

## Projet

# Hackathon

Enseignants :  
Frédéric Lemoine, Thomas Cokelaer

BAUMANN Ambre  
GOULET Lindsay  
MARCHMENT George  
SEBE Clémence

M2 AMI2B  
Université Paris Saclay

# Table des matières

---

<b>Introduction</b>	<b>3</b>
<b>1 Matériels &amp; Méthodes</b>	<b>4</b>
1.1 Fonctionnement général . . . . .	4
1.2 Étapes du workflow . . . . .	4
1.3 Analyse différentielle . . . . .	5
1.4 Workflow final . . . . .	6
<b>2 Résultats</b>	<b>7</b>
2.1 Résultats globaux . . . . .	7
2.2 Contrôle qualité . . . . .	7
2.3 Résultats d'analyse avant filtrage . . . . .	8
2.4 Résultats d'analyse après filtrage . . . . .	10
<b>3 Discussion</b>	<b>11</b>
3.1 Analyse des résultats . . . . .	11
3.2 Comparaison avec les papiers . . . . .	11
3.3 Conclusion et perspectives . . . . .	12
<b>Références</b>	<b>13</b>

La reproductibilité est un principe majeur de la méthode scientifique. C’est ceci qui nous permet d’acquérir davantage de connaissances, avec un niveau de certitude plus ou moins élevé, sur le monde qui nous entoure et les éléments qui le composent. Elle peut se définir comme la qualité d’une étude, expérience ou analyse à reproduire les mêmes résultats avec un haut degré de fiabilité lorsqu’on le répète. Depuis les dernières dizaines d’années cependant, les chercheurs ont été confrontés à la non reproductibilité de leurs résultats et ceux de leurs pairs. Ceci a mené à une crise de la reproductibilité [Baker, 2016]. La bioinformatique, étant un domaine scientifique avec un plus grand contrôle sur les paramètres et l’environnement de travail, n’est pas à l’abri de cette non reproductibilité. Les analyses et expériences effectuées peuvent subir de petites variations, sous forme de mises à jour de logiciel ou changements de système d’exploitation. Ces petites variations peuvent entraîner des modifications de l’analyse, changeant ainsi les résultats.

Par exemple, le *Sanger Companion* pipeline est composé de 39 outils et bibliothèques bioinformatiques indépendants. Des mises à jour fréquentes de logiciel peuvent modifier le fonctionnement de ces outils ou bibliothèques, potentiellement arrêter la bonne utilisation du pipeline [Di Tommaso et al., 2017]. Une autre difficulté avec l’utilisation de ce type de pipeline est qu’il est possible que ces outils aient des dépendances incompatibles. Imaginons un outil nécessitant `Python 2.7` et un autre nécessitant `Python 3.8`, ceci obligerait l’utilisateur à manuellement choisir quelle version utiliser et à quel moment. Pour faciliter la création de pipeline bioinformatique plus reproductible, un utilisateur a maintenant la possibilité de développer des pipelines avec des workflow frameworks (e.g. Galaxy ou Nextflow).

Au cours de ce projet, on a étudié deux articles [Harbour et al., 2013], [Furney et al., 2013] qui effectuent des analyses RNA-seq sur des patients souffrant d’un mélanome choroidien (un cancer de la choroïde). Pour faire cela, ils comparent deux groupes de patients : ceux ayant une mutation sur le gène *SF3B1* et les patients qui n’en ont pas. Ensuite, ils analysent notamment les gènes qu’on considère comme différentiellement exprimés (DE).

Le but de ce projet est de développer un workflow d’analyse RNA-seq pour effectuer le même type d’analyses que celles effectuées dans les deux études, et puis comparer les résultats obtenus avec ceux des études. Pour ce faire, on a à notre disposition les mêmes données brutes du transcriptome que les études précédentes [NCBI, 2013a]. Pour effectuer l’analyse RNA-seq correspondante, on a développé un workflow `Nextflow` [Di Tommaso et al., 2017]. Un workflow framework qui permet la création de pipelines d’analyse bioinformatique robuste, reproductible, flexible et échelonnable (*scalable*). Avec l’utilisation de `Nextflow`, on utilise aussi `Docker` [Merkel, 2014], une plateforme informatique qui permet la création de conteneurs de logiciels. `Nextflow` facilite l’utilisation de ces conteneurs logiciels, ce qui permet ainsi un plus grand contrôle des différentes versions de logiciels utilisés, garantissant par la suite une meilleure reproductibilité.

Dans la suite on présentera le fonctionnement du workflow développé, les résultats que l’on obtient et finalement une discussion sur ces résultats.

## 1.1 Fonctionnement général

On a développé un workflow d'analyse RNA-seq en **Nextflow** sous la version DSL2 (domain-specific language). Le workflow est composé de modules, sous-workflows et workflow. Un module est un fichier `.nf` comportant uniquement un process. Un sous-workflow correspond à l'assemblage de plusieurs modules pour effectuer une suite de process et effectuer une tâche spécifique. Le workflow en entier inclut les sous-workflows et les modules. Décomposer le workflow de cette façon permet l'augmentation de la réutilisation du code (dans un autre projet par exemple) et le rend plus modulable en changeant facilement les éléments qui le composent.

Notre workflow utilise de nombreux outils bioinformatiques distincts qui peuvent ne pas fonctionner ensemble en fonction de leurs dépendances respectives. Pour palier à ce problème, chacun de ces outils sont installés dans des containers **Docker** distincts, ce qui permet ainsi une plus grande réutilisabilité et reproductibilité de notre workflow. Dans chacun de ces containers, on retrouve un outil avec une version spécifiée et qui sera toujours la même. Chaque process utilisant un outil appellera l'image **Docker** correspondante et sera son environnement d'exécution.

Le workflow développé peut être lancé en entier (du téléchargement des données aux résultats de l'analyse différentielle). Il y a aussi la possibilité de spécifier seulement les étapes que l'on souhaite réaliser.

## 1.2 Étapes du workflow

La première partie de notre workflow consiste en le téléchargement de toutes les données nécessaires pour l'analyse RNA-seq. On les a inclus dans un premier sous-workflow. Ce sous-workflow contient deux étapes. Le téléchargement des échantillons est réalisé avec la méthode **fasterq-dump** (version 3.0.0) contenu dans l'outil **sra-tools** [NCBI, 2022]. Cette méthode prend seulement en entrée le terme SRR et télécharge les deux reads. On obtient des fichiers `fastq`. Ce format contient une suite de séquences. Pour chaque séquence, il y a 4 informations : nom de la séquence suivi de la séquence elle-même puis une ligne commençant par un '+' et la dernière ligne contenant le score de qualité de la séquence. Pour le génome humain, on a choisi de télécharger chaque chromosome un à un avant de les concaténer. Les fichiers obtenus sont au format `fa` (2 lignes par séquence : nom de la séquence suivi de la séquence elle-même).

Une fois cette première étape effectuée, on télécharge les annotations du génome humain à l'aide de **wget**. Le fichier obtenu est au format `gtf`.

Puis vient l'indexation du génome, on utilise l'outil **STAR** (version 2.7.10, [Dobin et al., 2012]), pour générer un dossier comportant toutes les informations et les index du génome humain.

Après l'étape d'indexation, toutes les données nécessaires au workflow sont téléchargées. On a développé un second sous-workflow comprenant deux process pour exécuter une analyse sur la qualité des données et si besoin nettoyer les reads en enlevant certaines parties. Un premier process porte sur l'analyse propre des reads avec l'outil **fastqc** (version 0.11.9-0, [Bioinformatics, 2019]). En sortie, on obtient un fichier `html` comportant les informations sur

la qualité des reads (scores de qualité des bases, des statistiques générales sur l'échantillon, *etc*). Si besoin, si la qualité des données est mauvaise, on peut utiliser le process trimming qui se charge d'enlever les adaptateurs avec l'outil java `trimmomatic` (version 0.35-6, [Bolger et al., 2014]). On a implémenté cet outil avec trois paramètres *Leading* (si une base au début d'un read à un score de qualité sous la valeur d'un seuil prédéfinie, elle sera supprimée), *Trailing* (même chose mais pour les bases à la fin d'un read) et *Minlen* (supprimer les reads qui sont inférieurs à une longueur spécifiée).

Une fois le téléchargement de toutes les données effectué et le contrôle qualité réalisé si spécifié, on réalise l'alignement des reads sur le génome de référence à l'aide de l'outil **STAR** (version 2.7.10, [Dobin et al., 2012]) qui crée des fichiers bam. Ces fichiers bam contiennent le résultat de l'alignement sous format binaire compressé. Mais ces fichiers obtenus ne sont pas triés, et la recherche d'informations dans ces fichiers est coûteuse en temps. Pour cela, on utilise à la suite de l'alignement l'outil **Samtools** (version 1.16.1, [Li et al., 2009]) avec comme méthode `index` créant un fichier bai pour chaque read. Un fichier bai contient une *table des matières* des éléments contenus dans le fichier bam permettant d'accéder plus rapidement à ces éléments lors de l'interrogation de ces fichiers.

L'étape suivant l'alignement des séquences consiste en la création d'un fichier de statistique, appelé table de comptage, contenant pour chaque gène, le nombre de reads qui lui sont alignés pour chacun des individus. Pour réaliser cette étape, on utilise **featureCounts** (version 2.0.3, [Liao et al., 2013]).

A partir de cette étape, on n'a plus besoin d'effectuer de calculs, de modification sur les fichiers. On utilise le fichier obtenu par **featureCounts** pour effectuer l'analyse différentielle.

### 1.3 Analyse différentielle

L'analyse différentielle a été effectuée en R à l'aide de la librairie **DESeq2** (version 1.38.1, [Love et al., 2014]) et la librairie **ggplot2** (version 3.4.0, [Villanueva and Chen, 2019]) pour certains graphiques. Notre script d'analyse peut être divisé en deux parties.

Premièrement, on réalise une analyse différentielle sur l'ensemble des données (sans aucun filtrage) avec la méthode *DESeq* en ayant au préalable transformé les données de la table de comptage dans le bon format (*DESeqDataSetFromMatrix*). Une fois les résultats obtenus, on réalise plusieurs graphiques pour analyser la qualité des données et les résultats :

- L'histogramme des p-values donne un aperçu de la qualité des données et des résultats de l'analyse effectuée.
- Un graphique **PlotMa** représentant le log fold-change en fonction de la moyenne des comptages normalisés des gènes. Ce graphique met aussi en avant (en couleur) les gènes différentiellement exprimés (DE).
- Après un changement d'échelle des données (en  $\log_2$ ) à l'aide de **rlog**, on a appelé la fonction **plotPCA** pour performer et afficher une analyse en composantes principales (ACP).
- Pour les gènes ayant soit une p-value ajustée la plus faible soit la plus haute, on a représenté leurs comptages normalisés pour chaque échantillon.

Puis, on applique une fonction de nettoyage des gènes en ne gardant que les gènes possédant au moins 5 reads alignés sur l'ensemble des échantillons. Et on génère les mêmes graphiques pour comparer l'avant et l'après filtrage.

On sauvegarde les représentations de comptages pour le gène *SF3B1* (avant et après filtrage) et un fichier de statistique est aussi généré à la fin de l'analyse comportant le nombre de gènes DE, les p-values minimales et maximales avant et après filtrage et les p-values ajustées sur le gène *SF3B1*.

## 1.4 Workflow final

Le workflow et les scripts pour créer les images **Docker**, ainsi que les instructions pour faire tourner l'analyse sont disponibles sur Github (<https://github.com/George-Marchment/hackathon>).

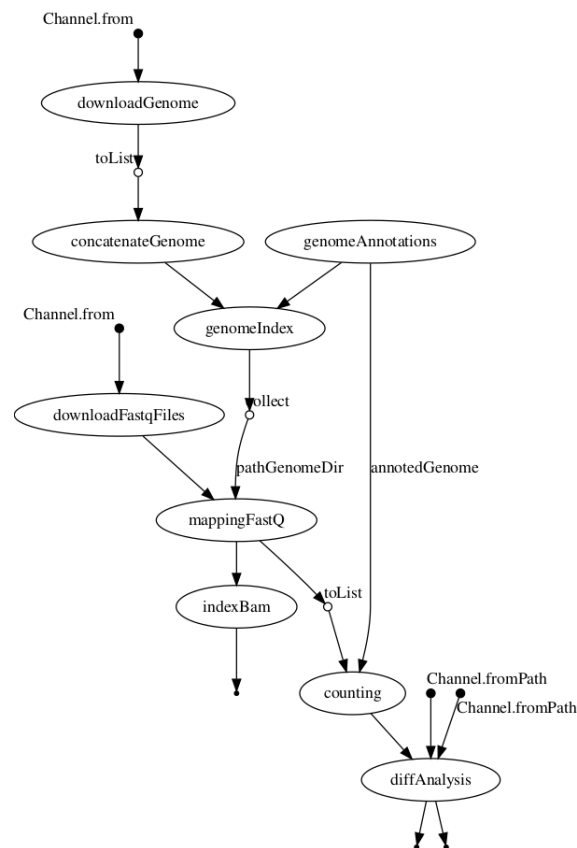


FIGURE 1 – DAG (direct acyclic graph) généré par Nextflow

## 2.1 Résultats globaux

Le but principal de ce projet était d'obtenir le pipeline suivant :

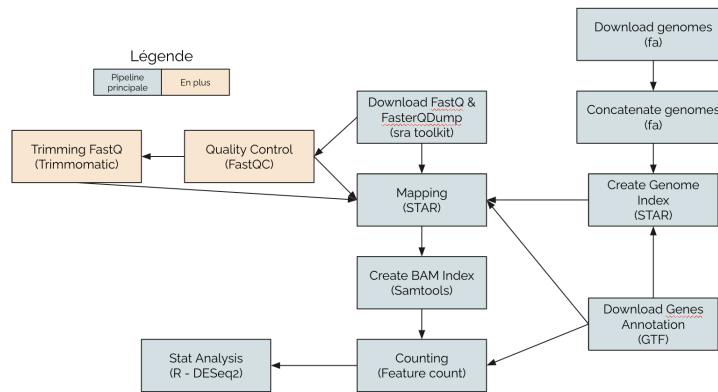


FIGURE 2 – Pipeline final

A partir des séquences (.fasta), du fichier d'annotation (.gtf) du génome et des fichiers contenant les séquences nucléotidiques et les scores de qualité pour les données RNA-seq (.fastQ), ce workflow permet d'obtenir des graphiques et résultats permettant d'effectuer une analyse différentielle sur des données RNA-seq. Deux tables contenant tous les gènes différentiellement exprimés (DE) avant et après filtrages ainsi qu'un fichier de statistiques résumant les tables de gènes DE ont été obtenus.

Les différentes étapes de ce pipeline ont été décrites précédemment, on va donc maintenant se concentrer sur les résultats obtenus après l'exécution, c'est-à-dire les différents graphiques d'analyses obtenus grâce au package R DESeq2 [Love et al., 2014].

## 2.2 Contrôle qualité

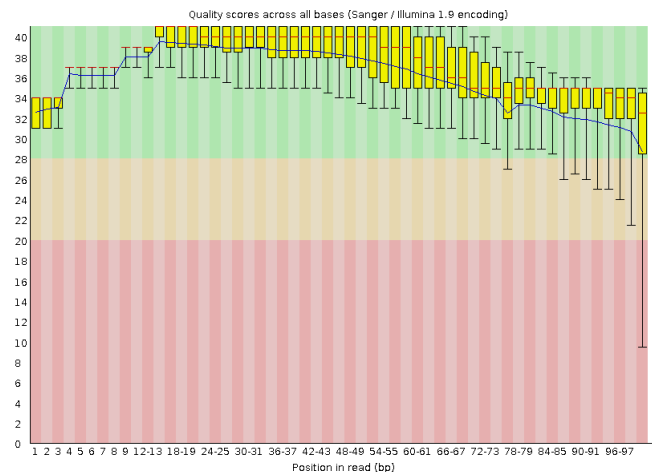


FIGURE 3 – Exemple de boxplot FastQC obtenu pour l'échantillon *SRR628582.1.fastq*

Avant toute analyse, on a vérifié la qualité des reads grâce à **FastQC**. En abscisse, il y a la position des reads en paire de bases et en ordonnée le score Phred. À chaque position, la qualité de tous les reads est représentée sous la forme d'un boxplot. Le code couleur indique les scores de très bonne qualité en vert, bonne qualité en orange et mauvaise en rouge. C'est ce graphique qui va indiquer s'il faut trimmer (avec **Trimmomatic**) ou non les différents reads.

## 2.3 Résultats d'analyse avant filtrage

Dans un premier temps, on a réalisé des graphiques afin de vérifier si les données obtenues étaient en adéquation avec l'analyse différentielle, autrement dit qu'elles n'avaient pas besoin de filtrage par exemple.

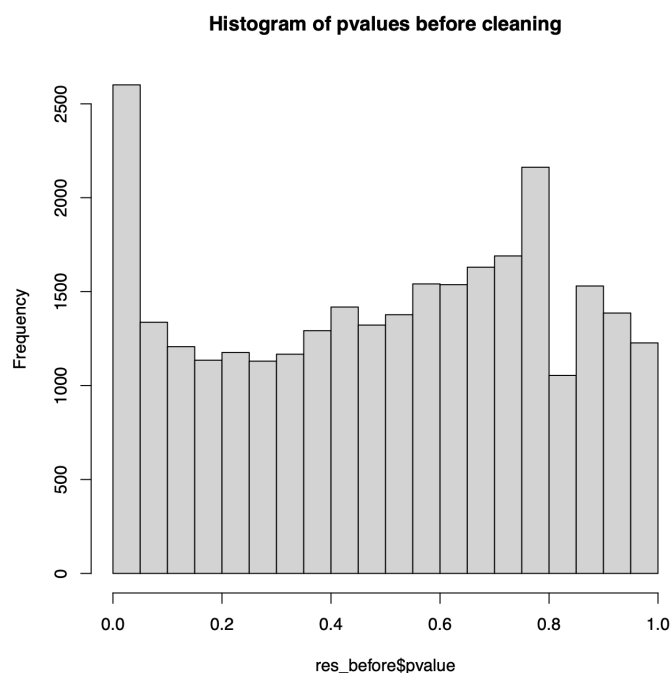


FIGURE 4 – Histogramme des p-values obtenus avec DESeq2 avant filtrage

Un histogramme des p-values brutes a été obtenu. Il permet de vérifier si les hypothèses faites dans la construction des tests sont bien en adéquation avec les données. Pour rappel, les hypothèses sont :

- $H_0$  : Pas de différence d'expression statistiquement significative entre les individus ayant la mutation sur le gène *SF3B1* (condition A) et les individus ayant une mutation (condition B).
- $H_1$  : Existence d'une différence d'expression statistiquement significative entre les deux groupes d'individus.



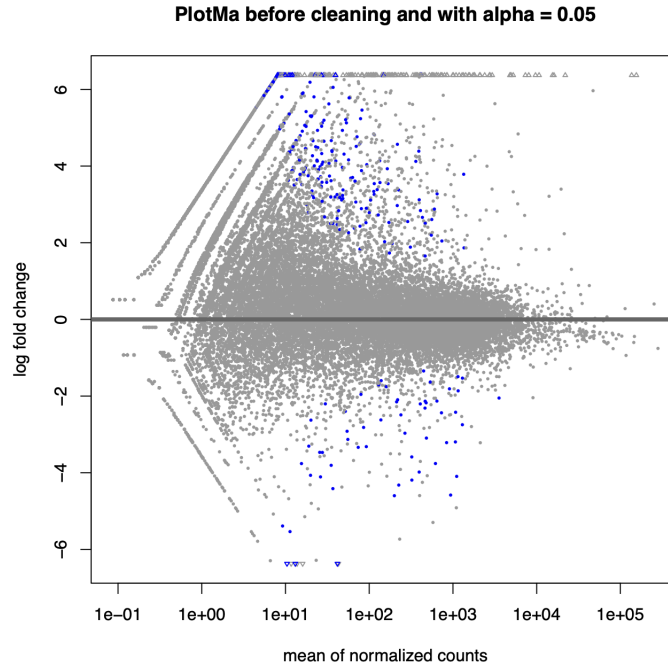


FIGURE 5 – PlotMA avant filtrage et avec  $\alpha = 0.05$

Puis, un graphique représentant le log fold-change (logFC) en fonction de la moyenne des comptages normalisés des gènes a été effectué grâce à la fonction `plotMA`. Le logFC est le logarithme du ratio de la moyenne des comptages normalisés de la condition A sur la moyenne des comptages normalisés de la condition B. Les points bleus représentent les gènes DE et les points gris sont ceux qui ont le même niveau d'expression. On considère les gènes comme étant différentiellement exprimés si la p-value ajustée est inférieure à 0.05.

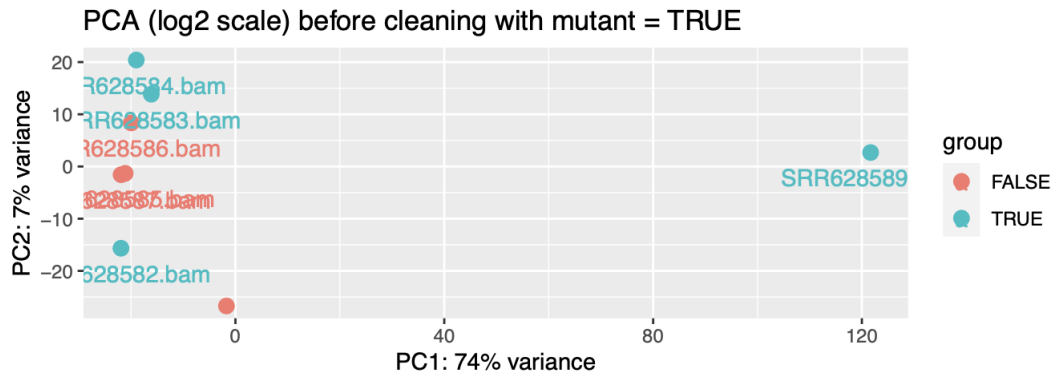


FIGURE 6 – ACP avec Mutant=True et Non-mutant=False avant filtrage

Le dernier graphique que l'on a eu grâce à notre pipeline est une analyse en composantes principales (ACP). Elle permet de visualiser la variabilité des données mais aussi de détecter les conditions qui discriminent le plus les échantillons. Grâce à cette dernière, on peut voir les échantillons corrélés entre eux, par exemple les échantillons mutants (TRUE) et non-mutants (FALSE) regroupés sous deux clusters différents.

Pour finir, le fichier de statistiques donne le nombre de gènes DE avant filtrage, c'est-à-dire 260, ainsi que les identifiants des plus hautes et plus basses p-values ajustées.

## 2.4 Résultats d'analyse après filtrage

Dans un second temps, les mêmes graphiques ont été refaits avec les données filtrées. En effet, deux filtrages ont été effectués sur les données. Le premier consiste en la suppression des gènes non exprimés, c'est-à-dire les gènes pour lesquels pour tous les échantillons leur somme de reads alignés vaut 0. Le second réside en la suppression des gènes peu exprimés. Ce sont ceux ayant moins de 5 reads alignés à travers les deux conditions. De plus, dans le fichier de statistiques, on obtient 243 gènes DE ainsi que les mêmes informations que précédemment mais sur les données filtrées.

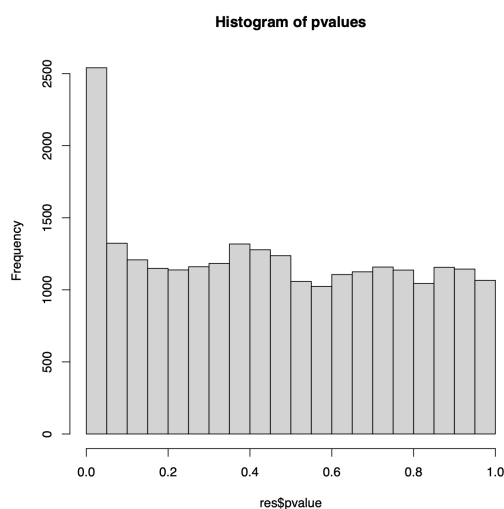


FIGURE 7 – Histogramme des p-values obtenus avec DESeq2 après filtrage

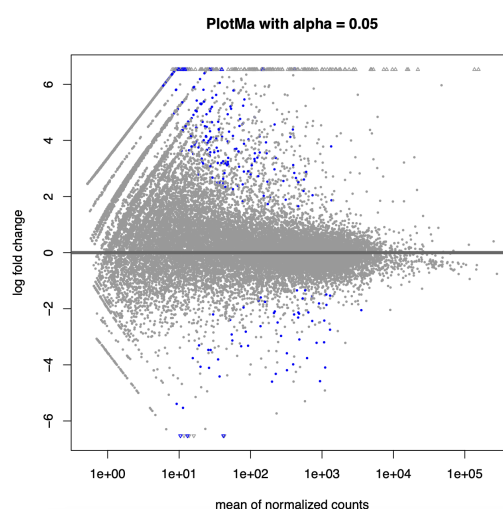


FIGURE 8 – PlotMA après filtrage et avec  $\alpha = 0.05$

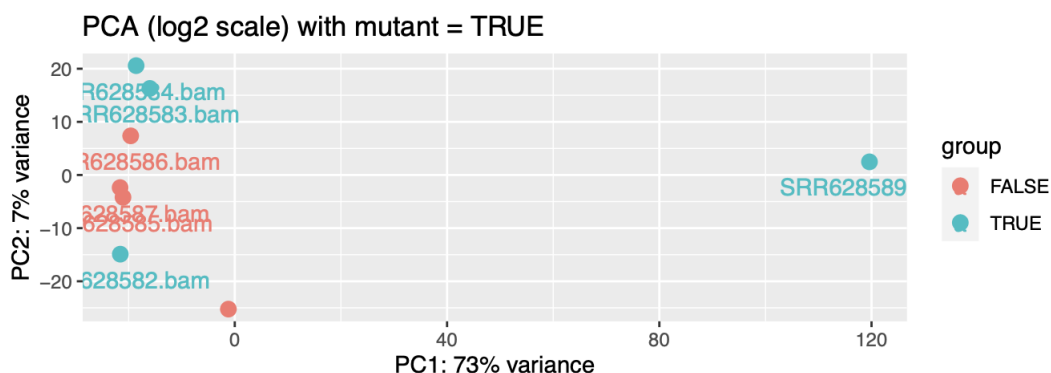


FIGURE 9 – ACP avec Mutant=True et Non-mutant=False après filtrage

Dans un dernier temps, l'analyse différentielle a été effectuée [Love et al., 2014]. Afin de pouvoir voir si la mutation de notre gène d'intérêt *SF3B1* implique une différence d'expression pour les autres gènes, on regarde les gènes DE. Cette analyse permettra de conclure si un des articles est en accord avec les résultats que l'on obtient.

Le projet Hackathon consistait en la reproductibilité d'un workflow d'analyse de données RNA-seq et de l'analyse différentielle, c'est-à-dire à la mise en évidence de gènes différentiellement exprimés dans deux conditions différentes (gène *SF3B1* muté ou non) des deux papiers mentionnés.

### 3.1 Analyse des résultats

Les graphes FastQC (contrôle qualité) permettant de juger de la qualité des reads obtenus sont, pour tous les reads étudiés, dans la partie verte, ce qui signifie que les reads sont *a priori* de bonne qualité et peuvent être utilisés dans le workflow sans étape supplémentaire (ils n'ont pas besoin d'être trimmés) (Figure 3).

Pour réaliser l'analyse différentielle, on a besoin des métadonnées, c'est-à-dire de données permettant de savoir quels individus correspondent à quelle condition (ici quels sont les sujets mutants et ceux qui ne le sont pas). Cette information n'est pas clairement donnée dans les papiers et a été compliquée à obtenir. On a donc utilisée les données obtenues avec **SRA run selector** et le numéro de l'étude [NCBI, 2013b]. Toute l'analyse différentielle réalisée dans la suite repose donc sur des données que l'on trouve difficilement, ce qui appuie le problème de non-reproductibilité de l'étude.

L'analyse différentielle réalisée sans filtrage des gènes permet de mettre en évidence 260 gènes différentiellement exprimés. Cependant, l'histogramme des p-values qui représente la distribution des p-values brutes avant correction et qui permet de s'assurer que les tests se comportent de façon attendue, montre que cette analyse n'est pas fiable, puisque on a en effet un pic dans les p-values proches de 0.7 (Figure 4).

On a donc filtré les gènes en retirant les gènes peu exprimés et on obtient un histogramme des p-values plus proche de l'attendu (Figure 7). Avec ce filtrage, on obtient 243 gènes différentiellement exprimés. On peut observer également que sur ces 243 gènes, les gènes sont plus souvent sur-exprimés ( $\log FC > 0$ ) que sous-exprimés (57 sous-exprimés soit 23%) (Figure 8).

Cependant, le filtrage des gènes ne permet en aucun cas de corriger l'ACP qui soulève un réel problème. En effet, l'ACP (Figure 9) ne permet pas de bien séparer les deux groupes (mutants et non mutants), ce qui signifie que l'hypothèse de sparsité de l'analyse différentielle n'est pas respectée. La réduction des dimensions n'est donc pas possible car, parmi les gènes étudiés, il n'y a pas de petit groupe de gènes suffisamment différents d'un groupe à l'autre pour permettre de les séparer. L'analyse différentielle ne serait donc, *a priori*, pas la bonne méthode à utiliser pour identifier les gènes, si on disposait plus de temps pour traiter le projet on pourrait tester des méthodes de clustering par exemple.

### 3.2 Comparaison avec les papiers

Pour répondre à la problématique du projet portant sur la reproductibilité des résultats, on a tout de même comparé les résultats obtenus avec ceux des papiers. Dans le papier de Harbour et *al.*, ils identifient uniquement 10 gènes comme différentiellement exprimés avec 7

sous-exprimés (soit 70%) (Supplementary Table 2 [Harbour et al., 2013]), alors que dans le papier de Furney et *al.*, ils en obtiennent 325 (Supplementary Table 8 [Furney et al., 2013]) avec 279 sous-exprimés (soit 85%).

Parmi les 243 gènes DE, on n'en a aucun en commun avec Harbour et *al.* et seulement 5 avec Furney et *al.*. De plus, on a noté que parmi les gènes identifiés comme différentiellement exprimés, on est loin du taux de gènes sous-exprimés des deux papiers. L'analyse de ces données mène donc à une troisième conclusion, différente des deux précédentes, et met bien en évidence le problème de reproductibilité de celle-ci.

### 3.3 Conclusion et perspectives

Ainsi, notre analyse nous a permis de mettre en lumière la non-reproductibilité des résultats évoqués dans les deux papiers et de la nécessité de celle-ci. Les deux études évoquent également le rôle de SF3B1 dans l'épissage alternatif, il pourrait alors être intéressant par la suite de réaliser également cette étude afin de prouver encore une fois, ou non, de la non-reproductibilité de ces études.

- [Baker, 2016] Baker, M. (2016). 1,500 scientists lift the lid on reproducibility. *Nature*, 533 :452–454.
- [Bioinformatics, 2019] Bioinformatics, B. (2019). Fastqc. <https://www.bioinformatics.babraham.ac.uk/projects/fastqc/>.
- [Bolger et al., 2014] Bolger, A., Lohse, M., and Usadel, B. (2014). Trimmomatic : A flexible trimmer for illumina sequence data. *Bioinformatics (Oxford, England)*, 30.
- [Di Tommaso et al., 2017] Di Tommaso, P., Chatzou, M., Floden, E. W., Barja, P., Palumbo, E., and Notredame, C. (2017). Nextflow enables reproducible computational workflows. *Nature Biotechnology*, 35 :316–319.
- [Dobin et al., 2012] Dobin, A., Davis, C., Schlesinger, F., Drenkow, J., Zaleski, C., Jha, S., Batut, P., Chaisson, M., and Gingeras, T. (2012). Star : ultrafast universal rna-seq aligner. *Bioinformatics (Oxford, England)*, 29.
- [Furney et al., 2013] Furney, S., Pedersen, M., Gentien, D., Dumont, A., Rapinat, A., Desjardins, L., Turajlic, S., Piperno-Neumann, S., de la Grange, P., Roman-Roman, S., Stern, M.-H., and Marais, R. (2013). Sf3b1 mutations are associated with alternative splicing in uveal melanoma. *Cancer discovery*, 3.
- [Harbour et al., 2013] Harbour, J. W., Roberson, E., Anbunathan, H., Onken, M., and Worley, L. (2013). Recurrent mutations at codon 625 of the splicing factor sf3b1 in uveal melanoma. *Nature genetics*, 45.
- [Li et al., 2009] Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., and Abecasis, G. (2009). Genome project data processing s : The sequence alignment/-map format and samtools. *Bioinformatics (Oxford, England)*, 25 :2078–9.
- [Liao et al., 2013] Liao, Y., Smyth, G., and Shi, W. (2013). Featurecounts : An efficient general purpose program for assigning sequence reads to genomic features. *Bioinformatics (Oxford, England)*, 30.
- [Love et al., 2014] Love, M., Huber, W., and Anders, S. (2014). Moderated estimation of fold change and dispersion for rna-seq data with deseq2. *Genome biology*, 15 :550.
- [Merkel, 2014] Merkel, D. (2014). Docker : lightweight linux containers for consistent development and deployment. *Linux Journal*, 2014.
- [NCBI, 2022] NCBI (2022). Sra tools. <https://github.com/ncbi/sra-tools>.
- [NCBI, 2013a] NCBI, S. D. (2013a). Sra dataset. [https://www.ncbi.nlm.nih.gov/Traces/study/?acc=SRP017413&o=acc\\_s%3Aa](https://www.ncbi.nlm.nih.gov/Traces/study/?acc=SRP017413&o=acc_s%3Aa).
- [NCBI, 2013b] NCBI, S. R. S. (2013b). Sra run selector. [https://www.ncbi.nlm.nih.gov/Traces/study/?acc=SRP017413&o=acc\\_s%3Aa](https://www.ncbi.nlm.nih.gov/Traces/study/?acc=SRP017413&o=acc_s%3Aa).
- [Villanueva and Chen, 2019] Villanueva, R. A. and Chen, Z. (2019). ggplot2 : Elegant graphics for data analysis (2nd ed.). *Measurement : Interdisciplinary Research and Perspectives*, 17 :160–167.