Université Paris Diderot - Paris 7 2016/2017

Master 2 Biologie-Informatique-Bioinformatique

Développement d'une méthode de classification pour les séquences répétées centromériques de primates

Florence Jornod

Sous la direction de :

Loïc Ponger

Structure et Instabilité des Génomes

MNHN - CNRS UMR 7196 / INSERM U1154 - Sorbonne Universités

Remerciements

Je tiens tout d'abord à remercier énormément Loïc Ponger, responsable de mon stage, pour son encadrement, ses conseils, ses relectures et son aide.

Je tiens également à remercier Christophe Escudé pour tous ses précieux conseils et pour le temps passer à relire mon rapport.

Je souhaite aussi remercier Evelyne Duvernois, pour ses conseils tant au niveau professionnel que personnel, pour les discussions et pour les pauses ménagerie.

Je remercie aussi chaleureusement tout le laboratoire pour son accueil chaleureux et pour les, nombreuses, pauses café très sympathiques.

Je souhaite également remercier ici Catherine Etchebest et Jean-Christophe Gelly, et l'ensemble de l'équipe pédagogique, pour cette année de master 2.

Je voudrais finir en remerciant la promotion M2BI 2016-2017 et plus particulièrement Athénais, Charlotte, Jaysen et Julie qui ont été d'un grand soutien. Merci également à Julien, son soutien a été d'une grande aide.

Table des matières

1 Introduction

1.1 Les séquences centromériques

Le centromère est une région chromosomique qui permet l'attachement du fuseau mitotique et la ségrégation des chromosomes durant la division cellulaire chez les eucaryotes [1]. Il est le site d'assemblage du kinétochore, un complexe protéique, permet l'attachement des microtubules aux chromosomes [2]. La chromatine centromérique se caractérise par la présence de CENP-A, un variant de l'histone H3 canonique très conservé au cours de l'évolution. Cette protéine fixe la position du kinétochore via un mécanisme encore mal connu [3].

Chez la plupart des eucaryotes, l'ADN centromérique est composé de séquences répétées en tandem nommées séquences satellites. Alors que la fonction du centromère et les protéines impliquées sont relativement bien conservées, les séquences et l'organisation de celles-ci peuvent varier entre les espèces [4]. Ces séquences peuvent représenter environ 5% du génome. La taille des unités de répétitions peut varier entre 7pb et 3,2kb [5] mais on trouve très souvent des séquences de 145-180kb.

1.2 L'ADN α-satellites

Chez les primates, les séquences centromériques se caractérisent par des séquences repétées en tandem, les monomères, riches en AT et long d'environ 171 pb. Elles sont appelées séquences α-satellites [6]. Ces séquences ont été mises en évidence pour la première fois chez *Chlorocebus aethiops* dans les années 1970 [7] et des homologues ont été retrouvés chez d'autres espèces de primate [8]. mais ces séquences ont été essentiellement étudiées chez l'homme.

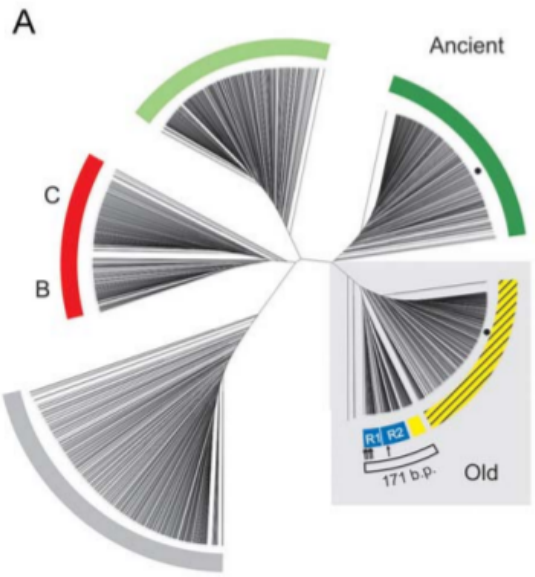

FIGURE 1 – **Arbre phylogénétique des séquences α-satellites du bras p du chromosome X**
L'arbre réunit 1431 monomères présent sur le contig nt011630 [9]

Les séquences ont un taux d'identité compris entre 60 et 100% [10]. Il est possible de distinguer des familles qui regroupent les séquences les plus similaires (figure 1). Ces familles résultent d'un même événement d'amplification. Il a été montré chez l'homme, qu'en plus d'avoir une proximité phylogénétique, les séquences d'une même famille se regroupaient aussi spatialement le long du chromosome [9]. Ces observations ont permis de proposer un modèle évolutif, avec des centromères en expansion, caractérisé par l'insertion des familles les plus récentes au cœur du centromère, dans la partie active et par le déplacement des anciennes familles dans les régions périphériques, les péricentromères. Les familles récentes présentent parfois une organisation en répétition d'ordre supérieur (High Order Repeat HOR), où un groupe de monomères appartenant à des familles différentes sont répétées en bloc les un derrière les autres (figure 2).

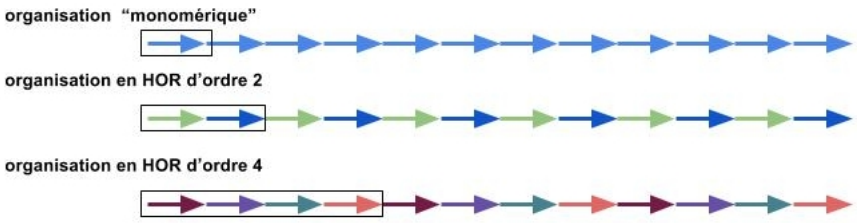

FIGURE 2 – **Schéma de l'organisation "monomérique" et en HOR des α-satellites**

Le rôle des séquences α satellite dans la fonction du centromère est très mal connu. Une seule protéine, CENP-B, est connue pour reconnaître spécifiquement un motif d'environ 17 pb

(CENP-B box) présent dans certaines familles et retrouvé chez de très nombreuses espèces de primates. Pour certaines familles, le motif CENP-B est remplacé par un autre motif permettant de fixer une autre protéine très mal caractérisée nommée pJα (pJα box) [11].

1.3 Le sujet de stage

Chez d'autres espèces de primates, les informations sont beaucoup plus éparses. L'équipe d'accueil de mon stage, ARChE (ADN repété, Chromatine, Évolution) a récemment développé une approche de séquençage à haut débit ciblée sur les séquences α-satellites chez deux espèces de cercopithèques [12]. Ce travail a mis en évidence l'existence de différentes familles d'α satellites chez ces espèces, ainsi que des différences entre les espèces qui ont été exploitées pour comprendre les mécanismes évolutifs de ces séquences. La seule autre étude publiée exploitant un grand nombre de séquences satellites de primates a été réalisé chez le gorille [13]. Cette études a tiré profit de l'existence de séquences α satellites de gorille dans les bases de données sur des fragments séquencés relativement long.

Les méthodes basées sur l'alignement et la phylogénie sont très limitées pour étudier ces séquences, en effet elles ne permettent pas de traiter de grands jeu de données, alors que plusieurs milliers de copies d'un satellites sont présentes dans un seul génome. De plus, ces méthodes ne sont pas objectives et ne permettent pas de comparer les résultats obtenus pour différentes espèces.

Pour répondre à ce problème, une méthode de classification automatique permettant de traiter des centaines de milliers de séquences sans a priori sur le nombre et la taille de ces familles a été mis au point dans l'équipe. Une première version de cette méthode de classification implémentée en R en 2016.

L'objectif de mon stage a été dans un premier temps d'améliorer la méthode avec pour objectifs la diminution du temps de calcul ainsi que la diminution de l'usage de la mémoire. Pour cela, une modification de l'algorithme était nécessaire. La méthode a aussi été ré-implémentée en Python. Dans un deuxième temps, le programme a été paramétré afin d'en ajuster la sensibilité du programme en l'étalonnant avec un jeu de données bien annoté. Une dernière étape a été

de caractériser les séquences α-satellites chez le Gorille, espèce décrite dans la littérature, afin de contrôler la qualité de notre classification.

2 Matériel et Méthodes

2.1 Séquences et monomères

Les séquences ont été obtenues fin 2015 à partir de génomes non assemblés disponibles sur GenBank. Les séquences non assemblées sont issues de séquençage générant des reads relativement long (> 171pb).
Les 1431 séquences utilisées pour la phase de paramétrisation ont été extraite du contig nt011630 du chromosome X humain annoté par Shepelev et al. (2009) [9]. Ce sont des séquences d'un contig assemblé réparties en 17 familles.
Les 37 000 séquences utilisées pour tester les améliorations en temps et en mémoire proviennent du génome humain assemblé (HGSC version 38).
Les 122104 séquences de gorille, utilisé pour la validation de la méthode sont issues du projet génome de gorille (version 5).

Un programme, développé par l'équipe, est utilisé afin d'isoler les séquences α-satellites à partir de séquences plus longues (chromosomes, reads longs,...). La phase, c'est-à-dire le nucléotide définissant le début du monomère, a été choisi arbitrairement. Les séquences contenant des N ainsi que les séquences très courtes (< 150 pb) et les séquences très longues (>210 pb) sont éliminées.

2.2 Méthode

La méthode implémentée permet, à partir d'un jeu données contenant des séquences au format fasta, de classer ces séquences en familles sur la base de leur similarité. Elle se base sur une classification hiérarchique dichotomique qui permet de séparer les séquences en fonction de la fréquence des k-mers qui les composent. Une fois la classification effectué, une validation des deux sous-groupes est faite. Si ils sont valides, ils sont alors redécoupé séparément , sinon le groupe initial est sauvegardé comme classe unique. L'algorithme est décrit sur la figure 3.

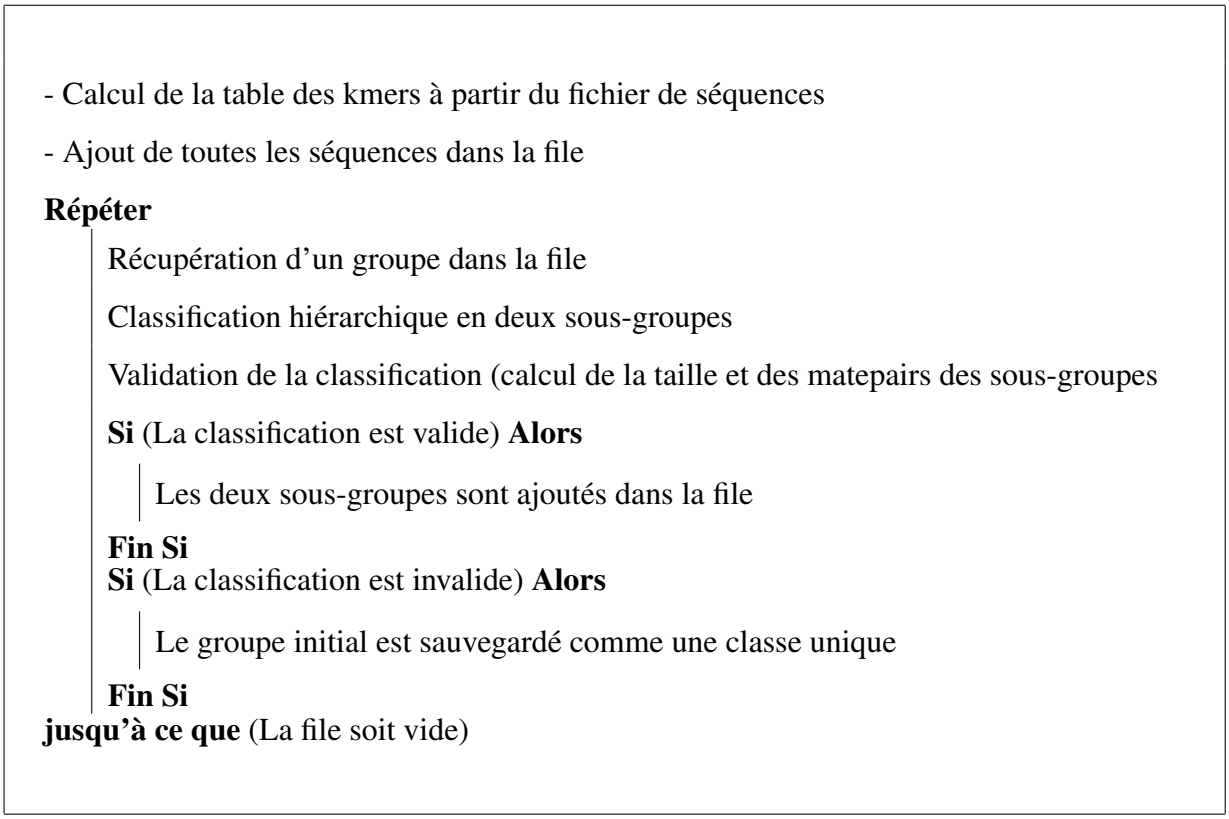

FIGURE 3 – **L'algorithme général montrant l'aspect itératif et dichotomique de la méthode**

2.2.1 Classification

La méthode exécute une boucle pour séparer les séquences en groupes de façon itérative et dichotomique tant que les nouveaux groupes formés sont divisibles. Chaque tour de boucle implique une ACP, une classification hierarchique sensus stricto, et une LDA si le jeu de données est très gros.

Calcul de la fréquence des k-mers La fréquence des k-mers est calculée pour chaque monomère. La valeur de k est un paramètre de l'algorithme dont la dépend de la nature des séquences. Le choix de la valeur de k n'est pas évident. De façon général, plus des séquences ont un pourcentage d'identité élevé plus il faudra une valeur de k élevée pour les différencier. Chez les cercopithèques, des 5-mers ont permis de décrire la diversité des séquences en optimisant l'usage de la mémoire (>150000 lignes x 1024 colonnes) [12]. Des k-mers de 5 nucléotides seront donc utilisés pour cette analyse. Cette table est calculée pour toutes les séquences en début de programme.

Analyse en Composantes Principales Une Analyse en Composante Principale (ACP) sur la table de fréquence des 5-kmers est effectuée afin de réduire les dimensions du jeu de données et d'obtenir des variables indépendantes (nécessaire à l'analyse linéaire discriminante (LDA)).

Classification hiérarchique sensus stricto Les distances euclidiennes sont calculées entre toutes les paires de séquences dans l'espace défini par les M premières composantes de l'ACP. A partir de ces distances, les séquences sont séparées en deux classes en utilisant une classification hiérarchique basées sur la méthode de Ward [14] qui consiste à former des classes de façon maximiser l'inertie interclasse.

Analyse Discriminante Linéaire L'étape de classification utilisant les distances entre toutes les séquences fait un usage important de la mémoire et ne permet pas en pratique de traiter de grands jeux de données. Une méthode d'apprentissage, l'Analyse Discriminante Linéaire (LDA), est utilisée sur un sous jeu de données formé par des séquences tirées aléatoirement. Puis le modèle construit sera appliqué sur toutes les séquences.

- Calcul d'une ACP sur la table de fréquences des kmers de toutes les séquences
- Calcul des distances euclidiennes des paires de séquences à partir des M premières composantes de l'ACP

Si (le nombre de séquences est supérieur à un seuil N) **Alors**
- Sélection aléatoire d'un échantillon de N séquences
- Classification hiérarchique de l'échantillon aléatoire en deux sous-groupes
- Apprentissage d'une LDA à partir des composantes de l'ACP et de la classification
- Affectation dans les deux sous-groupes de toutes les séquences utilisant la LDA

Sinon
- Classification hiérarchique en deux sous-groupes de toutes les séquences

Fin Si

FIGURE 4 – **L'algorithme détaillant la procédure de classification**

Ces différentes étapes séparent les données en deux sous-groupes. Une étape de validation permet de vérifier si les deux sous-groupes proposés ont un sens biologique.

2.2.2 Validation

La validation de la classification se fait à deux niveaux. Le programme vérifie que les deux sous-groupes obtenus sont bien distincts. Le fait que les deux sous-groupes correspondent à deux familles distinctes n'est pas trivial. En effet les familles anciennes pouvant inclure des séquences relativement divergentes (60 % de similarité) tandis que des séquences ayant plus de 90% de similarité peuvent appartenir à des familles différentes, correspondant à des amplifications récentes dans les centromères actifs. Il n'est donc pas possible d'utiliser un seuil de similarité pour la validation. Le matepair [15], un critère basé sur la connectivité intra/inter groupe est utilisé.

Matepair

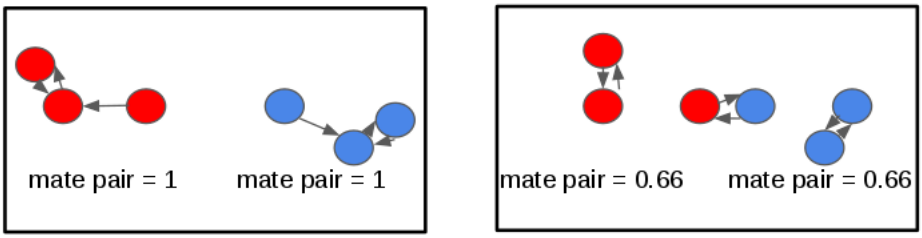

FIGURE 5 – **Représentation du matepair** Le matepair correspond à la proportion de monomères ayant son plus proche voisin dans le même groupe. Dans l'exemple de gauche, toutes les séquences "rouges" sont plus proches d'une séquence rouge que d'une séquence bleue et réciproquement. Les matepairs sont de 1. Par contre, dans l'exemple de droite, une séquence rouge est plus proche d'une bleue. Les matepairs sont de 0.66.

La classification proposée est validée en utilisant un critère externe, le matepair (voir figure 5). Il correspond, pour chaque classe, à la proportion de monomères ayant son plus proche voisin dans la même classe et se base sur les distances euclidiennes calculées précédemment. Des valeurs matepairs élevées indiquent des sous-groupes bien homogènes et séparés validant la classification. Si les matepairs sont au dessus d'un certain seuil, les deux sous-groupes sont ajoutés séparément à la file pour être potentiellement redivisés ultérieurement. En revanche, si au moins une des valeurs de matepair est au dessous de ce seuil, les sous-groupes seront considérés comme formant un seul groupe et la classification n'est pas validée. Le groupe initial est sauvegardé comme une classe unique.
Le choix de la valeur de ce seuil est essentiel pour déterminer la capacité de la méthode à distinguer les familles.

Taille des groupes La méthode peut théoriquement diviser le jeu de données pour former des familles de deux séquences. La pertinence biologique de ces familles n'étant pas connues, il a été décidé de limiter le nombre de très petites classes en imposant que les groupes de petite taille ne soient pas redécoupés et ce, même si la valeur de matepair est supérieure au seuil choisi. Pour certaines analyses de performances informatiques, ou pour l'étude de petits jeux de données (<100000 séquences), ce seuil sera fixé à 0. Pour l'analyse des séquences de gorille, le seuil sera fixé arbitrairement à 100.

2.2.3 Matériel informatique et langages

L'ordinateur utilisé est un PC sous Debian 8.6. avec 32 cœurs, 64Go de RAM et 64Go de swap.

La méthode présentée ici a été implémentée en Python 3.2 en utilisant notamment les modules sklearn [16], scipy [17], et fastcluster [18]. Les fichiers de résultats sont parsés en bash et les analyses statistiques ont été réalisées avec R (3.2.4). Les différents scripts utilisés sont disponible librement sur github ($https://github.com/flo-j/stage_m2$).

2.3 Alignement, phylogénie et consensus

Les séquences ont été alignées avec muscle [19]. La phylogénie est reconstruite avec SeaView [20] avec le modèle évolutif de Kimura à 2 paramètres (K2P) et l'arbre est construit avec l'algorithme BioNJ (neighbor joining)[21]. Les consensus de nos familles ont été reconstruits en utilisant un script Python développé par l'équipe. Les familles trop grandes pour pouvoir être entièrement alignées, ont été analysées à partir d'un échantillonnage de 1500 séquences choisies aléatoirement en utilisant un script Python développé par l'équipe. Les site de fixation CENP-B ([CT]TTCGTTGGAA[AG]CGGGA) et de pjα (TTCCTTTT[CT]CACC[AG]TAG) ont été identifiés en utilisant le logiciel fuzznuc (package EMBOSS) [22] et en autorisant 3 différence au maximum par rapport au consensus.

3 Résultats

3.1 Amélioration et paramétrisation de l'algorithme

L'algorithme pré-existant a été implémenté en R de façon récursive. Il permettait de classer jusqu'à 20000 séquences en utilisant uniquement la classification hiérarchique, sans utiliser de LDA.

3.1.1 Usage de la mémoire et diminution du temps de calcul

Le programme a été réécrit en Python et l'algorithme modifié afin de le rendre itératif (voir l'algorithme en 2.2). Cette modification permet ainsi de libérer la mémoire au fur et à mesure et non à la fin du programme comme dans la version récursive pré-existante. De plus, en Python, les tableaux sont transmis par référence, et non par copie, lors des appels de fonction évitant ainsi de remplir la mémoire avec des informations déjà dupliquées ailleurs.

Les performances des deux implémentations ont été comparées en utilisant un matepair de 0.9, les 100 premières composantes de l'ACP pour calculer les distances et la taille minimale des familles à découper est fixée à 0 (pas de limite de taille). Afin que les deux versions puissent fonctionner sans LDA, l'analyse est basée sur un jeu de données de 20000 séquences choisies aléatoirement à partir du jeu d'α satellite humain.

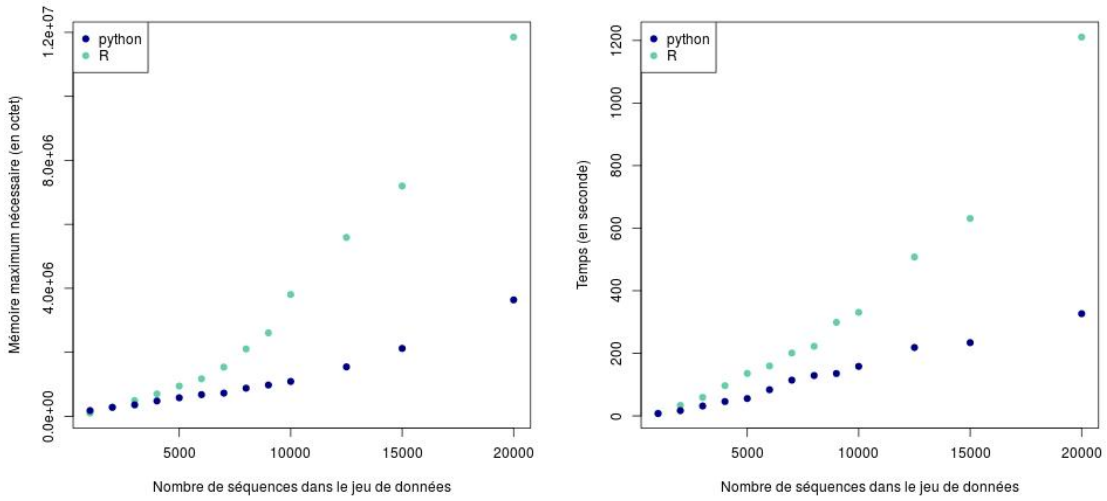

FIGURE 6 – **Usage de la mémoire et du temps** À gauche, Mémoire maximum (octet) durant l'analyse nécessaire aux deux versions du programme en fonction du nombre de séquences du jeu de données. À droite, Temps (seconde) d'exécution nécessaire aux deux versions du programme en fonction du nombre de séquences du jeu de données

La version Python, itérative, de l'algorithme est plus économe en mémoire (voir figure 6A). Pour un jeu de données de 5 000 séquences, cela représente une différence de 400 kilo-octets. Plus le nombre de séquences est important, plus la différence entre les deux versions est importantes. Pour un jeu de données de 20 000 séquences, la version R, récursive, a besoin de 3.25 fois plus de mémoire que la version python.
Cette amélioration en mémoire permet ainsi d'augmenter le nombre de séquences directement traitées par le programme et de limiter l'utilisation de la LDA qui necessite un tirage aléatoire pour fonctionner.

Le passage en Python et le changement pour un algorithme itératif ont aussi permis une diminution du temps de calcul (voir figure 6B). Comme pour la mémoire, plus le jeu de données est grand, plus la différence est importante. Pour un jeu de données de 5 000 séquences, la différence est de 80 secondes contre 884 secondes, soit près de quinze minutes, pour un jeu de données de 20 000 séquences.

3.1.2 Paramétrisation de l'algorithme

L'algorithme implémenté dépendant de 4 paramètres

- Le nombre de composantes de l'ACP utilisé pour le calcul de distance. Ce nombre influe sur le temps de calcul.
- Le nombre de séquences à partir duquel on utilise une LDA. Ce nombre est dépendant de la capacité en mémoire de l'ordinateur et de l'optimisation du programme.
- La valeur minimale de matepair qui permet de s'assurer que les deux sous-groupes sont valides.
- La taille minimale des familles à ne pas découper

Ces deux derniers paramètres permettent de valider la sensibilité du programme.
La taille minimale des familles non redécoupée est fixée arbitrairement à 100 pour les gros jeux de données.

Nombre de composantes de l'acp L'ACP réalisée en amont de la classification permet de réduire le nombre de variables en minimisant la perte d'information et d'obtenir des variables indépendantes pouvant être utilisées par la LDA. Afin de limiter le temps de calcul, la version pré-existante (R/récursive) du programme utilisait les 100 premières composantes de l'ACP

pour le calcul des distances. La réécriture en Python a permis une diminution du temps d'exécution et de l'usage de la mémoire, et une réévaluation de ce paramètre était nécessaire. Utiliser les 100 premières composantes de l'ACP permettent d'expliquer environ 60% de la variabilité totale du jeu de données. Plus le nombre de composantes utilisées est grand plus la variabilité du jeu de données est prise en compte. Les différences de temps entre prendre 100 ou 1024 composantes n'étant pas limitantes (de 10 minutes à 17 minutes pour un jeu de données de 37 000 séquences) l'utilisation de la totalité des composantes est appliquée pour le reste des analyses. Ce paramètre pourra être réestimé si la valeur de k est augmentée.

Échantillonnage des grands jeux de données Les tests effectués sur les machines de l'équipe indiquent qu'il n'est pas possible de traiter plus de 110 000 séquences sans LDA. Pour les plus grands jeu de données (>110 000 séquences), une LDA est utilisée sur un échantillon tiré aléatoirement parmi le jeu de données humain. Cette opération peut introduire une variabilité et une incertitude sur la classification. Pour vérifier que cet échantillonnage n'impacte pas la classification plusieurs valeurs de proportion de séquences utilisées pour l'échantillonnage sont testées. Les classifications obtenues sont ensuite comparées à une classification de référence, où toutes les séquences sont classées sans utilisation de la LDA.

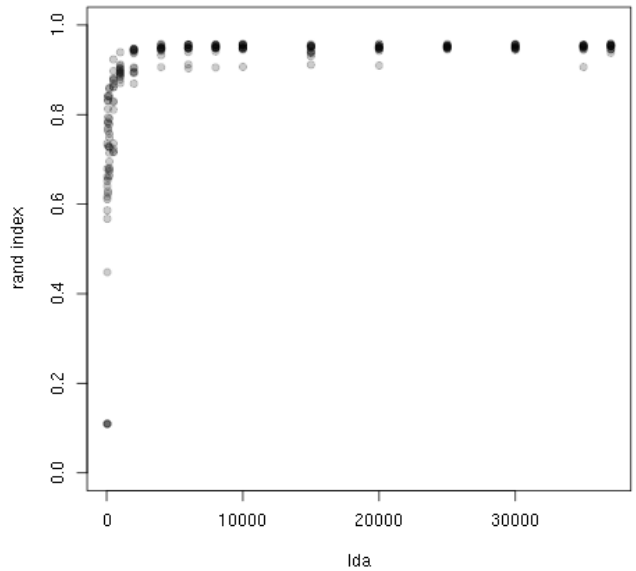

FIGURE 7 – **Variation de l'indice de rand en fonction de la proportion de séquences échantillonnées pour la LDA** Le programme est lancé 10 fois pour chaque valeur d'échantillonnage. Plus la proportion de séquences utilisées pour l'apprentissage est grand, plus l'indice de rand est grand et plus la variation entre deux réplicats est faible

L'indice de rand [23] est utilisé pour comparer les classifications avec la classification de référence. Cet indice, compris entre 0 et 1, permet de mesurer la ressemblance entre 2 classifications. Afin de quantifier la variation entre deux réplicats, le classification est répétée 10 fois. La figure 7 montre que plus le nombre de séquences échantillonnées pour la LDA est grand, plus l'indice de rand est grand et donc plus la classification est proche de la classification de référence, réalisée sans LDA. De même le nombre de séquences échantillonnées pour la LDA est grand, plus la variation entre deux réplicats est faible. La figure 7 montre qu'à partir de 5000 séquences échantillonnées parmi 37000, l'indice de rand varie peu d'un réplicat à un autre et est proche de 1. Cette analyse suggère qu'utiliser 15% du jeu de données est suffisant pour pouvoir l'analyser. Les machines de l'équipe pouvant gérer 110 000 séquences échantillonnées pour la LDA, il permettrait donc de classer correctement jusqu'à 700000 séquences environ.

Seuil de matepair

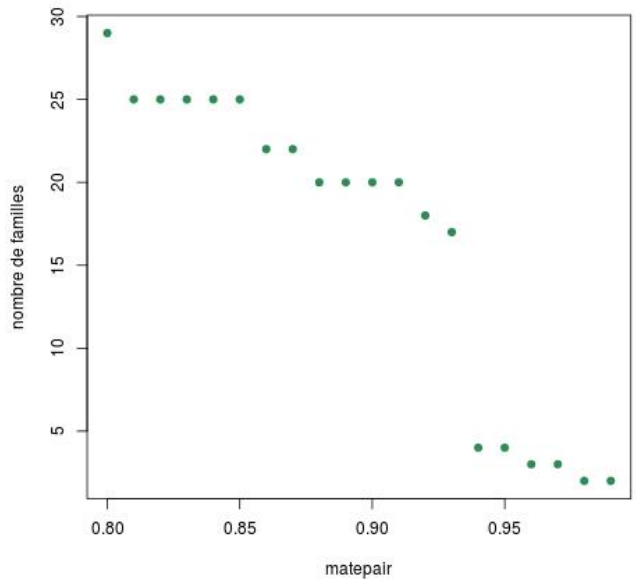

FIGURE 8 – **Nombre de familles obtenues en fonction du seuil de matepair** Cette figure représente l'impact du seuil de matepair sur le nombre de familles obtenues. Plus ce seuil est haut, moins le nombre de famille est grand. 17 familles ont été décrites sur ce jeu de données [10]

Le matepair correspond au nombre de séquences ayant la séquence la plus proche dans le même sous-groupes. Un seuil de matepair élevé indiquera des sous-groupes homogènes et bien séparé dans l'espace des variables tandis qu'un seuil matepair plus faible entraînera plus de classes. Afin de déterminer une valeur rationnelle de matepair le programme a été appliqué sur un jeu

de données décrit dans la littérature [10]. Ce jeu de données est composé de 1431 séquences réparties en 6 familles péricentromérique et 11 familles centromériques organisées en HOR. Plusieurs valeurs de matepair, entre 0.8 et 1, sont testées afin de déterminer celle permettant d'obtenir des résultats similaires à ceux de la littérature.

A partir de 0.91, 20 familles sont retrouvées (voir figure 8), dont les 6 familles centromériques et toutes celles composants le HOR (voir figure 9). Une zone de transition de 38 monomères est observée entre l'HOR et la partie monomérique. Ce qui explique que l'on observe pas précisément les 17 familles (6 péricentromériques + 11 centromérique/HOR) décrites dans la littérature. C'est pourquoi une valeur de matepair de 0.90 est donc choisi comme paramètre pour le programme.

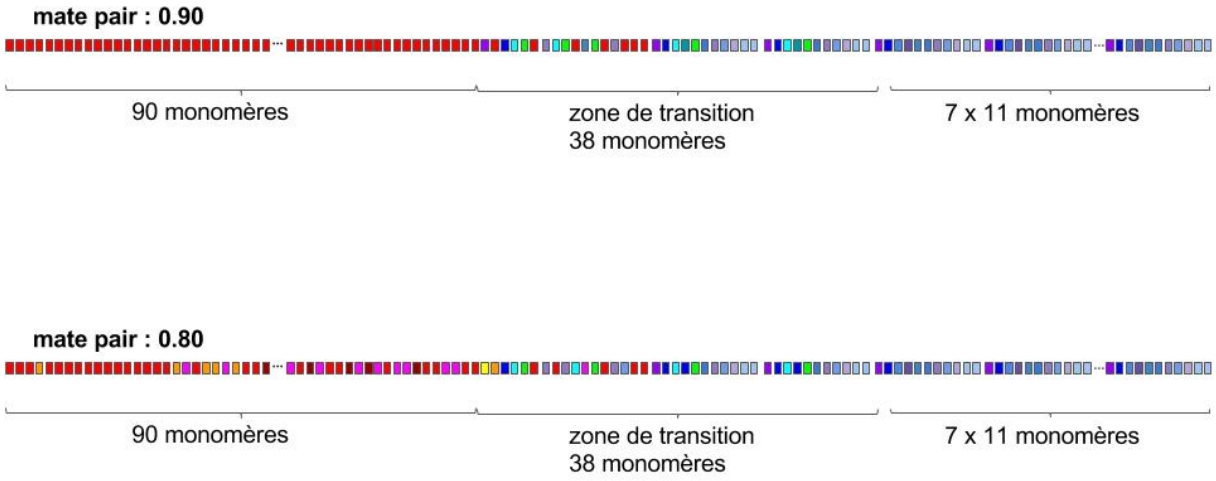

FIGURE 9 – **Représentation de l'HOR d'ordre 11 du brin p du chromosome X et de l'impact du seuil de matepair sur le nombre de famille** L'HOR d'ordre 11 est retrouvé dans les deux cas. Dans le cas d'un matepair de 0.9 une légère zone non structurée est retrouvée entre l'organisation monomérique et l'HOR. Quand le matepair est de 0.8, cette zone non structurée est plus grande.

3.2 Analyse du jeu de données des gorilles

Le jeu de données de gorille contient 122 104 séquences d'α-satellites de gorille et a été analysé en utilisant un matepair de 0.9, 1024 composantes de l'ACP, 110 000 séquences pour la LDA et un seuil de taille des classes à ne plus découper à 100 séquences.

3.2.1 Nombre et taille des familles

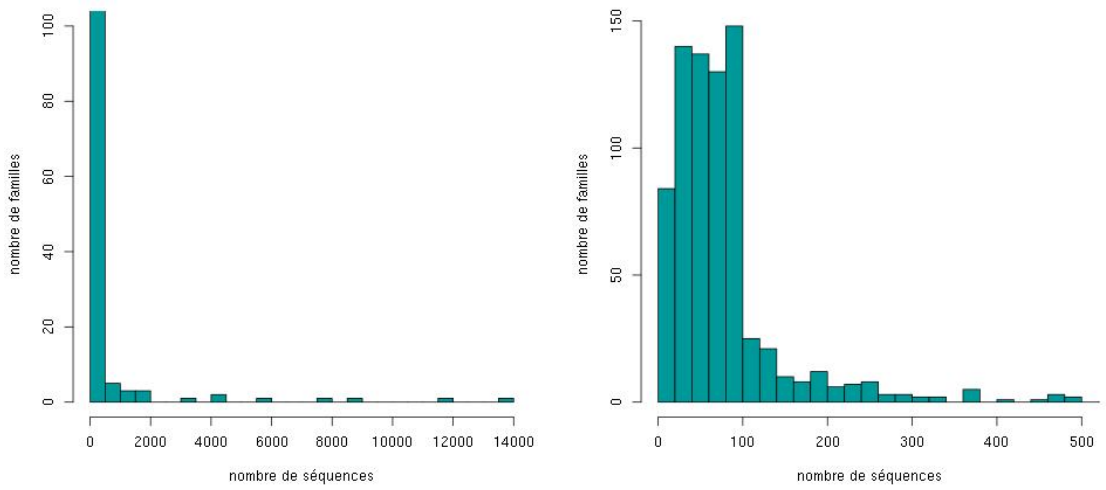

FIGURE 10 – **Histogrammes des tailles de familles** La figure de gauche représente l'histogramme du nombre de séquences par famille. Le premier pic est rogné en hauteur. Il correspond à environ 600 familles. Un zoom de cette partie est visible sur la figure de gauche

Le programme identifie 778 familles. Le nombre de monomères par famille varie de 4 à 13000 (voir figure 10) avec un pic entre 0 et 100. plusieurs familles sont entre 2000 et 6000 séquences et une famille de 13000 séquences est retrouvée. La famille d'environ 13000 séquence représente environ 10% du jeu de données initial. La présence de famille de très grandes tailles pourrait s'expliquer par un arrêt trop précoce de la classification. L'analyse de ces familles sera faite prochainement.

Sur la figure de droite, 30 familles ont moins de 10 séquences ce qui représente moins de 0.1% du jeu de données initial. Elles pourraient correspondre à des séquences rares. Le pic de familles entre 0 et 100 montre que le critère d'arrêt associé à la taille minimale des familles est important.

3.2.2 Longueur des séquences par famille

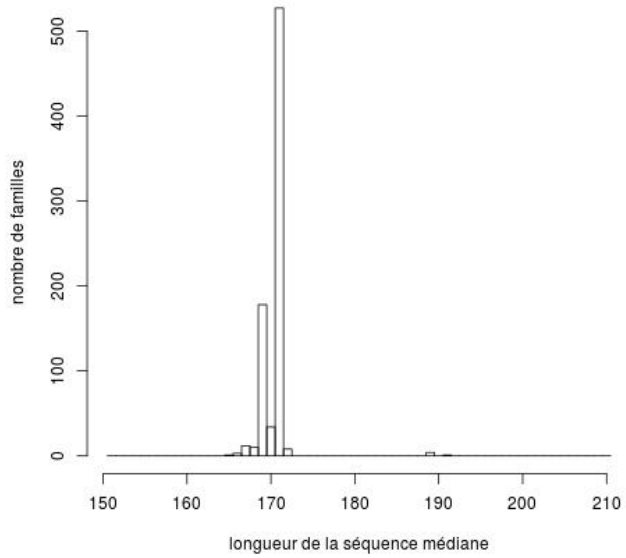

FIGURE 11 – **Distribution des longueurs médianes de séquences (pb) par famille**

La longueur médiane des séquences a été calculée pour chaque famille (figure11). Ces longueurs médianes varient entre 165 et 190 pb avec une distribution bimodale. Un pic majoritaire est observé à 171 pb avec environ 500 familles, un second pic est observé à 169 pb.
La figure 11 montre que plusieurs familles avec de longues séquences ont été identifiées. De façon surprenante, 4 familles à 189 pb ont été retrouvée alors qu'une seule famille avait été décrite par Catacchio. Afin de vérifier si les 4 familles ne sont pas issues d'une potentielle surclusterisation les séquences de ces familles ont été alignées et un arbre phylogénétique (non présenté) a été réalisé. Dans l'arbre, les 4 familles se séparent et forment 4 groupes monophylétiques. Ce résultat suggère qu'il y a 4 familles à 189 pb et pas de surclusterisation dans ce cas.

Parmi les 16 familles décrites par Catacchio et al., en plus d'une famille à 189pb la majorité des familles ont une longueur de 171pb. Les résultats obtenus par le programme sont donc similaire sur ces points. En revanche, seulement 2 familles à 169pb sont retrouvées dans l'article, là où le programme en trouve 170.

3.2.3 Présence des sites de fixation de CENP-B et pJα

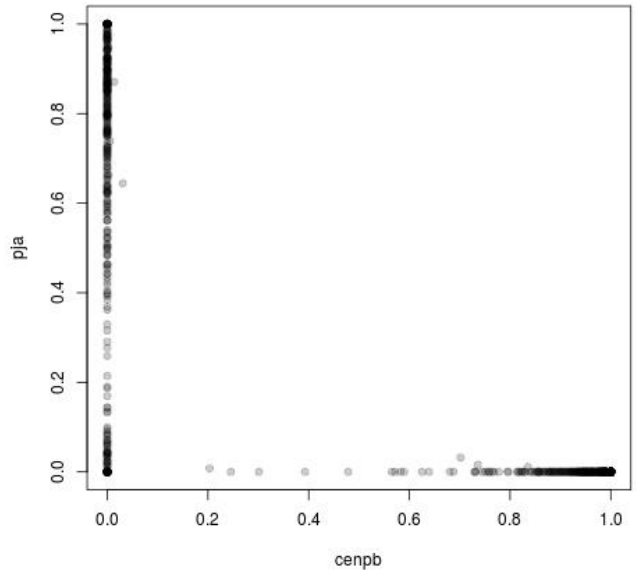

FIGURE 12 – **Proportion de CENP-B et pJα dans chacune des familles** La figure représente la proportion de séquences contenant le motif CENP-B en fonction de la proportion de séquences contenant le motif pJα. Il est important de noter qu'aucune famille ne présente des proportions élevées des deux motifs.

Le motif CENP-B est présent chez 53334 séquences tandis que le motif pJα est retrouvé 36000 fois sur 122 104 séquences. La figure 12 montre le pourcentage de séquences ayant le motif CENP-B par rapport au pourcentage de séquences ayant le motif pJα pour chaque famille. 3 grands groupes de familles sont identifiés : les familles avec une très forte fréquences de séquences ayant un motif CENP-B, les familles ayant une forte fréquence de séquences ayant un motif pJα et enfin des familles n'ayant pas ou très peu de motifs. Comme attendu, aucune famille contenant une proportion forte des deux motifs n'est observée. En effet, aucune famille réunissant des séquences avec les motifs CENP-B et pJα n'a été décrite dans la littérature. La présence de telles familles aurait été le symptôme d'une potentielle mauvaise classification.

3.2.4 Recherche des familles connues

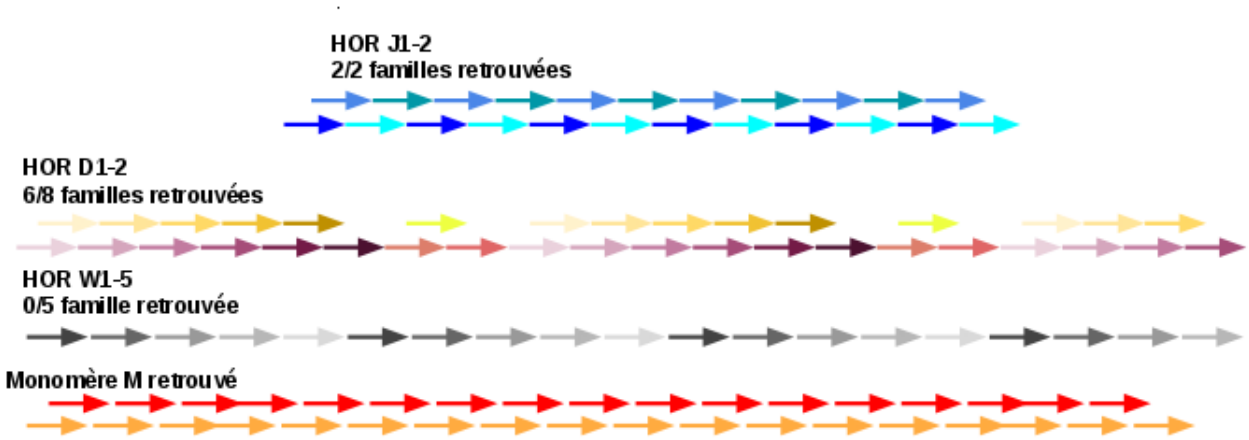

FIGURE 13 – **Recherche de familles décrites par Catacchio** L'HOR J1-J2 est entièrement retrouvé, le D1-2 partiellement. La famille M, ayant une organisation monomérique, est retrouvée. Aucun des monomères formant l'HOR W1-5 n'a été retrouvé.

Les 16 familles identifiées par Catacchio et al. ont été recherchées parmi les 778 familles retrouvées dans notre étude. Les consensus de nos familles ont été comparés avec ceux des 16 familles décrites dans la littérature.

Les monomères de notre étude n'étant pas dans la même phase que ceux publiés (voir 2.1), 16x16 dimères de consensus des familles publiées ont été construit afin d'avoir un recouvrement complet et afin de pouvoir considérer toutes les organisations possibles.

La famille M présente une organisation monomérique. La famille J correspond à un HOR d'ordre 2 contenant les familles J1 et J2. La famille D correspond à un HOR d'ordre 8 contenant les familles D1.0, D1.1, D1.2, D1.3, D1.4, D2.0, D2.2, D2.2. La famille W correspond a un HOR d'ordre 5 contenant les famille W1, W2, W3, W4, W5.

Parmi les 16 familles publiées, 9 sont aussi présentes dans les 778 familles de notre analyse (figure 13). L'HOR J1-J2 composé de 2 familles est entièrement retrouvé, sans misappariement. L'HOR D1-D2 composé de 8 familles est partiellement retrouvé avec 6 familles identifiées sans misappariement. Les deux familles manquantes sont retrouvées en autorisant 2 misappariements par rapport au consensus. La famille M1 est retrouvée avec son organisation monomérique avec un misappariement. En revanche, aucune famille composant l'HOR W1-W5 n'est retrouvée même en autorisant 3 misappariement.

4 Discussion

Le programme développé permet une classification des séquences α satellites. Une première version, écrite en R mais elle était lente et limitée en mémoire. Sa traduction en python et le

changement pour un algorithme itératif a permis une diminution du temps de calcul. En effet, la nouvelle version du programme tourne jusqu'à 3.5x plus vite que l'ancienne pour les paramètres testés.
Ces modifications ont également permis d'augmenter le nombre de séquences pouvant être traitées directement par le programme. En effet, sur un ordinateur ayant 64Go de RAM et 64Go de Swap, on passe de 20000 séquences à 110 000 séquences soit un facteur mémoire multiplié par 5.5.
Cela permet égalment de limiter le nombre de tirages aléatoires effectués. Les résultats suggèrent également que 15% du jeu de données sont suffisant pour obtenir une classification proche à celle que l'on obtiendrait en utilisant toutes les séquences. Cependant ce résultat a été obtenu en ne testant qu'un seul jeu de données, il serait intéressant de recommencer cette analyse sur d'autres séquences.
Le gain de mémoire permet également d'utiliser toutes les composantes de l'ACP pour le calcul des distances entre les séquences, permettant ainsi d'utiliser la totalité de la variabilité du jeu de données.

Pour obtenir des paramètres de matepair biologiquement vraisemblables, un jeu de données décrit dans la littérature a été utilisé. Il a été remarqué qu'un matepair de 0.90 permettait d'obtenir des familles en accord avec la littérature.

Une fois la paramétrisation effectuée, le programme a été utilisé pour analyser un jeu de séquences de gorille, espèce décrite dans la littérature. Ce jeu contient environ 120 000 séquences qui ont été classé en 778 familles.
Le nombre de familles retrouvé est bien plus important que ce qui a été décrit auparavant. Cette différence peut s'expliquer par la taille de l'échantillon utilisé dans cette analyse. En effet, en ayant un plus grand jeu de données, on peut s'attendre à observer des familles rares non détectées dans un jeu de données 40 fois plus petit. L'ordre de grandeur n'est pas abbérant, en effet une étude basée sur les séquences centromériques humaines avait identifié plus de 800 familles [24]. De plus la méthode de détection n'est pas la même ce qui pourrait également expliquer ces différences.
L'analyse des séquences α-satellites chez le gorille a mis en évidence un grand nombre de très petites familles. Certaines ne dépassant pas les 10 monomères. Cela signifierait que 0.01% du jeu de données initial pourrait correspondre à une famille. Il serait intéressant de regarder plus

en détail ces séquences et de regarder leur similitude avec le reste du jeu de données. Il se pourrait en effet que ces séquences proviennent d'erreur de séquençages ou qu'elles correspondent à des événements rares d'amplification.
Lors de cette analyse, les motifs CENP-B box et pJα ont aussi été retrouvés comme attendu. On a pu retrouver trois groupes de familles, celles contenant principalement le motif CENP-B, celles contenant principalement le motif pJα et celles ne contenant aucune des deux box. Ces deux motifs se trouvant à la même position sur la séquence et étant très différents, il n'est pas possible qu'une même famille contienne ces deux motifs en grande proportion. Cela nous indique que le programme a séparé correctement ces deux types de séquences.

Lors de la comparaison de nos familles avec celle de la littérature disponible sur le gorille, 8 consensus de familles sur 16 ont été retrouvées sans misappariement. La famille monomérique M a été retrouvée, cependant il a fallu autoriser un misappariement. Les familles J en HOR d'ordre 2 ont été retrouvées entièrement, les familles D en HOR d'ordre 8 ont été retrouvées partiellement (6/8) sans misappariement puis entièrement si l'on autorise 2 misappariements. Les familles W n'ont pas été retrouvées même avec l'autorisation de plusieurs mesappariements.
L'absence de toutes les familles de cet HOR est intriguant. Il faudrait vérifier la présence de ces familles au sein de toutes les séquences et non pas en comparant seulement les consensus. En effet il est possible que ces séquences ne soient isolées comme familles par l'algorithme et qu'elles se retrouvent mélangées à d'autres séquences. Par contre, si on ne les retrouve pas, il faudra s'interroger sur la représentativité du jeu de données. Le jeu de données utilisé est en effet plus grand que celui utilisé dans l'étude de Catacchio. On s'attendrait donc à obtenir toutes ces familles. On pourrait également étudier le jeu de données directement avec notre programme afin de voir si on retrouve ces familles.

Lors de cette étude, certains paramètres n'ont pas été étudié faute de temps. L'impact de la longueur des k-mers n'a pas été directement testé pour cette analyse et il serait intéressant de tester d'autres valeurs afin de confirmer les résultats.

5 Conclusion

Une nouvelle version de la méthode de classification a été développée permettant, à partir d'un fichier de fréquence de kmers une classification dichotomique, se basant sur les distances euclidiennes entre les séquences, est effectuée séparant le jeu en deux sous-groupes. Chaque sous-groupe est ensuite redécoupé séparément si cela est possible. Cela permet ainsi de retrouvée les familles présentent dans le jeu de données initiales. Cette version, implémentée en Python, permet une diminution du temps de calcul et de l'usage de la mémoire.
Ce programme a ensuite été paramétré et son application à des jeux de données connus donne des résultats encourageants permettant de valider la méthode.
Dans cette étude, la phase a été choisie arbitrairement. Il serait intéressant de vérifier qu'elle ne change rien aux résultats. Cependant, quelque soit la phase choisie, il est important qu'elle soit la même pour tous les jeux de données étudiés afin de pouvoir les comparer entre eux sans risquer d'introduire de biais. De plus, la taille des familles non redécoupée a un impact important sur les résultats, d'autres valeurs seront a testé dans le futur. Il serait intéressant également de s'intéresser à la localisation chromosomique des différentes familles de gorille retrouvées par notre algorithme.
Il est maintenant possible d'appliquer cette méthode à d'autres jeu de données d'autres espèces de primates. On pourra ainsi obtenir les différentes familles d'α-satellites de ces espèces. Ces différentes familles étant obtenues de la même façon, il sera possible d'effectuer une comparaison inter-spécifique et ainsi de d'étudier l'évolution des séquences 'α-satellites.

Références

[1] D. W. Cleveland, Y. Mao, and K. F. Sullivan, "Centromeres and kinetochores : from epigenetics to mitotic checkpoint signaling," *Cell*, vol. 112, no. 4, pp. 407–421, 2003.

[2] S. Santaguida and A. Musacchio, "The life and miracles of kinetochores," *The EMBO journal*, vol. 28, no. 17, pp. 2511–2531, 2009.

[3] K. F. Sullivan, M. Hechenberger, and K. Masri, "Human cenp-a contains a histone h3 related histone fold domain that is required for targeting to the centromere.," *The Journal of cell biology*, vol. 127, no. 3, pp. 581–592, 1994.

[4] S. Henikoff, K. Ahmad, and H. S. Malik, "The centromere paradox : stable inheritance with rapidly evolving dna," *Science*, vol. 293, no. 5532, pp. 1098–1102, 2001.

[5] G. Giannuzzi, C. R. Catacchio, and M. Ventura, *Centromere Evolution : Digging into Mammalian Primary Constriction*. INTECH Open Access Publisher, 2012.

[6] H. F. Willard, "Evolution of alpha satellite," *Current opinion in genetics & development*, vol. 1, no. 4, pp. 509–514, 1991.

[7] D. M. Kurnit and J. J. Maio, "Variable satellite dna's in the african green monkey cercopithecus aethiops," *Chromosoma*, vol. 45, no. 4, pp. 387–400, 1974.

[8] C. Lee, R. Wevrick, R. Fisher, M. Ferguson-Smith, and C. Lin, "Human centromeric dnas," *Human genetics*, vol. 100, no. 3, pp. 291–304, 1997.

[9] V. A. Shepelev, A. A. Alexandrov, Y. B. Yurov, and I. A. Alexandrov, "The evolutionary origin of man can be traced in the layers of defunct ancestral alpha satellites flanking the active centromeres of human chromosomes," *PLoS Genet*, vol. 5, no. 9, p. e1000641, 2009.

[10] I. Alexandrov, A. Kazakov, I. Tumeneva, V. Shepelev, and Y. Yurov, "Alpha-satellite dna of primates : old and new families," *Chromosoma*, vol. 110, no. 4, p. 253, 2001.

[11] L. Romanova, G. Deriagin, T. Mashkova, I. Tumeneva, A. Mushegian, L. Kisselev, and I. Alexandrov, "Evidence for selection in evolution of alpha satellite dna : the central role of cenp-b/pjα binding region," 1996.

[12] L. Cacheux, L. Ponger, M. Gerbault-Seureau, F. A. Richard, and C. Escudé, "Diversity and distribution of alpha satellite dna in the genome of an old world monkey : Cercopithecus solatus," *BMC genomics*, vol. 17, no. 1, p. 916, 2016.

[13] C. Catacchio, R. Ragone, G. Chiatante, and M. Ventura, "Organization and evolution of gorilla centromeric dna from old strategies to new approaches," *Scientific reports*, vol. 5, 2015.

[14] J. H. Ward Jr, "Hierarchical grouping to optimize an objective function," *Journal of the American statistical association*, vol. 58, no. 301, pp. 236–244, 1963.

[15] N. Altemose, K. H. Miga, M. Maggioni, and H. F. Willard, "Genomic characterization of large heterochromatic gaps in the human genome assembly," *PLOS Comput Biol*, vol. 10, no. 5, p. e1003628, 2014.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn : Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[17] E. Jones, T. Oliphant, P. Peterson, *et al.*, "SciPy : Open source scientific tools for Python," 2001–. [Online ; accessed <today>].

[18] D. Müllner, "fastcluster : Fast hierarchical, agglomerative clustering routines for R and Python," *Journal of Statistical Software*, vol. 53, no. 9, pp. 1–18, 2013.

[19] R. C. Edgar, "Muscle : multiple sequence alignment with high accuracy and high throughput," *Nucleic acids research*, vol. 32, no. 5, pp. 1792–1797, 2004.

[20] M. Gouy, S. Guindon, and O. Gascuel, "Seaview version 4 : a multiplatform graphical user interface for sequence alignment and phylogenetic tree building," *Molecular biology and evolution*, vol. 27, no. 2, pp. 221–224, 2009.

[21] N. Saitou and M. Nei, "The neighbor-joining method : a new method for reconstructing phylogenetic trees.," *Molecular biology and evolution*, vol. 4, no. 4, pp. 406–425, 1987.

[22] P. Rice, I. Longden, and A. Bleasby, "Emboss : the european molecular biology open software suite," 2000.

[23] W. M. Rand, “Objective criteria for the evaluation of clustering methods,” *Journal of the American Statistical association*, vol. 66, no. 336, pp. 846–850, 1971.

[24] K. H. Miga, Y. Newton, M. Jain, N. Altemose, H. F. Willard, and W. J. Kent, “Centromere reference models for human chromosomes x and y satellite arrays,” *Genome research*, vol. 24, no. 4, pp. 697–707, 2014.

6 Résumé

Chez de nombreuses espèces la région centromérique, site permettant l'attachement des microtubules aux chromosomes, est composée de séquences d'ADN répétées en tandem. Chez les primates, ces séquences centromériques nommées α satellites, font environ 171 pb. Elles ont un taux d'identité entre 60% et 100% et il est possible de les regrouper en familles. Pour étudier ces familles, des méthodes utilisant de la phylogénie existent mais elles ne permettent pas de classer un grand nombre de séquences (> 100 000). De plus, ces méthodes ne sont pas objectives et ne permettent pas de comparer les espèces entre elles. Pour répondre à ce problème, une méthode de classification permettant de traiter des centaines de milliers de séquences a été développée par l'équipe ARChE. Une première version du programme a été écrite en R en 2016. La première partie de mon stage a permis d'améliorer le temps de calcul et l'usage de la mémoire du programme. Pour cela un changement d'algorithme et une traduction en python ont été effectués. Une paramétrisation du programme sur un petit jeu de tests dont les familles ont été décrites dans la littérature a ensuite été réalisée. Une fois paramétré, le programme a été utilisé sur un grand jeu de données de séquences de gorille. Une comparaison avec des données disponibles dans la littérature nous a permis de valider notre approche.

7 Abstract

In many species, the centromere region is the binding site of chromosomes' microtubules. This region is composed of DNA short tandem repeats. In Primates these repeats are named α satellites and are about 171 pb. They share between 60% and 100% of sequence identity and are organised in families. To study these families, phylogenetic methods exist but they can't class a lot of sequences (>100 000). Moreover these methods are not objective and it's not possible to compare species. A classification method was designed to clusterize hundreds of thousands sequences and an R version of the program was developped in 2016. During the first part of my stage, execution time and memory usage was improved by changing the algorithm and translating the program in Python. In a second part, the program was finetuned with a small known dataset. Finally, we run the program on a large gorilla dataset. A comparison with the litterature let us validate our approach.