RAPPORT DE STAGE OPTION BIOSTIC

Analyse bioinformatique des viromes de deux espèces d'oiseaux migratrices en Guyane.

Développement d'outils dédiés à la visualisation des données "viromics".

Etudiante : Alba Thio i Pera

Tuteur de l'ECN: Olivier Roux

Tuteur à l'entreprise : Anne Lavergne et Sourakhata Tirera

Entreprise: Institut Pasteur de la Guyane

Adress de l'Entreprise: 23 Avenue Pasteur, BP 6010, Cayenne Cedex 97306

Dates: du 8 avril 2019 au 31 août 2019

TABLEAU DE MATIÈRES

1. INTRODUCTION

1.1 Contexte et environnement de travail

Le stage présenté dans ce rapport a été effectué à l'**Institut Pasteur de la Guyane** (IPG) du mois d'Avril à Août 2019. L'IPG fait partie du Réseau International des Instituts Pasteur qui regroupe 32 instituts dans 27 pays répartis sur les cinq continents. L'IPG a été crée le 7 décembre 1940 et est basé à Cayenne (chef-lieu de la collectivité territoriale de Guyane), en Amérique du Sud, à l'orée de la forêt amazonienne. Les principales missions de l'IPG sont de contribuer à la prévention et au traitement des maladies, en priorité infectieuses et s'articulent autour de trois composantes : la recherche, l'appui à la santé publique, l'enseignement et la formation.

L'IPG est constitué de cinq unités de recherche dont les activités se concentrent sur les maladies tropicales infectieuses et parasitaires (Paludisme, leishmaniose) et virales (arboviroses telles que la dengue et le chikungunya ou les hantavirus) d'intérêt local.

Plus concrètement, ce stage a été effectué au **Laboratoire des Interactions Virus-Hôtes** (LIVH). Cette unité de recherche s'intéresse aux interactions virus, hôtes et environnements abordées *via* différentes approches écologiques, évolutives et immunologiques. Ces interactions sont également étudiées par des outils de séquençage de haut débit pour la caractérisation de la diversité virale et ses variations en fonction des espèces et des environnements et ce afin de mieux comprendre la biologie des infections. Le laboratoire est composé de deux chercheurs, deux techniciens et deux étudiants en thèse et il est équipé des serveurs permettant de traiter les données issus du séquençage à haut-débit.

L'encadrement de ce stage a été effectué par Anne Lavergne, responsable du LIVH, et Sourakhata Tirera, ingénieur en bioinformatique du LIVH et doctorant à l'Université de Guyane. Dans le cadre de son projet de thèse, le laboratoire a entrepris une amélioration des chaînes de traitement des données de viromes issues de séquençage à haut-débit. Une partie de ce projet consistait à développer un algorithme d'identification des fragments d'acides nucléiques viraux en exécutant BLAST de façon récursive afin d'augmenter la précision et l'exhaustivité de la diversité virale.

1.2 *Birds and Diversity of Viruses*

Le stage que j'ai réalisé a pour cadre le projet *BirDiV - Birds and Diversity of Viruses* - qui a comme objectif principal de caractériser les virus hébergés chez différentes espèces d'oiseaux, potentiellement réservoirs de virus émergents, par une approche de séquençage à haut débit.

Un étude des organismes infectieux connues pour être pathogènes pour l'homme a démontré qu'une partie importante de ces agents a une origine zoonotique [**1**] ce qui souligne l'importance de mettre en place une surveillance des réservoirs naturels viraux. Les oiseaux font partie de ces réservoirs et hébergent de nombreux virus qui ont un impact important en santé publique comme par exemple, le virus West Nile ou bien encore le virus de la grippe. Ce projet est donc motivé par l'importance, dans ce contexte, d'investiguer la circulation et la diversité virale chez les oiseaux afin de diminuer les risques d'émergences qui peuvent survenir dans les zones d'interface entre les populations humaines, la faune et son environnement naturel.

La Guyane est un lieu d'escale unique au monde pour les espèces d'oiseaux migratrices, grâce à sa position géographique proche de l'équateur et à l'abondance de boues, en zones côtières, chargées en nutriments du fait de la proximité avec le fleuve Amazone. La Guyane compte plus de 450 espèces d'oiseaux en contact avec des milliers d'oiseaux migrateurs provenant du continent nord américain [**2**].

Les deux espèces d'oiseaux ciblées dans cette étude sont *Actitis macularia* et *Charadrius semipalmatus* (Figure 1). Ils sont communément dénommés le Chevalier grivelé et le Pluvier semipalmé respectivement et ils appartiennent aux oiseaux limicoles, de l'ordre des Charadriiformes. Ils sont fréquemment trouvés le long des rivages et des vasières où ils cherchent de la nourriture (comme des insectes ou crustacés) dans la boue ou le sable.

Figure 1 : *Charadrius semipalmatus* (gauche) et *Actitis macularia* (droite). Source: uniprot.org

Leur habitat de reproduction est en terrain ouvert proches des eaux fraîches du Canada et du nord des États-Unis pour *Actitis macularia,* et sur les plages du nord du Canada et de l'Alaska pour *Charadrius semipalmatus*. Pendant la saison de non-reproduction (hiver), ils migrent sur de longues distances vers les zones côtières du sud des États-Unis, les Caraïbes et une grande partie de l'Amérique du Sud (dont la Guyane). C'est pendant leur hivernage qu'ils se retrouvent en contact avec des espèces locales en

Guyane, soit en restant toute la durée d'hivernage, soit en faisant une halte migratoire avant de poursuivre leur migration plus au sud du continent vers le Brésil.

C'est le contact de ces espèces migratrices avec des espèces locales, partageant leurs habitats en période migratoire, qui pourrait contribuer à l'amplification et à la diffusion de virus hébergés au cours des migrations.

Ces deux espèces ont été capturées en Guyane pendant les périodes d'hivernage de 2016 et 2018. Le site de capture était localisé près du fleuve Mahury (proche de Cayenne). Lors de leur capture, les oiseaux étaient pesés, l'âge estimé et des prélèvements cloacaux et trachéaux ont été réalisés. Le sexe de chaque individu a été déterminé, a posteriori, par une approche moléculaire (amplification d'une portion du gène chromodomain helicase DNA binding-CHD) celle-ci n'étant pas réalisable sur des critères morphologiques. Au total 42 individus appartenant à l'espèce *Charadrius semipalmatus* et 113 à l'espèce *Actitis macularia* ont été collectés (Tableau S1 en annexe).

1.4 NGS et pipeline de traitement

Le séquençage à haut débit est une technologie qui permet de séquencer un grand nombre de fragments d'ADN de façon rapide. La technologie utilisée dans cette étude est Illumina, qui permet d'identifier simultanément les bases d'ADN lorsqu'elles sont incorporées dans la chaîne d'acides nucléiques. Chaque base émet un signal de fluorescence unique lorsqu'elle est ajoutée au brin en cours de synthèse, ceci permettant de déterminer la séquence d'ADN.

Malgré les nombreux avantages de générer de grandes quantités de séquences en peu de temps, ces données sont difficiles à analyser car elles correspondent à un volume de données très important pouvant inclure un certain nombre de biais. Ainsi, les méthodes les plus courantes ne produisent que des courts fragments (de 100 à 300 nucléotides de longueur). De plus, avant l'étape de séquençage, les fragments d'ADN sont fragmentés, puis amplifiés par PCR, ce qui peut induire (1) l'apparition de mutations erronées et l'**amplification préférentielle** de séquences (toutes les molécules d'ADN d'origine ne sont pas amplifiées dans les mêmes proportions) mais aussi (2) l'apparition de **séquences chimériques** qui se composent de plusieurs fragments d'ADN issus de différents organismes.

De manière globale, un métagénome viral est analysé de la manière suivante :

- On effectue un contrôle de qualité des reads en sortie de séquençage.
- Un assemblage est par la suite réalisé sur les reads nettoyés.
- Une identification des fragments viraux est réalisée par homologie par rapport aux bases de référence.

Au LIVH, un pipeline informatique suivant ces grandes étapes, a été développé (Figure 2) dans le but d'optimiser le traitement des données de virome.

La première étape de ce pipeline correspond au contrôle de qualité qui sert à voir la qualité générale des données directement après le séquençage. Ce contrôle permet d'enlever, entre autres (1) les lectures des séquences qui ont une mauvaise qualité générale, (2) les possibles lectures des séquences dupliquées ou surreprésentées et (3) les adaptateurs utilisés lors du séquençage qui sont inclus dans les lectures des séquences (read). La troisième étape correspond à l'assemblage qui va servir à reconstituer les fragments des séquences initiales et réduire la quantité des données. La dernière étape correspond à l'identification des fragments de gènes viraux par un script récursif de déchimérisation.

Figure 2 : *schéma simplifié des étapes du pipeline du laboratoire.*

Les deux parties qui vont faire l'objet de cette amélioration sont donc le nettoyage des données et le script récursif. Divers outils informatiques présentent différentes méthodes de nettoyage. La question se pose de savoir quel nettoyage s'adapte le mieux aux données et l'effet que ce nettoyage va avoir par rapport à la diversité virale dans les résultats. D'autre part, les paramètres du script récursif sont à tester pour voir leurs effets sur le temps d'exécution ou encore sur la diversité virale dans les résultats.

1.5 Visualisation des résultats

Les résultats de viromes obtenus en sortie de traitement sont sous formes de fragments d'acides nucléiques assignés à des séquences de virus connus. Dans ces résultats, des milliers de fragments sont chacun assignés à une séquence connue par son identifiant (**accession number**) et ont des informations complémentaires tels que le chemin taxonomique complet de l'espèce la plus proche à laquelle ils sont assignés, la longueur du fragment assigné, etc. Toutes ces informations sont complexes à analyser lorsqu'on les regarde sous forme de liste et une connaissance du shell linux est nécessaire pour y accéder.

Ce sont les biologistes du laboratoire qui vont se servir de ces informations pour la suite du projet et par conséquent, l'accès à ces informations nécessite qu'elles soient organisées et rendues accessibles simplement, visuelles et "user-friendly". Bien qu'il existe quelques outils, comme Krona [**3**] ou Keanu

[**4**], aucun outil à notre connaissance ne prend en charge tous les paramètres/aspects évoqués ci-dessus.

De plus, suite à l'identification des virus présents dans un échantillon, il est des fois utile de "pousser" la classification des fragments viraux afin de comprendre leurs liens évolutifs avec les virus existants. Pour cela, des reconstructions phylogénétiques peuvent être réalisées et nécessitent une étape d'alignement multiple (entre contigs et gènes homologues de la base de données). Ce travail peut s'avérer fastidieux si chaque identifiant de gène doit être retrouvé, rapatrié et (éventuellement) fragmenté manuellement.

1.6 Objectifs

L'objectif principal de ce stage est d'étudier la diversité virale présente chez deux espèces d'oiseaux migratrices capturées en Guyane en analysant leur virome.

Ce travail se base sur l'analyse de données issues de séquençage haut débit (Illumina Hiseq 2500). Il repose sur l'utilisation du pipeline développé au laboratoire mais aussi sur l'amélioration de ce dernier en réalisant des tests de différents paramètres que ce soit à l'étape de filtrage ou dans le script récursif.

En parallèle, une représentation graphique et interactive des résultats du pipeline sur la diversité virale et les alignements des fragments de gènes viraux trouvés sera développée.

2. METHODES

Partie I : Production et description des données

Les différents échantillons collectés en 2016 et 2018 ont été agrégés par espèce, par type de prélèvements (cloaques ou trachée), par sexe ainsi que par année. Un maximum de 24 prélèvements individuels ont été agrégés selon les critères indiqués ci-dessus en essayant d'obtenir des pools de taille homogène par espèce (Tableau S1 en annexe).

En premier lieu, les échantillons agrégés ont été filtrés afin d'éliminer un maximum de bactéries puis ultracentrifugés afin de concentrer les virus présents dans les échantillons. Pour réduire la quantité d'ADN et d'ARN contaminants, les échantillons ont été traités à la DNase et à la RNase afin de ne conserver « que » les particules virales encapsidées. Les échantillons ont ensuite été extraits en TRIzol® (pour l'obtention des ARNs viraux) et en phénol (ADN) puis amplifiés par amplification aléatoire (Whole Transcriptome and Whole Genome Amplification, respectivement). Les différents pools amplifiés ont été marqués individuellement en utilisant des adaptateurs spécifiques et agrégés en fonction de leur concentration (quantité égale pour chacun des pools). Ces banques ont été préparées selon des protocoles standards Illumina et analysées sur un séquenceur HiSeq 2500 permettant une lecture de fragment de 250 paires de base en double sens (paired end). Ainsi, chaque fragment initial est séquencé dans deux sens "forward" et "reverse", qui vont être représentés par deux reads. Quand un des reads est absent on parle de read "single".

Au total, 22 pools ont été séquencés et un contrôle négatif (le contrôle négatif correspondant à un échantillon de DMEM qui a été traité depuis l'étape d'extraction comme tous les autres pools). Les données de séquençage obtenues correspondent à des fichiers au format ".fastq", qui seront le point de départ des analyses faites pendant ce stage.

Les échantillons ont été nommés avec le codage suivante :

Prélèvement _ espèce _ sexe _ année _ pool
Cl/Sa _ AM/CS _ M/F _ 16/18 _[I/II]

Les types de prélèvement primaire peuvent être des écouvillons de salive (Sa) ou de cloaques (Cl), les deux espèces étudiées sont *Actitis macularia* (AM) et *Charadrius semipalmatus* (CS), le sexe femelle (F) ou mâle (M) et l'année 2016 (16) ou 2018 (18). Lorsque différentes pools représentaient les mêmes types d'échantillons (mêmes prélèvement, espèce, sexe et année) ils ont été nommés I ou II (Tableau S1 en Annexe).

Partie II : Pipeline

2.1 Contrôle de Qualité : FastQC

La première partie de toute analyse de métagénomique est la vérification de la qualité des données générées et le filtrage si nécessaire de ces données. Dans les projets de séquençage à haut-débit, les données doivent être traitées par des procédures de contrôle de qualité avant qu'elles puissent être utilisées pour l'analyse. La procédure de ce contrôle comprend habituellement l'identification et le filtrage de séquences tels que des lectures de mauvaise qualité et des lectures de contaminants tels que les séquences de PhiX174 (utilisée comme témoin de séquençage), ces séquences pouvant affecter considérablement et parfois induire des erreurs dans l'analyse en aval.

L'outil FastQC [**5**] a été utilisé pour cette partie du pipeline. Cette analyse permet de vérifier la qualité des données d'entrée pour pouvoir les corriger avant d'avancer dans les analyses.

La sortie de la commande FastQC est divisée en différents tesst, qui peuvent avoir des "warnings", "failures" ou bien avoir un "bon état". La liste de ces tests et une explication de leur fonction et interprétation peuvent être trouvées dans la Liste L1 en Annexe.

2.2 Filtrage des données

Actuellement, il existe un grand nombre d'outils bioinformatiques qui sont dédiés au filtrage des données. Trois outils différents ont été utilisés pour analyser les données brutes. En premier, pour l'ensemble des données l'outil FaQCs a été utilisé afin d'éliminer les séquences de mauvaise qualité, les séquences de PhiX174 et d'autres artéfacts. A partir des données de sortie de FaQCs (fichiers fastq), les outils fastq-mcf et bbnorm ont été utilisés en parallèle (Figure 3).

- fastq-mcf a été utilisé pour dédupliquer les données
- Bbnorm a été utilisé pour pour normaliser les données.

Les résultats de sortie de ces trois outils ont été comparés après assemblage pour voir lequel était le plus adapté aux traitements de nos données.

I. FaQCs

FaQCs est un logiciel de contrôle de qualité qui supprime les données de mauvaise qualité des séquences à partir d'un traitement algorithmique et parallèle [**6, 7**]. Cet outil a été principalement utilisé pour éliminer les adaptateurs du séquençage. Les données d'entrée sont les fichiers fastq bruts. La

sortie (**DUP**) sera donc le nouveau fichier de séquences filtrées ainsi qu'un fichier PDF avec des statistiques pour faire la comparaison des fichiers d'origine avec les nouvelles données générées (Figure 3). La liste de ces tests statistiques et une explication de leur fonction et interprétation sont présentées dans la Liste L2 en Annexe.

L'outil FaQCs présente différents paramètres pouvant être sélectionnés. Tous les paramètres ont été laissés par défaut sauf les suivants :

- -adapter yes : fonction pour couper les adaptateurs Ilumina présents dans les lectures.
- -polyA yes : fonction pour couper les polymères de base A qui ont une longueur supérieure à 15pb.
- -debug : fonction pour laisser les fichiers intermédiaires du traitement.
- -t : fonction pour fixer le nombre de CPUs pour l'exécution du script.
- -p : fonction pour indiquer que les reads sont en paired-end (avec une lecture forward et une reverse).
- -prefix : fonction pour indiquer le nom prédéterminé des fichiers de sortie.
- -d : fonction pour indiquer le répertoire de sortie.

II. Fastq-mcf

Fastq-mcf est un outil qui permet de filtrer les séquences présentes dans un fichier fastq. Cet outil filtre principalement par élimination des adaptateurs et des séquences dupliquées [**8, 9**]. Dans notre cas, l'outil a uniquement été utilisé pour filtrer les séquences dupliquées. Le fichier de sortie obtenu correspond aux reads qui ont été dédupliqués (**woDUP**) (Figure 3).

Différentes options de filtrage peuvent être utilisées. Les suivantes ont été choisies :

- -D 50 : critère pour éliminer les reads qui ont 50 bases identiques.
- -q 20 : une base avec une qualité inférieure à 20 sera éliminée.
- n/a fasta : fichier avec séquences des adaptateurs qui vont être comparés avec les séquences. Ne disposant pas de ce fichier, avec l'option n/a, aucune comparaison ne sera faite avec le fichier d'entrée.

III. Bbnorm

Bbnorm est un outil de normalisation pour les reads de séquençage haut-débit. La normalisation consiste à analyser tous les k-mers d'un jeu de données pour éliminer ceux qui sont sur-représentés correspondant à des zones de couvertures trop élevées. La normalisation permet de générer un jeu de données moins complexe car elle permet d'éliminer des k-mers contenant des erreurs et accélère l'assemblage en utilisant un jeu de données réduit. La normalisation est utilisée pour des grands jeux de données comme les métagénomes. Bbnorm, avec ses paramètres par défaut, a été utilisé pour normaliser les reads. Le fichier obtenu correspond aux reads normalisés (**NORM**) (Figure 3).

Ces trois jeux de données étant filtrés, ils sont prêts à être assemblés.

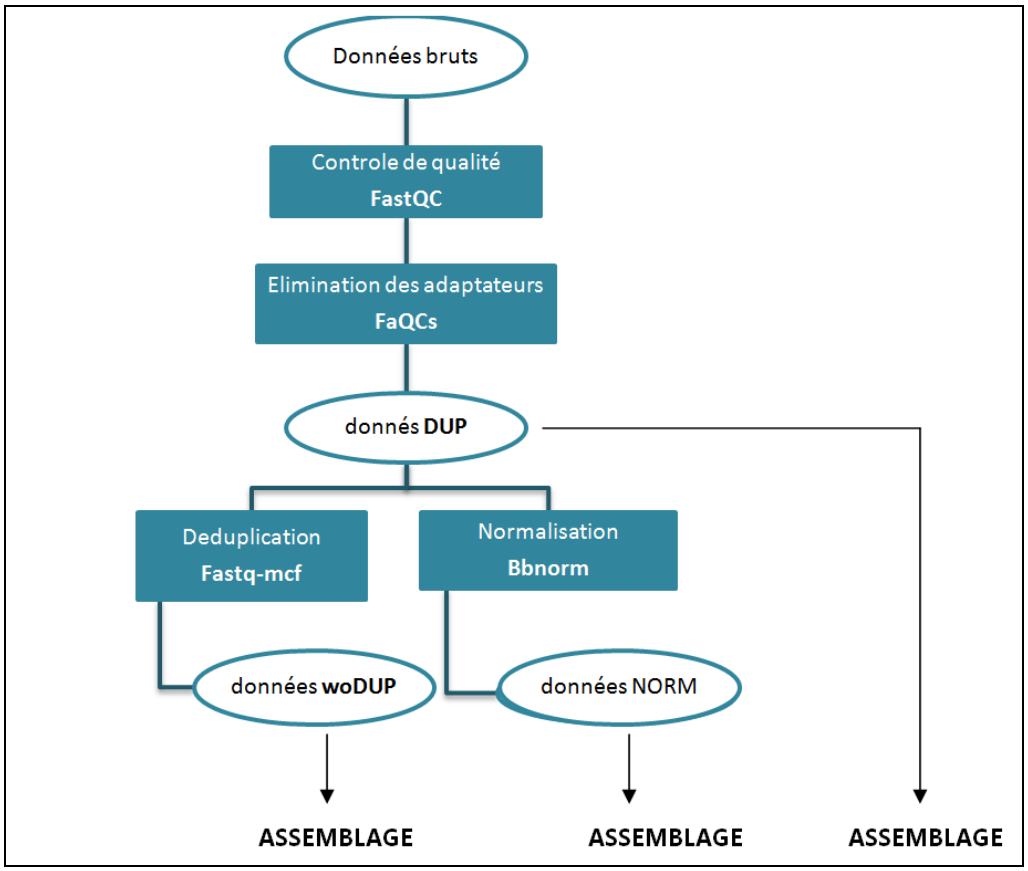

Figure 3 : *schéma de filtrage des données.*

2.3 Assemblage

La plupart des approches actuelles d'analyse des données de séquençage haut-débit reposent sur la comparaison des reads à des génomes de référence (mapping). Lorsqu'un génome de référence n'est pas disponible, l'assemblage *de novo* peut être utilisé. Dans ce projet, nous travaillons avec des métagénomes. Etant donné que plusieurs organismes sont présents dans nos échantillons et un génome de référence ne peut pas être utilisé. Par conséquent, la technique qui va être utilisée est l'assemblage *de novo*.

Un assemblage *de novo* vise à reconstituer les séquences des fragments initiaux, qui ont été auparavant fragmentés pour le séquençage. Les algorithmes se servent de chevauchements entre lectures. Pour cela, les reads sont découpés en plus petits fragments appelés k-mers. Ces k-mers sont mis dans un graphe appelé Graphe de De Bruijn (Figure 4). En dernier lieu, la lecture des chemins de ce graphe relie

les reads pour donner, en sortie, des contigs. Néanmoins, certains reads ne sont pas assemblés pour former des contigs.

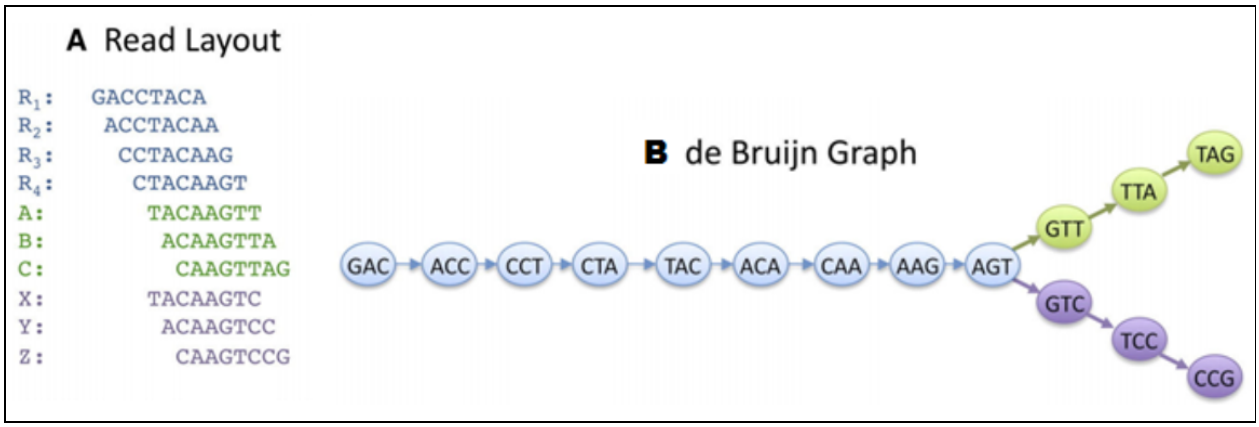

Figure 4 : Fonctionnement du Graphe de De Bruijn. Source [**10**].

En sortie d'analyse, la quantité des données, qui était initialement l'ensemble des reads, va être donc réduite aux contigs et aux reads qui ne sont pas assemblés diminuant ainsi la taille des jeux de données à analyser. Cette diminution du nombre de données (qui restent représentatives du jeu de données initial) est ici une étape importante car elle permet de réduire le temps de traitement pour la suite du pipeline (étape du BLAST).

Les assemblages sont évalués par la mesure de la longueur des contigs générés : la longueur du plus grand contig, la longueur moyenne des contigs, la longueur totale combinée de l'ensemble de contigs et la N50, la N50 étant la longueur du contig avec laquelle on atteint le 50% de la longueur totale quand on ordonne les contigs du plus grand au plus petit [**11**].

I. Megahit

Pour l'assemblage, le logiciel utilisé est Megahit, un outil conçu pour faire des assemblages *de novo* et qui permet le traitement de données métagénomiques complexes et volumineuses [**12**]. Megahit a été utilisé afin d'obtenir les différents contigs qui vont représenter nos reads. Cette étape est réalisée pour les reads filtrés obtenus à partir des trois méthodes de filtrage (DUP, woDUP et NORM) et ce afin d'obtenir trois assemblages distincts. La capacité de ceux-ci à représenter les données initiales et les résultats de taxonomie virale seront comparés.

Le paramètre --outprefix de megahit a été utilisé pour nommer automatiquement les sorties avec le préfixe donné, pour avoir une sortie spécifique pour chaque échantillon. Les paramètres -1 -2 et -r indiquent la liste des fichiers, séparés par une virgule, qui sont forward, reverse et single

respectivement. Tous les autres paramètres ont été laissés par défaut car ceux-ci semblent être optimaux, des évaluations ayant été précédemment faites au laboratoire.

2.4 Choix de filtre

Afin de faire le choix entre les filtres utilisés (DUP, woDUP et NORM), leur comparaison va être basée sur deux critères :

- la capacité des assemblages de chaque jeu de données à représenter l'ensemble des données initiales. Cette capacité est testée en réalisant un mapping des reads initiaux sur les contigs de chaque assemblage.
- la diversité virale obtenue avec l'assemblage de chaque jeu de données en comparant les familles virales obtenues.

I. Mapping

Le mapping est une technique qui consiste à assigner les reads à une position d'un contig ou d'un génome de référence, généralement. Les contigs issus de l'assemblage sont donc pris comme référence pour le mapping. Pour évaluer la qualité des assemblages, le pourcentage des reads qui ont mappé va être analysé, en fonction des contigs issus des différents assemblages selon le filtre utilisé. Le mapping est réalisé pour les trois assemblages : (1) assemblage issu des reads juste nettoyés et non dédupliqués (DUP), (2) l'assemblage issu des reads nettoyés et dédupliqués avec fastq-mcf (woDUP) et (3) l'assemblage issu des reads nettoyés et normalisés avec Bbnorm (NORM).

Pour cette partie du pipeline, un script créé par Sourakhata Tirera a été utilisé. Dans ce script, on utilise l'outil BWA (Burrows-Wheeler Aligner) qui aligne des reads courts pour faire le mapping [**13**]. À chaque fois, le mapping pour les reads Forward et Reverse est réalisé d'un côté (stockés dans un fichier Paired), et les Singles d'un autre côté (stockés dans un fichier Single).

Suite au mapping, divers outils de samtools ont été utilisés [**14, 15**]. Cet outil fournit divers utilitaires pour manipuler les alignements dans le format BAM, y compris le tri, la fusion, l'indexation et des statistiques de mapping. En premier lieu, l'outil *samtools view* est utilisé pour générer le fichier de sortie au format BAM. Puis, l'outil *samtools sort* est utilisé pour trier les alignements. Cette procédure a été faite pour les fichiers Paired et Single mappés sur les trois collections de contigs (DUP, woDUP, NORM). Enfin, les fichiers triés, Paired et Single, ont été poolés avec *samtools merge* et le fichier résultat indexé avec *samtools index*. De plus, *samtools flagstat* a été utilisé pour obtenir des statistiques sur les différents fichiers.

D'autre part, *samtools view -b -f4* a été utilisé pour obtenir, en fichier BAM, les séquences qui n'ont pas été mappées. Ces fichiers ont été passés au format fastq avec *samtools fastq* et au format fasta avec l'outil *fq2fa* [**16**].

Pour comparer les contigs résultant des trois filtrages, les résultats de sortie de *samtools flagstat* qui correspondent aux statistiques des mappings pour chaque bibliothèque de contigs ont été analysés.

Dans ce fichier de sortie, on trouve les spécifications suivantes du mapping pour chaque échantillon après chaque filtrage :

- Total : nombre total des reads pour chaque échantillon, calculé en faisant la somme des reads forward, reverse, single et les reads supplémentaires (reads ayant mappé à différents endroits).
- Supplementary : nombre de fois où un read chimérique a mappé à plusieurs endroits.
- Mapped : nombre de reads qui ont mappé sur les contigs, et un pourcentage sur le total.
- Paired in sequencing : somme des read1 et read2.
- Read 1 /Read 2 : nombre de reads forward et reverse respectivement.
- Properly paired : nombre de reads forward et reverse qui ont mappé aux même contig orientés l'un vers l'autre.
- With itself and mate mapped : nombre de reads appariées où les deux sont mappés.
- with mate mapped to a different chr : nombre de reads appariées où les deux sont mappés sur des contigs différents.
- Singletons : nombre de reads appariées où l'un est mappé et pas l'autre.

II. Diversité virale

Après BLAST, les familles virales détectées entre les différents échantillons pour les différents filtres utilisés (DUP, woDUP et NORM) ont été comparées afin de voir quel jeu de données reflète la plus grande diversité.

2.5 Déchimérisation et BLAST

I. Les chimères

Les **chimères** sont des recombinants artificiels entre deux séquences ou plus, de différents organismes, qui sont potentiellement générées pendant les étapes de ligation et l'amplification des banques ADN et ARN [**17**]. Dans notre cas, du fait de la faible quantité d'acides nucléiques viraux extraits, nous avons recours à des techniques d'amplification (WGA/WTA). Ces techniques nécessitent des matrices de grande taille avant l'étape d'amplification et ont donc recours à une phase de ligation des différents acides nucléiques extraits.

Ces molécules artificielles rendent difficile la distinction entre les différents fragments de séquences présents dans une chimère, ce qui peut entraîner une sous-estimation du niveau de diversité virale dans les échantillons. De plus, lors de l'assemblage *de novo,* du fait de l'utilisation de fragments courts appelés k-mers, il est possible qu'une concaténation artificielle se fasse ce qui induit une augmentation du nombre de chimères dans les contigs.

Après séquençage, les chimères doivent être identifiées à l'aide d'outils bioinformatiques. Cependant, leur détection n'est pas triviale car les longueurs des reads sont courts et que le début et la fin des possibles fragments de séquences virales (contenues dans les chimères) ne sont pas facilement détectables. Cela rend difficile la détection de tous les organismes présents dans une chimère et peut être la cause d'une perte d'information quant à la diversité virale.

En réponse à ces contraintes, le traitement de cette question a été intégré à l'étape du BLAST car si plusieurs séquences d'organismes différents sont concaténées, BLAST devrait les retrouver. Le script utilisé pour réaliser cette étape a déjà été développé au laboratoire et il est développé dans la partie suivante.

II. Script de traitement de BLAST récursif

Basic local alignment search tool (BLAST) est un outil informatique qui permet de rechercher des régions similaires entre séquences en acides aminés ou nucléotides et de réaliser un alignement de ces régions [**18, 19**]. Le programme permet de comparer les séquences des échantillons avec des séquences d'une base de données. Cette méthode est utilisée pour associer les séquences à des familles de gènes ou pour déduire des relations fonctionnelles et évolutives entre séquences.

Il existe différents types de BLAST en fonction de la nature de la séquence d'entrée et de la base de données utilisée. Dans ce projet, deux types de blast ont été utilisés :

- blastn : les données d'entrée sont des séquences nucléotidiques qui vont être comparées à une base de données de séquences nucléotidiques.
- blastx : les données d'entrée sont des séquences nucléotidiques qui vont être traduites en séquences protéiques et qui vont être comparées à une base de données de séquences protéiques.

Un script de déchimérisation qui exécute BLAST de façon récursive a été utilisé. Ce script est codé en langages python et bash et il suit les étapes illustrées dans les Figures 5 et 6. Ce script permet de comparer les séquences des contigs à la base de données nt (Nucleotid collection) [**20**], puis à une base protéique virale faite au laboratoire (téléchargement des protéines virales depuis NCBI (nr) et

déduplication à 100% d'homologie réalisé avec CD-HIT) et enfin à la base nr (Non-redundant protein database) [**21**].

Le filtrage d'un contig soumis à BLAST se fait par « plages » de coordonnées (Figure 5). Un ensemble de plages est définie par comparaison deux à deux des « subjects ». Un paramètre de décalage est introduit (réglable en ligne de commande), celui-ci est de 5 par défaut. Quand un « subject » a ses coordonnées de départ et d'arrivée complètement incluses dans une plage déjà définie, il y est ajouté sinon une nouvelle plage se crée. Les alignements jugés être sur la même plage sont soumis à une sélection pour retenir le meilleur sur le critère du bit-score. Les zones de contig qui ne donnent pas de résultats (longueur >= 50 nuc par défaut) en sortie de BLAST sont resoumises au même type de BLAST et ce tant qu'il reste de telles zones à soumettre à BLAST. A la fin de l'exécution d'un BLAST récursif, nous obtenons deux types de sorties : un fichier contenant des séquences assignées et un fichier contenant des séquences non assignées qui vont être soumises au BLAST récursif suivant.

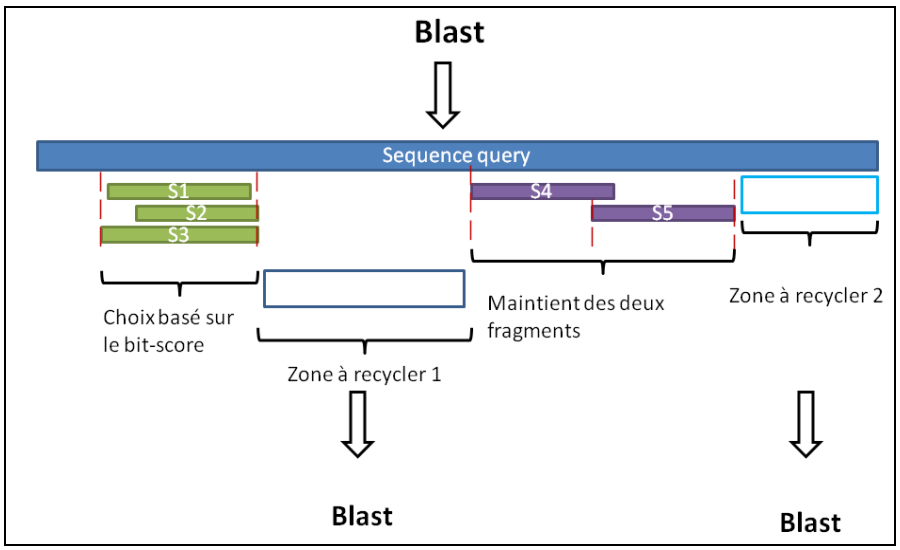

Figure 5 : *Fonctionnement de la récursivité du blast pour la déchimérisation.*

Les étapes pour un passage de blast récursif sont les suivantes :

1. Exécution du BLAST pour les fichiers fasta avec les séquences des contigs.

2. Filtrage des résultats de BLAST (Figure 6) qui va avoir les sorties : le fichier avec les contigs qui ont donné un résultat (Filt_CSV), les fragments non-identifiés (>50 nt), parties de contigs (N+1_Fasta) , les alignements (Q_S_aln) correspondants et le fichier avec les contigs qui n'ont pas donné un résultat (Bn_neg_fasta).

3. Soumission de façon récursive des zones non couvertes des contigs (N+1_Fasta) qui donnent un résultat positif (voir Figure 6). C'est dans cette partie là que la déchimérisation a lieu. Si on est dans le cas d'une séquence chimérique, les parties non couvertes par le blast précédent peuvent appartenir à d'autres organismes que celui trouvé initialement.

4. Assignement des taxons aux résultats positifs retrouvés dans le fichier Filt_CSV. La sortie est un fichier avec l'ensemble de séquences assignées à des taxons.

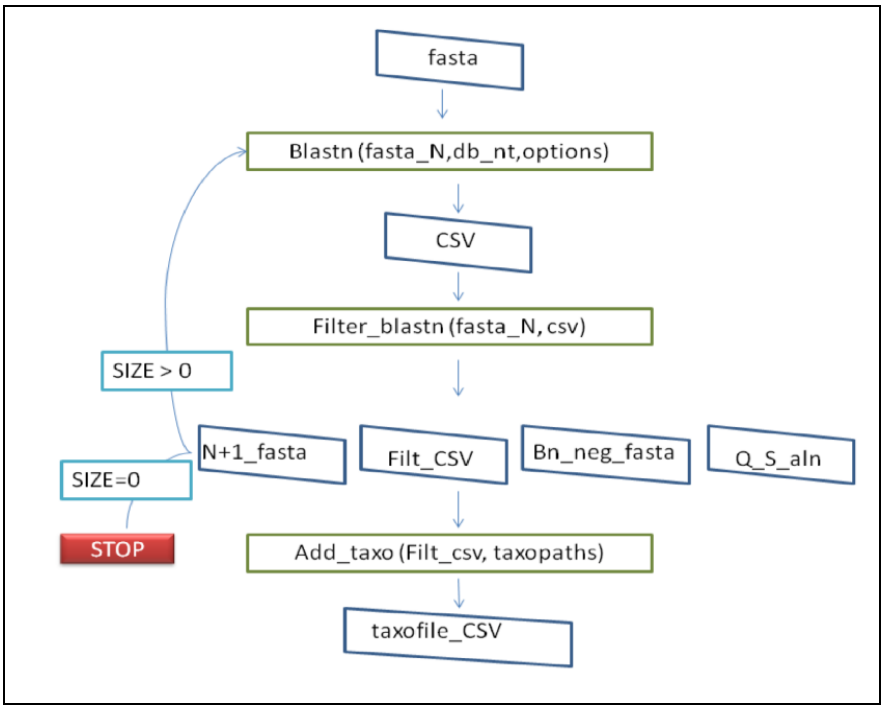

Figure 6 : *Schéma de fonctionnement du script de traitement des blast et déchimérisation.*

Cette boucle récursive sera exécutée trois fois (pour les 3 étapes d'assignation taxonomique). En premier, on réalise un blastn, puis un blastn2 et enfin un blastx. Les séquences sont d'abord soumises au blastn récursif. Les séquences non assignées par ce premier blastn (bn_neg_fasta) sont soumises à un deuxième boucle récursive avec le blastn2, plus sensibles aux insertions/délétions mais aussi plus lent. Pour finir, les séquences non assignées au blastn2 (bn2_neg_fasta) sont soumises à la troisième boucle récursive avec Blastx qui est divisé en deux étapes, d'abord contre la base virale locale qui va servir à gagner du temps, puis les séquences qui ont été détectées positives contre cette base sont re-soumises à un blastx contre la base protéique nr afin de détecter la présence de faux positifs. En effet, certaines séquences qui ont été détectées positives après un BLASTx uniquement sur la base virale peuvent s'avérer être des faux positifs car elles montrent une taxonomie qui n'est pas celle d'un virus mais d'un autre organisme après un BLASTx sur la base nr (e-value et bit-score en faveur d'un assignement à un organisme autre qu'un virus).

Fasta → BLASTN → bn_neg_fasta → BLASTN2 → bn2_neg_fasta → BLASTX

III. Tests des paramètres du blast

L'outil BLAST a différents paramètres d'entrée. Des tests sur deux paramètres vont être réalisés pour voir quels sont leurs effets sur les résultats du blast récursif et pour choisir les paramètres les plus optimaux pour le projet. Les paramètres à tester sont la longueur minimale de la zone couverte et la valeur de la e-value.

- Le paramètre de la **longueur** est la valeur de la longueur minimale des parties des contigs qui ont mappé avec la base de données. Ce même paramètre est la longueur minimale à partir de laquelle un fragment va être recyclé. Les valeurs de longueur de 10 pb, 100 pb et 200 pb seront testées.

- La **E-value** est une probabilité pour BLAST qui indique le nombre de hits attendus qui pourrait être trouvé juste par hasard. Par exemple, un e-value de 10 signifie que jusqu'à 10 hits peuvent être trouvés juste par hasard, étant donné la même taille d'une base de données aléatoire. Cela signifie que plus basse sera sa valeur, plus fiable sera le résultat correspondant. Ce paramètre est donc utilisé pour filtrer les résultats de BLAST pour obtenir uniquement des résultats avec une valeur inférieure ou égale à la valeur seuil [**22**]. Les valeurs de E-value de 10, 1e-2 et 1e-9 seront testés.

Pour faire les tests, 1000 séquences d'une taille minimale de 1000 paires de bases (pb) ont été utilisées, en format fasta et choisies aléatoirement parmi tous les échantillons et assemblages. Ces séquences ont été analysées par le script de BLAST récursif en changeant les paramètres. Le test a été réalisé pour les 9 combinaisons possibles des 6 paramètres.

Pour comparer les sorties des différents tests, la variation de la diversité virale ainsi que le taux de faux positifs en fonction des paramètres ont été étudiés. Même si cela n'a pas été un critère décisif dans notre cas, les différents temps d'exécution du script récursif ont également été étudiés. En effet selon la taille des jeux de données le temps d'exécution peut devenir un paramètre critique.

Pour voir la variation de la diversité virale, les résultats des taxonomies trouvées entre les différentes combinaisons ont été comparés. De plus, pour voir le taux de faux positifs, toutes les séquences qui ont donné des résultats viraux vont être passées manuellement en BLASTn et BLASTx (à partir du site BLAST de NCBI) et comparés avec la sortie de notre script de BLAST récursif. Pour finir, les différentes diversités après avoir éliminé les faux positifs ont été comparées.

Partie III: Visualisation

2.6 Recensement des besoins de la visualisation

La contrainte principale pour l'application à réaliser pour la visualisation est qu'elle soit accessible, interactive et qu'elle rende les informations visuelles. Afin de réaliser un cahier de charges pour fixer les besoins de cette visualisation, diverses réunions et échanges avec Anne Lavergne et Sourakhata Tirera ont eu lieu.

Les deux informations importantes à visualiser seront les chemins taxonomiques résultants ainsi que les alignements des contigs aux gènes correspondants. Pour rendre cela claire, ces deux informations ont été représentées dans deux visualisations indépendantes mais reliées grâce à des liens interactifs pour pouvoir naviguer de l'une à l'autre facilement.

A partir de là, les fonctions de la visualisation ont été divisées en deux : celles dédiées à la visualisation de la taxonomie et celles pour la visualisation des alignements.

I. Visualisation de la taxonomie

L'objectif principal reste de représenter de façon claire les résultats (taxonomie). Afin d'accomplir cela, diverses fonctions principales et complémentaires ont été implémentées. Sont considérées comme fonctions principales, celles qui doivent être réalisées pour considérer que la visualisation est claire, et les complémentaires, celles qui apportent des améliorations mais qui ne sont pas essentielles.

FP1.1 Information des taxons jusqu'à la famille virale représentée sous forme d'arbre.
FP1.2 Liaison des deux parties de visualisation : lorsqu'on clique sur les familles virales, la visualisation des alignements de cette famille va s'afficher.

FC1.1 Arbre rétractable : lorsqu'on clique sur les nœuds de l'arbre tous les descendants de ce nœud vont être cachés.
FC1.2 Choix du type d'arbre : un function permet de choisir si voir l'arbre linéaire ou circulaire.

II. Visualisation des alignements

L'objectif principal reste encore de représenter de façon claire les résultats, dans ce cas-là, la visualisation des alignements. Pour cette partie, nous nous sommes inspirés de la partie graphiques des résultats de BLAST lorsqu'on l'utilise en ligne. Afin d'accomplir cela, diverses fonctions principales et complémentaires ont été implémentées.

FP2.1 Alignements des gènes avec leurs contigs associés dessous dans la position d'alignement correspondante.

FP2.2 Les gènes sont organisés en clusters de similarité (accompagnés de leur contigs).

FC2.1 Les gènes et contigs sont cliquables et lorsqu'on les clique, une fenêtre avec des informations (telles que le nom du gène, de l'espèce etc.) est affichée.

FC2.2 Les contigs sont représentés de différentes couleurs en fonction de leur e-value.

FC2.3 Les protéines qui ont été détectées lors du blastx sont transformées en leur gène d'origine (en nucléotides) et c'est celui-ci qui va être représenté.

FC2.4 Les contigs qui ont donné des résultats en blastx sont transformées en nucléotides pour la longueur et les index dans la représentation.

FC2.5 Lorsqu'on clique sur les clusters des gènes, un fichier FASTA avec les séquences des gènes et contigs est affiché (en nucléotides).

FC2.6 Les contigs seront (potentiellement) fractionnées et seules les parties s'alignant avec des gènes de la base de données seront écrites dans le fichier fasta.

2.7 Fichiers d'entrée

Les fichiers d'entrée pour la partie de visualisation sont les trois fichiers de sortie du pipeline de traitement (Figure 7). Ces fichiers sont des listes où chaque ligne correspond à une association d'un contig de l'échantillon en question avec un gène de la base de données.

Dans chaque ligne les informations suivantes sont présentes :

- Identifiant taxonomique unique (TaxId) de l'organisme auquel appartient le gène de la base de données.
- Nom du contig en question.
- Pourcentage d'identité entre la partie mappé du contig et du gène.
- Accession number (Identifiant) du gène/protéine associée au contig en question.
- Longueur de l'alignement.
- E-value de l'alignement.
- Position de début et fin de l'alignement dans le contig.
- Position de début et fin de l'alignement dans le gène.
- Chemin taxonomique de l'organisme associé.

k141_9604	MK293717	89.355	1.36e-99	310	10	319	1316	1607	Viruses;Genomoviridae;	Gemycircularvirus
k141_9604	MK293717	89.355	1.36e-99	310	10	319	1316	1607	Viruses;Genomoviridae;	Gemycircularvirus
k141_10109	MK293717	99.644	6.71e-142	281	1	281	1903	2183	Viruses;Genomoviridae;	Gemycircularvirus
k141_1022	MK293717	99.291	4.17e-64	141	124	264	2004	2144	Viruses;Genomoviridae;	Gemycircularvirus
k141_11212	MK293717	99.647	5.19e-143	283	1	283	1917	2199	Viruses;Genomoviridae;	Gemycircularvirus
k141_1151	MK293717	99.647	5.19e-143	283	1	283	1904	2186	Viruses;Genomoviridae;	Gemycircularvirus
k141_13088	MK293717	93.421	2.92e-54	152	2	153	2026	2177	Viruses;Genomoviridae;	Gemycircularvirus
k141_16667	MK293717	99.647	5.19e-143	283	1	283	1913	2195	Viruses;Genomoviridae;	Gemycircularvirus
k141_18354	MK293717	99.524	1.40e-102	210	1	210	1967	2176	Viruses;Genomoviridae;	Gemycircularvirus
k141_2207	MK293717	99.593	1.65e-122	246	2	247	1954	2199	Viruses;Genomoviridae;	Gemycircularvirus
k141_2349	MK293717	98.707	5.60e-111	232	1	232	121	352	Viruses;Genomoviridae;	Gemycircularvirus
k141_2547	MK293717	98.835	0.0	515	2	516	1	515	Viruses;Genomoviridae;	Gemycircularvirus
k141_2590	MK293717	99.645	1.87e-142	282	2	283	1918	2199	Viruses;Genomoviridae;	Gemycircularvirus
k141_2631	MK293717	99.571	2.60e-115	233	1	233	1953	2185	Viruses;Genomoviridae;	Gemycircularvirus
k141_5259	MK293717	98.940	1.12e-139	283	1	283	212	494	Viruses;Genomoviridae;	Gemycircularvirus
k141_6187	MK293717	99.644	6.71e-142	281	1	281	1919	2199	Viruses;Genomoviridae;	Gemycircularvirus
k141_10109	MK293717	99.644	6.71e-142	281	1	281	1903	2183	Viruses;Genomoviridae;	Gemycircularvirus
k141_1022	MK293717	99.291	4.17e-64	141	124	264	2004	2144	Viruses;Genomoviridae;	Gemycircularvirus
k141_11212	MK293717	99.647	5.19e-143	283	1	283	1917	2199	Viruses;Genomoviridae;	Gemycircularvirus
k141_1151	MK293717	99.647	5.19e-143	283	1	283	1904	2186	Viruses;Genomoviridae;	Gemycircularvirus
k141_13088	MK293717	93.421	2.92e-54	152	2	153	2026	2177	Viruses;Genomoviridae;	Gemycircularvirus
k141_16667	MK293717	99.647	5.19e-143	283	1	283	1913	2195	Viruses;Genomoviridae;	Gemycircularvirus
k141_18354	MK293717	99.524	1.40e-102	210	1	210	1967	2176	Viruses;Genomoviridae;	Gemycircularvirus
k141_2207	MK293717	99.593	1.65e-122	246	2	247	1954	2199	Viruses;Genomoviridae;	Gemycircularvirus
k141_2349	MK293717	98.707	5.60e-111	232	1	232	121	352	Viruses;Genomoviridae;	Gemycircularvirus
k141_2547	MK293717	98.835	0.0	515	2	516	1	515	Viruses;Genomoviridae;	Gemycircularvirus

Figure 7 : *Extrait d'un exemple du fichier de sortie du pipeline.*

Les informations dans le rendu du fichier de sortie du pipeline ne sont pas très accessibles et cela illustre la nécessité de l'outil de visualisation.

2.8 Les Scripts

Plusieurs scripts ont été utilisés pour cette partie de visualisation (Figure 8). Un Tableau S2 en Annexes explique de façon résumée les fonctions principales de chaque script.

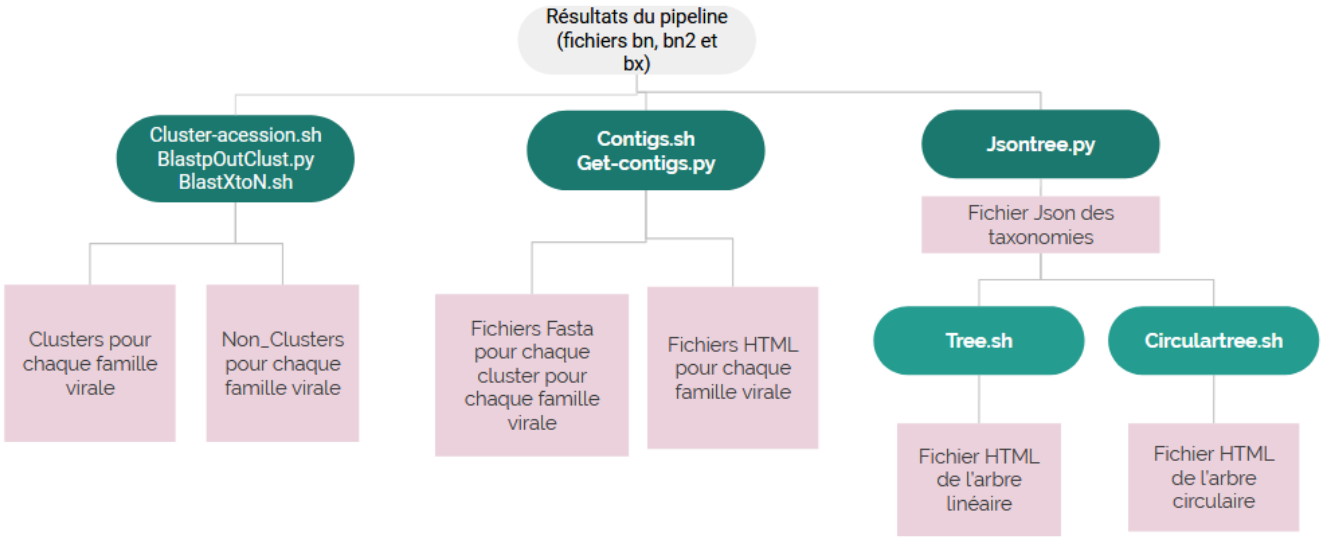

Figure 8 : *schéma des successions des scripts et leurs fichiers d'entrée et sortie. En rose les fichiers, en vert les scripts exécutables.*

I. Clusterization

Le script **cluster-accessions** est exécuté en premier. Ce script, codé en bash, prend comme fichiers d'entrée les trois fichiers décrits précédemment.
D'abord, une liste des familles virales présentes dans les fichiers d'entrée est créée. Ce script sera donc une boucle de cette liste pour faire le même traitement pour chaque famille.

Une liste des accessions numbers associés à la famille virale en question dans cet échantillon est créée ainsi qu'un fichier fasta des séquences de ses accession numbers. Les fichiers fasta seront utilisés ensuite pour faire un blast afin de comparer les séquences de chaque fichier entre elles.

Ensuite, les fichiers de sortie des BLASTn et BLASTp (les données d'entrée sont des séquences protéiques et qui vont être comparées à une base de données de séquences protéiques) sont traités de la façon suivante : toutes les lignes correspondant à la comparaison du même gène sont éliminées et les lignes conservées sont soumises au script **BlaspOutClust**.

Le script **BlaspOutClust**, codé en python par Sourakhata Tirera, prend la sortie du blast et sort des fichiers de clusters des accessions numbers. Ces fichiers sont des listes des accessions numbers qui sont très proches entre eux et que vont être considérés comme des clusters de gènes homologues.

La suite, dans le script cluster-accessions, est de comparer les fichiers clusters avec le fichier liste des accessions numbers pour mettre tous les accessions numbers qui n'ont pas été clusterisés dans un fichier non-cluster. Pour la famille virale en question, il y aura donc : des fichiers pour chaque cluster avec la liste des accessions numbers des clusters et un fichier pour tous les accessions numbers non clusterisés.

La dernière étape de ce script sera d'exécuter le script **BlastXtoN**, codé en bash. Nous passerons par ce script tous les clusters sortis du fichier d'entrée correspondant aux résultats du blastx. Les clusters du blastx n'ont que des accession number de protéines et donc ce script va aller chercher sur le site NCBI [**23**] le gène original qui code pour la protéine en question a été obtenue et il va refaire de nouveaux fichiers cluster mais cette fois avec les accession numbers correspondant aux gènes en nucléotides. Les protéines ne sont pas forcément codées par la totalité du "gène" donc les indexes de début et de fin de la région qui code pour la protéine sont aussi stockés.

Par exemple, pour l'échantillon Cl_AM_F_16_II et la famille virale *Poxviridae*, les sorties du script cluster-accession, conjointement avec les scripts BlaspOutClust et BlastXtoN sont representés dans la Figure 9.

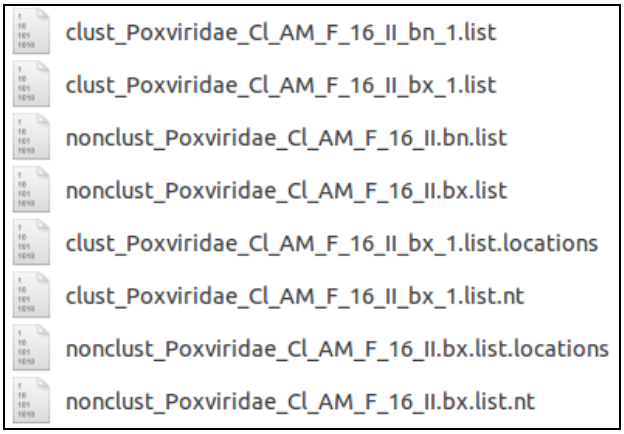

Figure 9 *: Exemple des fichiers de sortie de la partie clusterisation pour la famille virale Poxviridae dans l'échantillon Cl_AM_F_16_II.*

Pour cet exemple, il y a un cluster des résultats qui sortent du blastx (clust_*_bx_1.list) et également un des résultats de blastn (clust_*_bn_1.list). Il y a ensuite les fichiers avec des accession numbers des séquences des bases de référence (nr) non clusterisés (nonclust_*) pour blastn et blastx. Finalement, il y a tous les fichiers des clusters et des non clusterisés, résultats de blastx, qui ont été transformés en gènes (fichiers .nt) et leurs coordonnées (.locations).

II. Création des fichiers HTML d'alignements contigs-gènes

Le script **contigs.sh** commence par la création d'une liste avec toutes les familles virales présentes dans l'échantillon traité. La suite du script boucle sur ces familles virales. Un fichier texte va être créé pour chaque famille virale, qui va servir de données d'entrée pour le HTML.

Pour chaque famille virale, une boucle est faite autour de tous les clusters présents. Pour chaque cluster, un fichier fasta est créé avec toutes les séquences des accessions numbers dans les clusters. Pour la création de ces fichiers fasta, deux méthodes différentes sont utilisés : (1) d'abord la séquence au format fasta correspondant à l'accession number en question est cherchée dans la base de données nr, et si jamais l'accession number ne se trouve pas dans cette base, (2) la séquence fasta de l'accession number non trouvée est cherchée sur le site de NCBI via internet. Ensuite, le nom du cluster qui va servir de titre au cluster dans la visualisation va être écrit dans le fichier texte de la famille virale.

Dans la boucle des clusters, une autre boucle va être faite sur les accessions numbers présents dans le cluster. Dans cette boucle, l'accession number en cours de traitement va être cherché dans le fichier d'entrée du script car dans chaque ligne où on trouve cet accession number il y a l'information de son alignement avec un contig de l'échantillon. Ces informations vont être écrites dans le fichier texte qui va servir à représenter le contig aligné. Pour finir, le fichier **get-contigs.py** va servir à écrire dans le fichier fasta de ce cluster la partie du contig qui a été alignée avec le gène.

Le script **get-contigs.py** est codé en python et contient une fonction récursive pour couper les fichiers fasta des contigs originaux et retourne juste la partie qui a été alignée.

La dernière partie du script **contigs.sh** consiste à écrire, pour chaque famille virale, dans un fichier .html, le code en html, javascript et CSS, qui va servir à faire la visualisation des alignements.

Les sorties de cette partie sont un fichier html pour chaque famille virale qui contient la visualisation des alignements et qui a comme données le fichier ".txt" créé par le script (contigs.sh), et un fichier "fasta" pour chaque cluster qui contient les séquences des gènes associées ainsi que les parties des contigs qui lui ont été alignées.

III. Création des fichiers HTML arbre

La dernière étape est la création des fichiers html pour l'arbre. Pour cela, un fichier temporaire avec les chemins taxonomiques jusqu'à la famille virale est créé. Ce fichier va être soumis au script **jsontree.py** codé en python qui va transformer la liste des chemins taxonomiques en un arbre au format json.

Ce fichier, au format json, contient les données qui vont être écrites dans un fichier ".html", le code en html, javascript et CSS pour faire la visualisation circulaire et linéaire des arbres. L'arbre circulaire est écrit par le script **circulartree.sh** et le linaire par le script principal **tree.sh**.

3. RESULTATS

Partie I: Pipeline

3.1 Contrôle de Qualité

Au total, quatre-vingt douze rapports de contrôles de qualité ont été obtenus pour les 23 échantillons analysés. Pour simplifier l'explication des résultats dans ce rapport, la qualité de nos échantillons va être analysée de façon générale, en détaillant les cas spécifiques.

Un bon score de qualité a été obtenu pour l'ensemble des échantillons, tous les boxplots étant centrés dans la partie verte du graphique (28 à 40) (Figure 10). Comme prévu, la qualité des échantillons décroît vers la fin des reads, ceci étant inhérent au séquençage à haut débit illumina, la qualité du séquençage se détériorant au fur et à mesure de l'exécution des cycles de séquençage.

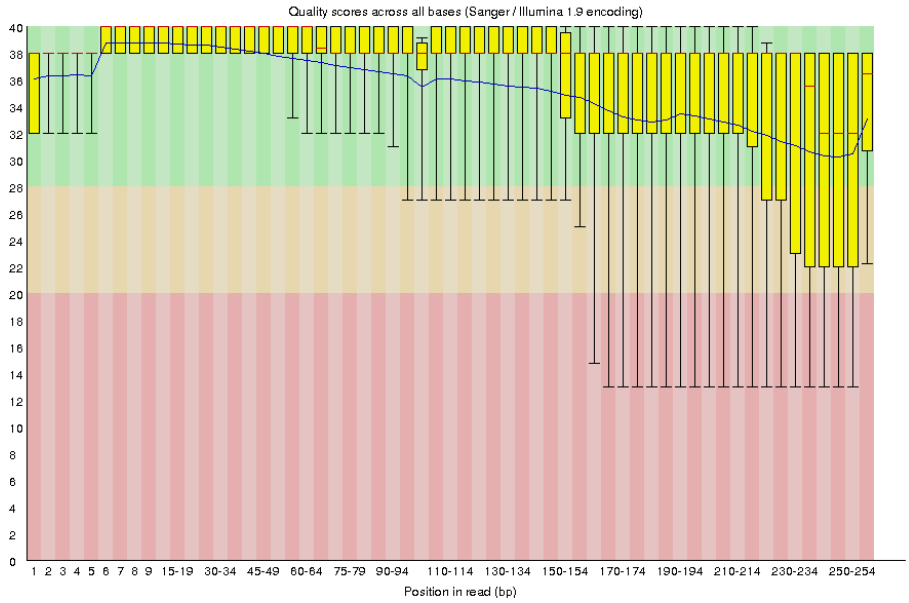

Figure 10 : Sortie fastQC de *Per base sequence quality* du Sa_CS_F_16_2_II_R.

La bonne qualité des séquences dans *Per base sequence quality* a également été vérifiée montrant que la qualité de toutes les séquences est supérieure ou égale à la qualité moyenne (données non présentées).

La répartition des bases de l'ADN parmi nos séquences a été vérifiée. Les résultats obtenus avec *Per base sequence content,* montrent une répartition quasi homogène des quatre bases de l'ADN. Ces résultats sont moins homogènes pour les échantillons issus de salive, mais restent néanmoins dans des proportions de 20-30% (Figure 11a). Dans la sortie de *Per sequence GC content,* les résultats ne suivent

pas une distribution normale (Figure 11b). Cela n'est pas, pour l'instant, un critère de mauvaise qualité de nos données car on travaille avec des données qui représentent des génomes différents qui n'ont pas forcément une répartition homogène connue de leurs bases.

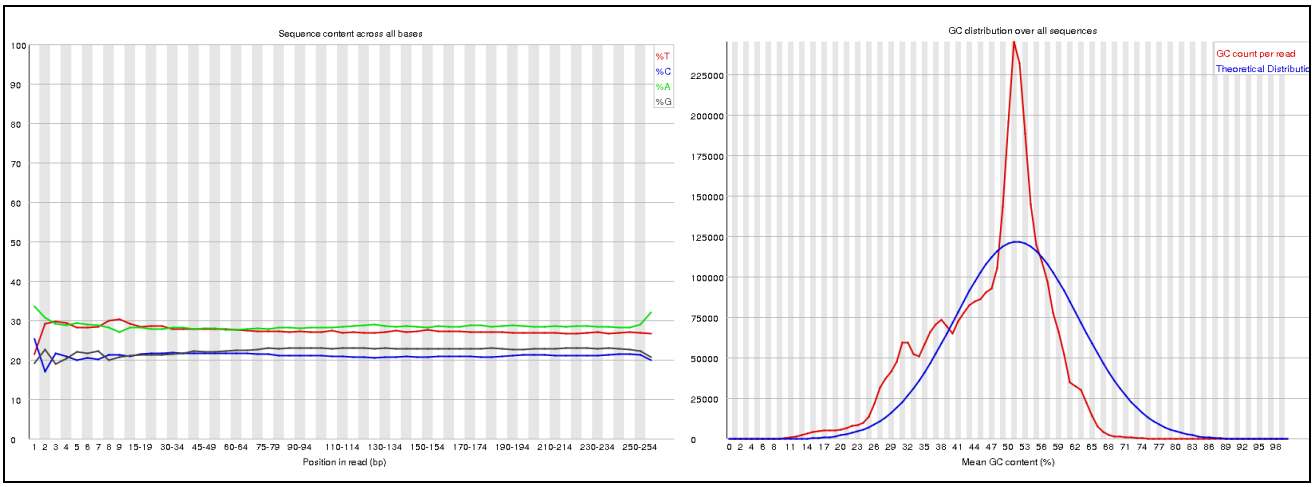

Figure 11 : a)Sortie fastQC de *Per base sequence content* de Sa_CS_F_16_1_II_R. b)Sortie fastQC de *Per sequence GC content* du Cl_AM_M_16_1_I_R.

D'après les sorties de *Per base N content,* il n'y a quasiment pas de N (base indéterminée) dans nos séquences, pour l'ensemble des échantillons.

La répartition de la longueur des séquences est homogène entre les différents échantillons. Pour la plupart des séquences, une longueur supérieure à 240 bases est observée (Figure 12).

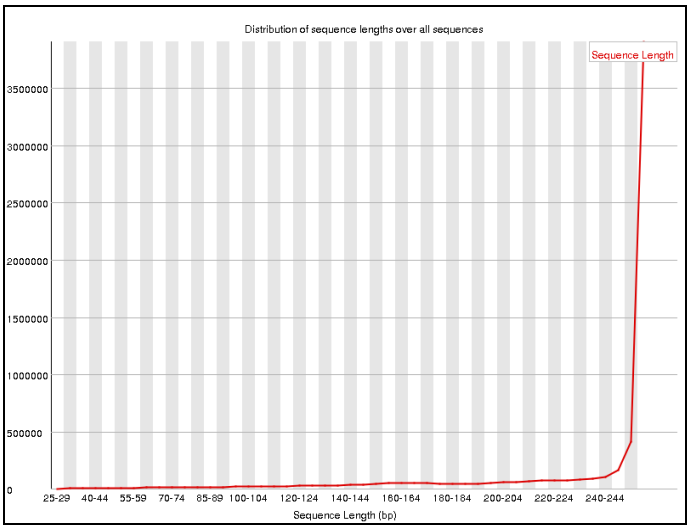

Figure 12 : Sortie fastQC de *Sequence Length Distribution* du Sa_AM_F_16_2_II_R.

Dans la Sortie *Sequence Duplication Levels*, des Warnings et Failure sont présents dans la plupart des échantillons ce qui indique un fort taux de duplication des reads (Figure 13). Ce résultat est probablement dû à une amplification préférentielle de certains matériels génétiques lors de l'étape de WGA ou WTA.

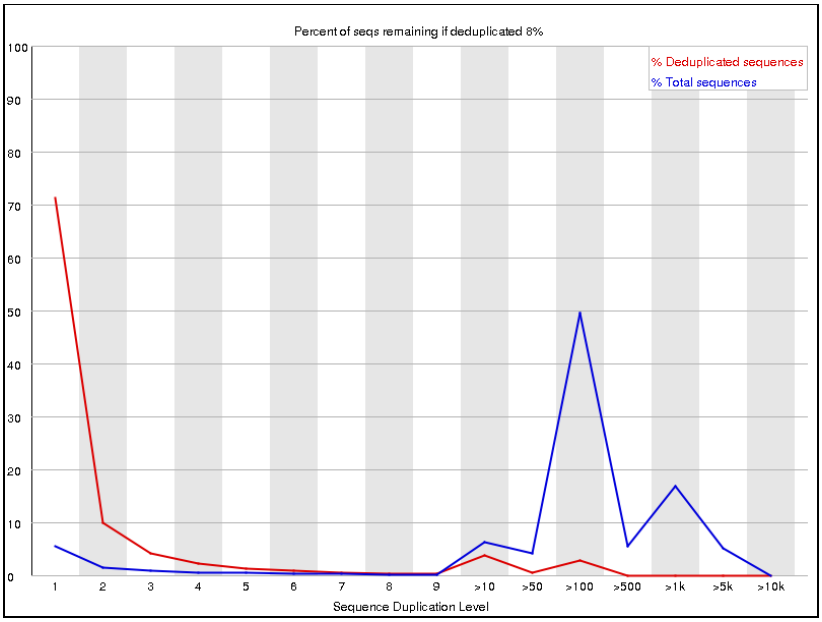

Figure 13 : Sortie fastQC de *Sequence Duplication Levels* du Cl_CS_M_18_2_F.

Pour finir, aucun échantillon ne contenait d'adaptateurs pour les reads R1 (lecture forward). En revanche, des adaptateurs (Illumina Universal Adapter) étaient uniquement retrouvés dans les reads R2 (lecture reverse) avec, selon les échantillons, un pourcentage allant de 5 à 20% des reads (Figure 14).

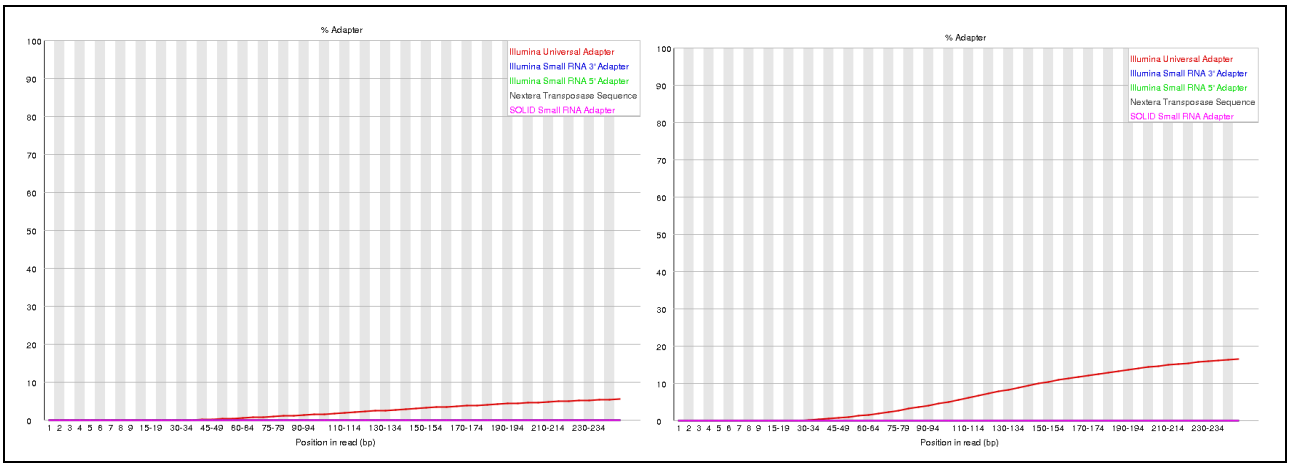

Figure 14 : Sortie fastQC de *Adapter Content* du Cl_AM_M_16_2_II_R et Sa_CS_F_16_1_II_R.

3.2 Filtrage des données

I. FaQCs

Après nettoyage par FaQCs on retrouve, dans les QC stats, les nombres de séquences et de bases qui ont été éliminées, leur proportion et la raison de leur suppression.

La plupart des adaptateurs a été coupée pour tous les reads 2 de l'ensemble des échantillons. Néanmoins, il reste une proportion de séquences, pour laquelle les adaptateurs n'ont pas été éliminés, allant de 2% à 9% du total des séquences et ce en fonction des échantillons.

Le nombre de séquences qui a été éliminé suite au contrôle qualité est de l'ordre de 2%, pourcentage qui peut varier en fonction des échantillons : 1,8% des séquences ont été éliminées pour ne conserver que celles d'une longueur supérieure à 50 pb, et 0, 2% pour ne garder que celles qui ne contiennent pas de base N. Nos données ne présentent pas d'autres problèmes de qualité parmi les critères analysés par FaQCs.

Les proportions de GC n'ont pas été modifiées suite au contrôle qualité FaQCs. Ainsi, on constate peu de changements de la qualité globale des séquences avant traitement et après traitement par FaQCs (Figure 15). Ce résultat est dû à la très bonne qualité initiale des séquences comme déjà observée dans les sorties des contrôles de qualité de Fastqc.

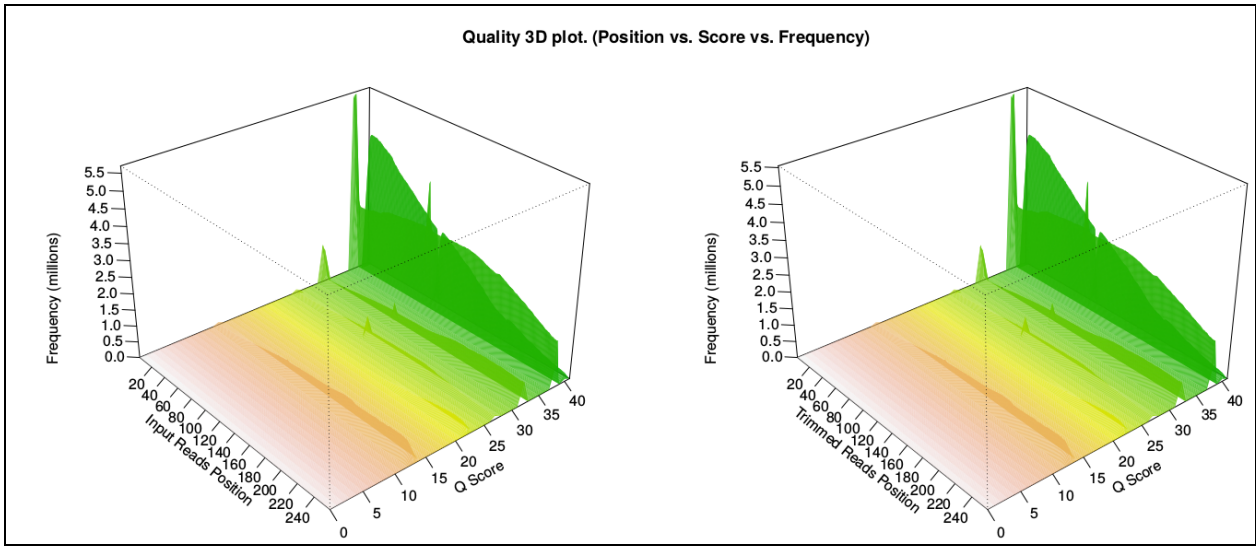

Figure 15: Sortie FaQCs de *Content Qu Per Cycle* du Cl_AM_F_16_1_II.

La seule partie où des changements significatifs peuvent être remarqués est la comparaison de la quantité de N par position entre les fichiers d'entrée et de sortie. Dans cette partie, une forte quantité de nucléotide N dans la partie centrale peut être observée (Figure 16). Il faut remarquer que cette densité de N n'avait pas été mise en évidence dans l'analyse de qualité faite par fastqc car leur proportion maximale représentait 0,0015% de bases N au total. Néanmoins, une majorité de base N a été éliminée par le filtrage FaQCs et seulement 0,0002% de N ont été conservés dans les positions 1-5.

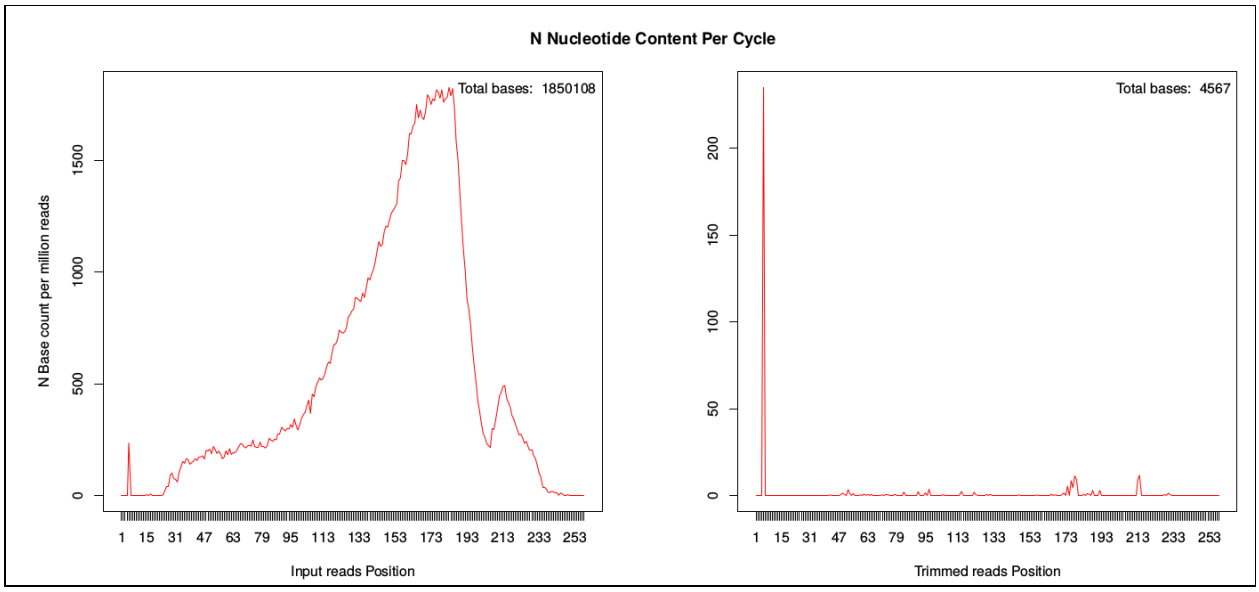

Figure 16: Sortie FaQCs de *N Nucleotide Content Per Cycle* du Cl_AM_F_16_1_II.

Pour vérifier le bon déroulement du nettoyage par faQCs et voir la qualité des jeux de données nettoyés, un contrôle de qualité fastQC a été refait. On observe de légères augmentations de la qualité globale, et ce plus particulièrement pour les résultats de la partie *adapter content* (Figure 17). Ainsi, on peut observer que les adaptateurs ont bien été éliminés.

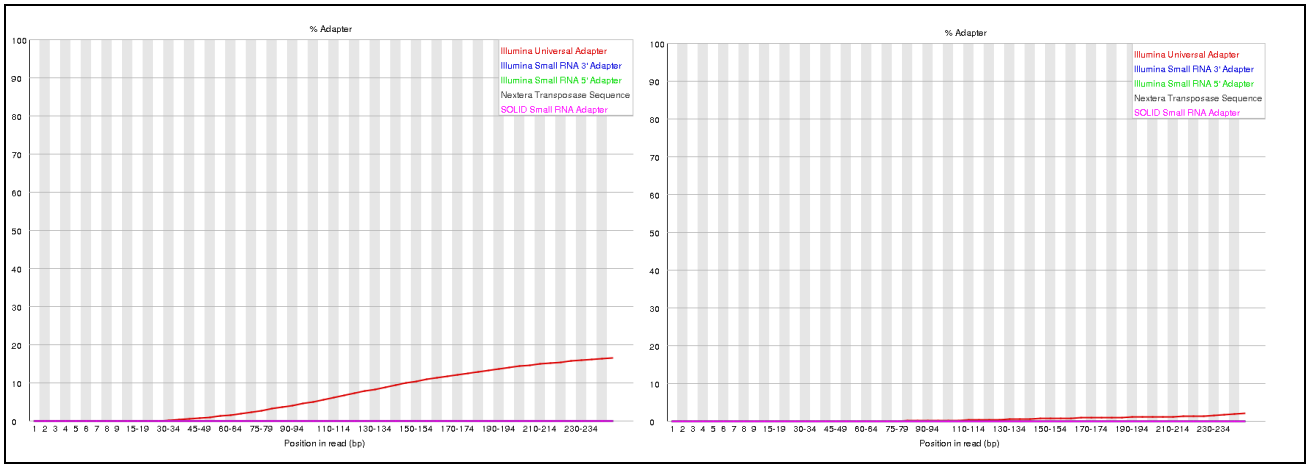

Figure 17: Sortie fastqc de *Adaptor Content* du Sa_CS_F_16_1_II_R avant et aprés faQCs.

Ainsi, tous les critères de qualité, après filtrage avec faQCs, ont été validés sauf le niveau de duplication des reads dans la partie *Sequence duplication levels.* Pour éliminer les duplicats, Fastq-mcf a été utilisé.

II. Fastq-mcf

Après analyse par Fastq-mcf, les séquences dupliquées ont été éliminées en totalité (Figure 18). Entre 9 et 23% des séquences du jeu de données en sortie de faQCs ont ainsi été conservés (fichier de sortie obtenu : woDUP).

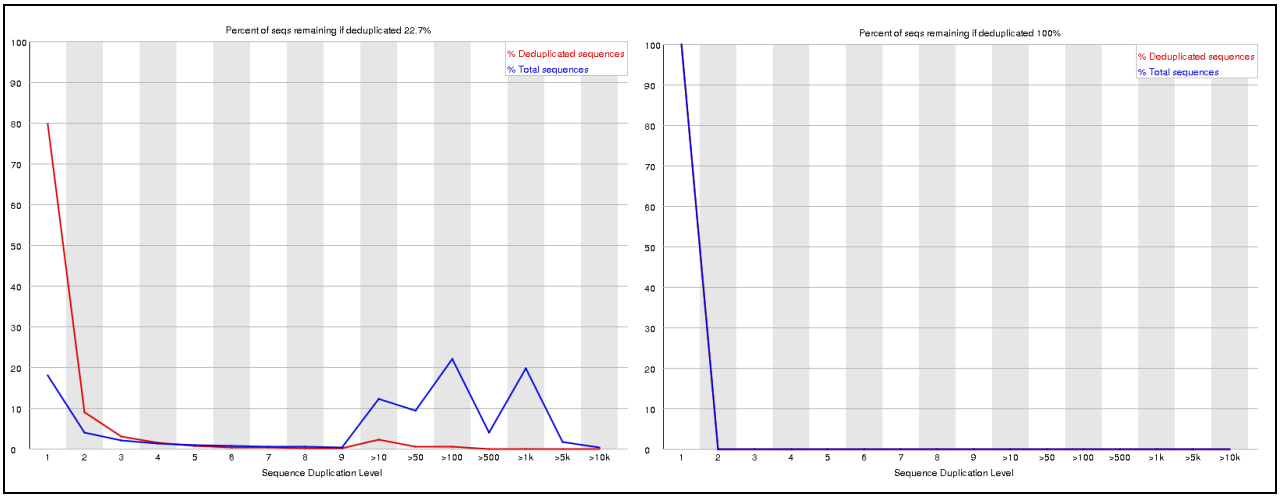

Figure 18: Sortie fastQC de *Sequence duplication levels* du Cl_AM_F_16_2_II_R avant et après le filtrage fastq-mcf.

III. Bbnorm

A partir des données issues de FaQCs, on utilise l'outil Bbnorm qui normalise les données et élimine également les duplicats.

Le warning de duplication des reads est levé pour les données de sortie (NORM) après Bbnorm. Cependant, il existe une petite zone de duplicats subsistant (Figure 19). Cela est dû au fait que Bbnorm garde un niveau de profondeur par défaut qui ici est de l'ordre de 3 à 4 copies maximum.

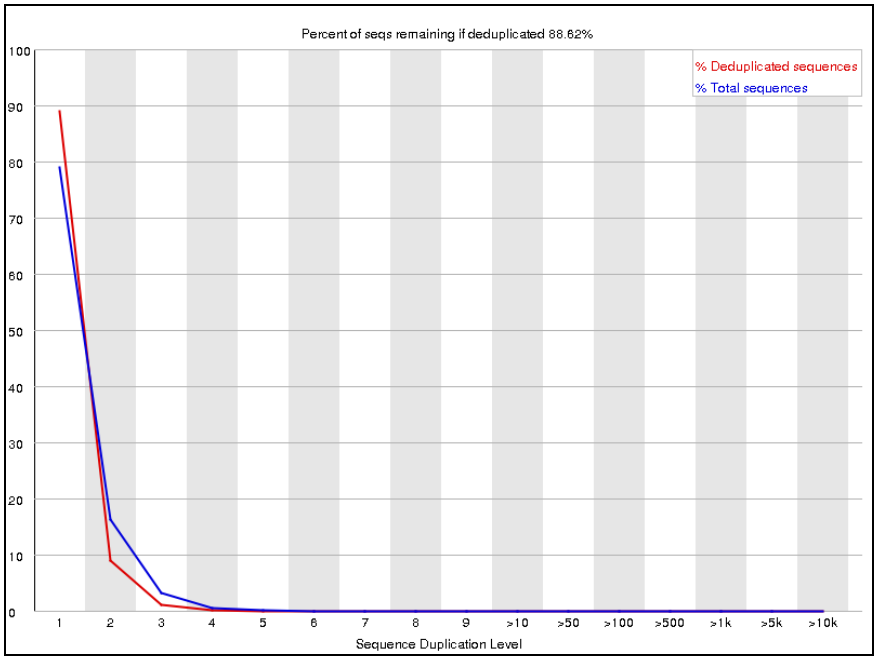

Figure 19: *Duplication level de l'échantillon CI_AM_F_16_I.*

IV. Comparaison des filtres utilisés

Une comparaison des différents résultats, après filtrage, en nombre de reads pour chaque échantillon a été réalisée (Tableau S3 en Annexe). Les reads dupliquées (DUP) ont été considérés comme point de départ pour pouvoir comparer les deux autres filtres (woDUP, NORM). Après filtrage des reads dupliqués (fastq-mcf - woDUP), il reste entre 2 et 12% des reads par rapport aux reads après faQCs (DUP) alors qu'après le filtrage de normalisation (Bbnorm - NORM), il reste entre 3 et 22% des reads selon les échantillons. En conclusion, la déduplication des reads par fastq-mcf est plus stricte comparée à celle de Bbnorm.

3.2 Assemblage

Trois fichiers de contigs (issus des données DUP, woDUP et NORM) ont été obtenus pour chaque pool après assemblage par megahit ainsi que des fichiers texte pour les différentes statistiques de chaque assemblage. Une comparaison de ces statistiques a été réalisée entre les contigs résultants des filtres woDUP et NORM avec DUP (Tableau S4 en Annexes).

Les résultats suggèrent que les assemblages des échantillons dupliqués (DUP) ont un nombre de contigs plus important et une longueur totale supérieure aux autres assemblages (woDUP et NORM), NORM étant l'assemblage présentant le plus petit nombre de contigs (Tableau 1). Par contre, les contigs issus de DUP ont une longueur médiane individuelle inférieure (Tableau 1).

Tableau 1 : *résumé des comparaisons des statistiques des contigs obtenus après assemblage (Megahit). Les différentes couleurs indiquent le jeu de données (woDUP en bleu, DUP en rose et NORM en gris).*

	#contigs	totalL	avgL	N50
plus grand	Dup	Dup	Norm	Norm
	woDup	woDup	woDup	woDup
plus petit	Norm	Norm	Dup	Dup

3.3 Choix du filtre

I. Mapping

Les reads de sortie de FaQCs ont été mappés sur chaque assemblage issu des trois filtres et ce afin de voir quel assemblage est le plus représentatif des données initiales. Cette représentativité s'appuie sur le pourcentage de reads mappés sur chaque assemblage. Pour cela on a utilisé les fichiers flagstat (Tableau S5 en Annexes).

Si on compare les résultats obtenus pour les données normalisées et dupliquées, les contigs de l'assemblage normalisé ont une meilleure performance pour le "properly paired". De façon générale, les contigs issus du filtrage de déduplication (woDUP) sont les moins représentatifs des reads initiaux car ils correspondent au mapping le moins élevé et pour la suite des analyses, l'assemblage woDUP a donc été rejeté.

II. BLAST Comparatifs pour Dup et NORM

Les contigs obtenus avec les reads dupliqués (Dup) ont été comparés aux contigs obtenus avec les reads normalisées (NORM). Pour cela les scripts de blast récursif ont été exécutés pour trois échantillons pour Dup et Norm. Les échantillons ont été choisis en fonction de leur différence de % de reads mappés sur les contigs entre Dup et NORM. L'échantillon avec la plus grande différence (Sa_AM_F_16_I), la plus petite différence (Cl_CS_M_16), et celui avec le valeur de différence médiane (Cl_AM_M_16_I) ont été choisis.

Nous avons comparé la diversité virale obtenue pour chaque échantillon pour DUP et NORM. Seules les différences de diversité virale au niveau de la famille selon le jeu de données (Dup, Norm) ont été étudiées car la perte d'une famille virale dans nos résultats serait un manque critique d'exhaustivité.
Les résultats de sortie du script de BLAST récursif des contigs montrent que trois familles virales ne sont pas identifiées avec l'assemblage Norm (Tableau 2). Ce résultat suggère que si on utilise les contigs normalisés, il y a un risque d'être moins exhaustif en ce qui concerne la diversité virale et ce pour l'ensemble des échantillons.

Tableau 2 : *Différences de familles virales entre Dup et Norm. Les familles qui apparaissent dans chaque colonne sont celles qui se trouvent uniquement dans le type de contigs (Dup ou Norm) de la colonne correspondante.*

CONTIGS	
Dup	Norm
Baculoviridae	
Lavidaviridae	
Polydnaviridae	

Si l'on prend en compte le temps d'exécution des différents blasts pour les contigs (voir Tableau 3), les contigs de Dup prennent **30 heures et 26 minutes** de plus que les contigs de Norm.

Tableau 3: Comparaisons des temps d'exécution des Blasts selon les filtres .

	DUP	NORM
Cl_AM_M_16_II_bn	6m3s	03m31s
Cl_AM_M_16_II_bn2	6h13m04s	1h40m17s
Cl_AM_M_16_II_bx	23h09m02s	5h12m01s
Cl_CS_M_16_bn	06m54s	05m19s
Cl_CS_M_16_bn2	3h14m45s	1h31m44s
Cl_CS_M_16_bx	4h28m38s	1h59m14s
Sa_AM_F_16_bn	05m02s	02m44s
Sa_AM_F_16_bn2	2h25m52s	35m35s
Sa_AM_F_16_bx	3h32m46s	1h13m43s

Après avoir analysé les résultats des BLASTs, nous avons choisi d'utiliser les contigs du filtrage dupliqué, car, même si cela prend plus de temps, la perte de diversité virale ne peut pas être négligée, surtout si ce résultat apparaît jusqu'au niveau des familles virales. La suite des analyses a donc été réalisée avec les **contigs dupliqués.**

3.4 Déchimérisation et BLAST

I. Tests des paramètres de blast

Les BLASTs ont été exécutés avec les différentes combinaisons des paramètres pour le fichier fasta avec 1000 séquences. Les résultats résumés des performances comparées des différents paramètres sont présentés Tableau 4. Les cinq points de comparaison, en fonction des paramètres de BLAST sont les suivants :

- Diversité virale : nombre d'espèces virales différentes obtenu dans l'échantillon.
- Nombre de fragments viraux : fragments de contigs assignés à des virus.
- Taux de faux positifs : pourcentage des virus trouvés qui sont des faux positifs.
- Diversité sans faux positifs : nombre d'espèces virales différentes obtenues dans l'échantillon une fois que les faux positifs sont enlevés.
- Temps d'exécution du script.

Lorsque l'on utilise une longueur minimale de 10 et une valeur de e-value élevée (10), on observe une diversité et un nombre de fragments viraux élevés (Tableau 4). Ces valeurs diminuent lorsque l'on utilise des longueurs plus grandes (100, 200) ou bien des valeurs de e-value plus faibles. Ce résultat semble cohérent avec les résultats attendus, les paramètres les moins stricts (longueur=10 et e-value=10), retenant des hits avec des valeurs de faible fiabilité quant à leur assignation lors du BLAST.

Un résultat similaire à celui observé pour la diversité virale est également obtenu pour le taux de faux positifs (Tableau 4). Pour les paramètres les moins stricts, on observe des valeurs de faux positifs plus élevées et pour les valeurs de paramètres plus strictes des taux plus bas. Ces résultats sont en accord avec les résultats attendus, les paramètres moins stricts impliquant une probabilité plus importante d'obtenir des faux positifs.

Lorsqu'on enlève les faux positifs, la diversité virale reste plus élevée avec les valeurs les moins strictes. Néanmoins, l'écart de diversité virale entre valeurs tolérantes et strictes est réduit. Cela montre qu'avec des valeurs restrictives, on obtient des faux négatifs.

Pour finir, le temps d'exécution est très variable selon les paramètres (de quelques minutes jusqu'à environ 7h). Le temps d'exécution augmente avec la diminution de la longueur minimale. Cela est dû au fait que plus la longueur est petite, plus il y a des séquences à recycler et plus le temps d'exécution est long. Par ailleurs, l'exécution devient plus lente pour des valeurs d'e-value différentes de la valeur par défaut (qui est 10). En effet, pour les autres valeurs de e-value, une étape de filtrage va être rajoutée à l'exécution de blast.

Tableau 4: Comparaison des performances des paramètres de BLAST. La diversité virale est exprimée ici en nombre d'espèce virale .

	Diversité virale	Nb. de fragments viraux	Taux de faux positifs	Diversité sans faux positifs	Temps d'execution
Long/E-value	10				
10	129	359	55.99%	31	4h00m38s
100	20	66	6.45%	17	1h43m34s
200	14	42	0.00%	14	1h21m14s
	1e-2				
10	40	206	12.14%	31	8h14m49s
100	27	101	7.92%	22	4h18m52s
200	15	66	6.06%	14	3h49m46s
	1e-9				
10	28	121	10.74%	22	6h57m06s
100	28	111	8.11%	22	4h18m14s
200	17	61	1.64%	16	4h15m40s

Jusque là, la diversité virale a tenu compte de tout le chemin taxonomique (jusqu'à l'espèce). Une fois les faux positifs éliminés, les résultats de détection au niveau de la famille virale ont été comparés pour chaque paramètre (Tableau 5).

Tableau 5 : *Différences de familles entre longueur avec e-value constante (haut) et e-value avec longueur constante (bas). Les familles qui apparaissent dans chaque colonne sont celles qui se trouvent uniquement avec la e-values de la colonne correspondante.*

10			**1e-2**			**1e-9**	
200	100	10	200	100	10	200	100+10
	Pycoviridae	*Pycoviridae* *Anelloviridae*		*Phycoviridae* *Anelloviridae*	*Phycoviridae* *Anelloviridae* *Marseilleviridae*		*Phycoviridae* *Anelloviridae*

10			**100**		**200**	
1e-9	1e-2	10	1e-9 + 1e-2	10		
	Marseilleviridae		*Anelloviridae*		=	

Pour faire le choix entre les différents paramètres, un tableau d'aide au choix a été réalisé (Tableau 6) où un score a été donné à chaque combinaison de paramètres en fonction de leurs performances par rapport aux facteurs suivants : **taux de faux positifs**, **diversité virale** et **temps d'exécution**. Les scores vont de un à cinq, cinq étant le poid le plus important. Par exemple, le score 5 a été donné pour le temps et le taux de faux positifs avec la combinaison de paramètres de e-value de 10 et de longueur 200 car c'est lui qui a les meilleures performances dans les deux facteurs. Les scores de diversité ont été donnés par rapport aux gains/pertes des familles virales.

Tableau 6 : *Tableau d'aide aux choix par poids des facteurs.*

	Diversite (0.7)	faux positives (0.2)	Temps (0.1)	SCORE
10_10	4	1	4	3.4
10_100	3	4	5	3.4
10_200	2	5	5	2.9
1e-2_10	**5**	**2**	**1**	**4**
1e-2_100	4	3	3	3.7
1e-2_200	2	3	4	2.4
1e-9_10	4	2	2	3.4
1e-9_100	4	3	3	3.7
1e-9_200	2	5	3	2.7

Pour finir, un pourcentage d'importance a été assigné à chaque facteur allant de 0,7 pour la diversité (donc le facteur le plus important) jusqu'à 0,1 pour le temps. Avec ces poids d'importance, un score finale a été calculé pour chaque combinaison de paramètres de la façon suivante :

$$\text{Score finale} = \sum (\text{score facteur} * \text{poids facteur})$$

Formule 1: *Calcul de score pour les combinaisons des paramètres.*

La combinaison avec le score final le plus élevé est celle avec une **longueur minimale de 10** et une **e-value de 1e-2.** Cette combinaison de paramètres a donc été utilisée pour la suite des analyses.

II.Résultats des viromes

En se basant sur les résultats les plus significatifs du script de déchimérisation, 39 familles virales connues ont été identifiées. Ces familles se répartissent de façon différente en fonction des espèces, des prélèvements, du sexe et de l'année. La répartition de présence/absence des familles virales est représentée dans la Figure 20. Parmi ces familles virales, sept sont présentes dans l'échantillon correspondant au contrôle négatif et dans tous les échantillons. En conséquence, leur présence dans les échantillons peut être soumise à caution.

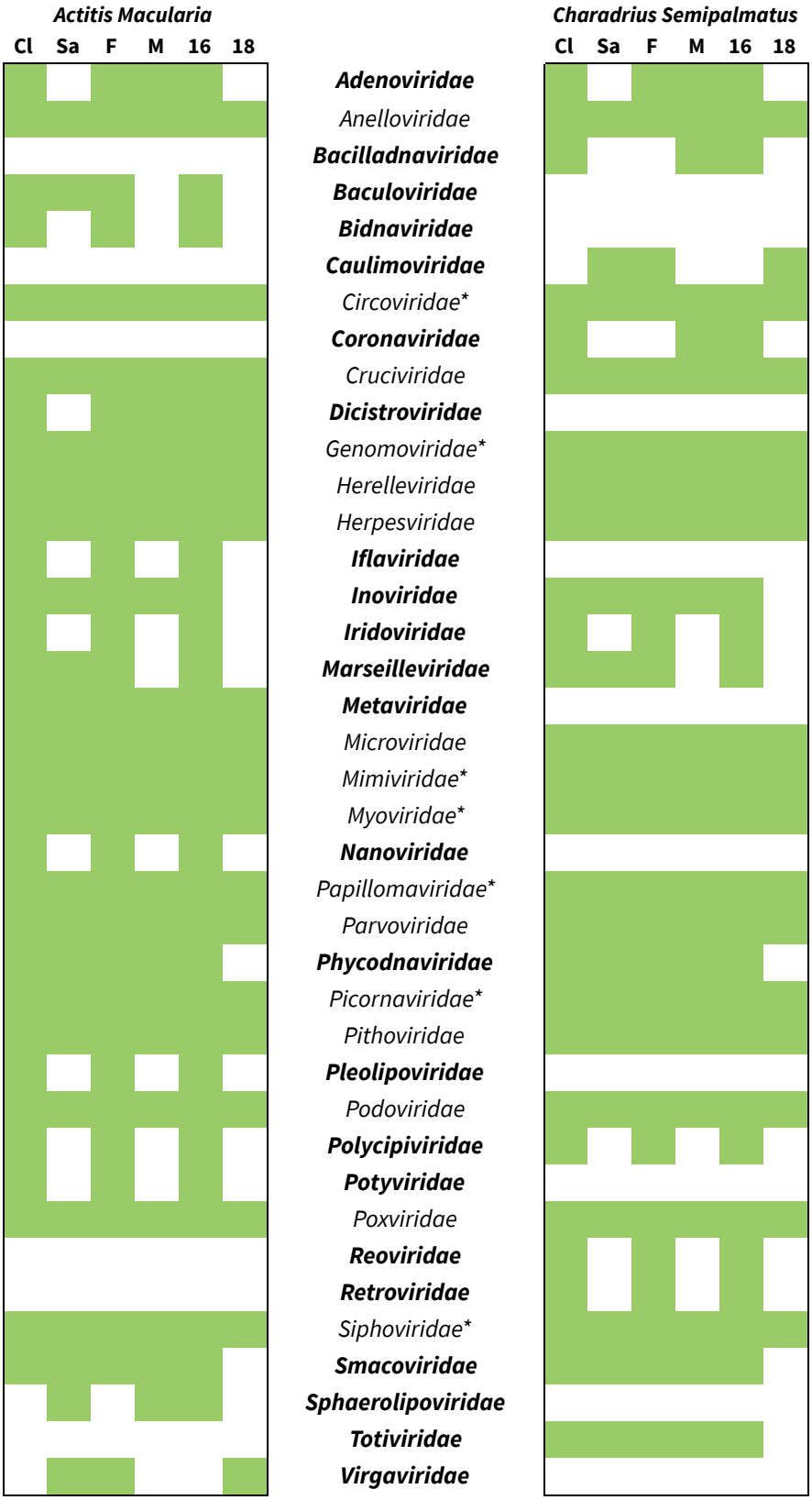

Figure 20 *: Présence (en vert) - absence (en blanc) de contigs associés aux familles virales en fonction des types d'échantillons. Les familles sont marquées avec * si elles sont présentes dans le contrôle négatif.*

Les familles virales ont été classifiées en fonction de leurs hôtes supposés (plantes ou bactéries pour exemple) et ce en fonction des types d'échantillons, des espèces, du sexe et de l'année de collecte (Figure 21). La répartition des types de familles virales (par hôte) est quasi similaire dans toutes les dimensions de comparaisons. Les familles virales qui affectent les vertébrés sont les plus présentes et semblent être les familles qui sont hébergées par les oiseaux. Ensuite, les familles virales infectant des invertébrés, plantes, champignons, plancton, amibes et algues sont possiblement liées à l'alimentation des oiseaux en milieu marin car ils cherchent leur nourriture dans la boue ou le sable des plages et rivages. D'autre part, les familles virales hébergées par des bactéries et archaea peuvent être liées à l'environnement de prélèvement et/ou du laboratoire.

La diversité virale est plus élevée dans les prélèvements de cloaques comparé à la salive au sein de chaque espèce. Ces résultats sont cohérents avec le type de prélèvements analysé. En effet, les prélèvements cloacaux font l'objet d'une accumulation plus importante de diversité microbienne qui peut refléter la diversité du régime alimentaire des oiseaux en comparaison des virus présents dans la salive.

Il y a également plus de familles virales trouvées dans les échantillons des femelles comparés aux mâles. La même différence est observée entre l'année 2016 et 2018, 2016 étant la période de collecte avec la diversité la plus élevée. Ce résultat est associé à un nombre de femelles plus important dans les pools ainsi qu'un effectif plus important en 2016, comparé à 2018.

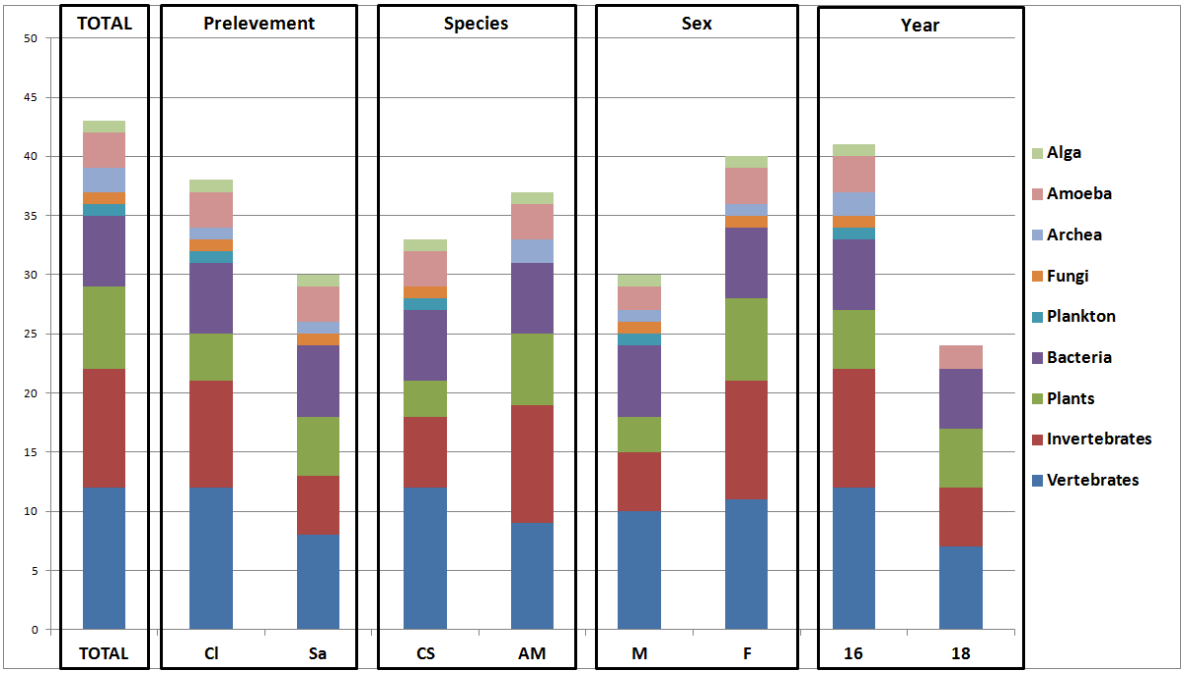

Figure 21 *: Répartition des hôtes supposés des familles virales en fonction des types de prélèvements, des espèces, du sexe et de la période de collecte.*

Partie II: Visualisation

Les résultats de la partie visualisation sont divers fichiers HTML liés entre eux. La visualisation est organisée sur trois niveaux différents pour chaque échantillon analysé : (1) la visualisation des résultats des taxonomies virales en forme d'arbres, (2) la visualisation des alignements des contigs aux gènes viraux retrouvés et (3) les fichiers fasta des alignements.

3.5 HTML Arbre taxonomique

Le premier niveau de la visualisation est donc l'arbre taxonomique. Cet arbre montre les taxonomies jusqu'à la famille virale des gènes viraux retrouvés dans chaque échantillon. Une visualisation (séries de fichiers "html") différente existe pour chaque échantillon.

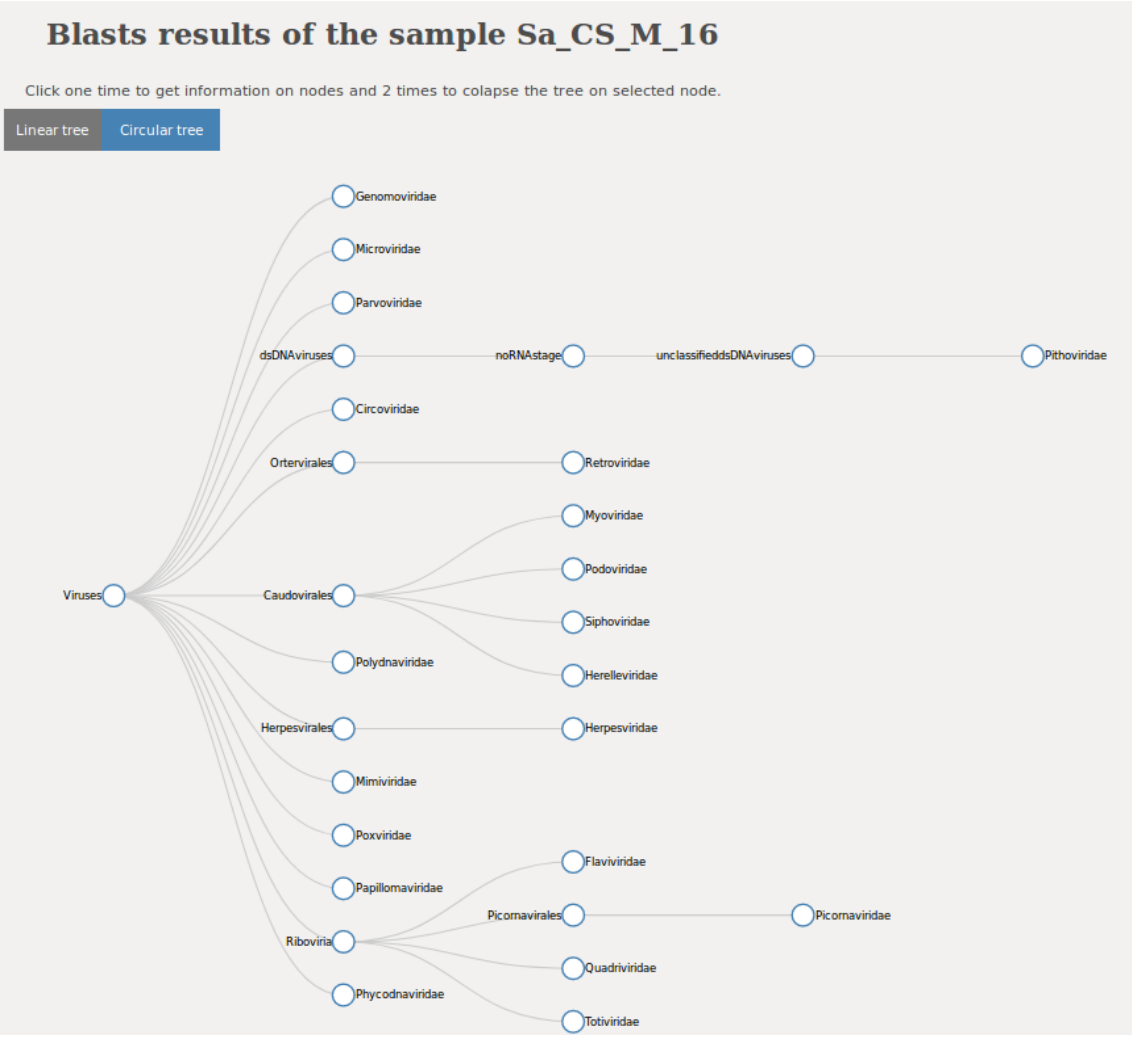

Figure 22 *: Arbre taxonomique (fichier html) des familles virales de l'échantillon Sa_Cs_M_16 en mode linéaire.*

Afin de rendre la visualisation plus souple, un option qui laisse le choix du type d'arbre à l'utilisateur est implémentée. L'arbre peut être linéaire (Figure 22), par défaut, ou circulaire (Figure 23).

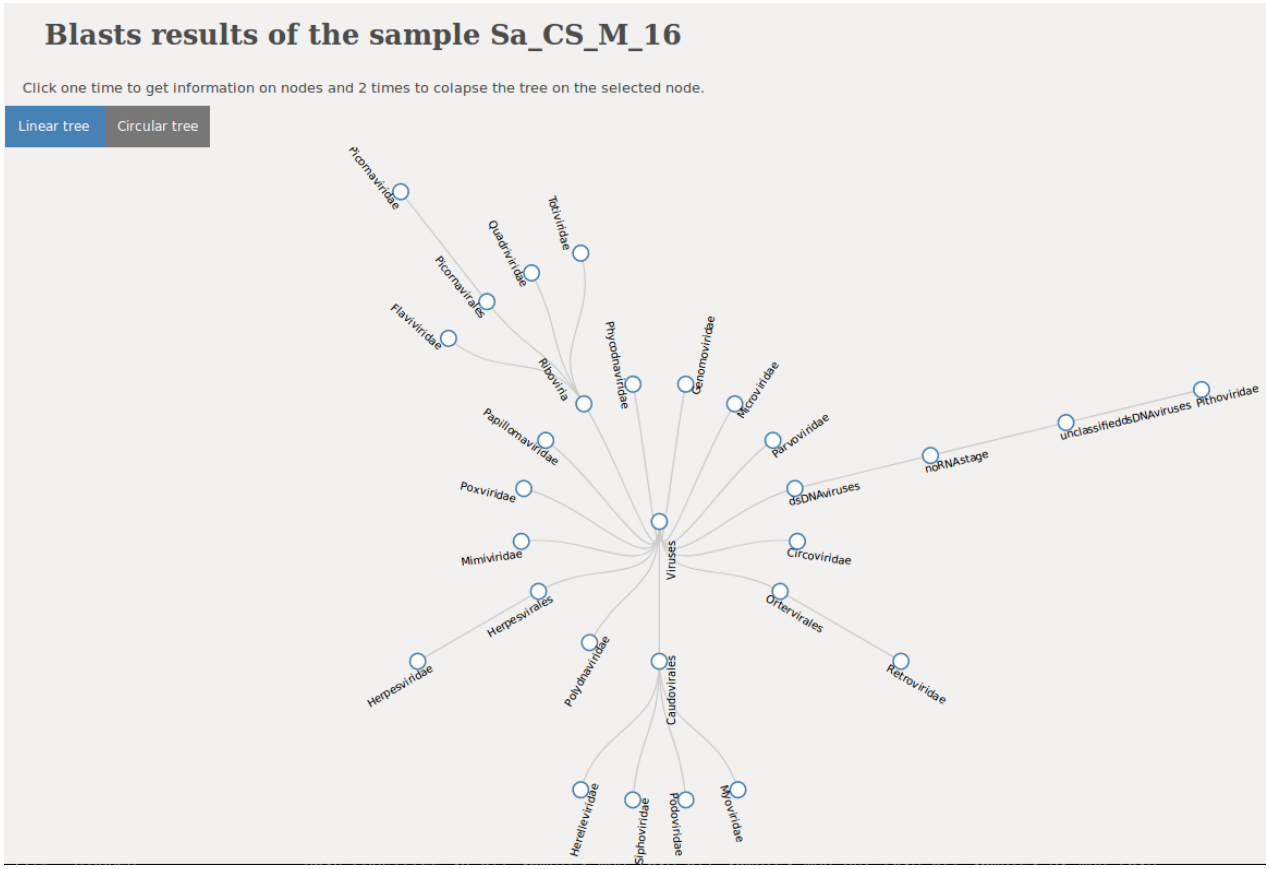

Figure 23 : *.Arbre taxonomique (fichier html) des familles virales de l'échantillon Sa_Cs_M_16 en mode linéaire.*

Les nœuds intermédiaires sont cliquables et lorsqu'on les sélectionne tous les nœuds fils vont disparaître, et lorsqu'on le re-clique ils vont réapparaître. Cela permet de pouvoir réduire l'information qui ne nous intéresse pas ou bien de clarifier la visualisation lorsqu'on a trop d'informations. La couleur du nœud intermédiaire réduit va devenir bleu clair au lieu de la couleur blanche pour nous rappeler qu'il y a des informations qui y sont cachées.

Finalement, quand on clique sur les nœuds terminaux, qui représentent toujours les familles virales, une nouvelle fenêtre s'ouvre avec le deuxième niveau HTML d'alignement. Cette nouvelle fenêtre va correspondre à la famille virale du noeud qui a été sélectionné. Il y a donc donc un fichier HTML lié à chaque nœud terminal.

3.6 HTML alignement gène - contigs

Le deuxième niveau de la visualisation correspond à la famille virale qui a été sélectionnée. Il existe donc des HTML d'alignements pour chaque famille virale présente dans chaque échantillon.

Cette partie contient des groupes de lignes rangés dans des clusters. Les lignes noires représentent les gènes identifiés avec leur accession number associé. Lorsque l'utilisateur clique sur la ligne noire qui représente les gènes, une fenêtre informative apparaît sur le côté gauche de l'écran (Figure 24). Dans cette fenêtre, il y a l'accession number comme titre accompagné de la longueur et des informations du gène en question. Si jamais ces informations ne sont pas retrouvées le gène va être représenté en gris au lieu de noir.

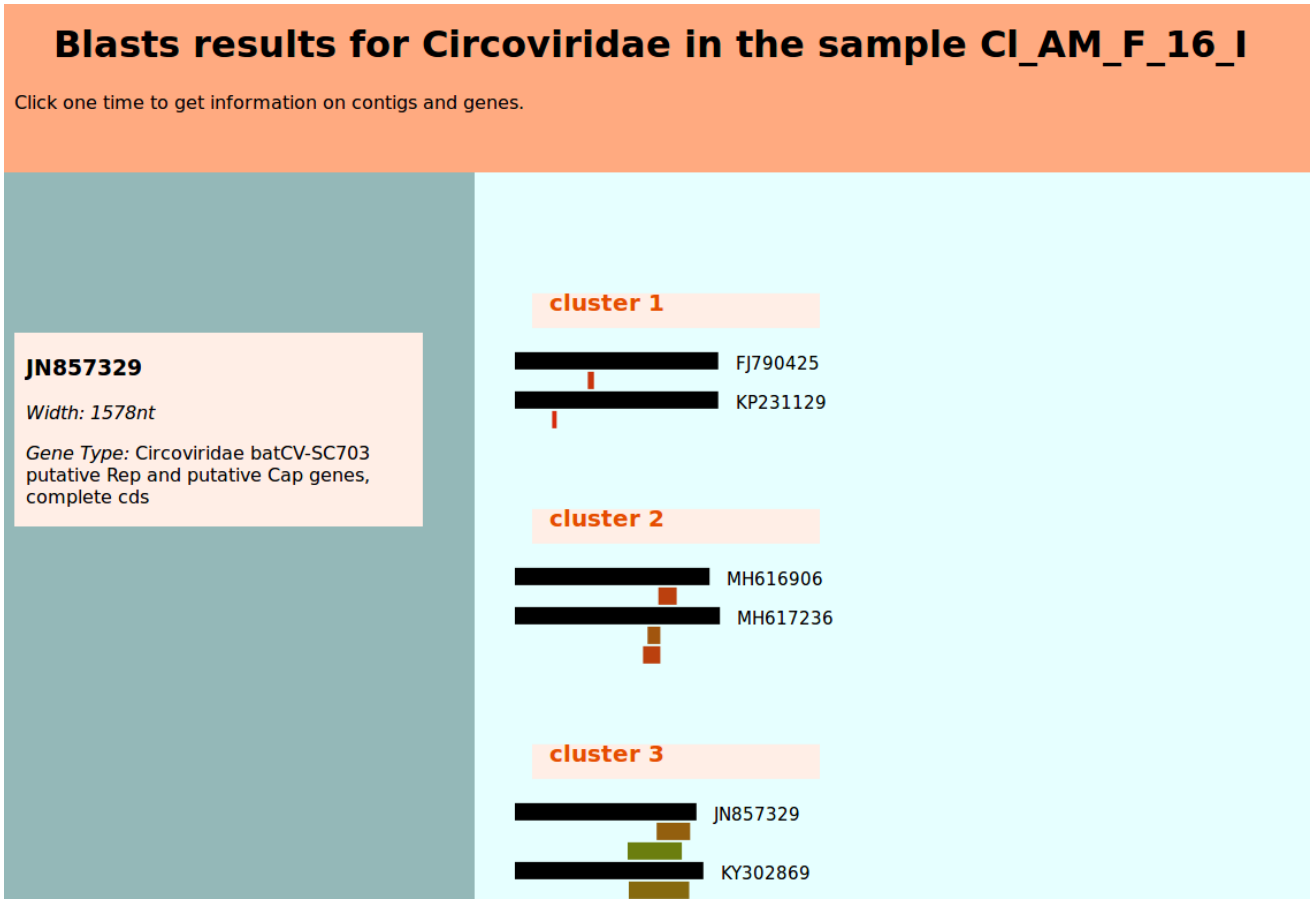

Figure 24 *: HTML d'alignement de la famille virale Circoviridae de l'échantillon Cl_AM_F_16_I.*

Sous chaque gène trouvé, les contigs qui lui sont associés sont représentés en forme de lignes de longueurs et couleurs variées (Figure 24). Ces lignes sont positionnées dans la zone où les contigs sont alignés sur le gène. La longueur de la ligne représente celle de l'alignement entre contig et gène. Les

couleurs des contigs correspondent à un gradient allant du vert au rouge selon la valeur de la e-value de cet alignement (confiance). De cette façon, les contigs qui sont représentés en couleur vert clair seront ceux avec une e-value très faible et inversement, ceux qui sont représentés en rouge seront ceux avec une e-value élevée associée à une confiance du résultat de BLAST faible.

Lorsque l'utilisateur clique sur les contigs une autre fenêtre informative similaire à celle pour les gènes va s'afficher aussi sur la gauche de l'écran. Cette fenêtre comprend les informations suivantes, nom du contig en question, les positions du début et de fin d'alignement sur le gène, la longueur de l'alignement, la e-value et le type de blast qui a permis cet alignement (blastx ou blastn). Les positions du début et fin d'alignement sont indiquées en acides aminés si elles sont retrouvées en blastx, par contre dans la visualisation les contigs sont représentés en nucléotides car, comme expliqué dans la partie des méthodes, les sorties en protéines passent par un script qui cherche leur gène source et c'est lui qui est représenté sur la visualisation. Les contigs sont aussi représentés en nucléotides.

Le dernier niveau de la représentation apparaît lorsqu'on clique sur les clusters (Figure 25). Cela donne la possibilité d'ouvrir une fenêtre qui contient les séquences "fasta" des gènes et des contigs présents dans le cluster en question. Les séquences des contigs dans le fichier fasta correspondent seulement aux parties des contigs qui ont été alignées.

Figure 25 : *Exemple de fichier FASTA proposé lorsqu'on clique un cluster.*

4. CONCLUSION

L'objectif principal de ce stage était de réaliser une analyse bioinformatique des viromes de deux espèces d'oiseaux migratrices. Pour cette analyse, deux tests ont été réalisés afin d'améliorer les paramètres du pipeline utilisés au laboratoire.

- L'effet de la déduplication sur la capacité de l'assemblage à représenter les reads initiaux ainsi que la diversité virale qui en découle,
- L'effet des paramètres de longueur et e-value sur les résultats de viromes.

Après avoir réalisé les contrôles de qualité, diverses possibilités de filtrages existaient. Le premier test réalisé a été la comparaison de ces méthodes de filtrage afin de choisir celui le plus adapté à notre jeu de données. En se basant sur la maximisation des résultats viraux, le filtre choisi a été l'élimination des adaptateurs sans normaliser ni dédupliquer les données. Néanmoins, l'outil de normalisation a été utilisé avec ses paramètres par défaut. Ce paramétrage reste à être testé pour vérifier ses performances.

La deuxième étape testée était le script récursif de blast. Des tests, sur les paramètres du BLAST, ont été réalisés afin de voir quels étaient ceux qui s'adaptaient le mieux au projet. Pour cela, les différents tests ont été résumés dans un tableau de poids. Ce tableau nous a permis, pour ce projet, d'établir que les paramètres les mieux adaptés sont une longueur de 10 et une e-value de 0,01. Par ailleurs, ce tableau pourra être réutilisé pour les autres projets dans le laboratoire, et en changeant les poids en fonction des besoins du projet, les paramètres les plus adaptés vont être choisis automatiquement.

A partir de la sortie de ce pipeline, les viromes des oiseaux ont été analysés. Trente neuf familles virales connues ont été identifiées et ont été classifiées en fonction de leurs hôtes. Une répartition en fonction des types d'échantillons a également été réalisée. Les échantillons de cloaques montrent une diversité virale plus importante. On a pu observer que le virome est différent selon l'espèce d'oiseau étudié et reflète leur alimentation ainsi que leur habitat.

Nous n'avons pas identifier des fragments viraux proches de pathogènes connus pour être hébergés chez les oiseaux. Néanmoins, les viromes caractérisés pour les échantillons des années 2016 et 2018 permettent déjà d'identifier des cohortes virales qui semblent ubiquitaires et pérennes chez les oiseaux (que ce soit entre espèce ou au cours du temps). De nouveaux jeux de données permettront de consolider ces résultats mais également d'identifier la part variable du virome en fonction des espèces et au cours du temps.

Parallèlement à l'exécution et les tests du pipeline, des outils dédiés à la visualisation des données “viromics” ont été développés. Avec ces outils, la visualisation des résultats du pipeline devient plus accessible et interactive. De plus, la nouvelle visualisation facilite le travail du laboratoire car les séquences homologues et les contigs correspondants sont classés dans des clusters. Cette visualisation permet aux biologistes d’accéder à des ensembles de séquences homologues au format fasta, base de départ des analyses en aval. De plus, cet outil pourra être utilisé pour les futures projets de viromes dans le laboratoire.

L’outil de visualisation est conçu pour représenter les échantillons individuellement et indépendamment. Pour améliorer cet outil, la suite sera de pouvoir visuellement comparer la diversité virale entre échantillons selon le sexe, l’année, l'espèce ou le type de prélèvement (métadonnées).

Du point de vue personnel, ce stage a été très enrichissant pour moi, car il m’a permis de découvrir le monde de la recherche et plus concrètement les études bioinformatiques appliquées à des métagénomes. Il m’a permis d'apprendre et appliquer les differents étapes du pipeline informatique. J’ai également appris depuis zéro la programmation en HTML et javaScript. Ce stage m’a aussi permis d’apprendre à travailler de façon indépendante et de gérer mes responsabilités et tâches.

5. REMERCIEMENTS

Je voudrai d’abord remercier mon encadrant dans l’Institut, Anne Lavergne de m’avoir fait confiance et de m’avoir montré le côté biologique de ce projet. Je voudrais également remercier Sourakhata Tirera de m’avoir donner l’occasion de travailler dans un projet de bioinformatique et m’avoir aidé nombreuses fois.

Enfin, je voudrais remercier les enseignants de l’option BioSTIC de l’Ecole Centrale de Nantes: Sophie Limou, Olivier Roux, Aurélien Serandour et Domenico Borzacchiello pour m'avoir fait découvrir le monde de la génétique et de la bioinformatique, qui est devenu probablement le domaine de mes futurs emplois.

BIBLIOGRAPHIE

[1] Taylor LH, Latham SM, Woolhouse ME. (2011) Risk factors for human disease emergence. Phil Trans R Soc Lond B Biol Sci.

[2] Hansen-Chaffard, E. 2000. Peuplement des oiseaux d'eau du littoral guyanais cas particulier des limicoles. Ecole pratique des hautes études, sciences de la vie et de la terre.

[3] Ondov B.D., Bergman N.H., Phillippy A.M. (2015) Krona: Interactive Metagenomic Visualization in a Web Browser. In: Nelson K.E. (eds) Encyclopedia of Metagenomics. Springer, Boston, MA

[4] Thrash A, Arick M, Barbato R, Jones R, Douglas TA, Esdale J, Perkins EJ, Reyero-Garcia N. (2019) Keanu: a novel visualization tool to explore biodiversity in metagenomes *BMC Bioinformatics*: 12859–20-S2-S12.

[5] Braham Bioinformatics. IN Simon Andrews. *Documentation de fastqc* [en ligne] [consulté avril 2019] Disponible sur: https://www.bioinformatics.babraham.ac.uk/projects/fastqc/

[6] Chienchi Lo, PatrickS.G. Chain (2014) Rapid evaluation and Quality Control of Next Generation Sequencing Data with FaQCs. BMC Bioinformatics.

[7] Github. In Chienchi Lo, PatrickS.G. Chain. *FaQCs: Quality Control of Next Generation Sequencing Data* [en ligne] [Consulté avril 2019] Disponible sur: https://github.com/LANL-Bioinformatics/FaQCs

[8] Erik Aronesty (2011). *ea-utils* . Command-line tools for processing biological sequencing data.

[9] Github. In Erik Aronesty. *Documentation de fastqMcf.* [en linge] [consulté avril 2019] Disponible sur: https://github.com/ExpressionAnalysis/ea-utils/blob/wiki/FastqMcf.md

[10] Schatz MC, Delcher AL, Salzberg SL. (2010) Assembly of large genomes using second-generation sequencing. Genome Res 2010, 20:1165–1173.

[11] Jason R.Miller, Sergey Koren, Granger Sutton. (2010). Assembly algorithms for next-generation sequencing data. Genomics.

[12] Li, D., Liu, C. M., Luo, R., Sadakane, K., & Lam, T. W. (2015). MEGAHIT: An ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph. Bioinformatics.

[13] Li H. and Durbin R. (2009) Fast and accurate short read alignment with Burrows-Wheeler Transform. Bioinformatics.

[14] Li H., Handsaker B., Wysoker A., Fennell T., Ruan J., Homer N., Marth G., Abecasis G., Durbin R. and 1000 Genome Project Data Processing Subgroup (2009) The Sequence alignment/map (SAM) format and SAMtools. Bioinformatics.

[15] Li H (2011) A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. Bioinformatics.

[16] Erik Aronesty (2013). TOBioiJ : "Comparison of Sequencing Utility Programs", DOI:10.2174/1875036201307010001

[17] Bradley RD, Hillis DM. Recombinant DNA sequences generated by PCR amplification. Mol Biol Evol. 1997;14:592–593.

[18] Altschul S. F., Gish W., Miller W., Myers E. W., Lipman D. J. (1990). Basic local alignment search tool. *J. Mol. Biol.* 215 403–410. 10.1016/S0022-2836(05)80360-2

[19] Mark Johnson, Irena Zaretskaya, Yan Raytselis, Yuri Merezhuk, Scott McGinnis, Thomas L. Madden, NCBI BLAST: a better web interface, *Nucleic Acids Research*, Volume 36, Issue suppl_2, 1 July 2008, Pages W5–W9.

[20] NCBI. *Nucleotide database*. [en ligne] [consulté Mai 2019] Disponible sur: ftp://ftp.ncbi.nih.gov/blast/db/nt.00.tar.gz

[21] NCBI. *Non-Redundant database*. [en ligne] [consulté Mai 2019] Disponible sur: ftp://ftp.ncbi.nih.gov/blast/db/nr.00.tar.gz

[22] Metagenomics. In Matthias Scholz. *E-value & Bit-score* [en ligne] [consulté Juin 2019] Disponible sur: http://www.metagenomics.wiki/tools/blast/evalue

[23] Peter Cooper, Melissa Landrum, Ilene Mizrachi, and Jane Weisemann. *Entrez Sequences Quick Start*. [en linge] [consulté Juillet 2019] Disponible sur: https://www.ncbi.nlm.nih.gov/books/NBK44863/

ANNEXES

Tableau S1 : Description des échantillons analysés. throat swabs of *Actitis macularia* (AM) and *Charadrius semipalmatus* (CS).

Le type d'échantillon (cloaque ou trachée), l'espèce (*Actitis macularia* (AM) et *Charadrius semipalmatus* (CS)) , le sexe, les années de captures sont indiqués ainsi que le nombre d'individus par pool. Le nombre de reads obtenus après séquençage est indiqué pour chaque pool.

* Échantillons négatifs poolés au séquençage.

Type	**Pool**	**Species**	**Sex**	**Session**	**n**	**Nombre de reads**
Cloaca	1	C. semipalmatus	Female	2016	14	12 383 819
	2	C. semipalmatus	Female	2016	15	7 108 553
	3	C. semipalmatus	Female	2018	6	11 797 584
	4	C. semipalmatus	Male	2016	4	8 885 761
	5	C. semipalmatus	Male	2018	3	11 048 666
	6	A. macularia	Female	2016	20	15 052 811
	7	A. macularia	Female	2016	20	13 852 746
	8	A. macularia	Female	2018	24	9 761 842
	9	A. macularia	Male	2016	15	10 072 544
	10	A. macularia	Male	2016	15	14 264 499
	11	A. macularia	Male	2018	19	13 301 860
	12*	Negative Control	-	-	1 ml DMEM	18 635 714
	13	C. semipalmatus	Female	2016	14	16 046 943
Saliva	14	C. semipalmatus	Female	2016	15	16 744 127
	15	C. semipalmatus	Female	2018	6	14 415 730
	16	C. semipalmatus	Male	2016	4	11 181 921
	17	C. semipalmatus	Male	2018	3	14 587 793
	18	A. macularia	Female	2016	20	13 116 600
	19	A. macularia	Female	2016	20	10 642 106
	20	A. macularia	Female	2018	24	19 118 098
	21	A. macularia	Male	2016	15	13 643 295
	22	A. macularia	Male	2016	15	18 636 244
	23	A. macularia	Male	2018	19	17 188 355
	24*	Negative Control	-	-	1 ml DMEM	poolé avec 12
					TOTAL READS	311 487 611

Liste L1 : sorties de l'outil de contrôle de qualité FastQC

- Basic Statistics : informations sur le fichier analysé : le nom, le type de fichier, le type de séquençage et le numéro de séquences processées, ainsi que la longueur de la plus courte et de la plus longue des séquences et le pourcentage de bases GC sur toutes les séquences.

- Per base sequence quality : répartition des valeurs de qualité sur toutes les bases à chaque position. Trois régions se trouvent dans ce graphique en fonction de la valeur de qualité : vert pour une bonne qualité (28-40), orange pour une qualité moyenne (20-28) et rouge pour une qualité faible (0-20), 40 représentant la qualité maximale.
  Pour chaque position, il y a un boxplot qui a : des lignes rouges centrale qui représentent la médiane et une bleue pour la médiane, interquartile (25-75%) représenté par la boite jaune et les moustaches représentant le 10 et 90% des points.

- Per tile sequence quality : écart par rapport à la qualité moyenne de chaque séquence. Les couleurs sont soit dans l'échelle des bleus soit dans l'échelle des rouges. Les bleus représentent une qualité égale ou supérieure à la moyenne et les rouges des valeurs inférieures à la moyenne.

- Per sequence quality score : distribution de la qualité parmi les séquences.

- Per base sequence content : proportion de chacune des quatre bases d'ADN dans chaque position.

- Per sequence GC content : comparaison du contenu en GC par rapport à une distribution normale modélisée du contenu en GC.

- Per base N content : quand le séquenceur est incapable d'interpréter une base avec suffisamment de confiance, il la substituera par un N. Dans ce graphique, la quantité de N par position peut être observé.

- Sequence Length Distribution : distribution des tailles des séquences dans le fichier analysé.

- Sequence Duplication Levels : représentation du degré de duplication pour chaque séquence. En bleu, la ligne du % de toutes les séquences et en rouge le % des séquences dupliquées.

- Overrepresented sequences : liste des séquences surreprésentées dans un tableau avec le nombre de fois où elles ont été identifiées, le pourcentage, et une possible cause pour leur répétabilité.

- Adapter Content : Distribution de la présence des adaptateurs dans les séquences.

Liste L2 : sorties de l'outil de filtrage FaQCs.

- QC stats : cette partie permet de voir les différences basiques entre les fichiers d'entrée et de sortie, ainsi que des informations sur des fichiers qui ont été rejetés. Ici se trouvent le nombre de séquences, le nombre de bases, et la longueur des séquences en entrée et en sortie.
  Pour les séquences et les bases rejetées, nous en avons le pourcentage et la raison pour laquelle elles ont été rejetées ou réduites (longueur coupée, base continue "N", faible ratio de complexité, qualité ou présence des adaptateurs).

- Reads Length Histogram : histogrammes des tailles des séquences à l'entrée et la sortie.

- Reads GC content : comparaison du contenu en GC, dans les séquences, représentée en histogramme, ainsi que des histogrammes pour le contenu des quatre différentes bases de l'ADN.

- Nucleotide Content Per Cycle : comparaison de la proportion de chacune des quatre bases d'ADN à chaque position entre les fichiers d'entrée et de sortie.

- N Nucleotide Content Per Cycle : comparaison de la quantité de N par position entre les fichiers d'entrée et de sortie.

- Reads Average Quality Histogram : Histogramme de qualité des séquences avec le pourcentage des reads qui ont une qualité supérieure à 20.

- Quality Boxplot Per Cycle : comparaison entre la répartition des valeurs de qualité sur toutes les bases représentée par des boxplots à chaque position.

- Quality 3D plot. (Position vs. Score vs. Frequency) : représentation 3D de la fréquence des différents scores de qualité en fonction de leur position.

- Quality report : comparaison entre la répartition de qualité représentée en diagramme de barres.

Tableau S2 : résumé des scripts pour la visualisation

Nom du Script	Informations
Tree.sh	**Langage** : Bash, hml, css et javascript **Objectifs** : - Ecrire dans un fichier html le code pour l'arbre d'un échantillon. - Lancé tous les autres scripts. **Fichiers entrée** : Trois fichiers résultants du blastn, n2 et x. **Fichiers sortie** : Fichier html de l'arbre d'un échantillon.
cluster-accesssions.sh	**Langage :** bash **Objectifs:** Création des clusters des gènes similaires pour chaque famille virale d'un échantillon. **Fichiers entrée** : Résultats des blastn et blastx mélangés d'un échantillon. **Fichiers sortie :** - Fichier avec la liste d'accession numbers de chaque cluster d'une famille virale. - Fichier avec la liste d'accession numbers qui ne sont pas dans les clusters pour chaque famille virale d'un échantillon.
blastXtoN.sh	**Langage :** bash **Objectifs :** Traduit les accession numbers des gènes correspondant aux protéines du cluster et leurs positions pour chaque cluster protéique en nucléotides. **Fichiers entrée** : - Fichier avec la liste d'accession numbers d'un cluster en protéines - Fichier avec la liste d'accession numbers des gènes correspondant aux protéines du fichier d'entrée. **Fichiers sortie :** Fichier avec la liste des index des protéines dans le gène.
Contigs.sh	**Langage :** Bash, hml, css et javascript **Objectifs :** Pour chaque famille virale d'un échantillon : - Obtenir le data dans un format json avec les informations des gènes et des contigs. - Ecrire dans un fichier html le code pour obtenir la visualisation des contigs et clusters.

	Fichiers entrée : Résultats de blastn et blastx mélangés d'un échantillon. **Fichiers sortie :** Fichier html de de chaque famille virale d'un échantillon.
get-contig.py	**Langage :** python **Objectifs :** Coupé les contigs pour en n'avoir que la partie qui colle avec le gène viral. **Fichiers entrée** : - Nom du contig. - Séquence fasta du contig. **Fichiers sortie :** - Nom du contig avec les nouveaux index. - Partie de la séquence fasta qui colle avec le gène viral.
jsontree.py	**Langage**: python **Objectif:** Obtenir un fichier de format json avec les informations des taxonomies d'un échantillon. **Fichiers entrée** : Résultats de blastn et blastx mélangés d'un échantillon **Fichiers sortie :** Arbre de la taxonomie en format de jsontree.
circulartree.sh	**Langage** : Bash, hml, css et javascript **Objectif :** Écrire dans un fichier html le code pour l'option d'arbre circulaire d'un échantillon. **Fichiers entrée** : Arbre de la taxonomie en format de jsontree. **Fichiers sortie :**Fichier html de l'arbre circulaire d'un échantillon.

Tableau S3 : comparaisons des résultats obtenus après filtrage des reads.

Nombre de reads par échantillon et par filtre (DUP, woDUP, NORM). Les trois premières colonnes indiquent le nombre de reads pour les trois jeux de données (DUP, woDUP, NORM), les trois suivantes indiquent les comparaisons de ces nombres de reads entre les trois filtres données en pourcentage.

	DUP	woDUP	NORM	DUP vs woDUP	DUP vs NORM	woDUP vs NORM
Cl_AM_F_16_I_1	7164466	328157	485514	4.58%	6.78%	2.20%
Cl_AM_F_16_I_2	7093967	326890	482184	4.61%	6.80%	2.19%
Cl_AM_F_16_II_1	6679333	845233	1468040	12.65%	21.98%	9.32%
Cl_AM_F_16_II_2	6600926	838850	1454648	12.71%	22.04%	9.33%
Cl_AM_F_18_1	4553844	309055	586466	6.79%	12.88%	6.09%
Cl_AM_F_18_2	4479397	306864	580814	6.85%	12.97%	6.12%
Cl_AM_M_16_I_1	4680114	183542	298040	3.92%	6.37%	2.45%
Cl_AM_M_16_I_2	4624208	182789	296424	3.95%	6.41%	2.46%
Cl_AM_M_16_II_1	6777911	288532	357802	4.26%	5.28%	1.02%
Cl_AM_M_16_II_2	6737006	288437	358590	4.28%	5.32%	1.04%
Cl_AM_M_18_1	6282543	273679	420876	4.36%	6.70%	2.34%
Cl_AM_M_18_2	6226422	272144	418732	4.37%	6.73%	2.35%
Cl_CS_F_16_I_1	5841468	366096	629490	6.27%	10.78%	4.51%
Cl_CS_F_16_I_2	5792191	365296	627392	6.31%	10.83%	4.52%
Cl_CS_F_16_II_1	3316401	285937	511180	8.62%	15.41%	6.79%
Cl_CS_F_16_II_2	3266319	283962	506496	8.69%	15.51%	6.81%
Cl_CS_F_18_1	5552121	159982	245364	2.88%	4.42%	1.54%
Cl_CS_F_18_2	5494143	159314	243990	2.90%	4.44%	1.54%
Cl_CS_M_16_1	4222479	274249	432852	6.49%	10.25%	3.76%
Cl_CS_M_16_2	4180455	272436	428748	6.52%	10.26%	3.74%
Cl_CS_M_18_1	5000024	87480	134372	1.75%	2.69%	0.94%
Cl_CS_M_18_2	4949064	87088	133674	1.76%	2.70%	0.94%
Neg_1	8142286	76474	105354	0.94%	1.29%	0.35%
Neg_2	8055852	76274	104058	0.95%	1.29%	0.34%
Sa_AM_F_16_I_1	6870796	131898	186390	1.92%	2.71%	0.79%
Sa_AM_F_16_I_2	6827130	132472	185798	1.94%	2.72%	0.78%
Sa_AM_F_16_II_1	6212556	539487	906662	8.68%	14.59%	5.91%
Sa_AM_F_16_II_2	6170564	536402	899774	8.69%	14.58%	5.89%
Sa_AM_F_18_1	4967659	181212	298348	3.65%	6.01%	2.36%
Sa_AM_F_18_2.fq	4919111	180312	295960	3.67%	6.02%	2.35%
Sa_AM_M_16_I_1	8928626	192540	286406	2.16%	3.21%	1.05%
Sa_AM_M_16_I_2	8889677	193316	286372	2.17%	3.22%	1.05%
Sa_AM_M_16_II_1	6423143	231148	338544	3.60%	5.27%	1.67%
Sa_AM_M_16_II_2	6364679	230189	335586	3.62%	5.27%	1.66%
Sa_AM_M_18_1	8755938	175141	275254	2.00%	3.14%	1.14%
Sa_AM_M_18_2	8718440	174506	274554	2.00%	3.15%	1.15%
Sa_CS_F_16_I_1	8735028	985909	1638906	11.29%	18.76%	7.48%
Sa_CS_F_16_I_2	8641452	981835	1633718	11.36%	18.91%	7.54%

Sa_CS_F_16_II_1	7557254	247227	438574	3.27%	5.80%	2.53%
Sa_CS_F_16_II_2	7459486	245544	435696	3.29%	5.84%	2.55%
Sa_CS_F_18_1	7824377	303113	544294	3.87%	6.96%	3.08%
Sa_CS_F_18_2	7747367	302070	538564	3.90%	6.95%	3.05%
Sa_CS_M_16_1	6799744	637914	1081640	9.38%	15.91%	6.53%
Sa_CS_M_16_2	6731586	635307	1077202	9.44%	16.00%	6.56%
Sa_CS_M_18_1	5159306	118697	198338	2.30%	3.84%	1.54%
Sa_CS_M_18_2	5116727	117661	195718	2.30%	3.83%	1.53%
moyenne				**5.04%**	**8.32%**	**3.28%**

Tableau S4 : Comparaisons des statistiques d'assemblage pour chaque filtre au filtre DUP pour chaque pool exprimées en pourcentage.

#contigs : nombre de contigs; total Lenght : longueur totale de l'assemblage; max Lenght : longueur du contig le plus long; Avg Length; longueur moyenne de contigs; N50 : longueur médiane des contigs.

Pools	#contigs		total Lenght		max Lenght		Avg Length		N50	
Samples	woDup/Dup	Norm/Dup	woDup/Dup	Norm/Dup	woDup/Dup	Norm/Dup	woDup/Dup	Nor/Dup	woDup/Dup	Norm/Dup
Cl_AM_F_16_I	59.95%	28.05%	80.17%	42.86%	100.00%	73.96%	133.68%	152.85%	181.89%	253.28%
Cl_AM_F_16_II	71.22%	41.88%	91.41%	59.63%	100.00%	100.00%	128.33%	142.39%	117.62%	134.87%
Cl_AM_F_18	87.05%	42.97%	89.31%	44.24%	100.05%	100.05%	102.54%	103.05%	101.62%	102.96%
Cl_AM_M_16_I	91.30%	45.12%	93.78%	45.98%	100.00%	89.72%	102.75%	101.96%	104.26%	100.85%
Cl_AM_M_16_II	35.95%	11.88%	62.15%	28.95%	100.00%	100.82%	173.01%	243.93%	329.77%	507.27%
Cl_AM_F_18	74.11%	32.07%	78.55%	36.44%	100.00%	66.33%	105.80%	113.46%	103.49%	110.47%
Cl_CS_F_16_I	86.06%	40.38%	91.87%	40.87%	100.00%	89.68%	106.90%	101.31%	110.24%	102.61%
Cl_CS_F_16_II	85.55%	45.26%	88.86%	46.43%	100.00%	100.00%	103.91%	102.69%	103.08%	102.31%
Cl_CS_F_18	75.98%	33.86%	91.44%	41.28%	100.00%	67.32%	120.35%	121.98%	125.36%	114.85%
Cl_CS_M_16	84.46%	44.74%	94.39%	54.88%	100.00%	98.35%	111.80%	122.68%	112.32%	125.97%
Cl_CS_M_18	76.12%	31.31%	87.65%	40.38%	100.00%	103.86%	115.10%	129.03%	141.06%	219.35%
Neg	42.56%	13.15%	53.68%	19.56%	106.61%	92.30%	126.18%	148.82%	122.81%	148.13%
Sa_AM_F_16_I	40.61%	13.21%	57.39%	19.45%	100.00%	100.00%	141.52%	147.34%	173.67%	199.70%
Sa_AM_F_16_II	67.30%	22.90%	90.07%	41.82%	100.00%	87.22%	133.82%	182.68%	133.87%	271.33%
Sa_AM_F_18	43.28%	24.90%	52.91%	32.44%	100.00%	100.00%	122.16%	130.11%	128.84%	136.99%
Sa_AM_M_16_i	81.30%	33.43%	85.74%	36.47%	79.30%	42.02%	105.48%	108.99%	105.84%	108.64%
Sa_AM_M_16_II	49.59%	18.49%	72.45%	30.73%	100.00%	119.22%	146.07%	166.18%	200.68%	352.99%
Sa_AM_F_18	90.60%	39.79%	89.18%	43.67%	100.00%	62.96%	98.21%	109.72%	94.43%	106.63%
Sa_CS_F_16_I	67.96%	30.19%	89.43%	49.24%	100.00%	99.39%	131.65%	163.19%	143.36%	256.87%
Sa_CS_F_16_II	61.67%	30.56%	71.15%	34.55%	100.00%	97.72%	115.13%	112.82%	115.11%	110.71%
Sa_CS_F_18	78.97%	37.49%	84.65%	40.43%	99.91%	71.16%	107.29%	107.76%	105.91%	104.37%
Sa_CS_M_16	89.09%	38.70%	96.87%	51.11%	94.81%	94.81%	108.76%	132.12%	108.44%	190.98%
Sa_CS_M_18	89.16%	36.65%	92.39%	37.22%	97.07%	100.02%	103.64%	101.56%	112.37%	106.86%

Tableau S5 : Statistiques de mapping des reads sur les trois assemblages des 23 échantillons.

Pourcentage total de reads mappés (*total*), proportion de reads mappés correctement en paire (*properly paired*).

	total			properly paired		
	dup	woDup	norm	dup	woDup	norm
Cl_AM_F_16_I	98.89%	97.40%	98.49%	72.80%	59.37%	83.08%
Cl_AM_F_16_II	99.80%	99.80%	99.58%	71.04%	73.54%	87.38%
Cl_AM_F_18	94.62%	87.87%	94.91%	73.50%	71.83%	85.77%
Cl_AM_M_16_I	94.33%	91.62%	97.17%	73.34%	74.00%	85.92%
Cl_AM_M_16_II	99.47%	99.57%	99.58%	76.61%	57.44%	85.15%
Cl_AM_M_18	95.61%	95.04%	97.90%	67.38%	55.55%	79.55%
Cl_CS_F_16_I	95.80%	93.13%	95.27%	79.74%	65.60%	78.27%
Cl_CS_F_16_II	96.07%	94.38%	95.83%	68.66%	75.39%	85.01%
Cl_CS_F_18	98.04%	97.73%	98.39%	80.37%	61.04%	90.35%
Cl_CS_M_16	97.90%	95.10%	99.34%	79.17%	59.95%	81.75%
Cl_CS_M_18	97.76%	97.05%	97.06%	84.21%	76.09%	86.98%
Neg	99.42%	98.06%	97.33%	76.57%	63.37%	92.26%
Sa_AM_F_16_II	98.83%	96.49%	93.21%	72.07%	68.28%	83.15%
Sa_AM_F_16_I	96.69%	93.06%	86.40%	72.91%	56.97%	82.74%
Sa_AM_F_18	98.11%	97.96%	89.54%	70.69%	75.53%	80.28%
Sa_AM_M_16_II	97.93%	95.04%	97.63%	67.30%	69.39%	80.60%
Sa_AM_M_16_I	96.87%	92.15%	98.50%	83.77%	61.60%	92.28%
Sa_AM_M_18	96.70%	94.10%	94.15%	46.14%	39.53%	89.33%
Sa_CS_F_16_II	91.92%	85.74%	94.61%	69.04%	69.72%	86.28%
Sa_CS_F_16_I	96.71%	94.87%	95.31%	77.89%	79.50%	88.71%
Sa_CS_F_18	95.73%	93.14%	95.53%	75.24%	70.25%	89.14%
Sa_CS_M_16	96.92%	95.39%	94.90%	80.88%	82.40%	87.03%
Sa_CS_M_18	98.20%	95.71%	96.03%	84.45%	60.99%	89.23%