UE Hackaton - M2 AMI2B 2022/2023

# Rapport du projet Hackaton
# Reproductibilité

**Étudiants:**
Siwar Hammami, Yanis Madi, Pauline Lim, Clara Toussaint

12 décembre 2022

# Table des matières

# 1 Introduction

## 1.1 Crise de la reproductibilité

La science est actuellement confrontée à une “crise de reproductibilité" des démarches scientifiques, notamment en bio-informatique. La reproductibilité désigne la capacité d'une équipe à reproduire les résultats d'une étude réalisée par une autre équipe tout en se basant sur les mêmes outils et les mêmes données. Loin d'être une pratique simple, la reproduction nécessite des indications précises sur le déroulement de l'expérience, le matériel requis ainsi que sur les méthodes d'analyses et de collecte des données.

Ce manque de reproductibilité remet en cause la fiabilité des résultats de nombreuses publications. D'après une enquête publiée par Nature en 2016, plus que 70% de chercheurs ont échoué à reproduire des expériences préexistantes et plus que 40% n'ont pas réussi à reproduire leurs propres expériences. Une des raisons expliquant cette crise est la culture du *publish* qui juge les chercheurs sur la quantité et l'impact de leurs publications scientifiques, les forçant ainsi à publier pour propulser et maintenir leur carrière.

On distingue trois types de reproductibilité : la reproductibilité statistique qui dépend du choix des tests statistiques à effectuer, des paramètres qui modélisent le phénomène étudié et des valeurs seuils à définir. Ensuite, la reproductibilité empirique qui étudie le caractère variable et aléatoire des phénomènes biologiques et des techniques expérimentales. Enfin, la reproductibilité computationnelle qui concerne les détails des logiciels, les versions et les dépendances entre les outils informatiques utilisés.

## 1.2 Présentation des articles

Le projet se base principalement sur deux publications :

Article 1 : SF3B1 mutations are associated with alternative splicing in uveal melanoma

Article 2 : Recurrent mutations at codon 625 of the splicing factor in uveal melanoma "

Les deux publications scientifiques reposent sur l'étude du mélanome de l'uvée. Il s'agit du cancer de l'œil le plus fréquent chez l'adulte, touchant en moyenne près de 600 personnes en France chaque année. Cette forme de cancer est associée à une mutation du gène SF3B1, qui est impliqué dans l'épissage des ARN pré-messagers. Les expériences menées par les chercheurs du premier article ont été reprises par le second groupe de chercheurs afin de déterminer si ces mutations ne pourraient pas modifier la manière dont ces gènes s'expriment.

## 1.3 Objectifs du projet

Lors de ce projet, nous utilisons le même jeu de données dans le but de comprendre l'origine des différences entre ces deux publications. Il sera question alors d'effectuer une analyse différentielle afin de déterminer les gènes qui ont une moyenne d'expression différente selon leurs appartenance à un sujet SFB3-mutant ou à un wild-type. Pour cela, nous avons mis en place un pipeline d'analyse de données RNAseq et des images Docker afin d'assurer la reproductibilité de nos propres résultats. Le jeu de données est composé de 8 échantillons répartis de la façon suivante : 5 individus wild-type et 3 individus mutants.

## 1.4 Répartition des tâches

Pour le développement du workflow, Clara, Yanis et Siwar ont réalisé l'ensemble des images Docker. Pauline s'est concentrée sur la structuration du workflow. Par la suite, nous nous sommes tous répartis les process et nous sommes entraidés dès la rencontre d'un obstacle. Pour l'analyse

statistique, Siwar a réalisé le code sur Rstudio dans un premier temps. Pauline s'est chargée de la reproductibilité du code et de l'exportation des résultats. Yanis a permis l'intégration du code dans le pipeline. Siwar et Clara ont réalisé l'analyse biologique et statistique. Pour ce projet, nous avons communiqué grâce à Messenger et Discord lorsque nous ne pouvions pas nous retrouver. Provenant tous les 4 d'une licence de biologie, nous avons su exploiter l'expérience que chacun a acquis durant la période de stage de master 1 et collaborer efficacement.

# 2 Matériel et méthodes

## 2.1 Choix des outils

Pour optimiser la reproductibilité de nos résultats, nous avons choisi de travailler avec l'outil Netflow pour le management de notre workflow et Docker afin de créer des containers utiles pour nos process Nextflow. L'outil Github s'est avéré très utile pour le partage et le versionnage de nos scripts.

Pour pouvoir exécuter notre pipeline, nous avons utilisé des machines virtuelles fournies par l'IFB Cloud. Nous nous sommes connectées en utilisant nos comptes de l'université Paris-Saclay et nous avons récupéré et ajouté la public key à nos comptes IFB Cloud, pour avoir accès au SSH (Secure Shell, pour la connexion à distance). Ensuite, nous avons lancé la VM nommée "BioPipes" à partir de l'onglet RainBio, en choisissant les paramètres par défaut suivants : le groupe *ReproHack2022*, le cloud *ifb-core-cloud* et le gabarit d'image cloud *ifb.m4.large*. Cette étape nous a permis d'obtenir l'adresse IP de la machine " : xxx.xxx.xxx.xxx". La connexion à cette machine virtuelle se fait en local, dans un terminal en copiant son adresse IP avec la commande "ssh -Y ubuntu@xxx.xxx.xxx.xxx". Nous avons tout d'abord choisi par défaut une machine virtuelle "ifb.m4.large" avec une configuration comme suit : 2 vCPU, 8Go GB RAM, 120 Go GB local disk. Cependant, nous nous sommes vite rendu compte que ces paramètres n'étaient pas suffisants pour exécuter notre workflow car nos process s'arrêtaient à cause du manque de ressources.

## 2.2 Description du workflow

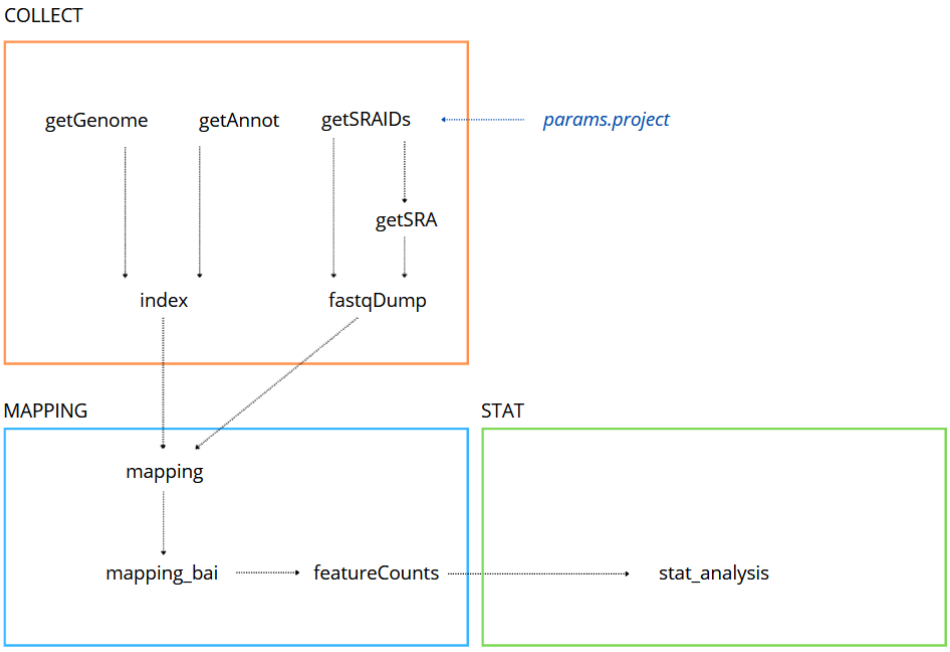


FIGURE 1 – Schéma du workflow Hackaton

Le workflow s'organise en 3 sous-workflow qui séparent la récupération de données, le mapping et l'analyse statistique des données. Ces étapes sont interdépendantes : l'étape d'analyse ne peut pas fonctionner dans que l'étape de mapping n'a pas tourné qui elle-même doit recevoir les output de l'étape de récupération de données. De plus, cette compartimentation permet la réutilisation des sous-workflows dans d'autres projets. Dans un souci d'organisation, les sorties de chaque process sont globalement compartimentés dans des répertoires différents.

**Récupération des données :**

Dans un premier temps, on récupère les identifiants des échantillons qui sont de type "Transcriptomic sequencing" seulement grâce aux fonctions *esearch* et *efetch*. Ces identifiants sont utilisés pour télécharger les fichiers sra dans le process getSRA qui seront par la suite processés par fastqdump dans le process fastsqDump.

Pour l'indexation du génome, le génome de référence (ref.fa) et le fichier d'annotation (gtf) sont téléchargés respectivement par les process *getGenome* et *getAnnot*. Le process *index* réalise l'indexation en utilisant le fichier du génome de référence et le fichier d'annotation avec STAR. Les fichiers d'indexation sont stockés dans le répertoire *index/*. Le mapping nécessite les données sous le format fastq, le fichier d'annotation de génome et le fichier d'indexation.

**Mapping :**

STAR est de nouveau utilisé mais pour le mapping. Les paramètres renseignés sont ceux qui ont été fournis sur la fiche du projet. Nous avons visualisé l'ensemble des résultats de mapping sur IGV en dehors du pipeline à partir des fichiers bai générés par le *process mapping_bai*. Enfin, on obtient la table de comptage qui constitue l'objet de l'analyse statistique en utilisant featureCounts.

**Analyse statistique :**

L'analyse statistique est constituée d'un seul process qui appelle un script R stocké dans le répertoire *bin/*. L'analyse a été menée par l'utilisation du package DESeq2. Dans cette étape ont été réalisés une ACP des différents échantillons et diverses analyses pour l'identification de gènes différentiellement exprimés. En parallèle, nous avons réalisé des analyses supplémentaires qui ont été résumées dans un fichier RMarkdown.

## 2.3 Méthodes d'analyse statistique

Cette partie consiste à identifier les gènes ayant une moyenne d'expression significativement différente entre deux différentes conditions ( Wild-Type et mutant-SFB3) parmi les 60612 gènes séquencés. Le script R stat_analysis.r, qui prend en entrée la matrice de comptage, permet l'identification de gènes différentiellement exprimés en calculant le log Fold Change et en appliquant des fonctions issues du package DESeq2. D'ailleurs, nous avons utilisé le container evolbioinfo/deseq2 :v1.28.1 pour utiliser ce dernier.

### 2.3.1 Pre-processing des données

Les données de comptage sont généréalement modélisées par une loi de poisson qui implique une propriété unique de cette distribution où la moyenne est égale à la variance. Cependant, cette loi n'est pas adéquate dans la mesure où la variance des données est très importante. Cette variance est dûe a fait que les données sont des résultats de séquençage issus de patients différents. La distribution binomiale négative s'avère alors plus adaptée pour modéliser ce type de données. Pour cela, nous avons tracé un ggplot qui modélise les moyennes et les variances de nombre de reads avec un ajustement par la distribution binomiale négative.

Analyse en composantes principales

L'ACP est une méthode d'analyse des données qui se base sur la statistique multivariée. Elle consiste à réduire le nombre de variables corrélées en nouvelles variables décorrélées. Ces nouvelles variables sont nommées composantes principales. Cette fonction génère un graphe de représentation des individus qui permet de voir à quel point la variation des gènes est similaire entre les échantillons. Cette étape d'analyse vise à évaluer la similitude globale entre les

échantillons. La visualisation des données a été réalisée grâce à la fonction PCA. Cette approche permet de savoir s'il est possible de distinguer le groupe des mutants du groupe des wild-type ou non.

Clustering

Afin de visualiser la corrélation entre les 8 échantillons nous avons utilisé la méthode de classification hiérarchique. Cette méthode indique la similarité entre les échantillons en calculant la corrélation de l'expression des gènes normalisés pour toutes les combinaisons de paires d'échantillons. Pour cela, la fonction "cor" calcule la corrélation de pearson par paire d'échantillon. Les résultats sont représentés sous forme de Heatmap en utilisant la fonction "heatmap".

### 2.3.2 Analyse différentielle

L'analyse statistique est constituée d'un fichier R où les package DESeq2 et FactoMineR ont essentiellement été utilisés. Dans cette étape ont été réalisés une ACP des différents échantillons et diverses analyses pour l'identification de gènes différentiellement exprimés. En parallèle, nous avons réalisé des analyses supplémentaires dont le code est disponible dans un RMarkdown et dont les figures ont été stockées dans le fichier furtheranalysis_figures.pdf.

# 3 Résultats

## 3.1 Analyse des p-values

Tout d'abord, on trace l'histogramme des p-values brutes des gènes afin de contrôler la qualité des données.

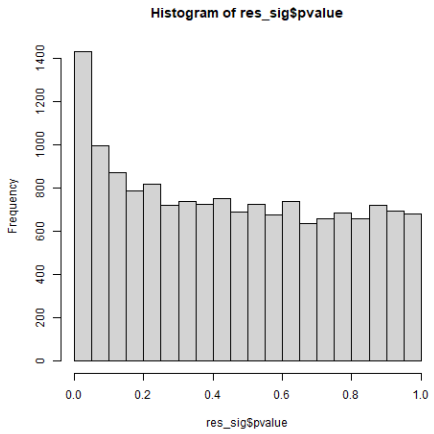


FIGURE 2 – Histogramme des p-values brutes des gènes

On observe un pic plus important en 0 mais une répartition forte et quasi uniforme des p-values entre 0 et 1. On observe aucune tendance entre 0 et 1. En effet, nous devons normalement observer une répartition uniforme des p-values entre 0 et 1 correspondant au gènes non différentiellement exprimés et un pic de valeurs en 0 qui correspond aux gènes différentiellement exprimés. Par conséquent, notre histogramme des p-values brutes des gènes est cohérent. On peut donc continuer l'analyse.

## 3.2 Clustering des échantillons

Pour explorer les similarités entre échantillons, nous calculons une matrice de distance entre échantillons et on la représente à l'aide d'une heatmap dont les lignes et colonnes ont été réorganisés à l'aide d'un clustering hiérarchique.

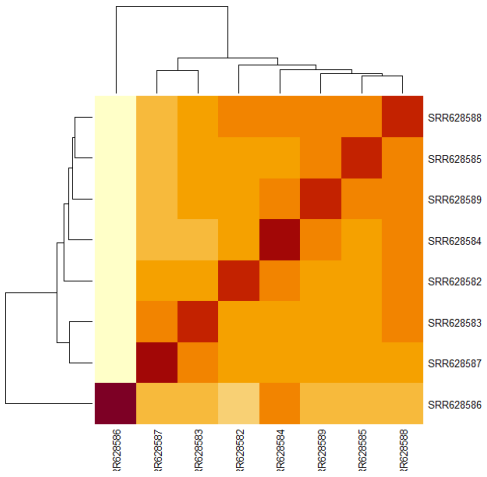


FIGURE 3 – Heatmap de la matrice de distance entre les 8 échantillons.

On remarque que l'échantillon "SRR628586" est le seul à être très différent comparé aux autres échantillons. Des échantillons mutés tels que "SRR62859" et 'SRR628588" se ressemblent entre

eux, ce qui est cohérent. Cependant, on retrouve des échantillons de type “Wild Type” qui ressemblent à des échantillons mutés. C’est notamment le cas pour l’échantillon “SRR628582” et l’échantillon muté “SRR628588”. Nous pouvons alors émettre une critique vis-à-vis des données. Nous allons poursuivre notre analyse en effectuant une Analyse en composantes principales pour confirmer nos résultats.

## 3.3 Analyse en composantes Principales

L’ACP constitue une seconde approche qualitative qui permettra potentiellement la distinction entre le groupe des mutants des non mutants. Nous effectuons une Analyse en Composantes Principales de nos 8 échantillons.

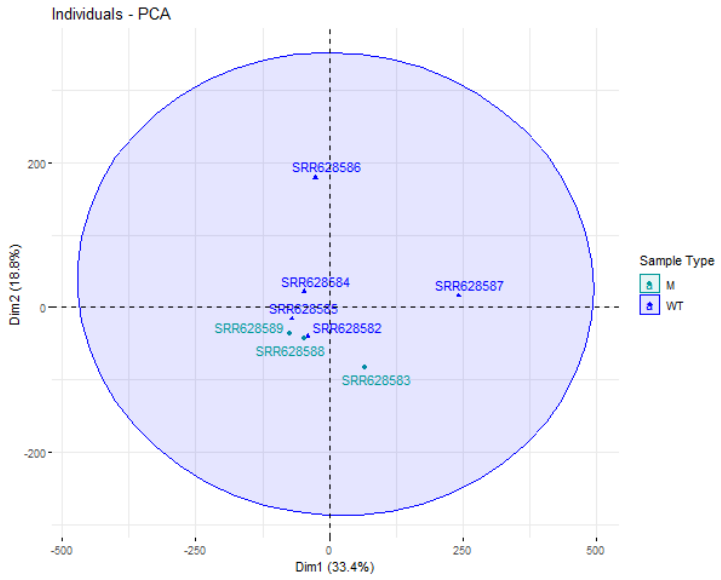


FIGURE 4 – Analyse en Composantes Principales des 8 échantillons

On remarque d’abord que 52.2% de la variation totale est expliquée, ce qui est correct. L’analyse peut donc être poursuivie. On observe que l’ensemble des individus sont regroupés au centre de notre ACP, excepté pour un individu : “SRR628586” qui est un échantillon muté. Comme précédemment, on remarque que des échantillons mutés sont proches des échantillons “Wild Type” tel que le couple “SRR628582” et “SRR628588”. Cette ACP confirme nos observations précédentes.

Il n’est pas possible de distinguer deux groupes distincts entre nos échantillons mutés et non mutés. Cela ne compromet pas la possibilité d’une expression différentielle entre ces deux groupes. Cette figure ne nous permet pas d’identifier une expression différentielle entre nos groupes. Toutefois, il aurait été intéressant de retirer les gènes de poids trop faible dans la définition des axes de projection et de réitérer l’ACP pour mieux distinguer les cas mutés des sauvages. Il serait possible de mettre en évidence les gènes avec une forte expression différentielle grâce à un Volcano Plot par exemple.

## 3.4 Expression différentielle des gènes

### 3.4.1 Ratio des gènes différentiellement exprimés

On construit un tableau où on calcul le ratio de gène différentiellement exprimé pour les gènes mutants surexprimés ainsi que les gènes “Wild Type” surexprimés.

| | Amount | Ratio (significantly differentially expressed) |
|---|---|---|
| overexpressed genes in Wild Types | 48 | 0.84 |
| overexpressed genes in Mutants | 9 | 0.16 |

FIGURE 5 – Tableau représentant le ratio des gènes différentiellement exprimés suivant s’ils sont mutants surexprimé et “Wild Type” surexprimé.

Dans notre analyse, nous n’avons gardé que 57 gènes significativement différentiellemment exprimés parmi le pool de gènes initial. La mesure utilisée est le Log Fold Change en prenant le type "Wild-Type" comme référence. Ainsi, un Log Fold Change inférieur à 0 indique une surexpression chez les non mutants et inversement, un Log Fold Change supérieur à 0 indique une surexpression chez les mutants. On remarque que pour nos 57 gènes différentiellement exprimés, 84% d’entre eux sont surexprimés chez les “Wild Type” tandis que 16% le sont chez les mutants. On en déduit que parmi nos 57 gènes les plus différentiellement exprimés, la majorité sont surexprimés chez les “Wild Type”. Ainsi, il y aurait globalement une perte d’expression de 84% des 57 gènes dans la condition dite "mutant" et un gain d’expression pour 16% d’entre eux dans la condition dite "wild-type".

### 3.4.2 Volcano Plot

Pour effectuer l’analyse différentielle on utilise le package DESeq2. Il va prendre en entrée la table de comptage pour chaque gène (en ligne) et les individus (en colonne). Une table des méta-données qui associe chaque individu au groupe expérimental auquel il appartient est aussi utilisée.

Un $Log_2\,Fold\,Change$ supérieur à 0 signifie une surexpression dans le cas muté par rapport au cas “Wild Type”. Un $Log_2\,Fold\,Change$ inférieur à 0 signifie une sous expression dans le cas muté par rapport au cas “Wild Type” et un $Log_2\,Fold\,Change = 0$ indique qu’on a pas de différence. Cela signifie que les cas de surexpression se situent dans les valeurs positives en ordonnée, tandis que les cas de sous expression se trouvent dans les valeurs négatives.

En abscisse on peut observer le $-Log_{10}P$ qui permet d’avoir une simplification de l’interprétation de l’axe des abscisses. Plus la p-value est faible (plus on a une forte significativité de l’expression différentielle) et plus la valeur de $-Log_{10}P$ est élevée tout en permettant d’afficher beaucoup de valeurs. Enfin, la p-value ajustée est obtenue par correction de Benjamini-Hochberg qui est la méthode par défaut de DESeq2.

Pour pouvoir sélectionner la zone des gènes à expression différentielle élevée et où la densité des points qui les représentent est suffisamment faible pour bien observer des points individuels, nous avons décidé de fixer un seuil de p-value adjusted à 0.05 et de $Log_2\,Fold\,Change$ à 0.5.

On remarque une certaine asymétrie du Volcano Plot. En effet, on observe plus de gènes sous exprimés dans les cas mutés que de gènes sur exprimés. Dans les 2 articles scientifiques, les chercheurs ont pu observer plus de gènes sous-exprimés que de gènes sur exprimés dans les cas mutés ; ce qui est en accord avec nos résultats.

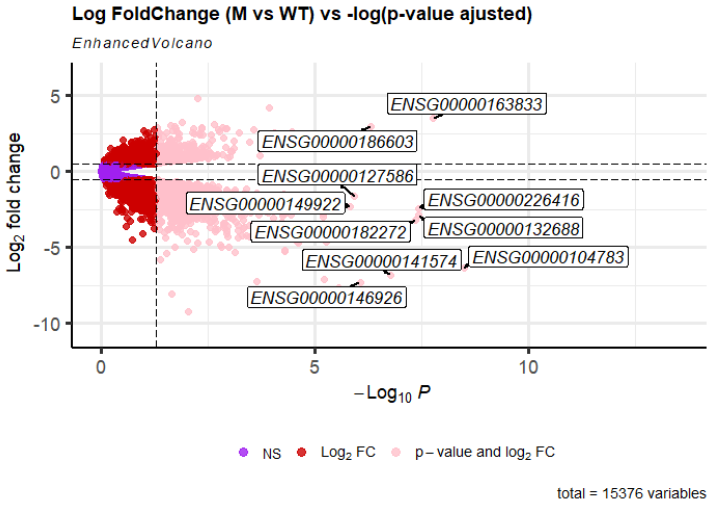


FIGURE 6 – Volcano plot sur 60612 transcrits.
Chaque point est associé à un gène dont le comptage à l'issue du workflow est non nul. Le $Log_2\,Fold\,Change$ est calculé en prenant en référence les cas "Wild Type". La zone d'intérêt est la zone en rose. Elle correspond aux gènes dont l'expression différentielle est plus forte et possède une forte significativité dans le cas muté par rapport au cas sauvage.

## 3.5 Comparaison des résultats d'expression différentielles avec ceux des articles

On construit un tableau contenant les 10 gènes ayant les expressions différentielles les plus importantes dans l'article comparé au 10 gènes possédant l'expression différentielle la plus importante d'après notre analyse.

| Numéro accession article 1 | Numéro accession HackCYPS |
|---|---|
| ENSG00000132824 | ENSG00000104783 |
| ENSG00000104872 | ENSG00000163833 |
| ENSG00000131503 | ENSG00000132688 |
| ENSG00000166136 | ENSG00000182272 |
| ENSG00000134759 | ENSG00000226416 |
| ENSG00000151882 | ENSG00000141574 |
| ENSG00000286588 | ENSG00000186603 |
| ENSG00000146085 | ENSG00000146926 |
| ENSG00000178607 | ENSG00000127586 |
| ENSG00000116641 | ENSG00000149922 |

FIGURE 7 – Tableau des 10 gènes avec l'expression différentielle la plus importante dans l'article 1 comparé à leur rang selon notre analyse

On remarque que les gènes possédant les expressions différentielles les plus importantes dans l'article 1 ne sont pas ceux retrouvés dans notre analyse. Ces différences peuvent être expliquées par le fait que nous n'ayons pas les mêmes versions des outils utilisés ou encore par le fait que nous n'ayons pas assez d'informations sur leurs utilisations et sur leurs méthodes d'analyses statistiques dans cette publication.

# 4 Conclusion

## 4.1 Problèmes rencontrés

Lors de ce projet, nous avons rencontré des difficultés liés au volume des fichiers. En effet, le volume des fichiers important a entraîné la saturation de stockage des machines virtuelles. Nous avons alors changé plusieurs fois la configuration de notre machine virtuelle, tout en veillant à finir le déploiement des VM précédentes. Les VM testées avaient des capacités de mémoires plus ou moins importantes mais aussi un nombre de CPUs (ou Central Processing Unit, le nombre d'unités de calculs) alloués fixé. Les process d'indexation et de mapping sont relativement très longs, et excessivement coûteux en mémoire. Une autre difficiculté a été le temps d'exécution du pipeline. Dans un premier temps, nous avons fait tourner le pipeline avec seulement 2 échantillons pour des questions de temps d'attente et d'utilisation des ressources IFB. Dans un second temps, nous avons tenté d'optimiser le temps d'exécution du pipeline en allouant la totalité des CPUs à ces process. Afin d'éviter le crash du pipeline, nous avons renoncé à la limitation de stockage alloué à chaque process.

## 4.2 Conclusion Générale

Au cours de ce projet, nous avons essayé de reproduire les résultats et les analyses des deux articles scientifiques de Harbour et al. et Furney et al dont le but était de montrer l'influence des mutations sur le gène SF3B1 tout en nous assurant que nos résultats soient reproductibles.

Tout d'abord, il faut noter que les deux publications arrivent à des résultats différents. Suite à notre analyse, nous ne trouvons pas tout à fait les mêmes résultats que les articles scientifiques. Certains de nos résultats concordent avec au moins l'un des articles mais d'autres diffèrent totalement. Il faut prendre en compte le fait que les études ont été réalisées en 2013 et que nous les avons reproduites en 2021, soit huit ans après. En effet, notre projet se présente dans un contexte différent des études entreprises dans ces deux articles. D'autant plus, que notre workflow utilise des versions récentes et des packages différents. Les différences notables des matériels et méthodes utilisées ont été regroupées dans le tableau suivant :

| Matériels et Méthodes | Article 1 (Harbour et al.) | Article 2 ( Furney et al.) | Notre Workflow |
|---|---|---|---|
| Mapping | TopHat(avec un RefSeq GTF) | TopHat + trimmer à 99bp | STAR v2.7.10a |
| Samtools | samtools v0.1.18 | Pas précisé | samtools v1.9 |
| Deseq | DESeq v1.8.3 | DEXseq | DESeq2 (R.4.1.1) |
| Tests utilisés | Test de Fisher, Test de Student binomial | Test de $\chi 2$ et Fisher | Test de Wald |
| Seuils utilisés | FDR≤0.05 | FDR≤ 0.1 & P-value≤0.05 | P-value ajustée≤ 0.05 & LFC |

Figure 8 – Tableau des différences notables des matériels et méthodes utilisées

En effet, il existe des différences au niveau des outils et leur version utilisés. Dans notre pipeline, nous avons choisi d'utiliser la fonction STAR pour l'alignement de nos reads RNAseq sur le génome de référence GRCh38, contrairement aux études qui ont utilisé TopHat et le génome antérieur GRCh37. Pour l'analyse statistique, nous avons appliqué le test de Wald pour déterminer si un gène est significativement différentiellement exprimé ou non, alors que les auteurs des deux articles ont choisi d'utiliser des tests de Fisher ainsi que le Student binomial et le test

du Chi2. Par conséquent, nous pouvons expliquer ce défaut de reproductibilité computationnelle et statistique par cette faiblesse dans le design de l'expérience et des informations concernant les matériels et méthodes utilisés.

Nous devons donc rester prudents vis-à-vis des résultats et des conclusions tirés sur le gène SF3B1 dans les deux articles car elles ne semblent pas assez fiables et robustes. Nous devons aussi garder un esprit critique dû au manque de reproductibilité face aux méthodes utilisées et aux modèles choisis dans les différents programmes impliqués dans les analyses.

Il n'existe pas vraiment de conclusion claire et définitive sur l'influence des mutations du gène SF3B1 dans le cancer uvéal à cause de ce problème de reproductibilité. Pour améliorer l'analyse et faire en sorte d'avoir une conclusion plus robuste et convaincante, on pourrait dans un premier temps revoir le jeu de données de telle sorte à diminuer la variabilité entre les échantillons. Il faudrait ainsi recommencer les analyses sur ce grand jeu de données. On pourrait également revoir les outils utilisés et lancer plusieurs études dans différents laboratoires afin de ne pas avoir de biais dans nos résultats et d'étudier plus finement ce cancer.

Des améliorations peuvent être apportées, notamment au niveau du temps de calcul. Par exempke, nous avons utilisé pour le mapping l'outil featureCounts (provenant du docker subread). Cependant, avec la révolution des outils bioinformatiques des dernières années, il existe aujourd'hui des algorithmes plus performants de normalisation des reads RNAseq. Ces algorithmes permettent notamment l'alignement des reads dit ambigus (par exemple les reads qui s'alignent en partie sur une jonction entre un exon et un intron).

Enfin, nous pensons qu'il serait intéressant de réaliser la même étude à partir des données de Whole Exome Sequencing (WES) pour la recherche de variants afin d'étudier les régions exoniques (codantes).