MASTER SCIENCES
ET NUMERIQUE POUR LA SANTÉ

Master 2 Sciences et Numérique pour la Santé
Parcours Bioinformatique, Connaissances, Données

HMSN408 : Stage long BCD
Rapport de stage

Évaluation de la qualité et comparaison d'assemblages de génomes

---

Auteur :
Anna Tran

Encadrantes :
Sèverine Bérard
Anne-Muriel Arigon Chifolleau
Tuteur :
Alban Mancheron

Version du
15 octobre 2019

Remerciements

Je souhaiterais tout d'abord remercier mes encadrantes Sèverine Bérard et Anne-Muriel Arigon Chifolleau. Cette expérience a été très enrichissante, puisqu'elle m'a permis de mener un nouveau projet sous vos conseils avisés. Ce stage m'a permis de m'enrichir autant professionnellement que personnellement. Je vous suis infiniment reconnaissante d'avoir cru en moi et de m'avoir accompagner tout au long de cette année de Master 2.

J'aimerais également remercier mes collègues de Master. L'année ne fut pas de tout repos mais l'entraide et un groupe soudé furent la clé d'une réussite. Les conseils et le soutient de la promotion 2019 auront été d'une grande aide. Un grand merci également à mes camarades de bureau, les rires et les brownies m'ont apporté de la bonne humeur ! Et un immense merci à Valentin Klein pour la relecture de mes rapports depuis le master 1.

Table des matières

1 Introduction

1.1 Problématique

Depuis le progrès des technologies de séquençage, la reconstruction de génomes reste un des problèmes majeurs en bioinformatique. Elle est composé de différentes étapes qui permettent l'assemblage de génomes (Figure 1).

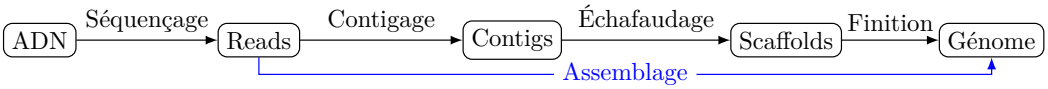

FIGURE 1 – Pipeline de reconstruction d'un génome

D'abord, le séquençage permet d'obtenir des *reads* (lectures) qui sont des séquences de taille variable (quelques centaines de paires de bases pour les *reads* courts et quelques kilobases pour les *reads* longs). Faisant suite au séquençage, le contigage permet l'obtention de contigs suite à la reconstruction de plus grands fragments à partir des *reads* grâce aux chevauchements entre ces derniers. Puis vient l'étape d'échafaudage qui consiste à ordonner et orienter les contigs afin de constituer des scaffolds. Enfin, une étape de finition permet de compléter le génome grâce à des traitements bioinformatiques supplémentaires ou à des résultats expérimentaux autour des zones d'incertitudes.

Dans la pratique, il est possible d'utiliser de multiples outils et d'appliquer des paramètres variés à ces derniers afin de reconstruire un génome. Ainsi, pour un même organisme, plusieurs assemblages peuvent être générés. Une des problématique qui en découle est l'évaluation de la qualité de ces assemblages et leur comparaison afin de choisir le meilleur en se basant sur différentes métriques (N50, %GC, nombre de gènes conservés, ...). Plusieurs compétitions comme l'Assemblathon et le Genome Assembly Gold-Standard Evaluation ont été menées afin d'évaluer différents assemblages obtenus à partir de divers outils disponibles. Cependant, la notion de "meilleur assemblage" n'est encore pas bien définie. De plus, le choix de l'outil le plus adapté aux données n'est pas trivial. Actuellement, il est impossible d'obtenir un assemblage complet qui ne contient aucune erreur [1].

Il existe actuellement des méthodes permettant de calculer des métriques relatives à la qualité des assemblages, notamment QUAST [2] ainsi que BUSCO [3] qui ont fait l'objet de mon rapport bibliographique [4]. Néanmoins, ces méthodes ne permettent pas de comparer instantanément différents assemblages ou d'apprécier la qualité globale de ces derniers.

Mon stage s'inscrit dans ce contexte d'évaluation de la qualité et de comparaison des assemblages. Il a été effectué au sein de l'ISEM (équipe Phylogénie et Évolution Moléculaire) et du LIRMM (équipe Méthodes et Algorithme pour la Bioinformatique) et encadré par Sèverine BÉRARD et Anne-Muriel ARIGON CHIFOLLEAU. L'objectif de ce stage était de permettre l'évaluation de la qualité des assemblages en se basant sur les métriques de qualité, la comparaison d'un ensemble d'assemblages pour un génome et la visualisation des résultats pour les utilisateurs.

1.2 Constitution des jeux de données

Afin de pouvoir mener les analyses, il a fallu dans un premier temps collecter des jeux de données. Différentes bases de données (European Nucleotide Archive et Nucleotide du NCBI) ont été parcourues ainsi que différentes bases de données spécifiques aux espèces. Dans la plupart des cas, il est difficile d'obtenir différentes versions d'un même assemblage puisque seule la dernière version est disponible. Nous avons pu récupérer deux jeux de données ; celui de l'oiseau blanc ainsi que celui de l'oursin.

1.2.1 Oiseau blanc (*Zosterops borbonicus*)

Pour le génome de l'oiseau blanc[5], les assemblages m'ont été fournis par l'équipe Phylogénie et évolution moléculaire de l'ISEM. L'oiseau blanc est un vertébré endémique de la Réunion. Il a été étudié par l'équipe dans le cadre de l'étude de l'évolution des chromosomes sexuels chez les *Aves*.

Le jeu de données est composé de 5 assemblages obtenus par différents moyens. Les données brutes sont de deux natures : certaines ont été obtenus suite à un séquençage Illumina permettant d'obtenir des *reads* courts *paired-end* et d'autres ont été obtenues suite à un séquençage Pacific Biosciences (Pacbio) permettant l'obtention de *reads* longs. Ces données ont été assemblées avec différents programmes, à savoir MaSuRCA[18] et SOAPdenovo[19].

La table 1 présente les principales métriques utilisées au cours du stage qui ont été calculées grâce à QUAST et BUSCO.

Assemblage	Nombre de contigs	Contig le plus long	Longueur totale	%GC	N50	N75	L50	L75	Nombre de N par 100 kpb	Nombre de gènes conservés
MaSuRCA	5 409	11 330 673	1 087 174 432	42.18	1 859 356	715 096	155	385	410.23	4 539
SOAP	130 278	2 861 829	1 157 532 025	41.63	488 989	234 226	682	1519	6592.06	4 616
SOAP_GC_rich	9 917	1 698 780	1 014 417 590	41.72	296 704	162 915	1063	2210	6096.76	4 143
SOAP_K27	99 405	2 903 302	1 261 302 657	41.40	505 574	243 071	731	1628	22062.57	4 340
SOAP_Pacbio	97 415	11 984 413	1 199 605 461	41.71	2 218 729	784 718	152	378	8473.40	4 657

TABLE 1 – Données calculées par QUAST et BUSCO pour l'oiseau blanc

1.2.2 Oursin (*Strongylocentrotus purpuratus*)

Le second jeu de données dont nous disposons est celui de l'oursin[6]. Ce dernier a pu être téléchargé à partir de l'EchinoBase. L'oursin est un échinoderme des côtes ouest américaine. Le génome de l'oursin a été séquence principalement en raison de l'utilisation de l'embryon comme organisme modèle de recherche pour la biologie moléculaire, évolutive et cellulaire.

Les assemblages ont été obtenus suite à différents séquençages :

— La version 0.5 a été obtenue par un *whole genome shotgun sequencing* en utilisant la technologie Sanger

— Pour la version 2.5, les contigs des versions précédentes ont été scaffoldées grâce aux données de séquençage SOLiD

— Pour la version 3.1, les données de séquençage Illumina ont été intégrées afin d'améliorer l'assemblage existant

— Pour la dernière version en date (4.2), les données de séquençage PacBio ont été incluses à l'assemblage existant.

La table 2 présente les principales métriques utilisées au cours du stage qui ont été calculées grâce à QUAST et BUSCO.

Assemblage	Nombre de contigs	Contig le plus long	Longueur totale	%GC	N50	N75	L50	L75	Nombre de N par 100 kpb	Nombre de gènes conservés
V0.5	187 943	962155	1 089 019 554	37.00	55852	17046	3398	12451	23 941.09	880
V2.1	114 222	1 466 901	902 885 471	36.99	125 100	22 996	1 513	6 151	10 756.09	884
V2.5	77 726	1 493 490	916 953 850	37.09	167 142	40 041	1 272	4 180	11 577.80	904
V3.1	32 008	2 423 607	935 002 910	37.00	403 385	162 113	661	1 572	12 901.38	917
V4.2	31 896	2 525 675	989 431 426	37.32	421 711	169 542	671	1 595	8 940.64	917

TABLE 2 – Données calculées par QUAST et BUSCO pour l'oursin

2 Étude de la corrélation des métriques de qualité des assemblages

Dans cette partie, nous souhaitons étudier la corrélation entre les différentes métriques qui ont été calculées grâce à QUAST[2] et BUSCO[3]. Au cours de mon projet bibliographique[4], les métriques avaient été classifiées en catégories. Néanmoins, aucun lien n'a été établi entre ces dernières. Cette étude permettra dans un premier temps de savoir s'il existe des corrélations, puis dans un second temps, de pondérer les métriques afin d'établir une fonction de score sans biais apportés par des informations redondantes. Pour cela, nous avons choisi deux tests statistiques : la corrélation de Spearman et l'analyse en composantes principales. Ces deux tests seront présentés dans les sous-sections suivantes.

2.1 Test de corrélation de Spearman

2.1.1 Principe

La corrélation permet de mettre en lumière un lien statistique entre deux variables X et Y, ce qui signifierait que l'une ne varie pas indépendamment de l'autre. Pour calculer le coefficient de corrélation et tester sa nullité, nous avons utilisé le test de corrélation de Spearman [7]. Il s'agit de la version non paramétrique du test de corrélation de Pearson qui permet de s'affranchir des conditions de normalité des distributions et d'homogénéité des variances indispensables à la fiabilité des tests paramétriques.

Le coefficient de corrélation de Spearman est calculé de la manière suivante :

$$r_s = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n^3 - n}$$

avec :

— n le nombre d'individus

— di^2 le carré de la différence entre les rangs de X_i et Y_i

Le coefficient de corrélation peut être compris entre -1 et 1. Plus la corrélation est proche de 1, plus les variables sont positivement corrélées. Inversement, lorsque la corrélation est proche de -1, plus les variables sont négativement corrélées.

Exemple Nous présentons ici un exemple d'application de la formule sur deux variables : la N50 et L50. La table 3 représente l'affectation des rangs pour les variables ainsi que le calcul de di.

Individu	1	2	3	4	5
N50, Yi	1 859 356	488 898	296 704	505 574	2 218 729
L50, Xi	155	682	1 063	731	152
Rang de la N50, R(Yi)	2	4	5	3	1
Rang de la L50, R(Xi)	4	3	1	2	5
di=R(Xi-Yi)	-2	-1	-4	-1	4
di^2	4	1	16	1	16

TABLE 3

Le coefficient r_s est calculé de la façon suivante :

$$r_s = \frac{6(4+1+16+1+16)}{5^3-5} = -0.9$$

Dans ce cas, r_s vaut -0.9, ce qui signifie que la corrélation est négative. Plus la N50 est élevée et plus la L50 est basse.

2.1.2 Significativité du test

La valeur r_s est une estimation de la corrélation. Afin de tester la significativité de cette valeur, il est nécessaire de réaliser un test hypothèse dont les hypothèses sont :

— H_0 «il n'y a pas de relation entre les deux variables X et Y»

— H_1 «il y a une relation entre les deux variables X et Y»

Un seuil de significativité est défini. Celui classiquement utilisé est $\alpha = 0.05$. Si la p-valeur est supérieure à α, il n'y a pas rejet de l'hypothèse H_0. Dans le cas, où la p-valeur est inférieur à α alors l'hypothèse H_0 est rejetée.

P-valeur : la probabilité, sous H_0, d'obtenir une statistique aussi extrême que la valeur observée sur l'échantillon

2.1.3 Analyse

Le test a été réalisé sur `R`. La significativé a été définie en fonction du seuil de $\alpha = 0.05$. Pour représenter les résultats (Figure 2), nous avons utilisé des corrélogrammes[8] générés par le package R `corrplot`.Ce dernier représente la matrice de corrélation sous forme de graphique. Les coefficients de corrélation significatifs sont colorés en fonction de leur valeur variant de 1 (corrélation positive en bleu) à -1 (corrélation négative en rouge).

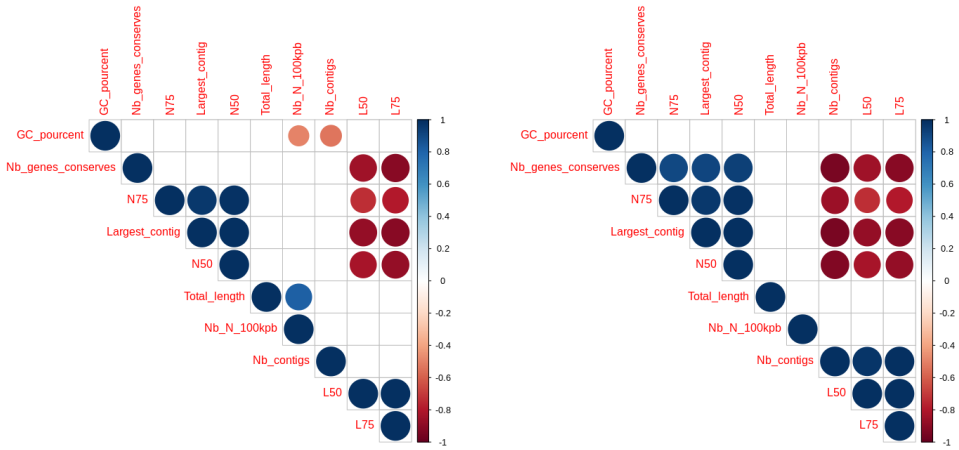

FIGURE 2 – Corrélogrammes pour les données issues de l'oiseau blanc (gauche) et de l'oursin (droite)

Sur nos jeux de données, certaines corrélations observées sont communes :

— Corrélation positive : L50/L75, N50/N75, N50/Contig le plus long, N75/Contig le plus long

— Corrélation négative : N75/L75, N75/L50, N50/L75, N50/L50, L75/Contig le plus long, L50/Contig le plus long, nombre de gènes conservés/L50, nombre de gènes conservés/L75.

D'autres corrélations sont spécifiques à chacun des jeux de données. Pour l'oiseau blanc :

— Corrélation positive : Longueur totale/Nombre de N pour 100 kbp

— Corrélation négative : %GC/Nombre de N pour 100 kbp, %GC/Nombre de contigs

Pour l'oursin :

— Corrélation positive : Nombre de gènes conservés/N75, Nombre de gènes conservés/N50, Nombre de gènes conservés/Contig le plus long, Nombre de contigs/L50, Nombre de contigs/L75

— Corrélation négative : Nombre de gènes conservés/Nombre de contigs, N75/Nombre de contigs, N50/Nombre de contigs, Contig le plus long/Nombre de contigs

Ces corrélations n'étant pas retrouvés dans les deux jeux de données, elles ne sont pas réellement exploitables.

Pour les deux jeux de données, nous n'observons pas de corrélation entre N50/75 et le nombre de contigs. Nous aurions pu partir de l'*a priori* que plus il y a de contigs plus la N50 est faible.

Toutes ces corrélations ont un sens mathématique, néanmoins cela ne signifie pas qu'il y a une causalité. Il faudrait également augmenter le nombre d'échantillons afin de confirmer l'existence de ces corrélations. De plus, il est important de noter

que la N50 a tendance à être maximisée puisque beaucoup d'assembleurs tentent de fusionner le plus de séquences possibles [9].

2.2 Analyse en composantes principales

L'utilisation de l'analyse en composantes principales (ACP) a été inspirée par «Feature-by-Feature – Evaluating De Novo Sequence Assembly»[9]. Elle permet d'étudier plus en profondeur les relations pouvant exister entre les métriques.

2.2.1 Principe

L'ACP [10] est une analyse multivariée qui permet d'explorer des jeux de données multidimensionnels. Elle permet de synthétiser les informations lorsque de nombreuses données quantitatives sont à traiter et interpréter. Ainsi, elle permet d'explorer les liaisons entre p variables quantitatives et les ressemblances entre n individus en remplaçant les p variables par q facteurs appropriés ($q<p$).

L'ACP consiste à chercher une représentation des n individus, dans un sous-espace F_k de l'espace R^p de dimension k. L'objectif est de définir k nouvelles variables correspondant à des combinaisons linéaires des p variables afin de limiter la perte d'information. Les nouvelles variables sont appelées les composantes principales et les axes qu'elles déterminent sont les axes principaux. Pour limiter la perte d'informations, la somme des carrés des distances des individus à F_k doit être minimale et ce sous-espace F_k est tel que le nuage projeté doit avoir une inertie maximale. Le nuage initial situé dans l'espace de dimension p sera réduit et projeté en dimension q.

Inertie : mesure de la dispersion totale du nuage de points

Trace : somme des coefficients diagonaux d'une matrice

En pratique, la matrice de corrélation C, dont la trace correspond à l'inertie totale, est construite à partir du tableau des données. Puis les vecteurs et valeurs propres associés à C sont calculés. Les valeurs propres sont ensuite rangées dans l'ordre décroissant. Le premier vecteur propre est associé à la plus grande valeur propre et ainsi de suite. Ces vecteurs propres déterminent les axes du nouveau repère et les valeurs propres représentent les parts d'inertie expliquées par chacun des axes principaux.

2.2.2 Analyse

Afin de choisir le nombre d'axes à retenir, nous nous sommes basés sur le critère du coude. Cela consiste à observer un décrochement dans les valeurs propres qui est suivi d'une forte décroissance. Les axes retenus sont ceux précédant le décrochement. Ainsi, en nous basant sur la figure 3, pour les deux jeux de données, les deux premiers axes sont conservés. Dans le cas de l'oiseau blanc, les deux premiers axes expliquent 93,1% de l'inertie et 92,3% pour l'oursin.

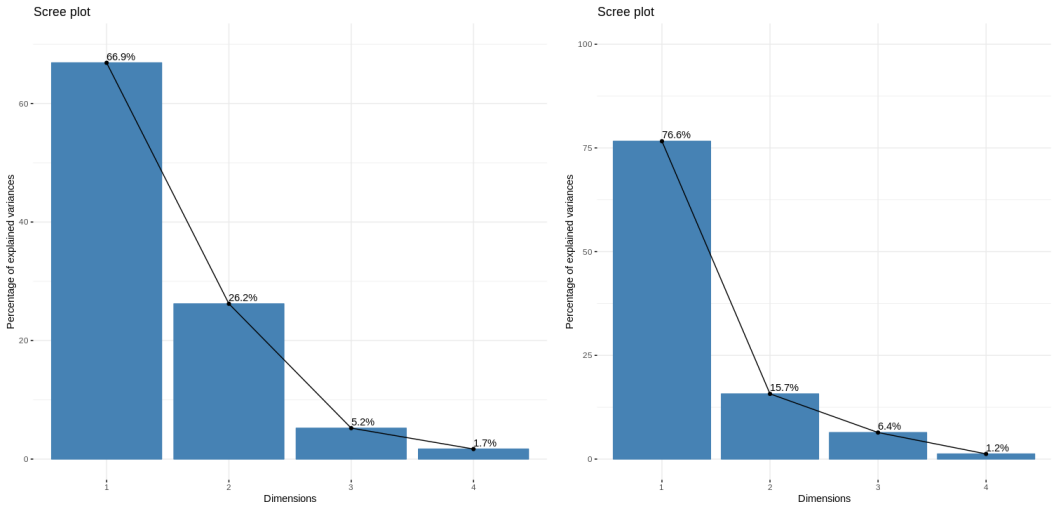

FIGURE 3 – Éboulis des valeurs propres en % de l'inertie totale (oiseau blanc à gauche et oursin à droite)

Les variables contribuant aux dimensions 1 et 2 sont présentées dans la figure 4 pour l'oiseau blanc et la figure 5 pour l'oursin.

Pour l'oiseau blanc (figure 4), l'axe 1 isole les métriques suivantes : contig le plus long, N75, L75, N50 et L50. L'axe 2 isole la longueur totale, le nombre N pour 100kb, le % de GC et le nombre de contigs.

Pour l'oursin (figure 5), les métriques isolées par l'axe 1 sont le nombre de contigs, K75, L50, contig le plus long, nombre de gènes conservés, N50 et N75. Concernant l'axe 2, les métriques isolées sont la longueur totale, le % de GC, le nombre N pour 100kb et N75.

Les variables expliquant la variabilité au sein du jeu de données diffèrent. Celles qui ne sont pas en corrélation avec un axe ou qui sont corrélées avec les derniers axes peuvent être supprimées puisqu'elles ont un faible apport. Pour l'oiseau blanc, certaines métriques ne semblent pas apporter d'informations notamment le nombre de gènes conservés. Tandis que pour l'oursin, toutes les métriques semblent être importantes pour expliquer la variabilité au sein du jeu de données.

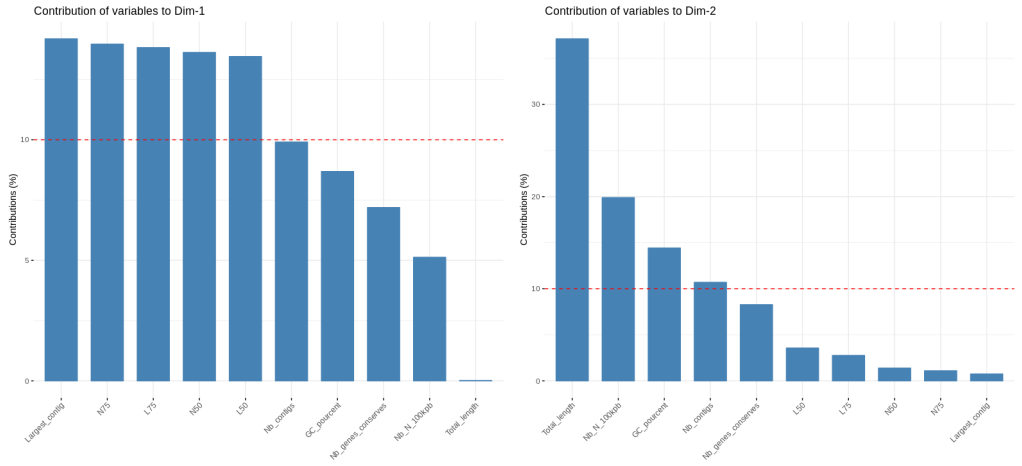

FIGURE 4 – Contribution des variables aux dimensions 1 et 2 pour les données issues de l'oiseau blanc

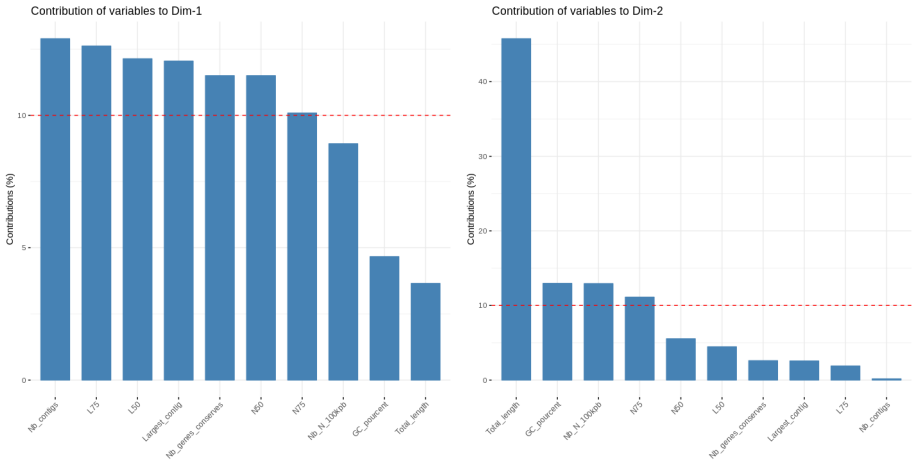

FIGURE 5 – Contribution des variables aux dimensions 1 et 2 pour les données issues de l'oursin

La figure 6 présente un biplot permettant de représenter simultanément les observations ainsi que les variables. Les vecteurs bleus sont une représentation de la corrélation des variables. La corrélation entre deux variables correspond au cosinus de l'angle entre les vecteurs. Ainsi, un angle de 90° équivaut à une corrélation nulle, un angle de 0° à une corrélation de 1 et un angle de 180° à une corrélation de -1. En outre, plus la flèche est éloignée de l'origine, meilleure est la qualité de la représentation de la variable.

Les résultats observés concordent avec ceux de la corrélation faite précédemment. Cette représentation permet cependant d'apprécier rapidement quelles sont les variables corrélées. Par exemple, pour l'oiseau blanc, les métriques N50, N75 et contig le plus long sont très positivement corrélées. Il en est de même pour la L75 et L50. Nous pouvons également voir que la L75 et L50 sont corrélées très négativement avec la N50, N75 et contig le plus long.

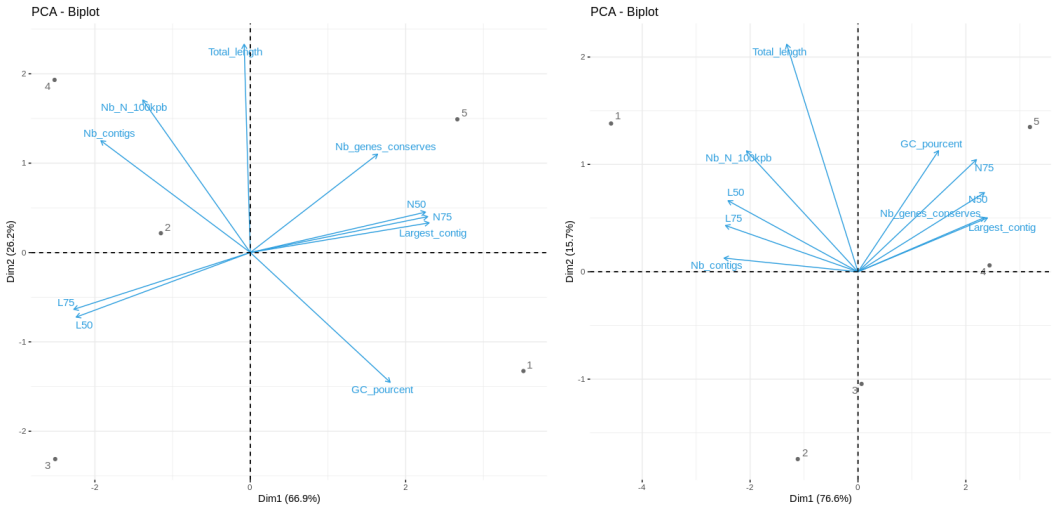

FIGURE 6 – Biplots pour les données issues de l'oiseau blanc (à gauche) et l'oursin (à droite)

3 Définition d'une fonction de score de qualité

En s'appuyant sur les métriques calculées par QUAST et BUSCO, nous souhaitions proposer un score « générique » permettant d'avoir une idée globale de la qualité des assemblages ainsi que des scores spécifiques à différentes utilisations possibles de l'assemblage.

D'abord, les résultats d'un sondage auprès de l'équipe Phylogénie et Évolution Moléculaire de l'ISEM ont été traités. Puis, nous avons procédé à une standardisation des données. En effet, les unités étant variables, cela permettra la comparaison des assemblages. Enfin, un prototype d'une fonction de score a été établi.

3.1 Sondage sur les métriques les plus utilisées

À l'occasion d'un séminaire de l'équipe Phylogénie et Évolution Moléculaire de l'ISEM, j'ai lors de la présentation de mon projet de stage proposé aux personnes présentes (chercheurs, enseignants-checheurs, post-doctorants, doctorants, techniciens en biologie et bioinformatique) de répondre à un questionnaire de ma conception (Annexe A). Ce dernier présentait les métriques recensées lors de mon stage bibliographique afin de voir lesquelles étaient les plus utilisées.

Dix questionnaires ont été recueillis. Les réponses sont présentées dans la figure 7. Comme nous pouvons le voir, il s'agit de métriques que nous avions retrouvées dans de nombreuses publications dont l'Assemblathon et Genome Assembly Gold-Standard Evaluation. En effet, les métriques les plus utilisées sont la N50, le nombre de contigs, le nombre de gènes, le %GC, le score BUSCO et la longueur totale de l'assemblage.

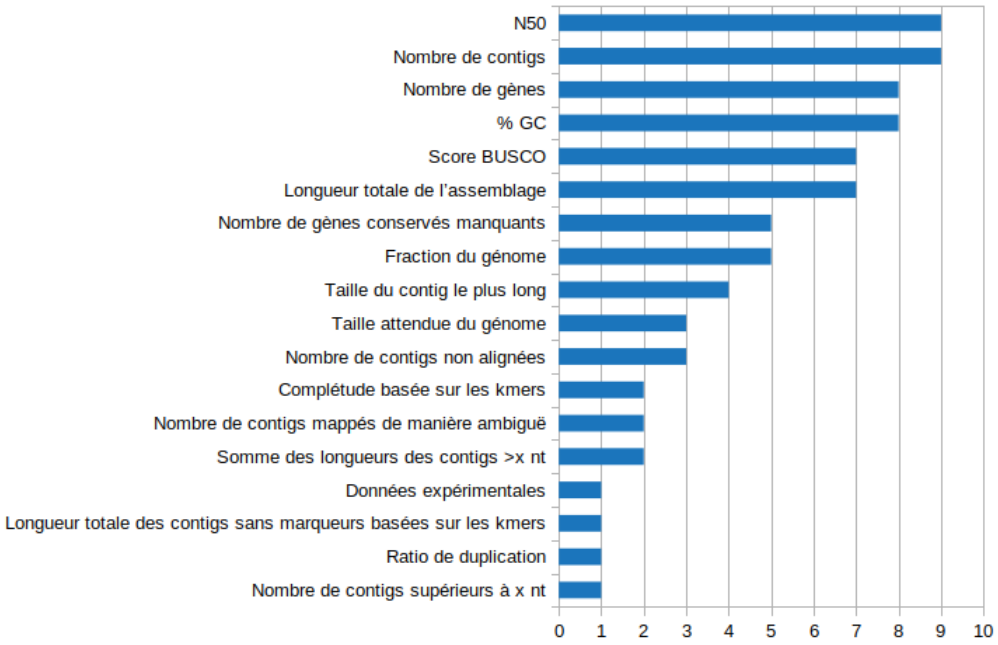

FIGURE 7 – Graphique présentant les résultats du questionnaire

Une question ouverte a également été posée concernant la vision qu'ils avaient d'un bon assemblage. Plusieurs personnes ont indiqué que l'utilisation finale de l'assemblage avait une importance. Par exemple :

- pour la génétique des populations : les chercheurs se basent sur l'homologie, ainsi ils recherchent les gènes orthologues et à avoir le même nombre de gènes qui seraient de mêmes longueurs
- pour un génome complet : la N50 ainsi que la longueur des contigs sont très importantes puisqu'elles représentent la contiguité dans l'assemblage
- en phylogénie : le nombre de substitutions et d'insertions/délétions doit être minimal
- en métagénomique : les métriques basées sur les k-mers sont les plus importantes

Ces informations nous permettront de proposer plusieurs manières pré-conçues de calculer différents scores propre à chaque application.

3.2 Standardisation des données

Afin de standardiser les données, il existe différentes méthodes, notamment la standardisation par le Z-score et la standardisation par le Min-Max qui sont les plus connues. Par exemple, l'Assemblathon 2[11] utilise le cumul des Z-scores pour la comparaison entre les assemblages.

Dans l'approche de standardisation Min-Max, les données sont mises à l'échelle sur une plage fixe qui est généralement comprise entre 0 et 1.

Les valeurs sont transformées en utilisant la formule suivante :

$$x = \frac{valeur - min}{max - min}$$

Le *min* étant la valeur minimum parmi les données et le *max* la valeur maximum.

La standardisation par le Z-score sert à transformer les données afin qu'elles aient les propriétés d'une distribution normale centrée-réduite avec $\mu = 0$ et $\sigma = 1$ où μ est la moyenne et σ est l'écart type par rapport à la moyenne.

Les valeurs sont transformées en utilisant la formule suivante :

$$z = \frac{valeur - \mu}{\sigma}$$

Exemple La table 4 présente un exemple de transformation de la variable N50 par les deux méthodes présentées précédemment.

Assemblages	**Métrique N50**	**Standardisation Min-Max**	**Standardisation Z-score**
MaSuRCA	1 859 356	0,81	0,88
SOAP	488 989	0,10	-0,65
SOAP_GC_rich	296 704	0	-0,87
SOAP_K27	505 574	0,11	-0,64
SOAP_Pacbio	2 218 729	1	1,28

TABLE 4 – Exemple de standardisation de la N50 par le Min-Max et Z-score sur le jeu de données de l'oiseau blanc

La standardisation par le Z-score a tendance à minimiser la variance ce qui permet de mieux prendre en compte les valeurs extrêmes. Cependant, elle ne permet pas d'obtenir des valeurs comprises dans un intervalle précis.

3.3 Prototype de la fonction de score appliquée aux jeux de données constitués

Nous avons implémenté le calcul de façon à obtenir les scores après transformation par les deux méthodes de standardisation. Nous savons que certaines métriques sont considérées comme meilleures quand elles sont élevées par exemple la N50, la N75 ou le pourcentage en GC. La N50 correspond à la taille du contig de longueur L pour laquelle les contigs de taille supérieure ou égale à L couvrent au moins 50% de l'assemblage. Plus celle-ci est élevée, plus le génome est couvert par de grands scaffolds. Pour le pourcentage en GC, les régions riches seraient plus riche en gènes (Clay et al., 1996). D'autres métriques au contraire sont meilleures lorsqu'elles sont faibles, par exemple la L50 qui correspond au nombre de contig de taille égale ou supérieur à la N50.

Afin de calculer le score, nous avons fait le cumul des valeurs standardisées.

$$score = \sum(a \times métrique)$$

avec a le coefficient assigné à la métrique et *métrique* la valeur standardisée de la métrique. Lorsque la métrique doit être élevée, la valeur est utilisée telle quelle. A l'inverse, lorsque la valeur doit être faible, l'opposé de cette valeur est pris en compte.

Dans les exemples suivant, c'est le score «générique» qui a été calculé où toutes les métriques avaient un poids identique.

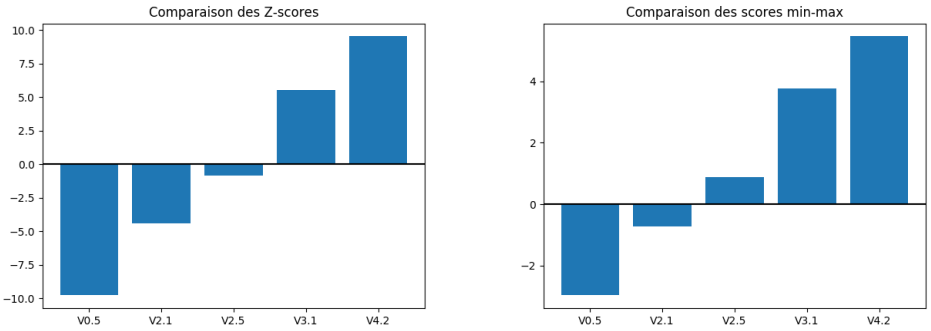

FIGURE 8 – Visualisation des scores de l'oursin

Lorsque nous observons les résultats pour l'oursin (Figure 8), nous pouvons constater que le score augmente entre chaque version. La version 0.5 est la plus mauvaise tandis que la version 4.1 est la meilleure. Ainsi, le résultat semble logique puisqu'à chaque *release*, l'assemblage doit être de meilleure qualité.

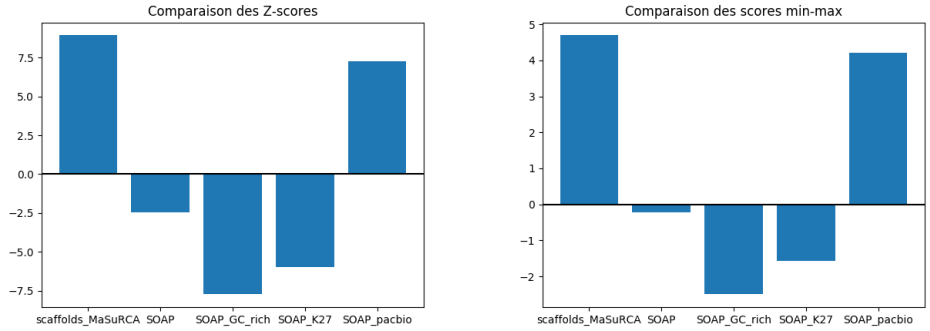

FIGURE 9 – Visualisation des scores de l'oiseau blanc

Concernant l'oiseau blanc (Figure 9), les deux meilleurs assemblages semblent être MaSuRCA et SOAP_Pacbio. Nous savons *a priori* que l'assemblage qui a été choisi par l'équipe qui a généré les données est celui produit par SOAP à partir des *reads* Pacbio auquel des corrections ont été apportées par les *reads* courts.

4 Étude des alignements deux à deux de scaffolds

Dans cette partie, nous nous intéressons plus particulièrement à la comparaison des alignements des assemblages de l'oiseau blanc. Pour cela, nous avons procédé à l'alignement des assemblages deux à deux en utilisant MUMmer[12] qui a fait l'objet d'une explication au cours de mon rapport au début du stage [13]. Dans un premier temps, le module `Nucmer`effectue les alignements et permet l'obtention d'un fichier delta (exemple en Annexe B) contenant toutes les informations concernant les alignements. Ce format répertorie les les coordonnées des régions alignées, la distance entre les insertions et les suppressions contenues dans celles-ci. Cependant, le format delta n'est que peu lisible, ainsi le module `Show-coords` a permis d'obtenir des fichiers de sortie plus facilement analysables (exemple en Annexe C). Ces derniers permettent d'obtenir les informations suivantes :

- [S1] [E1] [S2] [E2] : le début et fin de la région s'alignant dans la «référence» et dans la «requête»
- [LEN 1] [LEN 2] : la longueur de la région d'alignement dans la «référence» et dans la «requête» (en nucléotides)
- [% IDY] : le pourcentage d'identité (calculé de la façon suivante : (Nombre de *match* exacts) / (Longueur de la «référence» + insertions dans la requête)
- [LEN R] [LEN Q] : la longueur totale des séquences de la «référence» et de la «requête»
- [TAGS] : les *tags* :
  - identifiant de la «référence»
  - identifiant de la «requête»
  - des informations complémentaires sur l'alignement, notamment s'il y a un chevauchement ou si la région d'alignement est contenue dans un des scaffolds

Nous avons décidé de représenter les alignements entre scaffolds par deux types de modélisation :

- graphe biparti (comparaison 2 à 2)
- graphe de similarité (comparaison de tous les assemblages)

Lorsqu'un alignement est détecté, il y a création d'une arête entre deux sommets représentant deux scaffolds provenant de deux assemblages différents. Cela nous permettra dans un premier de quantifier le nombre de similarités mais également détecter des structures telles que les cliques. La détection d'une clique signifierait qu'il y a une correspondance entre les scaffolds provenant chacun d'un assemblage ce qui pourrait permettre de soutenir l'hypothèse que ce scaffold existe réellement.

Clique : sous ensemble de sommets tous en relation deux à deux

Par la suite, nous proposons des stratégies de visualisation des résultats permettant de faciliter l'interprétation des résultats.

4.1 Algorithme

Dans un premier temps, nous avons tenté d'utiliser un package Python, NetworkX[14], qui permet la création, la manipulation et l'étude de la structure, de la dynamique et des fonctions de réseaux complexes. Néanmoins certaines fonctions, notamment la recherche de cliques au sein du graphe, ont une complexité exponentielle lorsque les données sont volumineuses[16]. Ainsi, pour pallier au problème

de la recherche de cliques, un algorithme permettant de retrouver les composantes connexes a été développé. L'algorithme 1 définit les fonctions qui permettent de trouver les sommets présents dans une structure contenant les composantes connexes et de fusionner deux composantes connexes lorsqu'il y a un sommet commun. L'algorithme 2 permet de rechercher l'existence d'une arête formée par deux sommets parmi les composantes connexes existantes dans la structure et de la mettre à jour. Pour cela, les fichiers d'alignements sont analysés arête par arête.

Composante connexe : sous-graphe connexe maximal d'un graphe

Nous avons posé :

— n le nombre d'arêtes, une arête représentant un alignement deux à deux

— a le nombre d'assemblages comparés

Chaque assemblage étant comparé deux à deux et sans doublons, nous obtenons alors $\frac{a!}{(a-2)!2!}$ comparaisons à traiter. L'algorithme consiste à classifier les différentes arêtes en plusieurs composantes connexes dans une structure. Cela nous permet de récupérer les composantes connexes sans qu'il ne soit nécessaire de stocker le graphe complet dans une structure.

Le langage qui a été choisi pour implémenter l'algorithme est Python. Pour stocker les composantes, nous avons le choix entre deux types de structures de données proposées par le langage qui auront un impact sur la performance :

— la liste contenant un ensemble d'éléments ordonnés et qui est implémentée comme un tableau

— le set qui contient un ensemble d'éléments uniques non ordonnés implémenté avec une table de hachage

La complexité des structures de données implémentées en Python peuvent être retrouvées sur la page Wiki de Python[17].

La fonction Trouve(sommet,structure) utilise "dans" (`in` en python) qui est en O(n) sur une liste tandis qu'elle est en O(1) sur le set. La table 5 présente les différents cas possible. Ainsi, la complexité est de O(C) avec C représentant le nombre de composantes connexes existantes pour les sets. Dans le cas des listes, la complexité vaut O(n).

Quant à la fonction Fusion(C1,C2,structure), elle supprime des éléments ce qui est O(n) dans les listes mais O(1) en cas moyen dans les sets.

	Nombre de composantes connexes
"Meilleur" des cas	1
"Pire" des cas	n
Cas des cliques	$\frac{n}{a}$

TABLE 5 – Tableau présentant les valeurs de C possible pour chaque cas

Pour la structure contenant les composantes connexes est une liste pour que les éléments soient indexés. Les composantes connexes seront elles des listes ou de sets. Pour la recherche des composantes connexes, dans le cas des sets, la complexité est de O(Cn). Pour les listes, elle est de $O(n^2)$. Dans la théorie, les sets sont plus rapides,

compte tenu du fait que C«n. Nous avons comparé les temps d'exécution pour la comparaison deux à deux des assemblages en utilisant les deux structures.

Assemblage 1	**MaSuRCA**	**MaSuRCA**	**MaSuRCA**	**MaSuRCA**	**SOAP GC Rich**	**SOAP GC Rich**	**SOAP K27**	**SOAP**	**SOAP**	**SOAP**
Assemblage 2	**SOAP**	**SOAP K27**	**SOAP Pacbio**	**SOAP GC Rich**	**SOAP K27**	**SOAP Pacbio**	**SOAP Pacbio**	**SOAP GC Rich**	**SOAP K27**	**SOAP Pacbio**
Temps d'exécution avec les sets	0m27s	0m58s	0m16s	0m7s	1m17s	0m27s	44m34s	1m25s	157m40s	78m28s
Temps d'exécution avec les listes	0m47s	1m33s	0m23s	0m10s	1m55s	0m40s	63m2s	2m25s	215m26s	123m20s

TABLE 6 – Temps moyen d'exécution de l'algorithme d'analyse des alignements deux à deux

La structure privilégiée serait *a priori* les sets. Pour confirmer l'hypothèse, deux versions ont été implémentées. L'algorithme a été exécuté 3 fois sur l'ordinateur utilisé en stage (16Gb de RAM, 8 cœurs). Comme nous pouvons le voir dans la table 6, la version de l'algorithme implémentée avec les listes est plus lente que celle implémentée avec les sets. La structure finalement conservée est celle se basant sur l'utilisation des sets.

Algorithme 1 Déclaration des fonctions utilisés pour la recherche des composantes connexes

```
Début
1:  | Fonction TROUVE(sommet,structure)
2:  |   Si longueur(structure)==0 Alors
3:  |     Retourner -1
4:  |   Sinon
5:  |     Pour i allant de 0 à longueur(structure)-1 Faire
6:  |       Si sommet dans stucture[i] Alors
7:  |         Retourner i
8:  |       Sinon
9:  |         Retourner -1
10: |       Fin Si
11: |     Fin Pour
12: |   Fin Si
13: | Fin Fonction
14:
15: | Fonction FUSION_LISTE(C1,C2,structure)
16: |   Si longueur(stucture[C1]) ≥ longueur(stucture[C2]) Alors
17: |     Ajouter(stucture[C2]) dans structure[C1]
18: |     Supprimer(stucture[C2])
19: |   Sinon
20: |     Ajouter(structure[C1]) dans structure[C2]
21: |     Supprimer(structure[C1])
22: |   Fin Si
23: | Fin Fonction
Fin
```

Algorithme 2 Recherche des composantes connexes

```
Début
taille_structure=0
Pour chaque arêtes (a,b) Faire
    edge_a ← Trouve(a,structure)
    edge_b ← Trouve(b,structure)
    Si edge_a > 0 et edge_b < 0 Alors
        Ajouter(b) dans Structure[edge_a]
    Sinon Si edge_b > 0 et edge_a < 0 Alors
        Ajouter(a) dans Structure[edge_b]
    Sinon Si edge_a > 0 et edge_b > 0 Alors
        Fusion(Structure[edge_a],Structure[edge_b])
    Sinon
        Créer(Structure[taille])
        Ajouter(a) dans Structure[taille]
        Ajouter(b) dans Structure[taille]
        taille_structure++
    Fin Si
Fin Pour
Fin
```

4.2 Comparaison deux à deux des assemblages

4.2.1 Résultats

La table 7 présente le nombre de composantes connexes composées d'un seul sommet pour chaque assemblage et d'au moins deux sommets provenant de deux assemblages. Par exemple, pour la première colonne de la table 7 : 5 082 scaffolds sont uniques dans l'assemblage MaSuRCA, 123 438 scaffolds sont uniques dans l'assemblage SOAP et les deux assemblages partagent 322 composantes connexes. Nous pouvons constater qu'une grande majorité des scaffolds n'ont pas de correspondances dans d'autres assemblages.

Assemblage 1	**MaSuRCA**	**MaSuRCA**	**MaSuRCA**	**MaSuRCA**	**SOAP GC Rich**	**SOAP GC Rich**	**SOAP K27**	**SOAP**	**SOAP**	**SOAP**
Assemblage 2	**SOAP**	**SOAP K27**	**SOAP Pacbio**	**SOAP GC Rich**	**SOAP K27**	**SOAP Pacbio**	**SOAP Pacbio**	**SOAP GC Rich**	**SOAP K27**	**SOAP Pacbio**
Nombre de sommets uniques dans l'assemblage 1	5 082	5 131	5 231	5 312	9 242	9 301	83 639	129 290	107 179	85 761
Nombre de composantes connexes	322	276	175	94	673	616	14 974	748	22 576	43 707
Nombre de sommets uniques dans l'assemblage 2	129 438	98 858	97 104	9 778	98 259	96 386	81 856	9 044	75 488	53 105

TABLE 7 – Résultats de l'analyse des fichiers d'alignements deux à deux.

4.2.2 Visualisation

Lors du projet bibliographique [4], nous avions émis l'idée de visualiser les résultats par un diagramme de Venn [20]. Cette représentation permet de voir rapidement quels sont les assemblages présentant peu ou beaucoup de scaffolds en commun.

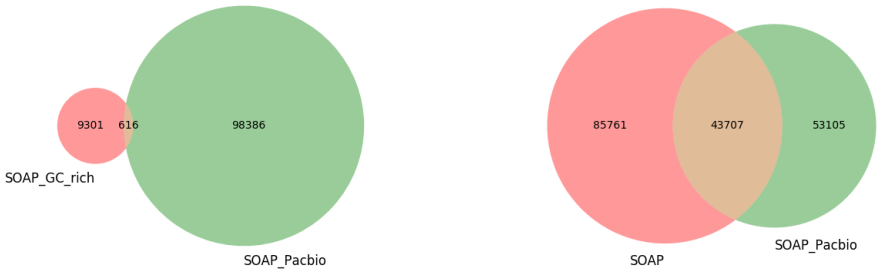

FIGURE 10 – Visualisation du nombre de composantes connexes par un diagramme de Venn

La figure 10 présente deux exemples issu de la table 7. En comparant l'assemblage SOAP_Pacbio avec le SOAP_GC_rich nous pouvons voir que les composantes connexes ne représentent qu'une infime partie de chacun des assemblages. Contrairement à la comparaison de SOAP_Pacbio avec SOAP, où nous observons une plus grande part de composantes connexes communes.

4.3 Comparaison de tous les assemblages

4.3.1 Résultats

Dans le cas le plus simple, chaque scaffold a un équivalent dans d'autres assemblages. Ainsi, par exemple dans le cas d'une comparaison de 5 assemblages, nous devrions retrouver des cliques de 5 scaffolds.

Le temps d'exécution est 920m43s (environ 15h) pour la comparaison des 5 assemblages. Nous avons détecté 16 745 composantes connexes qui sont composées au total de 35 323 scaffolds. La table 8 présente pour chaque assemblage le nombre de sommets uniques. Comme nous pouvons le voir, il n'y a qu'une faible proportion des scaffolds qui présentent des similarités. Il faut également noter que chaque composante connexe n'est pas forcément composée d'un scaffold issu de chaque assemblage. La partie visualisation 4.3.2 permet de voir quels sont les sous ensembles parmi les composantes connexes.

	SOAP	**SOAP_GC_rich**	**SOAP_K27**	**SOAP_Pacbio**	**MaSuRCA**
Nombre de scaffolds	130 278	9 917	99 405	97 415	5409
Nombre de composantes connexes ne présentant qu'un sommet	123 336	9 808	88 357	80479	5121

TABLE 8 – Nombre de composantes connexes composées d'un seul sommet pour chaque assemblage

4.3.2 Visualisation

Comme pour la comparaison deux à deux des assemblages, nous avions émis l'hypothèse d'une représentation des résultats grâce à un diagramme de Venn. Néanmoins, comme nous pouvons le voir dans la figure 11, cette modélisation est difficilement lisible lorsque le nombre d'assemblages augmente. Par exemple, nous observons 2 composantes connexes qui sont composées de scaffolds issus des assemblages SOAP, MaSuRCA et SOAP_GC_rich.

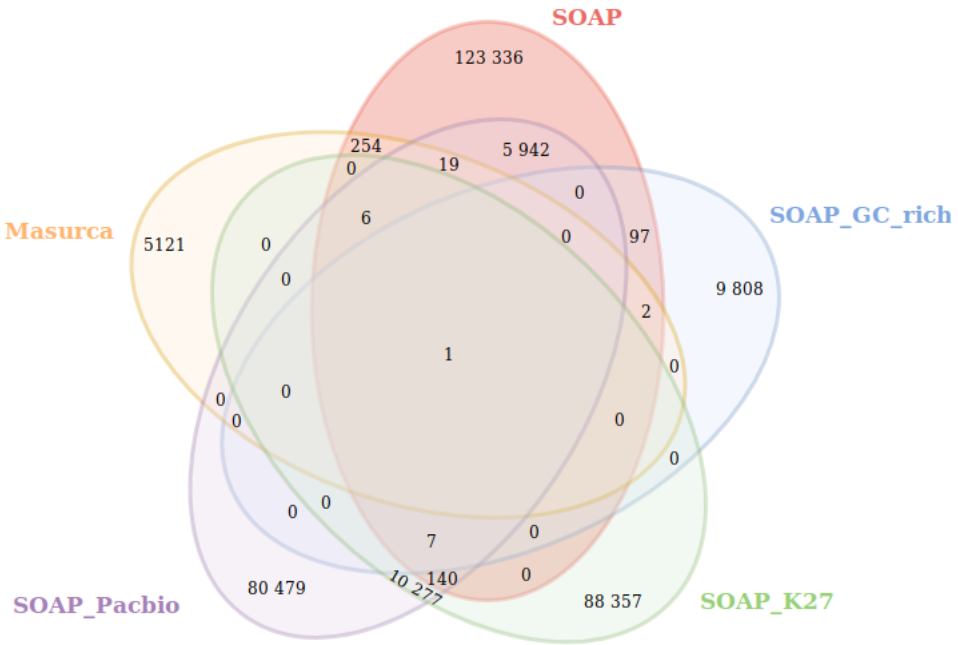

FIGURE 11 – Visualisation du nombre de composantes connexes par un diagramme de Venn

Une autre manière de représenter les données sont les graphiques UpSet[15]. Elle permet de visualiser les résultats d'une analyse quantitative d'ensembles, de leurs intersections et des agrégats d'intersections. UpSet visualise les intersections définies dans une mise en page matricielle. La disposition de la matrice permet la représentation effective des données associées, tels que le nombre d'éléments dans les agrégats et les intersections.

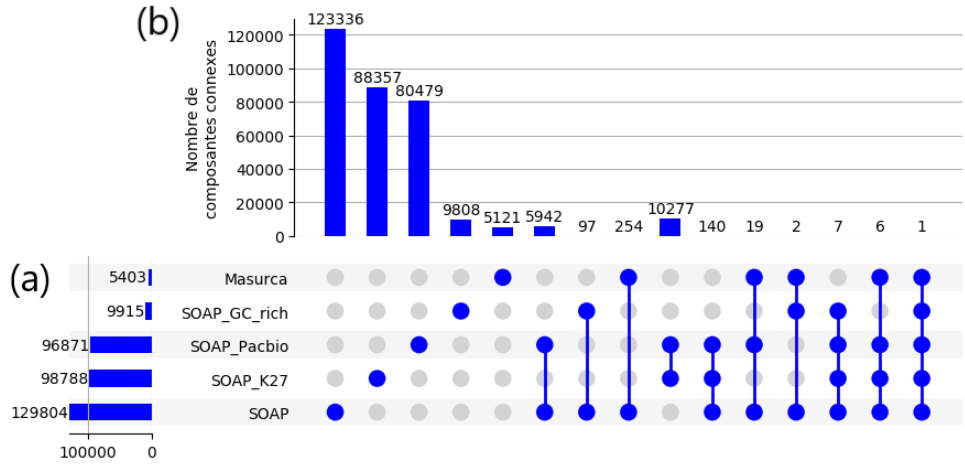

FIGURE 12 – Visualisation du nombre de composantes connexes par UpSet.
(a) Graphique présentant le nombre de composantes connexes totales pour chaque assemblage (à gauche) ainsi que les sous-ensembles (à droite)
(b) Graphique présentant le nombre de composantes connexes pour chaque sous-ensemble du graphique (a)

Sur la figure 12 nous pouvons voir par exemple que pour le sous-ensemble SOAP_Pacbio et SOAP_K27 qu'il y a 10 277 composantes connexes. SOAP_Pacbio est composé de 96 871 composantes connexes au total et SOAP_K27 de 98 788.

Comme nous pouvons le voir, de nombreux scaffolds ne présentent pas de correspondances dans d'autres assemblages. Il n'y a qu'une composante connexe présentant un scaffold similaire provenant de chaque assemblages. Nous pouvons également constater que la plupart du temps, les scaffolds assemblés avec SOAPdenovo[19] sont les plus similaires malgré un traitement différent lors de la génération des *reads*. Enfin, nous pouvons observer une différence entre le nombre de composantes connexes totales et le nombre de scaffolds total pour chaque assemblage. Cela est dû au fait que plusieurs scaffolds proviennent d'un même assemblage pour certaines composantes connexes.

5 Discussion et perspectives

L'évaluation de la qualité et la comparaison des assemblages de génome reste encore une des problématiques à étudier plus en profondeur. Actuellement, de nombreux outils tels que QUAST[2] ou BUSCO[3] permettent de calculer des métriques et de les visualiser. Néanmoins, celles-ci ne permettent pas d'apprécier la qualité globale des assemblages ni de pouvoir comparer à proprement dit des assemblages ne présentant pas de référence. Ainsi, l'objectif final de ce projet serait de pallier aux limites actuelles.

Au cours du stage, nous avons souhaité étudier les liens existants entre les métriques de qualité afin d'établir une fonction de score dans une première partie. Les analyses menées au travers de l'étude de la corrélation et l'analyse en composantes principales n'ont pas été très concluantes. Nous n'avons pas retrouvé une corrélation

négative attendue entre le nombre de contigs et la N50. Nous avons néanmoins pu observer une corrélation négative entre la L50/75 et la N50/75 confirmant que plus la N50/75 est élevé et plus la L50/75 est faible. Pour la suite du projet, de nouveaux jeux de données seront mis à disposition notamment grâce à de nouvelles collaborations. Cela permettra d'augmenter le nombre d'analyses et ainsi exposer les liens existants entre les métriques. Par ailleurs, le questionnaire posé a permis de mettre en avant des aspects importants à prendre en compte, notamment l'utilisation finale de l'assemblage qui influence la prise en compte de certains critères. La fonction de score actuellement mise en place pourra être plus amplement affinée grâce à de nouvelles analyses.

Dans la seconde partie du stage, nous avons souhaité comparer les assemblages. Pour cela, les scaffolds avaient été alignés deux à deux grâce à MUMmer. Sur le jeu de données de l'oiseau blanc, nous avons pu mettre en lumière peu de similarité entre les différents assemblages. L'analyse reste à approfondir notamment en explorant la composition des composantes connexes. En effet, nous avons pu voir que certaines composantes connexes pouvaient contenir plusieurs scaffolds provenant d'un même assemblage. Nous pourrons par la suite utiliser ces données afin de déterminer une distance entre chaque assemblage.

Les travaux effectués au cours de mon stage ont été essentiellement exploratoires. Cela a permis de dégrossir la problématique et visualiser les prochains axes de développement.

Références

Bibliographie

[1] Hind Alhakami, Hamid Mirebrahim et Stefano Lonardi. « A comparative evaluation of genome assembly reconciliation tools ». In : *Genome biology* 18.1 (2017), p. 93.

[2] Alla Mikheenko et al. « Versatile genome assembly evaluation with QUAST-LG ». In : *Bioinformatics* 34.13 (2018), p. i142–i150.

[3] Felipe A Simão et al. « BUSCO : assessing genome assembly and annotation completeness with single-copy orthologs ». In : *Bioinformatics* 31.19 (2015), p. 3210–3212.

[4] Anna Tran. « Critères de qualité des assemblages des génomes ». 2018.

[5] Thibault Leroy et al. « A bird's white-eye view on neosex chromosome evolution ». In : *bioRxiv 505610, ver. 4 peer-reviewed and recommended by PCI Evolutionary Biology* (2019). doi : 10.1101/505610.

[6] Erica Sodergren et al. « The genome of the sea urchin Strongylocentrotus purpuratus ». In : *Science* 314.5801 (2006), p. 941–952.

[7] Jerrold H Zar. « Spearman rank correlation ». In : *Encyclopedia of Biostatistics* 7 (2005).

[8] Michael Friendly. « Corrgrams : Exploratory displays for correlation matrices ». In : *The American Statistician* 56.4 (2002), p. 316–324.

[9] Francesco Vezzi, Giuseppe Narzisi et Bud Mishra. « Feature-by-Feature – Evaluating De Novo Sequence Assembly ». In : *PLOS ONE* 7.2 (fév. 2012), p. 1–12. doi : 10.1371/journal.pone.0031002.

[10] Brigitte Escofier et Jérôme Pagès. *Analyses factorielles simples et multiples. Objectifs méthodes et interprétation*. Dunod, 2008, p. 7–47.

[11] Keith R Bradnam et al. « Assemblathon 2 : evaluating de novo methods of genome assembly in three vertebrate species ». In : *GigaScience* 2.1 (2013), p. 10.

[12] Stefan Kurtz et al. « Versatile and open software for comparing large genomes ». In : *Genome biology* 5.2 (2004), R12.

[13] Anna Tran. « Évaluation de la qualité et comparaison d'assemblages de génomes ». 2019.

[14] Aric Hagberg, Pieter Swart et Daniel S Chult. *Exploring network structure, dynamics, and function using NetworkX*. Rapp. tech. Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[15] Alexander Lex et al. « UpSet : visualization of intersecting sets ». In : *IEEE transactions on visualization and computer graphics* 20.12 (2014), p. 1983–1992.

Ressources web

[16] NetworkX Developers. *NetworkX Online Documentation*. 2019. url : https://networkx.github.io/documentation/stable/reference/algorithms/generated/networkx.algorithms.clique.find_cliques.html#networkx.algorithms.clique.find_cliques.

[17] Jonathan Hartley. *Time Complexity*. 2017. url : https://wiki.python.org/moin/TimeComplexity.

Bibliographie non lue

[18] Aleksey V Zimin et al. *The MaSuRCA genome assembler*. 2013.

[19] Ruibang Luo et al. *SOAPdenovo2 : an empirically improved memory-efficient short-read de novo assembler*. 2012.

[20] Branko Grünbaum. *The Construction of Venn Diagrams*. 1984. doi : 10.1080/00494925.1984.11972776. eprint : https://www.tandfonline.com/doi/pdf/10.1080/00494925.1984.11972776.

A Sondage sur les métriques de qualité

Évaluation de la qualité et comparaison d'assemblages de génomes
Questionnaire
Anna TRAN - Master 2 SNS parcours BCD
Avril 2019

Quels sont les critères de qualité que vous utilisez ? Cochez les cases correspondantes.

1. Analyse des contigs
   - ☐ Nombre de contigs
   - ☐ Taille du contig le plus long
   - ☐ Longueur totale de l'assemblage
   - ☐ N50
   - ☐ NG50
   - ☐ NA50
   - ☐ Nombre de contigs >x nucléotides
   - ☐ Somme des longeurs des contigs >x nucléotides
2. Assemblages incohérents et variations structurales
   - ☐ Nombre d'assemblages incohérents
   - ☐ Nombre de contig contennant des erreurs d'assemblage
   - ☐ Nombre de contigs non alignés
   - ☐ Nombre de contigs mappés de manière ambigüe
3. Représentation du génome et des éléments fonctionnels
   - ☐ Fraction du génome
   - ☐ Ratio de duplication
   - ☐ % GC
   - ☐ Nombre mismatch pour 100kb
   - ☐ Nombre d'indels
   - ☐ Nombre de gènes
   - ☐ Nombre d'opérons
   - ☐ Nombre de gènes prédits
   - ☐ Nombre de gènes conservés manquants
   - ☐ Nombre moyen d'orthologues par gènes conservés
   - ☐ % de gènes constitutifs détectés ayant plus d'un orthologue
   - ☐ Score BUSCO
4. Analyses des k-mers
   - ☐ Complétude basée sur les k-mers
   - ☐ Nombre de k-mers mal raccordés
   - ☐ Longueur correcte basée sur les k-mers
   - ☐ Longueur basée sur les k-mers mal raccordés

- ☐ Longueur totale des contigs sans marqueurs basée sur les k-mers

5. Autres
   - ☐ Données expérimentales (fosmides, cartes optiques)
   - ☐ Taille attendue du génome
   - ☐ Distribution de la taille des inserts
   - ☐ Taux d'erreurs par base

Qu'est-ce qu'un bon assemblage selon vous ?

B Exemple de fichier delta

```
1 >scaffold1042 scaffold3_size3167600 225937 3167600
2 38324 39087 20223 20986 0 725 0
3 0
4 59539 60256 1298256 1298973 0 582 0
5 0
6 89694 89838 959459 959315 0 0 0
7 0
```

Listing 1 – Exemple de fichier delta obtenu suite à l'utilisation du module Nucmer issu de MUMmer

C Exemple de fichier obtenu après utilisation du module Show-coords

```
1  [S1] [E1] [S2] [E2] [LEN 1] [LEN 2] [% IDY] [LEN R] [LEN Q] [TAGS]
2  48507 48657 1 151 151 151 100.00 56637 151 scaffold10 scaffold36571_size151 [
       CONTAINS]
3  48507 48657 1 151 151 151 92.72 56637 151 scaffold10 scaffold36590_size151 [CONTAINS
       ]
4  48649 48847 200 1 199 200 93.50 56637 200 scaffold10 scaffold25111_size200 [CONTAINS
       ]
5  49041 49137 1771897 1771801 97 97 100.00 56637 10030335 scaffold10
       scaffold14_size10030335
6  49041 49137 2506017 2505921 97 97 100.00 56637 2916324 scaffold10
       scaffold220_size2916324
7  49043 49137 864814 864908 95 95 100.00 56637 2672435 scaffold10
       scaffold252_size2672435
8  49068 49144 30976 31052 77 77 100.00 56637 668223 scaffold10 scaffold612_size668223
9  49072 49151 1419411 1419332 80 80 100.00 56637 1660463 scaffold10
       scaffold374_size1660463
10 49171 49415 251 493 245 243 90.20 56637 537 scaffold10 scaffold11231_size537
11 49171 50546 4346 2936 1376 1411 89.46 56637 19811 scaffold10 scaffold1869_size19811
12 49411 56637 19375 12118 7227 7258 98.55 56637 19375 scaffold10
       scaffold1963_size19375 [END]
```

Listing 2 – Exemple de fichier obtenu suite à l'utilisation du module Show-coords issu de MUMmer