Tiphaine Denne

Projet Nextflow

M2 Bioinformatique et Biologie des Systèmes

Table des matières

Introduction

Tout au long de cette unité d'enseignement, nous avons découvert deux nouveaux outils bioinformatiques : Galaxy et Nextflow. Galaxy est une plateforme internet permettant des traitements bioinformatiques de façon relativement simplifiée et intuitive. Nextflow est un cadre de workflow qui peut être utilisé par un bioinformaticien pour intégrer tous ses scripts dans un seul pipeline cohérent qui est portable, reproductible, évolutif et contrôlé.

Durant l'enseignement, nous nous sommes familiarisé avec ces différents outils et pour ce faire, nous avons dû utiliser un cluster, Genotoul, qui nous permet d'accéder à de hautes performances de calculs.

Notre projet, que nous réaliserons à l'aide de Nextflow, consiste à réaliser un traitement RNAseq à partir de données sur les tomates. Afin de réaliser cela, nous avons utilisé une pipeline bioinformatique nommée nf-core/rnaseq qui permet d'analyser des données séquences d'ARN obtenues d'organismes avec un génome de référence et annotation.

Préparation du job

Préparation des données

Nos données sont composées de différents fichiers :

- Un fichier fasta regroupant la séquence du génome de référence pour le chromosome 6
- Un fichier gtf correspondant à l'annotation faite sur ce génome pour le chromosome 6
- 4 fichiers fastq portant sur le chromosome 6. Les fichiers fastq comportent différentes informations telles que la séquence du génome ainsi que la qualité associée.
    - 2 sont des réplicats sur des génomes WT
    - 2 sont des réplicats sur des génomes MT

Nous devons donc créer un fichier csv qui indique nos 2 échantillons composés chacun de deux fichiers fastq et une information concernant le brin.

lancementNextflow.sh | **tomates.csv**

```
1  sample,fastq_1,fastq_2,strandedness
2  1,/work/bleuet/projet/data/MT_rep1_1_Ch6.fastq.gz,/work/bleuet/projet/data/MT_rep1_2_Ch6.fastq.gz,unstranded
3  2,/work/bleuet/projet/data/WT_rep1_1_Ch6.fastq.gz,/work/bleuet/projet/data/WT_rep1_2_Ch6.fastq.gz,unstranded
```

Ceci est donc le fichier écrit.

Bien sûr, toutes nos données sont regroupées dans un répertoire data par question d'organisation.

Préparation du fichier bash de lancement Nextflow

Différents paramètres sont à prendre en compte lors de la création du fichier bash permettant le lancement de notre job. J'ai écrit le fichier suivant :

lancementNextflow.sh

```
#!/bin/bash
#SBATCH --time=23:59:59
#SBATCH -J TiphaineDenne
#SBATCH -e error.out
#SBATCH --mem=6G
#SBATCH --cpus-per-task=8
#SBATCH --mail-type=BEGIN,END,FAIL

#purge of previous modules
module purge

#load modules
module load bioinfo/nfcore-Nextflow-v21.04.1

#nextflow run command
nextflow run nf-core/rnaseq -r 3.4 -profile genotoul \
--fasta /work/bleuet/projet/data/ITAG2.3_genomic_Ch6.fasta \
--gtf /work/bleuet/projet/data/ITAG2.3_genomic_Ch6.gtf \
--input /work/bleuet/projet/data/tomates.csv \
```

On y indique que la durée du job doit être de 1 jour maximum, le nom du job, la sortie contenant les erreurs, la mémoire maximale de 6G, le nombre de CPUs par tâche de 8 et enfin ce qui est notifié par mail à l'utilisateur.

Ensuite on fait une purge de tous les modules et on charge la bonne version du module nf-core qui nous intéresse.

Enfin, nous pouvons donc faire tourner le pipeline sur nos données *via* Nextflow en indiquant où se trouvent les fichiers fasta, gtf et csv qui nous sert d'input (*cf* partie Préparation des données pour voir sa construction). La version du pipeline utilisée est la 3.4.

Lancement du job et suivi

Ainsi, nous pouvons lancer le job en utilisant la commande :

```
sbatch lancementNextflow.sh
```

Au départ, les fichiers tomates.csv et lancementNextflow.sh comportaient quelques erreurs qui faisaient donc échouer le run.

Lorsque l'on lance un job, on utilise ensuite la commande :

```
seff job_ID
```

qui permet de suivre l'évolution du job. Cette dernière nous retourne l'état de notre requête, à savoir en cours, fini ou bien échouée. Lorsque le job a échoué, cela signifie qu'il y a des erreurs dans nos fichiers de départ et donc soit dans notre csv qui pointe vers les fastq ou bien dans le script de lancement Nextflow. Un fichier est retourné comportant les erreurs : error.out.

On modifie donc ensuite si besoin les différents fichiers et on relance un job. Avant de relancer un job, on supprime les différentes sorties du job précédent pour être sûr de ne pas avoir de mauvais fichiers :

```
rm slurm-job_ID.out
```

NB : j'aurais pu utiliser la commande resume qui permet de reprendre un ancien job après modification des fichiers suite à un fail plutôt que relancer un nouveau job mais je n'ai pas réussi à la faire fonctionner et j'ai donc finalement choisi la solution qui me semblait le plus rapide.

Sorties

Une fois que le job est correctement réalisé, nous obtenons différents résultats. D'abord dans le répertoire courant nous avons trois fichiers :

- output.out et slurm-job_ID.out qui correspondent aux différents traitements effectués durant le job
- error.out qui est donc vide car le job s'est déroulé sans erreurs

Ensuite, un répertoire results a été créé. Il est composé de plusieurs sous-répertoires :

- fastqc
- genome
- multiqc
- pipeline_info
- star_salmon
- trimgalore

Répertoire fastqc

Ce répertoire retourne les fichiers FastQC pour chaque échantillon. Ce format permet de regrouper différents contrôles de qualité sur nos données. Il regroupe entre autres la qualité des séquences par base, le pourcentage de GC, la distribution de la longueur de séquence, etc.

Répertoire genome

Ce répertoire regroupe différents formats de fichiers référant au génome comme un fichier au format bed ou gtf par exemple.

Répertoire multiqc

MultiQC est un outil de visualisation qui génère un seul fichier html résumant tous les échantillons du projet. La plupart des résultats obtenus avec les différents outils du pipeline sont en fait visibles sur ce rapport (nommé multiqc_report.html). Cela nous donne aussi un répertoire multiqc_data qui lui comporte des statistiques approfondies. On retrouve donc dans le rapport les résultats de FastQC, Cutadapt, SortMeRNA, STAR, Salmon, SAMtools, etc. Ce sont ces résultats là que nous pourrons donc utiliser pour interpréter notre analyse.

Répertoire pipeline_info

Ce répertoire est composé de rapports variés portant sur l'exécution du pipeline.

Répertoire star_salmon

Ce répertoire regroupe les résultats de deux logiciels : STAR et Salmon.

STAR (Spliced Transcripts Alignment to a Reference) est un aligneur de lecture conçu pour la cartographie sensible à l'épissage typique des données de séquençage d'ARN.

Salmon est un outil de quantification ultra-rapide des transcrits à partir de données RNA-seq.

Ces deux logiciels retournent beaucoup de fichiers tels que des matrices de comptage obtenues *via* Salmon, ou bien des fichiers bam et bai.

Répertoire trimgalore

Trim Galore! Est un outil qui permet d'effectuer la qualité et le découpage de l'adaptateur sur les fichiers fastq. Donc le job retourne dans ce dossier différents rapports, un pour chaque réplicat, qui contiennent beaucoup d'informations telles que le mode de trimage, la version de cutadapt utilisée, l'adaptateur utilisé et les séquences supprimées.

Résultats

Afin de visualiser les résultats, il suffit donc de faire la commande suivante :

---

firefox multiqc_report.html

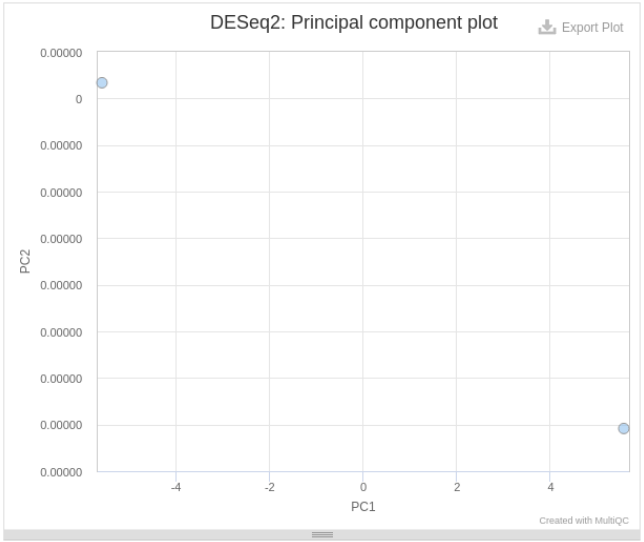
Qui nous ouvre donc le fichier html. Ce fichier est organisé en plusieurs parties qui représentent les résultats obtenus avec les différents outils.

General Stats

La première partie de ce fichier représente des statistiques générales telles que le pourcentage de rRNA, de duplication, de reads mappé, de GC, *etc*. en fonction des échantillons.

STAR_SALMON DESeq2 PCA plot

Une ACP (Analyse en Composantes Principales) permet de réduire l'information de variables à composantes.



Cette ACP a été obtenue avec la librairie DESeq2 sous R.

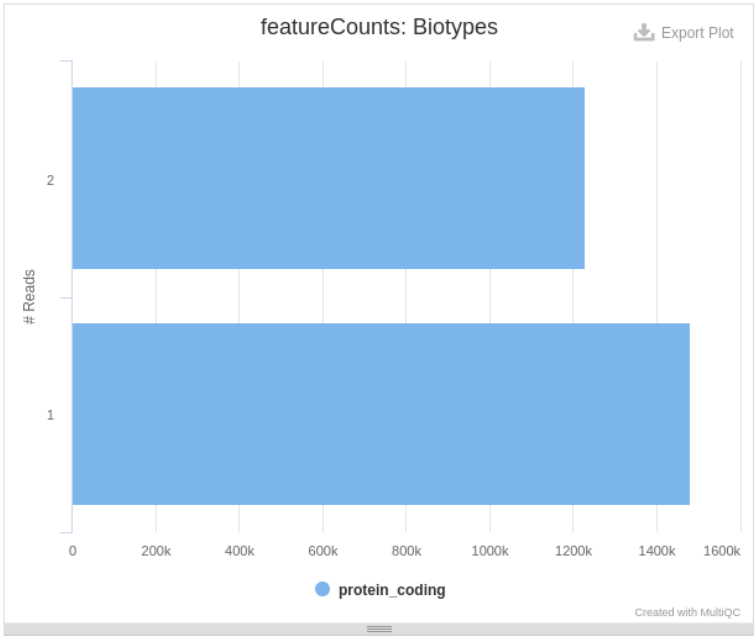
STAR_SALMON DESeq2 sample similarity

DESeq2 a également permis de construire une heat map à l'aide d'un clustering avec une distance euclidienne des échantillons.

Ces deux résultats ne sont pas très intéressants car nous avons que deux échantillons. Donc ils montrent juste que les échantillons sont différents.

Biotype Counts

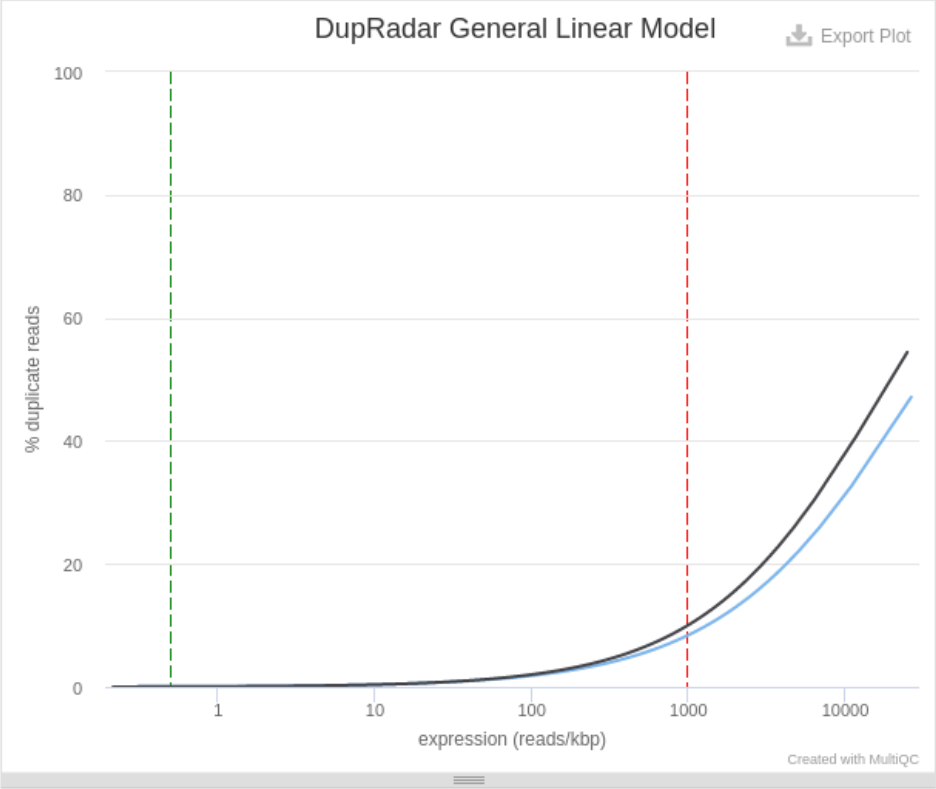
Ce résultat montre les chevauchements entre reads. Cela a été obtenu avec featureCounts.



Le chevauchement est relativement important mais on voit quand même que l'échantillon 2 possède une longueur codante de protéines plus longue.

DupRadar

Cet outil permet un contrôle qualité du taux de duplication pour les jeux de données de RNAseq. Il en ressort un plot qui montre un résumé des distributions de duplication de gènes.



Les gènes hautement exprimés ont plus de reads dupliqués. Cet effet est plus important pour l'échantillon 2.

Picard

Cet outil permet de manipuler les données de séquençage à haut débit. Il permet notamment de représenter les nombres de reads catégorisés par leur état de duplication.

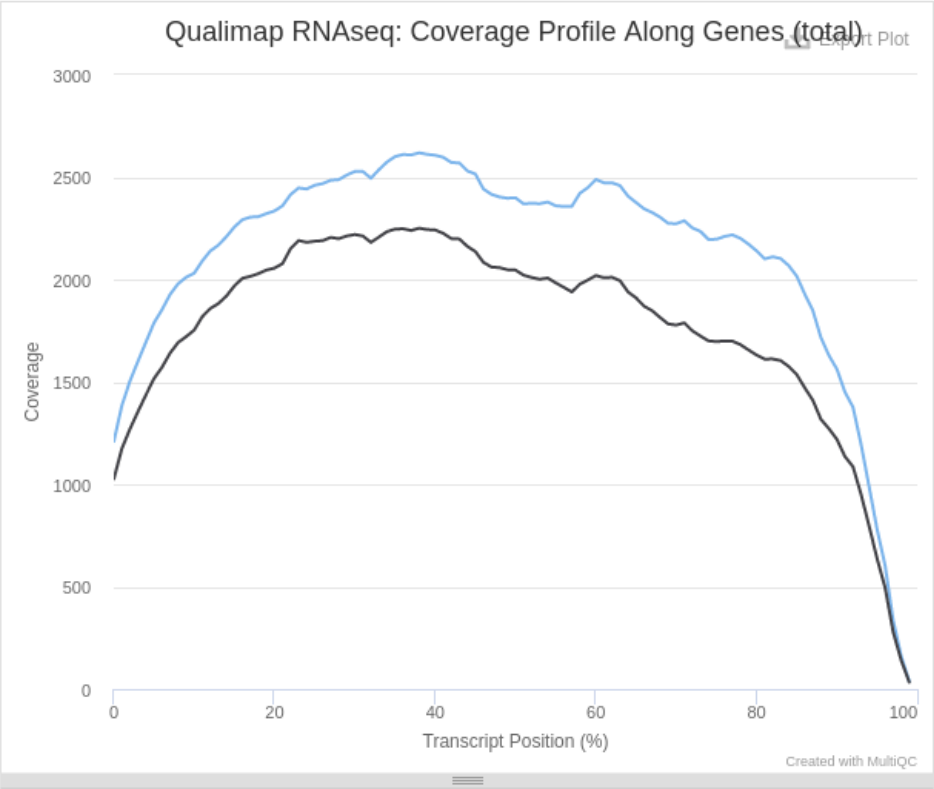
Preseq

Preseq estime la complexité d'une librairie en montrant combien de reads uniques supplémentaires sont séquencés pour augmenter le nombre total de reads. Selon les courbes données, il semblerait que l'échantillon 2 possède une moins bonne complexité.

QualiMap

Cette application permet de faciliter le contrôle qualité de l'alignement des données séquencées. Cela classe par exemple les reads mappés comme étant d'origine exonique, intronique ou de régions intergéniques. Pour les deux échantillons, environ 83% des reads sont d'origine exoniques, environ 8% introniques et 8% intergéniques.

On peut également comparer la distribution moyenne de profondeur de couverture à travers tous les transcrits mappés, ce qui nous donne pour nos données ce graphique :

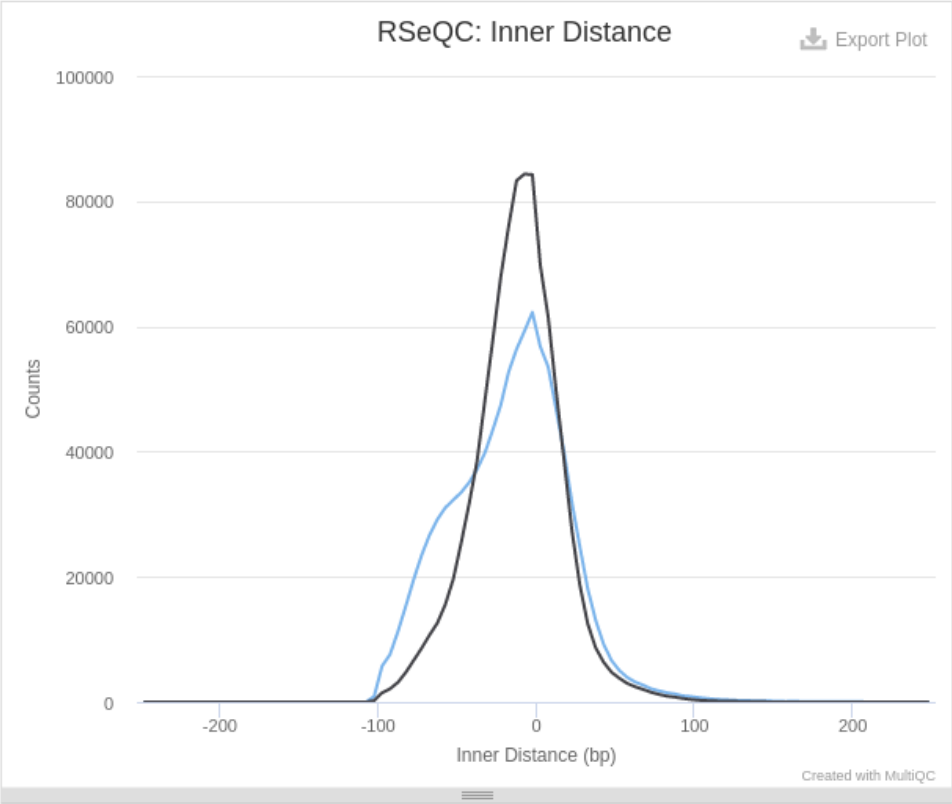


On voit que pour l'échantillon 2 (noir) la profondeur de couverture est toujours plus faible que pour l'échantillon 1.

RSeQC

Ce package réunit de nombreux modules utiles pour évaluer les données RNAseq. Tout d'abord la distribution des reads est analysée. Cela permet de calculer comment les reads mappés sont distribués, comme vu juste avant avec QualiMap en plus précis, selon les différentes caractéristiques : CDS_Exons, Introns, TSS_up_1kb, TSS_up_5kb, etc. Les distributions semblent assez similaires entre les deux échantillons.

La distance intérieure entre deux reads RNA paired est calculée puis est retourné un graphique montrant cela :



Pour l'échantillon, cette distance est plus élevée que pour l'échantillon 1. Ce qui, je pense, coïncide avec le fait que la profondeur de couverture est inférieure dans l'échantillon 2.

Un script python permet ensuite de calculer combien de positions alignées ont un nombre certain d'exacts duplicats. Un outil se nommant junction annotation compare les jonctions détectées à un modèle de référence. Un deuxième outil, junction saturation, compte le nombre de jonctions connues observées dans chaque jeu de données. S'il manque des jonctions, cela peut affecter l'analyse. On voit que le nombre de jonctions en fonction des reads est supérieur pour l'échantillon 1.

Infer experiment compte le pourcentage de reads et reads pairs qui matchent le sens du brin et des transcrits chevauchant. Pour les deux échantillons, il y a 50% de sens et 50% d'antisens.

Samtools

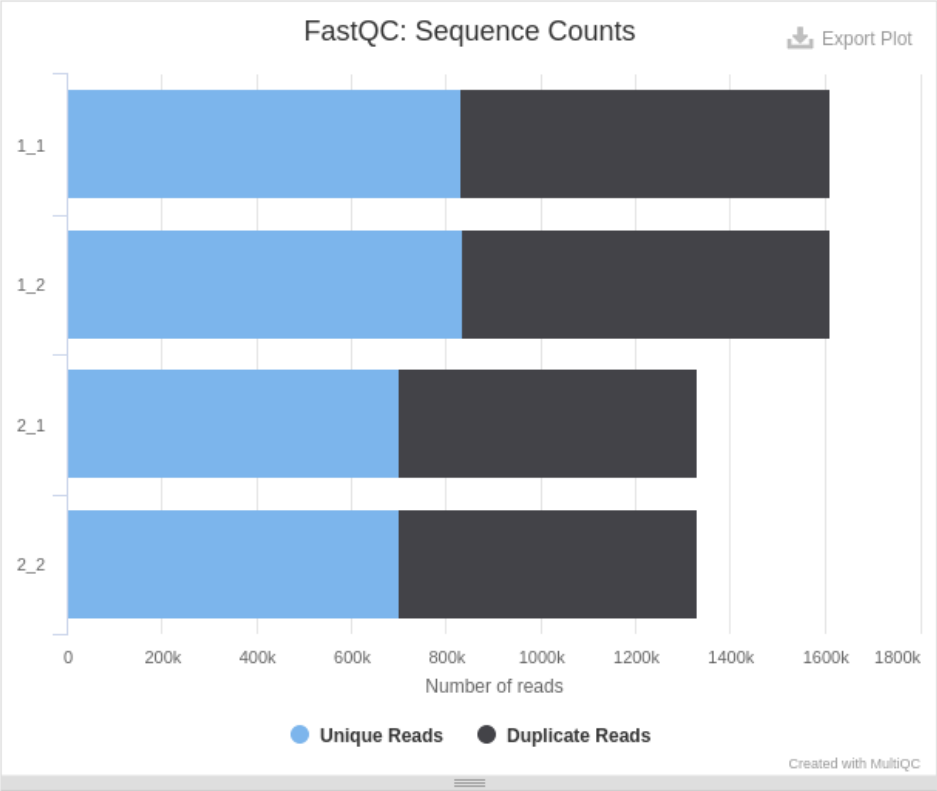
Samtools est une suite de programmes pour les données de séquençage haut débit. Cela nous retourne différentes observations, entre autres que le nombre de reads de l'échantillon 2 est inférieur à celui de l'échantillon 1,

FastQC

Nous disposons de deux types de résultats pour le FastQC : raw et trimmed. Lorsque les reads sont trimmés, cela signifie que l'on a retiré l'adaptateur, entre autres, donc on s'intéresse vraiment qu'aux séquences, ce qui fait que ces résultats sont plus fiables. Donc nous allons décrire uniquement les résultats obtenus avec FastQC trimmed.

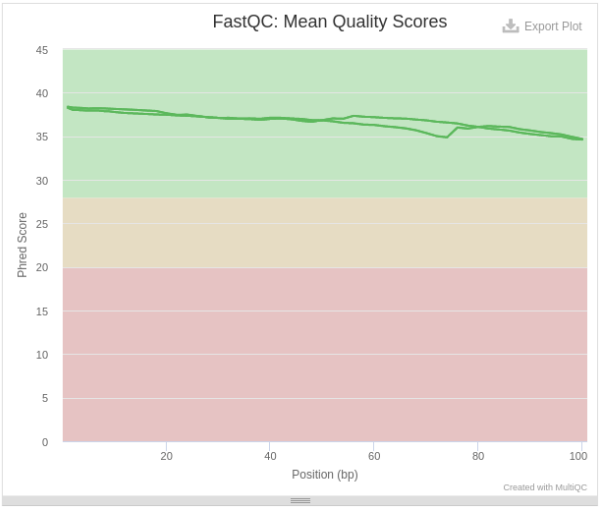
Pour obtenir ces séquences trimmées, il faut utiliser cutadapt qui permet d'éliminer les adaptateurs, les primers, les queues poly-A et d'autres types de séquences non voulues. Les longueurs des séquences trimmées sont similaires pour les deux échantillons.

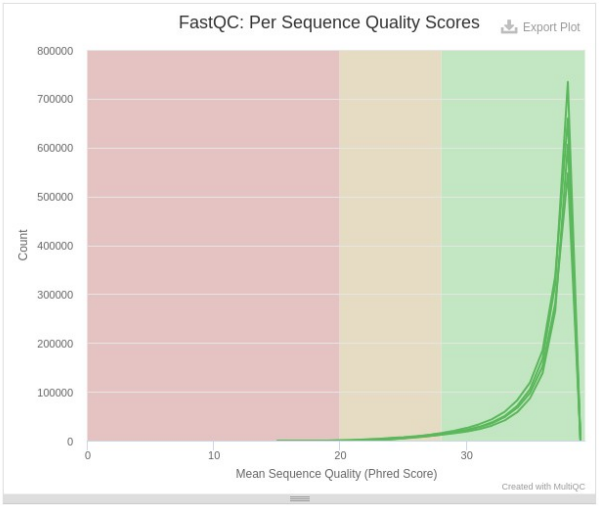
Une fois trimmés, on peut donc comparer les reads des différents échantillons.



Ce graphique obtenu à l'aide d'un outil de FastQC nommé Sequence counts montre que le nombre de reads (trimmés) est supérieur pour les réplicats de l'échantillon 1. Les réplicats de l'échantillon 2 possèdent légèrement plus de reads uniques (presque 53% contre 51% pour les réplicats de l'échantillon 1). Il serait intéressant de réaliser un test statitique pour voir si cette valeur est significative.

La qualité des séquences est globalement bonne pour les deux échantillons. De même le score de qualité par séquence est similaire et très bon.



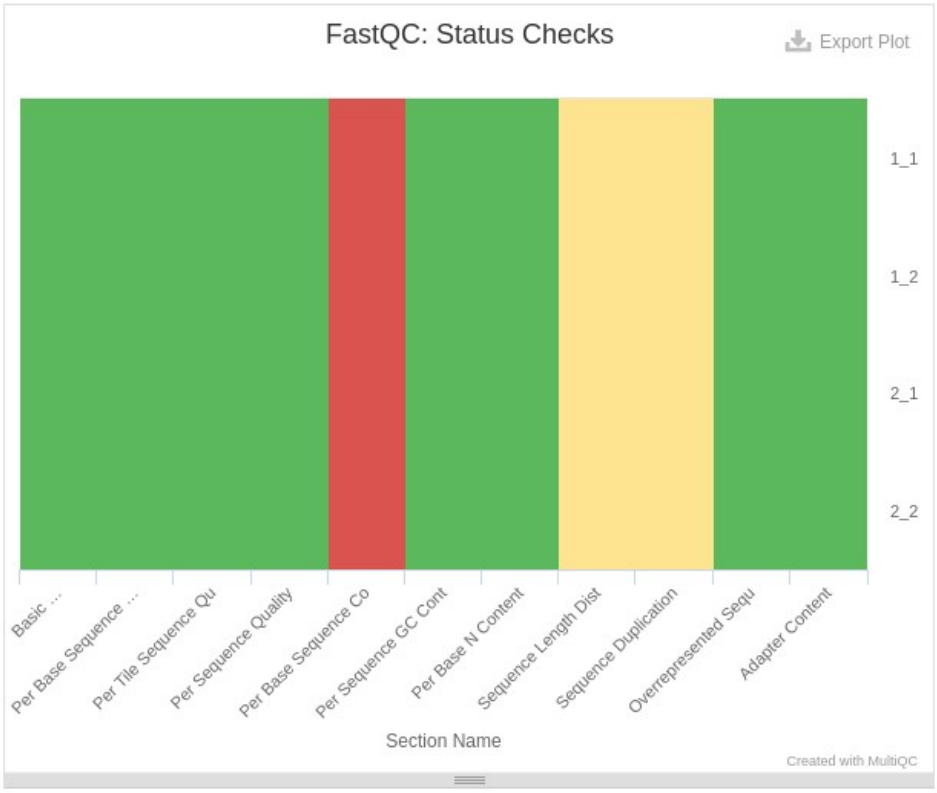


Le pourcentage de GC est similaire dans tous les réplicats également. Le pourcentage de N est également très bon et proche pour tous les échantillons. La distribution des longueurs des

séquences montre que les reads font tous moins de 100 paires, ce qui correspond surement à la longueur que le séquenceur a comme valeur. Si elles font moins de 100 pb c'est parce qu'elles ont été trimmées et donc des segments de la séquence ont été enlevés.

Les niveaux de duplication des séquences des quatre réplicats sont très proches. Tous les échantillons ont moins de 1% de reads avec des séquences sur-représentées. Aucun des échantillons n'a été trouvé avec une contamination par l'adaptateur de plus de 0.1%.

Une figure donnant un récapitulatif de tout ce qui a été décrit :



On voit bien que tous les échantillons ont des valeurs identiques pour chacune des caractéristiques mesurées.

Ce qu'on peut conclure sur toute cette partie de résultats de FastQC c'est que les quatre échantillons ont des résultats très similaires les uns par rapport aux autres. Hormis le fait qu'il y ait moins de reads provenant de l'échantillon 2.

nf-core/rnaseq

Deux derniers types de données sont retournées dans ce fichier html, plus en rapport avec l'exécution du pipeline :

- Software Versions : sont indiquées les versions de tous les softwares utilisés durant le pipeline
- Workflow Summary qui représente les différentes options utilisées pour lancer Nextflow, le chemin des inputs, la mémoire maximale, etc.

Pour revenir sur l'ensemble des résultats, la seule différence majeure entre les génomes de types WT et MT sont que les WT possèdent plus de reads. Donc les MT ont une partie du génome qui est manquante. Il serait intéressant d'approfondir cette analyse en identifiant les gènes manquants.

Discussion – Conclusion

Ce projet, et cette UE en général, était très intéressant car il nous a permis de découvrir de nouveaux outils. J'ai également beaucoup aimé l'utilisation de génotoul car, étant donné que nous devrons sûrement utiliser dans notre carrière un cluster de calculs, c'était très utile.

Un point négatif à noter tout de même serait l'utilisation de firefox pour les fichiers HTML. Il est très étrange de voir à quel point le temps de latence est élevé lorsque l'on ouvre le fichier HTML avec firefox depuis le cluster, ce qui rend l'interprétation des résultats très fastidieuse.

Selon tous les résultats observés, il semblerait qu'il y ait peu de différences entre les deux échantillons mais bien sûr il faudrait faire des analyses supplémentaires, notamment des tests statistiques pour voir si les différences qui existent sont significatives.

Les analyses montrent une très bonne qualité de séquençage pour les deux échantillons.

Nextflow est donc un outil très puissant qui a permis une analyse très complète en très peu de temps. Le fait de pouvoir développer et utiliser ainsi des pipelines est très intéressant et performant.