurs transpositions dans le domaine fréquentiel à partir d'une TFD 2D.

Conclusion

Le développement des techniques de séquençage de l'ADN a incontestablement été une importante révolution et a initié une nouvelle discipline appelée « génomique ». La génomique a pour objet, l'étude des êtres vivants à l'échelle de leur génome. Le séquençage, puis le développement de différentes techniques d'analyse ont entraîné l'émergence de multiples problématiques inhérentes à la gestion des données produites.

Ce second chapitre a présenté les différentes problématiques inhérentes à l'analyse des données génomiques. Dans une première partie, il a introduit une définition de la génomique en présentant les objectifs biologiques de cette discipline, puis il se focalise plus particulièrement sur différents aspects tels que le stockage des données génomiques, les formats d'échanges, les techniques de compressions et enfin il a présenté les principales méthodes de comparaison de séquences ADN présentes dans la littérature.

Les différentes problématiques abordées durant ce chapitre, présentent naturellement des similitudes avec de nombreux domaines qui ont permis d'évoluer grâce aux progrès induit par le développement du numérique. Durant ce travail, nous avons choisi d'étudier plus particulièrement les méthodes d'analyse et de comparaison des images numériques. Les sciences de l'imagerie ont connu un développement en amont de la génomique et elles ont en commun des modes de production des données qui n'ont cessé d'évoluer depuis plusieurs décennies, soulevant de nombreuses problématiques en termes de stockage, mais aussi et

surtout en termes d'analyse. Ainsi, un certain parallèl peut être établie entre la gestion et l'analyse des séquences biologiques et celles des images numériques.

Ainsi, les travaux menés durant cette thèse ont cherché à explorer la possibilité d'exploiter des concepts et des méthodes utilisés dans le domaine des images numériques afin de voir si ils pouvaient apporter des améliorations face à la problématique de comparaisons de séquences ADN et par extension de recherches au sein de bases de données.

Tableau de synthèse bibliographique

Compression de données	Textuelles	[35], [36], [37]
	Génomique	[38], [39], [40], [41], [42], [43], [44]
Comparaison et alignement de séquences ADN	Algorithmique du texte	[45], [12], [46], [32], [6]
	Indexation	[47]
	Arbre de suffixes	[50]
	Table des suffixes	[49], [52], [53], [54]
	Table de hachage	[55], [56], [57]
	Traitement du signal	[58], [59], [60], [60], [61], [62], [63], [64], [65], [66], [67], [68], [69], [10],

POINTS CLÉS :

- La génomique est la discipline qui permet d'étudier le fonctionnement d'un organisme à l'échelle de son génome
- L'amélioration des techniques de séquençage a permis de pouvoir étudier un organisme au sein de son écosystème.
- Les données générées ont connu une augmentation exponentielle
- Le stockage, le traitement et la comparaison des données génomique restent un enjeu de taille pour les années à venir.

Chapitre 3 - INDEXATION ET COMPARAISON DES IMAGES NUMERIQUES

RÉSUMÉ : Ce troisième chapitre a pour objet de présenter quelques concepts en imagerie numérique et plus particulièrement les aspects liés à l'indexation et la comparaison des images numériques. Après avoir rappelé les étapes de l'évolution des moyens de production et d'analyse des images numériques, il se focalisera plus particulièrement sur les systèmes de recherche d'images et sur leurs différents composants. Ainsi, il présentera les notions de descripteurs d'images locaux ou globaux qui permettent l'indexation des images numériques, puis il présentera différentes structures de données permettant le stockage et la recherche de données indexées.

SOMMAIRE :

Introduction

Depuis une cinquantaine d'années, les recherches autour des sciences de l'imagerie numérique, n'ont cessé de progresser. Cette évolution s'est tout d'abord traduite par des méthodes d'acquisition et de production des images numériques, basées sur des capteurs photographiques, qui sont au fil des générations devenus de plus en plus miniaturisés et performants. La plupart des appareils photos numériques produits aujourd'hui, sont capables de capturer des photographies numériques avec une palette de nuances de 16 millions de couleurs et une définition de plusieurs millions de pixels par cliché. Cette évolution a eu pour conséquence directe de démocratiser la production et l'utilisation des images numériques qui sont devenues en l'espace de quelques années, un élément important, au centre de nos vies numériques. Les images numériques sont une part importante de nos modes de communication et ceci s'est accéléré depuis l'apparition des réseaux sociaux. Mais leur rôle ne s'arrête pas là. En effet, de nombreux travaux de recherche ont permis d'automatiser leur analyse, dans des domaines telles que l'imagerie médicale, la microscopie, l'astrophysique ou l'automatisation de processus industriels…

La nécessité de classer et de traiter les images de façon automatique est une idée centrale en analyse d'images qui a émergé à partir des années 50 pour des problématiques en physique des particules, puis les techniques n'ont cessé de se perfectionner, parallèlement à l'évolution des capteurs photographiques et des puissances de traitements des ordinateurs.

Dans une première partie, ce chapitre introduira brièvement quelques notions autour des images numériques, puis il se focalisera plus particulièrement sur des concepts d'analyses d'images, notamment sur des méthodes permettant de rechercher des images proches au sein d'une base de données, à partir d'images requêtes.

1. Historique

Développées à partir des années 50 dans le cadre d'applications militaires, les techniques de production et d'analyse des images numériques ont connu, jusqu'à nos jours, une évolution

directement liée à l'histoire de l'informatique. L'histoire de l'imagerie numérique civile débute en 1957, avec la première image numérique crée par Russell A. Kirsch, un ingénieur du National Bureau of Standards Américain (NBS). Cette image fut numérisée à l'aide du premier scanner numérique développé par l'équipe de Russell Kirsch. Ce scanner permettait de tracer les variations d'intensité lumineuse à partir de la surface d'une photographie. Cette première image numérique était composée de pixels noirs et blancs d'une définition de 176x176 (30 976 pixels). Elle a été numérisée à partir d'un scan d'une pellicule photographique contenant une photo du fils d'A. Kirsch.

Figure 3.1 – *Première photo numérique crée à partir d'une photographie numérisée à l'aide d'un scanner*

C'est à partir des années 60, que les méthodes d'analyse d'images commencent à émerger. L'objectif était d'arriver à automatiser les processus d'analyse, d'interprétation et de classement des images numériques via l'outil informatique. Ainsi, les premières applications portent sur des techniques de reconnaissance de formes. Elles sont notamment utilisées pour effectuer de la reconnaissance optique de caractères ou pour réaliser un premier niveau d'analyse d'images médicales (image de microscopie ou radiographie). Dans le cas des images médicales, l'idée était d'automatiser les phases de pré-sélection, afin de ne concentrer l'expertise humaine que sur un panel d'images restreint et ayant une forte probabilité de répondre aux critères de sélection préétablis. Cependant, ces techniques étaient difficiles à

exploiter du fait de nombreuses contraintes. En effet, les optiques de l'époque étaient peu performantes et provoquaient de nombreuses aberrations chromatiques ou géométriques. La résolution des clichés était aussi très faible (plusieurs centaines de milliers de pixels) et la puissance de calcul disponible à l'époque ne permettait que des traitements basiques et rendait difficile le traitement sur des lots importants d'images. Le stockage et les opérations de lecture ou d'écriture des clichés s'avéraient très lents et très couteux. Afin de pallier à l'augmentation du volume des images à traiter, les premières techniques de compression d'images virent le jour à partir de la fin des années 60. Ce sont ces premiers travaux qui ont fondé les bases théoriques des formats de compression d'images tels que JPEG.

À partir des années 70, le développement de l'informatique et notamment des circuits électroniques dédiés aux traitements d'images ainsi que la démocratisation des scanners numériques contribuèrent à une évolution sans précédent des techniques de traitement et d'analyse d'images. Ces techniques purent aussi commencer à être réellement exploitées dans différents domaines scientifiques et grand public. Ainsi, il devint possible d'intervenir en temps réel sur des images ou des vidéos. L'image de référence appelée « Lena », permet d'illustrer cette époque, qui symbolise la transition entre l'utilisation d'images de faible qualité et le début de l'utilisation de photographie. En 1973 Alexander Sawchuk, à la recherche d'une nouvelle image pour illustrer un article scientifique pour une conférence, utilisa une photo issue d'un magazine, qui deviendra l'image de référence dans le domaine de l'analyse d'image.

Figure 3.2 – *Image de référence « Lena » (512x512 pixels)*

Image de référence Lena : « Lena » est l'image numérique de référence dans le domaine de l'analyse et du traitement d'image. Il s'agit d'une image d'une taille de 512x512 pixels numérisée à l'aide d'un scanner de type Muirhead. Alexander Sawchuk décida d'utiliser cette image à la place des traditionnelles images des années 60, issues des standards télévisuels des développés durant les années 60, qu'il qualifiait d'« ennuyeuses ». Mise à part, son caractère esthétique, l'image « Lena » a été choisie, tout d'abord car elle contient un visage humain ainsi que différentes zones texturées au niveau des cheveux, du chapeau et des plumes du personnage. Cette image contient aussi d'autres zones intéressantes, notamment, l'arrière-plan qui est légèrement flouté et qui contient des zones d'aplat et ombragées, pouvant servir à vérifier la dégradation des algorithmes de compression. Cette image possède aussi une faible variabilité au niveau de sa palette de couleurs. Par conséquent, ces différentes caractéristiques ont eu pour conséquence qu'elle soit souvent utilisée dans sa version en niveau de gris.

La production, le traitement ainsi que la diffusion, l'indexation et la recherche des images numériques sont devenus des tâches communes, qui sont aujourd'hui réalisables par le plus grand nombre. De nos jours, les images numériques sont devenues omniprésentes dans notre quotidien. Les différentes évolutions technologiques ont démocratisé leur création, leur production et leur diffusion. Accessibles au grand public à partir des années 80 avec la commercialisation des premiers micro-ordinateurs et des premiers scanners numériques, elles n'ont vraiment pris leur envol qu'à partir de la seconde moitié des années 90, avec l'apparition en masse des appareils photos numériques, puis des smartphones équipés de capteurs photographiques. Parallèlement, la miniaturisation des capteurs, l'augmentation des puissances de traitement, des capacités de stockage et des débits de connexion à Internet ont aussi joué un rôle majeur pour leur création, leur diffusion et leur publication.

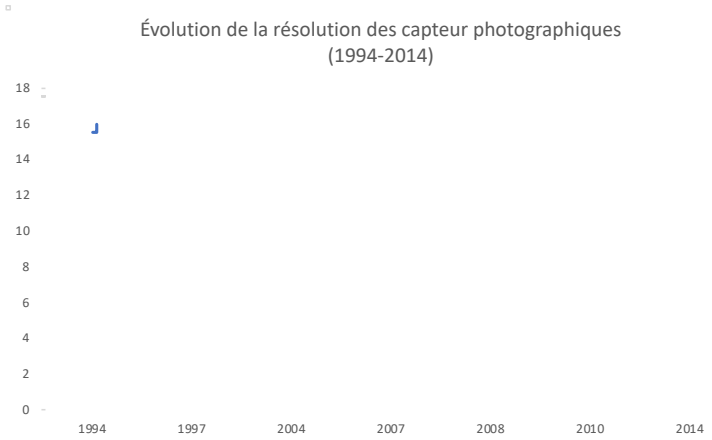


Figure 3.3 – *Évolution (en millions de pixels) de la résolution des capteurs photographiques entre 1994 et 2014*

Les différentes évolutions successives en termes de production et de diffusion conduisent à une spectaculaire augmentation des images numériques sur internet, ce qui pose de nombreux challenges, en termes de stockage mais aussi d'analyse de cette gigantesque masse de données produites quotidiennement. À titre d'exemple en 2012, Google revendiquait avoir indexé mille milliards d'images, en 2015 le service de partage de photos Flickr revendiquait plus de 13 milliards de clichés et pour l'année 2017, Instagram revendiquait recevoir 70 millions de nouvelles photos par jour et Facebook 8 millions de nouvelles images par heure.

2. Généralités sur les images numériques matricielles

Le terme « image numérique », désigne de façon générale tout document de type image, acquis par un capteur optique durant un processus de numérisation ou généré par un système numérique. Une image numérique se présente sous la forme d'une matrice, composée d'un ensemble fini de valeurs numériques, appelées pixels (Picture Element). Les pixels sont la plus petite composante d'une image. Ils sont généralement de forme carrée, cependant, certaines normes peuvent les définir comme étant rectangulaire. Leur couleur correspond à une valeur numérique dont l'intervalle correspond au niveau d'échantillonnage.

(a) (b)

Figure 3.4 – *(a) Image de référence « Lena », (b) matrice de 12x12 pixels, issue de l'image (a)*

Contrairement à une image physique, qui est constituée d'un signal continu composée d'une infinité d'intensités lumineuses, une image numérique est une discrétisation des coordonnées spatiales de ce signal représentant ses deux dimensions. Comme tout processus d'acquisition d'un signal analogique au format numérique, le processus de discrétisation est réalisé en effectuant une quantification, c'est à dire, la représentation d'un signal continu par un ensemble de valeurs numériques discrètes et ceci en fonction de différents critères d'échantillonnage. Les images numériques ont aussi comme particularité d'avoir leurs systèmes d'encodage au sein des supports de stockage dissociés de leurs représentations finales. Cette notion a son importance, car elle permet l'application de modifications, de réaliser des comparaisons ou des échanges au travers de transactions numériques. Ceci induit aussi, la notion d'interactivité puisque les supports numériques permettent un contrôle au niveau de l'affichage des images, mais aussi au niveau de leur ordre d'enchainement. La notion d'interactivité, est une notion essentielle, car elle est à la base de toute interface graphique et plus généralement de l'informatique moderne telle que nous la connaissons encore aujourd'hui.

2.1. Caractéristiques des images numériques

Une image numérique est une matrice composée d'éléments numériques appelés pixels. Plus l'image contient de pixels, plus elle aura un rendu détaillé. Mais de façon générale, une image numérique est caractérisée par quatre paramètres qui définissent les conditions de son rendu final.

2.1.1. La résolution

La résolution correspond à la densité de pixels utilisés par unité de mesure (PPP – Point Par Pouce), afin de représenter l'information spatiale visuelle. Ainsi, plus la densité de pixel est élevée, plus la représentation des détails de l'image sera restituée.

(a) (b)

Figure 3.5 – *(a) Image « Lena » avec une densité de 70 pixels par pouces et (b) une densité de 30 pixels par pouce*

2.1.2. La définition

La définition est le nombre de pixels servant à représenter les images numériques. Elle est définie par le nombre de pixels sur la hauteur de l'image multiplié par le nombre de pixels sur

la largeur de l'image. Le nombre de pixels est une valeur absolue, car une image d'une définition de N sur M pixels aura toujours le même nombre de pixels et ceci quel que soit la taille qu'on lui donnera par la suite.

(a) (b)

Figure 3.6 – *(a) Image « Lena » avec une taille de 512x512 pixels et (b) une taille de 256x256 pixels*

2.1.3. L'échantillonnage

L'échantillonnage, correspond au nombre de couleurs pouvant être encodé au sein des différents pixels qui composent une image numérique. Ainsi, une image encodée en niveau de gris (256 niveaux d'intensité lumineuse), aura un échantillonnage de 8 bits par pixels et une image couleur (16,5 millions de couleurs) aura un échantillonnage de 24 bits soit 8 bits par sous-pixels (Rouge, Vert, Bleu).

(a) (b)

Figure 3.7 – *(a) Image « Lena » avec un échantillonnage de 24 bits et (b) 8 bits*

2.1.4. Le format d'encodage

Les formats d'encodage des images permettent de décrire les conventions utilisées pour leur stockage au sein des supports numériques. Plusieurs formats de stockage ont été développés et standardisés. Ils permettent de faciliter la consultation des images ainsi que leurs échanges. Libres ou propriétaires, les formats d'encodage peuvent conserver les données brutes des capteurs (format RAW), permettant ainsi une phase de post-traitement approfondie ou au contraire, compresser les images, afin d'optimiser l'espace qu'elles occupent pour en faciliter leurs échanges. Les algorithmes de compression utilisés peuvent être sans perte (compression lossless) avec des taux de compression relativement faibles ou avec perte, entrainant ainsi des taux de compression beaucoup plus importants, au prix de pertes d'information et d'une dégradation certaine de l'image. Les algorithmes de compression avec perte tel que le format JPEG permettent de maîtriser les dégradations infligées aux images, en fonction du taux de compression souhaité.

Nom :	Description :
BMP (Windows Bitmap)	Format d'image développé par Microsoft et IBM et principalement utilisé au sein des premières version de Windows. Ce format a une structure simple et il est non compressé.
JPEG (Joint Photographic Expert Group)	Format d'image autorisant un haut niveau de compression avec perte. Ce format est considéré comme le standard pour la photographie.
TIFF (Tagged Image File Format)	Format d'image utilisé par les professionnels. Il privilégie la qualité au détriment de l'espace de stockage.
PNG (Portable Network Graphic)	Format d'image utilisant des méthodes de compression sans perte. Il devait supplanter le JPEG pour devenir le nouveau standard pour les images numériques.
GIF (Graphics Interchange Format)	Format d'image léger, ne proposant qu'une palette de 256 couleurs. Il a pour particularité de prendre en charger des images animées.
AV1 (Alliance for Open Media)	Format d'image utilisant l'algorithme de compression AV1. Ce format propose un haut niveau de compression des images sans perte d'information

Tableau 3.1 – *Principaux formats de fichiers pour le stockage d'images numériques*

3. Analyse d'images

L'analyse d'images est un ensemble de méthodes et de techniques qui consistent à extraire de façon automatique, des informations quantitatives ou structurelles, contenues au sein des images numériques. L'objectif étant de pouvoir les caractériser, comparer tout ou partie des éléments qui les composent, afin de pouvoir les classer. Cette notion de comparaison implique des mécanismes de reconnaissance de motif, de forme ou de texture, permettant l'identification totale d'une image, à partir de ses caractéristiques intrinsèques, ou de certains éléments qui la composent. De façon générale les techniques d'analyse d'images sont utilisées lorsqu'il est nécessaire d'effectuer de la reconnaissance de forme ou d'interroger une base de données d'images en fonction de critères spécifiques de sélection. Elles sont donc couramment utilisées dans divers domaines, tels que, la physique, la biologie, la médecine, la mécanique ou la robotique, et elles ont connu un changement d'échelle important depuis le développement d'internet et des réseaux sociaux. Compte tenu de cette masse de données d'images, qui ne cesse de croitre de façon exponentielle, les enjeux sont désormais de pouvoir

analyser de façon automatique cette masse d'information, avec un temps de réponse acceptable par rapport aux besoins des utilisateurs.

3.1. Généralités sur la comparaison d'images

La comparaison d'images est une problématique de base en analyse d'image. Elle a pour objet de déterminer si deux images i_1 et i_2 sont proches en fonction de critères de correspondance qui peuvent être locaux ou globaux. La comparaison d'image intervient dans différentes problématiques, telles que la compression d'images ou de vidéos, la détection d'objets ou de mouvements… Il s'agit d'une problématique qui doit faire face à une multitude de paramètres, les images à comparer pouvant être de taille, de résolution, ou d'échantillonnage différents. Les images à comparer peuvent aussi avoir subi des recadrages, des rotations ou des dégradations tels que l'application de filtres ou l'ajout de motifs... S'ajoute aussi à ces différents critères, la recherche d'images globalement proches au sein de bases de données qui est en plein essor, compte tenu de l'accroissement considérable des images mises en ligne sur le Web. Il devient donc nécessaire de pouvoir comparer une image requête avec toutes les images présentes au sein d'une base de données, afin de pouvoir obtenir les images les plus proches.

4. Recherche d'images au sein de bases de données

Les systèmes de recherche d'images au sein de base de données ont pour objectif de retourner, à partir d'une requête fournie en entrée, les images les plus pertinentes contenues dans la base de données, c'est à dire répondant à certains critères spécifiques. En fonction du type de système de recherche d'images, les requêtes peuvent êtes de type textuel ou être sous forme d'images. De façon générale, ces systèmes sont composés de quatre parties :

- **L'indexation** : le processus d'indexation consiste à soumettre chaque image à indexer, à une fonction d'indexation, afin de calculer les descripteurs correspondants et de les stocker au sein de la base de données d'indexes. L'idée est de substituer les images par

des descripteurs visuels moins volumineux, mais néanmoins toujours représentatifs des images originales.

- **Fonction d'indexation** : Une fonction d'indexation a pour objectif de calculer des descripteurs à partir des images passées en paramètre d'entrée. Les descripteurs doivent être de taille inférieure aux images d'entrée et toutefois, toujours être représentatifs de celles-ci. De plus, les descripteurs doivent pouvoir être comparés entre eux, de façon à pouvoir établir une notion de similarité entre les images.
- **L'interrogation** : le processus d'interrogation consiste à soumettre les images requêtes à la fonction d'indexation, afin d'en extraire les descripteurs visuels correspondants et de les comparer aux descripteurs présents dans la base de données, via une fonction de mesure de correspondance.
- **Fonction de mesure de correspondance** : Une fonction de mesure de correspondance permet la comparaison de deux ou plusieurs descripteurs, de façon à en établir une mesure de correspondance, c'est à dire une notion de similarité ou de distance entre les images comparées.

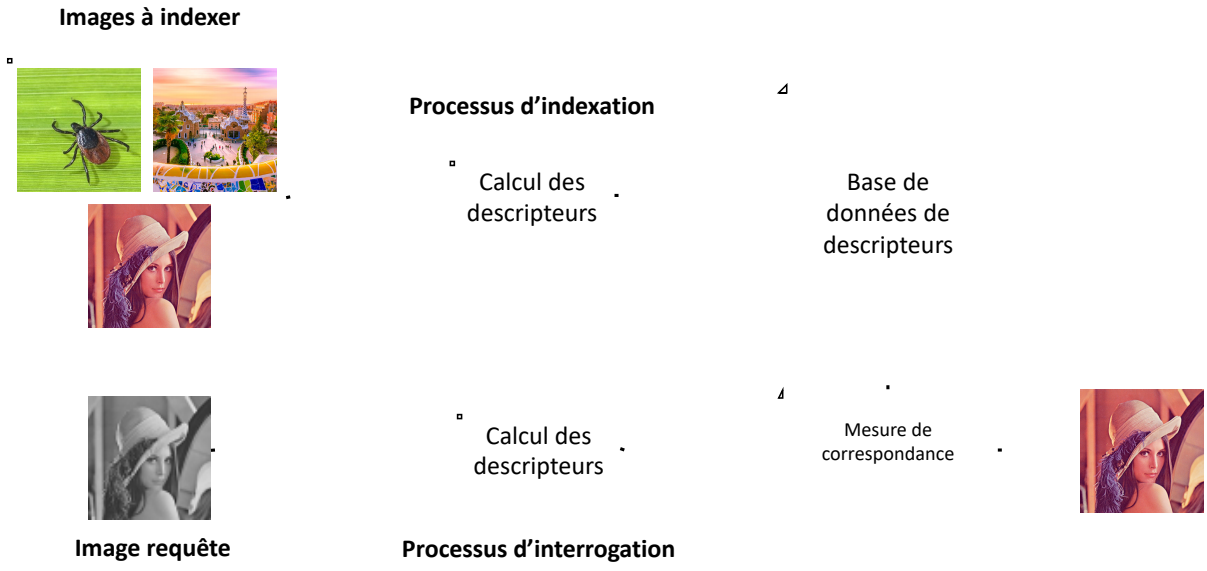


Figure 3.8 – *Principe générale d'un système d'indexation et de recherche d'images*

4.1. Systèmes de recherche d'images

Il existe deux principaux types de système de recherche d'images, les systèmes d'indexation textuelles et les systèmes d'indexation basés sur le contenu visuel. Au sein des systèmes d'indexation textuelle, les images sont annotées manuellement et nécessitent donc une intervention humaine. Ces systèmes sont notamment utilisés par les agences de presse, les musées ou les centres d'archives, lorsqu'il est nécessaire de pouvoir retrouver rapidement des images à partir de mots clés ou d'expressions. Cependant ces systèmes peuvent aussi être automatisés. C'est notamment le cas des moteurs de recherche tels que Yahoo ou Google qui utilisent les informations textuelles tels que les noms de fichiers ou le texte proche des images. Néanmoins, ils intègrent aussi des mécanismes d'indexation du contenu visuel des images. Les systèmes d'indexation du contenu visuel, indexent les images de façon automatique en se basant sur les caractéristiques intrinsèques des images tels que les couleurs, les motifs, les formes ou la dynamique fréquentielle. Ce genre de système est notamment utilisé pour l'identification de visages, pour détecter des doublons ou l'utilisation frauduleuse de photographies.

4.2. Systèmes basés sur l'annotation textuelle des images

Les premiers systèmes d'indexation furent mises en œuvre durant les années 80 avec des travaux présentant des systèmes permettant la recherche d'images par annotation textuelle [70]. Ils consistaient, lors du processus d'indexation, à étiqueter manuellement chaque image, avec des mots clés, ou des descriptions textuelles. Ces systèmes sont tributaires de la qualité d'annotation sémantique des images qui s'avère être subjective et très coûteuse en temps. De plus, dans beaucoup de cas, les descriptions textuelles ne décrivent pas suffisamment l'image et n'apportent souvent que des informations subjectives. Ces systèmes ont toutefois comme principal avantage de permettre aux utilisateurs de rechercher des images via un système de requêtes utilisant un langage de haut niveau, avec toutefois la difficulté que les personnes qui réalisent l'indexation n'utilisent pas forcément le même champ lexical que les utilisateurs du système. Compte tenu du changement d'échelle qui a eu lieu durant ces 15 dernières années, le travail d'annotation peut difficilement être réalisé manuellement, avec

un niveau de précision permettant une classification optimale. Cependant, quelques bases de données très spécialisées peuvent encore être indexées manuellement. Ainsi, ImageNet [71], est une base de données contenant environs 10 millions d'images annotées manuellement. Cette base de données est développée à destination des travaux de recherche en analyse d'images et sert notamment dans le cadre de recherche en vision par ordinateur.

Par la suite, sont apparus des systèmes permettant la classification sémantique via l'annotation textuelle automatisée des images. Dans la plupart des cas, ces méthodes s'appuient sur des méthodes de classifications supervisées multi-classes [72], [73] Cependant, ces approches ne permettent pas de réduire le problème du « *fossé sémantique* » [74], car elles ne peuvent extraire que la sémantique perceptuelle des images. Le fossé sémantique, correspond au décalage sémantique existant entre la représentation bas niveau d'une image et l'interprétation haut niveau de la même image faite par les humains.

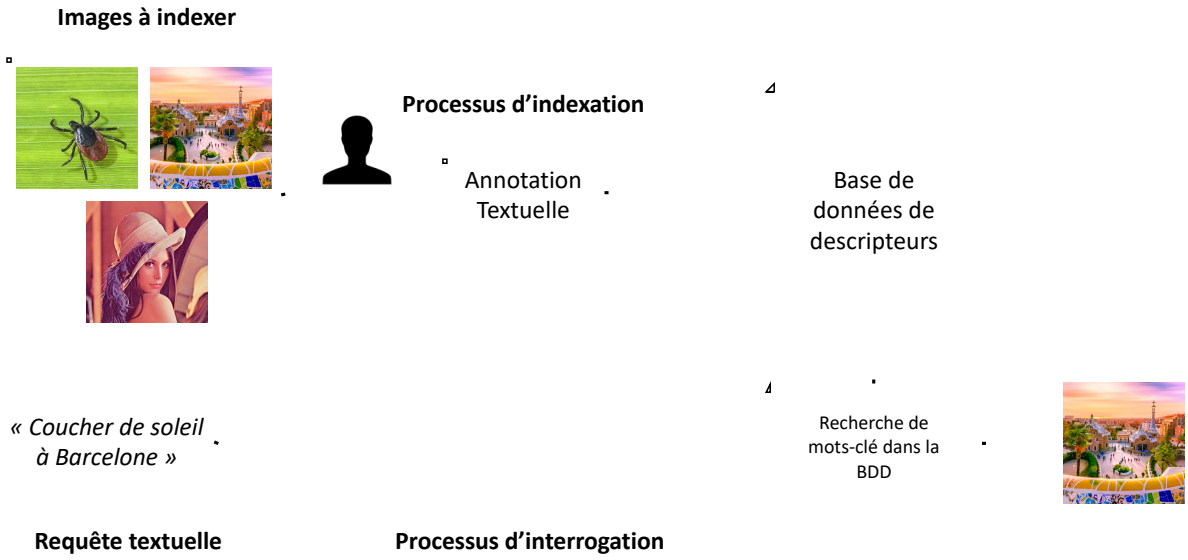


Figure 3.9 – ***Principe général d'un système d'indexation textuel des images numérique***

4.3. Recherche d'images par le contenu visuel

Les systèmes de recherche d'images par le contenu, ont pour objectif de rechercher des images au sein d'une collection, à partir des caractéristiques visuelles d'une image requête.

Compte tenu de l'augmentation sans précédent du nombre d'images à stocker, les méthodes d'analyse d'image ont dû évoluer en conséquence, afin de faire face à ce changement d'échelle.

La recherche d'images par le contenu visuel au sein de base de données est une problématique de recherche qui se situe à l'intersection des domaines de l'analyse d'image et des bases de données. Ces systèmes ont été développés à partir de la seconde moitié des années 90 [75]. Ils avaient pour objectif de pallier aux inconvénients des systèmes d'indexation textuel. Ce type de systèmes proposent une phase d'indexation automatisée, basée sur l'extraction de caractéristiques perceptuelles des images telles que les couleurs, les textures, les formes, afin d'en extraire des « signatures ». La phase d'indexation des images, consiste à extraire de façon automatisée des caractéristiques représentatives des images, mais d'une taille bien inférieure, en utilisant des fonctions d'analyse d'image appelées « descripteurs d'images ». De même, lors du processus d'interrogation, les descripteurs servent à encoder les images requêtes afin qu'elles puissent être comparées aux images présentes dans la base de données.

Parallèlement, il a aussi été nécessaire de développer des structures de données permettant une interrogation efficace des corpus d'images. Ces structures de données doivent être adaptées aux caractéristiques des descripteurs d'images mais aussi aux problèmes inhérents aux espaces de grande dimension. Le développement d'un système de recherche d'images par le contenu nécessite donc à la fois une expertise en analyse d'image mais aussi en base de données.

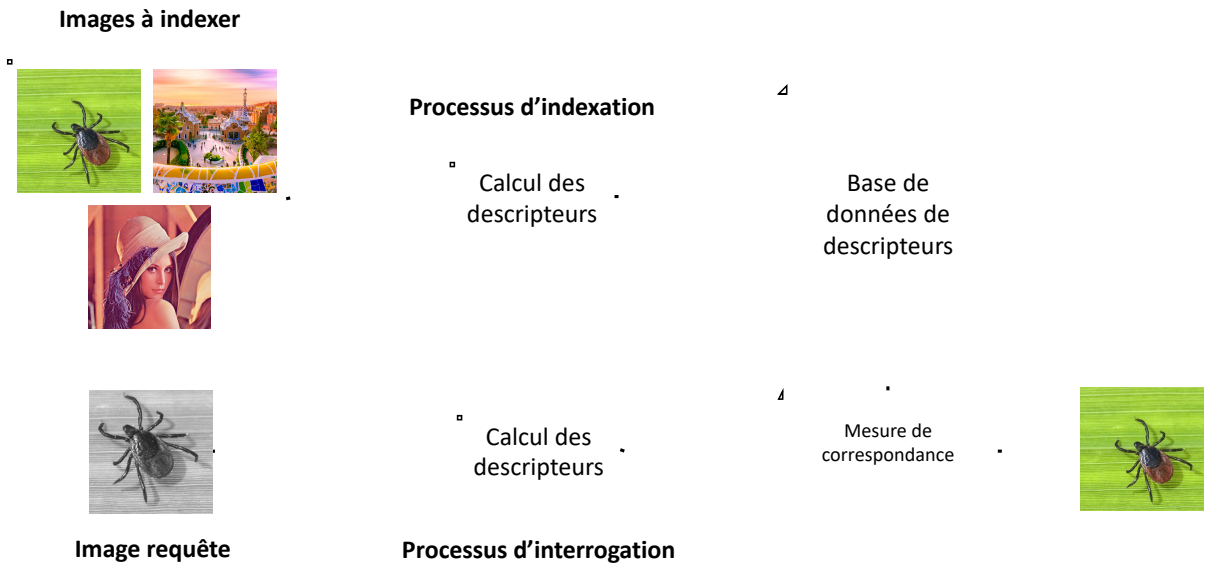


Figure 3.10 – *Principe général d'un système d'indexation et de recherche par le contenu visuel d'images numériques*

5. Descripteurs d'images

Les descripteurs d'images ont pour objectif d'extraire des caractéristiques visuelles à partir des images à indexer, afin d'en calculer des résumés qui seront représentatifs et comparables au sein du système. L'extraction des caractéristiques visuelles est réalisée en utilisant des descripteurs d'images. Les descripteurs d'images permettent de convertir les caractéristiques visuelles en valeurs numériques appelées signatures. Les descripteurs n'ont pas un caractère universel et doivent être développés en fonction des caractéristiques des images à traiter. Afin de pouvoir comparer les images entre elles, les descripteurs d'images sont associés à des mesures de similarité. Ainsi, rechercher des images similaires au sein d'une base de données, revient à évaluer les empreintes les plus proches au sens de la mesure de similarité qui leur est associée. Il est aussi nécessaire de rechercher un compromis entre précision de comparabilité et espace de stockage occupé par les signatures.

La littérature décrit deux grandes familles de descripteurs visuels : les descripteurs globaux et les descripteurs locaux. Les descripteurs globaux ont pour objectif d'indexer les images dans leur intégralité de façon à ne générer qu'un nombre restreint d'empreintes représentatives

de celles-ci. Tandis que les descripteurs locaux visent à indexer les images de façon locale, offrant la possibilité de réaliser des recherches sur une partie de l'image.

5.1. Descripteurs visuels globaux

Les descripteurs globaux ont pour objectif de calculer des empreintes prenant en compte les images dans leur intégralité. Ce type de descripteur peut servir à rechercher des images proches au sein d'une base de données, à détecter des doublons ou du contenu publié sans autorisation, en violation des droits d'auteur ou de propriété intellectuelle.

Afin d'indexer les images, les descripteurs globaux se basent sur trois caractéristiques de bases des images.

La couleur : Les travaux de *M. J. Swain et al.* [76] concernant l'indexation des couleurs, ont ouvert la voie en matière d'indexation des images. Ainsi l'utilisation de la couleur comme caractéristique de base permettant l'indexation est la plus décrite dans la littérature [77], [78], [79]. La perception visuelle de l'œil humain se base principalement sur la variation des couleurs, pour identifier ou différencier des images. De nombreuses méthodes se basent sur la répartition des couleurs au sein d'une image, décrites à l'aide d'histogrammes représentant leurs différentes fréquences d'apparition. Ce type de descripteur est relativement robuste aux transformations géométriques pouvant être subies par une image. Ceci permet l'utilisation de tels descripteurs dans le cadre de recherche d'images approximatives. Cependant, l'utilisation des couleurs comme caractéristique de base est hautement dépendante aux conditions de luminosité. Par conséquent, deux images identiques, avec des niveaux de luminosité différents pourraient ne pas être considérées comme proches par le système.

La texture : l'indexation des images à partir de leurs textures, permet d'obtenir une bonne représentation se basant sur les zones homogènes de celles-ci. Deux familles de descripteurs d'images basées sur les textures existent. La première est basée sur des descripteurs statistiques. Parmi ces approches statistiques, l'une des méthodes les plus utilisées se base sur la matrice de co-occurrence proposée par *Haralick et al.* [80]. La littérature présente aussi

des méthodes basées sur des run-length matrix [81], des higher order statistics [82], de la géométrie des fractales [83], des motifs binaires locaux [84] ou des approches probabilistes, basées sur des champs aléatoires de Markov [85]. La seconde famille de descripteurs regroupe des approches basées sur la théorie du signal. Ainsi, la littérature décrit des méthodes d'extraction des caractéristiques des textures basées sur la Transformée de Fourier [86], les transformées en Ondelette [87] ou les filtres de Gabor [88]. Le principal avantage de ces méthodes basées sur la théorie du signal, est leur capacité à être tolérante au bruit et aux changements de taille des images.

La forme : Les descripteurs de formes sont la troisième catégorie des descripteurs d'images et sans doute la plus utilisée. Comparé aux autres types de descripteurs (couleur et texture), ce type de descripteur s'avère être le plus difficile à mettre en œuvre. En effet, l'extraction de caractéristiques représentatives à partir des formes et les méthodes de comparaison peuvent être difficiles à calibrer. Cependant, ce type de descripteur est beaucoup plus efficace pour calculer des attributs sémantiques caractéristiques à partir du contenu des images [89], [90]. Ainsi, les descripteurs de forme s'appuyant sur l'utilisation des contours, basent leurs approches sur l'extraction de la structure générale de l'image ou le contour des objets, ce qui permet par la suite de pouvoir en déterminer des mesures simples. Parmi les méthodes basées sur l'utilisation des contours, nous pouvons tout d'abord citer celles utilisant une Transformée de Fourier comme élément de base. Le principe est d'obtenir une représentation 1D, à partir du contour des images, via l'utilisation de la Transformée de Fourier. L'idée de base est de considérer les contours comme étant des signaux fréquentiels et par conséquent de pouvoir les représenter par leurs primitives. Par conséquent, les primitives de base permettant de représenter la forme en cours d'analyse, peuvent être utilisées en tant que descripteurs de la forme entière [91].

Par la suite, d'autres méthodes proposant des systèmes d'encodage des formes ont été proposées. Nous pouvons citer le codage de Freeman [92], le codage par histogrammes chaînés [93], par histogramme de gradient orienté [94], par codage de contexte de forme [94], par courbure de l'échelle spatiale [96], les moments de Hu [97] ou de Zernike [98].

5.2. Descripteurs visuels locaux

Le second type de descripteur nommé « descripteur local », permet d'indexer le contenu des images localement. Contrairement aux descripteurs globaux, qui permettent d'indexer des images entières, les descripteurs locaux permettent d'indexer différentes zones d'une image. Ceci permet de pouvoir comparer et retrouver des objets similaires, plutôt que de comparer les images dans leur globalité. En effet, lors de la recherche d'un objet, une recherche basée sur un descripteur global peut, dans certains cas, se révéler approximative, puisqu'elle doit prendre en compte les images dans leur intégralité. Les descripteurs locaux permettent de décrire localement les régions des images, de telle sorte qu'il soit possible de pouvoir les distinguer des autres régions, tout en gardant la possibilité de pouvoir établir un degré de similarité avec celles qui sont suffisamment similaires.

Les descripteurs locaux peuvent être catégorisés en deux types, les régions d'intérêts et les points d'intérêts.

Régions d'intérêts : au cours du processus d'indexation, les images sont segmentées en différentes zones. Le processus de segmentation automatique doit prendre en compte les spécificités visuelles des images. Cette tâche peut s'avérer difficile et fortement dépendante du type d'image et d'objets à indexer [99].

Points d'intérêts : le principe général de ce type de descripteur est de représenter les images via un ensemble de points d'intérêts, extraits automatiquement durant la phase d'indexation. Les points d'intérêts doivent êtres représentatifs de l'image et sont issus des variations de la structure de celle-ci, notamment au niveau des fortes variabilités locales. Ils doivent être discriminants et robustes face aux déformations géométriques, aux bruits et aux différentes attaques que peuvent subir les images. L'ensemble des différents vecteurs de points d'intérêts constituent les empreintes caractéristiques des images [100].

Ces deux approches sont différentes et ne décrivent pas les mêmes informations issues des images. Les points d'intérêts semblent mieux adaptés à la discrimination d'objets fins ou de zones homogènes, tandis que les régions d'intérêts se focaliseront davantage sur les zones

dominantes de l'image. De plus le découpage de l'image étant réalisé durant la phase d'indexation, les phases de recherche ne pourront s'effectuer que sur ces mêmes régions, alors que les points d'intérêts semblent plus flexibles au niveau des zones de recherche. Cependant ce type de descripteur s'avère être plus difficile à mettre en place, la structure de données permettant le stockage des points indexés, devant être multidimensionnelle. Le choix d'un type de descripteur doit être réalisé en fonction des caractéristiques des images à indexer et du type de requête que l'on souhaitera réaliser. Cependant, [101] ont démontré la complémentarité de ces deux approches.

La littérature décrit de nombreux descripteurs locaux [102]. La problématique de la comparaison d'images via l'utilisation de descripteurs locaux fut décrite de façon théorique, pour la première fois en 1954. F. Attneave [103] observa qu'une forme pouvait être identifiée à l'aide de points d'intérêts, malgré des variations de ses contours. Ce n'est qu'à partir des années 70 [104], que les applications purent débuter. À l'époque, l'utilisation de descripteurs locaux, était l'approche la plus envisageable, compte tenu des puissances de calcul disponibles. La détection des caractéristiques se concentraient sur la détection d'objets simples au sein d'images très structurées, plutôt que des images contenant des scènes naturelles. P.R. Beaudet [105] décrivit en 1978 une méthode basée sur la matrice de Hesse, permettant de détecter les blobs et les crêtes à partir d'images simples.

À partir de la fin des années 70, les premiers descripteurs locaux pouvant exploiter des images comportant des scènes naturelles furent présentés. La littérature décrit des méthodes telles que l'opérateur d'intérêt de Moravec [104] basé sur la variation d'intensité dans le voisinage local d'un pixel ou une version améliorée, le détecteur de coin d'Harris [106]. Cependant, les méthodes proposées dans les années 1970 et 1980 avaient comme inconvénient d'être sensibles aux changements d'échelle. Par la suite, plusieurs approches ont été développées, en utilisant des variations multi-échelles de méthodes existantes [107], puis des méthodes basées sur l'invariance d'échelle [108] [109] ou la détection de points d'intérêts invariants [110], [111].

La littérature décrit aussi des descripteurs locaux basés sur des modèles Gaussiens [100], Laplaciens [109] ou à partir de fonction utilisées pour le traitement du signal. Ainsi nous

pouvons citer la méthode SURFT (Speeded-Up Robust Features) [112] et [113], utilisant des transformées en Ondelettes, des Transformées de Fourier [114] ou des Transformées en Cosinus Discrète [115].

5.3. Descripteur d'images binaires

Apparu il y a moins d'une dizaine d'années, les descripteurs binaires d'images sont une nouvelle catégorie de descripteurs qui ont pour avantages d'avoir une faible complexité algorithmique ainsi qu'un coût réduit en termes d'espace de stockage. Cependant, malgré leurs avantages, les descripteurs binaires peuvent souffrir d'un manque de fiabilité car, contrairement aux autres catégories de descripteurs, leur robustesse aux transformations, aux distorsions ou aux changements d'échelles que peuvent subir les images est limitée. Par conséquent, ce type de descripteur doit être adapté à une problématique ou un type d'image bien défini. Ils sont principalement utilisés dans le cadre d'applications mobiles ou de système ayant besoin d'avoir des temps de réponse proches du temps réel. En 2010, M. Calonder, et. Al. [116] présentèrent BRIF le premier descripteur binaire, basé sur l'utilisation d'un noyau de lissage Gaussien. Ce travail ouvrit la voie aux descripteurs binaires. Il permit de démontrer la possibilité de compacter l'information provenant des images, au sein d'empreintes au format binaire. Mais il démontra surtout la possibilité de pouvoir comparer les empreintes entre elles en utilisant une distance de Hamming. En effet la distance de Hamming s'avère être d'une complexité algorithmique inférieure à la distance Euclidienne, qui servait habituellement à comparer les empreintes issues des descripteurs générant des empreinte sous la forme de vecteurs de nombres réels. Par la suite, d'autres descripteurs ont vu le jour. La littérature décrit deux types de descripteurs binaires. La première approche consiste à utiliser des techniques de quantification [117], [118] et de hachage, pour calculer une représentation binaire à partir des images. Utilisées par différentes méthodes de compression d'images, les techniques de quantification permettent, dans le cadre des descripteurs binaire, de conserver uniquement les informations caractéristiques des images en leur appliquant une dégradation importante. La seconde approche consiste à appliquer des tests binaires en calculant les différences d'intensité à partir de paires de pixels sélectionnées et identifiées comme étant représentatives d'une zone d'une image [116], [119].

6. Stockage au sein d'une structure de données

La seconde phase du processus d'indexation est le stockage des empreinte correspondantes aux images, au sein d'une base de données. La distribution des signatures au sein de leur espace de description dépend directement des descripteurs et doit être prise en considération lors de la mise en place d'un système d'indexation. Les premiers systèmes de recherche d'images par le contenu, utilisaient le plus souvent des structures de données stockées en mémoire vive, dépourvue de système d'indexation performants. Avec l'augmentation du volume de données à gérer, il est rapidement devenu nécessaire de mettre en place des systèmes d'indexation capables d'organiser l'espace des signatures selon une structure d'index, permettant d'accélérer la recherche. Pour être performante, la structure d'indexes doit tenir compte des spécificités des empreinte qu'elle doit stocker. Ainsi, en fonction des descripteurs et des empreinte à stocker, il est possible de distinguer deux catégories de structure de données.

Structure à vecteur d'empreinte unique : ce type de structure stocke une empreinte par image. Il est généralement utilisé avec les descripteurs globaux ou certains types de descripteurs locaux, qui n'indexent qu'une partie d'une image, telles que des parties fixes et toujours situées au même endroit d'une image.

Structure à vecteur d'empreintes multiples : ce type de structure permet de stocker plusieurs empreintes par image, sous la forme de vecteurs de même dimension. Cette structure est plus complexe que la précédente, puisque toutes les empreintes issues de la même image peuvent avoir une notion de dimensions ou de contraintes. Ce type de structure est utilisé conjointement avec les descripteurs d'images locaux.

Outre les structures en mémoire vive, les bases de données de type relationnel ont été les premières structures de données à avoir été utilisées comme support pour les systèmes d'indexation textuels [120]. Dotés de structures simples, ils servaient à stocker les différentes annotations ainsi que leurs relations, correspondantes aux images indexées.

Par la suite, compte tenu de l'importante augmentation de la production des images et afin de pouvoir apporter une réponse à ce changement d'échelle, les bases de données relationnelles ont été remplacées par l'utilisation de bases de données de types NoSQL en tant que structure de données.

Conclusion

Développées à partir des années 50, les méthodes d'acquisition et d'analyse des images numériques n'ont cessé d'évoluer et sont aujourd'hui un élément central des technologies du numérique. Le développement de l'électronique a permis l'élaboration de circuits de plus en plus miniaturisés et parallèlement l'augmentation des puissances de calcul des ordinateurs, améliorant ainsi les méthodes d'acquisition de traitement et d'analyse des images numériques.

Ce troisième chapitre a présenté la notion d'images numériques et plus particulièrement les problématiques sous-jassantes d'indexation et de comparaison. Dans une première partie, il a tout d'abord introduit quelques rappels historiques, puis il a présenté les principales caractéristiques des images numériques ainsi que les principaux formats. Par la suite, il s'est focalisé sur les techniques d'analyses, notamment en ce qui concerne les tâches d'indexation via l'utilisation de descripteurs textuels et visuels, les systèmes de recherche d'images par le contenu et les systèmes de stockage au sein de structures de données.

Les problématiques abordées au sein de ce chapitre telles que les méthodes d'indexation et de comparaison sont communes par certains aspects avec les problématiques liées à la gestion et à l'analyse des séquences ADN. En effet, les images numériques et les données génomiques sont toutes issues d'un processus d'acquisition et subissent une transformation au format numérique en vue d'être stockées et analysées. Dès lors, les processus d'analyse de base sont semblables, puisqu'ils nécessitent de pouvoir comparer tout ou partie des images ou des séquences ADN. Ces comparaisons doivent être réalisées avec comme contrainte forte de ne pas être strict mais, de devoir prendre en compte des facteurs de redimensionnement, de dégradation ou de compression pour les images numériques et des erreurs de séquençage

ou les mutations pour les séquences ADN. C'est avec ce postulat que nous avons basé les travaux menés tout au long de cette thèse. Nous avons notamment orienté nos travaux, à partir des descripteurs visuels globaux qui permettent, à partir d'une image de calculer un identifiant statistiquement unique, pouvant être comparé avec d'autres afin d'en estimer une notion de distance. Ces types de descripteurs d'images s'apparentent aussi à une famille de fonction de hachage appelée hachage perceptuel.

Tableau de synthèse bibliographique

Indexation	Annotation textuelle	[70], [71], [72], [73], [74]
	Par le contenu	[75]
Descripteurs visuels globaux	Couleur	[76], [77], [78], [79], [94]
	Texture	[80], [81], [82], [83], [84], [85], [84], [86], [87], [88]
	Forme	[89], [90], [91], [92], [93], [94], [95], [96], [96], [97], [98]
Descripteurs visuels locaux		[99], [101], [100], [102], [103], [104], [105], [106], [108], [109], [110], [111], [112], [113], [114], [115]
Descripteurs binaires		[116], [117], [118], [119]

POINTS CLÉS :

- Le domaine de l'imagerie numérique n'a cessé d'évoluer. Il est devenu un élément central des technologies du numériques
- Les méthodes d'acquisition, d'analyse et de traitement sont en constante évolution
- La comparaison d'images est une problématique de base en analyse d'images
- Les descripteurs d'images ont pour objectif d'extraire des caractéristiques visuelles à partir des images à indexer, afin d'en calculer des résumés représentatifs et comparables

Chapitre 4 – FONCTIONS DE HACHAGE : THÉORIE ET APPLICATIONS

RÉSUMÉ : Ce quatrième chapitre a pour objet de présenter le concept de fonction de hachage. Après une introduction générale, il présentera les caractéristiques communes des fonctions de hachage, puis il se focalisera sur les différentes familles de fonction de hachage ainsi que sur leurs applications et usages.

SOMMAIRE :

Introduction

Le terme « hachage » fait référence au terme culinaire « hacher », désignant le fait de découper des aliments en petit morceaux, où à partir d'une infime partie de ceux-ci, il est toujours possible de les identifier. En informatique, le concept de fonction de hachage a émergé très tôt, avec le développement des premiers systèmes. Ainsi, en 1953, Hans Peter Luhn, ingénieur chez IBM décrivit un système de recherche, utilisant l'un des tout premiers mécanismes de hachage. De nos jours, dans un contexte où la production des données connaît une évolution exponentielle, il a été nécessaire de mettre en place des mécanismes capables de garantir l'intégrité de celles-ci, de façon rapide et transparente pour l'utilisateur final. Ce contrôle d'intégrité ne pouvant s'effectuer sur l'intégralité des données en un temps acceptable, des méthodes mathématiques ont été développées, afin de calculer des identifiants statistiquement représentatifs. Ces méthodes sont généralement regroupées sous la dénomination de fonction de hachage. Initialement utilisées depuis les années 60, en tant que méthodes de vérification lors du transfert de données à travers un réseau, elles sont devenues la pierre angulaire permettant de garantir la sécurité de nombreux systèmes. Elles servent notamment à stocker des mots de passe de façon sécurisée au sein d'une base de données, permettant ainsi de ne pas stocker les mots de passe en clair tout en étant capable de les vérifier. Elles permettent aussi de garantir des transactions financières et servent de technologie de base à des cryptomonnaies telles que le Bitcoin. Mais leur utilisation ne s'arrête pas là, puisqu'elles sont aussi utilisées comme mécanisme de création d'index au sein de structure de données, telles que des tables de hachage. Enfin, elles concourent aussi à l'authenticité des documents électroniques (signature électronique de documents), en garantissant les différents acteurs d'une transaction (émetteur ou receveur). Elles peuvent aussi servir de système de base pour la gestion des fichiers au sein des systèmes d'exploitation ou pour l'identification de documents multimédia, dans le cadre de systèmes anti-plagiat.

Ce chapitre présente le concept de fonction de hachage. Dans un premier temps, il proposera une définition générale, puis fera une description des caractéristiques communes à toutes fonctions de hachage. Enfin, dans un second temps, il présentera les différents types de fonction de hachage, leurs particularités ainsi que leurs usages.

1. Généralités sur les fonctions de hachage

Une fonction de hachage *H* est une fonction mathématique qui, à partir de données fournies en entrée, applique une succession de traitements reproductibles et renvoie une sortie, appelée valeur de hachage. Les données fournies en entrée peuvent être de taille fixe ou variable, cependant les données de sortie auront, pour une même fonction de hachage, une taille fixe. Contrairement aux fonctions de type Codes d'Authentification de Messages (CAM ou MAC), qui nécessitent l'utilisation de clés, conjointement aux valeurs d'entrée, pour calculer les valeurs de sortie, les fonctions de hachage se basent uniquement sur les valeurs d'entrée, pour calculer les valeurs de hachage. Ainsi, les valeurs de hachage sont directement issues des données d'entrée. Cependant, étant donnée la perte d'informations induite par les différents traitements effectués lors du calcul des clés, elles ne pourront que l'être qu'incomplètement. En fonction du contexte d'utilisation, l'empreinte renvoyée peut être nommée : haché, clé de hachage, condensé ou signature. Durant ce présent manuscrit, nous utiliserons préférentiellement le terme de « clé de hachage ».

$$H : \{0,1\}^* \rightarrow \{0,1\}^n$$
$$M \mapsto H(M)$$

Équation 2.1 – Définition mathématique d'une fonction de hachage *H*

Où $\{0,1\}^*$ désigne l'ensemble des données d'entrée possibles et $\{0,1\}^n$ l'ensemble des clés de hachage retournées de taille n

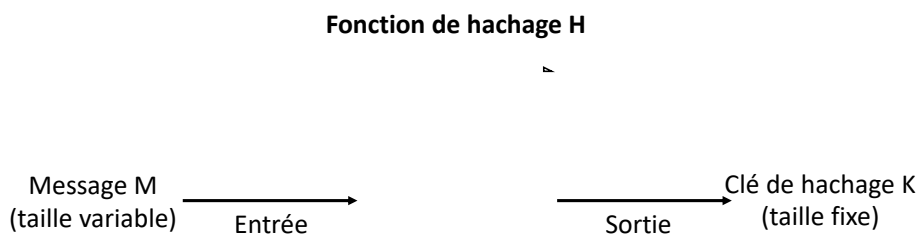


Figure 4.1 : *Représentation schématique d'une fonction de hachage H*

2. Caractéristiques communes des fonctions de hachage

Les fonctions de hachage possèdent des propriétés communes. Cependant, en fonction de leur cas d'utilisation, des caractéristiques spécifiques peuvent être recherchées, tels que « l'effet avalanche » ou la possibilité d'établir une distance entre deux clés.

Ainsi, les propriétés communes aux fonctions de hachage sont les suivantes :

2.1. Absence de clé additionnelle

Aucune clé additionnelle n'est nécessaire pour le calcul des clés de hachage. De ce fait, elles sont directement calculées à partir des données fournies en entrée de la fonction.

2.2. Faible complexité algorithmique

Les fonctions de hachage doivent avoir des complexités algorithmiques d'*O(n)*, de façon à pouvoir être exécutées en un temps très limité. Cette caractéristique est essentielle, car le calcul des clés par la fonction de hachage ne doit pas occuper un temps de calcul trop important par rapport au reste des instructions qui l'appellent. L'algorithme de calcul des clés de hachage ne doit toutefois pas être trop rapide, ceci pouvant favoriser l'utilisation d'attaque par force brute.

2.3. Représentativité

Une fonction de hachage doit générer des clés de hachage directement représentatives des données fournies en entrée et ceci de façon constante. Ainsi les différents traitements appliqués sur les données en entrée doivent être reproductibles. De ce fait, si l'on fournit en entrée d'une fonction de hachage plusieurs fois les mêmes données, les clés de hachage renvoyées en sortie, devront, elles aussi, être strictement identiques. Contrairement aux algorithmes de chiffrement tels que RSA [121], aucun mécanisme de type pseudo-aléatoire

n'est utilisé. Par conséquent, une clé de hachage est directement calculée à partir des données dont elle est issue et peut être considérée comme fiable, avec néanmoins, une probabilité de collision qu'il est nécessaire d'estimer.

2.4. Surjectivité

Une fonction de hachage, est dite surjective car tous les éléments de l'ensemble d'arrivée (l'ensemble des clés de hachages possibles) possèdent au moins un antécédent dans l'ensemble de départ (les données fournies en entrée). De ce fait, une clé de hachage faisant partie de l'ensemble d'arrivée a au minimum un antécédent et au maximum *n* antécédents. Ceci a pour effet d'engendrer une probabilité de collision, c'est à dire qu'une même clé de hachage puisse avoir été générée à partir de deux données d'entrée totalement différentes. Par conséquent, la notion de représentativité est statistiquement incomplète. De ce fait, une fonction de hachage est par définition dite non injective, puisque l'ensemble de départ est plus grand que l'ensemble d'arrivée.

2.5. Non réversibilité

Une fonction de hachage ne peut être réversible, on parle alors de fonction à sens unique ou One-Way Hash Function (OWHF) [122], [123]. À partir d'une clé de hachage, il n'est pas possible de recalculer les données dont elle est issue. Il ne peut donc exister de fonction de hachage inverse, la seule possibilité de retrouver les données d'entrée à partir d'une clé de hachage, étant de générer l'ensemble de départ possible ainsi que l'ensemble d'arrivée correspondant. On parle alors de méthode d'attaque par « force brute ». Étant donné le nombre de combinaisons possibles, ce genre d'attaque est difficilement réalisable en pratique du fait du temps ou de la puissance de calcul nécessaire pour sa réalisation. Cependant, des failles peuvent être découvertes au sein des différentes étapes de l'algorithme d'une fonction de hachage. Ces failles peuvent notamment permettre de compromettre intentionnellement la notion de représentativité en permettant de générer des collisions maîtrisées, rendant ainsi inutilisable la fonction de hachage corrompue.

2.6. Uniformité

Une fonction de hachage doit calculer et renvoyer des clés de hachage au sein d'un intervalle de valeurs définies, de façon uniforme. Afin de minimiser les probabilités de collision induites par la surreprésentation d'un intervalle de valeurs particulier, les clés de hachage doivent avoir la même probabilité d'être générées.

2.7. Probabilité de collision

Le terme collision, désigne, le fait qu'une fonction de hachage renvoie des clés de hachage identiques à partir de données d'entrée strictement différentes. Afin de garantir la représentativité des données, une fonction de hachage, doit avoir une faible probabilité de collision. Les fonctions de hachage étant par définition non injectives, ceci induit nécessairement une probabilité de collision. Lors du développement d'une fonction de hachage, il est ainsi nécessaire de faire une étude de probabilité de collisions. Ceci permet de quantifier la résistance aux collisions qui sont généralement induites par la taille des clés renvoyées et par conséquent le nombre de clés de hachage possible dans l'ensemble d'arrivée.

La probabilité de collision d'une fonction de hachage est calculée en utilisant une estimation probabiliste appelée le « Paradoxe des Anniversaires » [124]. Le Paradoxe des Anniversaires découle d'une vérité mathématique contre-intuitive pour laquelle la majorité des gens estiment très basse la probabilité que dans un groupe de 50 personnes, deux puissent avoir leur anniversaire le même jour du même mois, alors qu'elle se révèle pourtant être de 50% au sein d'un groupe de 23 personnes.

L'étude des probabilités de collision d'une fonction de hachage, revient donc à évaluer les différentes probabilités de collision en fonction du nombre de clés générées. Ainsi, une fonction de hachage qui renvoie des clés de hachage binaire de n bits, à 2^n clés de hachage binaires possibles. En prenant m séquences ADN différentes, la probabilité que deux séquences aient la même clé de hachage est donc de :

$$p = 1 - \left(1 - \frac{1}{2^n}\right)\left(1 - \frac{2}{2^n}\right) \dots \left(1 - \frac{m-1}{2^n}\right)$$

Équation 2.2 – Paradoxe des Anniversaires *p*

Où *n* est le nombre de bits servant à coder la clé de hachage pour *m* valeurs d'entrée différentes

2.8. Effet avalanche

Décrit pour la première fois par H. Feistel [125], l'effet avalanche est un mécanisme qui est une composante essentielle pour toutes fonction de hachage cryptographique et pour les algorithmes de chiffrement par bloc. Dérivé de la théorie de la Confusion et de la Diffusion de C. E. Shannon [126], il a pour effet de provoquer des modifications de plus en plus importantes sur les données, au fur et à mesure de l'exécution de l'algorithme. Par conséquent, en appliquant une modification sur un bit de données en entrée, la fonction doit renvoyer par propagation, une sortie très différente. Le cas échéant, il pourrait être envisageable pour un attaquant d'arriver à prédire les valeurs d'entrée à partir des valeurs de sorties. Les propriétés chaotiques nécessaires à l'effet avalanche ont pour objectif de rendre mécaniquement difficile, voire impossible le développement d'une fonction de type inverse. Ces propriétés sont notamment recherchées pour le développement des primitives de chiffrement nécessaires notamment aux fonctions de chiffrement par bloc.

Expression	HASH MD5	HASH SHA2
Effet avalanche	`54da4177b349c3cc4569db7cc8ca36ef`	`c1735de6c8214dde7f8837440c1f6b1d28d2ee3c`
effet avalanche	`35bf8be923a16168172e36cec3c44fac`	`a20aaedcce35d9a6e7285907d1fcb2c88b2f83bf`

Figure 4.2 – *Exemples de clés de hachage de type MD5 et SHA2 obtenues à partir de l'expression « effet avalanche » avec un « e » et un « E »*

Si la plupart des fonctions de hachage intègrent l'effet avalanche, afin de limiter la reconstitution des données d'entrée à partir des clés de hachage, dans certains cas, il peut

être intéressant de pouvoir établir une distance entre deux clés. Certaines fonctions de hachage sont dépourvues d'effet avalanche. L'objectif est de pouvoir comparer deux données d'entrée, au travers de leurs clés de hachage respectives. Ainsi, les données fournies en entrée sont transformées en une représentation de dimension plus faible, mais conservent néanmoins la propriété de comparabilité. Cette propriété est notamment utilisée dans le cadre de la recherche de similarité ou plus proche voisin [127], par les fonctions de type Locality Sensitive Hashing (LSH) [128] ou de hachage perceptuel [129].

3. Types de fonctions de hachage

De façon générale, les fonctions de hachage ont pour objectif de générer des clés de hachage directement représentatives des données fournies en entrée, afin de permettre un contrôle d'authenticité ou d'intégrité sur celles-ci. Cependant, elles diffèrent principalement au niveau de leur mise en œuvre et de leur utilisation. En effet, lorsqu'il est nécessaire de garantir l'authenticité d'un document avec des mécanismes n'autorisant aucune modification, même minime d'un document, afin de garantir son authenticité, il est nécessaire de recourir à une fonction de hachage cryptographique. À titre d'exemple, afin de garantir qu'une application téléchargée depuis un serveur distant n'ait subi aucune modification, il sera nécessaire de calculer la clé de hachage de la version téléchargée, puis de la comparer avec celle fournie par le serveur. Dans certains cas, notamment concernant les documents multimédias (image, son, vidéo), l'authentification se fera à l'aide des caractéristiques perceptuelles (visuelles ou auditives) et ne prendra pas en compte leurs représentations binaires. En effet, deux images d'apparence strictement identiques ne seront pas forcément encodées dans le même format, ou pourront avoir des facteurs de compression différents. Dans ce cas, l'utilisation d'une fonction de hachage cryptographique ne sera pas très adaptée et il conviendra d'utiliser une fonction de hachage perceptuel, pour satisfaire les besoins particuliers de sécurité des images numériques.

Il existe de nombreuses fonctions de hachage, et à moins de connaître à l'avance l'intégralité des données à stocker, l'option la plus sûre, consiste à utiliser une fonction de hachage qui

distribue uniformément les données de façon aléatoire, au sein de l'intervalle des valeurs de hachage possibles.

3.1. Fonctions de hachage cryptographiques

Développées à partir des années 70 [130], les fonctions de hachage cryptographiques sont les fonctions de hachage les plus connues et les plus couramment utilisées. Elles permettent de garantir une forte intégrité des données et constituent la pierre angulaire des mécanismes de sécurité modernes [131]. Initialement, les premières fonctions de hachages ont été développées pour simplifier la gestion des moteurs de bases de données. Elles ont notamment permis de simplifier les mécanismes de comparaison des données, les comparaisons ne se faisant plus directement sur les données, mais sur leurs clés de hachage correspondantes, de taille fixe et surtout de taille très inférieure. Outre les opérations de comparaison, l'utilisation des fonctions de hachage a aussi permis d'accélérer les fonctions de tri, d'insertion, mais surtout, un accès plus rapide aux données, via l'utilisation de structures telles que des tables de hachage.

Contrairement aux algorithmes de chiffrement par bloc ou par flot, les fonctions de hachage cryptographiques ne dépendent d'aucune clé, ce qui les rend assez singulières dans leurs fonctionnements. Les fonctions de chiffrement produisent des données de sortie d'un volume équivalant ou supérieur aux données d'entrée. De ce fait, leurs propriétés de non réversibilité et la taille fixe des clés qu'elles renvoient, ne permettent pas de les catégoriser en tant que méthodes de chiffrement.

La principale caractéristique recherchée pour ce type de fonction de hachage est la production de clés de hachage de façon non prévisible. De ce fait, si l'on considère deux données d'entrée ayant qu'un seul bit de différent, les clés de hachage seront quant à elles totalement différentes. Cette caractéristique est appelée Effet Avalanche. Par conséquent, le calcul de distance entre deux clés s'avère être impossible.

De façon générale, les fonctions de hachage cryptographique sont largement utilisées dans le domaine de la sécurité informatique. Elles sont à la base de nombreux mécanismes de sécurité. La résistance de ces mécanismes repose essentiellement sur les capacités de résistance des fonctions de hachage à différents types d'attaques. Ces attaques peuvent être menées via des méthodes de cryptanalyse telles que des attaques par force brute ou des attaques par création de collisions volontaires, exploitant des failles dans leurs algorithmes ou dans l'implémentation de ces derniers.

Il existe de nombreux types d'algorithme de hachage tels que Message Digest (MD2 [132], MD4 [133], MD5 [134, p. 5] and MD6 [135, p. 6], RIPEMD (RIPEMD-160 [136], Whirlpool (WHIRLPOOL [137] ou encore Secure Hash Function (SHA-1 [138], SHA-2 [139], and SHA-3) [140, p. 3]. Ces différentes fonctions reposent toutes sur la Construction de Merkle-Damgård [141], présentant les concepts de base méthodologique pour l'élaboration des fonctions de hachage cryptographique, en prenant en compte des aspects tels que l'Effet Avalanche et le Paradoxe des Anniversaires décrit plus haut. L'objectif de cette construction est tout d'abord, de découper en blocs de taille identique les données fournies en entrée, puis de leur appliquer une fonction de compression. La forme compressée de chaque bloc est envoyée aux blocs suivant selon plusieurs méthodes possibles [142], [143].

Actuellement, les algorithmes de hachage les plus utilisés sont SHA-256 et X11. Ils succèdent aux algorithmes MD5 et SHA-1, ayant fait l'objet de différentes cryptanalyses depuis une dizaine d'années [144]. Ces différentes cryptanalyses ont permis de démontrer des méthodes permettant d'augmenter drastiquement les probabilités de collision, rendant de ce fait ces fonctions non sûres et par conséquent obsolètes.

3.1.1. Famille de fonctions SHA-2

SHA-2 est la seconde famille de fonctions de hachage cryptographique apparentée au groupe SHA (Secure Hash Function). Elle a été labélisée en 2002 par le NIST et utilise toujours le principe de la Construction de Merkle-Damgård. SHA-2 se décline en 4 fonctions, SHA-224, SHA-256, SHA-384 et SHA-512. Les fonctions SHA2, produisent respectivement des

empreintes de taille fixe comprises entre 224, 256, 384 et 512 bits. Cependant, SHA-224 et SHA-384 sont des versions renvoyant des résultats tronqués des deux autres fonctions. La famille de fonctions SHA-2 est toujours considérée aujourd'hui comme étant robuste. Cependant le NIST à en 2012 organisé une compétition (la Hash Function Competition) qui a permis de sélectionner la fonction Keccak [145], qui devrait devenir le nouveau standard SHA-3.

3.1.2. Méta-fonction de hachage X11

Développée avec la crypto-monnaie Dash, X11 est une meta-fonction de hachage cryptographique. Elle a été conçue de façon à pouvoir soutenir ce système de monnaie, notamment pour en sécuriser les transactions mais aussi pour retarder la fabrication de puces dédiées (ASICs - Application-Specific Integrated Circuit), facilitant les opérations de minage. L'objectif était de donner suffisamment de temps à la monnaie pour se développer, avant que la prolifération des ASICs, ne permettent à certaines plateformes de minage, de devenir majoritaires et de ce fait entrainer la centralisation, qui est le risque majeur de toute crypto-monnaie. X11 est basée sur pas moins de 11 algorithmes de hachage cryptographique [146], BMW [147], Groestl [148], JH [149], KECCAK [145], SKEIN [150], luffa [151], CubeHash [152], SHA-vite [153, p.], SIMD [153], et ECHO [154].

L'idée majeure d'X11, était d'apporter un niveau de sécurisation théorique supérieur par rapport aux systèmes utilisant une seule fonction de hachage. En effet, bien que la probabilité qu'une fonction de hachage cryptographique, telle que SHA256 puisse être corrompue soit extrêmement faible, l'utilisation de 11 fonctions de hachage permet une sécurisation optimale. De ce fait, pour corrompre un tel système, il sera nécessaire de corrompre les 11 fonctions de hachage sur lesquels il se base. C'est donc ce niveau de sécurité qui est recherché par différents systèmes basés sur une BlockChain.

4. Domaines d'application

Les fonctions de hachage cryptographique ont de multiples applications. Étant à sens unique, c'est à dire non réversibles, elles permettent de garantir l'intégrité d'un message échangé entre plusieurs interlocuteurs. Cette notion est fondamentale et elle est utilisée au sein de nombreux systèmes.

4.1. Codes d'authentification de message

Les fonctions de hachage sont notamment à la base des codes d'authentification de message, aussi appelés MAC (Message Authentication Code). Les MAC servent, lors d'un échange d'informations entre deux interlocuteurs, à garantir l'intégrité de celles-ci. Les premiers MAC étaient basés sur l'utilisation de fonction de chiffrement [155]. Couplées à une clé secrète, connues des interlocuteurs, les fonctions de chiffrement leur permettent de calculer, à partir d'un message d'entrée de longueur arbitraire, un bloc chiffré de longueur fixe qui est renvoyé en sortie. Par la suite, certains types de MAC, les HMAC (Hash-based Message Authentication Code), ont utilisé des fonctions de hachage cryptographiques, telles qu'MD5 ou SHA-1. Par rapport à la simple utilisation d'une fonction de hachage, un HMAC se différencie par le fait que le message haché est généré conjointement à partir d'une clé secrète connue seulement des interlocuteurs de l'échange et d'une fonction de hachage. De ce fait, personne n'est en mesure de calculer la clé de hachage sans connaître la clé.

4.2. Signatures électroniques

Les signatures électroniques permettent aux utilisateurs de signer numériquement un document. Cette signature qu'il ne faut pas confondre avec une signature calligraphique manuscrite qui aurait été numérisée, permet de signer numériquement un document à l'aide d'une clé privée qui pourra par la suite être vérifiée par les destinataires, grâce à une clé publique. En 1976, *Diffie et al.* [130] décrivirent de façon théorique le premier système de signature électronique. Par la suite les premiers systèmes de signature utilisèrent des

méthodes de chiffrement tel que l'algorithme RSA (Rivest Shamir Adleman) [156]. Cependant, ces méthodes de chiffrement avaient comme principaux inconvénients d'être lents et de générer des sorties dont les tailles étaient identiques au document original. Par conséquent, l'utilisation de fonctions de hachage cryptographiques a été proposée [157], ce qui donna naissance à l'algorithme DSA (Digital Signature Algorithm). Certains systèmes se basent aussi sur l'utilisation de fonctions de hachage cryptographiques telles que MD5 [134] ou SHA-256 [158, p. 256]. L'idée majeure était de ne plus comparer l'intégralité des messages chiffrés, mais seulement l'empreinte associée et calculée à partir des messages originaux, ce qui eut pour conséquence d'améliorer grandement les performances.

4.3. Conservation des mots de passe au sein d'une base de données

L'un des aspects primordiaux d'un système d'authentification, est sa capacité à protéger les mots de passe des différents comptes des utilisateurs. Cet aspect est capital, puisque dans la majorité des cas, les utilisateurs sont authentifiés à partir de leur mot de passe. L'inconvénient principal de ce type d'authentification, est que, dès lors de la compromission de la sécurité d'un système, la liste des mots de passe pourra être dérobée, permettant ainsi aux attaquants de pouvoir utiliser n'importe quels comptes utilisateurs pour s'authentifier et bénéficier des différents niveaux de privilèges associés. *Chang et al.*, [159] décrivirent en 1994, un système d'authentification basé sur une fonction de hachage cryptographique et le crypto-système d'ElGamal. Par la suite des fonctions de hachage telles que MD5 ou SHA-1 ont été utilisées comme système d'encodage des mots de passe pour permettre leur stockage sécurisé. Le principal avantage de l'utilisation des fonctions de hachage cryptographiques, est qu'elles permettent de ne conserver que les empreintes des mots de passe. Par conséquent, un attaquant qui aurait un accès à la base des profils utilisateurs ne pourrait pas exploiter directement celle-ci, sans avoir au préalablement effectué un travail de brutforce pour récupérer les mots de passe.

4.4. Sécurisation des systèmes basés sur la blockchain.

Une blockchain (chaîne de blocs) est une technologie permettant de stocker et de transférer des données de façon transparente, hautement sécurisée et décentralisée. Les différents nœuds, stockent les informations relatives aux transactions effectuées par les utilisateurs. Ainsi, le système fonctionne à la manière d'une base de données distribuée, où toutes les données sont partagées par les différents utilisateurs, permettant ainsi à chacun de pouvoir vérifier la validité de la chaîne des transactions.

Le premier système de blockchain, a été développé en 2008 par Satoshi Nakamoto [160], pour servir de système de support à la crypto-monnaie Bitcoin. Ce modèle ayant largement fait ses preuves, d'autre crypto-monnaies ont aussi vu le jour. Cependant, les blockchains ont aussi un fort potentiel et peuvent servir de support à de nombreux autres services tels que :

- **Transfert d'actifs** : utilisation monétaire, titres, votes, actions, obligations…)
- **Registre** : utilisation en tant que système de traçabilité des produits et des actifs
- **Les « smart contracts »** : programmes autonomes, qui appliquent automatiquement les conditions et termes d'un contrat, sans nécessiter d'intervention humaine

Les champs d'exploitation sont immenses : banques, assurances, santé et industries pharmaceutiques, supply chain, de nombreux secteurs (agroalimentaire, luxe, commerce international, distribution, aéronautique, automobile…), industrie musicale, énergie, immobilier, système de vote électronique… De façon générale, des blockchains pourraient remplacer la plupart des « tiers de confiance » centralisés (métiers de banques, notaires, cadastre…) par des systèmes informatiques distribués.

La sécurité liée au contrôle des transactions est donc un aspect primordial pour garantir le succès de cette technologie. Le système étant décentralisé et la confiance entre les différents nœuds ne pouvant pas être garantie, il est nécessaire d'assurer que les différentes transactions ne puissent pas être altérées au cours de leurs transferts entre plusieurs utilisateurs. Les fonctions de hachage jouent donc un rôle primordial pour la sécurisation des blockchains. Elles servent précisément à comparer des données en grande quantité, afin de

déterminer si leur contenu n'a pas été altéré lors d'un passage dans un autre nœud. Ainsi, les clés de hachage seront toujours identiques tant que les données ne subiront pas de modification. Au contraire, une infime modification aura pour conséquence de générer des clés de hachage différentes lors des étapes de minage.

5. Fonctions de hachage non cryptographique

Les fonctions de hachage non cryptographique sont majoritairement utilisées au sein de structures telles que des tables de hachage. À partir d'une chaîne de caractère fourni en entrée, elles calculent et renvoient en sortie, une clé de hachage, sous forme d'un nombre entier. À l'instar des autres fonctions de hachage, la principale propriété recherchée, est que les sorties soient distribuées uniformément au sein de l'intervalle des sorties possibles. En revanche, contrairement aux fonctions de hachage de type cryptographique, ce type de fonction de hachage n'est pas conçu pour résister efficacement à des attaques par recherche de collisions. L'absence de mécanisme contribuant à la sécurisation de ces fonctions, tel l'effet avalanche, en font le type de fonction de hachage le plus rapide.

L'état de l'art autour de ce type de fonctions de hachage a rapidement progressé au cours de ces vingt dernières années. Ainsi, la première fonction de hachage non cryptographique, fût développée par *Jenkins et al.* [161] en 1997. Elle avait pour objectif d'accélérer la recherche au sein de table de hachage. B. Jenkins, proposa plusieurs évolutions de cette fonction, telle que la fonction Lookup3. Par la suite, une seconde génération de fonction non cryptographique est arrivée. Ainsi, Austin-Appleby proposa en 2008, MurmurHash [162], une fonction de hachage plus rapide que Lookup3, proposant des clés de hachage encodées sous 32 ou 64 bits. MurmurHash est ainsi rapidement devenu populaire grâce à son excellente vitesse d'exécution et ses propriétés statistiques. Elle fait encore actuellement office de référence. En 2011, une troisième génération de fonction de hachage est apparue. Ainsi l'entreprise Google publia CityHash [163]. CityHash est une fonction de hachage qui renvoie des clés de hachage d'une taille de 64, 128 et 256 bits. Son algorithme a été fortement optimisé pour tirer parti des instructions CRC32 présentes au sein du jeu d'instructions SSE4 des processeurs Intel, ce qui lui permet d'être deux fois plus rapide que Murmurhash. Parallèlement, Jenkins, [164] publia une fonction de hachage nommée SpookyHash. Cette fonction de hachage renvoie des clés de hachage d'une taille de 128bits et est optimisée pour

les processeurs Intel comportant les jeux d'instruction SSE2. Puis en 2016 Collet [165] proposa xxHash une fonction de hachage qui renvoie des clés de hachage de 32 ou 64 bits et qui a la particularité de pouvoir travailler à une vitesse proche de la mémoire vive.

6. Fonction de hachage en bio-informatique

Au cours du chapitre 2, nous avons introduit les différentes problématiques inhérentes à la comparaison des séquences ADN, dans un contexte d'augmentation exponentielle de la production des données. Ce n'est cependant qu'à partir de la fin des années 90 que la communauté bio-informatique a commencée à utiliser et à développer, des fonctions de hachage plus particulièrement adaptées aux problématiques des séquences ADN. Dans la grande majorité des cas, l'objectif était de pouvoir construire une table de hachage [166], permettant l'indexation d'une base de données, tout en essayant de bénéficier de la rapidité de recherche et de la diminution des données entre les séquences ADN/ARN d'origine et leurs clés de hachage correspondantes.

Concernant les différents types de fonction de hachage en bio-informatique, la littérature fait état de l'utilisation de fonctions de hachage existantes servant généralement à indexer du texte, mais aussi le développement de fonctions de hachage spécifiquement créées pour tenir compte des caractéristiques intrinsèques inhérentes aux séquences ADN/ARN.

6.1. Utilisation de fonctions de hachage existantes

L'utilisation de fonctions de hachage généralistes, semble être une pratique assez répandue, cependant très peu d'articles en font mention. L'hypothèse que nous pouvons en déduire est que, les données génomiques ne sont pas forcément considérées comme étant une catégorie de données à part, mais plutôt comme des données textuelles et donc indexées en tant que telle. Néanmoins, *P. Pandey et al.* [167] décrivent une méthode permettant le comptage de k-mers, qui utilise la fonction de hachage MurmurHASH [A. Appleby, 2012] [162]. Mohamadi et Al., [168] confrontent NTHash, leur algorithme de hachage spécifiquement conçu pour indexer

des séquences ADN/ARN avec les algorithmes de hachage textuels MurmurHash et CityHash [169], et xxHash [165].

6.2. Fonctions de hachage spécifiques

Concernant les fonctions de hachage spécifiques aux séquences ADN, la littérature ne décrit que peu de travaux sur le sujet. Nous pouvons tout de même citer la fonction SSAHA (Sequence Search and Alignment by Hashing Algorithm) [55], dont l'objectif est de proposer un système d'indexation des nucléotides par paire, soit un total de 16 paires possibles et une structure de données en mémoire vive qui contient pour chacune des paires possibles, leurs positions sur les séquences à indexer.

Publié en 2014, GHDNA [170] est une fonction de hachage qui permet l'indexation des séquences ADN en se basant sur des triplets de nucléotides et renvoie des clés de hachage sous forme de nombres entiers. Ainsi GHDNA est une fonction de hachage qui a été développée avec l'objectif de renvoyer des clés de hachage uniformément réparties dans l'intervalle de sortie, avec un nombre de clés de hachage possible de 10^{14}.

La fonction ntHash [168], est une fonction de hachage qui permet l'indexation de séquence ADN, en utilisant le principe du hachage récursif [171]. À l'instar des fonctions de hachage récursives, ntHash a pour particularité de découper les séquences à indexer en différents k-mers. Le calcul des clés de hachage s'effectue en calculant la clé de hachage de chaque k-mers à l'image d'une fenêtre glissante, le long de la séquence à indexer. Ainsi la valeur du k-mers en cours doit prendre en compte la valeur du k-mer précédent. La particularité de cette fonction de hachage est sa rapidité de temps de calcul, ce qui est une des caractéristiques attendues des fonctions de hachage récursives. Ainsi, elle se révèle au minimum 10x plus rapide que des fonctions de hachage classiques telles que CityHash ou MurmurHash. Face à ces concurrents, cela peut donc avoir une importance, puisque durant les tâches de comparaison ou d'alignement de séquences ADN, il est souvent nécessaire de travailler avec la séquence complémentaire inverse de la séquence en cours, ce qui a pour effet de multiplier par deux les données à comparer.

Il est à noter que toutes les fonctions de hachage utilisées actuellement dans le domaine de la bio-informatique ne permettent pas d'établir une relation de correspondance entre deux clés de hachage, mais uniquement la comparaison de séquences exactes.

7. Fonctions de hachage perceptuel

Dans un monde où la publication et l'échange de documents multimédia sont devenus des pratiques courantes, les contenus numériques sont largement produits, distribués et manipulés. De ce fait, le même contenu existe souvent sous diverses formes binaires. Il devient donc difficile de rechercher efficacement certains contenus, avec une grande rapidité et précision. Ce problème est appelé problème d'identification du contenu. Une solution prometteuse à ce problème est le hachage perceptuel. Le principe général des fonctions de hachage perceptuel est d'extraire des caractéristiques robustes et distinctives à partir de données multimédia afin de les utiliser comme identifiants discriminants, convertis en clés de hachage perceptuel compactes. Ces caractéristiques doivent notamment résister à certains types de distorsion volontaires ou accidentelles, tels que le changement des taux d'échantillonnages ou la compression. Elles ne doivent pas varier, tant que le contenu n'est pas modifié de manière significative. L'identification du contenu peut être efficacement réalisée par comparaison des clés de hachage.

Développées à partir du début des années 2000, les fonctions de hachage perceptuel sont notamment utilisées pour effectuer des tâches de tatouage numérique [172], [173] ou dans le cadre de systèmes anti-plagiat pour la détection de diffusion de contenus sous licences commerciales. Elles ont été spécifiquement développées pour l'authentification des documents multimédia (son, image, vidéo). En effet, les fonctions de hachages cryptographiques étant de par nature très sensibles aux contenus binaire, il a donc été nécessaire de développer une famille de fonctions se basant plutôt sur la perception humaine et répondant à différentes contraintes tels que la taille des images ou le taux d'échantillonnage pour un fichier audio. Ainsi, les fonctions de hachage perceptuel adaptées à la problématique des images numériques, se basent sur les caractéristiques visuelles des

données qu'elles doivent traiter. Elles doivent notamment être insensibles aux changements de taille des images, mais aussi aux taux de compression.

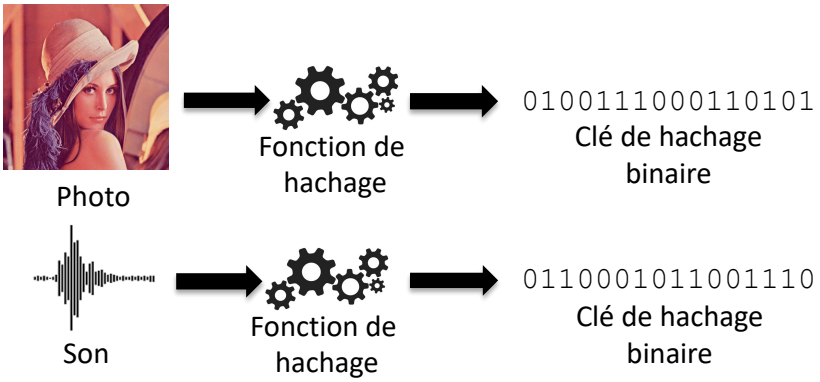


Figure 4.3 : *Principes des fonctions de hachage perceptuels pour des documents de type image ou son*

7.1. Propriétés

Les fonctions de hachage perceptuel, partagent les caractéristiques générales inhérentes à toutes les familles de fonctions de hachage. Cependant, elles se distinguent principalement par le fait qu'elles doivent aussi être accompagnées d'une fonction de vérification, permettant de comparer deux clés de hachage perceptuel et ainsi de déterminer si les valeurs des clés de hachage correspondent au même contenu en fonction de certains critères. Les caractéristiques les plus représentatives d'une image ou d'un son, servent donc d'identifiant discriminant du document et forment la clé de hachage. De ce fait, cette propriété essentielle rend possible la comparaison de documents proches ayant subis différents types d'altération. Les fonctions de hachage perceptuel ne sont donc pas sensibles à l'« Effet Avalanche » et elles sont particulièrement adaptées à la comparaison de documents pouvant subir des perturbations structurelles. À titre d'exemple, une image brute et sa version compressée au format JPEG seraient incomparables au niveau binaire, alors qu'une comparaison par hachage perceptuel permettrait d'établir une certaine notion de similarité entre ces deux images. Les clés de hachage calculées à partir de documents proches, auront la particularité d'être proches à leur tour et pourront être comparées à l'aide des fonctions de vérification. Les

fonctions de vérification calculent généralement une mesure de distance entre deux valeurs de hachage. Cette distance indique la similarité entre les objets multimédia correspondants. Un seuil de distance doit aussi avoir été spécifié afin de déterminer si les objets candidats sont similaires. Les métriques de distance les plus utilisées dans le hachage perceptuel, sont la distance Euclidienne, la distance de Hamming [174] ou la distance de Hausdorff [175][176].

7.2. Fonctions de hachage perceptuel audio

Les fonctions de hachage perceptuel audio ont pour objectif l'identification de documents de type audio. Elles sont utilisées dans le cadre de la recherche, où il est possible d'identifier au sein d'une base de données un extrait sonore fourni en tant que requête [177], et ainsi retourner différentes métadonnées le concernant [178].

Le hachage perceptuel audio possède de nombreuses applications. De façon générale, il est utilisé dans le cadre de la recherche de documents audio identiques ou similaires, ayant subis des variations mineures, ou pour faire de la recherche d'occurrence de séquences particulières, au sein de flux audio. Il peut donc être utilisé pour la reconnaissance de titre musicaux, pour la gestion d'une bibliothèque de fichiers audio [177] ou pour le suivi de diffusion de spots publicitaires.

Les systèmes d'identification musicale utilisant des fonctions de hachage perceptuel se basent sur l'élaboration de signatures ou clés de hachage [177], calculées à partir de caractéristiques acoustiques discriminantes, extraites à partir d'un signal audio [179]. Le calcul de clés de hachage permet ainsi de déterminer si deux extraits musicaux sont proches et ceci même lorsque leurs signaux audio ne sont pas strictement identiques. Ceci peut être le cas lorsque différents taux d'échantillonnage ou différents niveaux de compression sont utilisés [178]. Les clés de hachage doivent donc être résistantes à différents types de modification tels que la compression avec ou sans perte, le filtrage de bandes de fréquences, l'égalisation, le contrôle de la dynamique, la conversion analogique-numérique [178]. Cette robustesse doit permettre de répondre à certaines applications telle que la reconnaissance audio en environnement bruyant, avec l'envoi des échantillons à un serveur depuis un smartphone [180].

Les fonctions de hachage perceptuel audio ainsi que les clés qu'elles génèrent, sont donc l'élément permettant l'identification de documents audio. De plus, à l'instar des autres fonctions de hachage perceptuel les clés générées ont une taille bien inférieure [178].

Au sein des principaux systèmes d'identification musicale audio, les clés de hachage sont calculées à partir des signaux audio des titres indexés [181]. Une fois calculées, les clés de hachage sont conservées au sein de structures de données telles que des tables de hachage, où elles sont associées à des métadonnées concernant le titre musical. Lors de la phase d'identification d'un flux audio, des clés de hachage sont calculées de manière similaire. Elles sont ensuite comparées aux clés présentes au sein de la base de référence et en cas de correspondance, le système renvoie les métadonnées du titre venant d'être identifié.

7.3. Calcul des clés de hachage audio

De façon générale, il est admis que les clés de hachage perceptuel audio sont directement calculées à partir des caractéristiques principales d'un signale audio. La littérature décrit de nombreuses méthodes permettant le calcul de clés de hachage à partir d'un signal audio. Ainsi, des techniques utilisant des méthodes issues de l'analyse des données sont décrites telles que [182], présentant l'utilisation de multiples Analyses en Composantes Principales (ACP) ou des méthodes utilisées classiquement en traitement du signal tels qu'une mesure en platitude spectrale [183], un descripteur basé sur les pics sinusoïdales [184], l'utilisation de fonctions ondelettes pyramidales [185], l'utilisation de Transformée de Fourier [186], ou l'utilisation des Coefficients Mel-frequency Cepstrum (MFC) basée sur une Transformée en Cosinus [187].

L'approche principale est basée sur le découpage du spectre audio en sous-bandes de fréquences. Ainsi, *J. Laroche et al.* [188], décrivent une méthode basée sur la modulation d'énergie du spectre en sous-bandes de fréquences, *J. Pinguier et al.* [181] décrivent une technique utilisant l'analyse de magnitudes du spectre en sous-bandes, *D. Fragoulis et al.* [189], *C. Papaodysseus et al.* [190], une méthode de représentation sous forme d'état binaire

des sous-bandes fréquentielles du spectre, *J. Haitsma et al.* [191], *M. Park et al.* [192] et *C. Bellettini et al.* [193] une dérivée temps-fréquence des énergies en sous-bandes.

7.4. Fonction de hachage perceptuel des images numériques

Au cours du chapitre précédent, il a été question de l'évolution des capacités de production, et d'échange des images numériques. Il a aussi été présenté différentes méthodes permettant d'indexer les images, afin de pouvoir les comparer ou les rechercher au sein de bases de données. Néanmoins, la problématique de la comparaison des images numériques peut aussi être abordée au travers de l'utilisation de méthodes de hachage. En effet, les fonctions de hachage peuvent servir à vérifier l'intégrité ou l'authenticité des images. Cependant, l'utilisation de fonctions de hachage classiques, permet uniquement de pouvoir comparer et identifier comme identiques des images qui le sont de façon stricte, qui se baseront sur la forme binaire plutôt que sur des caractéristiques visuelles.

Nous avons vu dans le chapitre précédent que la problématique de la comparaison des images numériques nécessitait de pouvoir dissocier les aspects autour des formats d'encodage des images (forme binaire) et leurs représentations visuelles. Le processus de comparaison d'images, nécessite de prendre en considération des images ayant des représentations visuelles proches, pouvant avoir subi des modifications ou altérations volontaires ou non. Ainsi, les méthodes de comparaison ne doivent pas être trop sensibles aux modifications tels que la rotation, le changement de taille ou la dégradation dû à certains algorithmes de compression.

Les fonctions de hachage perceptuel font partie de la famille des descripteurs d'images locaux présentés dans le chapitre précédent. Elles ont été développées afin de pouvoir apporter une solution à la problématique de la comparaison d'images proches ou ayant subies des attaques ou dégradations. Elles représentent une sous-partie des descripteurs d'images globaux dont elles reprennent les principaux concepts. En effet, elles utilisent aussi comme mode de comparaison des méthodes de correspondances perceptuelles globales, plutôt qu'une approche de correspondance binaire. La principale différence avec les descripteurs globaux

réside dans le fait que les fonctions de hachage perceptuel ont une approche plus généraliste et qu'elles procèdent généralement à un regroupement de l'information plus important et surtout contraint par la taille des clés de hachage qui doit être la même pour toutes les images indexées.

7.5. Calcul des clés de hachage perceptuel

De façon générale, le processus de calcul des clés de hachage à partir d'une image est basé sur la détection de caractéristiques visuelles, à l'image du fonctionnement du système visuel humain. Ainsi, on distingue trois étapes pour le calcul d'une clé de hachage perceptuel à partir d'une image : l'étape de prétraitement, l'étape d'extraction de caractéristiques visuelles, et l'étape de compression ou de post-traitement. L'objectif est de supprimer les hautes fréquences des images qui contiennent les détails, pour ne retenir que les basses fréquences qui portent l'information structurelle représentative.

7.6. Étape de prétraitement

L'étape de prétraitement a pour objectif de préparer l'image à être hachée, c'est à dire en la calibrant de façon à pouvoir optimiser l'étape d'extraction des caractéristiques représentatives. Cette étape de prétraitement est réalisée via différentes techniques. Ainsi, une méthode courante consiste à redimensionner les images, [194], [195], [196], en une taille prédéfinie, souvent très réduite, (512x512 pixels). L'objectif étant de calibrer l'ensemble des images à indexer, mais aussi de synthétiser les données visuelles et d'alléger les coûts inhérents aux calculs de l'étape d'extraction des caractéristiques représentatives.

Une autre méthode couramment employée est la réduction de l'espace colorimétrique des images [197], [198]. C'est une opération souvent employée par les algorithmes de hachage d'images numériques. Elle consiste en une conversion des images couleurs en niveaux de gris afin de réduire le coût de calcul de l'extraction des caractéristiques visuelles. Enfin la normalisation de l'histogramme de luminosité des images [199], [200] est aussi utilisée. Elle

permet d'accentuer le contraste des images, autorisant ainsi par la suite, une meilleure extraction des caractéristiques visuelles.

7.7. Extraction de caractéristiques visuelles

L'extraction des caractéristiques visuelles robustes est l'une des étapes fondamentales des algorithmes de hachage perceptuel. L'objectif est d'extraire à partir des images des caractéristiques suffisamment représentatives et distinctes pour permettre la génération de clés de hachage uniques, mais en même temps des images perceptuellement proches doivent donner des clés de hachage proches et ceci compte tenu d'altérations ou d'attaques pouvant survenir sur une des images (flou, redimensionnement, compression, ajout de bruit...).

En termes de méthode d'extraction de caractéristiques visuelles robustes, la valeur des différents pixels peut être utilisée. Cependant pour des raisons de temps de calcul et de longueur des clés de hachage, l'intégralité des pixels de l'image, même ayant subi un redimensionnement ne peut être utilisée. Par conséquent, il est nécessaire d'employer des techniques de réduction des dimensions qui permettent néanmoins de préserver les caractéristiques perceptuelles des images.

La littérature décrit aussi l'utilisation de descripteurs basés sur la distribution statistiques de la valeur des pixels, pour en extraire des caractéristiques visuelles. Ainsi de nombreux articles font référence à l'utilisation de méthodes basées sur la moyenne de la luminosité des pixels et des pixels adjacents [201], [197] ou la variance [202], [203]. Les caractéristiques statistiques sont généralement plus robustes que les valeurs de pixels brutes contre les distorsions de bruit, de flou et de compression, cependant elles peuvent aussi avoir pour effet de diminuer les caractères représentatifs des images.

Des méthodes de hachage exploitant des concepts d'algèbre linéaire telle que la factorisation matricielle sont aussi utilisées dans la littérature, une image étant par définition une matrice à deux dimensions. Ceci permet l'extraction de caractéristiques représentatives et tolérantes aux attaques par ajout de bruit, de flou ou de compression. Nous pouvons notamment citer

des travaux utilisant la décomposition en valeurs singulières [204] ou l'utilisation de factorisation par matrices non négatives [205].

Des fonctions de hachage perceptuel se basant sur l'utilisation du domaine fréquentiel ont aussi été décrites. Ces fonctions se basent sur le fait que les coefficients du domaine fréquentiel peuvent servir de caractéristiques représentatives et suffisamment robustes contre de nombreuses manipulations pouvant être faites aux images. Ainsi, la littérature décrit l'utilisation de nombreuses fonctions, telles que la Transformée en Cosinus Discrète (DCT) [206], [207], [208], l'utilisation de transformés en ondelettes discrète (DWT) [209], [210], [211], de transformée de Fourier-Mellin [199], [212], [213], la Transformée de Radon [214], [200], [215] ou de filtres de Gabor [216]. De façon générale, les fonctions exploitant le domaine fréquentiel font partie des techniques les plus utilisées à l'heure actuelle, pour le calcul de descripteurs qui serviront à l'élaboration de clés de hachage perceptuel. L'utilisation du domaine fréquentiel permet une bonne robustesse contre des modifications de type redimensionnement ou distorsion géométrique. Cependant, cette robustesse se fait au détriment de vulnérabilités à d'autres types d'attaques, tels que l'ajout de bruit ou de flou dans l'image.

7.8. Étape de post-traitement

Une des caractéristiques importantes des fonctions de hachage perceptuel est leur capacité à produire des clés de hachage compacts, de quelques centaines de bits de données au maximum. Par conséquent après leurs extractions, les caractéristiques robustes des images doivent subir une étape de compression. Cette étape peut être considérée comme un processus de réduction des dimensions au cours duquel une méthode de compression avec perte doit être appliquée. La difficulté étant d'arriver à préserver les caractéristiques représentatives des images et ceci malgré la phase de compression.

8. Table de hachage

Une table de hachage est une structure de données qui a la particularité de permettre un accès rapide aux différents éléments qu'elle contient. Elle est composée d'une succession de couples clé-valeur qui sont stockés de façon non ordonnée. Les valeurs sont directement accessibles à partir de leurs clés. Les clés sont généralement calculées à partir de fonctions de hachage. Elles sont ensuite utilisées comme index pour identifier le compartiment d'accéder à la valeur d'entrée. Lors de la création d'une fonction de hachage pour calculer les différentes clés constituant une table de hachage, il est important de mettre en place des mécanismes permettant de limiter les collisions entre différentes clés.

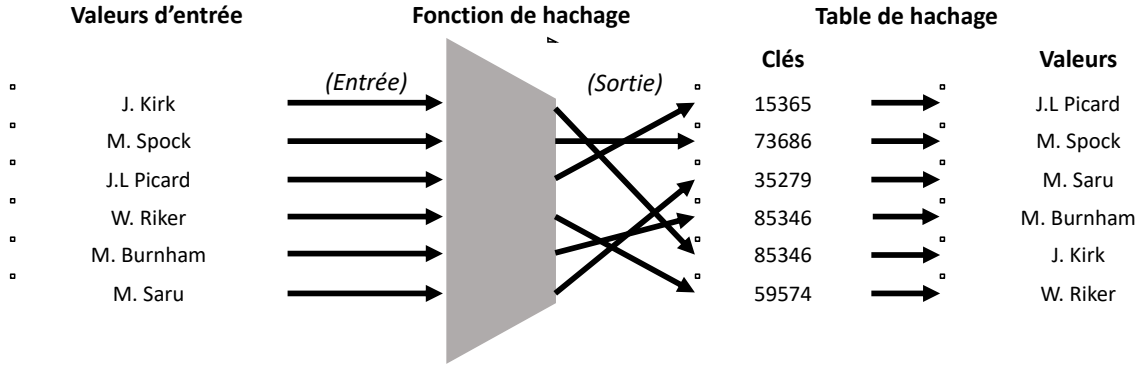


Figure 4.4 – Exemple de l'organisation des données au sein d'une structure de type table de hachage

À l'instar des variables de type tableau, un des principaux avantages des tables de hachage est qu'elles permettent un accès aux différents éléments qu'elles contiennent avec une complexité en temps moyen constant, d'O(1) et ceci indépendamment du nombre d'éléments qu'elles contiennent. Elles s'avèrent être en théorie plus performantes que les structures concurrentes comme les Arbre-B [217], qui ont une complexité d'O(log(N)). Cependant, en fonction du type d'implémentation logiciel réalisé et des caractéristiques de la plateforme sur laquelle elles s'exécutent, le temps d'accès à un élément particulier peut, dans de rares cas les plus défavorables, être d'*O(n)*.

Comparées à d'autres structures de données matricielles associatives, les tables de hachage s'avèrent être très utiles lorsqu'il est nécessaire de stocker un grand nombre d'enregistrements de données tout en ayant la capacité d'y accéder rapidement. Elles sont

notamment employées comme structures de données en mémoire, pour implémenter des tableaux associatifs, des ensembles et des systèmes de caches ou d'indexation au sein de différents moteurs de base de données (SGBDR ou NoSQl). Elles sont utilisées pour accélérer la recherche de chaînes de caractères, ou peuvent également être employées pour une utilisation conjointe avec des structures de données persistantes. Elles peuvent donc s'avérer très performantes en fonction du type de traitement voulu. Dans certains cas, elles ont comme principal inconvénient de ne permettre que des sélections par égalité, étant donné qu'elles ne conservent pas de notion d'ordre.

Conclusion

Le concept de fonction de hachage a été développé autour de l'idée de pouvoir créer des descripteurs représentatifs à partir de données fournies en entrée, en ayant cependant une taille très inférieure. Ainsi, en l'espace d'une cinquantaine d'années, les fonctions de hachage sont devenues incontournables et sont désormais à la base de nombreux mécanismes de sécurité ou d'indexation.

Ce quatrième chapitre a présenté la notion de fonction de hachage. Il a tout d'abord introduit quelques notions théoriques sur les fonctions de hachage et leurs principales propriétés et caractéristiques. Par la suite, il a abordé différents types de fonction de hachage et leurs usages, telles que les fonctions de hachage cryptographique, les fonctions de hachage utilisées pour l'indexation des données textuelles ou en bio-informatique, puis les fonctions de hachage de type perceptuel servant à comparer des sons et des images numériques.

Les différentes notions abordées au sein de ce chapitre et notamment les concepts autour des fonctions de hachage perceptuel sont à la base des développements méthodologiques réalisés durant cette thèse. À l'instar des descripteurs visuels des images numériques décrits au sein du chapitre précédent, les fonctions de hachage perceptuel permettent de comparer des documents de type image ou audio pouvant être perceptuellement proches mais binairement incomparables, du fait de taux de compression ou de format d'encodage différents.

Tableau de synthèse bibliographique

Caractéristiques des fonctions de hachage		[121], [122], [123], [125], [126], [127], [218], [129]
Fonction de hachage	Cryptographique	[130], [131], [132], [133], [134], [135], [136], [137], [138], [139], [140], [141], [142], [143], [144], [145], [146], [147], [148], [149], [150], [151], [152], [153], [154]
	Code d'authentification	[155]
	Signature électronique	[156], [157], [158]
	Conservation des mots de passe	[159]
	Block Chain	[160]
	Non cryptographique	[161], [161], [162], [164], [165]
	Bioinformatique	[166], [167], [168], [169], [55], [170], [171]
	Perceptuel	[172], [173], [174], [175], [176], [177], [178], [179], [180], [181], [182], [183], [184] [185], [186], [187], [188] [189], [190], [191], [192], [193], [194], [195], [196], [197], [198], [199], [200], [201], [202], [219], [204], [205], [206] [207], [208], [209], [210], [211], [212], [213], [214], [215], [216]
Table de hachage		[217]

POINTS CLÉS :

- Le domaine de l'imagerie numérique n'a cessé d'évoluer et il est devenu un élément central des technologies du numérique
- Les méthodes d'acquisition, d'analyse et de traitement sont en constante évolution
- La comparaison d'images est une problématique de base
- Les descripteurs d'images ont pour objectif d'extraire des caractéristiques visuelles à partir des images à indexer, afin d'en calculer des résumés représentatifs et comparables

SECONDE PARTIE – CONTRIBUTIONS

Cette seconde partie a pour objet de présenter les contributions réalisées au cours de cette thèse. S'appuyant sur l'état de l'art réalisé durant la première partie de ce mémoire, elle abordera la problématique de la comparaison de séquences ADN via l'utilisation de méthodes d'indexation et de comparaison d'images numériques.

Le chapitre 5, présentera tout d'abord la fonction de hachage PSH. Ainsi, la première partie de ce chapitre abordera les différentes étapes de calcul de la fonction PSH, puis différentes phases de validations théoriques ainsi que différentes phases d'évaluations à partir de données réelles.

Le chapitre 6 présente la méthode PSC qui permet de rechercher les zones communes entre deux séquences ADN/ARN. Cette méthode, qui est une extension de la fonction PSH utilise les propriétés de corrélation des signes des coefficients d'une TCD. Ce chapitre présentera tout d'abord les différentes étapes de la méthodes PSC puis les phases d'évaluations théoriques ainsi que des expérimentations menées à partir de données réelles.

Chapitre 5 **– FONCTION DE HACHAGE PSH**

RÉSUMÉ :

Ce chapitre présente la fonction de hachage perceptuel PSH (Perceptual Sequence Hashing), qui a été développée au cours de ce travail de Thèse. Dans un premier temps, il décrira les principes généraux de PSH, puis de façon détaillée, les différentes étapes de calcul d'une clé de hachage à partir d'une séquence ADN/ARN. Par la suite, il présentera la méthode permettant de calculer une distance entre deux clés de hachage puis les étapes de validation et les différentes phases de simulation, à partir de données simulées ou réelles, qui ont été réalisées.

SOMMAIRE :

Introduction

Ce chapitre a pour objectif de présenter la fonction de hachage PSH que nous avons développée. PSH, est une fonction de hachage perceptuel, dérivée de fonctions de hachage permettant l'indexation d'images numériques. Elle a été adaptée de façon à pouvoir répondre aux caractéristiques intrinsèques des séquences nucléotidiques ADN ou ARN. L'objectif a consisté à développer une fonction de hachage permettant l'indexation de séquences ADN/ARN, de façon à entrainer une diminution importante de données tout en conservant la possibilité de les comparer. Le processus de calcul des clés de hachage binaires correspondants aux séquences ADN/ARN, consiste tout d'abord à les convertir sous la forme d'une matrice de pixels afin d'en extraire des caractéristiques visuelles à l'aide d'une fonction TCD travaillant dans le domaine fréquentiel. Le processus de comparaison de deux clés de hachage est effectué via une Distance de Hamming, qui permet d'obtenir une notion d'homologie entre deux séquences ADN/ARN.

Dans une première partie, nous présentons de façon détaillée, les différentes étapes de calcul de la fonction PSH, puis la méthode permettant de comparer les clés de hachage entre elles. Par la suite, nous aborderons les aspects autour de la validation, via la présentation d'une étude de probabilité de collision. Nous présentons aussi les tests menés durant les phases de simulations théoriques et enfin, des tests à partir de données réelles utilisant une structure de données de type NoSQL Clé-Valeur que nous avons développée.

1. Intérêts de l'approche

Durant la première partie de ce manuscrit, nous avons pu constater qu'il existait certaines similitudes entre les données de type images et les données génomiques. Tout d'abord, leurs modes de production n'ont cessé d'évoluer depuis plusieurs décennies, ce qui soulève de nombreuses problématiques en termes de stockage, mais aussi et surtout en ce qui concerne leur analyse. En effet, les images numériques et les données génomiques sont toutes issues d'un processus d'acquisition et subissent une transformation au format numérique, avec pour l'une, l'acquisition de la réfraction de la lumière et pour l'autre la lecture d'un enchaînement

de molécules. Une fois numérisées, ceci a pour conséquence la production de documents numériques, composés de sous-unités. Ainsi, une image numérique est un signal discret pouvant être composé de quelques dizaines à plusieurs millions de sous-unités appelées pixels. De même, une séquence ADN/ARN est aussi un signal discret pouvant être aussi composé de quelques dizaines à plusieurs millions de sous-unités appelées nucléotides. Dès lors, les processus d'analyse de base sont semblables, puisqu'ils nécessitent de pouvoir comparer tout ou partie des images ou des séquences ADN. De plus les phases de comparaison doivent aussi prendre en compte des facteurs de dégradation, notamment le redimensionnement ou la compression avec perte, pour les images numériques et les erreurs de séquençage ou les mutations pour les séquences ADN. Cependant l'une des différences majeures réside dans le fait que les images aient deux dimensions, alors que les séquences ADN/ARN n'en possèdent qu'une.

Durant ce travail de thèse, nous sommes partis du postulat émis par *MC. Saldías et al.* [10], que les séquences ADN/ARN pouvaient être encodées sous formes d'images numériques, au sein de matrices de pixels à deux dimensions. Intuitivement, il aurait sans doute été plus naturel d'envisager des approches basées sur des méthodes de comparaison de signaux audio à une dimension. Cependant, *Y. Ke et al.* [220] ont décrit une technique d'identification de musique via une méthode d'analyse d'images, où le signal audio, un signal à une dimension (1D), est converti en plusieurs images à deux dimensions (2D). Cette méthode qui donne des résultats très satisfaisants, démontre l'intérêt d'effectuer une conversion d'un signal 1D en un signal 2D, afin de ne plus se baser sur les caractéristiques d'un signal audio mais sur sa représentation sous forme d'images. Le passage à deux dimensions apportant plus de diversité, ce qui est nécessaire au calcul des points d'intérêts qui servent de bases au calcul des clés de hachage.

Ceci est d'autant plus vrai pour les séquences ADN/ARN, puisqu'elles sont constituées de sous-éléments ne pouvant prendre que quatre états différents (A, T, C, G) correspondants aux quatre types de nucléotides (Adénine, Thymine, Cytosine, Guanine). Nous avons donc opté pour une conversion des séquences ADN/ARN sous la forme de matrices de pixels, en attribuant des tailles arbitraires à chacune des deux dimensions. Ce processus est totalement

réversible puisque lors de la phase de conversion, le nucléotide suivant le dernier d'une ligne se trouve être le premier de la ligne suivante.

2. Fonction de hachage PSH

En nous appuyant sur l'approche de *Saldías et al.* [10], qui consiste à utiliser une méthode de corrélation d'images numériques basée sur une fonction Transformée de Fourier Discrète (TFD) afin de comparer des séquences ADN, nous proposons une nouvelle fonction de hachage perceptuel permettant de calculer des empreintes binaires à partir de séquences ADN/ARN. La fonction PSH a été développée de façon à permettre une diminution des données, tout en conservant la possibilité de les comparer. À l'instar des fonctions de hachage perceptuel, l'objectif était de pouvoir utiliser des structures de type table de hachage pour stocker les données de séquences ADN/ARN indexées via la fonction PSH. Ce type de structure permet par la suite de pouvoir interroger les données, en exploitant la faible complexité algorithmique des tables de hachage.

Afin de pouvoir appliquer la fonction PSH, la séquence que l'on souhaite hacher doit être convertie sous la forme d'une matrice de pixels de taille $N \times M$. La fonction PSH accepte en entrée des séquences de taille W et renvoie des clés de hachage de taille W'. L'objectif est de calculer, pour une séquence S de taille W, une empreinte représentative, au format binaire de taille fixe W', appelée clé de hachage H. Une clé de hachage est directement représentative de la séquence S et sa taille W' est toujours inférieure à W. À l'instar des fonctions de hachage, la fonction PSH n'est pas réversible, ce qui signifie qu'à partir d'une clé de hachage il n'est pas possible de recalculer la séquence d'origine dont elle est issue.

Le principe général de la fonction PSH consiste tout d'abord à extraire des caractéristiques représentatives des matrices de pixels générées à partir des séquences ADN dont on souhaite calculer la clé de hachage. Une fois ces caractéristiques extraites, un algorithme de compression d'images est appliqué avec un fort taux de compression. Ceci a pour effet, d'entraîner une forte dégradation de la matrice contenant les caractéristiques de l'image. Cependant, même fortement dégradée la matrice de pixels résultant de la compression, reste

toujours représentative de l'image originale, donc de la séquence ADN/ARN dont elle est issue. Au final, les clés de hachage générées résultent d'une surcompression de l'image à l'intérieur duquel elles sont encodées.

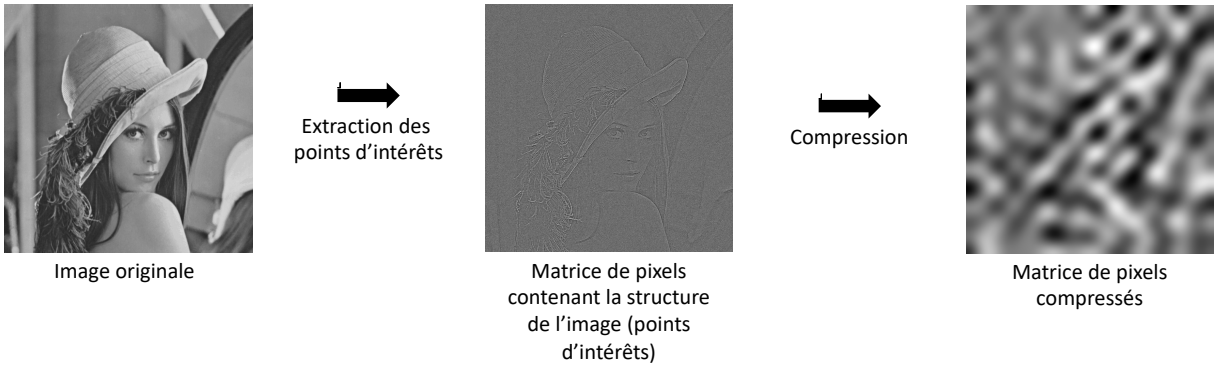


Figure 5.1 – *Processus de fonctionnement de la fonction PSH appliquée à une image représentant une photographie. À partir d'une image (a) est extraite sa structure (b), puis ne sont conservés que les coefficients les plus significatifs au cours du processus de compression (c)*

2.1. Principales étapes de calcul de la fonction PSH

Les étapes de calcul d'une clé de hachage sont illustrées en Figure 5.2. L'algorithme peut être décrit comme suit :

1. Soit une séquence d'ADN/ARN *S* de taille *W*
2. *S* est convertie en une matrice de pixels *P*, de taille *N x M*.
3. Une fonction TCD est appliquée sur la matrice de pixels *P*, afin de faire passer l'information spatiale de l'image dans le domaine fréquentiel appelée coefficients de la TCD et qui sont stockés au sein d'une matrice *P'* de dimensions équivalentes à la matrice *P*.
4. Une fonction *signe* (*sgn*) est appliquée à la matrice *P'*, afin de conserver uniquement les signes des différents coefficients de la TCD.
5. La clé de hachage est formée à partir de tout ou partie des coefficients binaires obtenus durant l'étape 4.

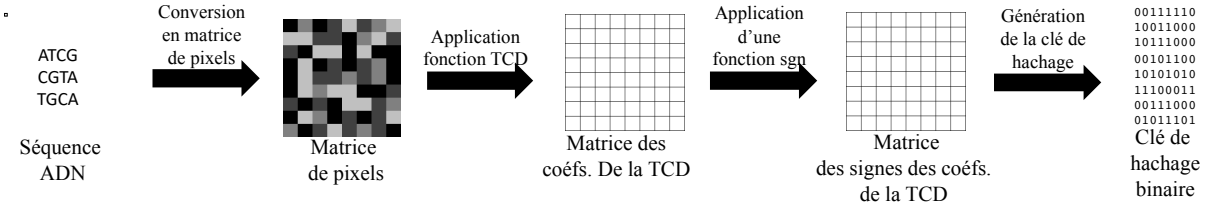


Figure 5.2 – *Principales étapes de calcul de la fonction PSH*

2.2. Encodage des séquences au sein d'une matrice de pixels

La première étape de la fonction PSH est la conversion des séquences ADN/ARN sous forme d'images, ou plus précisément, leur encodage au sein d'une matrice de pixels de taille *N x M* (*Cf. Figure 5.3*). Ainsi, chacun des nucléotides composant une séquence ADN/ARN devient un pixel de l'image. Les pixels sont disposés ligne par ligne en partant du coin supérieur gauche de l'image. Le nucléotide suivant directement le dernier d'une ligne, sera le premier de la ligne suivante.

Il est à noter que cette méthode d'encodage n'a aucun fondement ni justification biologique directe. Cependant, il s'agit d'une façon de réorganiser l'information, tout comme le texte d'un livre est mis en page afin d'occuper un maximum de place sur le papier où il est imprimé.

Ainsi, chaque nucléotide devient un pixel, ayant une valeur d'intensité lumineuse (L_1, L_2, L_3 et L_4), en fonction de son type (*A*, *T* ou *U*, *C*, *G*). Les pixels sont encodés au sein d'une échelle ayant 256 valeurs possibles de niveaux de gris, sur 1 octet. Ce type d'encodage correspond au type « caractère » utilisé par le format FASTA. À ce stade, il n'y a donc aucune diminution ni dégradation de la séquence et le processus d'encodage est une opération totalement réversible.

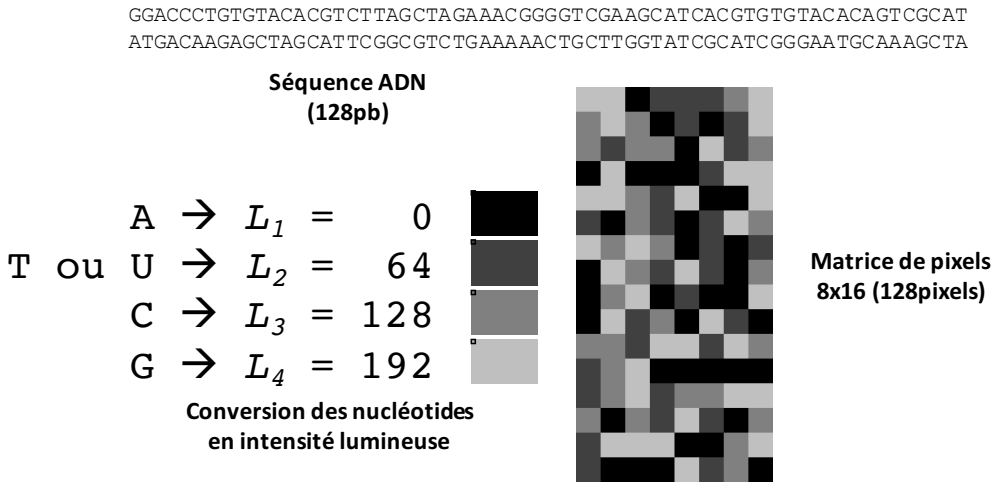


Figure 5.3 – *Conversion d'une séquence ADN/ARN en matrice de pixels*

À partir des séquences ADN/ARN, le processus d'encodage crée des images qui se rapprochent plus d'un signal aléatoire, que d'une image ayant une véritable structure. L'objectif de cet encodage, qui reste arbitraire, est de créer une certaine diversité anthropique qui permettra par la suite l'extraction de points d'intérêts significatifs et représentatifs. À l'instar de *Saldías et al.* [10], nous avons opté pour une approche exploitant le domaine fréquentiel afin de calculer des points d'intérêt. En effet, comme évoqué dans le chapitre 3, la littérature décrit de nombreuses fonctions permettant le calcul de descripteurs d'images. Cependant, étant donné le caractère peu structuré des images produites à partir des séquences ADN/ARN, qui se rapprochent plus d'un signal aléatoire ou d'un bruit blanc, que d'une véritable image structurée, les descripteurs globaux s'appuyant sur le domaine fréquentiel nous ont paru être les descripteurs les plus pertinents pour notre problématique.

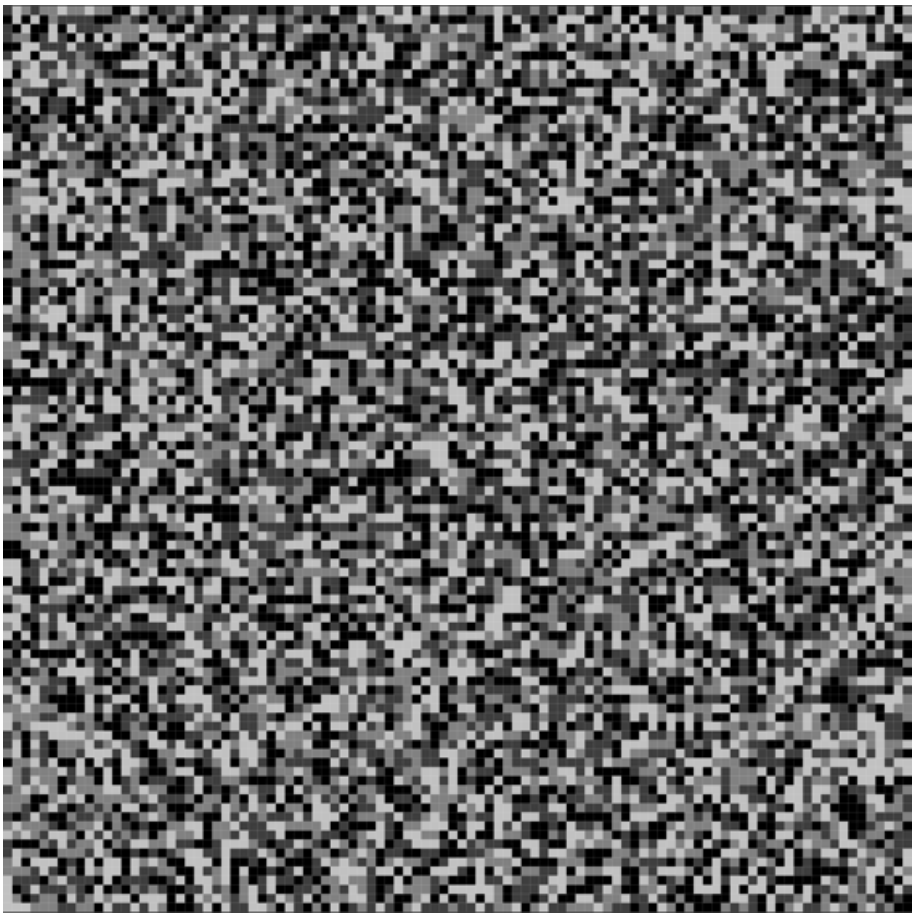


Figure 5.4 – *Séquence ADN encodée en sein d'une matrice de pixels (256x256 nucléotides ou pixels)*

2.3. Seconde étape : Application d'une Transformée de Cosinus Discrète (TCD)

La seconde étape consiste à appliquer une fonction mathématique appelée Transformée en Cosinus Discrète (TCD), sur la matrice de pixels. La fonction TCD permet de faire passer les données depuis le domaine spatial vers le domaine fréquentiel. La fonction TCD est réversible via sa fonction inverse (TCDi), ce qui n'entraine à ce stade aucune perte d'information. La TCD autorise simplement un changement de domaine d'étude, tout en gardant la même fonction étudiée.

2.3.1. Généralités sur la TCD

Initialement publiée en 1974 par *N. Hamed et al.* [221], la Transformée en Cosinus Discrète (TCD) (*Cf. Équation 4.1*) est une transformation mathématique proche de la Transformée de Fourier Discrète (TFD) [222], car elles produisent des fonctions similaires. Elles décomposent un vecteur à temps discret de longueur finie en une somme de fonctions de base, mises à l'échelle et décalées. La différence entre les deux est le type de fonctions de base utilisé par chaque Transformée ; la TFD utilise un ensemble de fonctions exponentielles complexes liées harmoniquement, produisant ainsi des coefficients de type complexes. Tandis que la TCD utilise uniquement des fonctions cosinus et renvoie des coefficients de type réels. Durant ce travail, nous nous sommes plus particulièrement focalisés sur une TCD à deux dimensions. La TCD 2D, possède des propriétés intrinsèques telles que le regroupement fréquentiel, qui la rendent mieux adaptée à certaines opérations de traitement du signal et notamment au niveau des traitements réalisés sur des images numériques.

(a)

Image de référence « Lena »

Domaine spatiale

Taille 512x512 pixels

(b)

Image de référence « Lena »

Domaine fréquentiel

Taille 512x512 pixels

Figure 5.5 – *Image de référence « Lena » (*256 niveaux de gris)*, dans le domaine spatial (a) et dans le domaine fréquentiel après l'application d'une TCD (b)*

Lorsque la fonction TCD est appliquée à une image (Cf. Figure 5.5), sa représentation fréquentielle est obtenue dans une matrice appelée matrice des coefficients fréquentiels. La matrice des coefficients fréquentiels a les mêmes dimensions que la matrice de pixels originale, dont elle est issue. Au sein d'une image, les fréquences représentent les variations

de l'intensité des pixels. La TCD a pour caractéristique principale de regrouper les hautes fréquences dans la partie supérieure gauche de la matrice des coefficients. Les hautes fréquences portent l'information relative à la structure des images, c'est à dire, les changements d'intensité rapides qui correspondent aux contours des formes. Tandis que les basses fréquences qui correspondent aux changements d'intensité lents, portent les zones homogènes.

2.3.2. Équations

La TCD 2D se calcule à partir d'une matrice *N x M*. L'Équation 5.1, présente la relation mathématique qui lie chacune des valeurs de la matrice des *pixels(x,y)* à la matrice DCT*(i,j)* des coefficients de la TCD.

$$DCT\ (i,j) = \frac{1}{\sqrt{2}} c(i)c(j) \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} pixel(x,y) cos\left(\frac{(2x+1)i\pi}{2N}\right) cos\left(\frac{(2y+1)j\pi}{2M}\right)$$

$$c(x) = \frac{1}{\sqrt{2}} \text{ si x vaut 0 et 1 si x > 0}$$

Équation 5.1 : *Définition de la fonction Transformée en Cosinus Discrète (TCD), pour une matrice N x M*

2.3.3. Caractéristique de la TCD 2D

La fonction TCD possède trois caractéristiques principales :

Génération de coefficients réels : Lorsque la TFD utilise un ensemble de fonctions exponentielles complexes et produit des coefficients sous forme de nombres complexes, la TCD implique uniquement l'utilisation de Cosinus et produit des coefficients réels. Par conséquent, la TCD peut se révéler d'une complexité algorithmique inférieure à une TFD.

Complexité algorithmique : Concernant la complexité algorithmique de la TCD, une implémentation naïve et directe de l'Équation 5.1, induirait une complexité en $O(n^2)$. Cependant, une simplification de l'équation, via la création d'une matrice C, de la transformée de cosinus (Équation 5.2), puis une multiplication matricielle entre C, la matrice des pixels et la transposée de C notée C^T, permettent ainsi de réduire la complexité en O(n).

$$C\ (i,j) = \begin{cases} \sqrt[1]{N} & si\ i = 0 \\ \sqrt{\frac{2}{n}}\cos\left(\frac{(2j+1)i\pi)}{2N}\right) & si\ i > 0 \end{cases}$$

$$\mathrm{DCT} = \mathrm{C}(i,j)\ \mathrm{x}\ \mathrm{pixel}(x,y)\ \mathrm{x}\ \mathrm{C}^{\mathrm{T}}$$

Équation 5.2 : *Définition de la fonction Transformée en Cosinus Discrète (TCD) Type II, optimisée*

Où C(i,j) est la matrice de la transformée de cosinus et C^T est la matrice transposée de C

Fonction réversible : La TCD permet de faire passer des données depuis le domaine spatial vers le domaine fréquentiel. Elle autorise un changement de domaine d'étude, tout en gardant la même fonction à étudier. De même, il est possible de réaliser le passage du domaine fréquentiel au domaine spatial, sans perte d'information, via une fonction TCD Inverse (TCDi) (Cf. Équation 5.1).

$$pixel(x,y) = \frac{1}{\sqrt{2N}} \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} c(i),\ c(j) DCT(i,j) cos\left(\frac{(2x+1)i\pi}{2N}\right) cos\left(\frac{(2y+1)j\pi}{2M}\right)$$

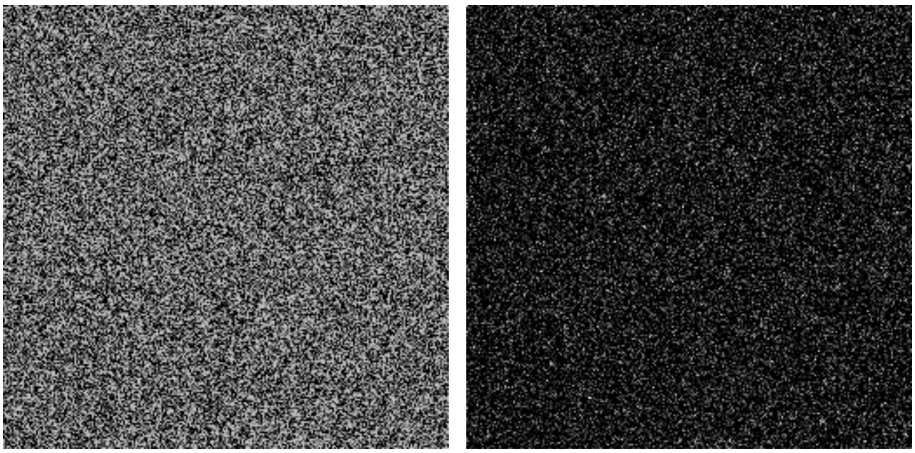
$$c(x) = \frac{1}{\sqrt{2}} \text{ si x vaut 0 et 1 si x > 0}$$

Équation 5.3 – *Définition de la fonction Transformée en Cosinus Discrète Inverse pour une matrice de coefficients N x M*

Concentration de l'information : Les valeurs de l'ensemble des pixels d'une image numérique admettant une grande continuité, lors du passage du domaine spatial vers le domaine fréquentiel, la TCD permet de pouvoir réorganiser l'information sous la forme de coefficients

fréquentiels. Dans le cadre de son application sur une image numérique, elle permet de distinguer les zones où les pixels sont homogènes, appelés aussi basses fréquences et les hautes fréquences représentant les brusques changements de valeur des pixels voisins, caractéristiques des contours des objets ou des formes de l'image. De façon générale, la séparation des hautes et des basses fréquences permet en fonction des traitements voulus sur une image de faire un tri sélectif. Cette notion de séparation des fréquences a été l'idée de base ayant conduit au développement de plusieurs algorithmes de compression d'image JPEG [223] et de compression de vidéo MPEG [224]. En effet, l'œil et l'oreille humaine étant peu sensibles aux basses fréquences, il devient alors possible d'en éliminer une partie afin de diminuer la taille des données, tout en conservant une qualité acceptable des documents.

2.3.4. Application sur des images de type ADN/ARN



(a) Matrice de pixels (256x256)

(b) Matrice des coefficients de la TCD (256x256)

Figure 5.6 – *(a) Image représentant une séquence ADN, (b) Image représentant l'application d'une TCD sur l'image (a)*

Contrairement aux images représentant des photographies, les matrices de pixels générées à partir de séquences ADN/ARN ont un aspect homogène (Cf. Figure 5.6), ce qui entraîne une faible structure, aucune structure géométrique n'étant dominante. Ceci se traduit, lors de l'application de la TCD, par une matrice des coefficients ne faisant ressortir aucune plage de

fréquence particulière. Cependant, le passage dans le domaine fréquentiel apporte suffisamment de diversité pour permettre l'élaboration de clés de hachage représentatives des séquences ADN/ARN dont elles sont issues.

2.4. Seconde étape : Récupération des signes des coefficients de la matrice des coefficients

Cette étape a pour objet d'extraire la structure représentative de l'image de départ. Elle est calculé à partir de la matrice des coefficients de la TCD, à laquelle on applique une fonction *sgn()* afin de ne conserver que les signes des coefficients.

$$sgn\big(TCD(i,j)\big) = \begin{Bmatrix} 0\ si\ TCD(i,j) < 0 \\ 1\ si\ TCD(i,j) > 0 \end{Bmatrix}$$

Équation 5.4 : *Définition mathématique de la fonction sgn*

Cette méthode est notamment décrite par *H. Stark* [225] dans l'ouvrage « Image recovery : Theory and Application », sous la dénomination de « Sign-Only Synthesis » (SOS). Le concept de SOS est applicable à de multiples fonctions permettant la représentation fréquentielle d'un signal telles que la Transformée de Hadamard, la Transformée de Fourier ou la Transformée de Karhunen-Loeve. De façon générale, il est défini comme étant la Transformée Inverse de la matrice binaire des signes des coefficients.

Les signes des coefficients de la TCD, ont un rôle essentiel car ils sont toujours porteurs de l'information structurelle de l'image et ceci malgré la suppression des valeurs des coefficients fréquentiels. Cette étape n'est pas réversible, étant donné l'importante suppression des données dû à l'unique conservation des signes des coefficients.

Figure 5.7 – *Image obtenue après application d'une méthode SOS*

2.5. Troisième étape : Création des clés hachage binaire

Les clés de hachage sont calculées à partir de la matrice des signes des coefficients. Par conséquent, elles sont nativement au format binaire. Le nombre de coefficients binaires étant directement en relation avec le nombre de coefficients de la TCD, qui sont eux-même directement issus du nombre de pixels de l'image d'origine ; la taille maximale d'une clé de hachage ne peut donc pas être supérieure à celle-ci. En fonction de la taille que l'on souhaite donner aux clés de hachage, il est possible de ne conserver qu'une partie des coefficients binaires. Dans ce cas, on notera que plus une clé de hachage a une taille réduite, plus sa probabilité de collision augmente et moins elle contient d'informations structurelles représentatives de l'image d'origine.

2.6. Diminution théorique des données

Étant donné le mode de calcul des clés de hachage, nous pouvons en déduire que la fonction PSH entraine une diminution importante des données. En effet, si la clé de hachage est formée à partir de tous les coefficients de la matrice binaire, elle encodera chaque nucléotide sur un seul bit de données. De plus, à l'instar de l'algorithme de compression JPEG, il est possible de ne garder qu'une partie des coefficients de la matrice binaire pour générer la clé de hachage. Une clé de hachage encodée sur 64bits (8 octets) sera donc 32x moins volumineuse que la séquence de 256pb dont elle est issue.

Taille de la séquence	Encodage d'un nucléotide	Taille de la clé PSH	Ratio
64 pb	1 octet (FASTA)	64 bits	8x
64 pb	2 bits (Binaire)	64 bits	2x
256 pb	1 octet (FASTA)	64 bits	32x
256 pb	2 bits (Binaire)	64 bits	8x

Tableau 5.1 – *diminution des données entre une séquence et sa clé de hachage PSH*

2.7. Comparaison des clés de hachage

Une des caractéristiques essentielles de la fonction PSH, est qu'il est possible de déterminer une distance entre deux clés de hachage. Ceci est dû à la manière dont elles sont calculées, car, malgré le fait qu'elles soient au format binaire, elles contiennent toujours une partie de l'information structurelle de la matrice de pixels (image), dont elles sont issues (propriété de représentativité). Afin de calculer la distance entre deux clés de hachage, la distance de Hamming [226] est utilisée. La Distance de Hamming est la distance de référence pour comparer deux chaines binaires de tailles identiques. Elle exprime la somme des différences entre deux séquences de même longueur (Cf. Équation 4). Les séquences peuvent être des suites de nombres binaires mais aussi se composer d'éléments provenant d'autres systèmes numériques ou alphanumériques.

Pour calculer la Distance de Hamming entre deux séquences binaires, son implémentation ne nécessite que l'utilisation d'instructions de très bas niveau, en standard dans tous les processeurs modernes (XOR et PopCount). Nous avons vu précédemment que les clés de hachage issues des séquences sont générées au format binaire. La distance de Hamming renvoie donc un indice de distance : plus cet indice est faible et plus les séquences sont similaires. En revanche, la distance de Hamming entre deux clés de hachage, n'est pas assez précise pour permettre de déterminer le taux de mutation entre deux séquences.

$$d(a,b) = \sum_{i=0}^{n-1} (a_i \oplus b_i)$$

Équation 4.5 : *Définition de la Distance de Hamming entre deux séquences binaires a et b. $\oplus$ désigne l'opérateur OU exclusif*

3. Évaluations de la fonction PSH

Différentes phases d'évaluations théoriques et phases d'expérimentations ont été menées sur la fonction PSH. L'objectif a tout d'abord été d'étudier sa stabilité au travers de différents aspects, telle que l'étude de la distribution des distances de Hamming en fonction du taux de mutation entre deux séquences, l'étude de la distribution statistique des clés de hachages au sein de l'intervalle des valeurs possibles et l'étude de l'évolution des probabilités de collision en fonction du nombre de clés de hachage. Par la suite, différentes expérimentations ont été menées à partir de données simulées. Elles ont permis d'étudier la vitesse de calcul de clés de hachage, l'influence du système d'encodage des nucléotides au sein de la matrice de pixels, puis les capacités de comparaison des clés de hachage via l'évolution des Distance de Hamming en fonction du taux de mutation entre deux séquences.

Durant les phases d'expérimentations, nous avons paramétré la fonction PSH afin qu'elle renvoie des clés de hachage d'une taille de 64 bits. La taille des séquences ADN utilisées ainsi que le système d'encodage des nucléotides au sein des matrices de pixels sont des paramètres qui seront précisés pour chaque expérimentation.

3.1. Distribution statistique des clés de hachage

L'un des éléments qui concourt à la validation d'une fonction de hachage est l'intervalle de valeur au sein duquel les clés de hachage sont générées et leurs distributions statistiques au sein de cet intervalle. Ainsi, une fonction de hachage équilibrée, aura une distribution statistique des clés de hachage homogène sur tout l'intervalle des valeurs possibles. La surreprésentation d'un sous-intervalle pourrait entraîner une probabilité plus forte que deux clés de hachage entrant en collision.

Cette phase d'évaluation des distributions statistiques a consisté à générer aléatoirement un million de séquences ADN, puis les clés de hachage binaires correspondantes. La Figure 5.8 représente le graphe de densité du nombre de clés de hachage triées sur tout l'intervalle possible. La *Figure 5.9* représente la répartition de chacune des valeurs des clés de hachage au sein de l'intervalle.

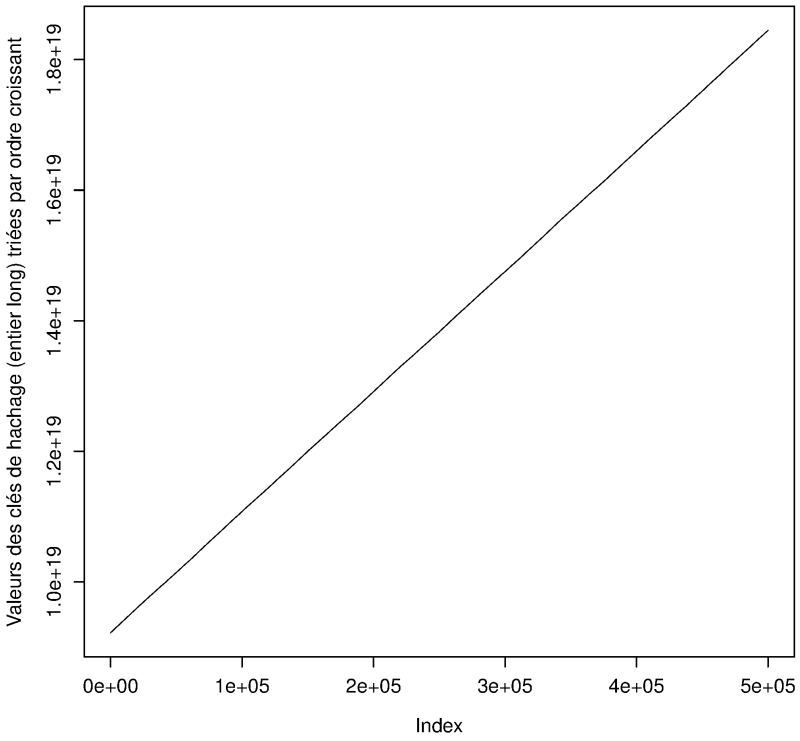


Figure 5.8 – *Représentation graphique des valeurs de 500 000 clés de hachage, triées par ordre croissant*

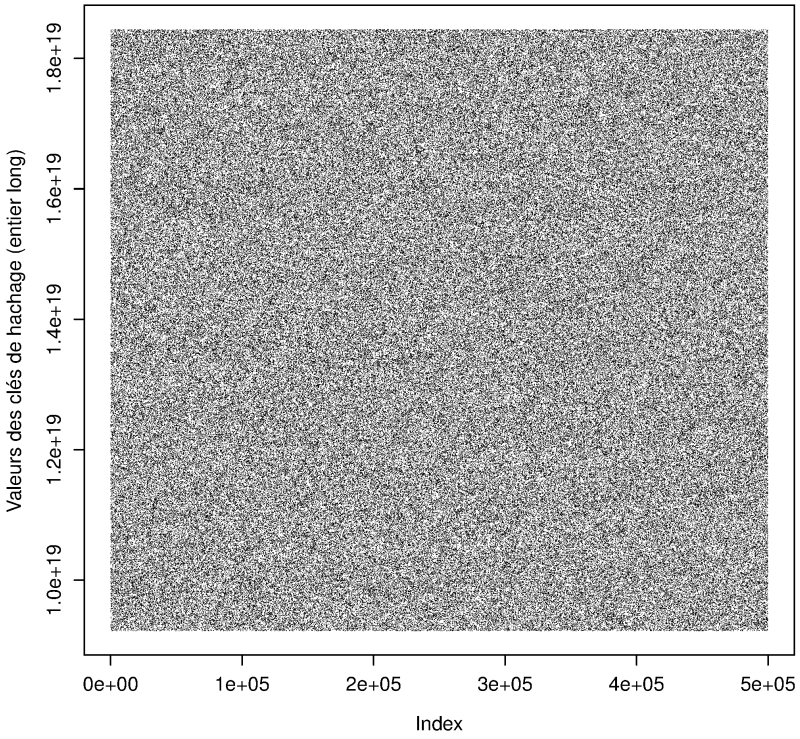


Figure 5.9 – *Distribution statistique des valeurs de 500 000 clés de hachage*

Les résultats observés pour cette phase d'évaluation montrent que les clés de hachage renvoyées par la fonction PSH à partir de séquences ADN, ont une distribution statistique uniforme et qu'aucun intervalle n'est surreprésenté ou sous-représenté (Cf. Figure 5.8). Cette caractéristique est essentielle pour une fonction de hachage et permet de garantir l'uniformité sur laquelle s'appuiera le calcul des probabilités de collision.

3.2. Probabilité de collision

Lors du développement d'une fonction de hachage, qui par définition n'est pas injective, il est nécessaire de faire une étude de probabilité de collision [227]. Ceci permet de déterminer les limites induites par la taille des clés de hachage calculées et par conséquent du nombre de combinaisons possibles dans l'espace des clés. Le terme collision désigne pour une fonction de hachage, le fait d'obtenir une clé de hachage identique, mais calculée à partir de deux

documents strictement différents. Cette probabilité est calculée via une loi de probabilité appelée le Paradoxe des Anniversaires [228]. La Figure 5.10, montre la probabilité d'avoir une collision en fonction du nombre de clés de hachage données.

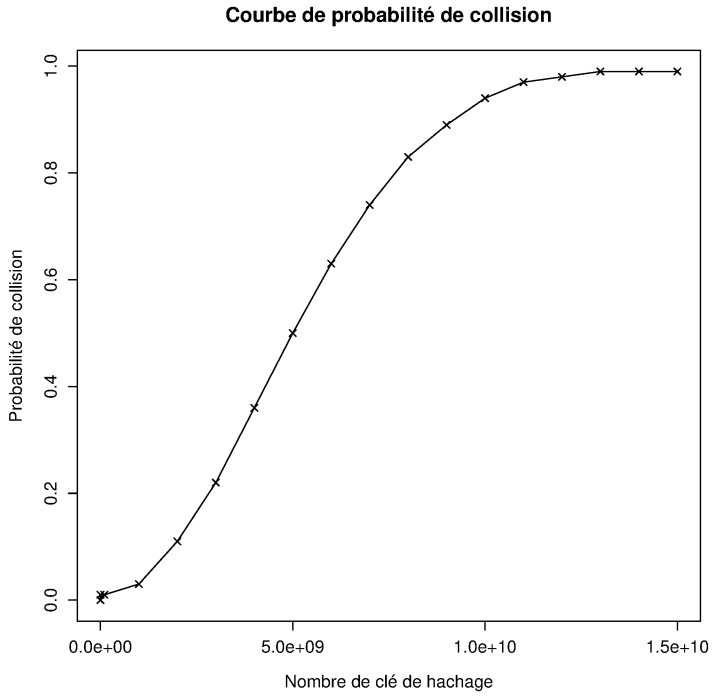


Figure 5.10 – *Probabilité (en ordonnée), d'avoir une collision en fonction du nombre de clés de hachage (en abscisse).*

L'étude de probabilité de collision, a démontré qu'en renvoyant des clés de hachage de 64 bits, PSH avait une probabilité de 0,5 de générer deux clés identiques à partir de deux séquences différentes pour un total de 5 milliards de clés de hachage.

3.3. Temps de calcul

Au cours du développement d'une fonction de hachage, le temps nécessaire pour le calcul des clés de hachage doit être pris en considération. Dans le cas des fonctions de hachage perceptuel, des temps de calcul trop longs risqueraient de pénaliser leur utilisation au sein

d'un système d'indexation. En revanche, une fonction de hachage dont les temps de calcul seraient trop rapides, risquerait de faciliter des attaques par force brute.

Afin d'étudier les temps de calcul de la fonction PSH, nous avons procédé au calcul de différentes clés de hachage par groupes de 1 milliard (groupe A à groupe I).

Chacun des 10 groupes de simulations a été lancé sur deux machines différentes :

- Machine A : station de travail doté d'un processeur Intel™ Core™ i7-9720HQ, 4 Cores / 8 Threads @ 3,1Ghz
- Machine B : serveur de calcul doté d'un processeur Intel(R) Xeon(R) CPU E7-4850 @ 2.30GHz, 48 Cœurs / 96 Threads

Le tableau 5.2, présentes les différents paramètres utilisés pour les tests de performance et le tableau 5.3, les temps de calculs obtenus par groupe et sur chacune des deux machines (A et B).

Groupes	A	B	C	D	E	F	G	H	I
Taille des séquences ADN	64pb	128pb	256pb	512pb	1024pb	2048pb	4096pb	8160pb	16384pb
Taille des clés de hachage	64bits	128bits	256bits	512bits	1024bits	2048bits	4096bits	8160bits	16384bits
Taille des matrices de pixels	8x8	8x16	16x16	16x32	32x32	32x64	64x64	64x128	128x128

Tableau 5.2 – *Paramètre pour les groupes de simulation A à I*

Les résultats présentés, montrent les temps de calcul bruts des clés de hachage et ne prennent pas en compte les temps de génération aléatoire des séquences ADN à partir desquels ont été calculées les clés de hachage.

Groupes	A	B	C	D	E	F	G	H	I
Machine A	1 246s	1 625s	2 271s	3 829s	6 744s	1 1986s	24 721s	48 940s	105 535s
Machine B	542s	588s	699s	958s	1 615s	2 925s	5 602s	10 5535s	57 413s

Tableau 5.3 – Temps d'exécution pour les groupes A à I

4. Simulations théoriques

Après l'étude des caractéristiques de la fonction PSH, nous avons mené différentes phases de simulations théoriques à partir de séquences ADN, générées de façon totalement aléatoire. Ainsi, afin de générer une séquence S_1 de taille n, les différents nucléotides qui la composent ont été générés avec la même probabilité de pouvoir devenir l'un des quatre types de nucléotides possibles.

Concernant le calcul de séquences divergentes S_2 à partir d'une séquence S_1, nous avons utilisé le modèle évolutif Juke et Cantor [229]. Ainsi, chaque nucléotide de S_1 devant subir une mutation a la même probabilité de devenir un des trois autres nucléotides.

4.1. Évolution des Distances de Hamming par rapport aux taux de mutation en séquences ADN

Afin de mettre en évidence la capacité de la fonction PSH à produire des clés de hachage comparable via l'utilisation de Distance de Hamming, nous avons procédé à différents groupes de simulations théoriques. Pour chaque groupe de simulation, un jeu d'un milliard de séquences ADN a été généré de façon aléatoire (séquences primaires) et les clés de hachage correspondantes ont été calculées. Puis à partir des séquences primaires, des séquences divergentes ont été générées ainsi que les clés de hachage correspondantes. À une séquence primaire correspond donc 6 séquences divergentes, ayant des taux de mutation de 2%, 5%, 10% 20%, 30% et 50%.

Les phases de simulation ont consisté à calculer les Distance de Hamming entre chacune des séquences primaires et les 6 séquences divergentes correspondantes. L'objectif était de pouvoir étudier les distributions statistiques des Distances de Hamming, en fonction des tailles de séquences (entre 64pb et 1024pb) et des tailles des clés de hachages calculées (entre 64bits et 128bits).

Ainsi, pour cette phase de simulation, 10 groupes ont été constituées pour un total de 10 milliards de séquences primaires, 60 milliards de séquences divergentes et 60 milliards de comparaisons. Ces chiffres ont été sélectionnés pour être statistiquement représentatifs mais aussi pour que les différents groupes de simulation puissent s'exécuter avec des temps de calcul ne dépassant pas quelques heures.

Le tableau 5.4 présente les différents groupes avec les paramètres retenus pour chacun des groupes.

Groupes	A	B	C	D	E	F	G	H	I	J
Taille des séquences ADN	64pb	128pb	256pb	512pb	1024pb	64pb	128pb	256pb	512pb	1024pb
Taille des matrices de pixels	8x8px	8x16px	16x16px	16x32px	32x32px	8x8px	8x16px	16x16px	16x32px	32x32px
Taille des clés de hachage	64bits	64bits	64bits	64bits	64bits	128bits	128bits	128bits	128bits	128bits

Tableau 5.4 – *Paramètres pour les groupes de simulation théoriques A à J*

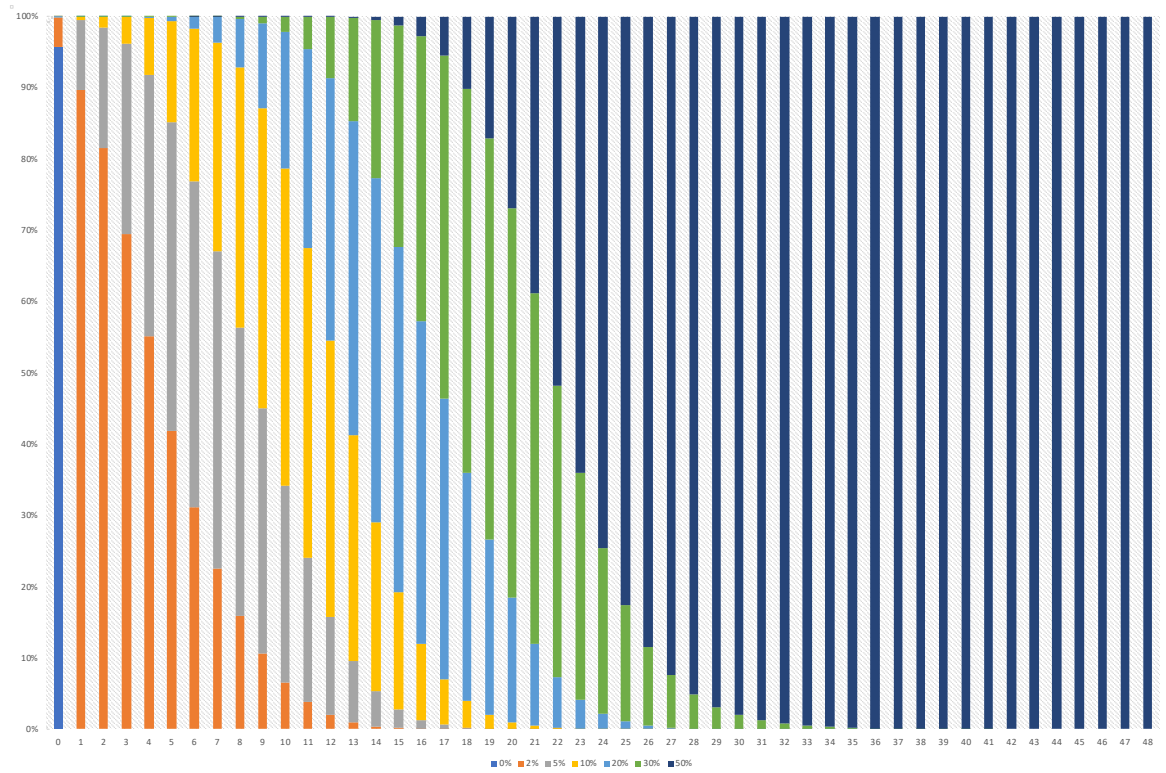


Figure 5.11 – *Distribution en % des taux de mutations (ordonnée) pour chaque distance de Hamming (abscisse) pour le groupe A*

L'étude de distribution statistique des distances de Hamming entre deux clés, en fonction des taux de mutations, a permis d'illustrer la propriété de représentativité de la fonction PSH. La Figure 5.11, montre une relation forte entre la Distance de Hamming et les taux de mutation des séquences dont elles sont issues. Néanmoins, une Distance de Hamming entre deux clés de hachage, ne permet pas précisément de déduire les taux exacts de mutation entre deux séquences. Les différents graphiques correspondants aux distributions statistiques des distances de Hamming pour les groupes A à I sont présentés.

4.2. Étude de l'influence de l'encodage des nucléotides au sein de la matrice de pixels

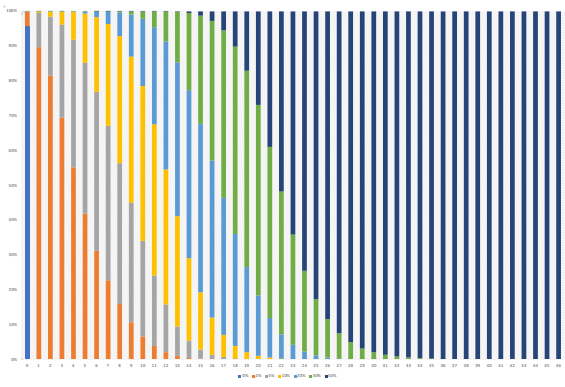
La seconde phase de simulations théoriques avait pour but d'étudier l'influence de l'encodage des nucléotides au sein des matrices de pixels. Nous avons donc cherché à étudier si la façon d'encoder les nucléotides pouvait avoir une influence particulière sur les résultats de la

fonction PSH. En se basant sur la méthodologie de la première phase de simulation, cette seconde phase a consisté à générer aléatoirement 6 groupes supplémentaires (groupe K à groupe P) contenant 1 milliards de séquences ADN et pour chacune des séquences 6 séquences divergentes avec des taux de mutation de 2%, 5%, 10% 20%, 30% et 50%. Concernant la longueur des séquences ADN générées, nous avons opté pour une taille de 64pb, avec une taille de clé de hachage de 64bits. En revanche, pour chaque groupe, les nucléotides composants les séquences ont été encodés avec différentes valeurs d'intensité lumineuse.

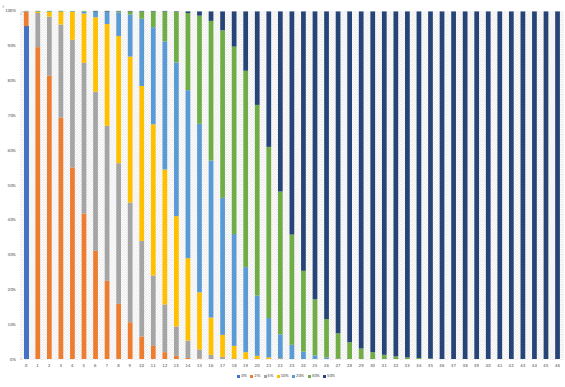
Le Tableau 5.5 présente les différents groupes avec les paramètres retenus pour chacun d'entre eux.

Groupe	K	L	M	N	O	P
Adénine	1	192	1	16	1	1
Thymine	64	128	16	192	2	1
Cytosine	128	64	180	1	3	196
Guanine	192	1	196	176	4	196

Tableau 5.5 – *Valeurs d'encodage des nucléotides pour les groupes K à P*



Groupe K



Groupe L

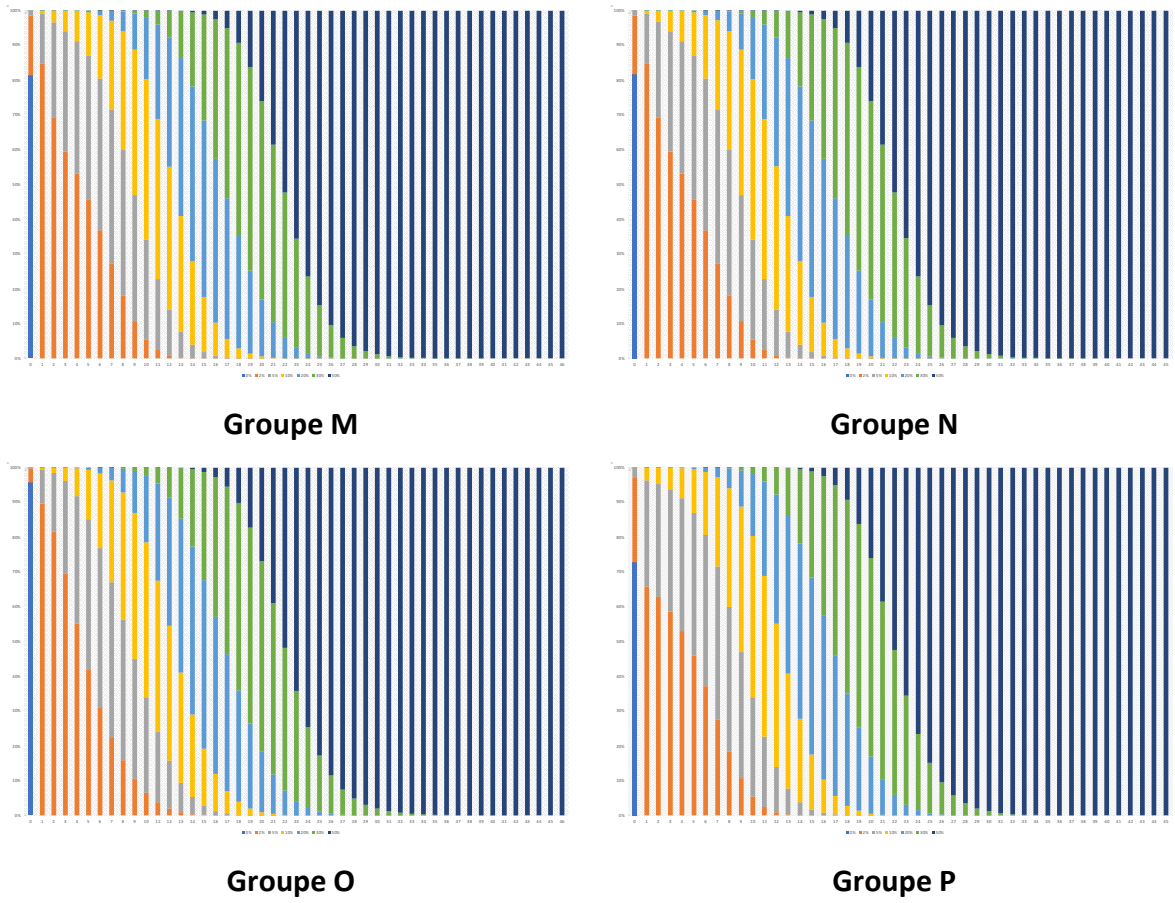


Figure 5.12 : *Distribution des taux de mutation en fonction des distances de Hamming pour les groupes K à P*

Les résultats obtenus durant les phases de simulation K à P, montrent que les valeurs d'intensité lumineuse pour l'encodage des nucléotides au sein des matrices de pixels ont une influence plutôt limitée sur la distribution des distances de Hamming obtenues en fonction des taux de mutation entre les séquences comparées. Cependant, nous pouvons tout de même noter des variations pour les groupes M et N, plus particulièrement pour les distances de Hamming proches de 0, qui apparaissent comme étant moins précises. Ceci peut s'expliquer par le fait que les valeurs d'intensités lumineuses ayant servi pour l'encodage des nucléotides de ces deux groupes n'avaient pas un écart constant. Inversement lorsque les valeurs avaient un écart constant (groupes K, L, O), les distributions des distances de Hamming obtenues semblent plus précises.

Durant cette expérimentation nous avons aussi cherché à étudier la capacité de la fonction PSH à comparer des séquences, où les types des nucléotides complémentaires (A-T et C-G) avaient été encodés avec une seule valeur d'intensité lumineuse par paire de nucléotides. Les résultats observés montrent que pour le groupe *P*, les distributions des distances de Hamming ont un profil se rapprochant des groupes *M* et *N*, avec cependant une perte de sensibilité par rapport à ces deux groupes.

5. Évaluation à partir de données réelles

Les phases d'évaluation de la fonction PSH à partir de données réelles, ont été menées avec l'objectif de se placer dans un cas d'utilisation classique en bio-informatique, qui est la recherche de séquences au sein d'une base de données de séquences de références. Ainsi, nous avons conduit deux phases d'expérimentations. La première phase a consisté à étudier les capacités d'indexation de la fonction PSH puis la recherche de séquences ADN exactes (exact matching) au sein d'une base de données de références. La seconde expérimentation avait pour objectif d'évaluer la recherche de séquences proches indexées au sein d'une base de données via la fonction PSH. En complément des capacités de recherche, nous avons aussi cherché à observer le volume des données obtenu après indexation. En effet, l'un des objectifs de PSH était d'apporter une importante diminution des données et nous avons cherché à évaluer cette diminution dans un contexte de cas d'utilisation standard.

Afin de préparer les différentes expérimentations, il nous était nécessaire de trouver une structure de données permettant le stockage des clés de hachage. Compte tenu de la simplicité de notre structure de données, de type table de hachage, nous avons opté pour une solution de type NoSQL Clé-Valeur. Les bases de données clé-valeur, représentent la catégorie des bases de données NoSQL ayant la structure de données la plus minimaliste, où l'on associe une ou plusieurs valeurs à une clé unique. Elles ont été conçues avec l'hypothèse que, dans la majorité des cas, les opérations les plus courantes sont des requêtes de lecture à partir d'un identifiant. Elles sont composées de structures de type table de hachage avec des mécanismes de persistance et de répartition sur plusieurs machines, afin de répondre aux problématiques de tolérance aux pannes, mais surtout de montée en charge tant au niveau du volume des

données, qu'au niveau des requêtes effectuées par les clients. Tout comme les bases de données orientées colonne, les requêtes pouvant être effectuées sont uniquement sur les clés mais pas sur les valeurs. Ce type de bases permet de stocker des chaînes de caractères totalement déstructurées ou structurées sous la forme de documents ou d'objets complexes.

Les opérations possibles sur ce genre de bases sont les suivantes :

- Affectation d'une valeur à une clé
- Récupération d'une valeur à partir d'une clé
- Modification d'une valeur affectée à une clé
- Suppression d'une clé et de sa valeur associée

Initialement, notre choix s'est porté sur le moteur NoSQL REDIS (https://redis.io). Ce moteur avait été choisi, car il proposait une solution libre de droit, présentant une structure de type clé-valeur in-memory et des requêtes de type GET avec une complexité algorithmique temporelle en O(1). Cependant, après plusieurs tentatives de création de notre base de données, nous avons constaté l'incapacité de REDIS à stocker des chaînes binaires de façon optimale. Après avoir évalué d'autres moteurs NoSQL Clé-Valeur tels que VoltDB (https://www.voltdb.com), ou Memcached (https://memcached.org), nous avons constaté que la majorité des moteurs de bases de données clé-valeur existants ne proposent pas de stocker les clés sous la forme de chaînes binaires. En effet, les clés doivent être converties en Entiers Longs (encodées sous 64 bits), ce qui rend nativement les calculs de distance de Hamming impossibles, sans devoir effectuer au préalablement des conversions de type de données. Une autre stratégie pourrait consister à stocker les clés de hachage binaire au sein de champ de type « chaîne de caractères », cependant ceci se révèle non optimal en termes d'espace de stockage, puisque chaque bit de donnée est alors stocké sur un octet, entrainant de fait la non utilisation de 7 bits.

L'impossibilité de pouvoir stocker les clés de hachage en tant que chaînes binaires et la difficulté de pouvoir effectuer des requêtes de type Distance de Hamming sur les clés de hachage, ont motivé le développement de PSH-DB, un moteur NoSQL minimaliste, qui reprend les caractéristiques principales de REDIS et ajoute des caractéristiques adaptées.

5.1. Structure de données PSH-DB

PSH-DB (Perceptual Sequence Hashing Data Base), est la structure de données développée conjointement avec la fonction PSH, pour les phases d'expérimentations. Son moteur a été développé autour d'un modèle client-serveur TCP multi-thread, permettant la connexion simultanée de plusieurs clients. Sa structure de données de type clé-valeur, utilise la fonction PSH comme fonction de hachage. Les clés de hachage sont stockées nativement sous le type de données *bitset* proposé par la bibliothèque standard (*std*), du langage *C++11*, pour un stockage optimal des chaines binaires. Le modèle de données repose sur une structure de type *unordered_map* de la bibliothèque standard (*std*) du langage *C++11*. Cette structure a été choisie car elle permet de stocker des données de type *bitset* et elle permet de renvoyer les valeurs associées à une clé avec une complexité temporelle de *O(1)*.

Fonctions	Descriptions
set <*clé*> <*valeur*>	Associe une valeur à une clé
get <*clé*>	Retourne la valeur correspondante à une clé
get_hamming <*clé*> <*distance*>	Renvoie toutes les clés ayant une distance de Hamming inférieur à la distance de la clé passée en paramètre
del <*clé*>	Supprime un couple clé-valeur à partir de sa clé
dbsize	Renvoie le nombre total de clés
flushall	Supprime tous les couples clé-valeur

Tableau 5.6 – *Liste des fonctions principales de PSH-DB*

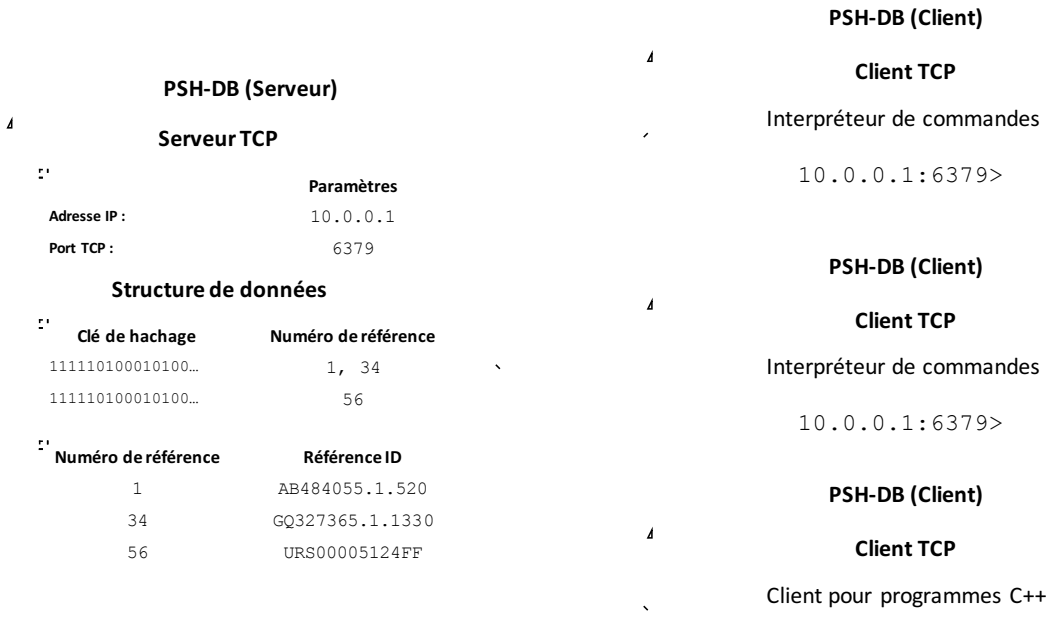


Figure 5.13 – *Structure générale de PSH-DB*

5.2. Paramètres utilisés

Durant les phases de validations théoriques et les expérimentations, les paramètres suivants ont été utilisés :

Taille de séquence :	1024 pb
Taille de sous-séquence :	128 pb
Taille de la matrice de pixels :	8x16
Taille de la matrice des coefficients :	8x16
Taille des clés de hachage	64 bits (8 x 8)

Tableau 5.7 – *Liste des paramètres utilisés pour les expérimentations*

Ces paramètres ont été sélectionnés pour être représentatifs des données utilisées en bio-informatique, où les séquences peuvent varier de plusieurs dizaines, à plusieurs milliers de paires de bases. De plus, la taille des k-mers servant à calculer les clés de hachage a été fixée en fonction de la vitesse de calcul de la fonction PSH et de la diminution des données induite

entre une séquence et sa clé de hachage. La fonction PSH prend alors en entrée une séquence de 128pb et génère une clé de 64 bits (8 octets), ce qui entraine une diminution théorique de 93,75% par rapport au format FASTA.

5.3. Recherche de séquences exactes

La première expérimentation avait pour objectif d'évaluer la sensibilité et les temps d'exécution du système PSH-DB durant des tâches de recherche de sous-séquences exactes. Une base de données, regroupant 3 bases de données publiques d'ARN de référence, a été constituée, pour un total de plus de 17 millions de séquences de référence et 14,6 Go de données (Cf. Tableau 5.8). Toutes les séquences de référence ont été hachées et stockées au sein du moteur PSH-DB. L'objectif était de rechercher dans notre base de données des séquences de référence ayant au moins 128pb en commun avec un ensemble de requêtes composé de 2000 séquences.

Base de données de références	Nombre de séquences	Taille
RDB (https://rdp.cme.msu.edu)	3 070 243	3,6 Go
SILVA RNA (https://www.arb-silva.de)	4 984 666	4,8 Go
RNA Central (https://rnacentral.org)	9 386 122	6,2 Go
TOTAL :	**17 441 031**	**14,6 Go**

Tableau 5.8 – *Séquences de référence utilisées pour l'expérimentation n°1*

Afin d'évaluer la sensibilité de notre système nous avons utilisé l'outil de référence en bio-informatique BLAST, que nous avons paramétré pour qu'il ne recherche que des séquences exactes de 128pb au sein de sa base de données. Néanmoins, il est important de relativiser les effets que pourraient avoir cette contrainte sur les résultats renvoyés par BLAST, car l'une des premières étapes de l'algorithmes de BLAST est de rechercher des sous-séquences exactes. Durant les expérimentations le paramètre word_size a donc été fixé à 128pb, de façon à ce que BLAST recherche des séquences exactes, avec une taille supérieure ou égale à 128pb. Par la suite, afin de comparer notre structure de données PSH-DB en termes de

stockage et de temps d'exécution des requêtes, nous avons effectué la même expérimentation, mais en utilisant le moteur NoSQL Redis.

5.3.1. Indexation des références

Pour procéder à l'indexation des séquences de référence au sein de PSH-DB, chaque séquence a été découpée en sous-séquences ou k-mers, d'une taille de 128pb, avec un décalage de 64pb (Cf. Figure 5.14). Pour indexer l'intégralité d'une séquence, dès lors que sa taille n'est pas un multiple de la taille des k-mers fixée en paramètre (128 pour cette expérimentation), il est nécessaire de définir un k-mer de fin de séquence. Ainsi, pour une séquence de taille de 530pb et des k-mers de 128, le k-mer de fin de séquence débutera à partir de la position 402 (530 – 128).

Pour chaque k-mer une clé de hachage correspondante a été calculée. C'est cette clé de hachage qui est stockée au sein de la structure PSH-DB.

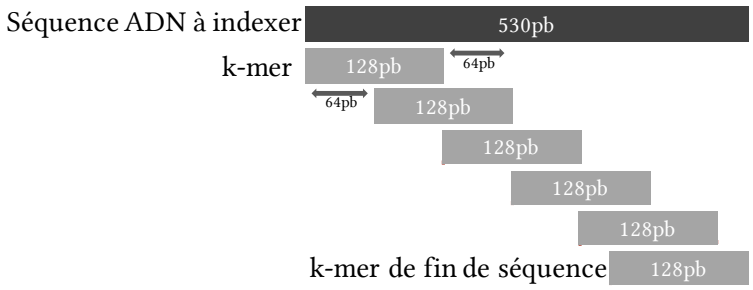


Figure 5.14 : *Processus d'indexation d'une séquence de référence par découpage en k-mers*

5.3.2. Création du jeu de requêtes

Pour cette expérimentation, un jeu de requêtes a été généré à partir des séquences issues de la base de données d'ARN de référence. À partir des séquences de référence de la base de données, 1000 requêtes, d'une taille de 1024pb, ont été extraites aléatoirement et 1000 requêtes supplémentaires de même taille, ont été générées de façon aléatoire.

5.3.3. Interrogation de la base de données

Pour chacune des 2000 séquences requêtes, les différents k-mers possibles, d'une taille de 128pb, ont été calculés à partir du début et jusqu'à la fin de la séquence, avec un décalage d'une paire de bases. Pour une séquence, de taille *W*, le nombre de k-mers de taille *K* possible est donc de *W* – *K*. Les clés de hachage correspondantes aux différents k-mers ont été calculées via la fonction PSH. Ces clés de hachage ont été alors recherchées au sein de la base de données.

5.3.4. Paramétrage de BLAST

Pour cette expérimentation la version de BLAST 2.2.28+ a été utilisée. BLAST a été paramétré de façon à rechercher, des sous-séquences exactes d'une taille minimale de 128pb, avec 100% d'identité, entre les séquences requêtes et les séquences de références de la base de données. Concernant les résultats renvoyés, nous avons utilisé les paramètres par défaut où, BLAST renvoie le top 500 des séquences ayant un plus fort taux d'alignement. Afin de ne pas être pénalisée par les temps d'accès au disque dur, par rapport à PSH-DB, la base de données de BLAST a été placée sur un RAMDisk, de même que les fichiers de sortie de résultats.

5.3.5. Résultats

La première expérimentation, qui consistait à rechercher des k-mers exacts au sein de la base de données, avait pour objectif d'évaluer la fonction PSH, sur des critères de sensibilité, de structure de données et de temps d'exécution.

En termes de sensibilité, la Figure 5.15 et le Tab 5.8 montrent que PSH-DB renvoie en moyenne par requête un nombre de séquences de références beaucoup plus important que BLAST. Ceci s'explique par le fait que BLAST a été paramétré pour ne renvoyer uniquement que les résultats avec une e-value de 0. Concernant PSH, d'après l'étude de collision (cf. Sec.3.2), si

deux clés de hachage sont identiques, la probabilité qu'elles soient issues de deux séquences différentes est très faible.

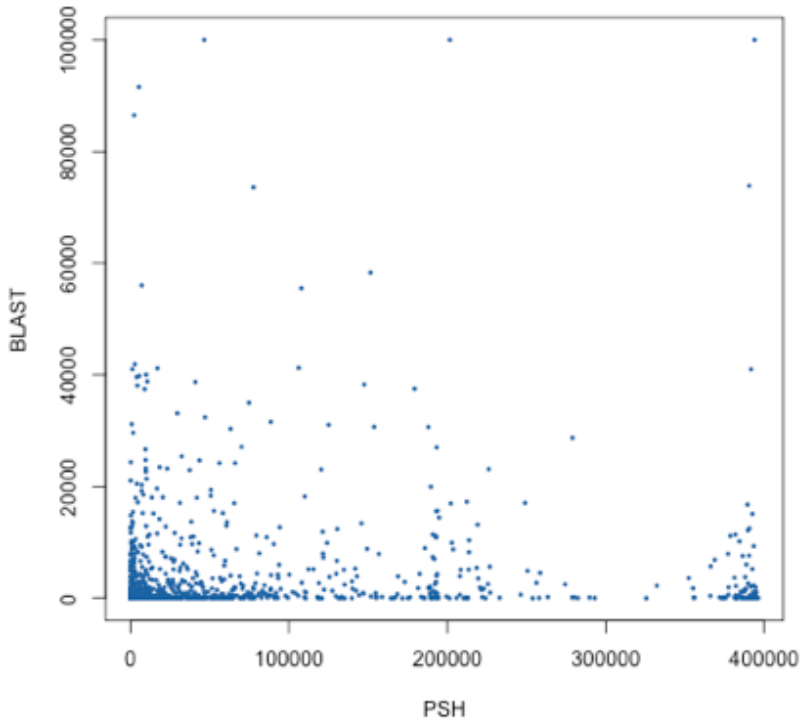


Figure 5.15 : *Comparaison du nombre de séquences de référence renvoyées par PSH (en abscisse) et BLAST (en ordonnée)*

	PSH-DB :	BLAST :	Taux :
Nombre moyen de résultats renvoyés par requête	74 624	4 384	-
Résultats avec requêtes aléatoires	0	0	0%
Résultats communs	291 047	319 951	90,96%
Résultats communs (top 500 BLAST)	53 804	53 819	99,97%

Tableau 5.9 – *Synthèse des résultats renvoyés par BLAST et PSH-DB*

Concernant l'étude des résultats identiques, la Figure 5.16, qui présente le nombre de résultats communs entre PSH et les résultats de BLAST, démontre la capacité de PSH à détecter 99% des séquences du top 500 des résultats de BLAST.

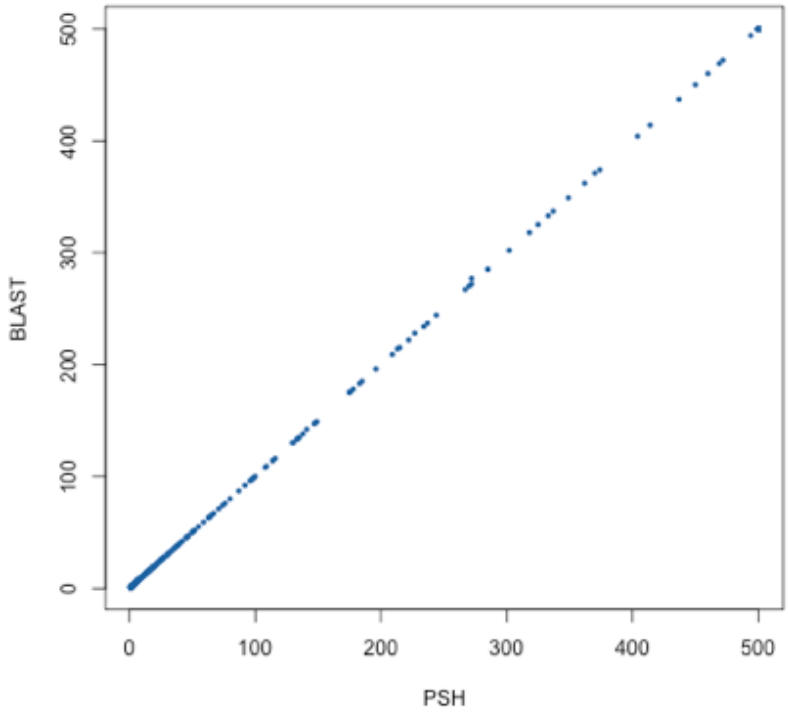


Figure 5.16 – *Nombre de résultats renvoyés par PSH (en abscisse), communs au top 500 des résultats de BLAST (en ordonnée)*

Cette première expérimentation a aussi été l'occasion de comparer la taille des bases de données entre les systèmes BLAST, REDIS et PSH-DB. Pour des structures de données in-memory, la taille des données est naturellement un critère important. Le Tableau 5.9 montre que PSH-DB occupe le plus petit espace mémoire, suivi de BLAST et REDIS. Concernant BLAST, il est toutefois nécessaire de souligner que la base de données contient l'intégralité des séquences. REDIS, quant à lui, a été tributaire de son incapacité à gérer des clés au format *bitset* et du fait que l'espace mémoire occupé pour une clé, est de 50 octets au minimum, alors que dans notre cas les clés avaient une taille de 8 octets.

	BLAST	PSH-DB	REDIS
Taille des bases de données	6 Go	4 Go	12 Go
Temps d'indexation	35 min	153 min	181 min
Temps de recherche	198 min	21 min	24 min

Tableau 5.10 – *Taille des structures de données et des temps d'exécution entre BLAST, PSH-DB et REDIS*

En ce qui concerne les temps d'exécution présentés Tab 5.9, les phases d'indexation de PSH-DB et REDIS s'avèrent être plus lentes que BLAST. Ceci peut s'expliquer par le fait que BLAST utilise une méthode d'indexation différente de PSH-DB, où le calcul de millions de clés de hachage a été nécessaire pour cette expérimentation. Au niveau des temps de recherche de k-mers exacts, PSH-DB et REDIS s'avèrent être plus rapides que BLAST, avec un léger avantage pour PSH-DB. La Figure 5.17, montre la distribution des temps d'exécution entre PSH-DB et BLAST. Ceci s'explique en grande partie par la structure de type clé-valeur de ces deux systèmes où les requêtes de type GET ont une complexité temporelle constante. Les variations des temps de recherche de PSH-DB et REDIS (Cf. Figure 5.18) sont principalement dues au nombre de résultats renvoyés par les serveurs aux clients.

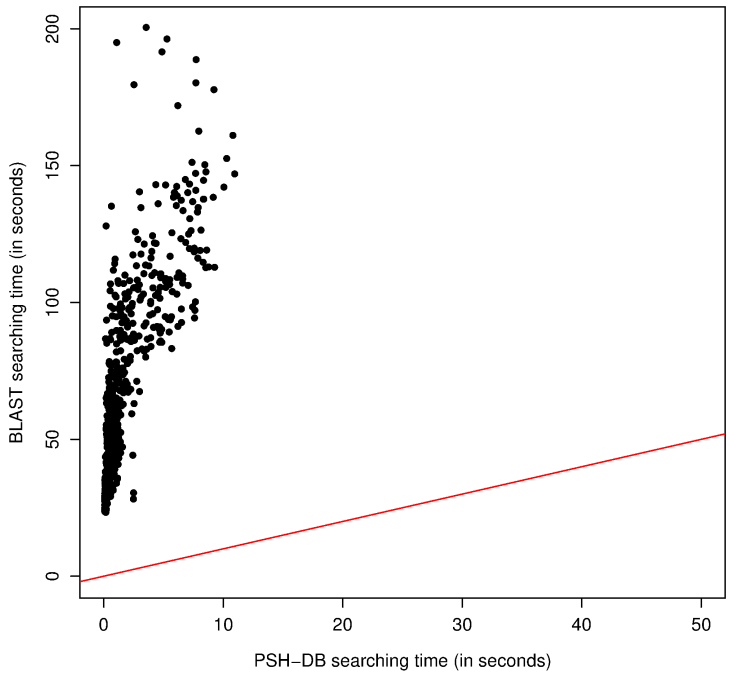


Figure 5.17 – *Comparaison des temps d'exécution de 1000 requêtes entre PSH (abscisse) et BLAST (ordonné)*

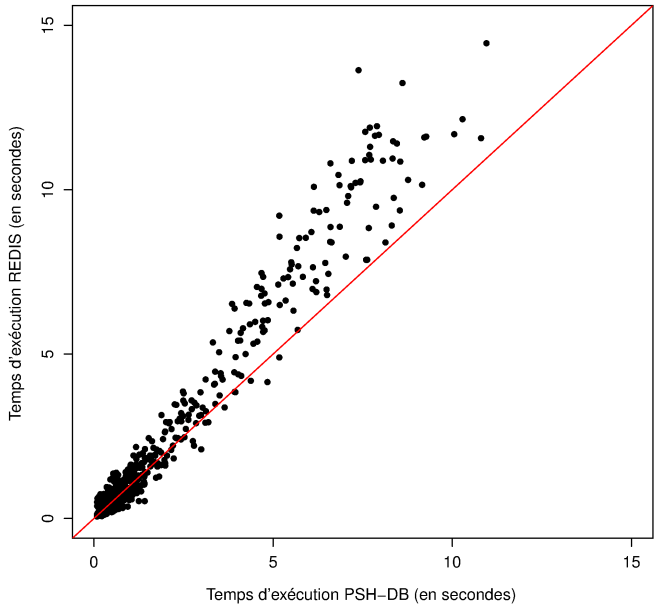


Figure 5.18 – *Comparaison des temps d'exécution de 1000 requêtes entre PSH (abscisse) et REDIS (ordonnée)*

5.4. Recherche de séquences proches

La seconde expérimentation que nous avons mené avait pour objectif d'évaluer la recherche de k-mers proches mais non exacts au sein de PSH-DB, en utilisant les propriétés de comparabilité des clés de hachage calculées par la distance de Hamming. Une base de données, de 4 séquences d'ADN de référence, a été constituée (Cf. Tableau 5.10). Ces séquences ont été choisies car elles ne présentaient aucune homologie. Elles ont été indexées et stockées au sein de PSH-DB en utilisant la méthode décrite en section 4.2. L'objectif était de calculer les distances de Hamming entre les clés de hachage issues d'une séquence requête et toutes les clés présentes dans la base de données. Ceci afin de déterminer s'il était possible de fixer un seuil en fonction des paramètres utilisés et au final de retrouver la séquence de référence dont était issue une séquence requête comportant des taux de mutation. Il est cependant important de noter que les taux de mutation ont été appliqués sur les séquences entières et non pas sur les k-mers. Ce qui signifie qu'en fonction des positions de mutation appliquées sur les séquences requêtes, les k-mers qui en sont issus, peuvent avoir des taux de mutation variables, compris entre 0% et 100%.

Nom de la séquence de référence	Taille
Borrelia burgdorferi B31, complete genome	910 724 pb
Escherichia coli O157:H7 str. Sakai DNA, complete genome	5 498 450 pb
West Nile virus lineage 2, complete genome	10 962 pb
Salmonella enterica subsp. enterica serovar Typhimurium str. LT2 chromosome, complete genome	4 857 432 pb

Tableau 5.11 – *Séquences de référence utilisées pour l'expérimentation n°2*

Durant cette expérimentation, nous nous sommes plus particulièrement focalisés sur la sensibilité de la méthode de recherche. Les temps de traitement présentés Tableau 5.11, ne sont donnés qu'à titre indicatif et ne peuvent pas être considérés comme représentatifs. En effet, l'expérimentation a consisté à comparer toutes les clés issues des différents k-mers des séquences requêtes avec toutes les clés de la base de données, ceci afin de pouvoir établir des statistiques sur les données obtenues.

5.4.1. Création du jeu de requêtes

Pour cette seconde expérimentation, 6000 séquences requêtes, d'une taille de 1024pb ont été utilisées. Le Tab.5 présente le nombre de séquences requêtes qui ont été extraites à partir des séquences de référence et les taux de mutation qui leur ont été appliqués. Il est à noter qu'une des séquences de référence n'a pas été utilisée pour extraire des requêtes et que nous avons ajouté 1000 requêtes générées totalement aléatoirement.

Séquences de référence	Nombre de requêtes	Taux de mutation
Borrelia burgdorferi	1000	5%
Borrelia burgdorferi	1000	10%
Escherichia coli	1000	5%
Escherichia coli	1000	10%
WestNile	1000	5%
WestNile	1000	10%
Salmonella enterica	0	-
Séquences aléatoires	1000	-

Tableau 5.12 – *Séquences de référence utilisées pour la génération de requêtes pour l'expérimentation n°2*

5.4.2. Résultats

La seconde expérimentation, avait pour objectif d'évaluer la possibilité de retrouver pour chaque séquence de notre panel, les séquences de référence les plus proches, malgré des taux de mutation compris entre 5% et 10%. Les résultats illustrés par la Fig.5.18 montrent la distribution cumulée totale des distances de Hamming, calculées à partir des séquences requête ayant été générées depuis les séquences de référence, avec un taux de mutation de 10%. La colonne « positifs » représente le nombre de clés de hachage appartenant aux séquences requête ayant été attribuées avec succès, à la séquence de référence dont elles étaient issues. La colonne des « négatifs » représente les clés ayant été attribuées de façon

erronée à une séquence de référence. En observant les distributions des distances de Hamming, issues de ces deux colonnes, il est possible de définir un intervalle significatif (distance de Hamming <= 8), permettant de déterminer la proximité entre deux clés de hachage, donc entre deux k-mers et par extension, une zone d'homologie entre deux séquences.

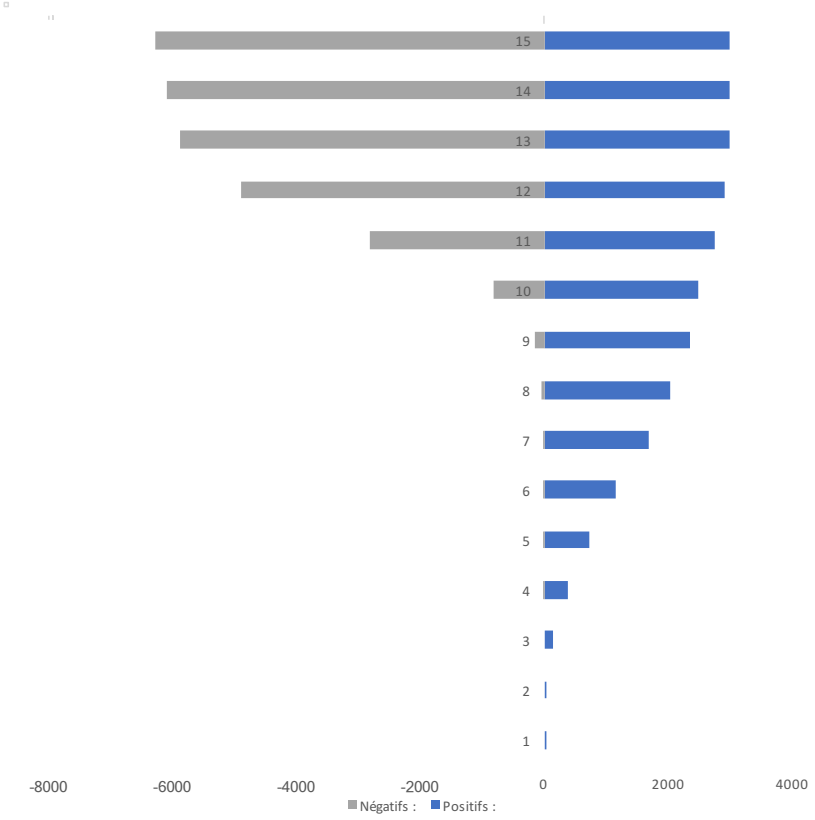


Figure 5.19 – *Distribution du nombre de requêtes (positives ou négatives) en fonction de la Distance de Hamming avec des séquences requêtes ayant un taux de 10% de mutation*

La Figure 5.20, montre la distribution cumulée totale des Distances de Hamming, calculée à partir des séquences générées aléatoirement. De façon identique à la Figure 5.19, nous pouvons observer qu'aucune des clés de hachage de ces séquences requête n'a eu de Distance de Hamming <= 8 avec les séquences de référence.

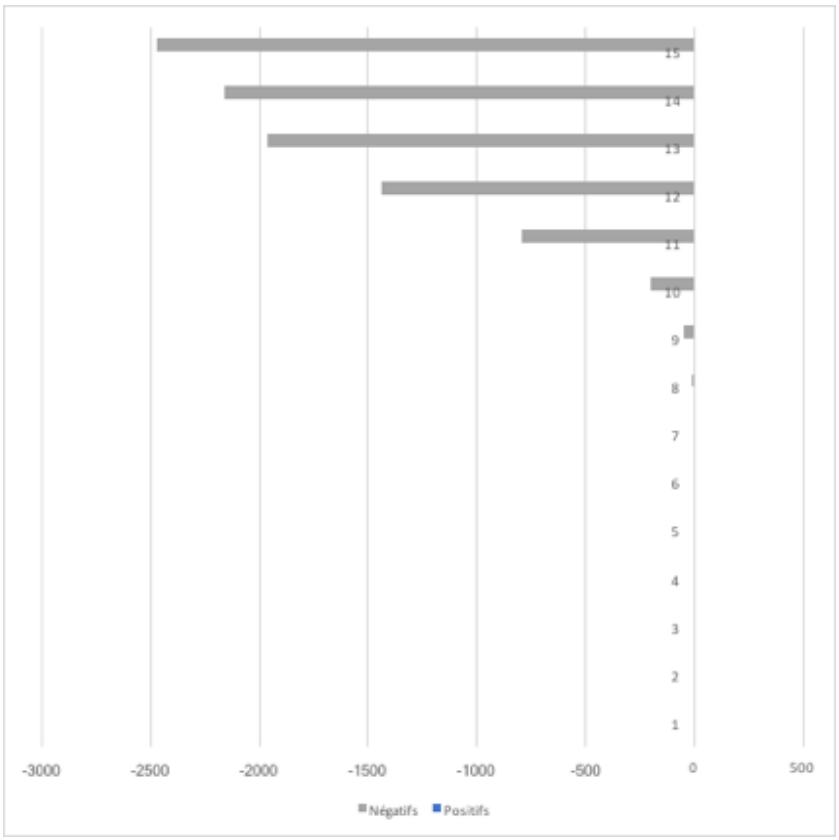


Figure 5.20 – *Distribution des requêtes issues des séquences générées aléatoirement, en fonction de la distance de Hamming*

Enfin le *Tableau 5.10*, présente la synthèse de cette expérimentation. Les résultats présentés pour les jeux de séquences requête présentant des taux de mutation de 5% et 10% montrent des résultats très encourageants avec de forts taux de détection >99% et des taux de faux positifs compris entre 2% et 3%, avec un seuil de distance de Hamming établi à 8.

Taux de mutation	Positifs (seuil < 8)	Négatifs (seuil <= 8)	Temps d'exécution
5%	100%	2,53%	486 min
10%	99,33	2,83%	492 min

Tableau 5.13 – *Synthèse des résultats renvoyés par PSH-DB en fonction du taux de mutation*

Conclusion

Les méthodes d'identification de documents de type image ou audio basées sur des fonctions de hachage perceptuel sont reconnues pour leurs capacités à comparer des documents proches et pour leurs faibles complexités algorithmiques. Cependant, bien que certaines problématiques soient communes avec les données génomiques, la littérature ne décrit aucune approche concernant l'utilisation de fonctions de hachage perceptuel appliquées aux séquences ADN/ARN.

Au cours de ce cinquième chapitre, nous avons présenté la fonction PSH. PSH est une fonction de hachage basée sur des concepts de hachage perceptuel. Elle a été développée avec l'idée de pouvoir proposer une fonction de hachage capable d'identifier une séquence ADN/ARN via le calcul d'une empreinte binaire de taille inférieure tout en gardant la propriété de pouvoir être comparée avec d'autres.

Ce chapitre a tout d'abord introduit le principe général ainsi que les différentes étapes de calcul de la fonction PSH puis la méthode de comparaison des clés de hachage. Par la suite, il a présenté différentes phases d'évaluations théoriques par simulation. Ces phases d'évaluation ont permis d'étudier le comportement de PSH afin d'en déterminer ses limites. Enfin, les expérimentations menées à partir de données réelles ont permis d'étudier le comportement de PSH couplé avec une structure de données, puis de comparer les résultats avec l'outils BLAST.

POINTS CLÉS :

- La fonction PSH est une fonction de hachage perceptuel adaptée à la problématique de séquences ADN/ARN
- La fonction PSH permet de comparer des séquences de même taille afin d'en déterminer une notion de similarité
- Les clés de hachage calculées par la fonction PSH ont des tailles très inférieures aux séquences dont elles sont issues
- Les phases d'évaluations théoriques montrent la stabilité de PSH
- Les expérimentations montrent une bonne sensibilité de PSH pour l'identification de génomes référence à partir d'une séquence requête

Chapitre 6 **– COMPARAISON DE SÉQUENCES ADN VIA LA MÉTHODE PSC**

RÉSUMÉ :

Ce chapitre présente la seconde contribution de cette thèse qui a consisté au développement et à l'étude d'une méthode permettant la comparaison de séquences ADN/ARN, via l'utilisation de représentations perceptuelles des images numériques. Cette méthode de comparaison qui étend les concepts développés durant le chapitre précédent, utilise les propriétés de corrélation de la TCD-CS pour faire ressortir les zones communes entre deux images et par extension entre deux séquences ADN/ARN.

Dans une première partie, nous présentons les différentes étapes nécessaires à la réalisation de cette méthode, puis les différentes phases d'étude, de validations théoriques et à partir de données réelles que nous avons menées.

SOMMAIE :

Introduction

Au cours du processus d'analyse des données génomiques, la recherche de similarité et la comparaison de séquences restent des tâches de base incontournables. Ainsi, le second chapitre de ce manuscrit, a abordé différentes méthodes permettant d'effectuer des comparaisons de séquences ADN par alignement des nucléotides entre plusieurs séquences. L'objectif étant de déterminer le degré de similarité général entre deux séquences S_1, et S_2 ou de déterminer les zones communes entres ces séquences. L'utilisation de ces méthodes est nécessaire notamment durant les phases d'assemblage, pour détecter des mutations, ou pour déterminer la diversité biologique d'un échantillon lors d'études en métagénomique. Elles servent notamment à déterminer des séquences consensus issues de sous-séquences, à détecter les mutations entre plusieurs séquences, ce qui permet de déduire l'évolution des espèces biologiques ou d'identifier par homologie la séquence de référence la plus proche d'une séquence requête. Ce chapitre présente une nouvelle méthode de comparaison de séquences ADN, basée sur des concepts de hachage perceptuel à l'instar de la fonction PSH présentée précédemment durant le chapitre 5. Elle permet d'effectuer la comparaison de deux séquences ADN S_1 et S_2, de taille W_1 et W_2, en permettant de rechercher S_2 au sein de S_1.

1. Corrélation d'images numériques

La corrélation d'images numériques est un ensemble de méthodes et de techniques permettant notamment de faire ressortir les zones communes entre une image de référence et une image requête. À l'origine de travaux en traitement du signal menés durant les années 40, les premières applications de ces méthodes furent dédiées aux systèmes de détection des sonars et des radars. Par la suite, les progrès en mathématiques appliquées, en informatique et le développement de l'imagerie numérique, ont conduit au développement de l'analyse des signaux et par extension aux travaux d'analyse d'images, notamment au niveau de la reconnaissance de formes ou d'objets.

Le principe de la méthode, est de faire glisser une image requête sur l'image de référence et pour chaque position, de calculer le produit des pixels de l'image motif avec les pixels de l'image de référence, puis d'en déterminer les valeurs les plus significatives. Cependant, en pratique cette méthode n'est quasiment jamais calculée sous cette forme, au sein du domaine spatial, étant donné le nombre de calculs nécessaires pour sa réalisation. Il est beaucoup plus avantageux, en termes de temps de calcul, d'effectuer cette opération dans le domaine fréquentiel. Ainsi, l'utilisation de

Transformée de Fourier via une méthode appelée Corrélation de Phase permet de déterminer le degré de similarité entre deux images, via le calcul de pics de corrélation, obtenus à partir du produit d'Hadamard des deux matrices de phase correspondant aux images à comparer. La localisation du pic donnant des informations sur la position des images, l'une par rapport à l'autre.

Les applications sont nombreuses. Elles servent notamment à réaliser de la détection de mouvements, à la reconnaissance d'objets, à faire la mise au point, au sein de certains systèmes d'autofocus d'appareils photos numériques ou à comparer des visages ou des empreintes digitales. Dans le domaine de la bio-informatique *Saldías et al.* [10] ont pu démontrer la possibilité d'utiliser une méthode de corrélation de phase pour aligner des séquences ADN, avec des taux de précision très satisfaisants, mais une vitesse de réalisation qui s'est avérée 100x plus lente que BLAST.

Nous avons donc cherché à poursuivre ce premier travail en développant une méthode semblable, avec toutefois une approche différente, basée sur l'utilisation d'une TCD, à la place d'une TFD. *Ito et al.* [230] ont présenté une méthode basée sur la corrélation des matrices de signes des coefficients d'une Transformée en Cosinus Discrète. Ainsi, ils ont pu démontrer une relation mathématique entre une Phase Only Correlation (POC) et une Sign Only Correlation (SOC), qui s'avèrent être très proches au niveau de leurs approches. En effet la POC utilise la phase de fourrier des images afin de faire ressortir les contours, alors que la SOC utilise les signes des coefficients de la TCD pour l'obtention d'un résultat proche.

(a) (b)

Figure 6.1 – *(a) Image obtenue à partir des signes d'une TCD, et (b) à partir des signes du TFD*

Cependant, la SOC s'avère avoir plusieurs avantages par rapport à la POC. En effet, la fonction TCD utilisée pour la SOC, possède de meilleures capacités de regroupement fréquentiel que la TFD utilisée

pour la POC. Cette caractéristique a notamment été recherchée pour la norme de compression d'images JPEG. Ceci se traduit par une meilleure sensibilité de la SOC notamment lorsque les images à comparer ont subi des compressions avec pertes. De plus, par rapport à une TFD, la TCD s'avère être d'une complexité algorithmique inférieure, puisqu'elle renvoie des nombres réels alors qu'une TFD renvoie des nombres complexes.

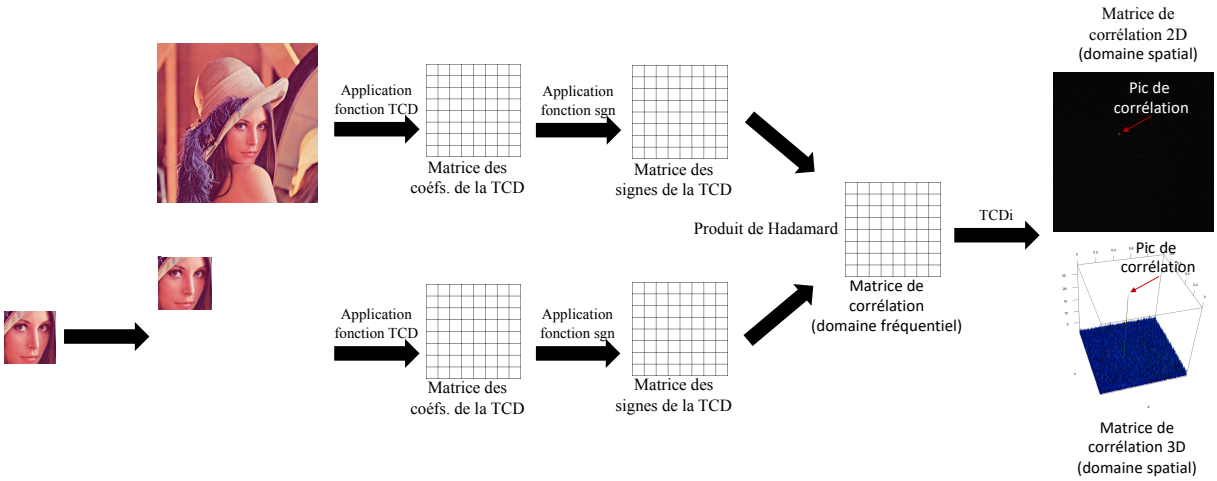


Figure 6.2 – *Processus de corrélation d'une image requête avec une image de référence*

Au cours de ce travail, nous avons cherché à adapter la SOC à la problématique de la comparaison des séquences ADN/ARN. En effet, le développement d'une méthode de comparaison de séquence ADN par SOC s'avérait être une suite logique des travaux présentés au cours du chapitre 5. La SOC est une extension de la fonction PSH, dont elle reprend les premières étapes. Les principales différences résident dans leurs méthodes de comparaisons, la fonction PSH utilisant une Distance de Hamming et la méthode CTCDS le produit de Hadamard pour évaluer la corrélation des matrices des signes des coefficients.

2. Méthode proposée

La méthode PSC, de recherche des zones communes entre deux séquences ADN/ARN par corrélation de TCD Signées, proposée au sein de ce chapitre est une extension de la méthode utilisée par la fonction PSH, puisqu'elle se base sur les concepts de hachage perceptuel identiques. Cependant, leurs objectifs divergent, car la fonction PSH a pour objectif de fournir un indice de similarité entre deux séquences ADN/ARN tout en diminuant les données à comparer, alors que la méthode présentée au

cours de ce chapitre permet de faire ressortir les zones communes entre deux séquences ainsi que leur localisation. De plus, contrairement à la fonction PSH qui nécessite de travailler avec des séquences en entrée de tailles identiques, la méthode CTCDS permet de travailler avec des séquences de tailles différentes S_1 et S_2 de taille W_1 et W_2. L'objectif étant de retrouver la séquence ayant la taille la plus petite au sein de la séquence la plus grande. Lors de la conversion des séquences à comparer, si l'une d'entre elles a une taille inférieure, l'image sera comblée avec une intensité neutre (L_5), afin qu'elle soit de même taille. Ceci aura pour effet de générer des matrices de coefficients de mêmes longueurs. Par la suite, la tentative de corrélation entre les deux matrices de coefficients binaires est calculée à partir du produit matriciel de Hadamard. Puis, afin de déterminer si la corrélation est significative, on applique une Transformée en Cosinus Discrète Inverse (*TCDI*), sur la matrice de corrélation et on recherche la valeur du pic maximum. Pour déterminer un pic de corrélation significatif, deux critères sont à prendre en considération :

- La valeur du pic de corrélation doit être supérieure à un seuil *k*
- Afin de déterminer qu'il s'agit d'un pic significatif, l'écart entre la valeur du pic et la moyenne des autres coefficients doit être supérieur à un seuil *p* exprimé en pourcentage

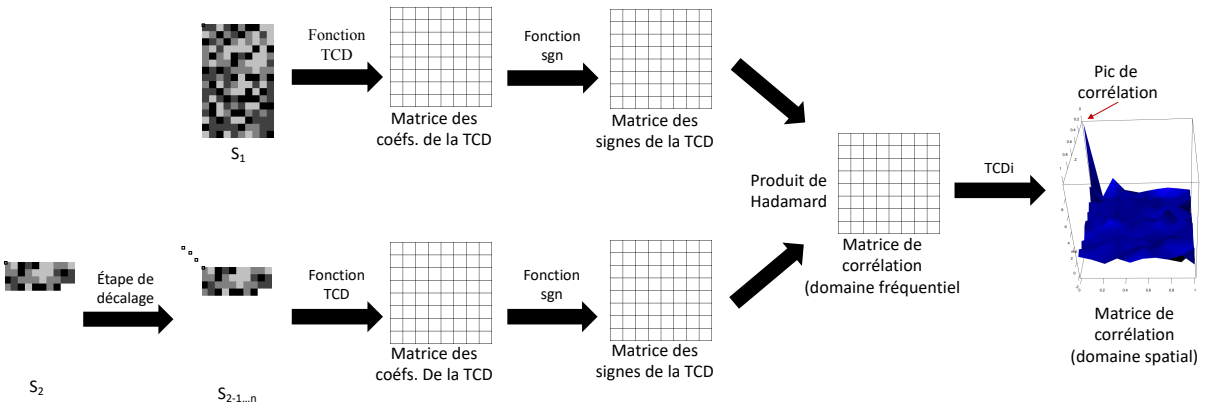


Figure 6.3 – Processus de corrélation de deux séquences ADN

2.1. Principales étapes de calcul

Les étapes permettant de déterminer si S_2 se trouve dans S_1 en utilisant la méthodes PSC sont les suivantes :

1. Soit deux séquences S_1 et S_2 de taille W_1 et W_2, avec W_1 supérieur à W_2

2. *S_1 et S_2* sont converties en deux matrices de pixels *P_1 et P_2*, de même taille *N x M*
3. Une fonction TCD est appliquée sur les matrices de pixels *P_1* et *P_2*, afin de faire passer l'information spatiale dans le domaine fréquentiel. Les coefficients de la TCD des deux matrices sont stockés au sein de deux matrices P_1' et P_2'.
4. Une fonction *sgn* est appliquée aux matrices *P_1' et P_2'*, afin de conserver uniquement les signes des différents coefficients.
5. Le produit matriciel de Hadamard est calculé entre les matrices *P_1'* et *P_2'* et son résultat est stocké au sein d'une matrice H de taille *N x M*.
6. On applique une fonction TCD Inverse sur la matrice H
7. On recherche au sein de la matrice H, un pic de valeur significatif qui indique les coordonnées *X* et *Y* des zones communes entre les séquences S_1 et S_2.

2.2. Conversion des séquences en matrice de pixels

Le processus de conversion des séquences ADN/ARN en matrice de pixels est identique à l'étape de conversion décrite durant le chapitre 5 relatif à la fonction PSH. Cependant, afin de permettre à la méthode PSC de travailler à partir de séquences de tailles différentes, il est nécessaire d'obtenir des matrices de pixels de tailles identiques *N x M*. Ainsi, un cinquième niveau d'intensité lumineuse L_5, est employé pour compléter la matrice issue de la séquence la plus petite.

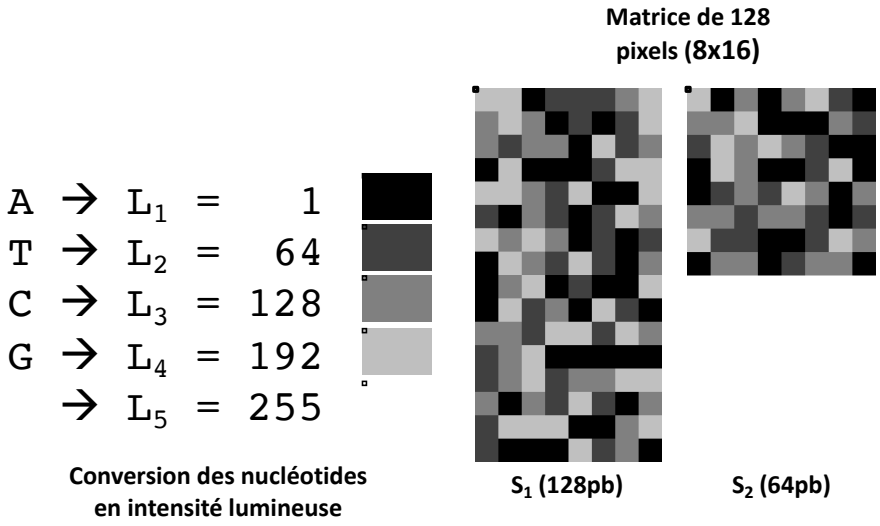


Figure 6.4 – *Encodage des nucléotides au sein d'une matrice de pixels*

2.3. Calcul des signes des coefficients de la TCD

Le calcul des matrices des coefficients binaires est strictement identique aux étapes 3 et 4 de la fonction PSH. Tout d'abord, une TCD est appliquée aux deux matrices de pixels, afin de faire passer l'information du domaine spatial vers le domaine fréquentiel. Une fois cette étape effectuée, une fonction sgn est appliquée aux deux matrices contenant les coefficients des TCD.

2.4. Calcul de la corrélation

L'étape de calcul de corrélation permet de comparer les séquences S_1 et S_2 et plus particulièrement de localiser la séquence S_2 sur la séquence S_1. Elle consiste à calculer le produit matriciel de Hadamard entre les deux matrices P1' et P2'.

2.4.1 Produit de Hadamard

Le produit de Hadamard est un produit matriciel réalisé à partir de deux matrices A et B, ayant strictement la même dimension. Il consiste en la réalisation du produit terme à terme des deux matrices.

$$A \odot B = A_{(i,j)} \times B_{(i,j)}$$

Équation 6.1 – Définition mathématique du produit matriciel d'Hadamard à partir de deux matrices A et B

2.4.2 Produit de Hadamard sur des matrices binaires

Le calcul de la corrélation devant s'effectuer à partir des matrices des signes des coefficients des TCD, le produit de Hadamard doit donc être réalisé à partir de deux matrices binaires, où dans un but d'optimisation, l'opérateur de multiplication peut être remplacé par l'opérateur binaire « AND ».

$$A \odot B = A_{(i,j)}\ AND\ B_{(i,j)}$$

Équation 6.2 – Produit matriciel d'Hadamard à partir de deux matrices binaires A et B

2.5. Application d'une TCD Inverse

Après avoir obtenu la matrice binaire résultant du produit d'Hadamard, il est nécessaire de faire repasser l'information dans le domaine fréquentiel afin de pouvoir rechercher le pic de corrélation. Cette opération est effectuée à partir d'une fonction TCD inverse.

2.6. Recherche de la corrélation optimale

Étant donnée la méthode utilisée pour encoder les nucléotides composant les séquences sous forme de pixels, où le premier pixel d'une ligne suit immédiatement le dernier pixel de la ligne n – 1 ; il est nécessaire d'effectuer plusieurs décalages de nucléotides, afin de rechercher la corrélation optimale. La méthode adoptée, a été d'effectuer des décalages successifs en supprimant à chaque décalage, le premier pixel en coordonnée (x=0 et y=0) de l'image avec un nombre de décalage égal au nombre de pixels composant une ligne. Par conséquent, afin de minimiser les temps de corrélation, il est donc nécessaire de définir des images d'une largeur faible.

L'exemple suivant montre deux séquences S1 et S2 d'une taille de 160pb pour la première et de 15pb pour la seconde. Avec un encodage de S1 et S2 au sein de deux matrices de pixels de taille 10x10, il est ainsi nécessaire d'effectuer 9 décalages successifs afin d'obtenir la meilleure configuration possible pour que la séquence S1 soit la même que S2. L'intérêt d'effectuer différents décalages successifs, est de garantir la meilleure corrélation possible entre deux séquences. Cela a aussi pour effet de garantir le pic de corrélation en X = 0, ce qui nous donne un indice supplémentaire pour déterminer l'endroit de la corrélation. Pour comparer deux séquences, on effectue donc autant de tentatives de corrélation, qu'il y a de pixels sur une ligne de la matrice, puis on garde le meilleur ratio entre le pic et la moyenne des différentes corrélations effectuées.

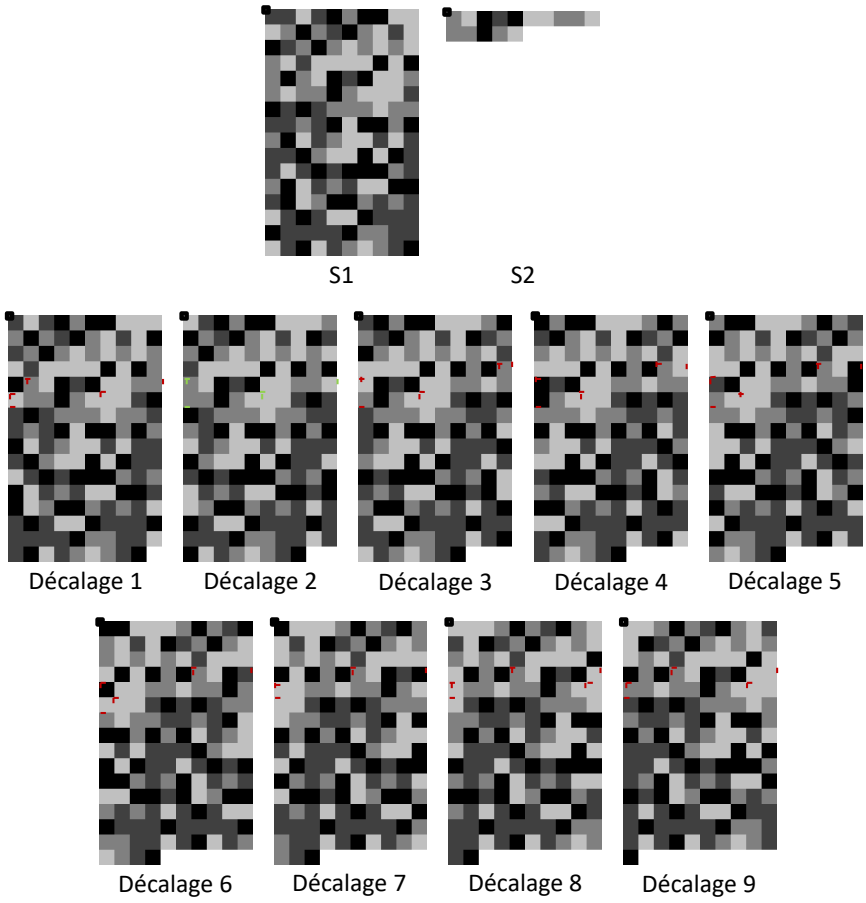


Figure 6.5 – *Étapes de décalage de la séquence S_2* par rapport à *S_1*

3. Résultats expérimentaux

Différentes phases de validation par simulations théoriques ont été menées. L'objectif était d'évaluer la stabilité de la méthode PSC, en étudiant l'évolution de différents paramètres telle que la taille des séquences à comparer ainsi que leurs taux de mutation. Nous avons donc cherché à évaluer la stabilité des valeurs de corrélation entre deux séquences homologues de même taille, puis de tailles différentes, au cours de plusieurs phases de simulations théoriques.

3.1. Environnement et protocole expérimental

À l'instar des expérimentations présentées durant le chapitre 5, la méthode PSC a été implémentée en langage *C++11*. Les différentes phases de simulations ainsi que les expérimentations ont été réalisées

sur un serveur fonctionnant sous Linux Ubuntu 14.04, équipé de 4 processeurs Intel(R) Xeon(R) E7-4850 pour un total de 96 threads en parallèle.

3.1. Évaluation avec des séquences de même taille

La première phase de simulation a consisté à étudier le comportement de la méthode PSC lors de comparaisons de séquences de taille identique. Ainsi, 35 groupes de simulations ont été réalisés avec des séquences variant de 100pb à 100000pb et des taux de mutation compris entre 2% et 50%. Afin de pouvoir obtenir des résultats statistiquement significatifs, nous avons réalisé pour chacun des groupes, 10 millions de comparaisons entre une séquence S_1 générée aléatoirement et une séquence divergente S_2 générée avec le modèle évolutif Juke-Cantor (Cf. Tableau 6.1). Puis, pour chaque paire de séquences S_1-S_2, une tentative de corrélation a été effectuée en utilisant la méthode PSC.

Groupe :	Taille :	% Mutation :	Groupe :	Taille :	% Mutation :
A1	100	2%	**B1**	200	2%
A2	100	5%	**B2**	200	5%
A3	100	10%	**B3**	200	10%
A4	100	20%	**B4**	200	20%
A5	100	50%	**B5**	200	50%
C1	500	2%	**D1**	1000	2%
C2	500	5%	**D2**	1000	5%
C3	500	10%	**D3**	1000	10%
C4	500	20%	**D4**	1000	20%
C5	500	50%	**D5**	1000	50%
E1	2000	2%	**F1**	5000	2%
E2	2000	5%	**F2**	5000	5%
E3	2000	10%	**F3**	5000	10%
E4	2000	20%	**F4**	5000	20%
E5	2000	50%	**F5**	5000	50%
G1	10000	2%	**H1**	100000	2%
G2	10000	5%	**H2**	100000	5%
G3	10000	10%	**H3**	100000	10%
G4	10000	20%	**H4**	100000	20%
G5	10000	50%	**H5**	100000	50%

Tableau 6.1 – *Paramètres des groupes de simulation A1 à G5*

Durant cette première phase de simulations, nous avons cherché à étudier l'évolution des valeurs de corrélation en fonction de la taille des séquences et des taux de mutations associés. Le *Tableau 6.2*

présente la synthèse des résultats obtenus à partir des 35 groupes de simulations. Pour chaque groupe, il présente le taux de corrélations positives et des statistiques sur les pics de corrélations tels que les valeurs de pics minimum, maximum, la moyenne des pics ainsi que l'écart-type.

Les *Figures 6.6* et *6.7* montrent les distributions des valeurs des pics ordonnées de façon croissante pour les groupes A et H. L'intégralité des graphiques correspondants aux groupes A à N sont consultables en *Annexe C.*

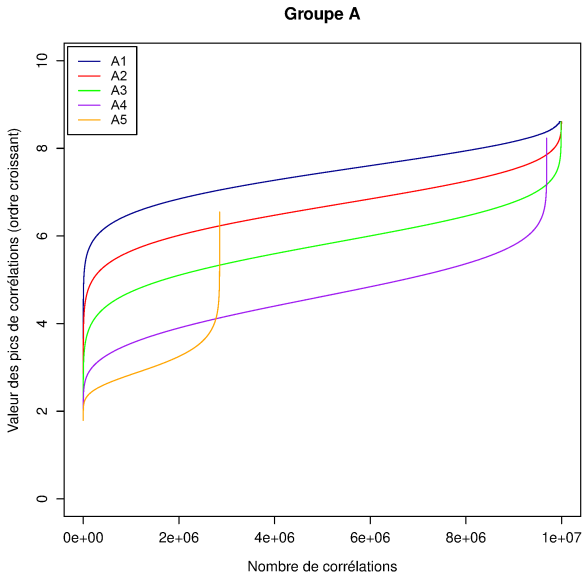


Figure 6.6 – *Exemple de distributions statistiques triées par ordre croissant des pics de corrélation pour les groupes A1 à A5*

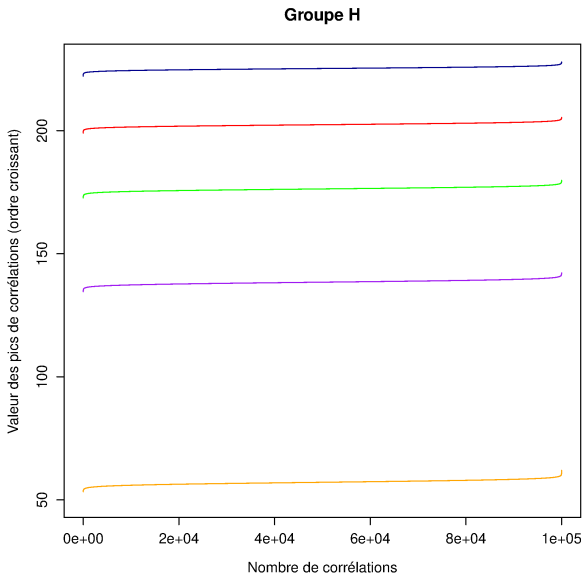


Figure 6.7 – *Exemple de distributions statistiques triées par ordre croissant des pics de corrélation pour les groupes H1 à H5*

Groupes :	% Positifs :	Pic (min) :	Pic (max) :	Pic (moy) :	Pic (SD) :
A1	100%	2,428	8,607	7,375	0,640
A2	99,99%	2,024	8,607	6,619	0,725
A3	99,90%	2,013	8,607	5,768	0,793
A4	96,89%	1,871	8,129	4,590	0,806
A5	28,51%	1,796	6,399	3,059	0,464
B1	100%	6,137	11,999	10,253	0,639
B2	100%	4,682	11,852	9,211	0,725
B3	100%	2,816	11,378	8,022	0,797
B4	99,93%	2,327	10,369	6,300	0,865
B5	37,80%	2,022	7,428	3,516	0,548
C1	100%	12,073	18,460	16,040	0,637
C2	100%	10,285	17,560	14,411	0,724
C3	100%	7,893	16,169	12,552	0,797
C4	100%	5,259	14,057	9,854	0,868
C5	77,57%	2,313	8,512	4,411	0,727
D1	100%	18,746	25,318	22,603	0,635
D2	100%	16,136	23,733	20,310	0,723
D3	100%	13,071	21,526	17,692	0,797
D4	100%	9,049	18,432	13,889	0,868
D5	98,48%	2,722	10,654	5,768	0,902
E1	100%	28,010	34,889	31,907	0,635
E2	100%	24,549	32,138	28,672	0,723
E3	100%	20,878	28,929	24,976	0,796
E4	100%	14,956	23,939	19,609	0,868
E5	99,99%	3,674	12,950	8,094	0,939
F1	100%	47,032	53,465	50,396	0,634
F2	100%	41,353	48,934	45,287	0,723
F3	100%	35,371	43,481	39,450	0,796
F4	100%	7,135	17,828	12,787	0,939
F5	100%	67,700	74,393	71,245	0,634
G1	100%	60,157	67,680	64,023	0,723
G2	100%	51,361	59,773	55,772	0,796
G3	100%	39,067	48,474	43,789	0,868
G4	100%	13,449	23,070	18,077	0,939
G5	100%	222,719	228,049	225,219	0,633
H1	100%	222,719	228,049	225,219	0,633
H2	100%	199,310	205,591	202,394	0,726
H3	100%	172,735	179,856	176,320	0,799
H4	100%	134,455	142,157	138,432	0,867
H5	100%	53,213	61,311	57,145	0,940

Tableau 6.2 – *Synthèse des résultats obtenus pour les groupes A1 à H5*

3.2. Évaluations avec des séquences de tailles différentes

Durant la seconde phase de simulations théoriques, nous avons cherché à étudier le comportement de la méthode PSC lors de la comparaison de séquences ADN de tailles différentes avec différents taux de mutations. Ainsi, 54 groupes de 10 millions de séquences ont été générés avec des séquences S_1 variant de 1000pb à 5000pb, des sous-séquences S_2 extraites aléatoirement à partir des séquences S_1 avec des tailles variant de 10% à 50% et des taux de mutation variant de 0% à 50%. Pour chacun des groupes nous avons cherché à étudier le taux de corrélations positives et des statistiques sur les pics de corrélation tels que les valeurs de pics minimum, maximum, la moyenne des pics ainsi que l'écart-type.

Groupe	Taille S_1	Taille S_2	Mutation(s)	Groupe	Taille S_1	Taille S_2	Mutation(s)
I1-1	1000	100	0	J1-1	2000	200	0
I1-2	1000	100	2	J1-2	2000	200	4
I1-3	1000	100	5	J1-3	2000	200	10
I1-4	1000	100	10	J1-4	2000	200	20
I1-5	1000	100	20	J1-5	2000	200	40
I1-6	1000	100	50	J1-6	2000	200	100
I2-1	1000	200	0	J2-1	2000	400	0
I2-2	1000	200	4	J2-2	2000	400	8
I2-3	1000	200	10	J2-3	2000	400	20
I2-4	1000	200	20	J2-4	2000	400	40
I2-5	1000	200	40	J2-5	2000	400	80
I2-6	1000	200	100	J2-6	2000	400	200
I3-1	1000	500	0	J3-1	2000	1000	0
I3-2	1000	500	10	J3-2	2000	1000	20
I3-3	1000	500	25	J3-3	2000	1000	50
I3-4	1000	500	50	J3-4	2000	1000	100
I3-5	1000	500	100	J3-5	2000	1000	200
I3-6	1000	500	250	J3-6	2000	1000	500
K1-1	5000	500	0	K2-4	5000	1000	100
K1-2	5000	500	10	K2-5	5000	1000	200
K1-3	5000	500	25	K2-6	5000	1000	500
K1-4	5000	500	50	K3-1	5000	2500	0
K1-5	5000	500	100	K3-2	5000	2500	50
K1-6	5000	500	250	K3-3	5000	2500	125
K2-1	5000	1000	0	K3-4	5000	2500	250
K2-2	5000	1000	20	K3-5	5000	2500	500
K2-3	5000	1000	50	K3-6	5000	2500	1250
L1-1	10000	1000	0	L2-1	10000	2000	0
L1-2	10000	1000	20	L2-2	10000	2000	40
L1-3	10000	1000	50	L2-3	10000	2000	100
L1-4	10000	1000	100	L2-4	10000	2000	200
L1-5	10000	1000	200	L2-5	10000	2000	400
L1-6	10000	1000	500	L2-6	10000	2000	1000
L3-1	10000	5000	0	M1-1	20000	2000	0
L3-2	10000	5000	100	M1-2	20000	2000	40
L3-3	10000	5000	250	M1-3	20000	2000	100
L3-4	10000	5000	500	M1-4	20000	2000	200
L3-5	10000	5000	1000	M1-5	20000	2000	400
L3-6	10000	5000	2500	M1-6	20000	2000	1000
M2-1	20000	4000	0	M3-1	20000	10000	0
M2-2	20000	4000	80	M3-2	20000	10000	200
M2-3	20000	4000	200	M3-3	20000	10000	500
M2-4	20000	4000	400	M3-4	20000	10000	1000
M2-5	20000	4000	800	M3-5	20000	10000	2000
M2-6	20000	4000	2000	M3-6	20000	10000	5000
N1-1	50000	5000	0	N2-1	50000	10000	0
N1-2	50000	5000	100	N2-2	50000	10000	200
N1-3	50000	5000	250	N2-3	50000	10000	500
N1-4	50000	5000	500	N2-4	50000	10000	1000
N1-5	50000	5000	1000	N2-5	50000	10000	2000
N1-6	50000	5000	2500	N2-6	50000	10000	5000
N3-1	50000	25000	0				
N3-2	50000	25000	500				
N3-3	50000	25000	1250				
N3-4	50000	25000	2500				
N3-5	50000	25000	5000				
N3-6	50000	25000	12500				

Tableau 6.3 – *Paramètres des groupes de simulation I1-1 à N3-6*

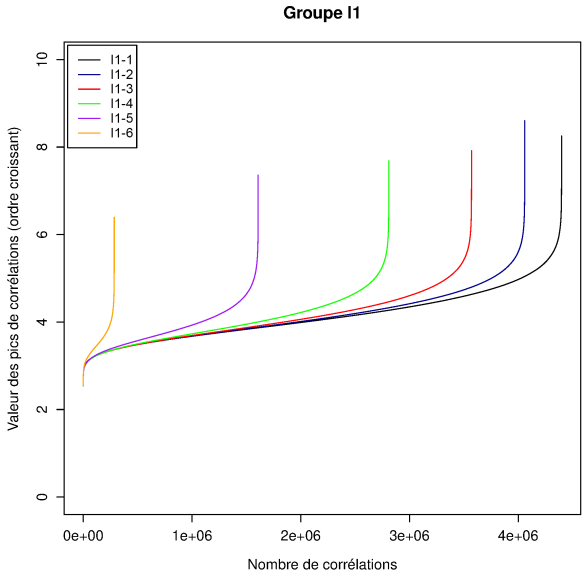


Figure 6.8 – *Exemple de distributions statistiques triées par ordre croissant des pics de corrélation pour les groupes I1-1 à I1-5*

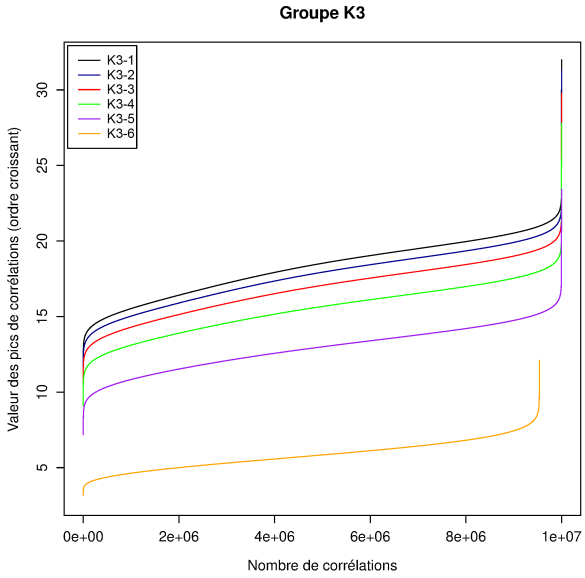


Figure 6.9 – *Exemple de distributions statistiques triées par ordre croissant des pics de corrélation pour les groupes K3-1 à K3-5*

Groupe :	% Positifs :	% Négatifs :	Pic (min,) :	Pic (max,) :	Pic (moy,) :	Pic (SD) :
I1-1	43,98%	56,02%	2,62	8,25	4,14	0,58
I1-2	40,59%	59,41%	2,58	8,60	4,10	0,57
I1-3	35,70%	64,30%	2,59	7,92	4,06	0,56
I1-4	28,10%	71,90%	2,54	7,68	3,98	0,53
I1-5	16,08%	83,92%	2,58	7,36	3,85	0,48
I1-6	2,85%	97,15%	2,57	6,40	3,56	0,36
I2-1	91,30%	8,70%	2,66	10,28	5,02	0,84
I2-2	88,99%	11,01%	2,64	10,21	4,92	0,82
I2-3	84,79%	15,21%	2,59	9,98	4,78	0,79
I2-4	75,75%	24,25%	2,60	9,76	4,58	0,74
I2-5	52,21%	47,79%	2,50	8,57	4,25	0,63
I2-6	5,05%	94,95%	2,50	6,81	3,65	0,40
I3-1	100%	0,00%	3,43	15,73	8,17	1,09
I3-2	100%	0,00%	3,18	15,66	7,92	1,08
I3-3	99,99%	0,01%	3,01	14,91	7,55	1,07
I3-4	99,95%	0,05%	2,89	14,25	6,94	1,05
I3-5	98,63%	1,37%	2,75	12,51	5,82	0,98
I3-6	19,99%	80,01%	2,57	8,38	3,91	0,51
J1-1	86,44%	13,56%	2,86	10,41	5,03	0,80
J1-2	83,44%	16,56%	2,81	10,32	4,95	0,78
J1-3	78,23%	21,77%	2,83	10,55	4,83	0,75
J1-4	67,70%	32,30%	2,82	9,94	4,64	0,70
J1-5	43,15%	56,85%	2,77	8,64	4,36	0,60
J1-6	2,94%	97,06%	2,82	6,61	3,83	0,39
J2-1	99,90%	0,10%	3,24	13,61	6,93	1,05
J2-2	99,82%	0,18%	3,13	13,38	6,74	1,04
J2-3	99,59%	0,41%	3,04	13,17	6,46	1,02
J2-4	98,57%	1,43%	2,90	12,45	6,01	0,99
J2-5	90,22%	9,78%	2,90	10,72	5,22	0,86
J2-6	9,22%	90,78%	2,72	7,53	3,99	0,45
J3-1	100%	0,00%	6,19	21,45	11,55	1,33
J3-2	100%	0,00%	5,81	20,75	11,19	1,31
J3-3	100%	0,00%	5,31	20,48	10,66	1,28
J3-4	100%	0,00%	4,70	18,96	9,80	1,23
J3-5	100%	0,00%	3,47	16,31	8,17	1,14
J3-6	48,21%	51,79%	2,83	8,89	4,43	0,63
K1-1	99,98%	0,02%	3,67	15,26	7,67	1,09
K1-2	99,96%	0,04%	3,45	14,67	7,47	1,08
K1-3	99,88%	0,12%	3,20	14,17	7,16	1,06
K1-4	99,48%	0,52%	3,23	13,06	6,66	1,03
K1-5	94,52%	5,48%	3,20	11,85	5,73	0,92

K1-6	9,13%	90,87%	3,08	7,52	4,26	0,46
K2-1	100%	0,00%	5,93	20,20	11,00	1,34
K2-2	100%	0,00%	5,47	19,63	10,69	1,32
K2-3	100%	0,00%	4,92	18,92	10,23	1,29
K2-4	100%	0,00%	4,68	18,64	9,47	1,24
K2-5	99,99%	0,01%	3,50	15,47	7,98	1,15
K2-6	38,10%	61,90%	3,09	8,84	4,58	0,59
K3-1	100%	0,00%	11,75	31,99	18,25	1,89
K3-2	100%	0,00%	10,77	31,37	17,68	1,85
K3-3	100%	0,00%	10,21	29,76	16,84	1,78
K3-4	100%	0,00%	9,10	27,79	15,48	1,68
K3-5	100%	0,00%	7,19	23,42	12,89	1,49
K3-6	95,36%	4,64%	3,17	12,11	5,83	0,96

Tableau 6.4 – *Synthèse des résultats obtenus pour les groupes I1-1 à N3-6*

3.3. Temps d'exécution

Durant ces différentes phases de simulation, nous avons cherché à évaluer les temps d'exécution en fonction des tailles des séquences ADN. Le *Tableau 6.5* présente les temps d'exécution du groupe *A* au groupe *H*, les groupes suivants ne sont ici pas représentés étant donné qu'ils reprennent des tailles identiques.

Nous avons cherché à évaluer spécifiquement les temps de calcul inhérents au processus de comparaison d'une séquence S_1 et S_2 par la méthode PSC. De ce fait, les temps de générations aléatoires des séquences S_1 ainsi que les temps correspondants au processus de calcul des séquences divergentes S_2, ne rentrent pas en compte dans notre évaluation.

	Temps total (96 threads)	Corrélations par / s (96 threads)	Temps total (1 threads)	Corrélations par /s (1 thread)
Groupe A	149	67 114,09	9539	10 48,32
Groupe B	184	54 347,82	12154	822,77
Groupe C	320	31 250	24244	412,47
Groupe D	566	17 667,84	44290	225,78
Groupe E	1042	9 596,92	84194	118,77
Groupe F	3799	26 32,27	244400	40,91
Groupe G	8498	1 176,74	490512	20,38
Groupe H	543550	18,39	13941274	0,71

Tableau 6.5 – *Synthèse des temps d'exécution obtenus pour les groupes de simulation A à H*

4. Expérimentation à partir de séquences de référence

La seconde phase a consisté à travailler à partir de données réelles. L'objectif était d'évaluer la capacité de la méthode PSC à pouvoir localiser des séquences requêtes sur le génome de référence dont elles étaient issues. Un panel de séquences de référence composé de différentes espèces bactériennes (*Cf. Tableau 6.6*), provenant du NCBI a été constitué. À partir de ces données de référence, cinq groupes composés de 8000 séquences requêtes chacun, ont été générés aléatoirement à partir des séquences de référence. Les requêtes avaient des tailles comprises entre 1000pb et 5000pb et des taux de mutations compris entre 2% et 50%. À ces différents groupes s'est ajouté un groupe supplémentaire composé de 1000 séquences générées de façon totalement aléatoire.

Nom du génome :	Pourcentage GC
Streptomyces rimosus	71%
Desulfovibrio vulgaris	63%
Synechococcus elongatus	55%
Salmonella enterica	52%
Escherichia coli	50%
Coxiella burnetii	42%
Borrelia burgdorferi	28%

Tableau 6.6 – *Panel des séquences de référence et leurs taux de nucléotides GC associés*

Le processus d'interrogation a consisté à rechercher pour chacune des séquences requêtes, les séquences de référence dont elles étaient issues. Ainsi, pour chaque requête, une tentative de corrélation a été effectuée avec les différentes séquences de référence afin de rechercher le pic de corrélation maximale. Nous avons cherché à déterminer si les pics de corrélation les plus élevés permettaient de retrouver les génomes de référence des requêtes. De plus, en nous basant sur les résultats obtenus à partir des précédentes phases de simulations théoriques, nous avons pu en déduire une valeur de pic seuil, représentative d'une corrélation positive entre deux séquences.

Pour cette expérimentation, les séquences de référence ont tout d'abord été indexées et stockées au sein d'une instance du moteur PSH-DB. Chaque séquence a été découpée en sous-séquences d'une taille de 20000pb avec un chevauchement de 5000pb, qui correspond à la taille maximum que pouvaient avoir les séquences requêtes. De plus, afin de pouvoir indexer les séquences dans leur

intégralité, une séquence terminale d'une taille de 5000 nucléotides a été calculée à partir de la position correspondant à la taille de la séquence moins 5000pb.

Le tableaux 6.7 présente les paramètres utilisés pour cette expérimentation :

Taille des sous-séquences :	20 000pb :
Taille des requêtes :	De 500 à 5000pb
Taille des images :	10x2000px
Taille des matrices binaires :	10x2000bits

Tableau 6.7 – *Paramètres utilisés pour l'expérimentation*

Parallèlement, BLAST a aussi été utilisé avec le même jeu de données, afin de pouvoir comparer les résultats en termes de sensibilité avec notre méthode.

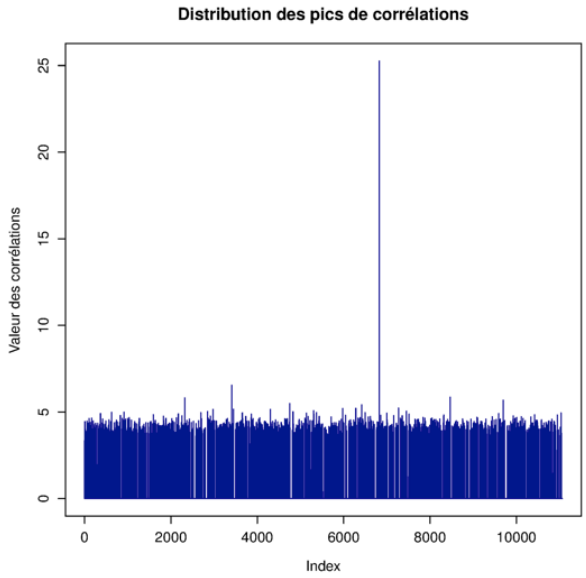


Figure 6.10 – *Exemple de distribution des différentes valeurs de corrélation d'une clé de hachage requête avec les clés de hachage du panel de séquences de référence. Nous pouvons observer un pic de corrélation significatif, ce qui illustre la similarité de la requête avec une séquence de référence*

4.1. Tableau de résultats

	0%		2%		5%		10%		20%	
	PSC	**BLAST**	**PSC**	**BLAST**	**PSC**	**BLAST**	**PSC**	**BLAST**	**PSC**	**BLAST**
B. burgdorferi	98,6%	100%	98,5%	100%	97,7%	100%	96,7%	100%	94,8%	44,2%
E. coli	99,4%	100%	99,6%	100%	99,2%	100%	99,1%	100%	97,7%	49,2%
C. burnetii	98,5%	100%	98,3%	100%	97,4%	100%	96,2%	100%	93,6%	48,8%
D. vulgaris	98,5%	100%	98,4%	100%	97,6%	100%	97,7%	100%	92,8%	47,4%
S. enterica	98,4%	100%	97,8%	100%	97,2%	100%	96,5%	100%	93,2%	48,2%
S. elongatus	98,7%	100%	97,7%	100%	96,6%	100%	96,5%	100%	93,9%	52,0%

Tableau 6.8 – *Synthèse des résultats obtenus par génome, avec la méthode PSC et BLAST*

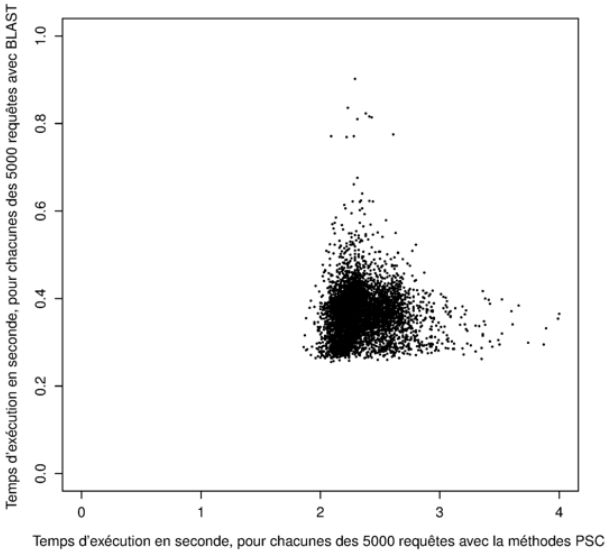


Figure 6.11 – *Comparaison des temps de recherche entre PSC et BLAST au sein de la base de données de séquence de référence*

	Temps d'exécution (en seconde)
PSC	1 834s
BLAST	11 731s
Ratio	6,40x

Tableau 6.9 – *Synthèse des temps d'exécution obtenus avec la méthode recherche PSC et l'outil BLAST*

Conclusion

Les méthodes de corrélations d'images ont pour objectif principal de faire ressortir les zones communes entre deux images. Parmi celles-ci, celles qui exploitent les caractéristiques du domaine fréquentiel ont démontré un certain intérêt quant à leurs sensibilités. Ceci est notamment le cas en bioinformatique avec les travaux de Saldías et al. [10].

Au cours de ce sixième chapitre, nous avons présenté la méthode de corrélation PSC. Cette méthode a pour objectif de faire ressortir les zones communes entre deux séquence ADN/ARN. Elle utilise des concepts identiques dont elle reprend les bases notamment en ce qui concerne l'exploitation des signes des coefficients de la fonction TCD. Par rapport à PSH elle n'a pas pour objectif d'indexer les séquences ADN mais de pouvoir les comparer en dépit de tailles différentes.

Dans une première partie nous introduisons les principes de la corrélation des images numériques, puis nous décrivons les différentes étapes de la méthodes PSC. Puis nous décrivons différentes expérimentations réalisées à partir de données simulées pour les phases d'études théoriques et à partir de données réelles avec une comparaison face à l'outils BLAST.
En termes de sensibilité, les résultats obtenus sont très encourageants, puisqu'ils présentent des taux de sensibilité proches de ceux de BLAST et de ceux présentés par Saldías et al. [10] tout en ayant des temps d'exécution très inférieurs, mais toutefois encore insuffisants par rapport à BLAST. Par rapport à l'approche de Saldías et al., une avancée significative a néanmoins été réalisée puisque PSC est 6,4x plus lent que BLAST alors que la méthode par corrélation de Transformée de Fourrier Discrète s'avérait 100x plus lent que BLAST.

POINTS CLÉS :

- La méthode PSC est une extension de la fonction PSH
- PSC utilise une méthode de corrélation des signes d'une fonction TCD.
- PSC permet de faire ressortir les zones communes entre deux séquences ADN/ARN
- PSC peut comparer des séquences de tailles différentes
- Les temps d'exécution sont 6,4x inférieurs à BLAST
- PSC a une sensibilité proche de BLAST voir supérieure, lorsque S_1 et S_2 ont des taux de mutation >20%

Chapitre 7 **– CONCLUSION GÉNÉRALE ET PERSPECTIVES**

La comparaison de séquences ADN/ARN et par extension les tâches d'indexation et de recherche au sein de larges bases de données de séquences de références sont des problématiques communes à toute étude en génomique. Au cours de cette thèse nous avons cherché à explorer une voie qui a été peu décrite dans la littérature, à savoir l'utilisation de concepts et de méthodes permettant la comparaison d'images numériques adaptés à la problématique de comparaison de séquences ADN/ARN. De par certains aspects, les domaines de l'imagerie numérique et l'analyse de séquences ADN/ARN possèdent des problématiques communes. Les images numériques et les données génomiques sont issues d'un processus d'acquisition et subissent une transformation au format numérique, avec pour l'une, l'acquisition de la réfraction de la lumière et pour l'autre la lecture d'un enchaînement de molécules. Une fois numérisée, ceci a pour conséquence la production de documents numériques, composés de sous unités appelées pixels, dans le cas des images numériques et nucléotides, dans le cas des séquences ADN/ARN. Ces deux types de données, doivent pouvoir être comparés, malgré qu'ils aient subis différents types de dégradations, afin de pouvoir être recherchés au sein de grands ensembles ou bases de données. De plus, la mise en œuvre de mécanismes de diminution des données ou d'indexation devient nécessaire, compte tenu de l'accroissement exponentiel des capacités de production qui est toujours en cours.

Durant l'élaboration de ce travail de thèse, les différentes études bibliographiques, les discussions et réflexions autour des domaines telles que la comparaison de séquences ADN/ARN, l'imagerie numérique, l'indexation des données et l'étude des fonctions de hachage ont permis, par l'intermédiaire de deux contributions, de démontrer l'intérêt que pouvaient

avoir des approches utilisant des méthodes de comparaison d'images pour comparer des séquences ADN.

Ainsi, les différents développements méthodologiques et leurs processus de validation décrits au sein de ce manuscrit ont permis de démontrer la pertinence de l'encodage de séquences ADN/ARN sous forme de matrice de pixels et l'utilisation de fonctions mathématiques telle que la Transformée en Cosinus Discrète, qui autorise le passage de données représentées sur un plan spatial au sein du domaine fréquentiel, permettant ainsi de bénéficier des caractéristiques de regroupement fréquentiel exploitées pour l'analyse des images numériques. Ces deux concepts sont le point de départ de ce travail de thèse et ils ont été développés au travers de deux contributions.

La première contribution présentée est la fonction de Hachage PSH (Perceptual Sequence Hashing). Elle a été développée avec l'idée de pouvoir proposer une fonction de hachage capable d'identifier une séquence ADN/ARN via le calcul d'une empreinte binaire de taille inférieure tout en gardant la propriété de pouvoir être comparée. La capacité de pouvoir comparer des clés de hachage, c'est à dire, de pouvoir établir une notion de distance entre deux clés est un concept qui n'avait jamais été étudié jusqu'alors en bio-informatique. Les développements méthodologiques qui ont donné lieu à la fonction PSH sont une illustration et une application directe de ces concepts. Les différentes phases d'étude et de validation de PSH ont permis tout d'abord de démontrer la stabilité de cette fonction. Tout d'abord, l'étude de la répartition des valeurs des clés de hachage, au sein de l'intervalle des valeurs possibles, a permis de démontrer que la répartition était homogène et qu'aucun sous-intervalle n'était représenté. L'étude de probabilité de collision a pu démontrer une probabilité faible, de l'ordre de 0,5, pour 5 milliards de clés, lorsque PSH renvoie des clés de 64 bits. Les différentes phases de simulations théoriques, menées à partir d'un grand volume de données simulées, ont permis d'observer les distributions statistiques des Distances de Hamming. Celles-ci étaient issues de la comparaison de séquences ADN/ARN, ayant différentes façons d'encoder les séquences au sein des matrices de pixels, des tailles différentes ainsi que des taux de mutation différents. Ceci nous a donc permis d'explorer les périmètres d'utilisation de la fonction PSH ainsi que la fonction de méthode de comparaison des clés de hachage.

La seconde contribution est une méthode nommée PSC, qui permet de déterminer la présence d'une séquence S_2 au sein d'une séquence S_1 ayant une taille supérieure. Elle reprend l'approche développée pour la fonction PSH, notamment concernant la méthode d'encodage des nucléotides au sein d'une matrice de pixels et l'approche proposée par *Saldía et al.* pour les étapes de calcul de corrélation. La principale différence étant l'utilisation d'une TCD au lieu d'une TFD qui permet de simplifier les calculs, la TFD renvoyant des coefficients de type Complexe alors que la TCD renvoie des coefficients de type Réel. Les différents processus de validation et de simulation ont pu démontrer le périmètre optimal d'utilisation de la méthode PSC. Ainsi, les pics de corrélation obtenus permettent de déterminer efficacement la présence d'une séquence S_2 au sein d'une séquence S_1 et ceci malgré des différences de tailles entre S_1 et S_2 et des taux de mutation pouvant aller jusqu'à 20%. En revanche, excepté sous certaines conditions, les valeurs des pics de corrélation ne permettent pas de déterminer les taux de mutation entre S_1 et S_2. Enfin, les expérimentations à partir de données réelles ont permis de démontrer de bons niveaux de sensibilité comparés à BLAST, lors de la recherche de séquences de référence à partir de séquences requêtes ayant subi des mutations, avec cependant des temps de recherche qui se sont avérés beaucoup plus longs que BLAST.

Ce travail de thèse, a permis de confirmer l'approche initiée par *Saldia et al.* Les deux contributions proposées, ont permis de démontrer la possibilité d'utiliser les méthodes de comparaisons d'images face à la problématique de la comparaison des séquences ADN. En termes de perspectives, de nombreuses pistes restent toutefois encore à explorer.

Bien que les expérimentations présentées dans les chapitres 5 et 6 portaient sur la recherche de séquences proches au sein du moteur PSH-DB, notre travail s'est plus particulièrement focalisé sur l'évaluation des capacités des méthodes PSH et PSC à pouvoir retrouver des séquences proches afin d'évaluer leurs sensibilités. De ce fait, les fonctions de recherche qui se basent principalement sur la capacité à rechercher des Distances de Hamming n'ont pas pu faire l'objet d'un travail particulier. Il serait néanmoins intéressant de pouvoir explorer les différentes méthodes permettant d'optimiser les recherches au sein des espaces de Hamming, telles que décrites par *Xiao et al.* [231] et *Lai et al.* [232].

Afin de tenter d'améliorer les temps de calculs nécessaires aux méthodes PSH et PSC, il pourrait être envisageable de les implémenter pour s'exécuter sur des plateformes de type GPU. En effet, les algorithmes de PSH et PSC étant issus de méthodes servant au traitement d'images, leur implémentation sous GPU, pourrait permettre d'accélérer considérablement leurs temps de calcul. Ceci pourrait être non seulement utile pour les tâches de calcul des clés de hachage PSH ou pour le calcul de corrélation PSC, mais cela pourrait également ouvrir la voie au développement d'un système de compression de séquence ADN, avec décompression via des méthodes de force brute.

La comparaison de séquences protéiques porte des problématiques identiques à celles rencontrées pour les séquences ADN. En effet, une séquence protéique est une suite d'enchaînements d'acides aminés, le long d'un polypeptide. Les acides aminés sont au nombre de 20 et représentés par un alphabet composé de 20 lettres. Tout comme les séquences ADN, les séquences protéiques doivent pouvoir être comparées, alignées, indexées et organisées au sein de bases de données. Il serait donc envisageable d'évaluer l'utilisation des méthodes PSH et PSC avec des séquences protéiques. En effet, celles-ci pourraient s'avérer facilement adaptables en attribuant simplement une valeur d'intensité lumineuse pour chacun des 20 acides aminés.

BIBLIOGRAPHIE

[1] A. B. R. McIntyre *et al.*, « Nanopore sequencing in microgravity », *Npj Microgravity*, vol. 2, nº 1, déc. 2016.
[2] M. Pop et S. L. Salzberg, « Bioinformatics challenges of new sequencing technology », *Trends Genet. TIG*, vol. 24, nº 3, p. 142-149, mars 2008.
[3] W. Li, Y. Liu, et X. Xue, « Robust audio identification for MP3 popular music », in *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '10*, Geneva, Switzerland, 2010, p. 627.
[4] L. Chang, W.-G. Yan, et W.-D. Wang, « Research on Robust Image Perceptual Hashing Technology Based on Discrete Cosine Transform », in *Business, Economics, Financial Sciences, and Management*, vol. 143, M. Zhu, Éd. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, p. 799-809.
[5] R. S. Boyer et J. S. Moore, « A fast string searching algorithm », *Commun. ACM*, vol. 20, nº 10, p. 762-772, oct. 1977.
[6] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, et D. J. Lipman, « Basic local alignment search tool », *J. Mol. Biol.*, vol. 215, nº 3, p. 403-410, oct. 1990.
[7] N. Välimäki et E. Rivals, « Scalable and Versatile k-mer Indexing for High-Throughput Sequencing Data », in *Bioinformatics Research and Applications*, vol. 7875, Z. Cai, O. Eulenstein, D. Janies, et D. Schwartz, Éd. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, p. 237-248.
[8] L. Feng, A. Jean, C. P. Leng, et L. Danbo, « Identification of DNA Signatures via Suffix Tree Construction on a Hybrid Computing System », vol. 9, nº 2, p. 10, 2012.
[9] S. Misra, A. Agrawal, W. Liao, et A. Choudhary, « Anatomy of a hash-based long read sequence mapping algorithm for next generation DNA sequencing », *Bioinforma. Oxf. Engl.*, vol. 27, nº 2, p. 189-195, janv. 2011.
[10] M. Curilem Saldías, F. Villarroel Sassarini, C. Muñoz Poblete, A. Vargas Vásquez, et I. Maureira Butler, « Image Correlation Method for DNA Sequence Alignment », *PLoS ONE*, vol. 7, nº 6, p. e39221, juin 2012.
[11] B. Hesper et P. Hogeweg, « Bioinformatica: een werkconcept », p. 1(6): 28–29, 1970.
[12] S. B. Needleman et C. D. Wunsch, « A general method applicable to the search for similarities in the amino acid sequence of two proteins », *J. Mol. Biol.*, vol. 48, nº 3, p. 443-453, mars 1970.
[13] A. M. Maxam et W. Gilbert, « A new method for sequencing DNA », *Proc. Natl. Acad. Sci. U. S. A.*, vol. 74, nº 2, p. 560-564, févr. 1977.
[14] F. Sanger et A. R. Coulson, « The use of thin acrylamide gels for DNA sequencing », *FEBS Lett.*, vol. 87, nº 1, p. 107-110, mars 1978.
[15] F. Sanger *et al.*, « The nucleotide sequence of bacteriophage φX174 », *J. Mol. Biol.*,

vol. 125, nº 2, p. 225-246, oct. 1978.
[16] K. Mullis, F. Faloona, S. Scharf, R. Saiki, G. Horn, et H. Erlich, « Specific Enzymatic Amplification of DNA In Vitro: The Polymerase Chain Reaction », *Cold Spring Harb. Symp. Quant. Biol.*, vol. 51, nº 0, p. 263-273, janv. 1986.
[17] F. S. Collins, « The Human Genome Project: Lessons from Large-Scale Biology », *Science*, vol. 300, nº 5617, p. 286-290, avr. 2003.
[18] W. Gilbert, « Origin of life: The RNA world », *Nature*, vol. 319, nº 6055, p. 618-618, févr. 1986.
[19] U. Wolf, H. Ritter, N. B. Atkin, et S. Ohno, « Polyploidization in the fish family Cyprinidae, order Cypriniformes. I. DNA-content and chromosome sets in various species of Cyprinidae », *Humangenetik*, vol. 7, nº 3, p. 240-244, 1969.
[20] S. C. Schuster, « Next-generation sequencing transforms today's biology », *Nat. Methods*, vol. 5, nº 1, p. 16-18, janv. 2008.
[21] E. R. Mardis, « DNA sequencing technologies: 2006–2016 », *Nat. Protoc.*, vol. 12, nº 2, p. 213-218, janv. 2017.
[22] R. R. Schaller, « Moore's law: past, present and future », *IEEE Spectr.*, vol. 34, nº 6, p. 52-59, juin 1997.
[23] J. Eid *et al.*, « Real-time DNA sequencing from single polymerase molecules », *Science*, vol. 323, nº 5910, p. 133-138, janv. 2009.
[24] D. Branton *et al.*, « The potential and challenges of nanopore sequencing », *Nat. Biotechnol.*, vol. 26, nº 10, p. 1146-1153, oct. 2008.
[25] G. Church, D. W. Deamer, D. Branton, R. Baldarelli, et J. Kasianowicz, « United States Patent: 5795782 - Characterization of individual polymer molecules based on monomer-interface interactions », 5795782, 18-août-1998.
[26] S. L. Castro-Wallace *et al.*, « Nanopore DNA Sequencing and Genome Assembly on the International Space Station », *Sci. Rep.*, vol. 7, nº 1, déc. 2017.
[27] J. Quick *et al.*, « Real-time, portable genome sequencing for Ebola surveillance », *Nature*, vol. 530, nº 7589, p. 228-232, févr. 2016.
[28] M. Jain *et al.*, « Nanopore sequencing and assembly of a human genome with ultra-long reads », *Nat. Biotechnol.*, vol. 36, nº 4, p. 338-345, janv. 2018.
[29] Y. Erlich et D. Zielinski, « DNA Fountain enables a robust and efficient storage architecture », *Science*, vol. 355, nº 6328, p. 950-954, mars 2017.
[30] M. Ferrer, F. Martinezabarca, et P. Golyshin, « Mining genomes and 'metagenomes' for novel catalysts », *Curr. Opin. Biotechnol.*, vol. 16, nº 6, p. 588-593, déc. 2005.
[31] D. J. Rigden et X. M. Fernández, « The 2018 Nucleic Acids Research database issue and the online molecular biology database collection », *Nucleic Acids Res.*, vol. 46, nº D1, p. D1-D7, janv. 2018.
[32] W. R. Pearson et D. J. Lipman, « Improved tools for biological sequence comparison », *Proc. Natl. Acad. Sci. U. S. A.*, vol. 85, nº 8, p. 2444-2448, avr. 1988.
[33] P. J. A. Cock, C. J. Fields, N. Goto, M. L. Heuer, et P. M. Rice, « The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants », *Nucleic Acids Res.*, vol. 38, nº 6, p. 1767-1771, avr. 2010.
[34] H. Li *et al.*, « The Sequence Alignment/Map format and SAMtools », *Bioinformatics*, vol. 25, nº 16, p. 2078-2079, août 2009.
[35] P. Deutsch, « DEFLATE Compressed Data Format Specification version 1.3 », RFC Editor, RFC1951, mai 1996.
[36] J. Ziv et A. Lempel, « A universal algorithm for sequential data compression », *IEEE Trans. Inf. Theory*, vol. 23, nº 3, p. 337-343, mai 1977.
[37] D. Huffman, « A Method for the Construction of Minimum-Redundancy Codes », *Proc. IRE*, vol. 40, nº 9, p. 1098-1101, sept. 1952.

[38] S. Grumbach et F. Tahi, « Compression of DNA sequences », in *[Proceedings] DCC `93: Data Compression Conference*, Snowbird, UT, USA, 1993, p. 340-350.
[39] S. Grumbach et F. Tahi, « A new challenge for compression algorithms: Genetic sequences », *Inf. Process. Manag.*, vol. 30, nº 6, p. 875-886, nov. 1994.
[40] E. Rivals, J.-P. Delahaye, M. Dauchet, et O. Delgrange, « A guaranteed compression scheme for repetitive DNA sequences », in *Proceedings of Data Compression Conference - DCC '96*, Snowbird, UT, USA, 1996, p. 453.
[41] Xin Chen, S. Kwong, et Ming Li, « A compression algorithm for DNA sequences », *IEEE Eng. Med. Biol. Mag.*, vol. 20, nº 4, p. 61-66, août 2001.
[42] U. Ghoshdastider et B. Saha, « GenomeCompress: A Novel Algorithm for DNA Compression », p. 6, 2007.
[43] P. R. Rajeswari, D. A. Apparao, et D. R. K. Kumar, « HUFFBIT COMPRESS – ALGORITHM TO COMPRESS DNA SEQUENCES USING EXTENDED BINARY TREES. », p. 6, 2005.
[44] X. Xie, S. Zhou, et J. Guan, « CoGI: Towards Compressing Genomes as an Image », *IEEE/ACM Trans. Comput. Biol. Bioinform.*, vol. 12, nº 6, p. 1275-1285, nov. 2015.
[45] V. I. Levenshtein, « Binary codes capable of correcting deletions, insertions, and reversals », *Volume 163, Number 4, 845–848 - Dokl. Akad. Nauk SSSR*, 1965.
[46] T. F. Smith et M. S. Waterman, « Identification of common molecular subsequences », *J. Mol. Biol.*, vol. 147, nº 1, p. 195-197, mars 1981.
[47] J. P. Dumas et J. Ninio, « Efficient algorithms for folding and comparing nucleic acid sequences », *Nucleic Acids Res.*, vol. 10, nº 1, p. 197-206, janv. 1982.
[48] M. Farach, P. Ferragina, et S. Muthukrishnan, « Overcoming the memory bottleneck in suffix tree construction », in *Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat. No.98CB36280)*, Palo Alto, CA, USA, 1998, p. 174-183.
[49] D. Minkley, M. J. Whitney, S.-H. Lin, M. G. Barsky, C. Kelly, et C. Upton, « Suffix tree searcher: exploration of common substrings in large DNA sequence sets », *BMC Res. Notes*, vol. 7, p. 466, juill. 2014.
[50] A. L. Delcher, S. Kasif, R. D. Fleischmann, J. Peterson, O. White, et S. L. Salzberg, « Alignment of whole genomes », *Nucleic Acids Res.*, vol. 27, nº 11, p. 2369-2376, janv. 1999.
[51] M. I. Abouelhoda, S. Kurtz, et E. Ohlebusch, « Replacing suffix trees with enhanced suffix arrays », *J. Discrete Algorithms*, vol. 2, nº 1, p. 53-86, mars 2004.
[52] H. Li et R. Durbin, « Fast and accurate short read alignment with Burrows-Wheeler transform », *Bioinforma. Oxf. Engl.*, vol. 25, nº 14, p. 1754-1760, juill. 2009.
[53] R. Li *et al.*, « SOAP2: an improved ultrafast tool for short read alignment », *Bioinforma. Oxf. Engl.*, vol. 25, nº 15, p. 1966-1967, août 2009.
[54] B. Langmead, C. Trapnell, M. Pop, et S. L. Salzberg, « Ultrafast and memory-efficient alignment of short DNA sequences to the human genome », *Genome Biol.*, vol. 10, nº 3, p. R25, 2009.
[55] Z. Ning, A. J. Cox, et J. C. Mullikin, « SSAHA: a fast search method for large DNA databases », *Genome Res.*, vol. 11, nº 10, p. 1725-1729, oct. 2001.
[56] C. Fondrat et P. Dessen, « A rapid access motif database (RAMdb) with a search algorithm for the retrieval patterns in nucleic acids or protein databanks », *Comput. Appl. Biosci. CABIOS*, vol. 11, nº 3, p. 273-279, juin 1995.
[57] W. J. Kent, « BLAT--the BLAST-like alignment tool », *Genome Res.*, vol. 12, nº 4, p. 656-664, avr. 2002.
[58] E. A. Cheever, D. B. Searls, W. Karunaratne, et G. C. Overton, « Using signal processing techniques for DNA sequence comparison », in *Proceedings of the Fifteenth Annual Northeast Bioengineering Conference*, Boston, MA, USA, 1989, p. 173-174.

[59] E. R. Dougherty, I. Shmulevich, et M. L. Bittner, « Genomic Signal Processing: The Salient Issues », *EURASIP J. Adv. Signal Process.*, vol. 2004, nº 1, déc. 2004.
[60] J. Gu, X. Wang, et J. Zhao, « A Novel Method for Multiple Sequence Alignment Based on Wavelet Package Transform », vol. 11, nº 8, p. 9, 2005.
[61] K. Katoh, K. Misawa, K. Kuma, et T. Miyata, « MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform », *Nucleic Acids Res.*, vol. 30, nº 14, p. 3059-3066, juill. 2002.
[62] H. Skutkova, M. Vitek, K. Sedlar, et I. Provaznik, « Progressive alignment of genomic signals by multiple dynamic time warping », *J. Theor. Biol.*, vol. 385, p. 20-30, nov. 2015.
[63] T. Rakthanmanon *et al.*, « Searching and mining trillions of time series subsequences under dynamic time warping », in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '12*, Beijing, China, 2012, p. 262.
[64] S. Saini et L. Dewan, « Application of discrete wavelet transform for analysis of genomic sequences of Mycobacterium tuberculosis », *SpringerPlus*, vol. 5, nº 1, déc. 2016.
[65] J. Felsenstein, S. Sawyer, et R. Kochin, « An efficient method for matching nucleic acid sequences », *Nucleic Acids Res.*, vol. 10, nº 1, p. 133-139, janv. 1982.
[66] D. C. Benson, « Fourier methods for biosequence analysis », *Nucleic Acids Res.*, vol. 18, nº 21, p. 6305-6310, nov. 1990.
[67] E. A. Cheever, G. C. Overton, et D. B. Searls, « Fast Fourier transform-based correlation of DNA sequences using complex plane encoding », *Bioinformatics*, vol. 7, nº 2, p. 143-154, 1991.
[68] S. Rajasekaran, X. Jin, et J. L. Spouge, « The efficient computation of position-specific match scores with the fast fourier transform », *J. Comput. Biol. J. Comput. Mol. Cell Biol.*, vol. 9, nº 1, p. 23-33, 2002.
[69] A. L. Rockwood, D. K. Crockett, J. R. Oliphant, et K. S. J. Elenitoba-Johnson, « Sequence alignment by cross-correlation », *J. Biomol. Tech. JBT*, vol. 16, nº 4, p. 453-458, déc. 2005.
[70] R. Pagare et A. Shinde, « A Study on Image Annotation Techniques », *Int. J. Comput. Appl.*, vol. 37, nº 6, p. 42-45, janv. 2012.
[71] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, et Li Fei-Fei, « ImageNet: A large-scale hierarchical image database », in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, 2009, p. 248-255.
[72] K. Barnard, P. Duygulu, D. Forsyth, et N. de Freitas, « Matching Words and Pictures », p. 29, 2003.
[73] V. Lavrenko, R. Manmatha, et J. Jeon, « A Model for Learning the Semantics of Pictures », p. 8, 2003.
[74] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, et R. Jain, « Content-based image retrieval at the end of the early years », *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, nº 12, p. 1349-1380, déc. 2000.
[75] P. Aigrain, H. Zhang, et D. Petkovic, « Content-based representation and retrieval of visual media: A state-of-the-art review », *Multimed. Tools Appl.*, vol. 3, nº 3, p. 179-202, nov. 1996.
[76] M. J. Swain et D. H. Ballard, « Color indexing », *Int. J. Comput. Vis.*, vol. 7, nº 1, p. 11-32, nov. 1991.
[77] T. Hurtut, Y. Gousseau, et F. Schmitt, « Adaptive image retrieval based on the spatial organization of colors », *Comput. Vis. Image Underst.*, vol. 112, nº 2, p. 101-113, nov. 2008.
[78] T.-C. Lu et C.-C. Chang, « Color image retrieval technique based on color features and image bitmap », *Inf. Process. Manag.*, vol. 43, nº 2, p. 461-472, mars 2007.
[79] O. A. B. Penatti et R. da S. Torres, « Color Descriptors for Web Image Retrieval: A

Comparative Study », in *2008 XXI Brazilian Symposium on Computer Graphics and Image Processing*, Campo Grande, Brazil, 2008, p. 163-170.
[80] R. M. Haralick, K. Shanmugam, et I. Dinstein, « Textural Features for Image Classification », *IEEE Trans. Syst. Man Cybern.*, vol. SMC-3, nº 6, p. 610-621, nov. 1973.
[81] Xiaoou Tang, « Texture information in run-length matrices », *IEEE Trans. Image Process.*, vol. 7, nº 11, p. 1602-1609, nov. 1998.
[82] V. Murino, C. Ottonello, et S. Pagnan, « Noisy texture classification: A higher-order statistics approach », *Pattern Recognit.*, vol. 31, nº 4, p. 383-393, avr. 1998.
[83] R. Lopes, P. Dubois, I. Bhouri, M. H. Bedoui, S. Maouche, et N. Betrouni, « Local fractal and multifractal features for volumic texture characterization », *Pattern Recognit.*, vol. 44, nº 8, p. 1690-1697, août 2011.
[84] T. Ojala, M. Pietikainen, et T. Maenpaa, « Multiresolution gray-scale and rotation invariant texture classification with local binary patterns », *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, nº 7, p. 971-987, juill. 2002.
[85] F. S. Cohen, Z. Fan, et M. A. Patel, « Classification of rotated and scaled textured images using Gaussian Markov random field models », *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, nº 2, p. 192-202, févr. 1991.
[86] B. S. Reddy et B. N. Chatterji, « An FFT-based technique for translation, rotation, and scale-invariant image registration », *IEEE Trans. Image Process.*, vol. 5, nº 8, p. 1266-1271, août 1996.
[87] J. Z. Wang, G. Wiederhold, O. Firschein, et S. Xin Wei, « Content-based image indexing and searching using Daubechies' wavelets », *Int. J. Digit. Libr.*, vol. 1, nº 4, p. 311-328, mars 1998.
[88] D. Zhang, A. Wong, M. Indrawan, et G. Lu, « Content-based Image Retrieval Using Gabor Texture Features », p. 4, 2000.
[89] E. Persoon et K.-S. Fu, « Shape Discrimination Using Fourier Descriptors », *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, nº 3, p. 388-397, mai 1986.
[90] T. Pavlidis, « A review of algorithms for shape analysis », *Comput. Graph. Image Process.*, vol. 7, nº 2, p. 243-258, avr. 1978.
[91] C. T. Zahn et R. Z. Roskies, « Fourier Descriptors for Plane Closed Curves », *IEEE Trans. Comput.*, vol. C-21, nº 3, p. 269-281, mars 1972.
[92] H. Freeman, « Computer Processing of Line-Drawing Images », *ACM Comput. Surv.*, vol. 6, nº 1, p. 57-97, janv. 1974.
[93] J. Iivarinen et A. J. E. Visa, « Shape recognition of irregular objects », présenté à Photonics East '96, Boston, MA, 1996, p. 25-32.
[94] A. K. Jain et A. Vailaya, « Image retrieval using color and shape », *Pattern Recognit.*, vol. 29, nº 8, p. 1233-1244, août 1996.
[95] S. Belongie, J. Malik, et J. Puzicha, « Shape matching and object recognition using shape contexts », *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, nº 4, p. 509-522, avr. 2002.
[96] F. Mokhtarian, S. Abbasi, et J. Kittler, « Robust and Efficient Shape Indexing through Curvature Scale Space », in *Procedings of the British Machine Vision Conference 1996*, Edinburgh, 1996, p. 33.1-33.10.
[97] Ming-Kuei Hu, « Visual pattern recognition by moment invariants », *IEEE Trans. Inf. Theory*, vol. 8, nº 2, p. 179-187, févr. 1962.
[98] A. Khotanzad et Y. H. Hong, « Invariant image recognition by Zernike moments », *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, nº 5, p. 489-497, mai 1990.
[99] N. Boujemaa et J. Fauqueur, « Ikona : Interactive Specific and Generic Image Retrieval », 2001.
[100] D. G. Lowe, « Object recognition from local scale-invariant features », in *Proceedings*

of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, 1999, p. 1150-1157 vol.2.
[101] N. Boujemaa, J. Fauqueur, et V. Gouet, « What's beyond query by example? », *IMEDIA - Image and multimedia indexing, browsing and retrieval*, Inria Paris-Rocquencourt, 2003.
[102] K. Mikolajczyk et C. Schmid, « A performance evaluation of local descriptors », *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, nº 10, p. 1615-1630, oct. 2005.
[103] F. Attneave, « Some informational aspects of visual perception », *Psychol. Rev.*, vol. 61, nº 3, p. 183-193, mai 1954.
[104] H. Moravec, « Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover », *Carnegie Mellon University*, 1980.
[105] P. Beaudet, « Rotationally invariant image operators », *In Proceedings of the International Conference of Pattern Recognition, pp. 579–583*, 1978.
[106] C. Harris, « A combined corner and edge detector », *Proceedings Fourth Alvey Vision Conference*, Manchester, p. 147-151, 1988.
[107] J. L. Crowley et A. C. Sanderson, « Multiple resolution representation and probabilistic matching of 2-d gray-scale shape », *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, nº 1, p. 113-121, janv. 1987.
[108] T. Lindeberg, « Scale-space for discrete signals », *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, nº 3, p. 234-254, mars 1990.
[109] T. Lindeberg, « Feature Detection with Automatic Scale Selection », *International Journal of Computer Vision*, p. pp 79–116, 1998.
[110] K. Mikolajczyk et K. Mikolajczyk, « Scale & Affine Invariant Interest Point Detectors », *Int. J. Comput. Vis.*, vol. 60, nº 1, p. 63-86, oct. 2004.
[111] D. G. Lowe, « Distinctive Image Features from Scale-Invariant Keypoints », *Int. J. Comput. Vis.*, vol. 60, nº 2, p. 91-110, nov. 2004.
[112] H. Bay, A. Ess, T. Tuytelaars, et L. Van Gool, « Speeded-Up Robust Features (SURF) », *Comput. Vis. Image Underst.*, vol. 110, nº 3, p. 346-359, juin 2008.
[113] S. R. Dubey, S. K. Singh, et R. K. Singh, « Local Wavelet Pattern: A New Feature Descriptor for Image Retrieval in Medical CT Databases », *IEEE Trans. Image Process.*, vol. 24, nº 12, p. 5892-5903, déc. 2015.
[114] S. C. F. Lin, C. Y. Wong, G. Jiang, M. A. Rahman, et N. M. Kwok, « Radial fourier analysis (RFA) image descriptor », in *2014 11th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, Xiamen, China, 2014, p. 814-819.
[115] C. Gottschlich, « Convolution Comparison Pattern: An Efficient Local Image Descriptor for Fingerprint Liveness Detection », *PLOS ONE*, vol. 11, nº 2, p. e0148552, févr. 2016.
[116] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, et P. Fua, « BRIEF: Computing a Local Binary Descriptor Very Fast », *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, nº 7, p. 1281-1298, juill. 2012.
[117] Y. Gong et S. Lazebnik, « Iterative quantization: A procrustean approach to learning binary codes », in *CVPR 2011*, Colorado Springs, CO, USA, 2011, p. 817-824.
[118] X. Boix, M. Gygli, G. Roig, et L. Van Gool, « Sparse Quantization for Patch Description », in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, Portland, OR, USA, 2013, p. 2842-2849.
[119] Y. Gao, Y. Qiao, Z. Li, et C. Xu, « LTD: Local Ternary Descriptor for image matching », in *2013 IEEE International Conference on Information and Automation (ICIA)*, Yinchuan, China, 2013, p. 1375-1380.
[120] N. S. Chang et K. S. Fu, « A relational database system for images », in *Pictorial Information Systems*, vol. 80, S. K. Chang et K. S. Fu, Éd. Berlin, Heidelberg: Springer Berlin

Heidelberg, 1980, p. 288-321.
[121] E. Milanov, « The RSA Algorithm ». 2009.
[122] R. Merkle Charles, *Secrecy, authentication, and public key systems*, Stanford University Stanford, CA, USA. 1979.
[123] R. C. Merkle, « One Way Hash Functions and DES », in *Advances in Cryptology — CRYPTO' 89 Proceedings*, vol. 435, G. Brassard, Éd. New York, NY: Springer New York, 1990, p. 428-446.
[124] J. Naus, « Probabilities for a Generalized Birthday Problem », *J. Am. Stat. Assoc.*, vol. 69, nº 347, p. 810-815, sept. 1974.
[125] H. Feistel, *Cryptography and Computer Privacy*, Scientific American., vol. Vol. 228, No. 5, p. 15-23. 1973.
[126] C. E. Shannon, « Communication Theory of Secrecy Systems* », *Bell Syst. Tech. J.*, vol. 28, nº 4, p. 656-715, oct. 1949.
[127] J. Wang, H. T. Shen, J. Song, et J. Ji, « Hashing for Similarity Search: A Survey », *ArXiv14082927 Cs*, août 2014.
[128] P. Indyk et R. Motwani, « Approximate nearest neighbors: towards removing the curse of dimensionality », in *Proceedings of the thirtieth annual ACM symposium on Theory of computing - STOC '98*, Dallas, Texas, United States, 1998, p. 604-613.
[129] C. Zauner, M. Steinebach, et E. Hermann, « Rihamark: perceptual image hash benchmarking », présenté à IS&T/SPIE Electronic Imaging, San Francisco Airport, California, USA, 2011, p. 78800X.
[130] W. Diffie et M. Hellman, « New directions in cryptography », *IEEE Trans. Inf. Theory*, vol. 22, nº 6, p. 644-654, nov. 1976.
[131] B. Preneel, « The First 30 Years of Cryptographic Hash Functions and the NIST SHA-3 Competition », in *Topics in Cryptology - CT-RSA 2010*, vol. 5985, J. Pieprzyk, Éd. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, p. 1-14.
[132] B. Kaliski, « The MD2 Message-Digest Algorithm », RFC Editor, RFC1319, avr. 1992.
[133] R. L. Rivest, « The MD4 Message Digest Algorithm », in *Advances in Cryptology-CRYPT0' 90*, vol. 537, A. J. Menezes et S. A. Vanstone, Éd. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991, p. 303-311.
[134] R. Rivest, « The MD5 Message-Digest Algorithm », RFC Editor, RFC1321, avr. 1992.
[135] R. Rivest, « The MD6 hash function - A proposal to NIST for SHA-3 », RFC Editor, oct. 2008.
[136] H. Dobbertin, A. Bosselaers, et B. Preneel, « RIPEMD-160: A strengthened version of RIPEMD », in *Fast Software Encryption*, vol. 1039, D. Gollmann, Éd. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, p. 71-82.
[137] W. Stallings, « The Whirlpool Secure Hash Function », *Cryptologia*, vol. 30, nº 1, p. 55-67, janv. 2006.
[138] D. Eastlake et P. Jones, « US Secure Hash Algorithm 1 (SHA1) », RFC Editor, RFC3174, sept. 2001.
[139] S. Turner, « Using SHA2 Algorithms with Cryptographic Message Syntax », RFC Editor, RFC5754, janv. 2010.
[140] M. J. Dworkin, « SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions », National Institute of Standards and Technology, NIST FIPS 202, juill. 2015.
[141] I. B. Damgård, « A Design Principle for Hash Functions », in *Advances in Cryptology — CRYPTO' 89 Proceedings*, vol. 435, G. Brassard, Éd. New York, NY: Springer New York, 1990, p. 416-427.
[142] B. Preneel, R. Govaerts, et J. Vandewalle, « Hash functions based on block ciphers: a synthetic approach », in *Advances in Cryptology — CRYPTO' 93*, vol. 773, D. R. Stinson, Éd.

Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, p. 368-378.
[143] S. . Matyas, C. . Meyer, et J. Oseas, « Generating Strong One-Way Functions With Cryptographic Algorithm », 1985.
[144] K. Vlastimil, *Tunnels in Hash Functions: MD5 Collisions Within a Minute*. 2006.
[145] G. Bertoni, J. Daemen, M. Peeters, G. Van Assche, et R. Van Keer, « KECCAK, implementation overview », 2012.
[146] J.-P. Aumasson, W. Meier, R. C.-W. Phan, et L. Henzen, *The Hash Function BLAKE*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014.
[147] D. Gligoroski, V. Klima, S. Knapskog Johan, M. El-Hadedy, et J. Amundsen, « Cryptographic hash function Blue Midnight Wish », 2009.
[148] P. Gauravaram *et al.*, « Grøstl – a SHA-3 candidate », 2011.
[149] R. Bhattacharyya, A. Mandal, et M. Nandi, « Security Analysis of the Mode of JH Hash Function », in *Fast Software Encryption*, vol. 6147, S. Hong et T. Iwata, Éd. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, p. 168-191.
[150] N. Ferguson *et al.*, « The Skein Hash Function Family », 2010.
[151] C. De Cannière, H. Sato, et D. Watanabe, « Hash Function Luffa Specification », 2008.
[152] D. J. Bernstein, « CubeHash specification (2.B.1) », 2010.
[153] E. Biham et O. Dunkelman, « The SHAvite-3 Hash Function », 2009.
[154] R. Benadjila, O. Billet, H. Gilbert, G. Macario-Rat, T. Peyrin, et M. Robshaw, « SHA-3 Proposal: ECHO », *Orange Labs*, Issy-les-Moulineaux, France, 2008.
[155] M. Bellare, J. Kilian, et P. Rogaway, « The Security of the Cipher Block Chaining Message Authentication Code », *J. Comput. Syst. Sci.*, vol. 61, nº 3, p. 362-399, déc. 2000.
[156] R. L. Rivest, A. Shamir, et L. Adleman, « A method for obtaining digital signatures and public-key cryptosystems », *Commun. ACM*, vol. 21, nº 2, p. 120-126, févr. 1978.
[157] T. Elgamal, « A public key cryptosystem and a signature scheme based on discrete logarithms », *IEEE Trans. Inf. Theory*, vol. 31, nº 4, p. 469-472, juill. 1985.
[158] D. Ravilla et C. S. R. Putta, « Implementation of HMAC-SHA256 algorithm for hybrid routing protocols in MANETs », in *2015 International Conference on Electronic Design, Computer Networks & Automated Verification (EDCAV)*, Shillong, India, 2015, p. 154-159.
[159] C.-C. Chang et W.-Y. Liao, « A remote password authentication scheme based upon ElGamal's signature scheme », *Comput. Secur.*, vol. 13, nº 2, p. 137-144, avr. 1994.
[160] S. Nakamoto, « Bitcoin: A Peer-to-Peer Electronic Cash System », 2008.
[161] B. Jenkins, « Hash functions », *Dr. Dobb's Journal*, 1997.
[162] A. Appleby, « Murmur hash function », 2008.
[163] G. Pike et J. Alakuijala, « Introducing CityHash », 2011.
[164] B. Jenkins, « SpookyHash: a 128-bit noncryptographic hash », 2011.
[165] Y. Collet, « xxHash, an extremely fast non-cryptographic hash algorithm », 2016.
[166] M. S. Waterman, *Introduction to computational biology: maps, sequences and genomes*, 1. CRC Press reprint. Boca Raton, Fla.: Chapman & Hall/CRC, 2000.
[167] P. Pandey, M. A. Bender, R. Johnson, et R. Patro, « Squeakr: An Exact and Approximate k-mer Counting System », mars 2017.
[168] H. Mohamadi, J. Chu, B. P. Vandervalk, et I. Birol, « ntHash: recursive nucleotide hashing », *Bioinforma. Oxf. Engl.*, vol. 32, nº 22, p. 3492-3494, 15 2016.
[169] G. Pike, « Introducing CityHash », 2011.
[170] P. GAGNIUC et C. IONESCU-TÎRGOVIŞTE, « GHDNA: A HASH FUNCTION FOR DNA SEGMENT-BASED ALIGMENTS AND MOTIF SEARCH », 2014.
[171] J. D. Cohen, « Recursive hashing functions for n-grams », *ACM Trans. Inf. Syst.*, vol. 15, nº 3, p. 291-320, juill. 1997.

[172] J. Fridrich et M. Goljan, « Robust hash functions for digital watermarking », in *Proceedings International Conference on Information Technology: Coding and Computing (Cat. No.PR00540)*, Las Vegas, NV, USA, 2000, p. 178-183.
[173] T. Kalker, J. Haitsma, et J. C. Oostveen, « Issues with digital watermarking and perceptual hashing », présenté à ITCom 2001: International Symposium on the Convergence of IT and Communications, Denver, CO, 2001, p. 189-197.
[174] S. Fakhfakh, M. Tmar, et W. Mahdi, « Image Retrieval Based on Using Hamming Distance », *Procedia Comput. Sci.*, vol. 73, p. 320-327, 2015.
[175] V. Monga, D. Vats, et B. L. Evans, « Image Authentication Under Geometric Attacks Via Structure Matching », in *2005 IEEE International Conference on Multimedia and Expo*, Amsterdam, The Netherlands, 2005, p. 229-232.
[176] D. P. Huttenlocher, G. A. Klanderman, et W. J. Rucklidge, « Comparing images using the Hausdorff distance », *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, nº 9, p. 850-863, sept. 1993.
[177] P. Cano, E. Batlle, T. Kalker, et J. Haitsma, « A Review of Audio Fingerprinting », *J. VLSI Signal Process. Syst. Signal Image Video Technol.*, vol. 41, nº 3, p. 271-284, nov. 2005.
[178] J. Haitsma, « Audio Fingerprinting "a New Technology to Identify Music" Title: Audio Fingerprinting "a New Technology to Identify Music" Table of Contents », 2002.
[179] R. Lancini, F. Mapelli, et R. Pezzano, « Audio content identification by using perceptual hashing », in *2004 IEEE International Conference on Multimedia and Expo (ICME) (IEEE Cat. No.04TH8763)*, Taipei, Taiwan, 2004, p. 739-742.
[180] A. L.-C. Wang, « An Industrial-Strength Audio Search Algorithm », *Proc. 4 Th Int. Conf. Music Inf. Retr.*, p. 7, 2003.
[181] J. Pinguier, « Indexation sonore : recherche de composantes primaires pour une structuration audiovisuelle », 2004.
[182] C. J. C. Burges, J. C. Platt, et S. Jana, « Distortion discriminant analysis for audio fingerprinting », *IEEE Trans. Speech Audio Process.*, vol. 11, nº 3, p. 165-174, mai 2003.
[183] J. Herre *et al.*, « The Reference Model Architecture for MPEG Spatial Audio Coding », 2005.
[184] A. Wang et D. Culbert, « Robust and invariant audio pattern matching », US20090265174A9, 2002.
[185] S. Baluja, M. Covell, et S. Ioffe, « Permutation grouping: intelligent Hash function design for audio & image retrieval », in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, Las Vegas, NV, USA, 2008, p. 2137-2140.
[186] A. Ramalingam et S. Krishnan, « Gaussian Mixture Modeling Using Short Time Fourier Transform Features for Audio Fingerprinting », in *2005 IEEE International Conference on Multimedia and Expo*, Amsterdam, The Netherlands, 2005, p. 1146-1149.
[187] L. de C. T. Gomes, P. Cano, E. G□mez, M. Bonnet, et E. Batlle, « Audio Watermarking and Fingerprinting: For Which Applications? », *J. New Music Res.*, vol. 32, nº 1, p. 65-81, mars 2003.
[188] J. Laroche, « Process for identifying audio content », US6453252B1, 2000.
[189] D. Fragoulis, G. Rousopoulos, T. Panagopoulos, C. Alexiou, et C. Papaodysseus, « On the automated recognition of seriously distorted musical recordings », *IEEE Trans. Signal Process.*, vol. 49, nº 4, p. 898-908, avr. 2001.
[190] C. Papaodysseus, G. Rousopoulos, et D. Fragkoulis, « New approach to the automatic recognition of musical recordings », *Journal of the Audio Engineering Society. Audio Engineering Society*, p. 49(1):23-35, 2001.
[191] J. Haitsma, « Robust Audio Hashing for Content Identification », *In Content-Based Multimedia Indexing (CBMI*, 2001.
[192] M. Park, H.-R. Kim, et S. H. Yang, « Frequency-Temporal Filtering for a Robust

Audio Fingerprinting Scheme in Real-Noise Environments », *ETRI J.*, vol. 28, nº 4, p. 509-512, août 2006.
[193] C. Bellettini et G. Mazzini, « A Framework for Robust Audio Fingerprinting », *J. Commun.*, vol. 5, nº 5, mai 2010.
[194] M. K. Mıhçak et R. Venkatesan, « New Iterative Geometric Methods for Robust Perceptual Image Hashing », in *Security and Privacy in Digital Rights Management*, vol. 2320, T. Sander, Éd. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, p. 13-21.
[195] V. Monga et B. L. Evans, « Perceptual Image Hashing Via Feature Points: Performance Evaluation and Tradeoffs », *IEEE Trans. Image Process.*, vol. 15, nº 11, p. 3452-3465, nov. 2006.
[196] E. Y. Chang, J. Z. Wang, C. Li, et G. Wiederhold, « RIME: a replicated image detector for the World Wide Web », présenté à Photonics East (ISAM, VVDC, IEMB), Boston, MA, 1998, p. 58-67.
[197] C. Kim, « Content-based image copy detection », *Signal Process. Image Commun.*, vol. 18, nº 3, p. 169-184, mars 2003.
[198] R. Venkatesan, S.-M. Koon, M. H. Jakubowski, et P. Moulin, « Robust image hashing », in *Proceedings 2000 International Conference on Image Processing (Cat. No.00CH37101)*, Vancouver, BC, Canada, 2000, p. 664-666.
[199] J. Fridrich, « Visual hash for oblivious watermarking », présenté à Electronic Imaging, San Jose, CA, 2000, p. 286-294.
[200] J. S. Seo, J. Haitsma, T. Kalker, et C. D. Yoo, « A robust image fingerprinting system using the Radon transform », *Signal Process. Image Commun.*, vol. 19, nº 4, p. 325-339, avr. 2004.
[201] C. Kailasanathan et R. Safavi Naini, « Image Authentication Surviving Acceptable Modifications Using Statistical Measures and K-mean Segmentation », 2001.
[202] S. Bhattacharjee et M. Kutter, « Compression tolerant image authentication », in *Proceedings 1998 International Conference on Image Processing. ICIP98 (Cat. No.98CB36269)*, Chicago, IL, USA, 1998, vol. 1, p. 435-439.
[203] Y.-K. Chan et C.-C. Chang, « A COLOR IMAGE RETRIEVAL METHOD BASED ON COLOR MOMENT AND COLOR VARIANCE OF ADJACENT PIXELS », *Int. J. Pattern Recognit. Artif. Intell.*, vol. 16, nº 01, p. 113-125, févr. 2002.
[204] S. S. Kozat, R. Venkatesan, et M. K. Mihcak, « Robust perceptual image hashing via matrix invariants », in *2004 International Conference on Image Processing, 2004. ICIP '04.*, Singapore, 2004, vol. 5, p. 3443-3446.
[205] V. Monga et M. K. Mihcak, « Robust Image Hashing Via Non-Negative Matrix Factorizations », in *2006 IEEE International Conference on Acoustics Speed and Signal Processing Proceedings*, Toulouse, France, 2006, vol. 2, p. II-225-II-228.
[206] Jen-Hao Hsiao, Chu-Song Chen, Lee-Feng Chien, et Ming-Syan Chen, « A New Approach to Image Copy Detection Based on Extended Feature Sets », *IEEE Trans. Image Process.*, vol. 16, nº 8, p. 2069-2079, août 2007.
[207] C. Kailasanathan, R. Safavi Naini, et P. Ogunbona, « Compression tolerant DCT based image hash », in *23rd International Conference on Distributed Computing Systems Workshops, 2003. Proceedings.*, Providence, Rhode Island, USA, 2003, p. 562-567.
[208] Ching-Yung Lin et Shih-Fu Chang, « A robust image authentication method distinguishing JPEG compression from malicious manipulation », *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, nº 2, p. 153-168, févr. 2001.
[209] G. Laimer et A. Uhl, « Key-Dependent JPEG2000-Based Robust Hashing for Secure Image Authentication », *EURASIP J. Inf. Secur.*, vol. 2008, nº 1, p. 895174, 2008.
[210] O. Harmanci, V. Monga, et M. K. Mihcak, « Geometrically Invariant Image Watermarking via Robust Perceptual Hashes », in *2006 International Conference on Image*

Processing, Atlanta, GA, 2006, p. 1397-1400.
[211] C.-S. Lu et C.-Y. Hsu, « Geometric distortion-resilient image hashing scheme and its applications on copy detection and authentication », *Multimed. Syst.*, vol. 11, nº 2, p. 159-173, déc. 2005.
[212] X. Lv et Z. J. Wang, « An Extended Image Hashing Concept: Content-Based Fingerprinting Using FJLT », *EURASIP J. Inf. Secur.*, vol. 2009, p. 1-16, 2009.
[213] S. H. Srinivasan et N. Sawant, « Finding near-duplicate images on the web using fingerprints », in *Proceeding of the 16th ACM international conference on Multimedia - MM '08*, Vancouver, British Columbia, Canada, 2008, p. 881.
[214] F. Lefebvre, J. Czyz, et B. Macq, « A robust soft hash algorithm for digital image signature », in *Proceedings 2003 International Conference on Image Processing (Cat. No.03CH37429)*, Barcelona, Spain, 2003, vol. 3, p. II-495-8.
[215] D. Wu, X. Zhou, et X. Niu, « A novel image hash algorithm resistant to print–scan », *Signal Process.*, vol. 89, nº 12, p. 2415-2424, déc. 2009.
[216] Y. Meng et E. Y. Chang, « Image copy detection using dynamic partial function », présenté à Electronic Imaging 2003, Santa Clara, CA, 2003, p. 176.
[217] R. Bayer, « Binary B-trees for virtual memory », in *Proceedings of the 1971 ACM SIGFIDET (now SIGMOD) Workshop on Data Description, Access and Control - SIGFIDET '71*, San Diego, California, 1971, p. 219.
[218] P. Indyk et R. Motwani, « Approximate nearest neighbors: towards removing the curse of dimensionality », in *Proceedings of the thirtieth annual ACM symposium on Theory of computing - STOC '98*, Dallas, Texas, United States, 1998, p. 604-613.
[219] Y.-K. Chan et C.-C. Chang, « A COLOR IMAGE RETRIEVAL METHOD BASED ON COLOR MOMENT AND COLOR VARIANCE OF ADJACENT PIXELS », *Int. J. Pattern Recognit. Artif. Intell.*, vol. 16, nº 01, p. 113-125, févr. 2002.
[220] Yan Ke, D. Hoiem, et R. Sukthankar, « Computer Vision for Music Identification », in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, San Diego, CA, USA, 2005, vol. 1, p. 597-604.
[221] N. Ahmed, T. Natarajan, et K. R. Rao, « Discrete Cosine Transform », *IEEE Trans. Comput.*, vol. C-23, nº 1, p. 90-93, janv. 1974.
[222] E. O. Brigham et R. E. Morrow, « The fast Fourier transform », *IEEE Spectr.*, vol. 4, nº 12, p. 63-70, déc. 1967.
[223] B. Buchanan, « Image Compression (JPEG) », in *Handbook of Data Communications and Networks*, Boston, MA: Springer US, 1999, p. 51-67.
[224] G. K. Wallace, « The JPEG still picture compression standard », *IEEE Trans. Consum. Electron.*, vol. 38, nº 1, p. xviii-xxxiv, févr. 1992.
[225] H. Stark, *Image Recovery Theory and Application.* Saint Louis: Elsevier Science, 2014.
[226] R. W. Hamming, « Error Detecting and Error Correcting Codes », *Bell Syst. Tech. J.*, vol. 29, nº 2, p. 147-160, avr. 1950.
[227] B. Schneier, *Applied cryptography: protocols, algorithms, and source code in C*. New York: Wiley, 1994.
[228] M. Peyravian, A. Roginsky, et A. Kshemkalyani, « On probabilities of hash value matches », *Comput. Secur.*, vol. 17, nº 2, p. 171-176, janv. 1998.
[229] T. H. Jukes et C. R. Cantor, « Evolution of Protein Molecules », in *Mammalian Protein Metabolism*, Elsevier, 1969, p. 21-132.
[230] I. Ito et H. Kiya, « DCT Sign-Only Correlation with Application to Image Matching and the Relationship with Phase-Only Correlation », in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, Honolulu, HI, 2007, p. I-1237-I-1240.

[231] Q. Xiao, Y. Daito, K. Matsumoto, M. Suzuki, et K. Kita, « Fast Search Method for Audio Fingerprinting Systems Based on Query Multiplexing », *Electron. Commun. Jpn.*, vol. 97, n° 6, p. 43-50, juin 2014.
[232] H. Lai et Y. Pan, « Improved Search in Hamming Space using Deep Multi-Index Hashing », *ArXiv171006993 Cs*, oct. 2017.

ANNEXE A

1. Algorithme de la fonction PSH

```
ALGORITHME PSH
  Fonction PSH ( Chaîne S )
       Entier : N, M, W, W', L1, L2, L3, L4, i, j, k
       Matrice : P, P', P'',
       Entier : K <- 0
       Chaîne Binaire : HASH

       Pour i allant de 0 à N - 1
              Pour j allant de 1 à N - 1
                     Si k < W alors
                            Selon S[ k ]
                                   "A" alors P[ i,j ] <- L1
                                   "T" alors P[ i,j ] <- L2
                                   "U" alors P[ i,j ] <- L2
                                   "C" alors P[ i,j ] <- L3
                                   "G" alors P[ i,j ] <- L4
                            Fin selon
                     Fin si
                     k <- k + 1
              Fin pour
       Fin pour

       P' <- TCD( P )
       P'' <- SGN( P' )

       Pour i allant de 0 à N - 1
              Pour J allant de 0 à M - 1
                     HASH <- HASH + P''[ i,j ]
              Fin pour
       Fin pour

       Retourne sous-chaîne ( HASH, 0, W' )
  Fin fonction
```

2. Algorithme de la Distance de Hamming

ALGORITHME DISTANCE_HAMMING

```
Fonction DIST_HAMMING ( Binaire HASH1, Binaire HASH2 )
    Retourne PopCount ( HASH1 Xor HASH2 )
Fin fonction
```

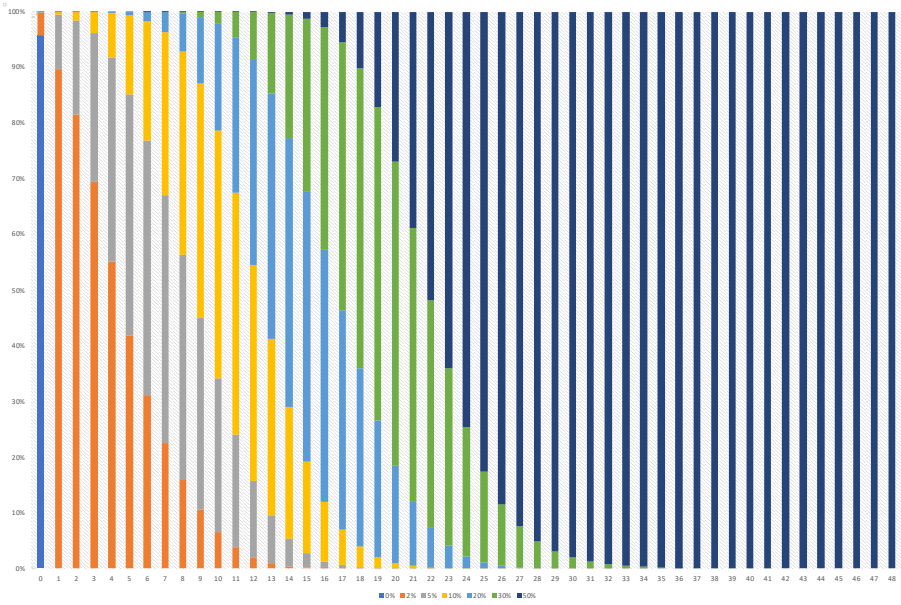
3. Algorithme de la fonction PSC

ALGORITHME PSC

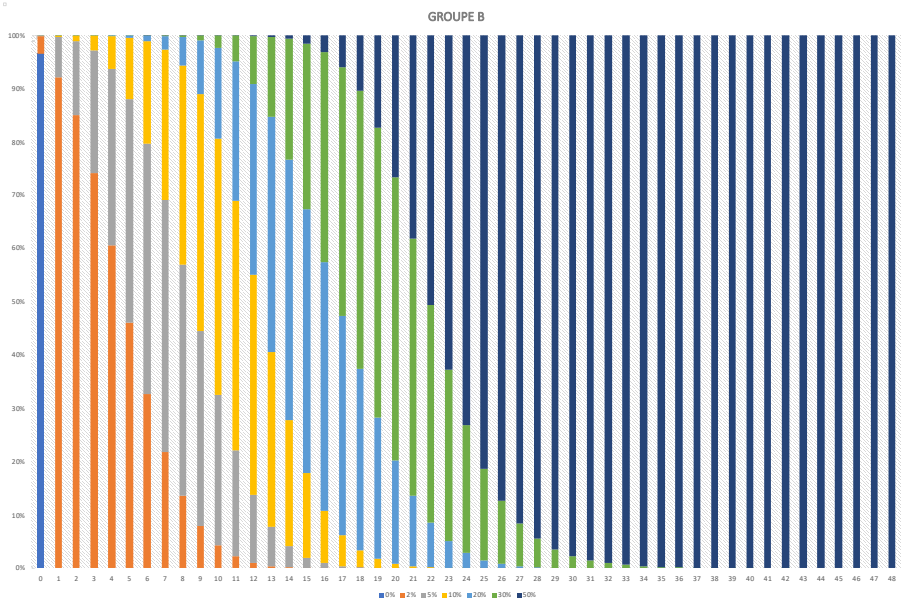
```
Fonction DNA_TO_IMAGE ( Chaîne S )
     Entier : N, M, W, L1, L2, L3, L4, i, j, k
     Matrice : P
     Pour i allant de 0 à N - 1
            Pour j allant de 1 à N - 1
                   Si k < W alors
                          Selon S[ k ]
                                 "A" alors P[ i,j ] <- L1
                                 "T" alors P[ i,j ] <- L2
                                 "U" alors P[ i,j ] <- L2
                                 "C" alors P[ i,j ] <- L3
                                 "G" alors P[ i,j ] <- L4
                          Fin selon
                   Fin si
                   k <- k + 1
            Fin pour
     Fin pour
Retourne P


Fonction PSC ( Chaîne S1, Chaîne S2 )
Matrice : P1, P1', P1'', P2, P2', P2'', H
P1 = DNA_TO_IMAGE ( S1 )
P1' = TCD ( P1 )
P1'' = SGN ( P1' )
P2 = DNA_TO_IMAGE ( S2 )
P2' = TCD ( P2 )
P2'' = SGN ( P2' )
H = Hadamard ( P1'', P2'' )
H = TCDi ( H )
```

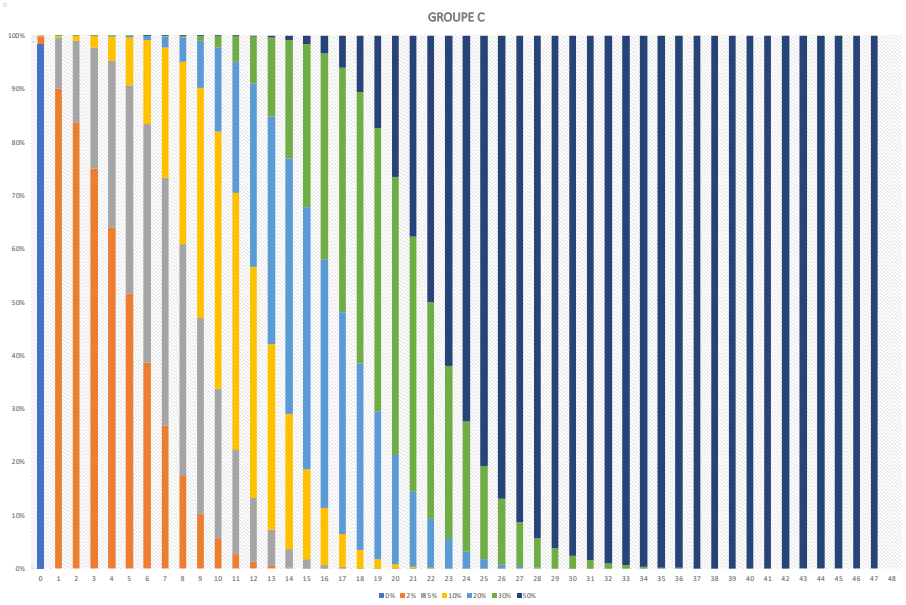
ANNEXE B



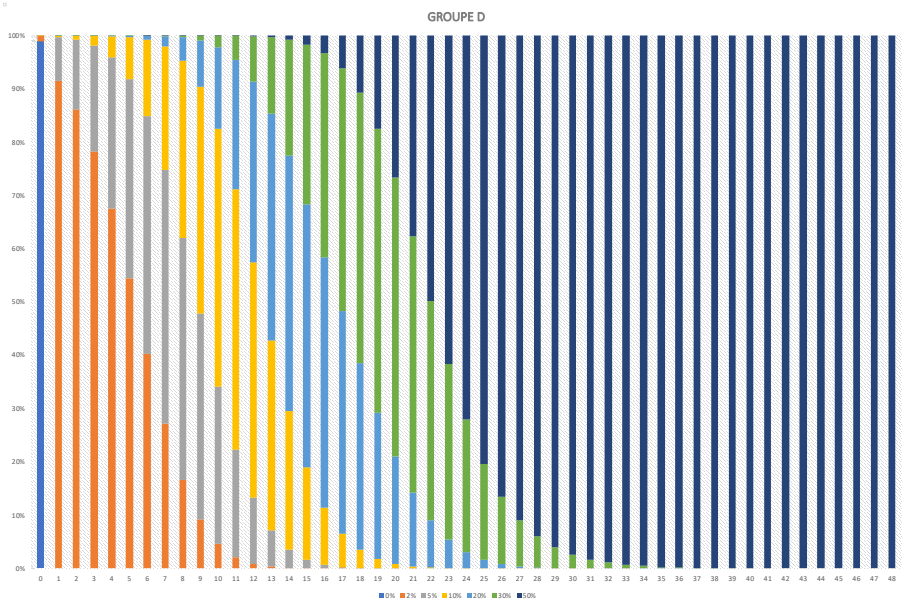
Groupe A



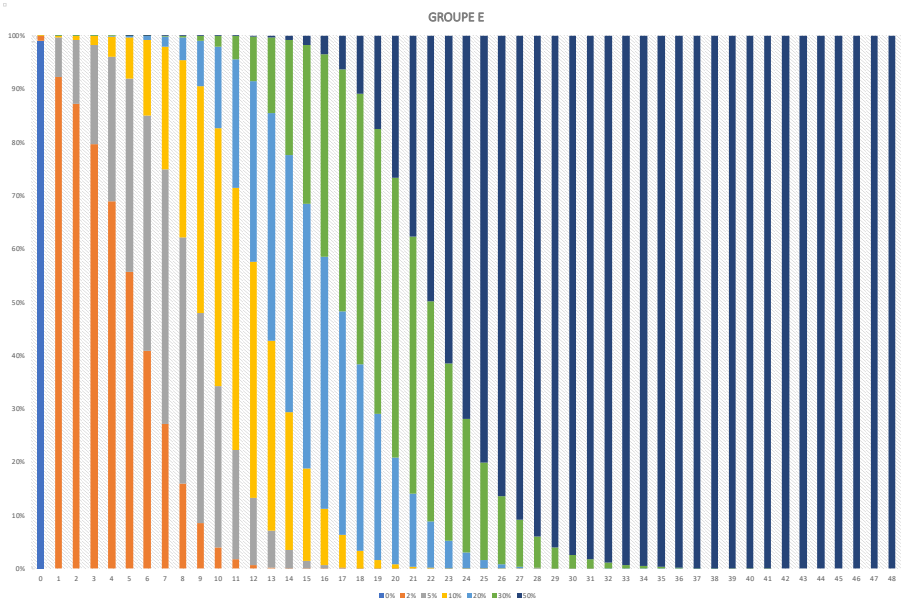
Groupe B



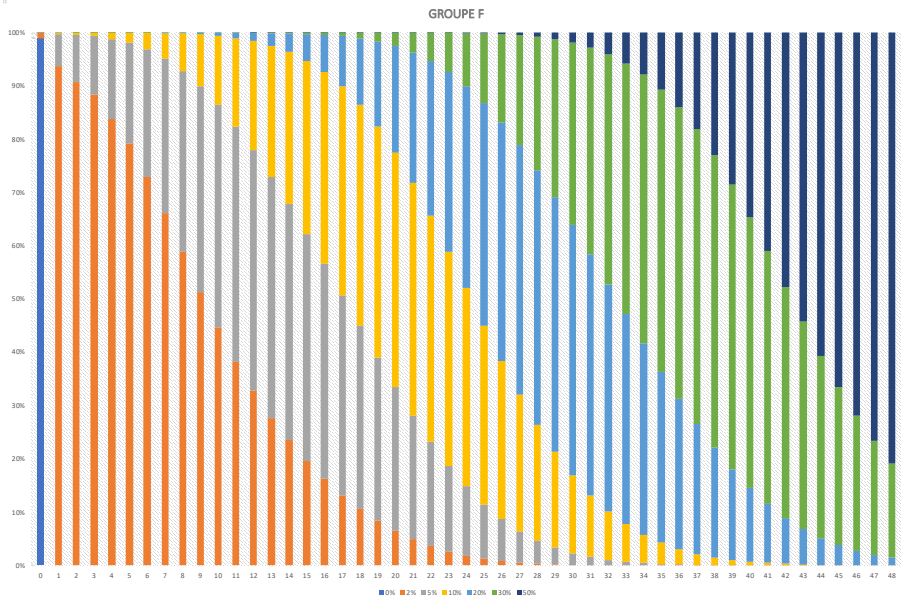
Groupe C



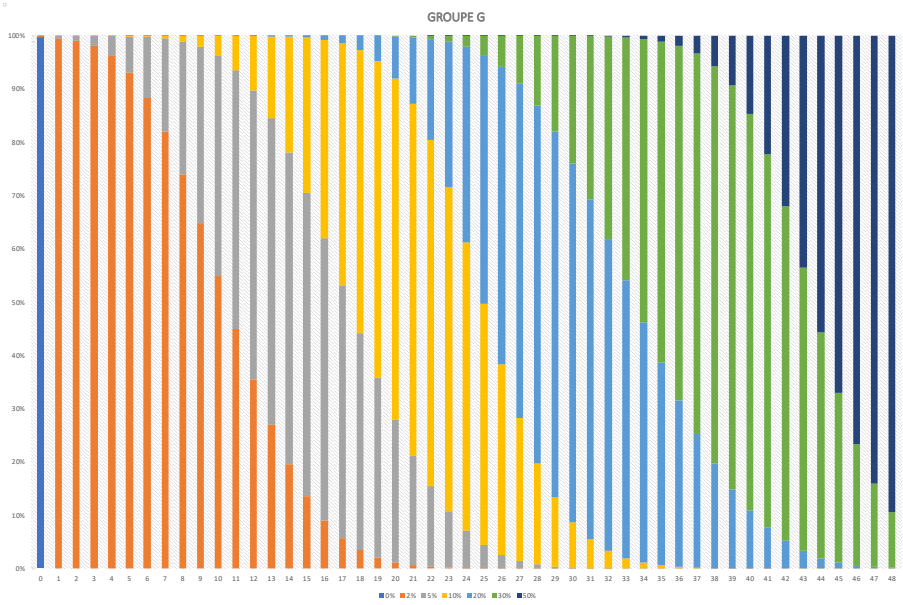
Groupe D



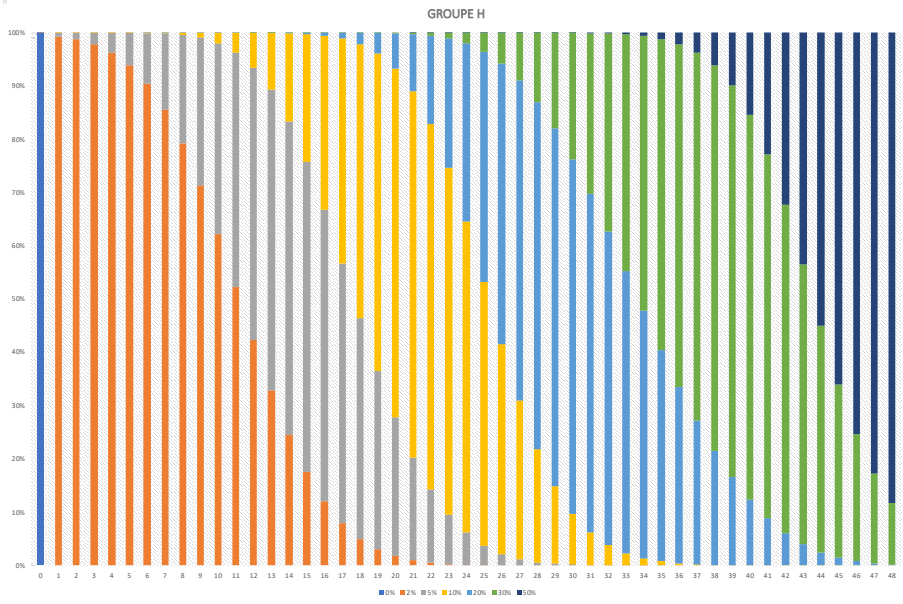
Groupe E



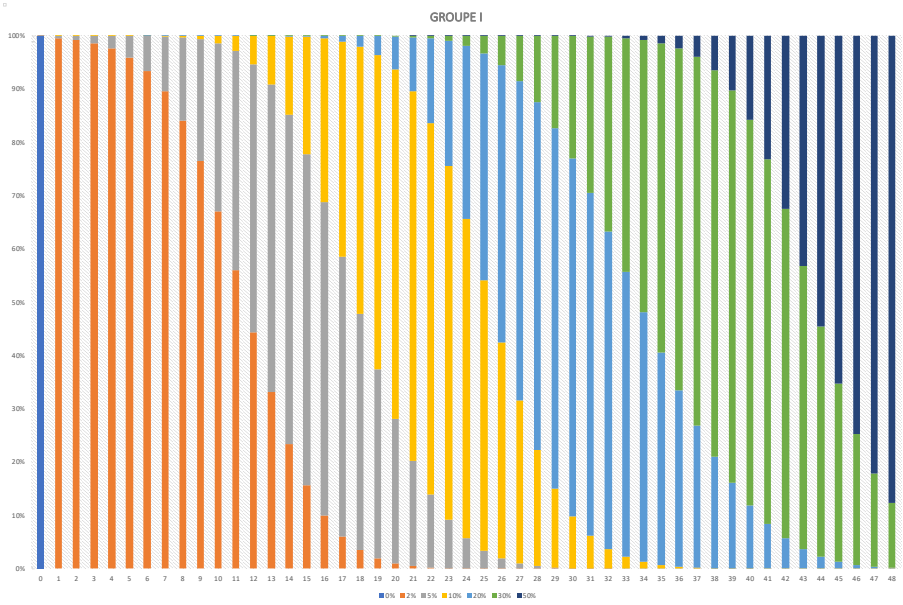
Groupe F



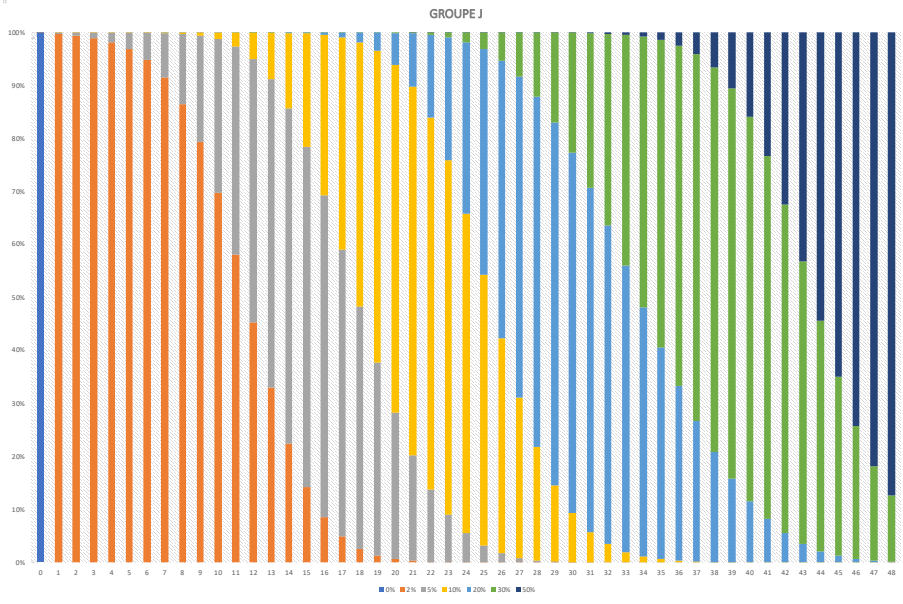
Groupe G



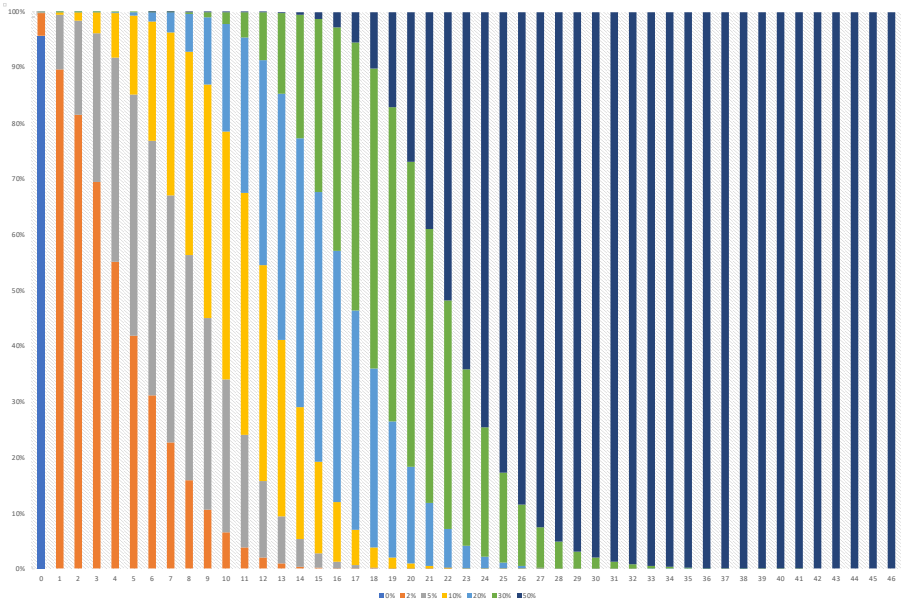
Groupe H



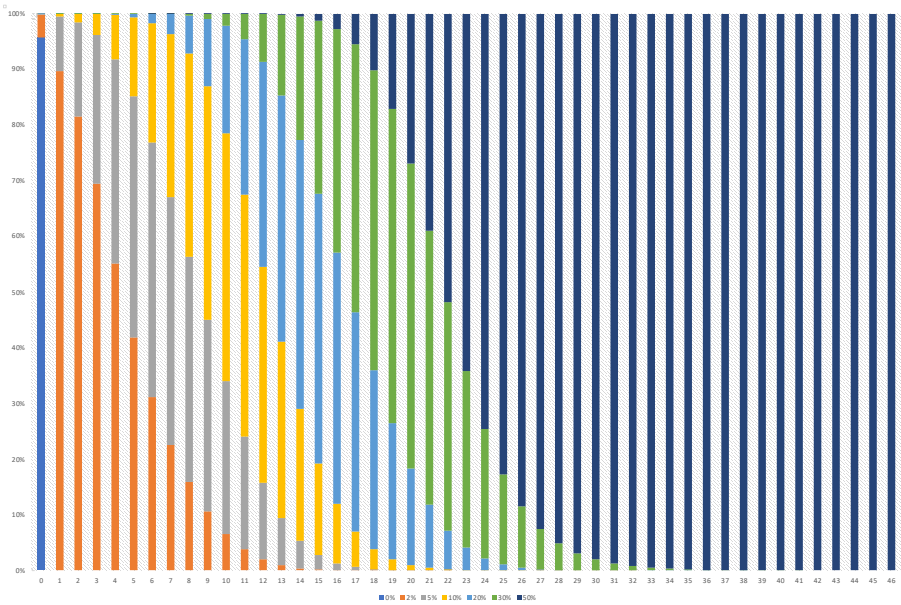
Groupe I



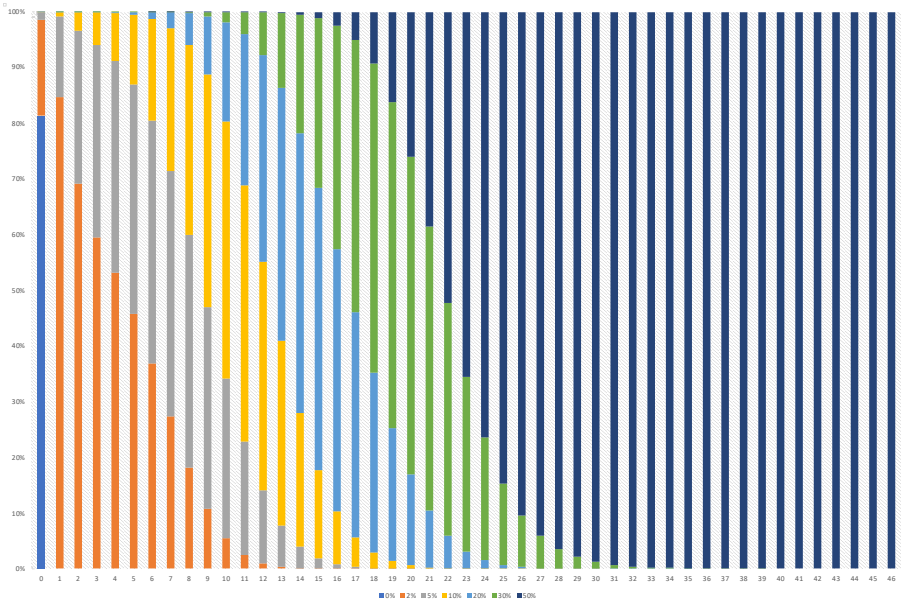
Groupe J



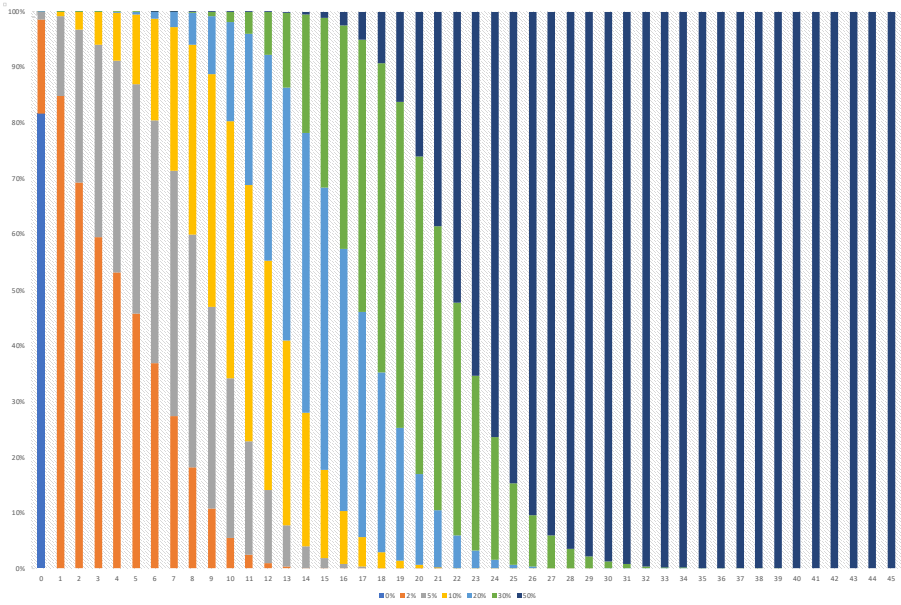
Groupe K



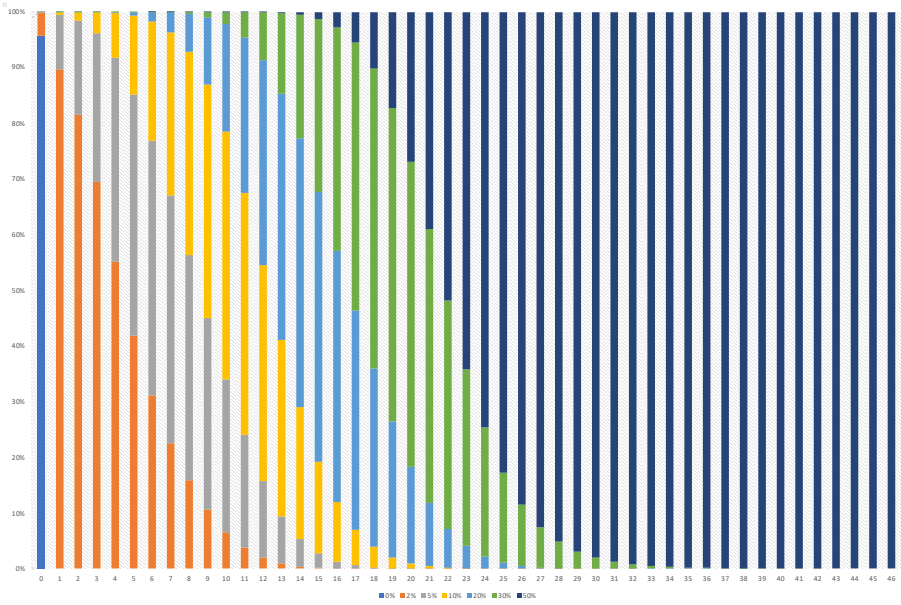
Groupe L



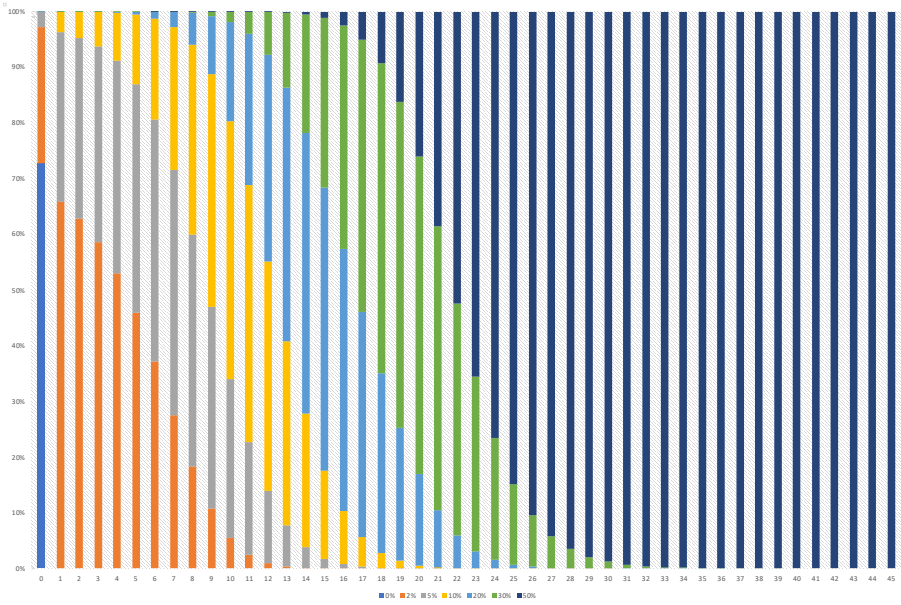
Groupe M



Groupe N

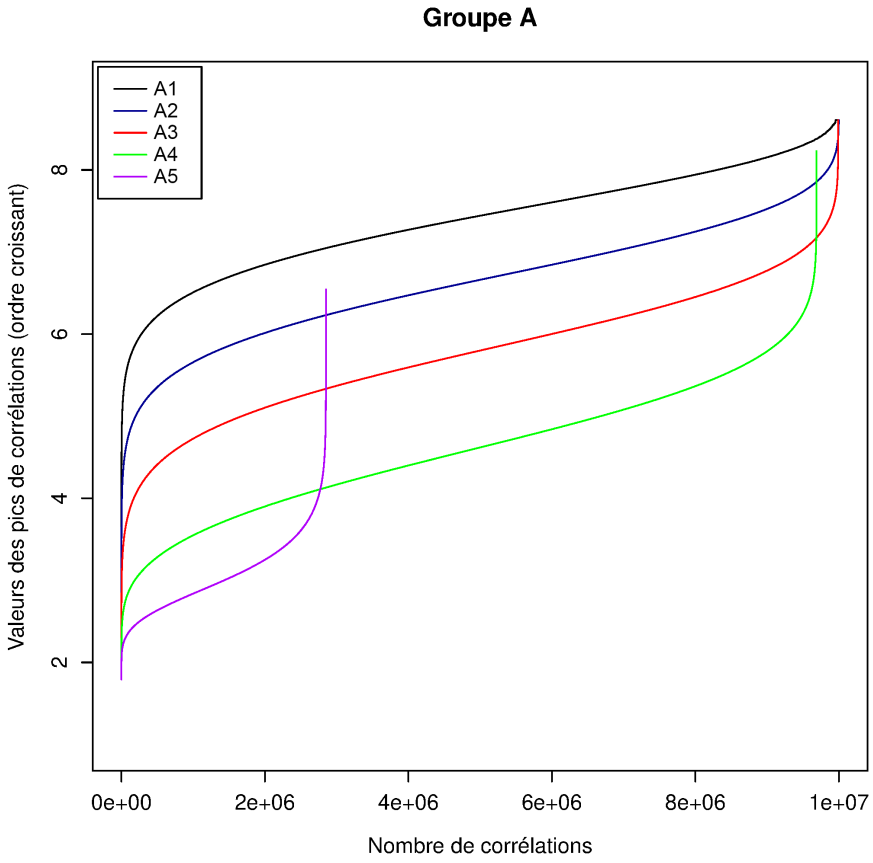


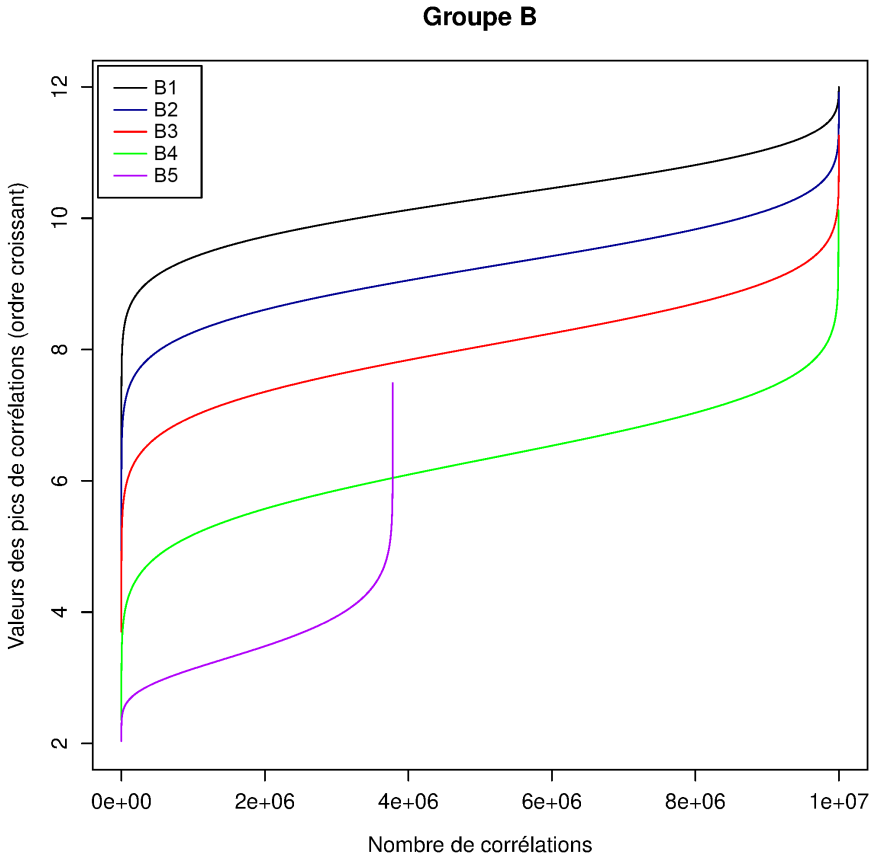
Groupe O

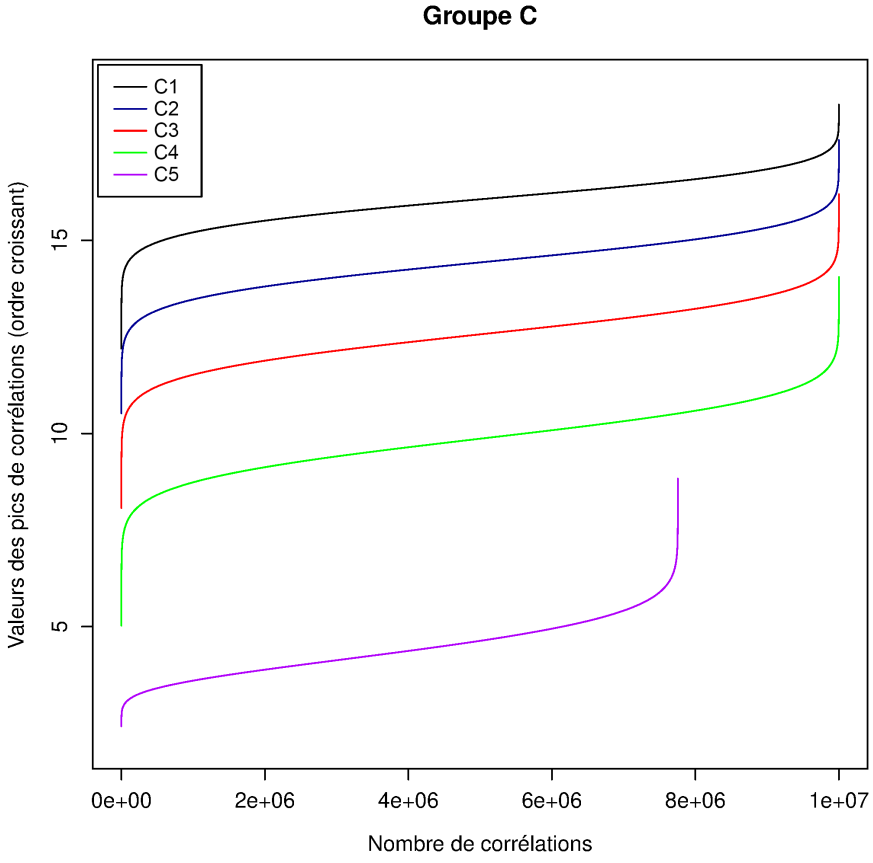


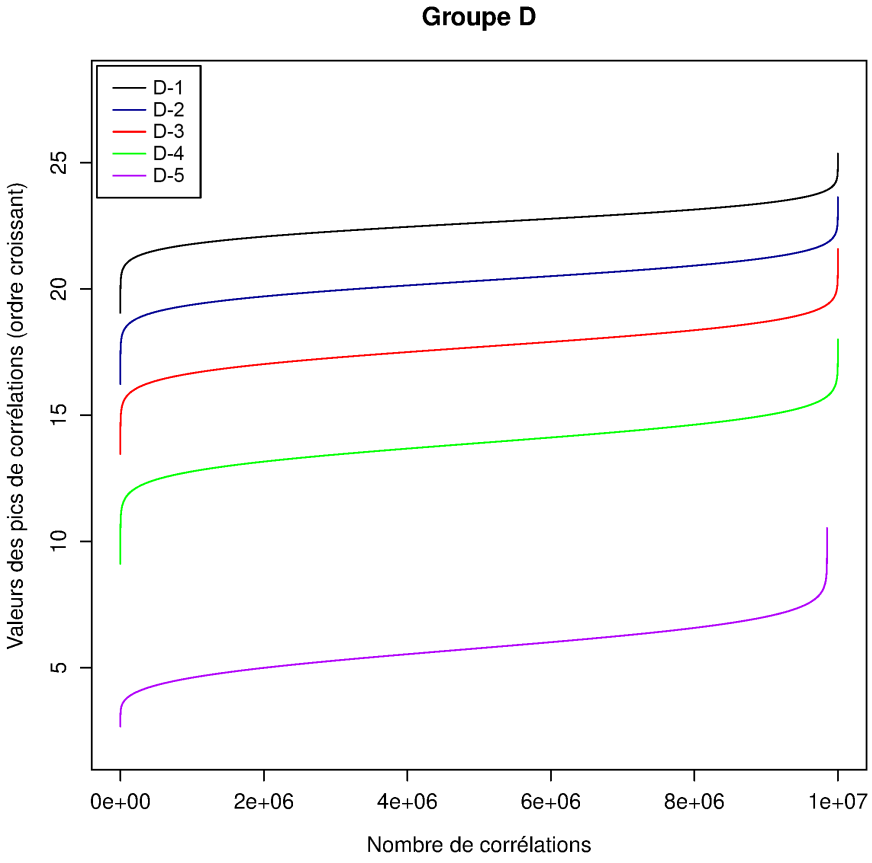
Groupe P

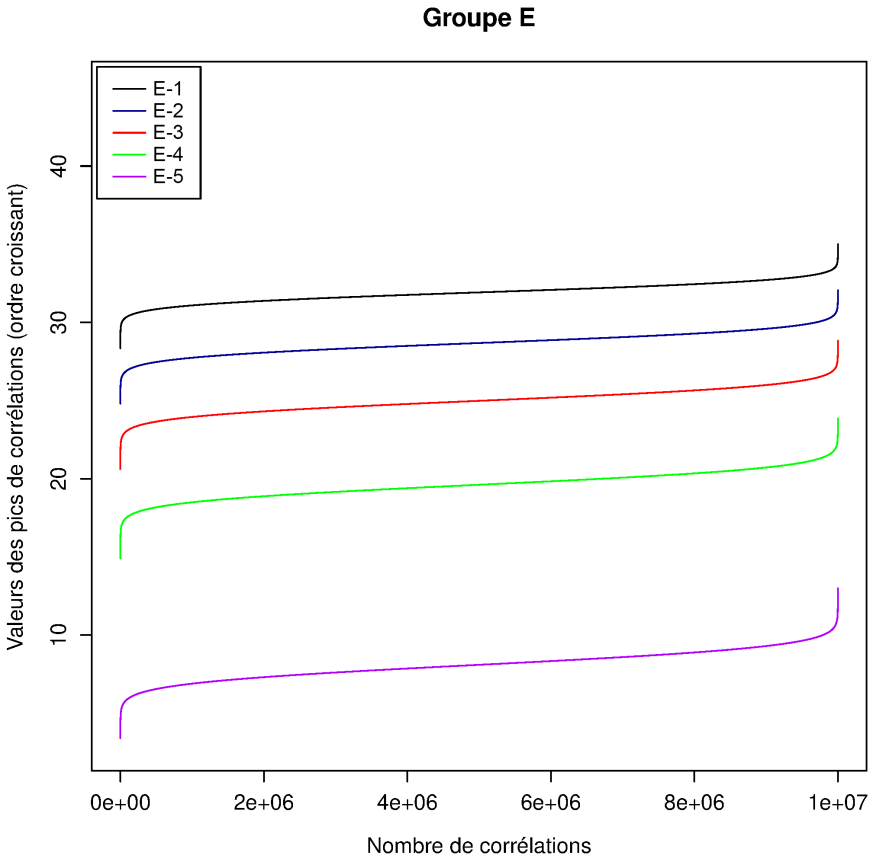
ANNEXE C

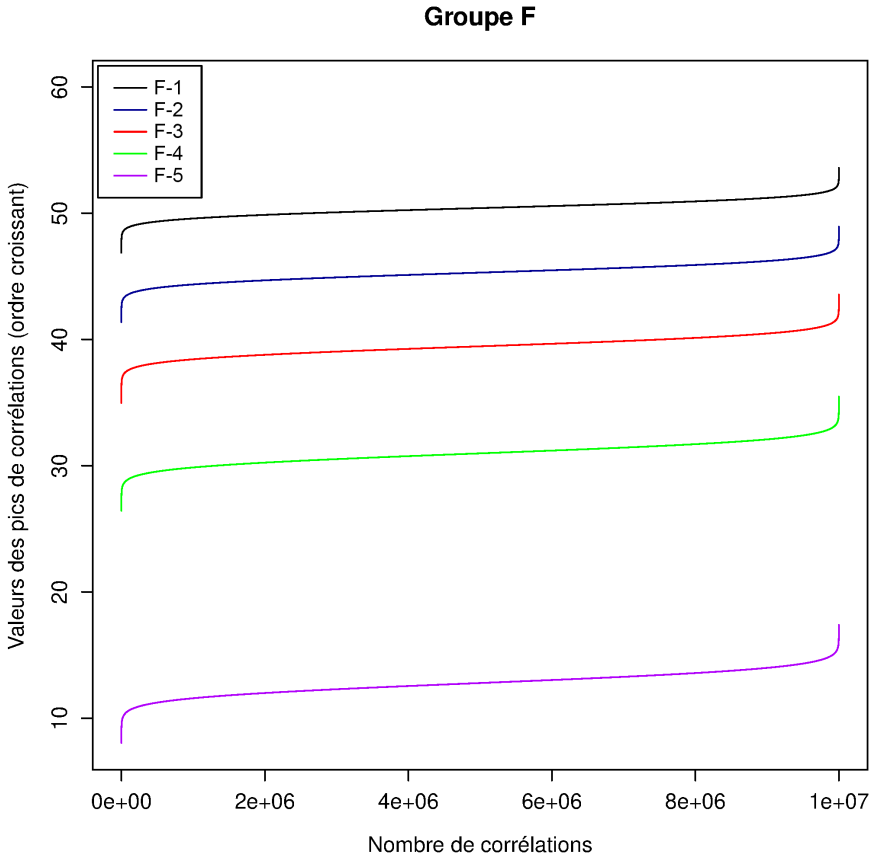

Groupe A
A1
A2
A3
A4
A5
Valeurs des pics de corrélations (ordre croissant)
2
4
6
8
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

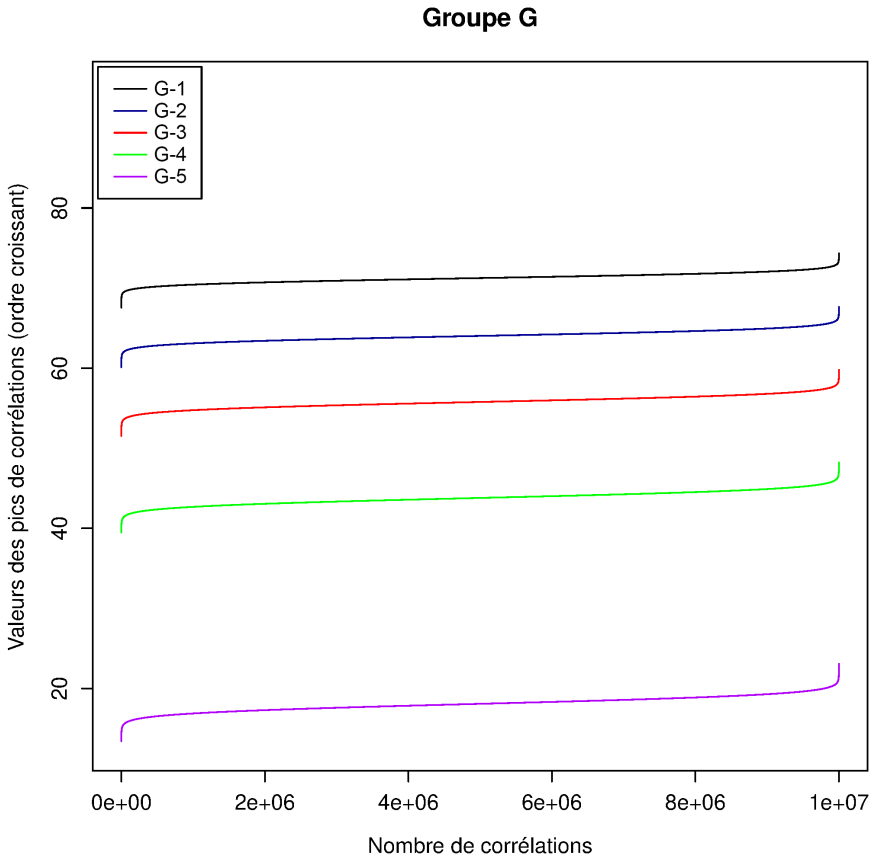

Groupe B
B1
B2
B3
B4
B5
Valeurs des pics de corrélations (ordre croissant)
2
4
6
8
10
12
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

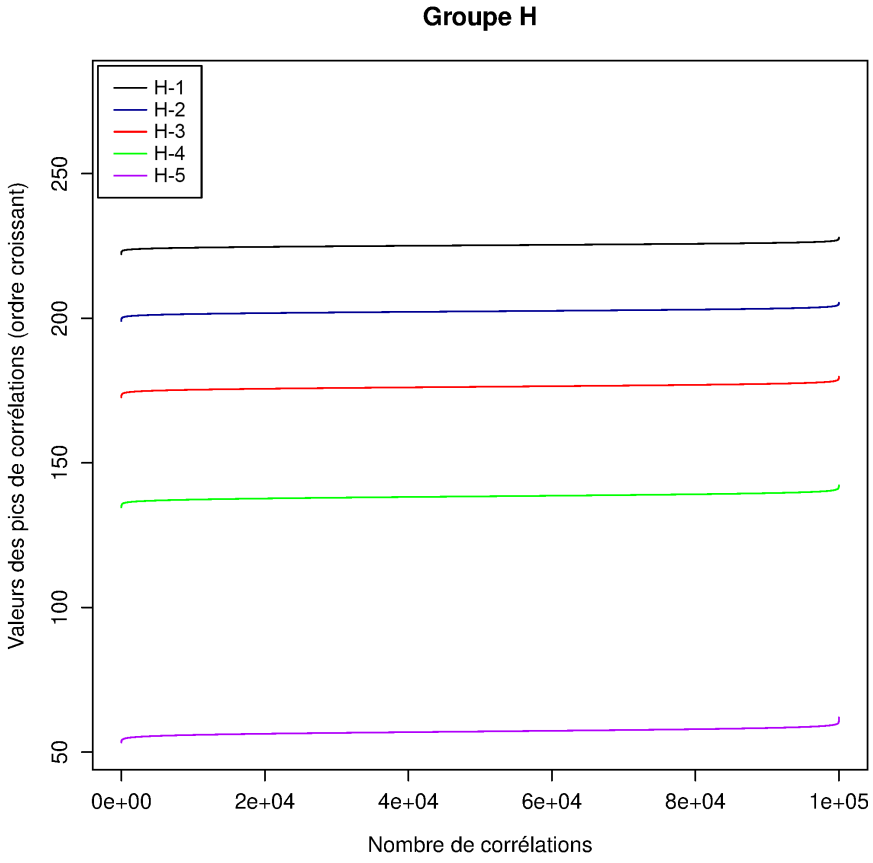

Groupe C
C1
C2
C3
C4
C5
Valeurs des pics de corrélations (ordre croissant)
5
10
15
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

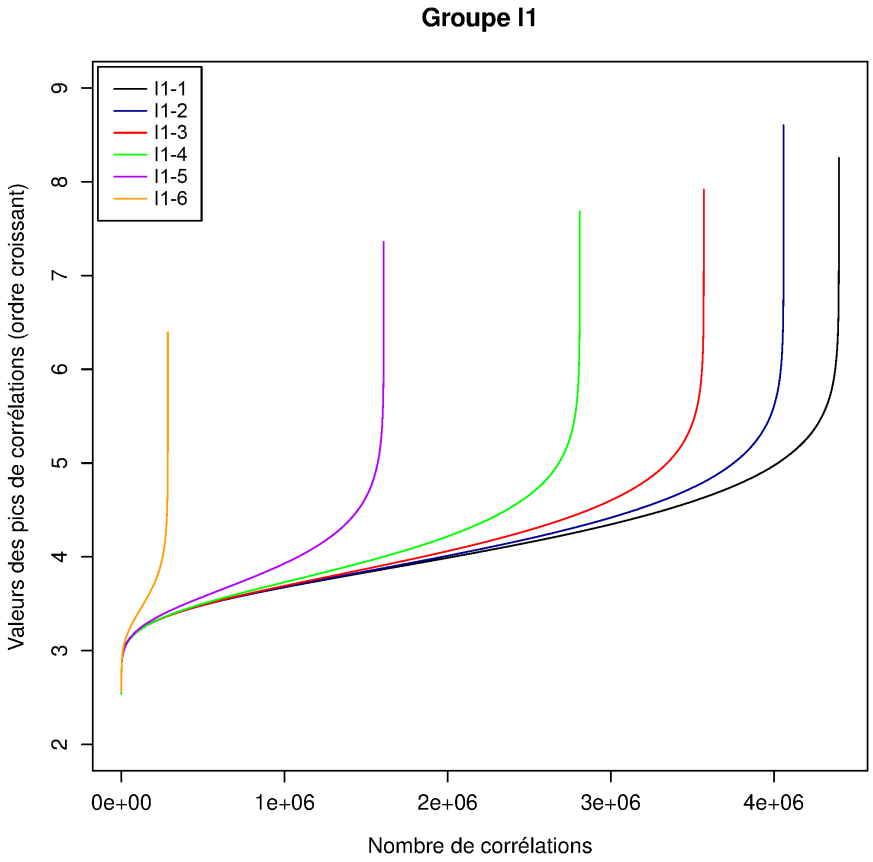

Groupe D
D-1
D-2
D-3
D-4
D-5
Valeurs des pics de corrélations (ordre croissant)
5
10
15
20
25
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

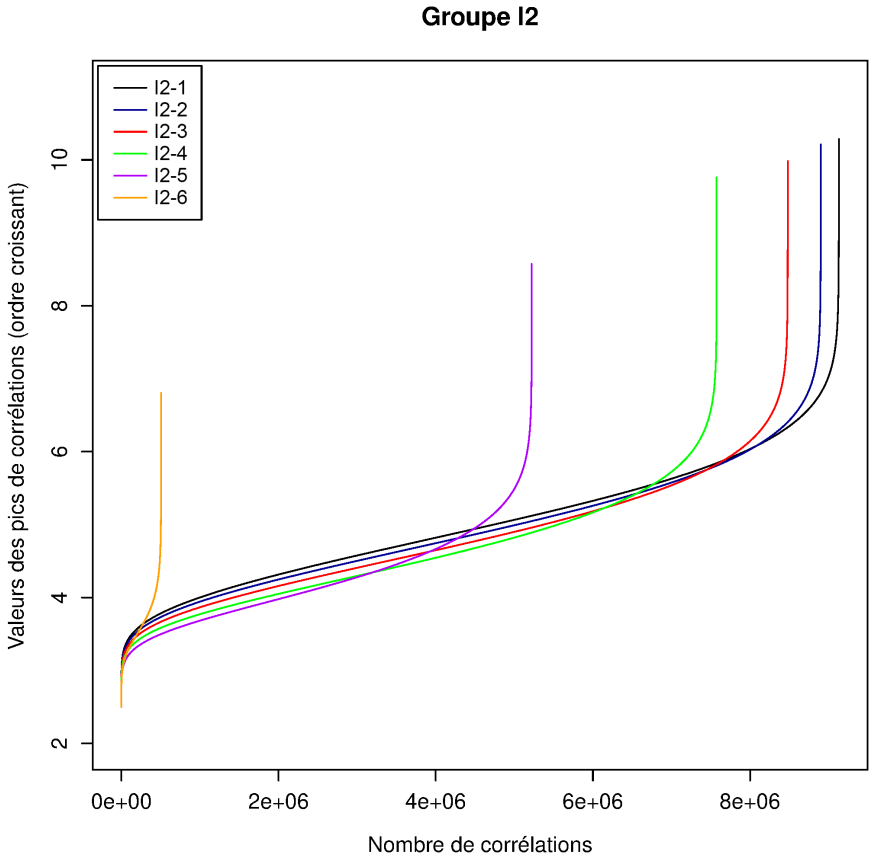

Groupe E
E-1
E-2
E-3
E-4
E-5
Valeurs des pics de corrélations (ordre croissant)
10
20
30
40
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

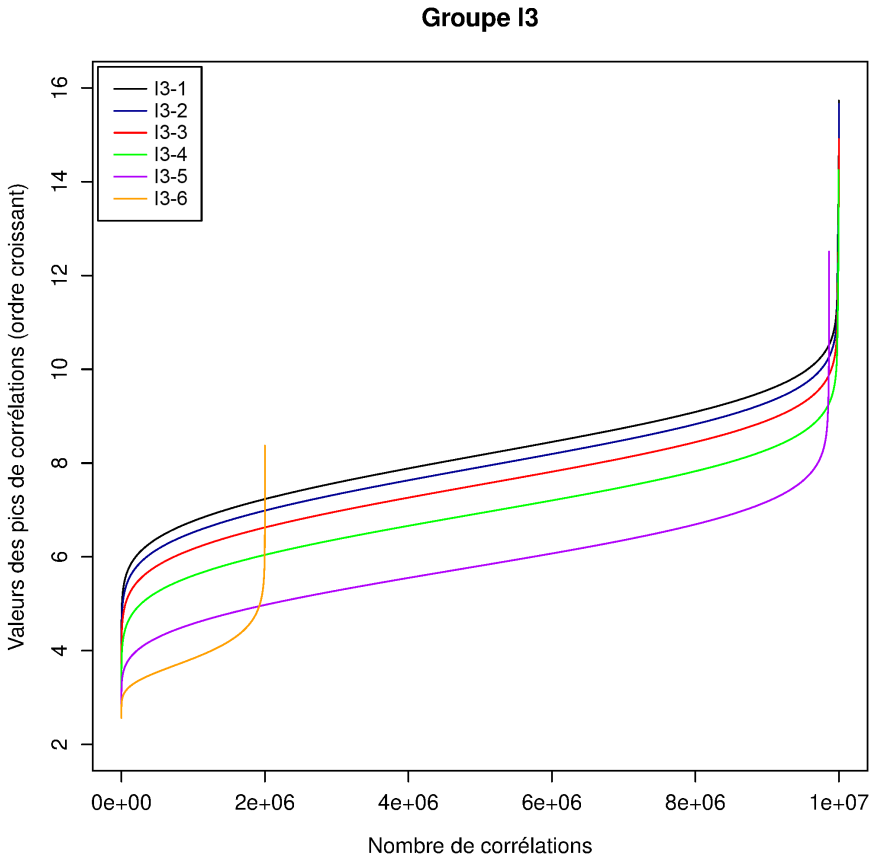

Groupe F
F-1
F-2
F-3
F-4
F-5
Valeurs des pics de corrélations (ordre croissant)
10
20
30
40
50
60
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

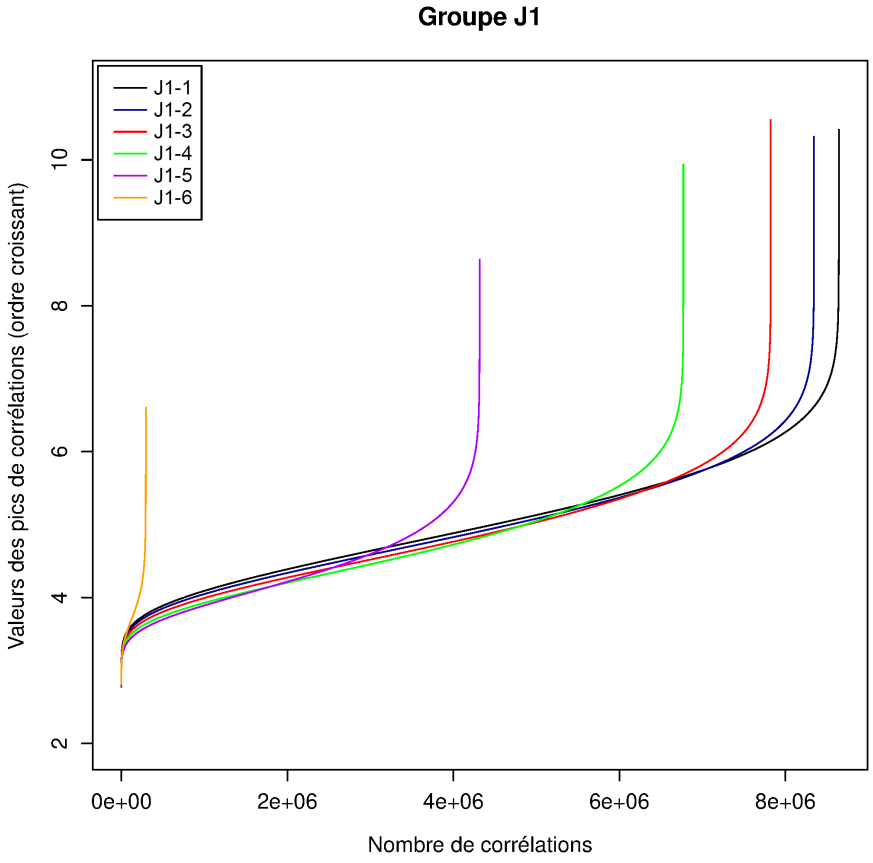

Groupe G
G-1
G-2
G-3
G-4
G-5
Valeurs des pics de corrélations (ordre croissant)
20
40
60
80
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

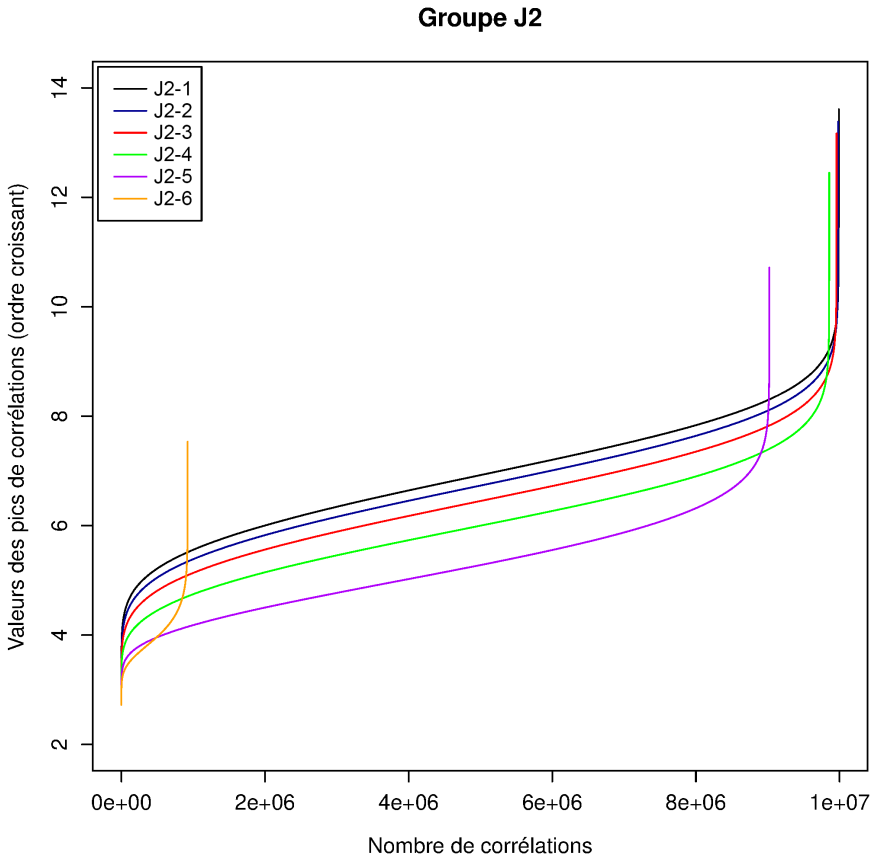

Groupe H
H-1
H-2
H-3
H-4
H-5
Valeurs des pics de corrélations (ordre croissant)
50
100
150
200
250
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05
Nombre de corrélations

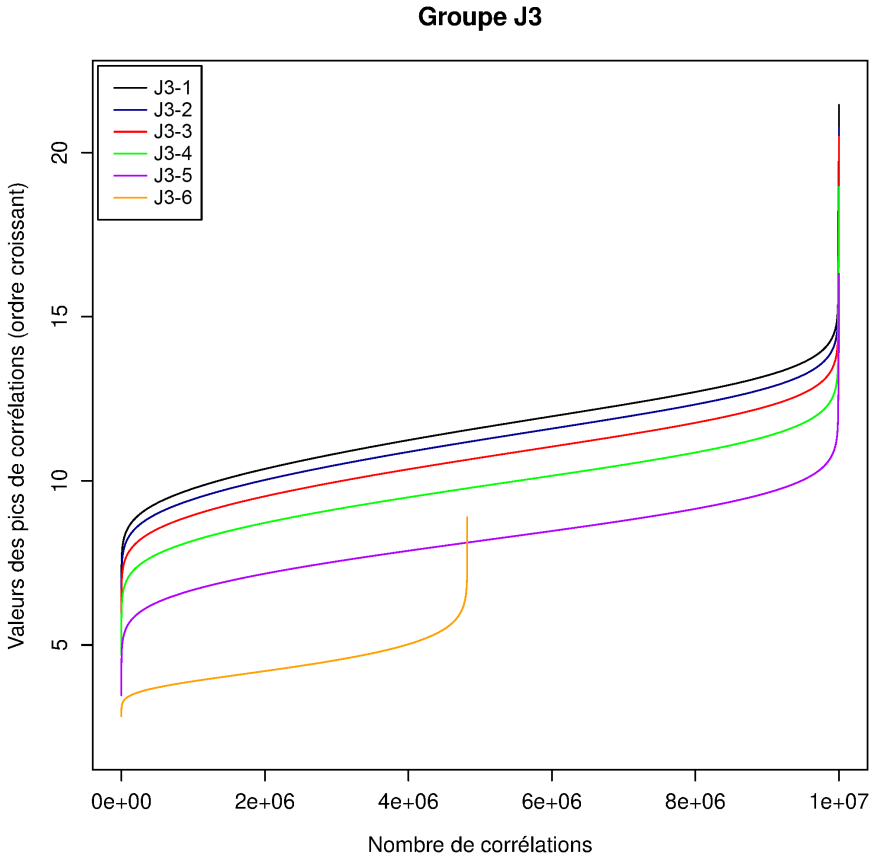

Groupe I1
I1-1
I1-2
I1-3
I1-4
I1-5
I1-6
Valeurs des pics de corrélations (ordre croissant)
2
3
4
5
6
7
8
9
0e+00
1e+06
2e+06
3e+06
4e+06
Nombre de corrélations

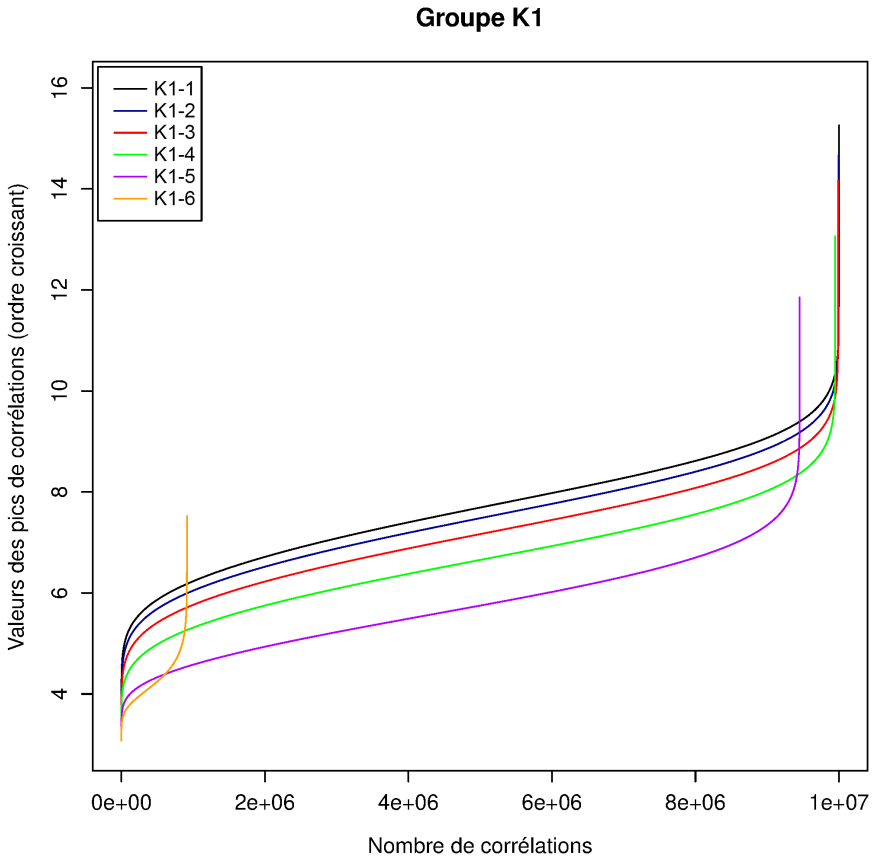

Groupe I2
I2-1
I2-2
I2-3
I2-4
I2-5
I2-6
Valeurs des pics de corrélations (ordre croissant)
2
4
6
8
10
0e+00
2e+06
4e+06
6e+06
8e+06
Nombre de corrélations

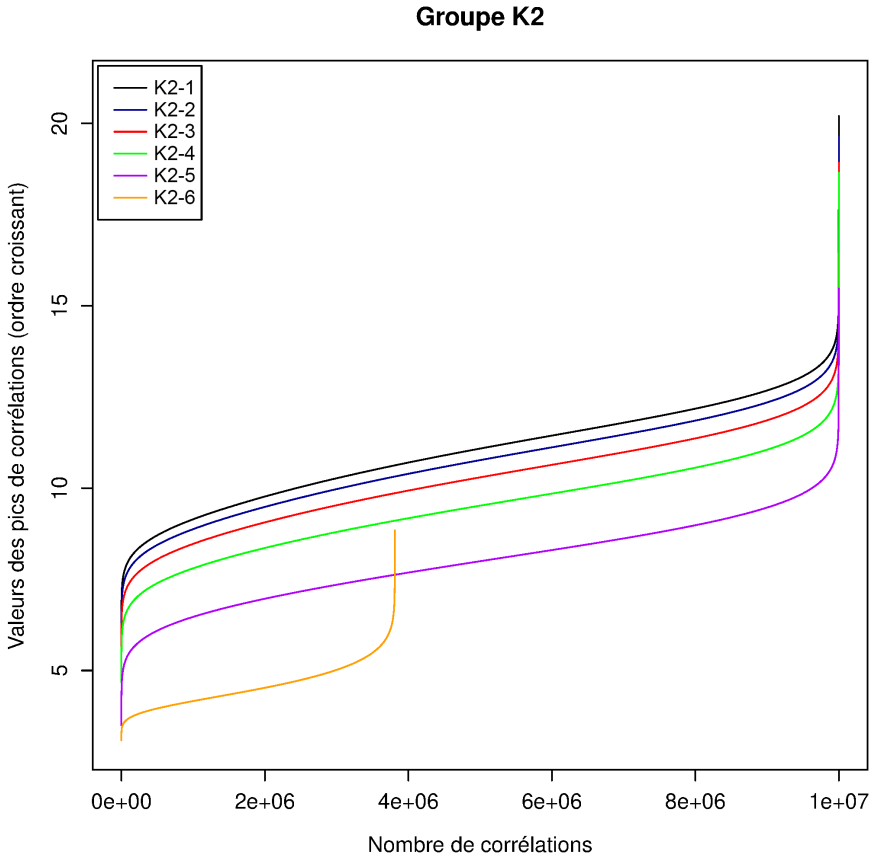

Groupe I3
I3-1
I3-2
I3-3
I3-4
I3-5
I3-6
Valeurs des pics de corrélations (ordre croissant)
2
4
6
8
10
12
14
16
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

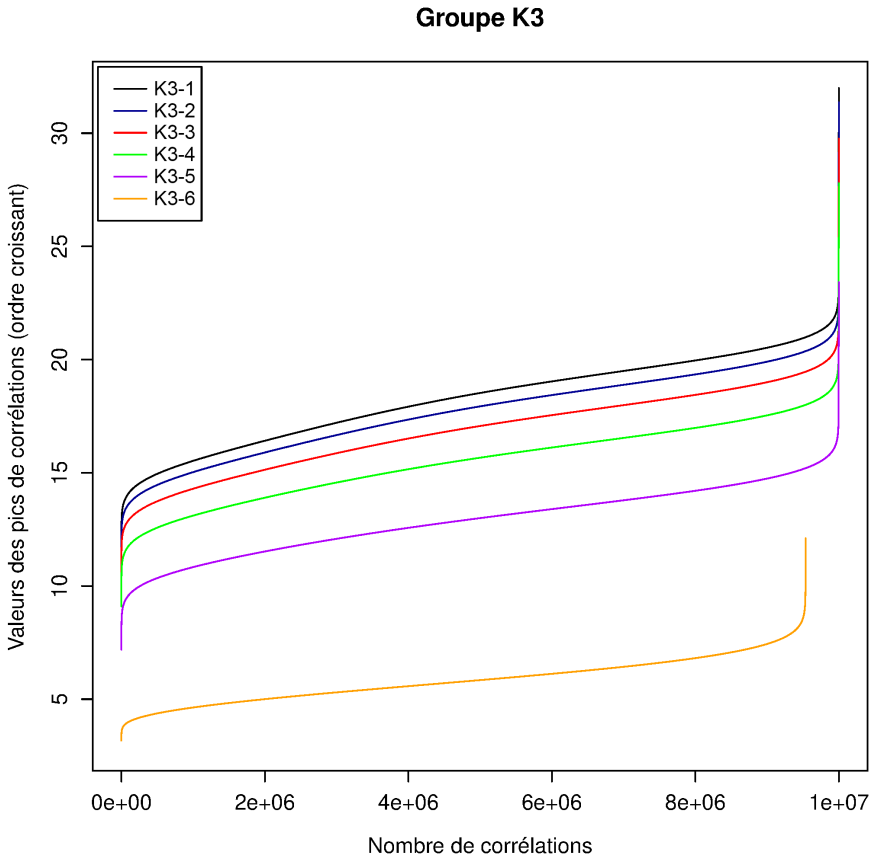

Groupe J1
J1-1
J1-2
J1-3
J1-4
J1-5
J1-6
Valeurs des pics de corrélations (ordre croissant)
2
4
6
8
10
0e+00
2e+06
4e+06
6e+06
8e+06
Nombre de corrélations

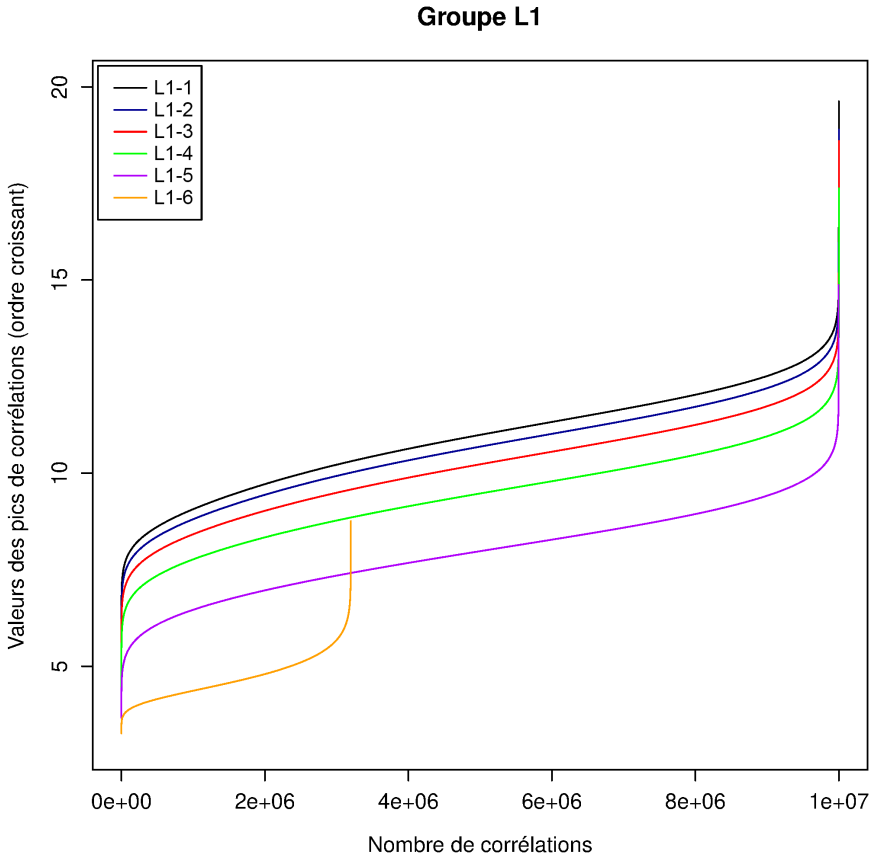

Groupe J2
J2-1
J2-2
J2-3
J2-4
J2-5
J2-6
Valeurs des pics de corrélations (ordre croissant)
2
4
6
8
10
12
14
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

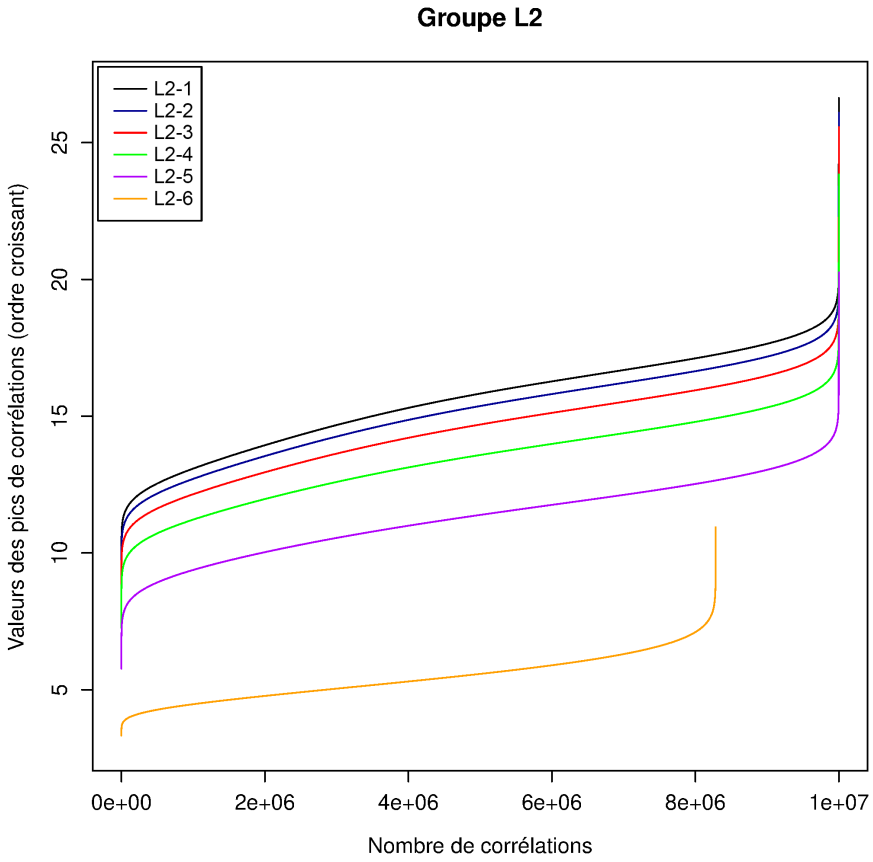

Groupe J3
J3-1
J3-2
J3-3
J3-4
J3-5
J3-6
Valeurs des pics de corrélations (ordre croissant)
5
10
15
20
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

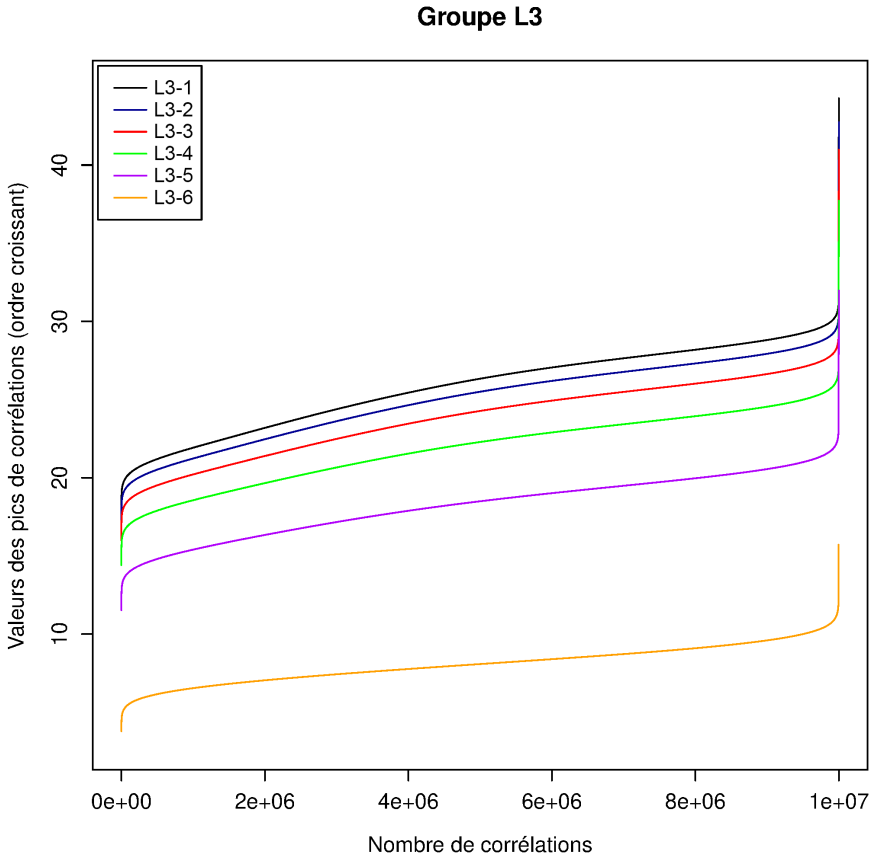

Groupe K1
K1-1
K1-2
K1-3
K1-4
K1-5
K1-6
Valeurs des pics de corrélations (ordre croissant)
4
6
8
10
12
14
16
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

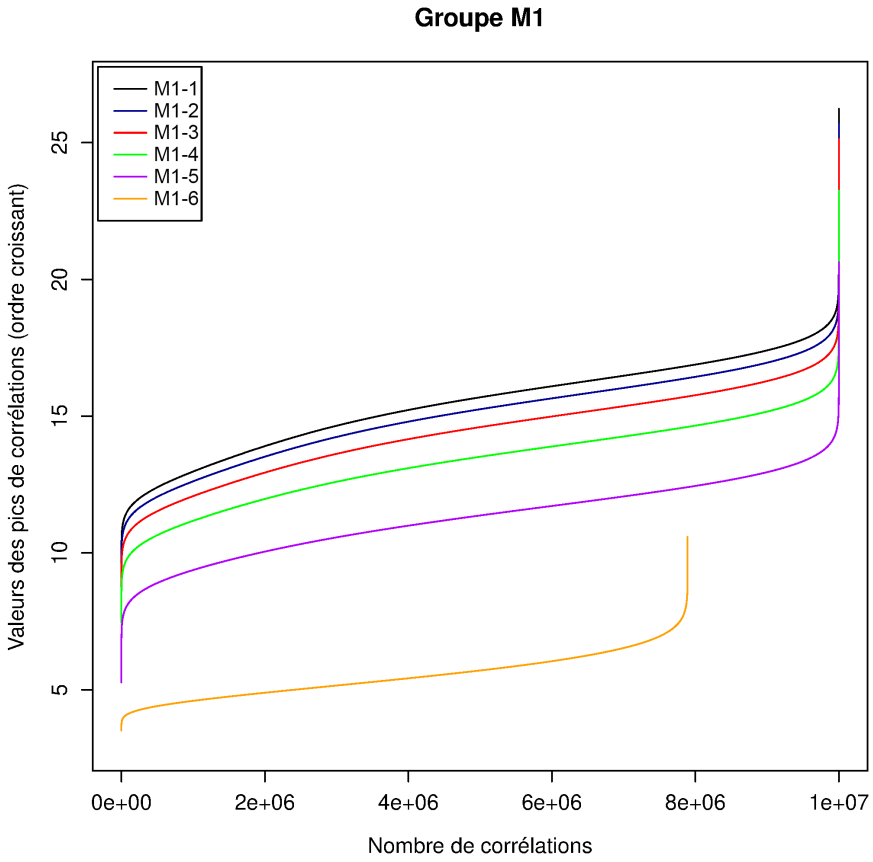

Groupe K2
K2-1
K2-2
K2-3
K2-4
K2-5
K2-6
Valeurs des pics de corrélations (ordre croissant)
5
10
15
20
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

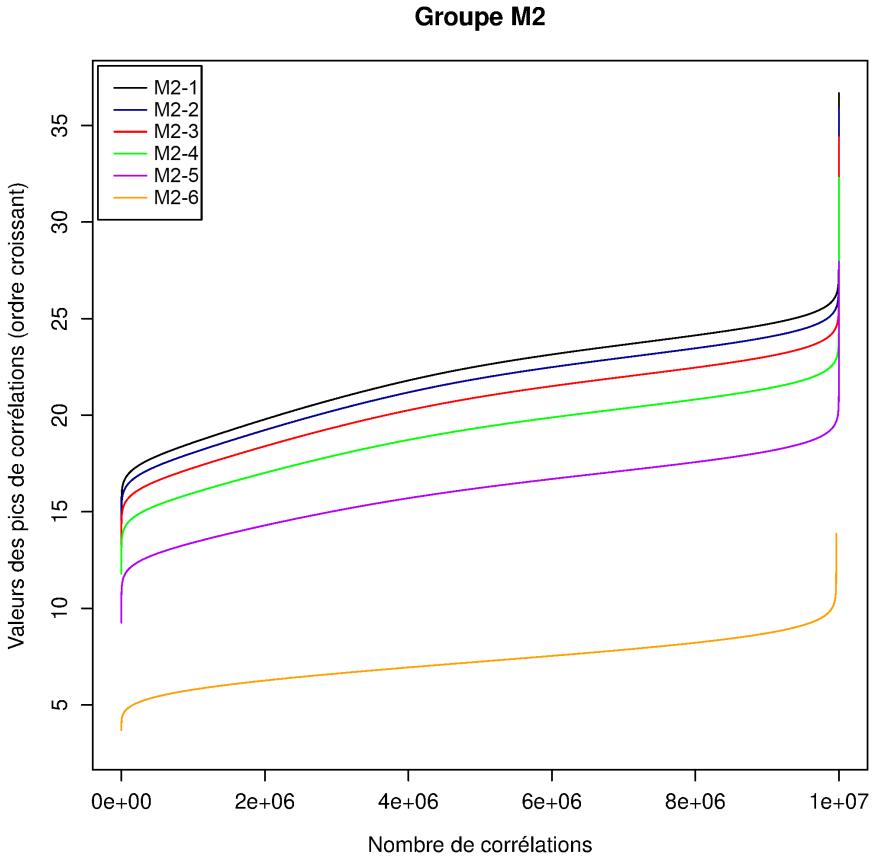

Groupe K3
K3-1
K3-2
K3-3
K3-4
K3-5
K3-6
Valeurs des pics de corrélations (ordre croissant)
5
10
15
20
25
30
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

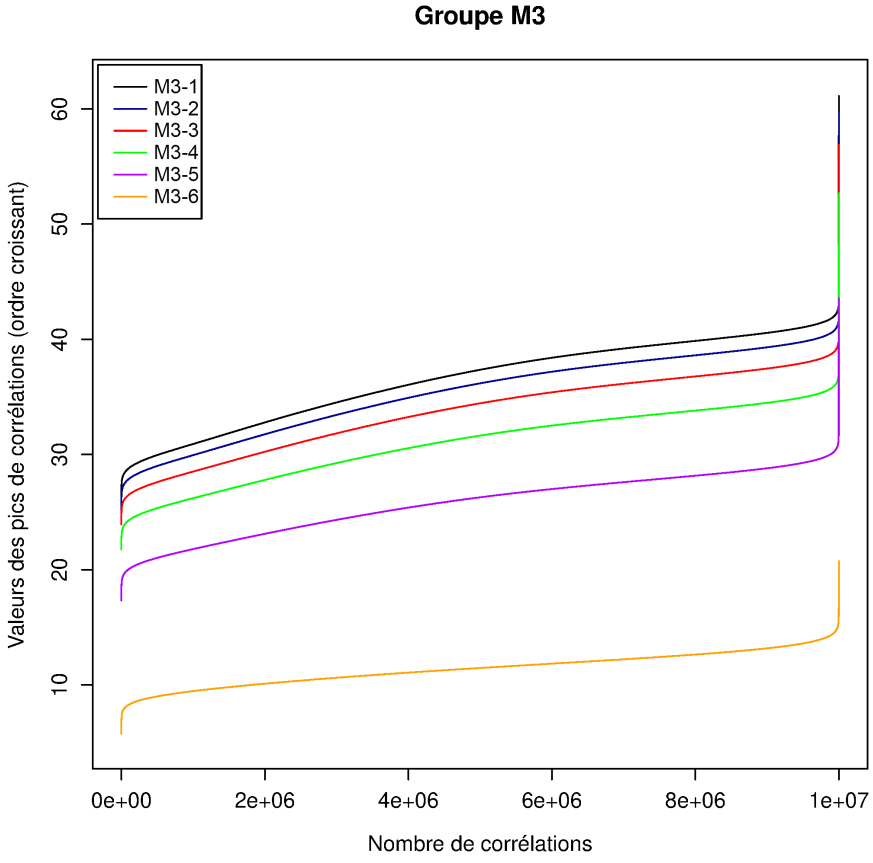

Groupe L1
L1-1
L1-2
L1-3
L1-4
L1-5
L1-6
Valeurs des pics de corrélations (ordre croissant)
5
10
15
20
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

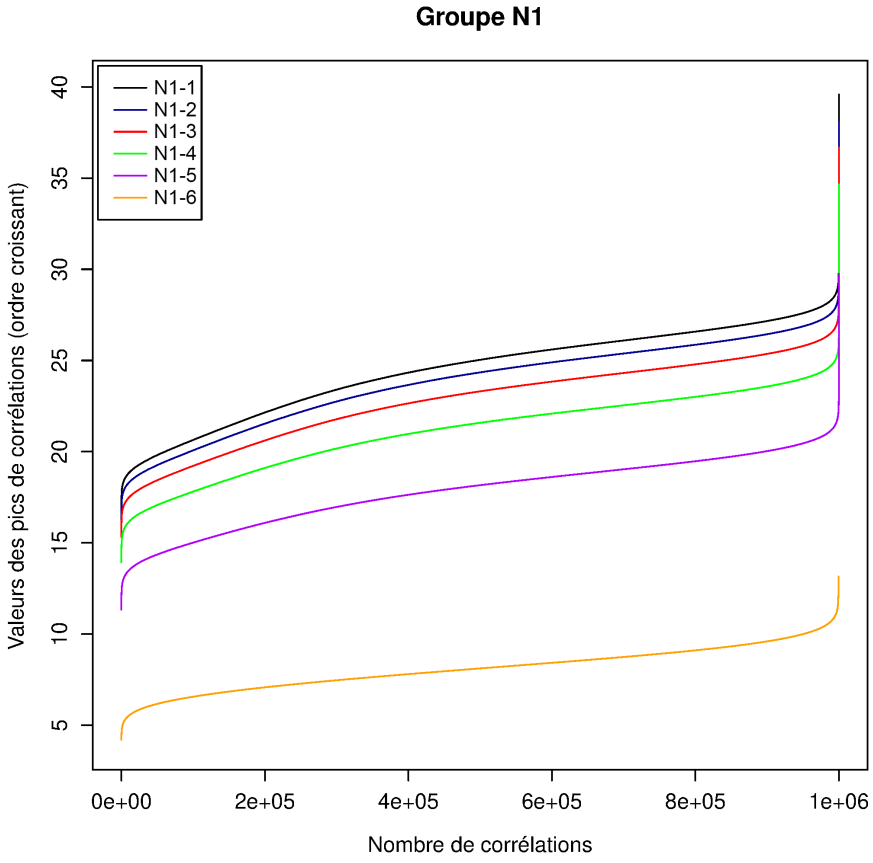

Groupe L2
L2-1
L2-2
L2-3
L2-4
L2-5
L2-6
Valeurs des pics de corrélations (ordre croissant)
5
10
15
20
25
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

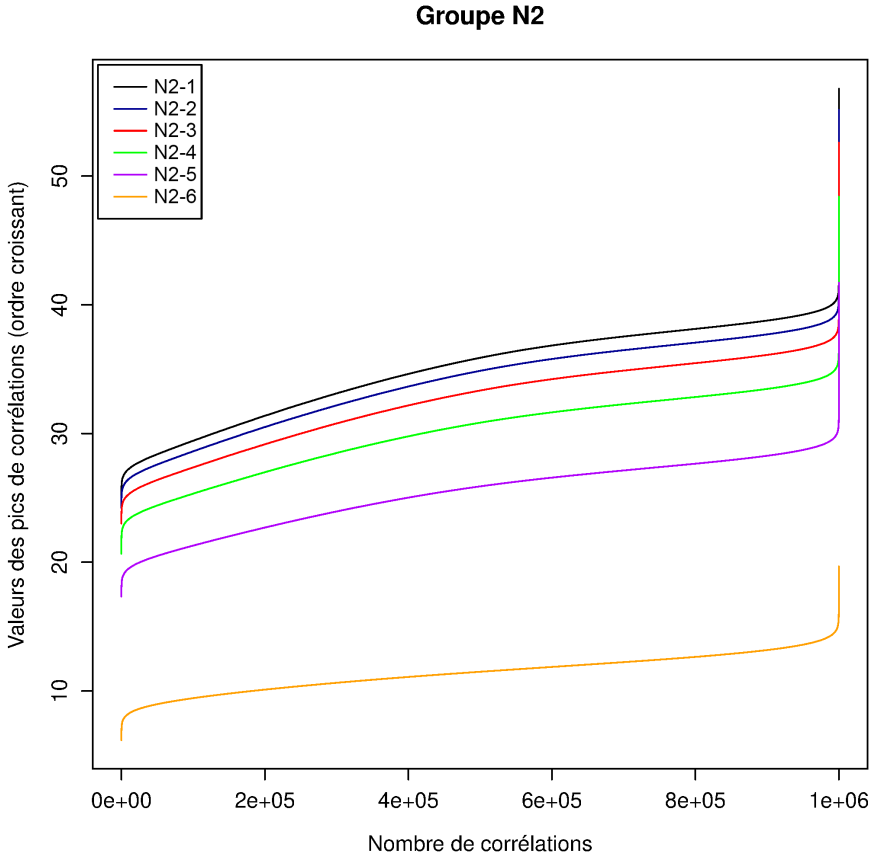

Groupe L3
L3-1
L3-2
L3-3
L3-4
L3-5
L3-6
Valeurs des pics de corrélations (ordre croissant)
10
20
30
40
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations

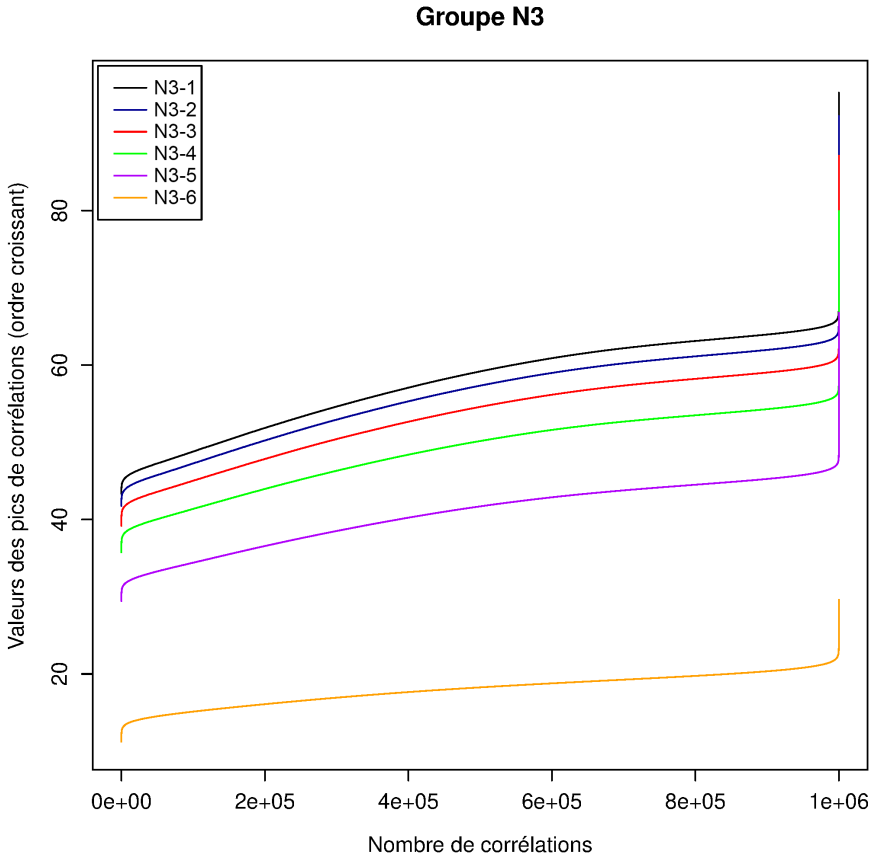

Groupe M1
M1-1
M1-2
M1-3
M1-4
M1-5
M1-6
Valeurs des pics de corrélations (ordre croissant)
5
10
15
20
25
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations


Groupe M2
M2-1
M2-2
M2-3
M2-4
M2-5
M2-6
Valeurs des pics de corrélations (ordre croissant)
5
10
15
20
25
30
35
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations


Groupe M3
M3-1
M3-2
M3-3
M3-4
M3-5
M3-6
Valeurs des pics de corrélations (ordre croissant)
10
20
30
40
50
60
0e+00
2e+06
4e+06
6e+06
8e+06
1e+07
Nombre de corrélations


Groupe N1
N1-1
N1-2
N1-3
N1-4
N1-5
N1-6
Valeurs des pics de corrélations (ordre croissant)
5
10
15
20
25
30
35
40
0e+00
2e+05
4e+05
6e+05
8e+05
1e+06
Nombre de corrélations


Groupe N2
N2-1
N2-2
N2-3
N2-4
N2-5
N2-6
Valeurs des pics de corrélations (ordre croissant)
10
20
30
40
50
0e+00
2e+05
4e+05
6e+05
8e+05
1e+06
Nombre de corrélations


Groupe N3
N3-1
N3-2
N3-3
N3-4
N3-5
N3-6
Valeurs des pics de corrélations (ordre croissant)
20
40
60
80
0e+00
2e+05
4e+05
6e+05
8e+05
1e+06
Nombre de corrélations