# Rapport de projet3 MASB Assemblage

Le but de ce TP est d'écrire un assembleur basé sur un graph de De Bruijn afin d'assembler plusieurs jeux de données en mettant en évidence divers défis en gérant par exemple les erreurs de séquençage et les répétitions et pouvoir évaluer la qualité de l'assemblage produit avec quast.

**Données utilisées** : https://nextcloud.univ-lille.fr/index.php/s/Ao8kz6iXkA83mJR

**IMPLÉMENTATION DU CODE**

- script final fonctionnel : **project3_assembly.py**
Le script **project3_assembly.py** ne traite pas directement les erreurs de type bulle, tips ou motifs compliqués. Cependant, il permet de résoudre les conflits qui peuvent survenir lorsque plusieurs chemins sont possibles à partir d'un nœud donné dans le graphe de de Bruijn. Plus précisément, les cas où un nœud a plusieurs successeurs, en choisissant le successeur qui génère le plus long contigs possible en explorant récursivement le graphe à partir de ce successeur. Cette méthode permet de sélectionner le chemin le plus probable dans le graphe de de Bruijn, en utilisant la longueur du contig comme critère de sélection. Cela peut aider à éviter les fausses connexions et à améliorer la qualité de l'assemblage final en résolvant les conflits entre les différents chemins possibles.

**project3_assembly.py :**
***extract_sequences :*** Cette fonction permet d'extraire les séquences nucléotidiques à partir d'un fichier au format FASTA ou FASTQ. Elle prend en entrée le nom d'un fichier, lit le contenu du fichier, et extrait les séquences nucléotidiques dans les object records du fichier à l'aide de la bibliothèque Biopython. Les séquences sont stockées dans une liste.

***kmer_index :*** La fonction prend en entrée une liste de séquences d'ADN et un entier k, et renvoie un dictionnaire contenant les k-mers présents dans les séquences et leur nombre d'occurrences. Elle utilise un modèle d'expression régulière pour valider que la séquence qui ne contient que les nucléotides A, C, G et T.

***filter_kmers*** Elle permet de filtrer les k-mers dont l'occurrence est inférieure à un seuil spécifié, afin de ne conserver que les k-mers les plus pertinents, car certaines séquences peuvent être des artefacts ou des erreurs de séquençage et ne sont donc pas pertinentes pour l'analyse. En filtrant les k-mers, on peut améliorer la qualité des résultats obtenus. Elle prend en entrée un dictionnaire contenant les k-mers et leur nombre d'occurrences, ainsi qu'un seuil minimum d'occurrence à conserver pour chaque k-mer. Elle renvoie un nouveau dictionnaire ne contenant que les k-mers dont l'occurrence est supérieure ou égale au seuil spécifié.

***predecessors_and_successors*** permet de construire un dictionnaire qui contient pour chaque k-mer dans la fonction kmer_index, une liste de ses prédécesseurs et de ses successeurs(k-mer identiques au k-mer considéré avec uniquement une différence sur le premier nucléotide).

***simple_path*** permet la construction de chemins simples dans le graphe de De Bruijn pour construire chaque contig. Elle prend en entrée un index représentant un graphe de De Bruijn et qui retourne une liste de contigs construits à partir de l'index en suivant les chemins

simples. La fonction utilise une fonction récursive "build_contig" pour construire les contigs en explorant les successeurs et prédécesseurs des k-mers du graphe. Les contigs sont stockés dans un dictionnaire pour éviter de recalculer les contigs déjà construits. Elle explore chaque k-mer dans l'index qui n'a pas été visité pour construire la liste de contigs, qu'elle retourne. L'approche récursive me semblait plus logique même si elle peut avoir des cout de mémoire élevée, pour construire les contigs à partir d'un graphe de De Bruijn. Mais, l'utilisation de la récursion permet de traiter tous les chemins possibles dans le graphe de De Bruijn de manière systématique, sans avoir à énumérer tous les chemins possibles même si j'ai fini par gérer les cas de plusieurs prédécesseurs et ou successeurs. Avec la récursivité, les contigs sont construits de manière plus efficace, vu que certains chemins peuvent être éliminés dès le début de la fonction récursive, sans avoir à les explorer entièrement ce qui peut améliorer les performances du programme pour les grands jeux de données.

***write_contigs_to_fasta*** : Fonction permettant de stocker les contigs assemblés dans un fichier au format FASTA.

***main*** : Fonction permettant d'utiliser le script et d'afficher dans le terminal le nombre de séquences extraites , Nombre de k-mers indexés, Nombre de k-mers filtrés, Nombre de k-mers conservés, Nombre de contigs assemblés, Le fichier dans lequel les contigs ont été ajoutés , le temps mis pour l'exécution du programme ainsi que la mémoire vive utilisé.

# RÉSULTATS

Toutes les sorties ont été obtenues avec exécuté par défaut, avec un k-mer de taille 11 et un filtre de 2.

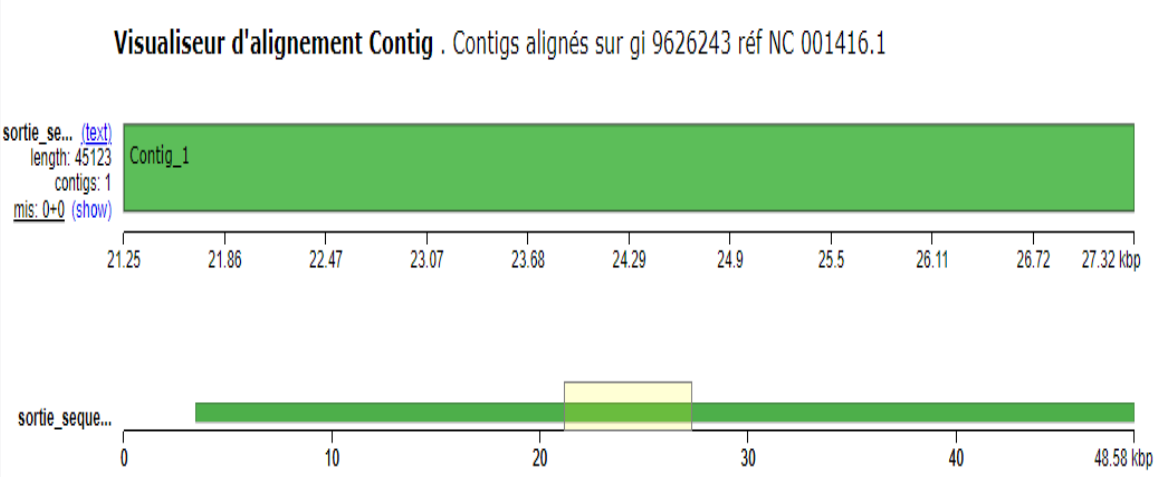
**Séquence de référence utilisée** : Reference_1.fa

## I – Assemblage avec des données sans erreurs de séquençage

Fichier résultat : sortie_sequencagesserreur.fasta

```
reference chromosomes:
    gi_9626243_ref_NC_001416.1_ (total length: 48576 bp, total length without N's: 48576 bp, maximal covered length: 45103 bp)

total genome size: 48576

gap min size: 50
partial gene/operon min size: 100
```

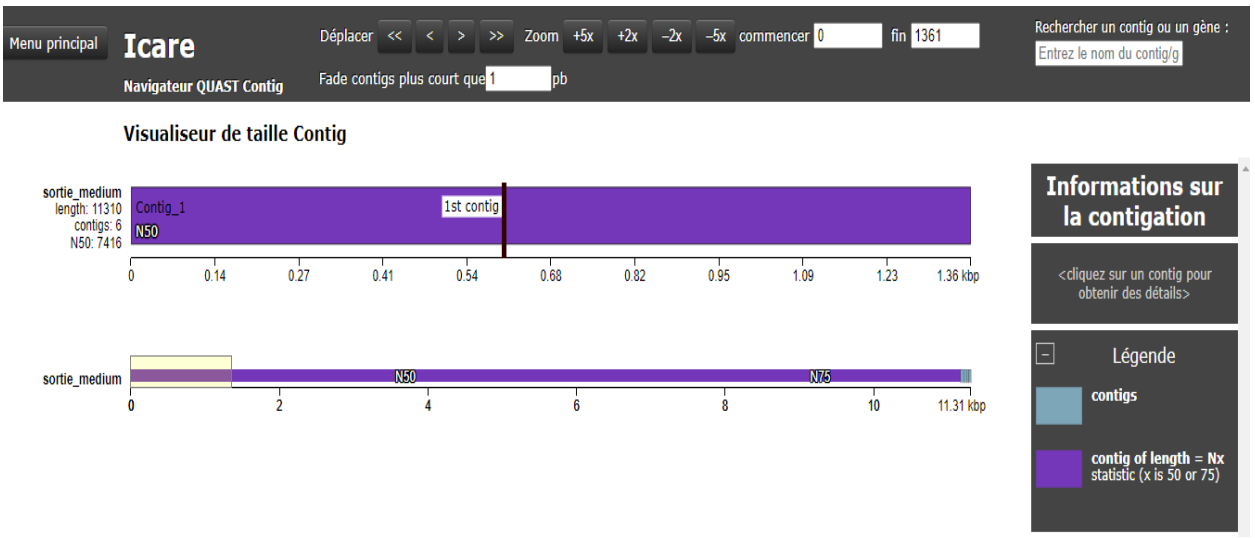
| assembly | genome fraction | duplication ratio | gaps number | genes | partial genes | operons | partial operons |
|---|---|---|---|---|---|---|---|
| sortie_sequencagesserreu | 92.850 | 1.000 | 1 | - | - | - | - |




Ces résultats nous indiquent que le génome de référence utilisé pour l'analyse est un chromosome d'une longueur totale de 48576 pb. La valeur de "maximal covered length" indique que la séquence d'assemblage couvre une grande partie de ce chromosome, soit 45103 pb, ce qui est un bon résultat.Le nombre de gaps dans l'assemblage est de 1, ce qui signifie qu'il y a un endroit dans l'assemblage où la séquence n'a pas pu être alignée. Le seuil minimal pour les gaps est de 50 pb, donc l'assemblage a un gap d'au moins cette taille.Il n'y a pas d'informations sur les gènes dans cet assemblage ce qui suggère qu'aucun gène ni opéron n'a été trouvé, ou que ces informations ne sont pas disponibles.Enfin, la fraction génomique moyenne de l'assemblage est de 92,85%, ce qui est un bon résultat pour une séquence d'assemblage unique. Le N50 de 45123 indique que la moitié de l'assemblage est constituée de contigs d'au moins 45123 pb de longueur et couvre la plupart du génome cible.Globalement, ces résultats suggèrent que l'assemblage est de bonne qualité et couvre une grande partie du chromosome de référence, mais qu'il y a un petit gap et aucune information sur les gènes.

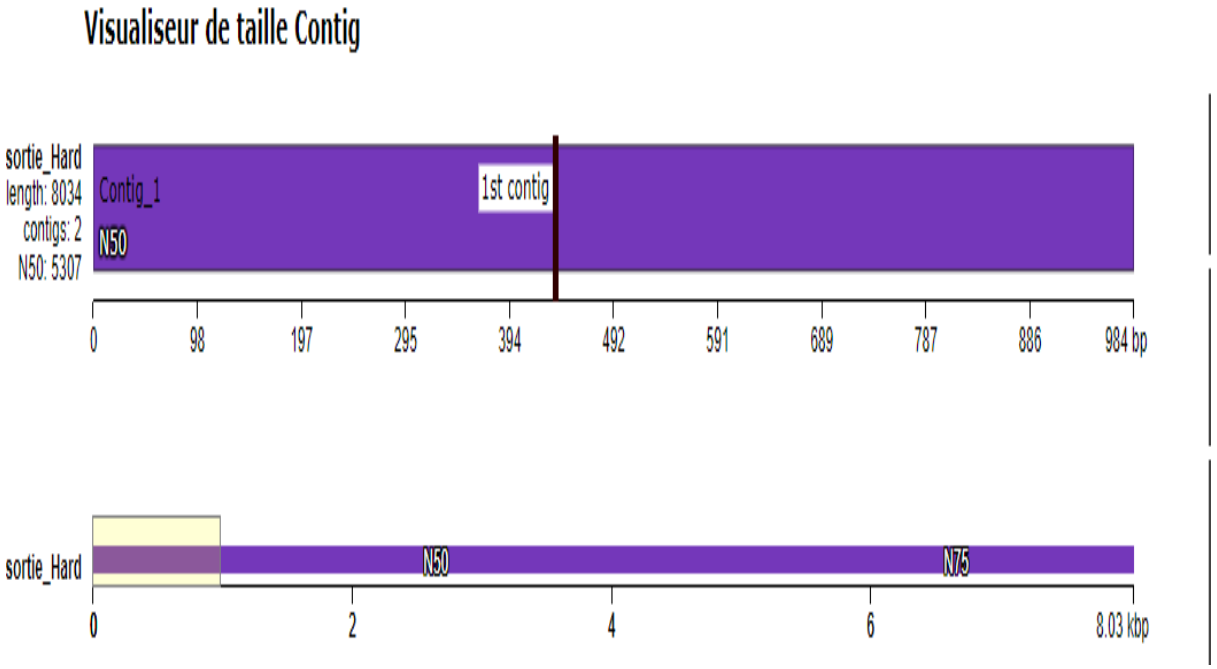
## II – Assemblage avec les données Medium.fa

Fichier résultat : sortie_medium.fasta




L'assemblage "sortie_medium" donne 6 contigs, dont seuls 2 ont une longueur supérieure à 1000 bp et un seul contig a une longueur supérieure à 5000 bp. Le plus grand contig a une longueur de 7416 bp et la longueur totale de tous les contigs est de 11310 bp. Le contenu en GC de l'assemblage est de 38,01 %, ce qui est inférieur au contenu en GC du génome de référence (49,87 %). La N50 est de 7416 pb, ce qui signifie qu'au moins la moitié de l'assemblage est contenue dans des contigs de cette longueur ou plus longs. Parmi les 6 contigs, aucun ne s'alignent sur le génome de référence.

## III – Assemblage avec les données Hard.fasta



L'assemblage est composé de deux contigs avec une longueur totale de 8034 pb. Un des contig est plus grand que l'autre, avec une longueur de 5307 pb, et est le plus grand contig de l'assemblage. Le N50 est également de 5307 pb, ce qui indique que la moitié de l'assemblage est contenue dans des contigs de cette longueur ou plus longs. La teneur en GC de l'assemblage est de 39,38%. L'assemblage n'a pas pu s'aligner sur le génome de référence, qui a une longueur de 48576 pb.

## IV – Assemblage avec les données Sequence 1.fa


```
reference chromosomes:
	gi_9626243_ref_NC_001416.1_ (total length: 48576 bp, total length without N's: 48576 bp, maximal covered length: 21133 bp)

total genome size: 48576

gap min size: 50
partial gene/operon min size: 100
```

| assembly | genome fraction | duplication ratio | gaps number | genes | partial genes | operons | partial operons |
|---|---|---|---|---|---|---|---|
| sortie_sequencage1 | 43.505 | 1.001 | 71 | - | - | - | - |

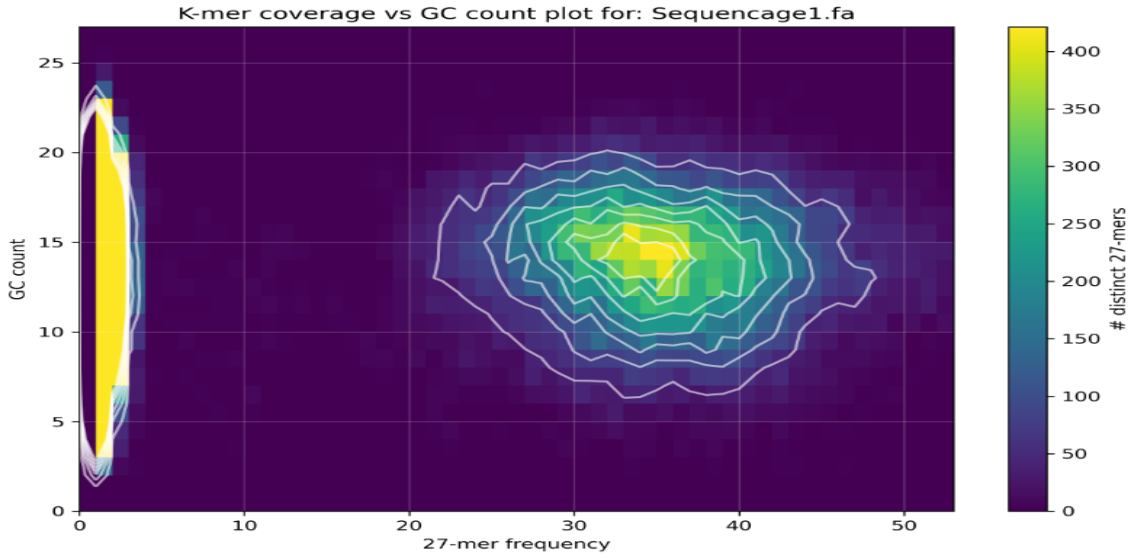
L'assemblage de **Sequence1.fa,** contient un grand nombre de contigs (586) avec une grande dispersion des tailles, allant de 39 pb à 45 261 pb. La longueur totale des contigs est de 62 259 pb, dont 45 261 pb correspondent à des contigs de longueur supérieure ou égale à 1 000 pb. La qualité de l'assemblage est moyenne car le taux de fraction génomique est de 43,5%. La comparaison avec le génome de référence montre également des différences importantes, avec une fréquence de GC de 50,53%, légèrement supérieure à celle de la référence (49,87%), et un nombre élevé de mismatches (331,24 par 100 kbp). Il n'y a pas eu de mis-assemblages détectés, mais un seul contig n'a pas pu être aligné. Le grand nombre de contigs (586) et le faible pourcentage de la fraction génomique couverte (43,5%) suggèrent que l'assemblage peut être incomplet ou présenter des régions mal assemblées.

## V – Evaluation de la qualité des 4 Assemblages


Cela signifie qu'il y a un seul fragment dans l'assemblage qui a une longueur de 48 576 paires de bases (pb). La fraction génomique moyenne, qui mesure la proportion du génome de référence qui est couverte par l'assemblage, est de 34,089%. Cela indique que l'assemblage n'a pas réussi à couvrir une grande partie du génome de référence.
De plus, il y a 192 blocs mal assemblés, ce qui signifie qu'il y a des parties de l'assemblage qui ont été assemblées de manière incorrecte ou incomplète. Ces blocs mal assemblés peuvent être dus à divers facteurs, tels que la qualité de la séquence, la complexité du génome, la présence de régions répétitives, ou des erreurs dans le processus d'assemblage.

**Exploration des pistes afin de savoir qu'elle pourrait être la cause de ces blocs mal assemblés au niveau de l'assemblage de Sequençage 1.fa** :

Afin de ne pas perdre du temps avec le nettoyage ou faire le contrôle de qualité je suis passer directement à l'outils KAT avec son module gcp qui permet de détecter des contaminant éventuelle : **srun kat gcp -t 24 -o kat_gcpseq Sequencage1.fa**




D'après ce qu'on observe sur l'histogramme de densité , on voit les k-mers d'erreur ayant une couverture faible, une propagation GC de 3 à 22, ainsi qu'un contenu authentique d'environ 1 à 4X. De plus, une observation inattendue confirme bien qu'il y a une contamination, avec une couverture d'environ 30X et un GC de 11 et 18. Étant donné que cette observation a levé

un doute, je vais essayer de procéder à l'isolement de ce contaminant avec kat_filter pour extraire les k-mers couverts par le contaminant dans les limites GC, afin d'obtenir le hachage des k-mers contenant uniquement les k-mers trouvés dans la région où le contaminant a été identifié. les lignes de commandes utilisées sont :

# Extraction des k-mers ayant une fréquence comprise entre 27 et 10000 à partir du fichier Sequencage1.fa et stocké dans le fichier appelé k-mers_contaminant.

# Permet d'identifier les k-mers potentiellement associés au contaminant présents dans notre donnée .

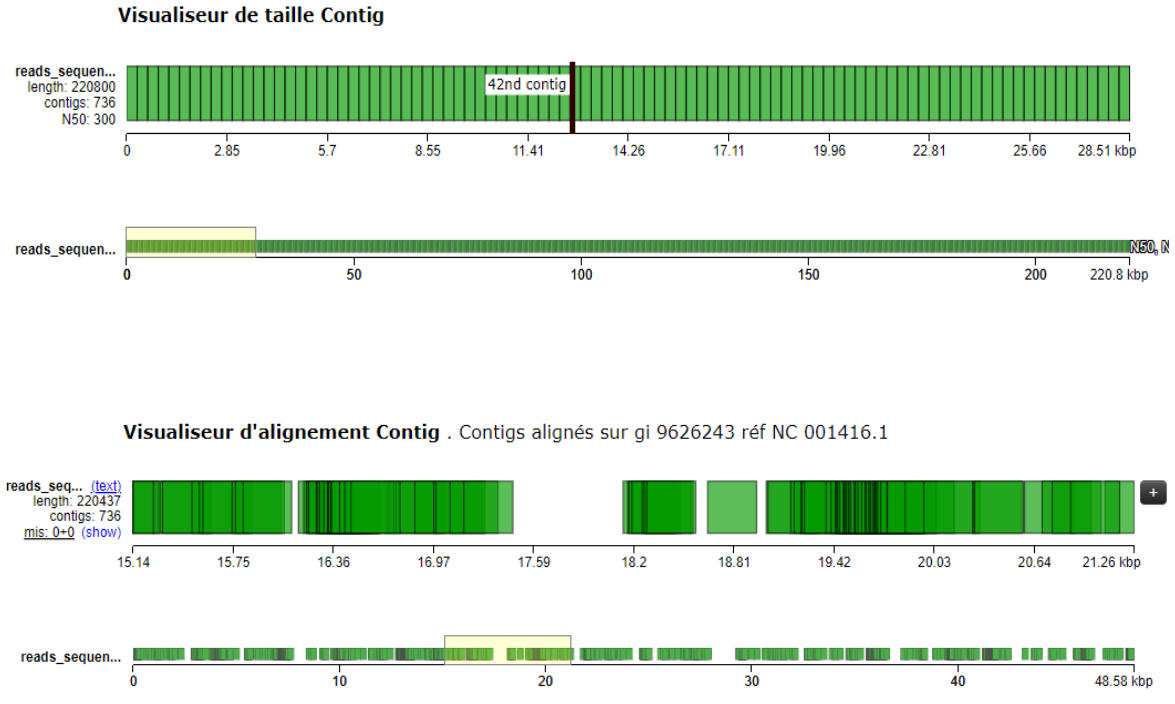
La deuxième ligne de commande permet simplement de faire le contraire de la première ligne afin de récupérer notre donnée de séquençage sans le contenant.

- srun kat filter kmer -o kmers_contaminant --low_count=27 --high_count=10000 Sequencage1.fa
- srun kat filter kmer -o kmers_sequencage_1 --low_count=1 --high_count=26 Sequencage1.fa

# obtention des reads dont proviennent les k-mers en utilisant les k-mers extraits précédemment pour filtrer les séquences d'ADN associées aux contaminants présents dans notre données. Elle utilise la commande "kat filter seq" pour filtrer les séquences qui contiennent des k-mers ayant une fréquence supérieure à 26 et inférieure à 10000, et qui ont une similarité supérieure ou égale à 50% avec les séquences d'ADN de référence des contaminants. Les séquences filtrées sont stockées dans un fichier appelé reads_contaminant. On procède de même pour la deuxième ligne de commande**.**

- srun kat filter seq -o reads_contaminant --threshold=0.5 --seq=Sequencage1.fa kmers_contaminant-in.jf27
- srun kat filter seq -o reads_sequencage1 --threshold=0.5 --seq=Sequencage1.fa kmers_sequencage_1-in.jf27

Une fois que j'ai obtenu les reads sans le contaminant j'ai relancé le quast avec cette dernière toujours avec le génome de référence . Le résultats obtenu est le suivant :



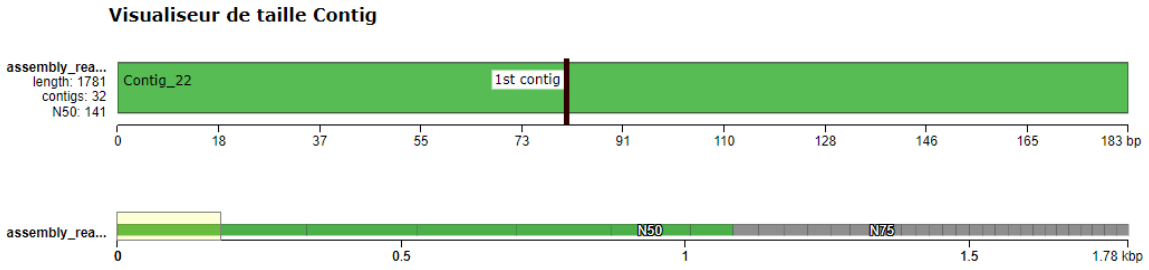
```
reference chromosomes:
    gi_9626243_ref_NC_001416.1_ (total length: 48576 bp, total length without N's: 48576 bp, maximal covered length: 40899 bp)

total genome size: 48576

gap min size: 50
partial gene/operon min size: 100


assembly               | genome    | duplication | gaps      | genes     | partial   | operons   | partial   |
                       | fraction  | ratio       | number    |           | genes     |           | operons   |
=============================================================================================================
reads_sequencage1_in   | 84.196    | 5.399       | 28        | -         | -         | -         | -         |
```

Le rapport indique que l'assemblage **reads_sequencage1_in** présente un taux de couverture du génome de 84,196%, ce qui est relativement bon. Cependant, le rapport indique également la présence de 28 (gaps) dans cet assemblage. Le rapport suggère également que le seuil de taille minimale des gaps et des gènes partiels devrait être de 50. Ces gaps peuvent indiquer des erreurs dans l'assemblage. Le seuil de taille minimale des gaps et des gènes partiels peut être utilisé pour améliorer la qualité de l'assemblage en ajustant la façon dont les gaps sont traités dans l'assemblage. Si le seuil de taille des gaps est trop petit, cela peut conduire à la création d'assemblages fragmentés avec de nombreux petits contigs, vu que les espaces qui devraient être considérés comme des gaps sont inclus dans les contigs. D'un autre côté, si le seuil de taille des gaps est trop grand, cela peut conduire à l'omission de régions importantes de la séquence d'ADN qui devraient être présentes dans l'assemblage. En sachant, à quel type d'organisme appartient la donnée et s'il s'agit d'un organisme dont le génome est bien caractérisé et que les données de séquençage sont de haute qualité et je peux prendre un seuil de taille des gaps plus petit pour obtenir un assemblage plus précis, Mais dans le cas contraire si la qualité est inférieure et que c'est un organisme dont le génome est moins bien caractérisé, un seuil de taille des gaps plus grand serait idéale pour produire un assemblage plus complet.

**<u>Ajuster au niveau de l'assemblage la taille des k-mers :</u>**

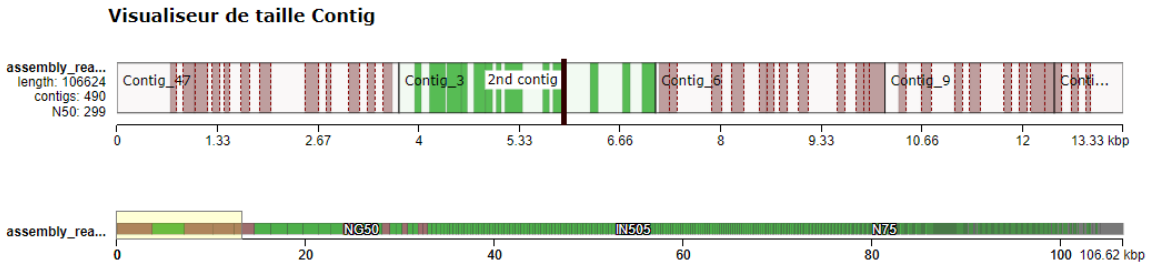
- **<u>avec une taille de k-mer 21 et un filtre de 20 on obtient :</u>**

32




32 contigs de notre donnée, avec un fraction génomique moyenne : 2,234 % et aucun blocs mal assemblés.

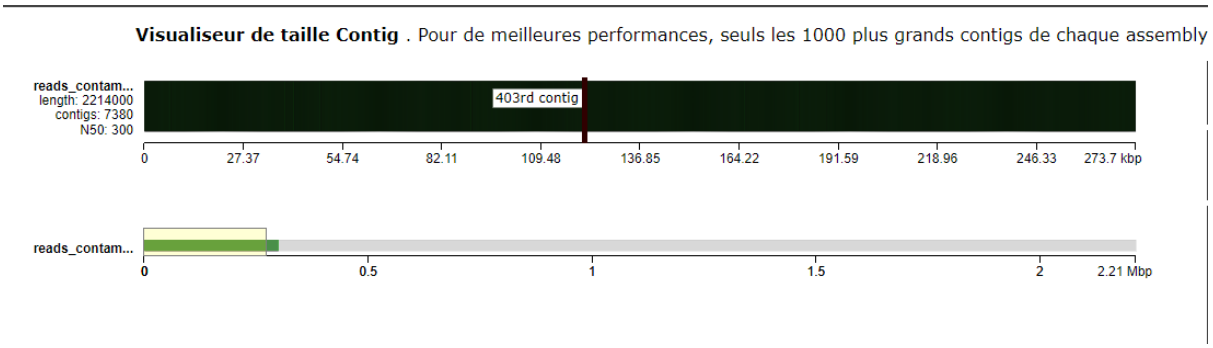
- **<u>avec un assemblage de taille de k-mer 50 sans filtre on obtient :</u>**

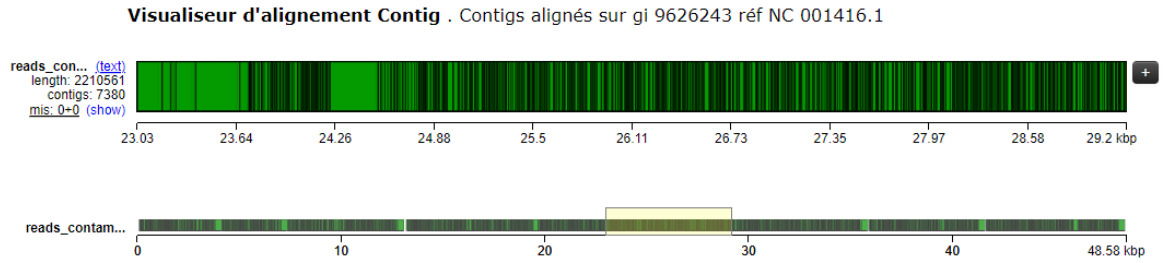



490 contigs et une fraction de couverture de 76,573 % et 127 blocs mal assemblés. Ce qui de tout évidence pourra être jugés meilleurs ou non que en fonction de nos attentes de départ compte tenu de la question qu'on se pose.

Avec toutes les variations des paramètres on remarque qu'il faut trouver un juste milieu pour avoir un résultat assez probant en fonction de ce qu'on cherche. Le fait de ne pas avoir pu gérer les tips, bulles et motifs complexes ne nous permettent pas de savoir s' il y a que quelques régions ou séquences de notre donnée qui s'aligne sur le génome de référence . Si on disposait des fichier de séquençage directement en format fastq avec les qualités de chaque séquence on aurait pu aller chercher déjà la qualité de nos donnée et savoir si de base le problème serait en amont liés à la préparation de la bibliothèque , ou lors du prélèvement de l'échantillons ou en avale s'il s'agit d'erreurs de séquençage . On procédera ensuite au nettoyage des données avec un outil comme fastp qui éliminerait les éventuels adaptateurs et autres biais avant qu'on ne procède maintenant à l'évaluation avec KAT afin d'identifier s' il y a toujours une contamination, puis l'extraire comme il a été fait ci dessus . Ce n'est qu'après toutes ces étapes qu'on pourrait procéder à l'assemblage et on saura ajuster les paramètres. En raison du temps, dont on ne dispose pas, les pistes éventuelles qu'on pourrait suivre sont limitées . j'aurai plutôt fait le KAT en premier avant de faire l'assemblage pour notre jeu de donnée,et utiliser seqkit pour essayer d'asseyer d'avoir le format fastq de nos donnée . Mais néanmoins je vais tester metaquast pour savoir a quoi correspond la contamination extraite , et comprendre peut être d'où elle provient.

## Ce que devient notre contaminant :

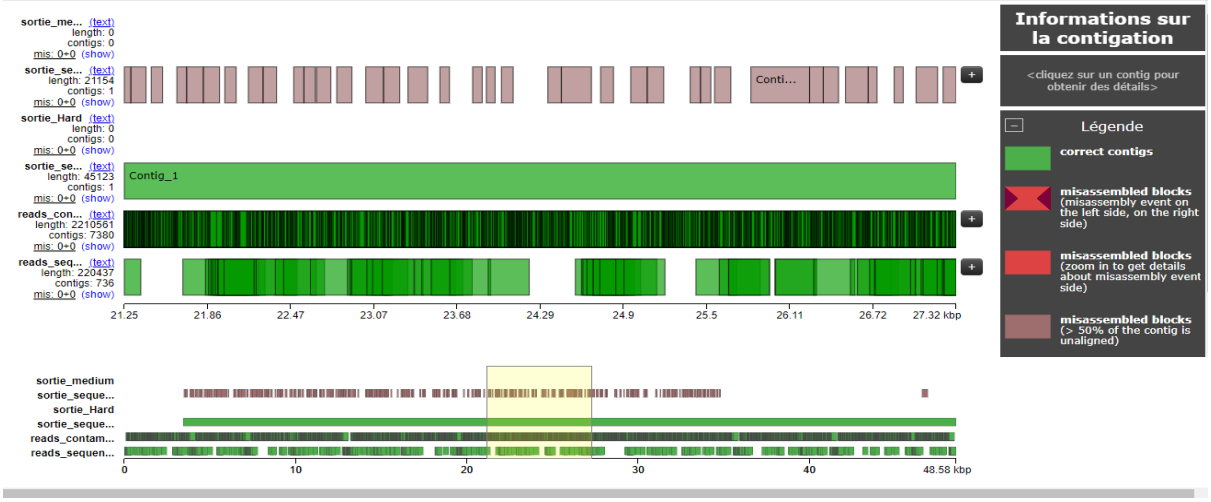





Les résultats présentent une fraction du génome de 99,212 %, ce qui indique une couverture élevée du génome de référence avec 7380 contigs. Cependant, il y a une prévalence significative de duplications, avec un rapport de duplication de 45 940. Cela suggère une représentation sûre de certaines régions du génome.En ce qui concerne les lacunes, il y a seulement 4 , gaps,  identifiées dans l'assemblage.

## Conclusion :

Pour conclure, j'ai fait un quast sur tous les reads à nouveau en ajoutant également les reads extraites,  et voici le résultats .




```
reference chromosomes:
  gi_9626243_ref_NC_001416.1_ (total length: 48576 bp, total length without N's: 48576 bp, maximal covered length: 48193 bp)

total genome size: 48576

gap min size: 50
partial gene/operon min size: 100


assembly                | genome    | duplication | gaps      | genes     | partial   | operons   | partial   |
                        | fraction  | ratio       | number    |           | genes     |           | operons   |
================================================================================================================
sortie_sequencage1      | 43.505    | 1.001       | 71        | -         | -         | -         | -         |
sortie_sequencagesserreu | 92.850    | 1.000       | 1         | -         | -         | -         | -         |
reads_sequencage1_in    | 84.196    | 5.399       | 28        | -         | -         | -         | -         |
reads_contaminant_in    | 99.212    | 45.940      | 4         | -         | -         | -         | -         |
```

L’assemblage sortie_sequencage1 à une fraction de génome assemblée de 43.505% et un ratio de duplication est de 1.001, ce qui suggère une légère présence de duplications. On a 71 gaps identifiés dans l'assemblage. Ensuite après avoir identifié et extrait la contamination de cet assemblage on a obtenu l’assemblage, reads_sequencage1_in qui à une fraction de génome assemblée est de 84.196%, ce qui est bien mieux que la fraction d’origine, le ratio de duplication par contre est de 5.399, ce qui suggère une présence plus importante de duplications par rapport aux autres séquences et surtout à l’assemblage de départ. Les gaps sont passer à 28 au lieu de 71, ce qui est aussi un meilleurs résultat. l’assemblage reads_contaminant_in issu toujours de notre assemblage de départ séquencage1, et qui constitue le contaminant de ce dernier donne une fraction de génome assemblée de 99.212%, avec un ratio de duplication de 45.940, ce qui indique une présence très élevée de duplications avec seulement 4 gaps, identifiées dans cet assemblage. Il semble bien que l’assemblage soit mieux aligné sur le génome de référence comparé à l'assemblage même d’origine. Ce qu’il faut pouvoir résoudre est donc le niveau de duplication, son origine et comment l'intégrer pour que lors de l’assemblage ces erreurs puissent être corrigées. En ce qui concerne l’assemblage sortie_sequencagesserreu la fraction de génome assemblée est de 92.850% avec un ratio de duplication de 1.000, indiquant une absence de duplications avec un seul gaps qui s'aligne parfaitement sur le génome de référence.

Sachant que les duplications sont très souvents causées par des répétitions de certains régions répétés dans le génome de référence , ou des erreurs de séquençage lors du séquençage avec des duplications artificielles dans les donnée, ou encore des contamination si les échantillons utilisés pour le séquençage contiennent des séquences provenant d'autres organismes.

Pour résoudre ce problème de duplication , on peut procéder à l’ajout de données de séquençage supplémentaires, provenant de différentes technologies ou de différentes bibliothèques d'ADN, pouvant aider à résoudre les duplications et à améliorer la qualité de l'assemblage. Également améliorer les outils d’assemblage ce qui nous concerne dans le cadre de ce projet, en utilisant le graph de De Bruijn pour gérer les régions répétées. Ce sont ces régions répétées dans le génome qui peuvent conduire à des nœuds multiples dans le graphe de De Bruijn et ces nœuds peuvent être détectés en identifiant les bifurcations ou les branches dans le graphe, où plusieurs arêtes se connectent à un même nœud.On en vient donc à la gestion de ces bifurcations qui consiste à simplifier le graphe en fusionnant les nœuds redondants et en éliminant les arêtes( relation entre deux k-mer k-1 qui se chevauchent ) non significatives, c’est à dire que les nœuds qui représentent les mêmes séquences peuvent être fusionnés en un seul nœud, et les arêtes avec une faible fréquence peuvent être supprimées, ce qui revient à l’importance du choix du seuil des khmer à filtrer dans le programme de l’assemblage. Pour finir, il faut noter que la gestion des duplications peut être un défi complexe et que différentes approches peuvent être nécessaires en fonction de la nature des duplications et des données disponibles.