

RAPPORT

SCMULTIOME

- INSULATOR-MEDIATED 3D CHROMATIN LOOPING -
NEW CO-FACTORS, NEW ROLES, NEW METHODS

MAËVA ABADIE--LIMA & EMMA RODRIGUEZ

CHROMATIN DYNAMICS & CELL PROLIFERATION, CNRS
OLIVIER CUVIER



TABLES DES MATIÈRES

TABLES DES MATIÈRES	0
CONTEXTE ET OBJECTIF	1
Contexte :	1
Objectifs :	2
Équipe :	2
MATERIEL et METHODE	3
Matériel	3
Logiciel utilisé	3
Méthode	3
Contrôle Qualité	3
Analyses préliminaires et Hétérogénéité Cellulaire	4
Intégration des données Pseudo-bulk et single-cell	5
RÉSULTATS	6
Bilan du contrôle qualité	6
Bilan des analyses préliminaires	11
Bilan sur l'Étude de l'Hétérogénéité Cellulaire	13
Bilan de l'intégration des données Pseudobulk et single-cell	14
Identification des gènes différentiellement exprimés	14
Évaluation de la variabilité du signal ATAC	15
Évaluation de la corrélation de la relation entre le signal ATAC et l'expression génique	15
CONCLUSION GÉNÉRALE	16
Conclusion	16
Limites	16
OUVERTURE	17
GESTION DE PROJET	
Calendrier	
Répartition des tâches	
RÉFÉRENCES	
ANNEXE	

CONTEXTE ET OBJECTIF

Contexte :

L'ADN s'enroule autour d'un octamère d'histones pour former l'unité structurale de base de la chromatine appelée le nucléosome. Les nucléosomes protègent l'information génétique des mutations, et jouent des rôles fondamentaux dans la régulation de processus incluant l'expression des gènes, la différentiation et le développement, ou la réparation de l'ADN. Ces régulations ont notamment lieu selon que la chromatine soit accessible ou non. En effet, la structure de la chromatine peut adopter deux grandes formes différentes appelées l'euchromatine et l'hétérochromatine [1]. L'euchromatine est moins condensée, accessible et ainsi plus perméable à l'activation de processus tels que la transcription et la régulation des gènes, tandis que l'hétérochromatine est plus condensée, moins accessible, et ainsi généralement associée à la répression génique [2].

Les modifications épigénétiques sont des mécanismes de régulation transcriptionnelle qui peuvent avoir lieu soit au niveau de l'ADN (e.g. méthylation), soit au niveau des histones qui constituent les nucléosomes. Ces changements ne modifient toutefois pas la séquence d'ADN en tant que telle, mais ils sont transmissibles d'un individu à sa descendance (au cours des générations, et notamment de la méiose), ou par extension d'une cellule mère à ses cellules filles.

Ces modifications épigénétiques ont souvent une fonction particulière telle que la possibilité de favoriser ou non l'expression de gènes où ces modifications ont eu lieu. Lorsque ces modifications se trouvent sur les promoteurs ou les corps de gènes. Ces effets peuvent aussi influer sur l'expression d'un gène situé à distance lorsque ces modifications ont lieu sur des éléments régulateurs tels que des enhancers ou des insulateurs.

Notre analyse a été menée notamment sur les domaines d'association topologique (TAD) que nous savons être présents chez une multitude d'espèces, notamment chez *Drosophila Melanogaster*.

Un TAD est un domaine génomique constitué de sites qui auto-interagissent entre eux, au sein d'une unité structurale continue, ce qui signifie que les régions à l'intérieur d'un TAD interagissent physiquement les unes avec les autres avec une plus grande fréquence qu'avec les régions voisines[3].

Les TAD seraient délimités par des barrières, définies par des séquences génomiques qui correspondent à des sites de fixation de protéines insulatrices. Ces protéines insulatrices réduisent les fréquences d'interaction entre deux éléments situés de part et d'autre de la protéine insulatrice. Ainsi, les éléments insulateurs créent des domaines topologiques où les fréquences d'interaction sont favorisées. Cette organisation en TADs ou en domaines génomiques permet de réguler l'expression de gènes dans le temps et dans l'espace.

Les gènes sont activés par des éléments amplificateurs ou "enhancers" qui peuvent stimuler l'expression des gènes en contactant le promoteur de ces gènes à longue distance.

Cette activation transcriptionnelle à longue distance a lieu notamment grâce au rapprochement physique entre l'élément régulateur et le promoteur du gène cible via une boucle ou loop. L'organisation du génome en TADs suggère qu'un enhancer peut réguler l'expression d'un gène situé dans le même TAD dû à une fréquence d'interaction plus accrue entre régions génomiques situées dans le même TAD.

La perte de la structure en TAD pourrait causer des interactions ectopiques entre enhancer et promoteur situés dans des TADs voisins. Ces interactions ectopiques pourraient résulter en une dérégulation de l'expression génique.

Objectifs :

L'objectif principal du projet a été d'identifier et de caractériser de nouvelles interactions qui pourraient résulter de la déplétion d'une protéine insulatrice majeure connue chez la drosophile, appelée BEAF32. La déplétion de la protéine insulatrice BEAF32 engendre-t-elle de nouvelles interactions entre les différents domaines d'association topologique chez *Drosophila melanogaster* ?

Il est nécessaire d'ajouter aux objectifs de l'évaluation de l'efficacité de la déplétion du gène BEAF32, car pour que nos analyses aient un sens, nous devons nous assurer que cette manipulation a fonctionné.

De surcroît, un clustering devra être effectué afin de voir si nous pouvons rattacher les différentes conditions aux différents clusters.

La variabilité de l'expression génique autour des bordures TAD après déplétion de BEAF32 a été examinée. Cela a été fait dans le but d'évaluer l'hétérogénéité de l'expression des gènes autour des bordures TAD en condition BEAF32_KD par rapport à la condition non déplétée pour comprendre comment cette déplétion affecte la variabilité de l'expression génique. Les questions soulevées étaient les suivantes :

Comment la variabilité de l'expression génique est-elle affectée autour des bordures TAD après déplétion de BEAF32 ?

La suppression de BEAF32 entraîne-t-elle une augmentation de la variabilité d'expression des gènes autour des bordures TAD ?

Après cela, les variations d'interactions entre les promoteurs et les pics ATAC ainsi que ceux situés aux bordures des TAD entre les différentes conditions expérimentales ont été analysées. L'objectif serait de vérifier que toutes les interactions enhancer-promoter trouvées comme significatives se trouvent entre 2 TADs différents (que la BEAF32 qui les sépare est au niveau d'une bordure). Cela permet de caractériser les nouvelles interactions. Les problématiques étaient les suivantes :

Comment les interactions entre les promoteurs et les pics ATAC situés aux bordures des TAD varient-elles entre les différentes conditions expérimentales ?

Quelles nouvelles interactions entre les promoteurs et les pics ATAC des bordures TAD qui sont observées après la déplétion de BEAF32 ?

Équipe :

Le projet a été réalisé par Emma Rodriguez et Maëva Abadie--Lima, étudiantes en première année de master Bioinformatique et Biologie des systèmes à l'Université Paul Sabatier. L'équipe encadrante travaillant sur ce sujet est composée du chef de projet Olivier Cuvier, chercheur INSERM à l'Unité de biologie moléculaire, cellulaire et du développement du Centre de biologie intégrative de Toulouse (MCD/CBI – CNRS, Université Toulouse III – Paul Sabatier). Ainsi que de Robel Tesfaye, post-doctorant en bioinformatique, épigénétique et dynamique de la chromatine, et de Naomi Schickele, étudiante doctorante dans cette équipe.

MATERIEL et METHODE

Matériel

Les données analysées résultent d'expérimentations Singlecell Multiome, intégrant des données de séquençage ARN (scRNA-seq) et d'accessibilité chromatine (scATAC-seq) sur des échantillons de *Drosophila Melanogaster*, intégrés *in silico* via 10x Genomics. Des données issues de la chromatine immunoprecipitation (ChiPseq) des sites BEAF32 ont été effectuées afin de déterminer les sites de liaison des protéines BEAF32 spécifiques sur le génome. Ces données ont été générées par l'équipe d'Olivier Cuvier. Ces données comportent une condition où la protéine de la luciférase a été déplétée, ce qui représente notre condition contrôle, et nous avons une condition où la protéine insulatrice BEAF32 a été déplétée, qui est notre condition expérimentale. Deux réplicats par condition ont été effectués.

Logiciel utilisé

L'analyse des données *in silico* a été effectuée en utilisant le logiciel R, version 4.0.2, sur le serveur Genotoul de l'INRAE (Institut National de Recherche Agronomique-Laboratoires)

Cela a permis de travailler sur un environnement sécurisé pour le stockage et le traitement des données, évitant ainsi la fuite des données et le stockage local de données volumineuses sur nos ordinateurs personnels.

Plusieurs bibliothèques R ont été utilisées pour cette analyse, telles que Seurat et Signac pour l'analyse intégrée des données issues du scRNA-seq et scATAC-seq, TxDb.Dmelanogaster.UCSC.dm6.ensGene de FlyBase pour l'accès aux annotations génomiques de la drosophile, ggplot2 et cowplot pour la visualisation des données, ainsi que magrittr, tidyverse, et rtracklayer pour la manipulation des données. Des objets Seurat ont été créés avec les fonctions CreateSeuratObject() (scRNAseq) et CreateChromatinAssay() (scATACseq).

Pour faciliter la collaboration et le partage sécurisé des scripts, un dépôt GitLab privé, et un serveur discord ont été créés.

Méthode

Contrôle Qualité

Le contrôle qualité vise à ne conserver que les cellules que l'on peut exploiter pour les analyses. Pour cela il faut identifier des gouttelettes non vides, à minimiser le nombre de doublets et à garantir la présence de cellules vivantes. La filtration des cellules a été réalisé avec la fonction subset de R afin d'appliquer des seuils définis à partir des analyses de qualité suivantes :

Pour le scRNA-seq, les cellules ont été filtrées selon les seuils des quantiles pour **nCount_RNA** (nombre total de molécules d'ARN détectées dans une cellule) et **nFeature_RNA** (nombre de gènes détectés dans chaque cellule), éliminant ainsi les cellules avec des valeurs extrêmement basses ou élevées qui pourraient indiquer des gouttelettes vides, des cellules mortes ou des doublets. Les seuils minimum et maximum fixés pour ces mesures sont les quantiles 5% et 95% respectivement.

Les cellules avec un pourcentage de transcrits mitochondriaux (**Percent of mitochondrial counts**) supérieur à la médiane ont été exclues pour minimiser l'impact des cellules potentiellement

endommagées ou en train de mourir. Ce pourcentage a été calculé avec la fonction `PercentageFeatureSet()`.

La détection et l'élimination des doublets a été réalisée avec les fonctions `scrublet.Scrublet()` et `scrub_doublets()` de l'outil *Scrublet*, un module Python puis intégré dans R via le package *reticulate* (**Doublets**). Cet outil fonctionne en créant des doublets artificiels et en marquant les cellules dont le profil ressemble à celui de ces doublets. Les gouttelettes du *singlecell* ayant capturé deux cellules (doublets) ont été retirées de l'ensemble de données pour assurer que seules les cellules uniques soient analysées.

Pour les données de scATAC-seq, les signaux nucléosomaux ont été calculés (**Nucleosomal signals**). Le ratio des fragments mononucléosomiques aux fragments libres de nucléosomes est calculé via la fonction `NucleosomeSignal()` car un ratio trop élevé peut signifier une cellule mourante ou endommagée, or, nous souhaitons que des cellules vivantes.

Le score d'enrichissement des sites de début de transcription (**TSS enrichment score**), calculé par la fonction `TSSEnrichment()`, a permis d'évaluer la proportion du signal ATAC aux sites TSS par rapport aux régions adjacentes.

Un faible score d'enrichissement des pics ATAC au niveau des TSS peut indiquer une faible expression génique, assimilant ainsi le signal à un bruit de fond. Les cellules avec un score d'enrichissement des pics ATAC au niveau des TSS inférieur au seuil défini ont été éliminées.

Analyses préliminaires et Hétérogénéité Cellulaire

Les données scRNAseq ont été normalisées en utilisant la fonction `SCTtransform()` de Seurat. Ceci pour stabiliser la variance des comptages de gènes tout en minimisant les effets dus aux différences de profondeur de séquençage.

Une Analyse en Composantes Principales (PCA) a été réalisée pour réduire la dimensionnalité des données RNAseq avec `RunPCA()`.

Pour les données scATACseq, les fonctions `FindTopFeatures()` et `RunTFIDF()` ont été utilisées pour identifier les pics les plus significatifs et transformer les données en vue d'une réduction de dimensionnalité par SVD (Décomposition en Valeurs Singulières) `RunSVD()`, similaire à LSI.

Le nombre de composantes pour chaque analyse RNAseq et ATACseq à retenir pour la suite a été déterminé graphiquement à l'aide de `ElbowPlot()`.

Une seconde réduction d'information, non linéaire cette fois, a été effectuée séparément aux données réduites par PCA et LSI en utilisant UMAP (Uniform Manifold Approximation and Projection) avec `RunUMAP()` et le t-SNE (t-distributed Stochastic Neighbor Embedding) avec `RunTSNE`. Cette opération a pour but de résumer et projeter la proximité entre les cellules dans un espace bidimensionnel permettant de distinguer les groupes ou clusters de cellules avec des profils similaires.

L'analyse a tiré parti de la nature bimodale des données en utilisant la fonction `FindMultiModalNeighbors()` pour appliquer la méthode du voisin le plus proche pondéré (WNN).

Le WNN calcule un graphe de voisins joints en intégrant à la fois les données scRNAseq et scATACseq où les cellules sont regroupées selon leur similitude à la fois génétique et épigénétique.

Les clusters ont ensuite été identifiés à l'aide de la fonction `FindClusters()` avec une résolution de 0.8, afin de contrôler la finesse des clusters, en se basant sur le WNN. Cette résolution élevée augmente le nombre de clusters identifiés. Cela mène à des clusters plus petits et plus spécifiques.

L'efficacité de la déplétion de BEAF32 a dû être contrôlée. Pour cela, les densités d'expression de la protéine BEAF-32 ont été créées en utilisant la fonction `RidgePlot()` du package Seurat. Les tracés ont

étaient configurés pour regrouper les données d'expression génique par échantillon (BEAF32-KD1, BEAF32-KD2, Luc-KD1, LucKD2). L'expression de la protéine Beaf32 a été visualisée à travers les UMAP précédente RNA et ATAC, par condition (BEAF32, Luc) avec la fonction FeaturePlot().

Intégration des données Pseudo-bulk et single-cell

Pour transformer des données en PSEUDO-BULK, les données ont été regroupées en conditions BEAF32KD et LucKD. Une variable *condition* a été ajoutée à l'objet Singlecell pour identifier la condition de chaque cellule analysée. Les gènes différentiellement exprimés entre les deux conditions ont été identifiés à l'aide de la fonction FindMarkers(). Les résultats ont ensuite été filtrés pour ne conserver que les gènes ayant une augmentation d'expression en condition BEAF32_KD. Pour mesurer cela, il faut utiliser la moyenne du changement de l'expression génique exprimée en logarithme en base 2 (log2FC). Nous avons ensuite appliqué un seuil supérieur à 0.3 à ce log2FC avec une p-valeur ajustée inférieure à 0.05. Les noms des gènes ainsi filtrés ont été extraits.

Les différences de signal ATAC entre les conditions "BEAF32-KD" et "Luc-KD" ont été évaluées à l'aide de la fonction FindMarkers(), en regroupant par condition. Les résultats ont ensuite été filtrés pour ne conserver que les régions présentant un changement de signal d'accessibilité de la chromatine, qui peut être mesuré log2FC qui ici, quantifie le changement relatif d'accessibilité de la chromatine entre les deux conditions. Un seuil à 0.3 et une p-valeur ajustée inférieure à 0.05 ont été ensuite appliqués. Les noms des régions ainsi filtrées ont été extraits.

Les pics ATAC ont été associés aux gènes identifiés précédemment. La fonction LinkPeaks() a été utilisée sur l'objet Singlecell, en utilisant les données d'ATAC (assay "ATAC") et l'expression génique normalisée (assay "SCT") pour l'expression génique. Les gènes utilisés pour l'association ont été sélectionnés parmi les gènes différentiellement exprimés précédemment identifiés, et ceci pour réduire le temps de calcul de cette fonction. Cette fonction permet d'associer un ou plusieurs pics ATAC à un gène d'intérêt via une analyse de corrélation entre le signal du pic ATAC et l'expression génique. La relation entre le signal ATAC et l'expression génique a été examinée par la corrélation de Pearson. Afin d'estimer la significativité de la corrélation, la corrélation est comparée à des corrélations background (associations peu plausibles) entre le gène d'intérêt et des pics ATAC situés sur d'autres chromosomes. Les gènes associés à une augmentation de l'accessibilité de la chromatine ont été identifiés. Cela permet de filtrer parmi les gènes identifiés précédemment comme différentiellement exprimés, ceux où l'accessibilité à la chromatine a augmenté en condition BEAF32.

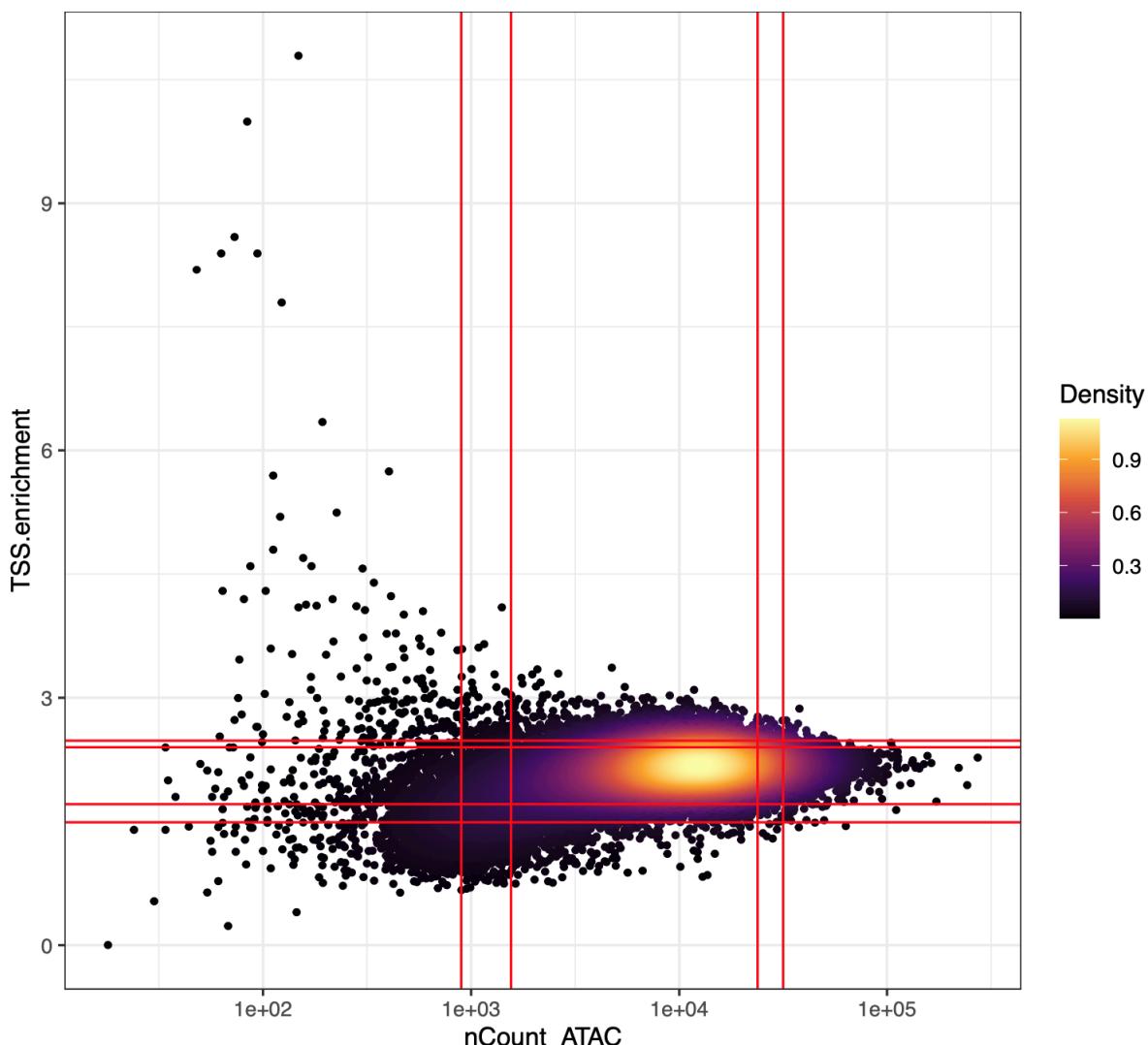
Les gènes obtenus seront les gènes significativement différentiellement exprimés avec des pics différentiellement actifs, permettant de conclure quant à notre problématique. Un DotPlot a été généré pour ces gènes afin de visualiser leur expression dans les clusters et dans les différentes conditions. Les profils de couverture pour ces gènes ont été générés avec la fonction CoveragePlot() et ExpressionPlot().

RÉSULTATS

Bilan du contrôle qualité

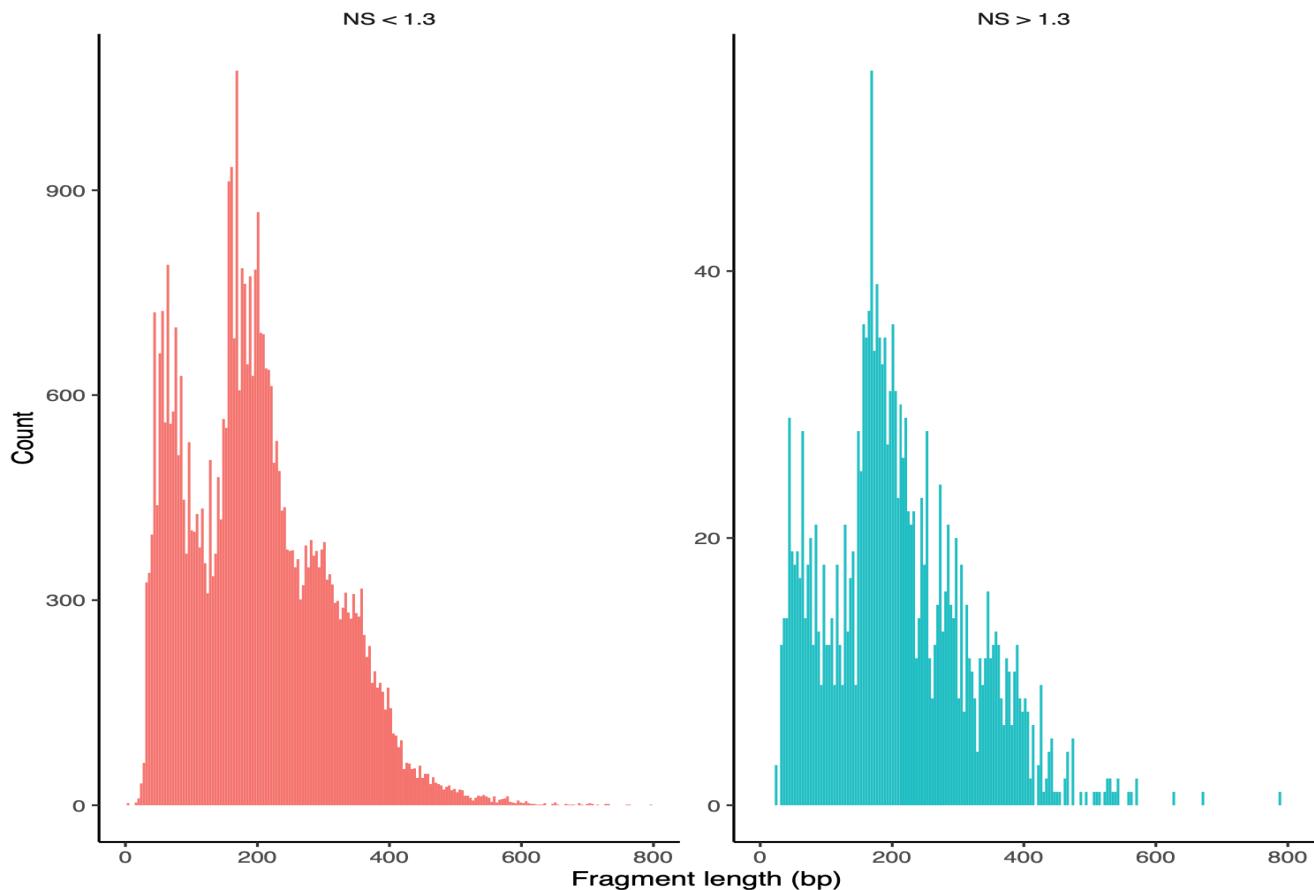
Quantiles

nCount_ATAC: 5%:896 10%:1555.2 90%:23786 95%:31544.8
TSS.enrichment: 5%:1.49 10%:1.71 90%:2.4 95%:2.48



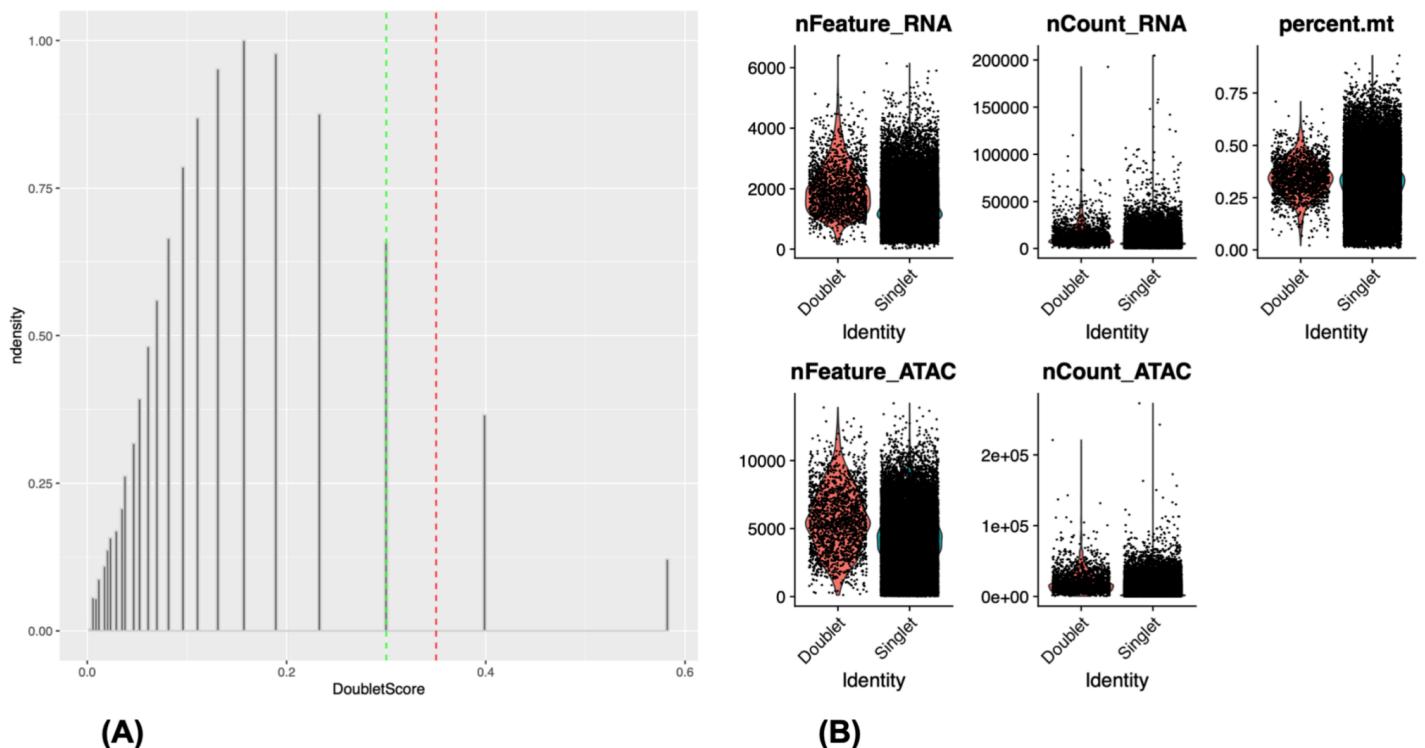
[Figure1 - DensityScatter de la proportion du signal ATAC aux sites TSS par rapport aux régions adjacentes des sites de démarrage de la transcription en fonction du nombre de fragments alignés dans une région du génome (nCount_ATAC) et donc de l'accessibilité de la chromatine. Plus le nCount_ATAC est élevé et plus la chromatine est accessible.]

Le densityScatter (Figure1) permet d'observer que l'essentiel des cellules ont un score d'enrichissement des TSS en pic ATAC entre 1,7 et 2,5 (noyau dense en jaune). Grâce aux quantiles apparaissant au-dessus du graphique, il est possible de voir que 5% de l'échantillon se situe en dessous d'un TSS enrichment de 1,49. C'est pourquoi une valeur seuil de 1 a pu être établie. Les cellules ayant donc une valeur de TSS sous ce seuil ont pu être écartées pour la suite des manipulations.



[Figure2 - Nucleosome signal. Histogramme du nombre de fragments en fonction de la taille des fragments. En rouge 95% des cellules, filtrées grâce à un nucléosome signal inférieur à 1,3. En bleu, 5% des cellules, filtrées grâce à un nucléosome signal supérieur à 1,3]

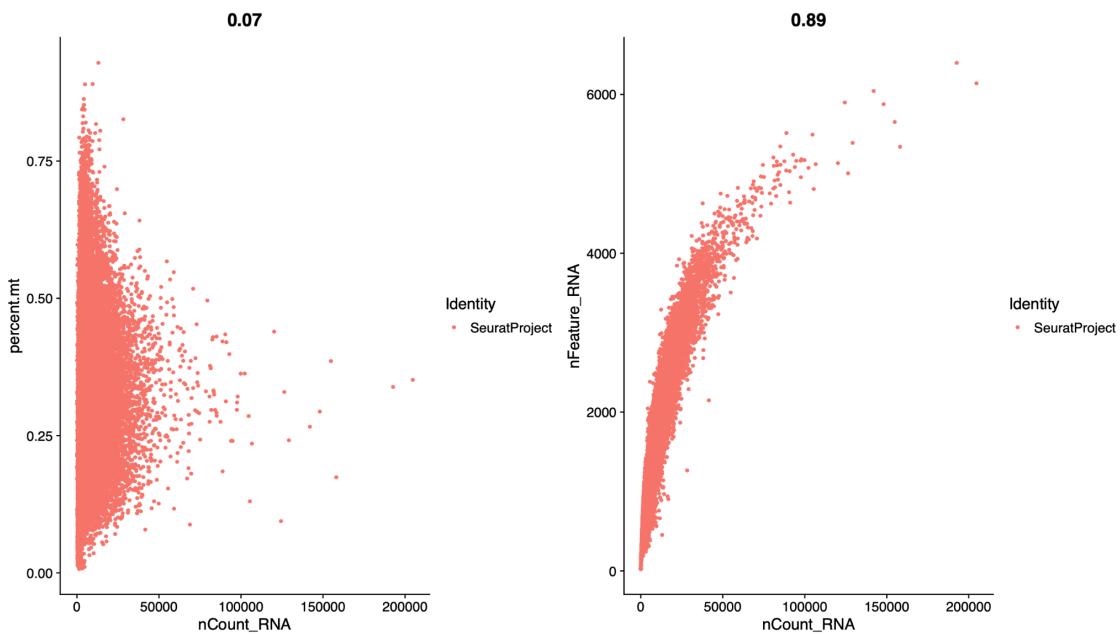
Le nucléosome signal permet d'évaluer la distribution des reads ATAC autour des nucléosomes, ce qui a pour but de s'assurer de l'organisation de l'ADN cellulaire en nucléosomes. Cet histogramme de fragment (**Figure2**) nous permet d'observer la taille des fragments selon si le nucléosome signal est inférieur à 1,3 ou supérieur à 1,3. Ce seuil a été déterminé à partir du quantile à 95%. Le plot rose avec le signal inférieur à 1,3 représente 95% des cellules. Si le fragment est inférieur à 147pb, alors il n'a pas de nucléosome (nucleosome free). Si il est 1 fois supérieur à 147pb alors il y a un nucléosome. Au delà de 1 fois supérieur, il est poly-nucléosomal. Plusieurs pics sont observables, un aux alentours de 50pb et un aux alentour de 200pb. Celui correspondant à 50pb signifie que ce sont des fragments sans nucléosome tandis que celui à 200pb correspond à un signal mono-nucléosomal. Cette figure atteste dans un premier temps d'une digestion enzymatique réussie et dans un second de l'accessibilité de la chromatine car des signaux mono-nucléosomiques correspondent plutôt à de l'euchromatine alors que des signaux poly-nucléosomiques correspondent plutôt à de l'hétérochromatine.



[Figure3 A - Histogramme des scores de doublets. Densité normalisée du nombre de cellules en fonction des scores de doublet. En rouge: valeur seuil trouvé sur scrublet à partir de laquelle une gouttelette avec un score de doublet strictement supérieur est considéré comme étant des doublets. En vert : valeur seuil choisie à partir de laquelle une gouttelette avec un score strictement supérieur est considérés comme étant des doublets.]

[Figure3 B - Violin Plot montrant , le nombre d'ARN identifié (nFeature_RNA), le nombre total d'ARN (nCount_RNA), le pourcentage d'ADN mitochondrial (percent.mt), nombre de régions de chromatine accessibles détectées dans l'échantillon (nFeature_ATAC) et la somme totale de toutes les lectures dans toutes les régions de chromatine accessibles (nCount_ATAC), en fonction de si la gouttelette contient deux cellules (doublet) ou une seule (Singlet).]

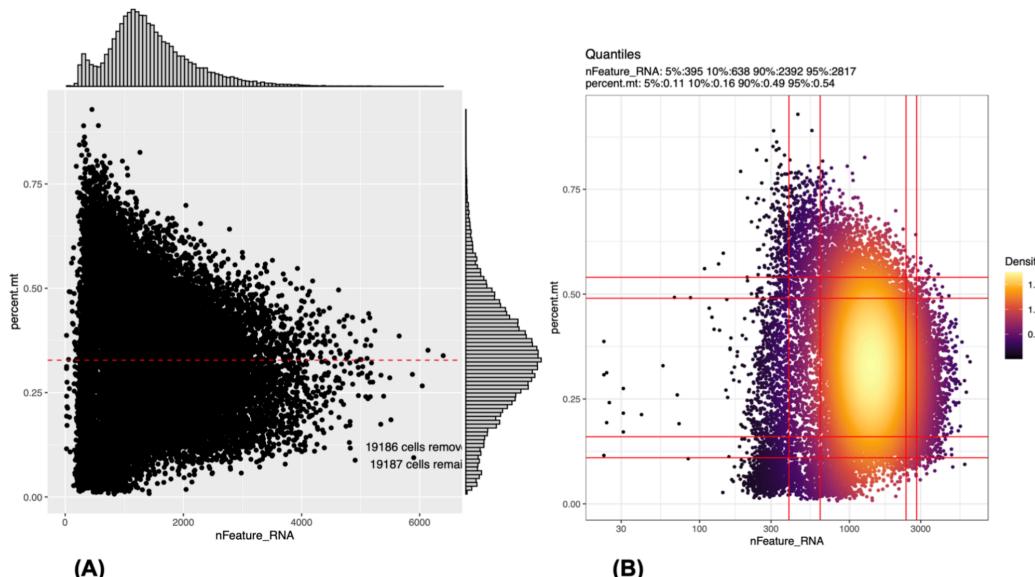
Afin d'identifier les gouttelettes contenant plus d'une cellule, l'outil scrublet a été utilisé. En bref, l'outil génère des cellules avec des caractéristiques moyennes entre différents groupes de cellules, ou doublets artificiels, puis estime la ressemblance entre chacune des cellules avec les cellules doublets artificiels. La figure 3A est un histogramme permettant de voir les scores de doublets. Grâce au seuil optimal de 0.3 trouvé avec l'outil scrublet, l'outil estime que 1822 cellules sont des doublets ce qui représente 4,9% des cellules totales. La figure 3B permet d'observer la distribution des cellules selon s'il s'agit de singulets ou de doublets prédis. Ces deux figures permettent d'identifier les doublets afin de les exclure des prochaines analyses.



[Figure4 -Scatter Plot. A gauche le pourcentage d'ARN mitochondrial dans les cellules en fonction de l'ARN total. A droite, le nombre d'ARN identifiés en fonction de l'ARN total.]

Le scatter plot en Figure4 montre à gauche la relation entre le pourcentage de transcrits mitochondriaux et le nombre total d'ARN par cellule, et ne permet pas de mettre en évidence une corrélation entre les 2 (coefficient de Pearson de 0.07). Le pourcentage de transcrits mitochondriaux varie beaucoup d'une cellule à l'autre, indépendamment du nombre total d'ARN mesuré dans ces cellules. Un pourcentage élevé de transcrits mitochondriaux signifie que les cellules pourraient être en état de stress ou en processus de mort cellulaire.

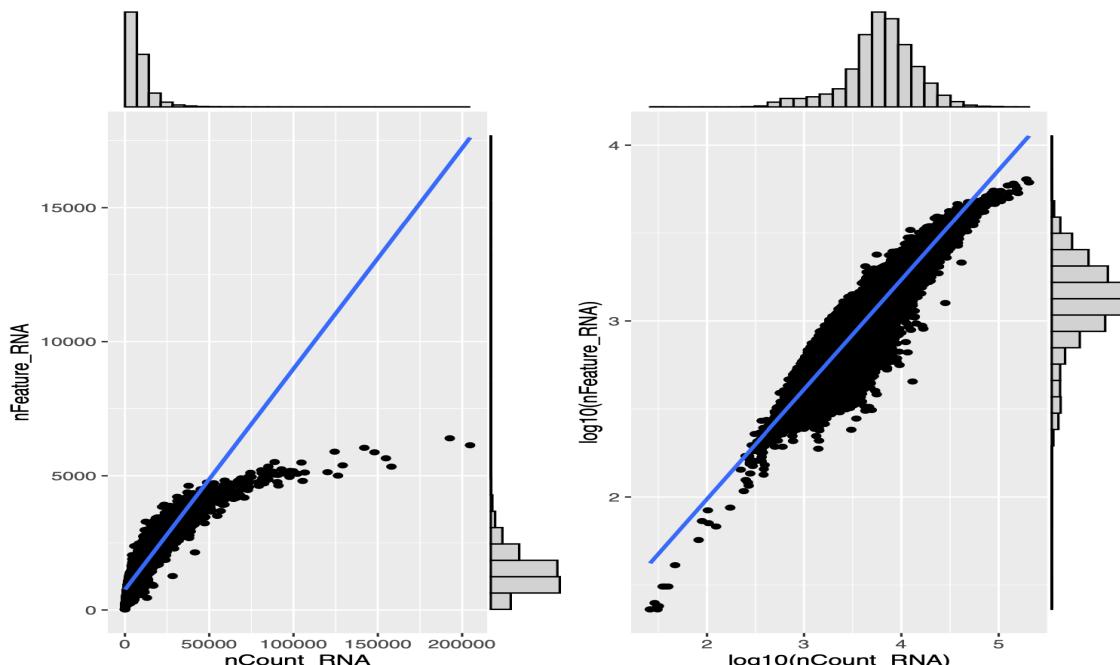
A droite, est examinée la relation entre le nombre de gènes détectés et le nombre total d'ARN par cellule. Une forte corrélation positive est mise en évidence (coefficient de Pearson de 0.89). Par conséquent, plus une cellule contient d'ARN total, plus elle exprime un nombre varié de gènes. Une divergence de cette tendance linéaire est observée dans des cas extrêmes ou outliers, notamment au-delà de 5000 gènes détectés (nFeature_RNA), ce qui pourrait être expliqué par la présence de doublets.



[Figure5 A -Feature Scatter. Pourcentage d'ARN mitochondrial en fonction du nombre d'ARN identifiés. En haut et à droite, un histogramme de la répartition des cellules respectivement selon le nombre d'ARN identifié et selon le pourcentage d'ARN mitochondrial. La médiane est représentée en ligne pointillée rouge. 19186 cellules se trouvent au dessus et sont éliminées et 19187 sont en dessous et sont conservées selon la légende.]

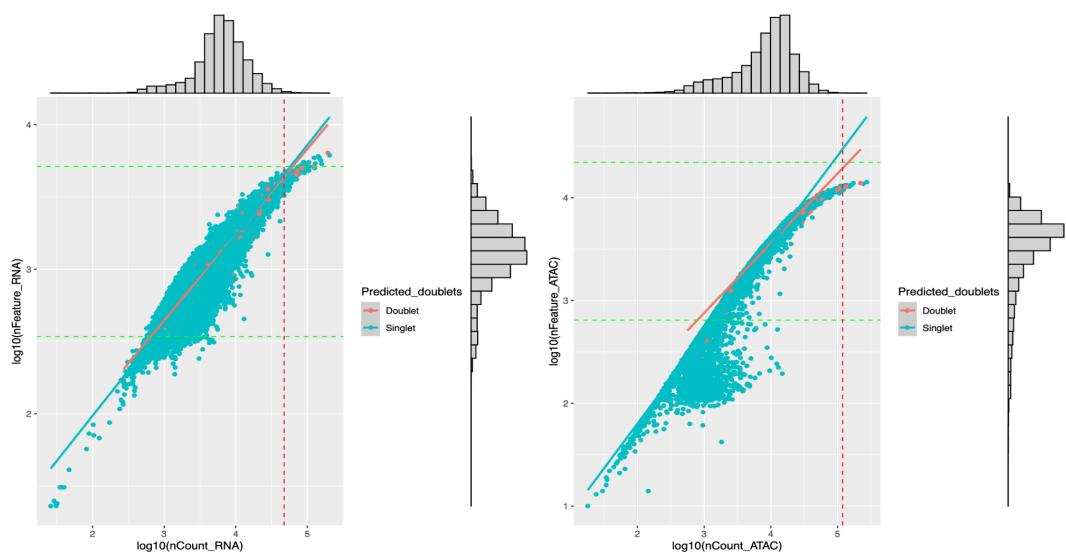
[Figure5 B -Density Scatter. Pourcentage d'ARN mitochondrial en fonction du nombre d'ARN identifiés. Au dessus, les différents quantiles à 5%, 10%, 90% et 95% apparaissent.]

Le density Scatter en **Figure 5**, permet d'identifier à partir de quel seuil, le pourcentage d'adn mitochondrial est trop élevé pour conserver les cellules. Nous observons à gauche que la médiane se trouve à 0,3. Le seuil est défini à la médiane et non à 95% parce qu'il s'agit d'une lignée cellulaire de drosophile. Le seuil est sévère afin de ne conserver que les cellules avec une grande diversité de transcrits afin d'observer l'effet sur le transcriptome nucléaire en général au lieu du transcriptome mitochondrial uniquement.



[Figure6 -Scatter Plot. A gauche Nombre d'ARN identifiés en fonction du nombre d'ARN total. A droite Nombre d'ARN identifiés en fonction du nombre d'ARN total après élimination des cellules ayant un pourcentage d'ARN mitochondrial au-dessus de la médiane. La répartition des données sous forme d'histogramme peut être observée au-dessus et à droite des deux figures. La droite bleue représente la droite qui passe par le plus de points (cellules) possibles.]

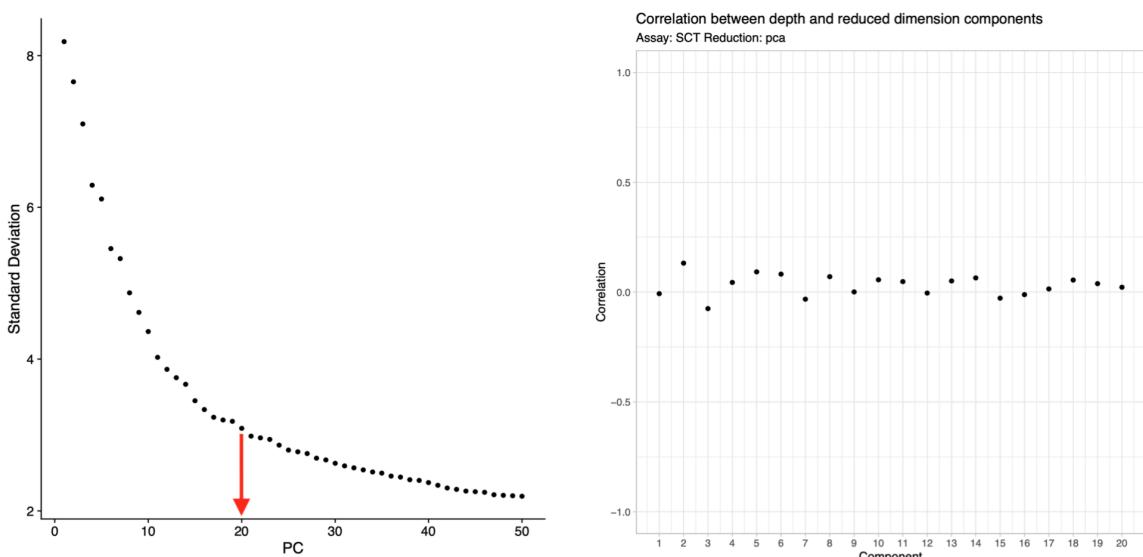
Le scatter plot en **figure 6** permet d'observer à gauche, l'absence de corrélation entre le nombre de gènes détectés et le nombre total d'ARN lorsque toutes les cellules sont présentes. Ces figures montrent la quantité d'ARN identifiée en fonction de l'ARN total. A droite, les cellules dépassant le seuil de pourcentage de transcrits mitochondriaux ont été exclues. Ainsi la corrélation entre le nombre de gènes détectés et le nombre total d'ARN est mis en évidence, ce qui est plus cohérent, d'où la nécessité de supprimer les cellules exprimant trop de gènes mitochondriaux.



[Figure 7 -Scatter Plot. A gauche Nombre d'ARN identifiés en fonction du nombre d'ARN total. A droite le nombre de régions de chromatine accessible détectées dans l'échantillon et la somme totale de toutes les lectures dans toutes les régions de chromatine accessibles.]

Le ScatterPlot en Figure7 permet de voir en rouge, même si peu sont visibles, les doublets prédis et en bleu les singulets. Une courbe de tendance est également tracée pour les doublets et les singulets représentant la corrélation entre le nombre de gènes détectés et le nombre total d'ARN. La différence est que pour la figure de gauche, il s'agit du RNAseq, et que la figure de droite traite l'ATACseq. Les courbes dans les deux conditions sont proches pour RNAseq mais moins pour ATACseq, cela peut signifier que la présence de doublet affecte plus particulièrement les signaux ATAC, d'où la nécessité de les enlever.

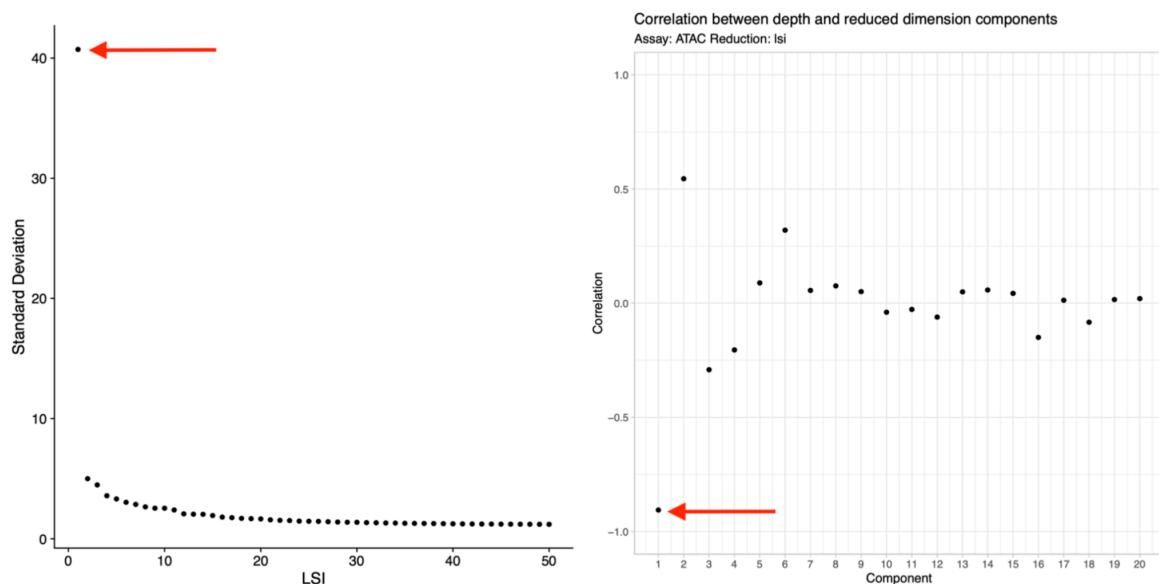
Bilan des analyses préliminaires



[Figure 8 -ElbowPlot et Analyse de la corrélation avec la profondeur du séquençage des données issues du RNAseq. A gauche, ElbowPlot montrant le poids des composantes en fonction des composantes

[Figure 8 - Elbow plot et Analyse de la Corrélation avec la Profondeur de Séquençage des données issues de l'ATACseq. A gauche, ElbowPlot montrant le poids des composantes principales en fonction des 20 premières composantes principales.]

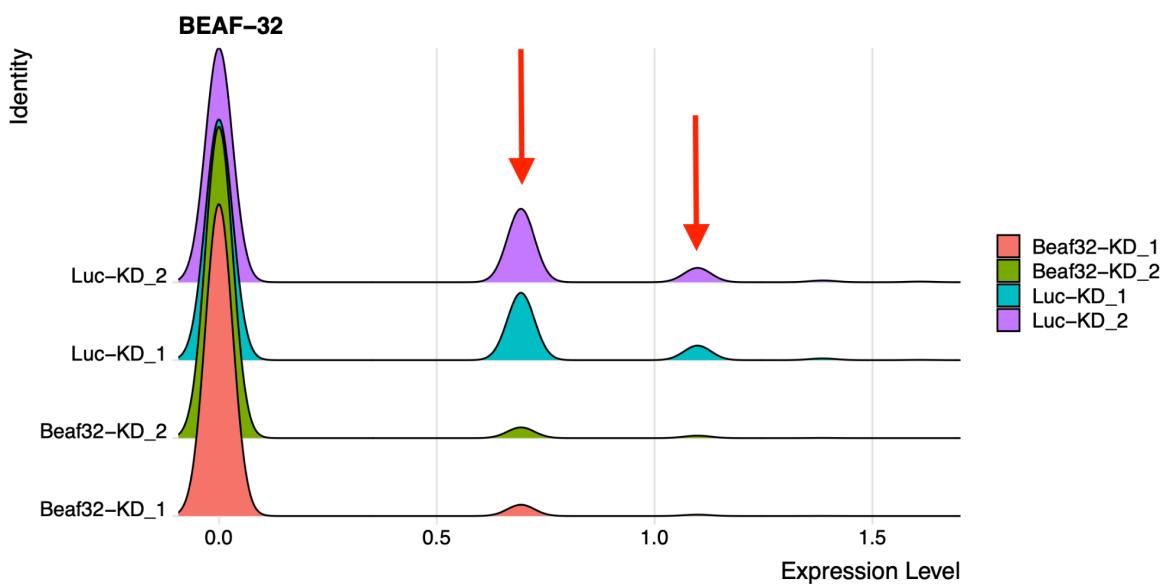
L'elbow plot de la Figure 8 montre en abscisse les composantes principales et en ordonné le poids de leur influence. Après la vingtième composante, l'ajout d'autre composante supplémentaire semble apporter des différences très faibles, seules les 20 premières seront donc conservées pour la suite des analyses du RNAseq. L'analyse de la Corrélation avec la Profondeur de Séquençage montre que les 20 composantes principales ne sont pas dépendantes de la profondeur de séquençage, indiquant que ces composantes n'expliquent pas une différence technique.



[Figure 9 - ElbowPlot et Analyse de la corrélation avec la profondeur du séquençage des données issues de l'ATACseq. A gauche, ElbowPlot montrant le poids des composantes principales en fonction des 20 premières composantes principales.]

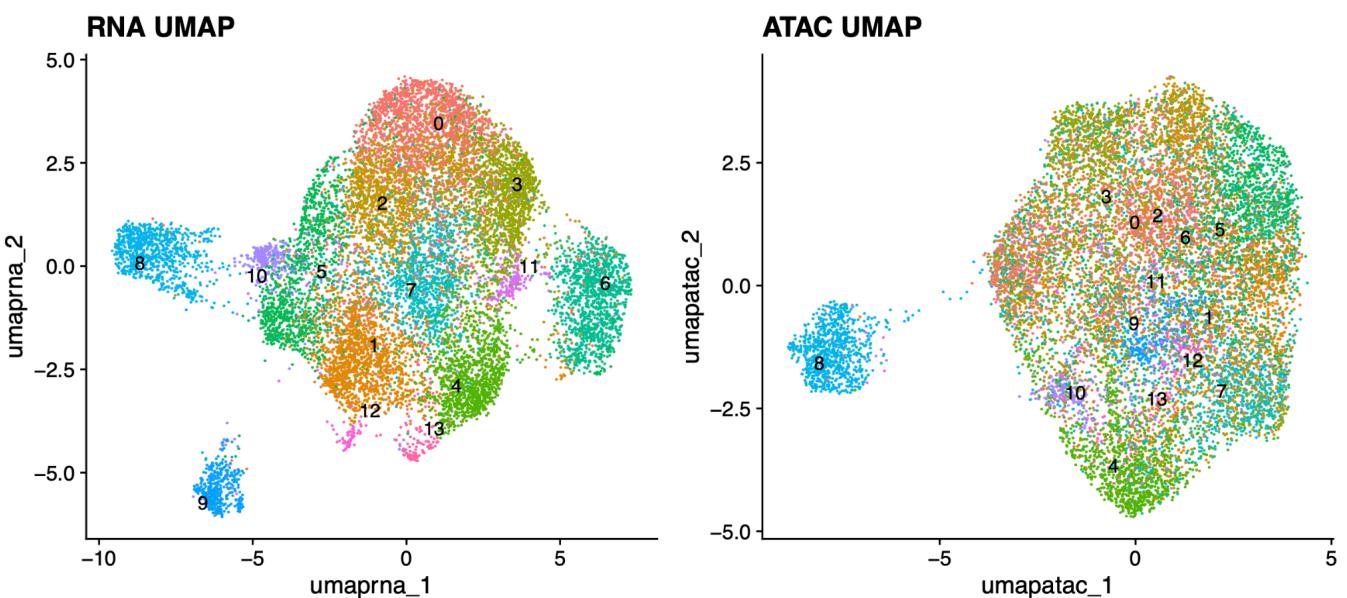
L'elbow plot de la figure 9 pour l'analyse des composantes principales de l'ATACseq, montre une première composante principale avec une valeur très élevée et les autres plus proches. Les 20 premières sont de nouveau sélectionnées car un point d'inflexion est visible autour de la quinzième et que par sécurité il est préférable d'en prendre plus. L'analyse de la Corrélation avec la Profondeur de Séquençage montre que la première composante est anti corrélée à la profondeur de séquençage. Cette composante est donc exclue pour la suite des analyses, car elle explique des différences associées au nombre de reads total par cellule.

Bilan sur l'Étude de l'Hétérogénéité Cellulaire



[Figure 10 RidgePlot. Visualisation des différents niveaux d'expression de la protéine BEAF32 dans les différentes conditions (Luciferase_knowdown1, Luciferase_knowdown2, Beaf32_knowdown1 et Beaf32_knowdown2).]

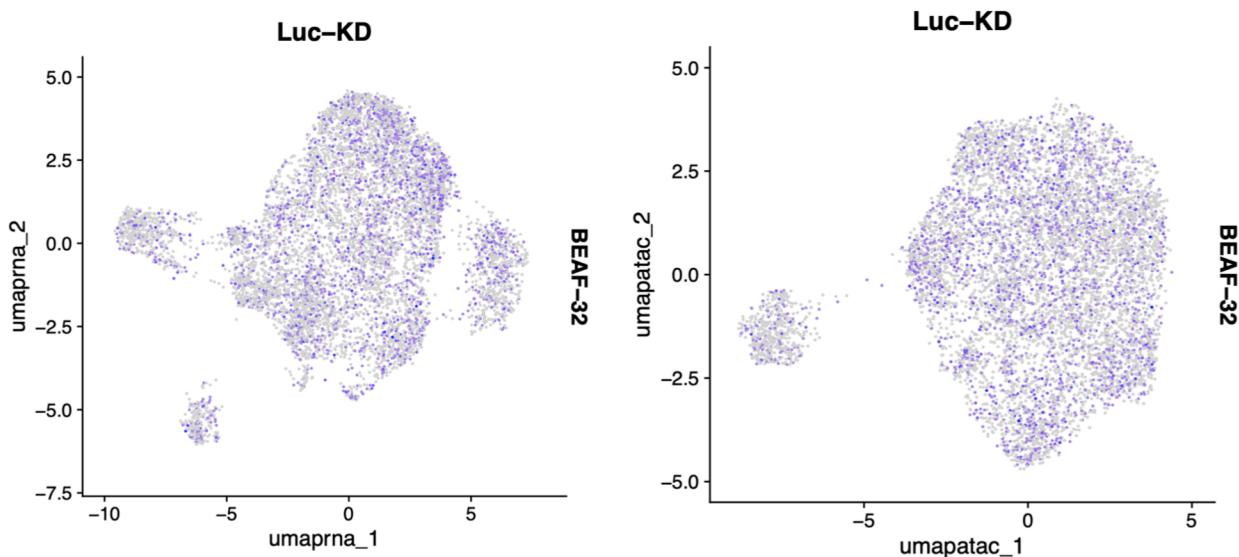
L'efficacité de la déplétion de BEAF32 a été évaluée grâce à un RidgePlot en figure 10. Pour les conditions LucKD1 et LucKD2 un certain niveau d'expression de BEAF32 est constaté. Ce niveau est presque inexistant dans les conditions BEAF32_KD1 et BEAF32_KD2. Ces résultats confirment que la déplétion de BEAF32 réduit de façon reproductible le nombre de cellules exprimant BEAF32.



[Figure 11 -UMAP (Uniform Manifold Approximation and Projection). A gauche, les différents clusters déterminés avec les données issues du RNAseq .A droite, les différents clusters déterminés avec les données issues de l'ATACseq.]

La seconde partie de l'hétérogénéité cellulaire portait sur l'identification de clusters de cellules similaires à l'aide de méthodes de clustering. Cette analyse a révélé 14 clusters distincts.. Des facteurs autres que la déplétion de BEAF32 influencent la séparation des sous-populations cellulaires visibles en figure 11.

La structure et fonction de chaque sous population de cellules des clusters sont trop similaires pour les différencier et leur associer des marqueurs et donc leur associer une condition.



[Figure 12 -UMAP (Uniform Manifold Approximation and Projection) avec recherche de BEAF32. A gauche, la UMAP a été déterminée avec les données issues du RNAseq .A droite, la UMAP a été déterminée avec les données issues de l'ATACseq. En bleu, sur chaque, apparaissent les cellules exprimants Beaf32.]

De plus, comme il est observable sur la figure 12, la protéine insulatrice BEAF32 est présente de façon éparsé dans les conditions Luc_KD.

La conclusion de ces analyses est qu'il est mieux de poursuivre les analyses en séparant les données par condition plutôt que par clusters, ce qui a été fait pour la suite de l'étude avec un pseudo BULK.

Bilan de l'intégration des données Pseudobulk et single-cell

Identification des gènes différentiellement exprimés

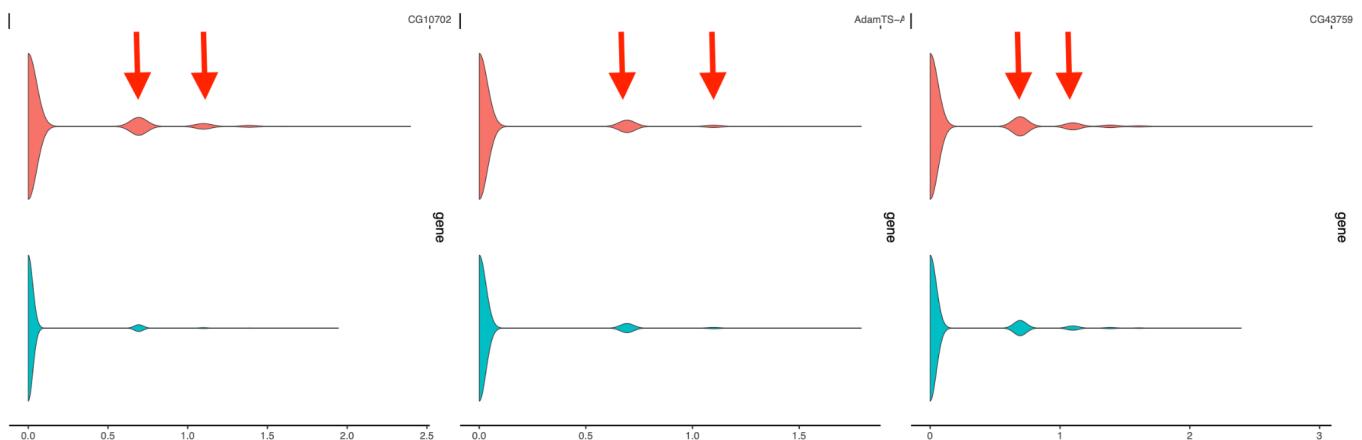
Afin d'identifier l'impact de la déplétion de BEAF32 sur la transcription génique des gènes situés aux frontières des TADs (Topologically Associating Domains), il a été comparé l'expression génique entre les conditions Beaf32-KD et Luc-KD. Cette analyse a révélé que 94 gènes montrent des différences significatives d'expression, indiquant un potentiel rôle de ces gènes dans la régulation de la structure chromatinnienne. Les six gènes qui semblent avoir le plus de différence d'expression sont CG7299" "CG16713" "CG10702" "CG32037" "CG42807" "Cyp12a4".

Évaluation de la variabilité du signal ATAC

Pour évaluer l'impact de la déplétion de BEAF32 sur la variabilité du signal ATAC, qui est une mesure de l'accessibilité de la chromatine, il a été comparé les profils ATAC-seq entre les conditions Beaf32-KD et Luc-KD. Cette analyse a permis d'identifier 32 signaux ATAC présentant des différences significatives d'accessibilité, indiquant des changements potentiels dans l'organisation de la chromatine qui pourraient influencer la régulation génique. Les six signaux les plus marqués par cette différence se situent dans les régions génomiques 2R-14771639-14772572, X-13713303-13714139, 2L-19069642-19070402, X-8554059-8554940, X-4129135-4129967, et 3R-15461777-15462655.

Évaluation de la corrélation de la relation entre le signal ATAC et l'expression génique

Nous avons trouvé trois gènes, localisés en bordure TAD, CG10702, AdamTS-A et CG43759 dont l'expression génique et l'accessibilité de la chromatine augmente en condition BEAF32KD.



[Figure 13 -Expression Plot. Le premier sur le gène CG10702, au milieu sur le gène AdamTS-A et à droite sur le gène CG43759. En rouge la condition Beaf32_KDn et en bleu la condition Luc_KD. Des flèches rouges ont été ajoutées afin de souligner là où il y a une augmentation de l'expression des gènes.]

Des expressions plot en **figure 13** ont permis de mettre en évidence l'augmentation de l'expression génique tandis que les coveragePlot (annexe **figure 14**, **figure 15 & figure 16**) mettent en évidence les augmentations d'accessibilité à la chromatine en condition BEAF32_KD .

On observe pour les trois gènes, une augmentation significative de l'expression génique en condition BEAF32KD, au niveau des sites BEAF32.

Pour le gène CG10702, les pics atac sont à une distance de 1kb du promoteur, et à 10kb pour AdamTS-A.

Cette augmentation de l'expression génique pourrait être causée par l'accessibilité du promoteur en l'absence de la protéine insulatrice BEAF32, et non pas par de nouvelles interactions...

CONCLUSION GÉNÉRALE

Conclusion

Notre objectif final était de comprendre si, lorsque l'on perturbe la structure des domaines d'association topologique, des nouvelles interactions entre les gènes des différents domaines se créent. La perturbation de cette structure a été permise par la déplétion de BEAF32.

La déplétion de BEAF32 entraîne de nouvelles interactions au niveau des gènes situés aux frontières TAD, notamment aux sites BEAF-32, qui se traduit par une augmentation de l'expression génique.

Ces observations suggèrent que la perte de BEAF-32 modifie l'accessibilité du promoteur et peut induire des changements dans l'expression génique par des mécanismes de régulation épigénétique.

L'incapacité d'associer des clusters aux conditions des données bulk est probablement lié au fait dans une lignée cellulaire, l'hétérogénéité cellulaire est réduite comparée à des échantillons d'expérience animal. La culture *in vitro* a pour effet de sélectionner des sous-populations homogènes et que la déplétion de BEAF32 n'est pas un facteur assez discriminant pour qu'il soit la seule cause de la création de différents clusters.

Il est observé que la déplétion de la protéine insulatrice BEAF32 ne conduit pas à des modifications transcriptionnelles majeures. Cela est en concordance avec la déplétion de CTCF (facteur de transcription et un isolant transcriptionnel impliqué dans l'organisation des domaines topologiquement associés), qui montre aussi peu d'effet sur la régulation du transcriptome.

D'autres protéines insulatrices chez la drosophile, telles que M1BP, IBF1, IBF2, et Chromator, peuvent compenser la perte de BEAF-32.

Étant donné que 94 gènes différentiellement exprimés ont été trouvés, dont seulement 3 où une meilleure accessibilité à la chromatine, ces 91 gènes restants pourraient provenir de cette compensation.

Limites

La technologie single-cell présente des limitations inhérentes, notamment en ce qui concerne la profondeur de séquençage et le nombre de cellules analysées. Le nombre de gènes détectés est fortement dépendant de ces paramètres, ce qui peut entraîner une non-détection des gènes dérégulés.

En effet, en singlecell, les gènes qui vont être les plus dominants sur l'analyse différentielle et qui vont "écraser" les gènes faiblement exprimés vont être capturés. Si des interactions existent uniquement sur des gènes peu abondants en ARN, elles peuvent ne pas être détectées. La variabilité observée peut concerner des gènes peu actifs dans toutes les cellules, entraînant une sous-détection dans les analyses.

Les méthodes de réduction de dimensionnalité, telles que PCA ou UMAP, visent à extraire les principales variations dans les données, ce qui implique que des variations mineures mais potentiellement importantes, peuvent être perdues. Les gènes peu détectés et très variables peuvent être exclus s'ils ne font pas partie des variations majeures conservées.

OUVERTURE

La question subsidiaire est comment justifier cette augmentation d'expression des 94 gènes différemment exprimée, alors que l'on a pas d'augmentation significative de l'activité des peaks associés ?

Une hypothèse se tournerait vers les changements dans la conformation des TADs.

En effet, les changements dans les TADs déclenchés par des stress, peuvent mener à des expressions géniques différencierées qui sont essentielles pour la réponse et l'adaptation au stress. Des gènes qui contribuent à la réponse au stress pourraient être up-régulés par la relocalisation de leurs régions régulatrices à l'intérieur de TADs spécifiques.**[4]**.

Des mécanismes similaires auraient pu se mettre en place lors de la déplétion de la protéine BEAF32 en réponse au stress cellulaire que la déplétion aurait engendré.

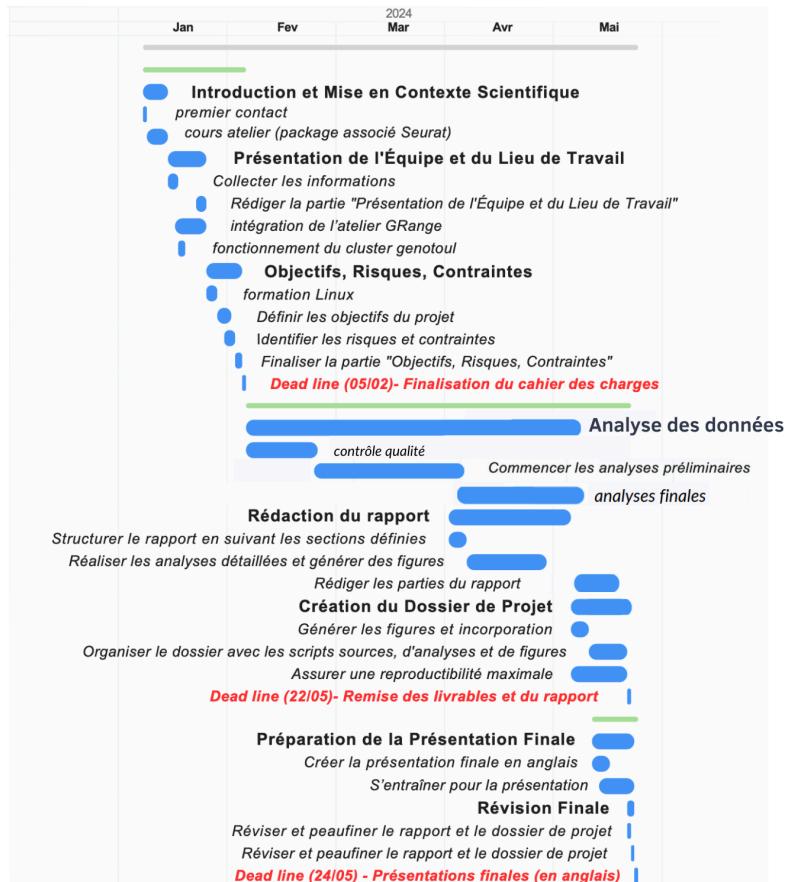
Il serait alors intéressant de quantifier le stress engendré pendant l'expérimentation et de caractériser plus en détails les changements de conformation des TADs, pour valider ou non cette hypothèse.

GESTION DE PROJET

Des réunions hebdomadaires ont été tenues presque tous les mercredis matin, de 9h30 à 12h30, au sein du Centre de Bioinformatique (CBI), dans les locaux occupés par les encadrants.

L'objectif a été d'instaurer des séances de travail où des questions pouvaient être posées, qu'elles soient d'ordre biologique, concernant la cohérence des manipulations avec le contexte biologique, ou d'ordre informatique, en cas de dysfonctionnement avec des commandes.

Calendrier



Au cours du mois de février, le contrôle qualité a été réalisé, opérant un tri des cellules à inclure ou exclure de nos analyses ultérieures. Dans la mesure du possible, une avancée autonome a été priorisée. Le cahier des charges a été rendu.

Les mois de mars et avril ont été principalement dédiés à la pré-analyse et l'analyse des données. Cette phase s'est avérée plus complexe, car contrairement au contrôle qualité, il existe peu de documentation ou pipeline concernant la drosophile. Chaque étape nécessitait une réflexion préalable sur les méthodologies à adopter, puis leur exécution. Cette phase a donc demandé davantage de temps et d'investissement.

La charge de travail s'est progressivement alourdie avec l'arrivée de nouveaux projets et la préparation des examens, rendant difficile de consacrer du temps au projet en dehors des créneaux hebdomadaires prévus.

Au début du mois de mai, la période d'exams et rendus des projets n'a pas permis d'y consacrer du temps. Après cette période, la rédaction du rapport a été commencé en parallèle des dernières manipulations bioinformatiques nécessaires pour conclure les analyses.

Répartition des tâches

Pour la répartition des tâches, la répartition du travail en deux parties distinctes n'a pas été possible étant donné que la plupart des étapes étaient dépendantes entre elles, où chacune nécessitait les résultats de la manipulation précédente. Cette contrainte n'a pas rendu possible une répartition du travail indépendamment des parties.

Au vu de la complexité de l'analyse Singlecell Multiome, il a été décidé de réaliser l'analyse en parallèle l'une de l'autre.

Cette approche a offert plusieurs avantages notamment pour vérifier la cohérence des résultats. Si des résultats différents apparaissent, il était plus facile de caractériser l'erreur commise et donc de la corriger. Cela a aussi permis d'avoir une vision globale et multimodale du projet, de ce fait la charge de travail a été équitablement répartie.

Au vu du déroulé du projet, cette approche a été plus que bénéfique pour développer des compétences telles que la capacité d'apprentissage, la communication en équipe, la sécurisation des données et le travail de recherche fondamentale que représentent les recherches dans la littérature en autonomie.

De manière ponctuelle (vacances scolaires, maladie), il est arrivé de ne pas pouvoir nous rendre toutes les deux simultanément au Centre de bioinformatique (CBI) au rendez-vous hebdomadaire. Dans de telles situations, des « réunions » à deux ont été organisées afin d'expliquer à l'autre ce qui a été fait, pour garantir que chacune maîtrise l'intégralité des tâches du projet, ainsi que les nouvelles commandes R implémentées.

L'IA générative a pu être utilisée pour compléter l'aide au débogage de Stack Overflow (<https://stackoverflow.com>).

Une copie du script R de l'analyse *in silico* a été sauvegardée régulièrement sur nos ordinateurs respectifs, ainsi qu'une copie sur une clé USB. Un GitLab privé a également été créé fin mars. Cela a permis de faciliter l'utilisation, le partage et la récupération du code en cas de perte de celui-ci, et d'augmenter la sécurité en cas de vol de données, en plus d'utiliser le serveur Genotoul.

- [1] Chai H, Tjong H, Li P, Liao W, Wang P, Wong CH, Ngan CY, Leonard WJ, Wei CL, Ruan Y. ChiATAC is an efficient strategy for multi-omics mapping of 3D epigenomes from low-cell inputs. *Nat Commun.* 2023 Jan 13;14(1):213. doi: 10.1038/s41467-023-35879-5. PMID: 36639381; PMCID: PMC9839710.
- [2] Vogelmann,J., Valeri,A., Guillou,E., Cuvier, O., et Nollmann,M., (2011) Roles of chromatin insulator proteins in higher-order chromatin organization and transcription regulation, *Nucleus*, 2:5, 358-369, DOI: 10.4161/nucl.2.5.17860
- [3] Luzhin AV, Flyamer IM, Khrameeva EE, Ulianov SV, Razin SV, Gavrilov AA. Quantitative differences in TAD border strength underly the TAD hierarchy in Drosophila chromosomes. *J Cell Biochem.* 2019 Mar;120(3):4494-4503. doi: 10.1002/jcb.27737. Epub 2018 Sep 27. PMID: 30260021.
- [4] Chathoth KT, Mikheeva LA, Crevel G, Wolfe JC, Hunter I, Beckett-Doyle S, Cotterill S, Dai H, Harrison A, Zabet NR. The role of insulators and transcription in 3D chromatin organization of flies. *Genome Res.* 2022 Apr;32(4):682-698. doi: 10.1101/gr.275809.121. Epub 2022 Mar 30. PMID: 35354608; PMCID: PMC8997359.



ANNEXE

INSULATOR-MEDIATED 3D
CHROMATIN LOOPING:
NEW CO-FACTORS, NEW
ROLES, NEW METHODS

FIGURES SCRIPT



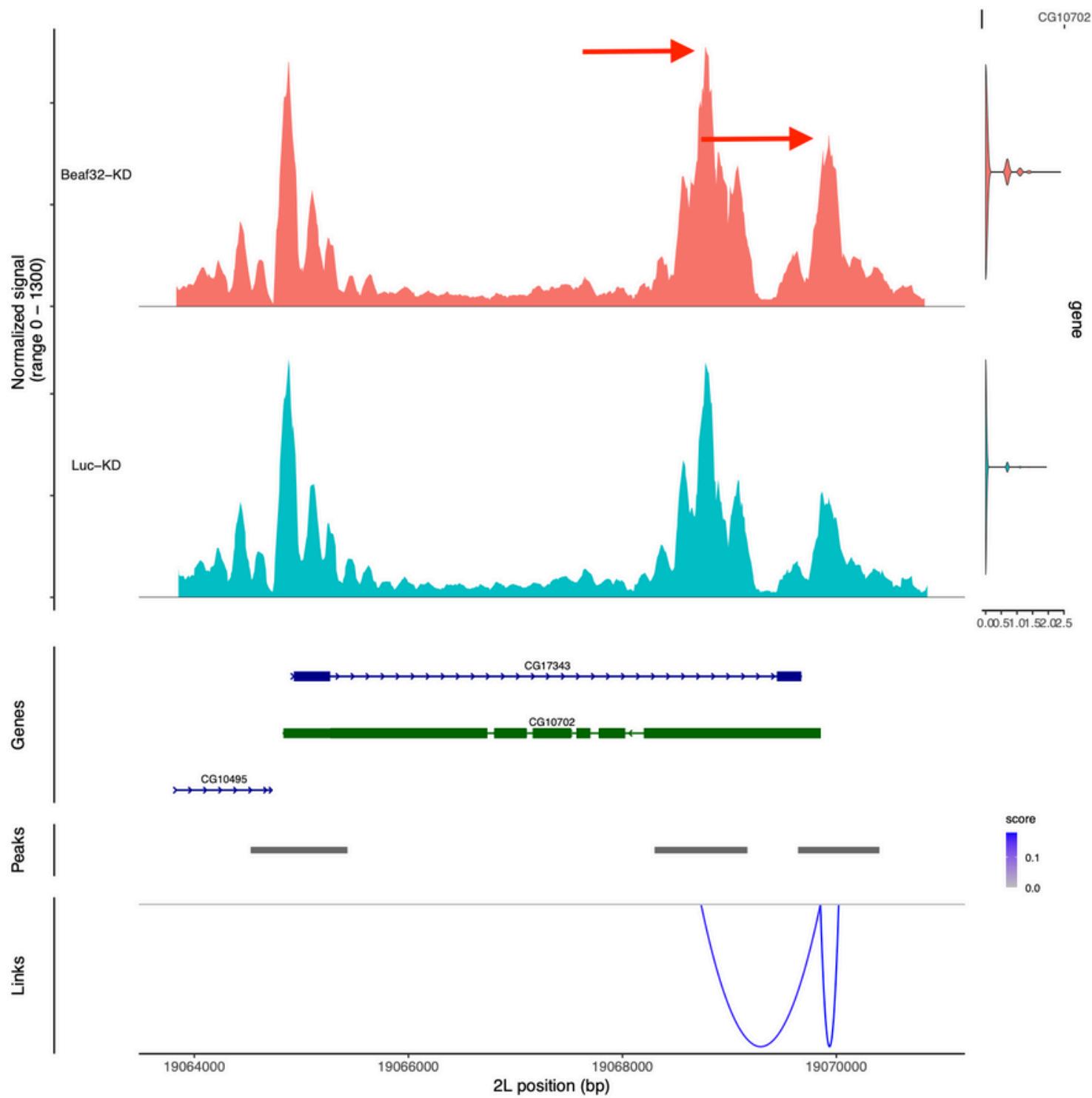


Figure 14 - Cartographie de la couverture et de l'expression du gène *CG10702* en fonction des conditions, avec extension de 1000 paires de bases en amont et en aval

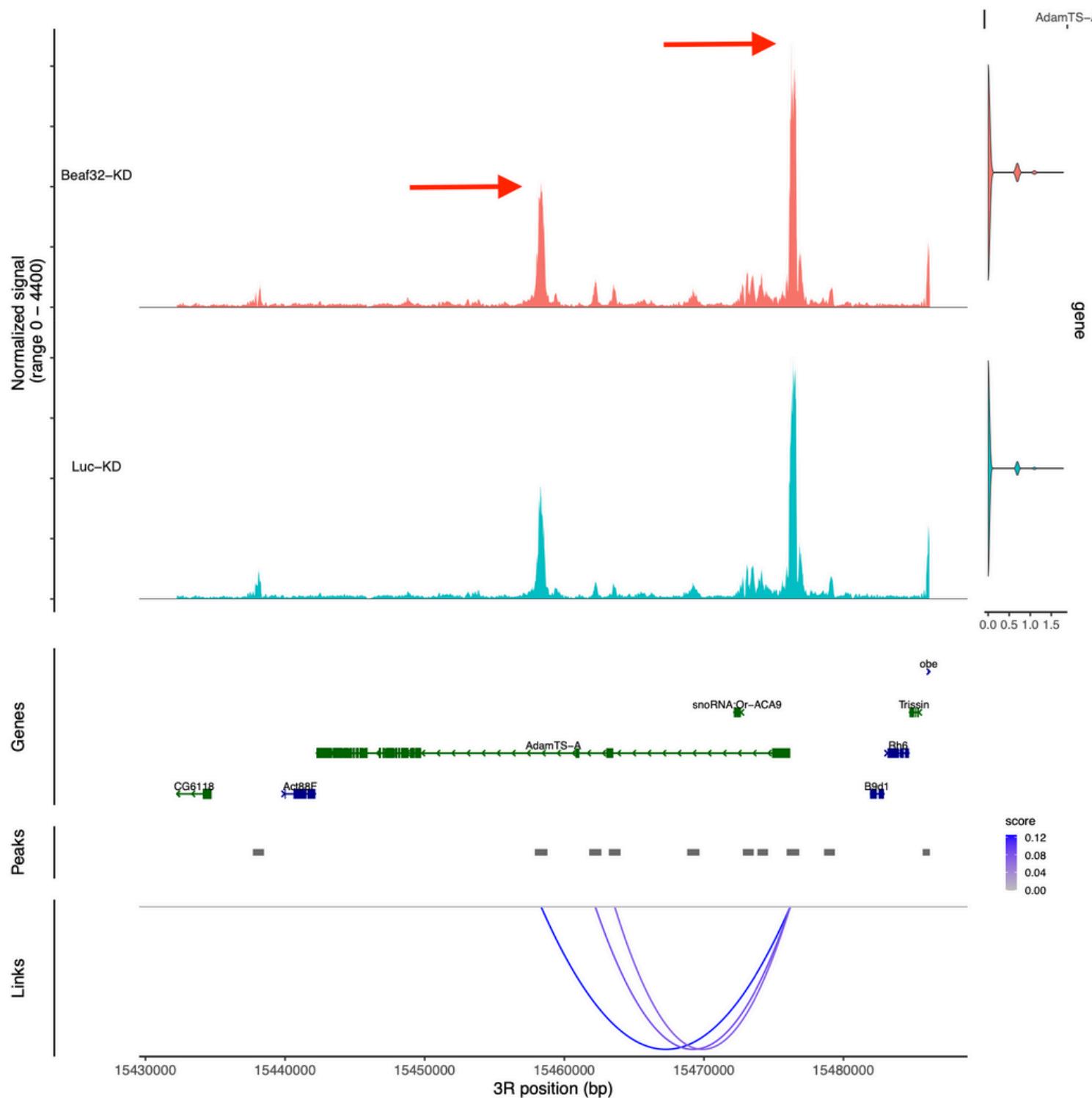


Figure 15 - Cartographie de la couverture et de l'expression du gène AdamTS-A en fonction des conditions, avec extension de 10000 paires de bases en amont et en aval

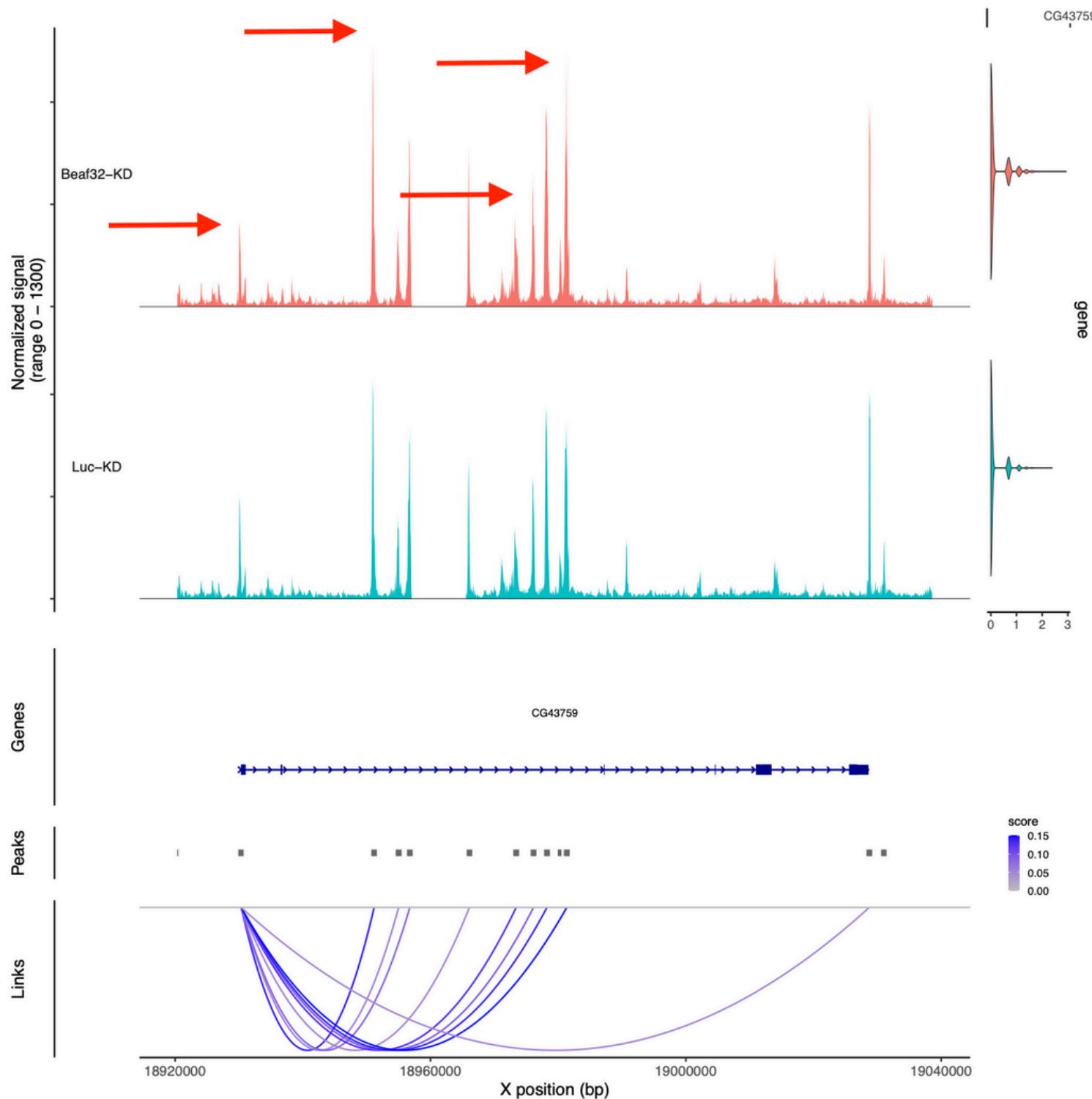


Figure 16 - Cartographie de la couverture et de l'expression du gène CG43759 en fonction des conditions, avec extension de 10000 paires de bases en amont et en aval

script

```
[ ]: # Script PTUT scMultome

#####
# Chargement des librairies
#####
library(Seurat)
library(Signac)
library("TxDb.Dmelanogaster.UCSC.dm6.ensGene")
library(ggplot2)
library(cowplot)
library(magrittr)
library(tidyverse)
library(rtracklayer)

#####
# dataset from 10x import in R
#####
fragpath <- "/work/user/ocuvier/scMultiome/Beaf32/outs/atac_fragments.tsv.gz"
datascM <- Read10X(data.dir = "/work/user/ocuvier/scMultiome/Beaf32/outs/
  ↪filtered_feature_bc_matrix/")
class(datascM)
head(names(datascM))
class(datascM[[1]])
head(colnames(datascM[[1]]))
all(colnames(datascM[[1]]) == colnames(datascM[[2]]))
head(rownames(datascM[[1]]))
head(rownames(datascM[[2]]))
## Representing the data in dgMatrix or similar classes that can contain
  ↪sparse
## data saves a lot of memory space. In these classes, 0 values become dots "."
object.size(as.matrix(datascM[[1]]))
# 5471023832 bytes
object.size(datascM[[1]])
# 670217944 bytes
## get gene annotation
annotation <- import(paste0("/work/user/ocuvier/scMultiome/genes/genes.gtf.
  ↪gz"))
```

```

class(annotation)
## Création d'un objet Seurat contenant les données d'expression génique (ARN)
scM <- CreateSeuratObject(
  counts = data$`Gene Expression`,
  assay = "RNA"
)
## Création de l'assay ATAC et ajout à l'objet Seurat
scM[["ATAC"]] <- CreateChromatinAssay(
  counts = data$Peaks,
  sep = c(":", "-"),
  fragments = fragpath,
  annotation = annotation
)
## Checking for 38373 cell barcodes
str(scM)

#####
# Quality Control
#####

# Pourcentage de molécules d'ARN mitochondrial
#-----#
DefaultAssay(scM) ="RNA" # Définition de l'assay par défaut comme étant "RNA"
## Identification des gènes mitochondriaux:
mito.genes = grep(pattern = "mt:",
  x = rownames(x = scM@assays$RNA@data), value = TRUE)
## Calcul du pourcentage de gènes mitochondriaux par cellule
percent.mito <- Matrix::colSums(scM@assays$RNA@counts[mito.genes, ])/
  Matrix::colSums(scM@assays$RNA@counts)
scM <- AddMetaData(object = scM, metadata = percent.mito, col.name = "percent.
  mt") # Ajout des métadonnées au Seurat objet
## Singlecell[["percent.mt"]] <- PercentageFeatureSet(object = Singlecell,
  pattern = "mt:")
## Singlecell[["percent.ribo"]] <- PercentageFeatureSet(object = Singlecell,
  pattern = "rRNA:")
# Trouver le seuil
library(ggExtra) # Chargement de la bibliothèque ggExtra pour une
  fonctionnalité supplémentaire
## Calcul du seuil maximum pour le pourcentage de gènes mitochondriaux
max.mito.thr <- median(Singlecell$percent.mt) + 3*mad(Singlecell$percent.mt)
## Médiane plus trois fois l'écart absolu médian (MAD) du pourcentage de gènes
  mitochondriaux
max.mito.thr #68.9512
## For mitochondrial percentages the lower the value, the better it is,
## so no lower bound threshold to calculate
## min.mito.thr <- median(Singlecell$percent.mt) - 3*mad(Singlecell$percent.mt)

```

```

## Calcul du quantile à 95% du pourcentage de gènes mitochondriaux:
thr_quantiles = quantile(Singlecell$percent.mt,0.5) # On prend la mediane et ↵
    ↵pas le 3 eme quartile
## certes il y a un risque de perdre de l'information mais on veut etre sure ↵
    ↵d'eliminer les genes
## mitochondriaux
## Ce quantile est utilisé comme seuil pour filtrer les valeurs extrêmes
thr_quantiles
###0.3277228
## Seuil definit à 0.3

# TSSEnrichment ↵
    ↵-----
##On regarde les sites TSS = des emplacements spécifiques sur un brin d'ADN où ↵
    ↵la transcription
##(le processus de copie de l'ADN en ARN) commence. Les sites de début de ↵
    ↵transcription sont cruciaux
##dans l'expression génique, car ils marquent le début des instructions pour ↵
    ↵créer une protéine spécifique
##ou une molécule d'ARN.
## 2 métriques majeures pour les données ATAC :
DefaultAssay(scM) ="ATAC" # Définition de l'assay par défaut comme étant "ATAC"
scM <- TSSEnrichment(object = scM, fast = TRUE) # Enrichissement du signal de ↵
    ↵TSSE (Transcription Start Site)
saveRDS(scM,"/home/ocuvier/work/conda/eData/save_data/scM_preprocessed.RDATA")
scM = readRDS("/home/ocuvier/work/conda/eData/save_data/scM_preprocessed.RDATA")
#dir.create("results")
fileout_scatter="/home/ocuvier/work/conda/eData/figures/figure1.pdf"
pdf(fileout_scatter)
## graphique de dispersion de densité avec les données ATAC-seq
DensityScatter(scM, x = "nCount_ATAC", y = "TSS.enrichment", log_x = TRUE, ↵
    ↵quantiles = TRUE)
dev.off()
# Le seuil est definit à 1
# Ajouter une colonne "high.tss" à scM en fonction de la valeur de TSS.
    ↵enrichment
scM$high.tss <- ifelse(scM$TSS.enrichment > 1 , 'High', 'Low')

# NucleosomeSignal ↵
    ↵-----
scM <- NucleosomeSignal(object = scM) # Calcul du signal de nucléosome
## Ajouter une colonne "nucleosome_group" à scM en fonction de la valeur de ↵
    ↵nucleosome_signal:
summary(scM$nucleosome_signal)
quantile(scM$nucleosome_signal, probs = 0.95)
## On definit le seuil à 1.3

```

```

names(scM$nucleosome_signal)
scM$nucleosome_group <- ifelse(scM$nucleosome_signal > 1.3, 'NS > 1.3', 'NS < 1.
  ↵3')
pdf("/home/ocuvier/work/conda/eData/figures/Figure2.pdf")
## histogramme des fragments en fonction du groupement par 'nucleosome_group' et de la région spécifiée "2L-1-20000"
FragmentHistogram(object = scM, group.by =
  ↵'nucleosome_group',region="2L-1-20000")
dev.off()

# Doublets
←
### Scrublet SUR PYTHON ####
import scrublet as scr
import scipy.io
import numpy as np
import os
## Répertoire contenant les données d'entrée
input_dir = "/work/user/ocuvier/scMultiome/Beaf32/outs/"
## Lecture de la matrice de comptage filtrée
counts_matrix = scipy.io.mmread(input_dir + "filtered_feature_bc_matrix/matrix.
  ↵mtx.gz").T.tocsc()
## Initialisation de l'objet Scrublet
scrub = scr.Scrublet(counts_matrix,
                      expected_doublet_rate=0.1,
                      sim_doublet_ratio=2,
                      n_neighbors = 8)
## Exécution du pipeline par défaut
doublet_scores, predicted_doublets = scrub.scrub_doublets(min_counts=1,
  min_cells=3,
  ↵min_gene_variability_pctl=85,
  n_prin_comps=25)

##Preprocessing...
##Simulating doublets...
##Embedding transcriptomes using PCA...
##Calculating doublet scores...
##Automatically set threshold at doublet score = 0.35
##Detected doublet rate = 4.7%
##Estimated detectable doublet fraction = 13.9%
##Overall doublet rate:
##      Expected    = 10.0%
##      Estimated   = 34.2%
##Elapsed time: 44.5 seconds
## exporter les resultat en .csv

```

```

df = pd.DataFrame({'DoubletScore': doublet_scores, 'PredictedDoublet':predicted_doublets})
output_file = "/work/user/ocuvier/scMultiome/Beaf32/Doublet_Scores.csv"
df.to_csv(output_file, index=False)
print("Les résultats du Scrublet ont été enregistrés dans le fichier CSV:", output_file)
##Les résultats du Scrublet ont été enregistrés dans le fichier CSV: /work/user/ocuvier/scMultiome/Beaf32/Doublet_Scores.csv

### SUR R ####
py = read.csv("/work/user/ocuvier/scMultiome/Beaf32/Doublet_Scores.csv")
scM@meta.data$DoubletScore = py$DoubletScore
library(ggplot2)
## Création d'un histogramme des scores de doublet
pdf("/home/ocuvier/work/conda/eData/figures/figure3.pdf")
ggplot(scM@meta.data, aes(x = DoubletScore, after_stat(ndensity))) +
  geom_histogram(bins = 200, colour ="lightgrey") + # Ajout de l'histogramme
  geom_vline(xintercept = 0.35, colour = "red", linetype = 2) + # Ligne verticale rouge à 0.35
  geom_vline(xintercept = 0.3, colour = "green", linetype = 2) # Ligne verticale verte à 0.3
dev.off()
##Pour détecter le nombre de doublet, je laisse le seuil à 0.35 (j'ai également essayé 0.3 mais ça donne la même chose)
scM@meta.data$Predicted_doublets = ifelse(py$DoubletScore > 0.3, "Doublet", "Singlet")
## Affichage du tableau de distribution des doublets prédis et des singlets
table(scM@meta.data$Predicted_doublets)

# Visualisation
-----
library(cowplot) # Pour la personnalisation avancée des graphiques
library(Seurat)
## Calcul du pourcentage de gènes mitochondrial par cellule
scM$percent.mt = PercentageFeatureSet(scM, pattern = "^\mt:")
## Sélection des gènes ribosomiques
ribo.genes <- grep(pattern = "rRNA", x = rownames(x = scM@assays$RNA@data), value = TRUE)
pdf("/home/ocuvier/work/conda/eData/figures/figure4.pdf")
## Création d'un graphique de violon pour plusieurs caractéristiques
VlnPlot(scM, features = c("nFeature_RNA", "nCount_RNA", "percent.mt", "nFeature_ATAC", "nCount_ATAC"),
group.by = "Predicted_doublets", # Groupement par les doublets prédis
ncol = 3)
dev.off()
## Création du premier graphique de dispersion de fonctionnalités

```

```

plot1 <- FeatureScatter(scM, feature1 = "nCount_RNA", feature2 = "percent.mt")
## Création du deuxième graphique de dispersion de fonctionnalités
plot2 <- FeatureScatter(scM, feature1 = "nCount_RNA", feature2 = "nFeature_RNA")
pdf("/home/ocuvier/work/conda/eData/figures/figure5.pdf",height = 8,width = 14)
## Fusion des deux graphiques dans un seul
plot1 + plot2
dev.off()
saveRDS(scM,"/home/ocuvier/work/conda/eData/save_data/Singlecell_doublet.RDATA")
Singlecell = readRDS("/home/ocuvier/work/conda/eData/save_data/
↪Singlecell_doublet.RDATA")

# Assignment des Samples
-----

## Assignment des noms d'échantillons aux cellules en fonction des noms de
↪colonnes
Singlecell$sample[which(grepl(colnames(Singlecell),pattern="-1"))] = "Luc-KD_1"
Singlecell$sample[which(grepl(colnames(Singlecell),pattern="-2"))] = "Luc-KD_2"
Singlecell$sample[which(grepl(colnames(Singlecell),pattern="-3"))] =
↪"Beaf32-KD_1"
Singlecell$sample[which(grepl(colnames(Singlecell),pattern="-4"))] =
↪"Beaf32-KD_2"
## Affichage du tableau de distribution des échantillons pour lesquels le
↪pourcentage de gènes mitochondriaux est inférieur à 30
table(Singlecell$sample[which(Singlecell$percent.mt<30)])
##Beaf32-KD_1 Beaf32-KD_2 Luc-KD_1 Luc-KD_2
## 10313 9054 9521 9485

# Pré-filtration des cellules
-----

# RNA
## Calcul du nombre de cellules exprimant chaque gène
counts_RNA <- Singlecell@assays$RNA@counts # Accès aux comptages de gènes pour
↪l'assay RNA
num.cells <- rowSums(counts_RNA > 0) # Calcul du nombre de cellules exprimant
↪chaque gène pour l'assay RNA
## Cette ligne nous permet d'abord avec rowSums de faire la somme de chaque
↪ligne, puis on sélectionne les lignes >0. Celle étant = 0 signifiant que le
↪gène est exprimé dans aucune cellule.
## "rowSums" calcule la somme de chaque ligne, indiquant le nombre de cellules
↪exprimant chaque gène
## Affichage du nombre total de gènes
length(num.cells) #17807
## Affichage du nombre de gènes exprimés dans plus de 100 cellules

```

```

length(num.cells[num.cells>100]) #8038 Cette ligne nous permet de sélectionner
  ↵parmis les num.cells, celle où il y a une expression dans 100 cellules.□
  ↵Parce que si c'est que quelques cellules qui l'exprime c'est juste du bruit
  ↵et peu significatif.
## Sélection des gènes exprimés dans moins de 10 cellules et mise à jour des
  ↵données
Singlecell@assays$RNA@data = Singlecell@assays$RNA@data[names(num.
  ↵cells>10)],]
Singlecell@assays$RNA@counts = Singlecell@assays$RNA@counts [names(num.cells[num.
  ↵cells>10]),]
#> length(Singlecell@assays$RNA@counts)
#[1] 401304834
#> length(Singlecell@assays$RNA@data)
#[1] 401304834

# ATAC
counts_ATAC <- Singlecell@assays$ATAC@counts
# Calcul du nombre de cellules exprimant chaque pic ATAC
num.cellsATAC = rowSums(counts_ATAC>0)
# "rowSums" calcule la somme de chaque ligne, indiquant le nombre de cellules
  ↵exprimant chaque pic ATAC
# Affichage du nombre total de pics ATAC
length(num.cellsATAC) #14727
# Affichage du nombre de pics ATAC exprimés dans plus de 100 cellules
length(num.cellsATAC[num.cellsATAC>100]) #14727
# Sélection des pics ATAC exprimés dans moins de 3 cellules et mise à jour des
  ↵données
Singlecell@assays$ATAC@data = Singlecell@assays$ATAC@data[names(num.
  ↵cellsATAC[num.cellsATAC>3]),]
# Sélection des pics ATAC exprimés dans moins de 10 cellules et mise à jour des
  ↵comptages
Singlecell@assays$ATAC@counts = Singlecell@assays$ATAC@counts [names(num.
  ↵cellsATAC[num.cellsATAC>10]),]
#> length(Singlecell@assays$ATAC@data)
#[1] 565119171
#> length(Singlecell@assays$ATAC@counts)
#[1] 565119171

# VISUALISATION percent mito par rapport au seuil de 0.3
library(ggplot2)
## Création du graphique de dispersion des caractéristiques ARN avec le
  ↵pourcentage de gènes mitochondriaux
p1 <- ggplot(Singlecell@meta.data, aes(x=nFeature_RNA, y=percent.mt)) +
  geom_point() +
  geom_hline(aes(yintercept = thr_quantiles[1]), colour = "red", linetype =
  ↵2) +

```

```

# geom_hline(aes(yintercept = thr_quantiles[1]), colour = "red", linetype=2) +
  annotate(geom = "text",
    label = paste0(as.numeric(
      table(Singlecell$percent.mt > thr_quantiles[1])[2]),
      " cells removed\n",
      as.numeric(
        table(Singlecell$percent.mt > thr_quantiles[1])[1]),
      " cells remain"), x = 6000, y = 0.1)

pdf("/home/ocuvier/work/conda/eData/figures/figure6.pdf")
p1
dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figure7.pdf")
## Ajoute un histogramme marginal à p1
ggMarginal(p1, type = "histogram", fill="lightgrey", bins=100)
dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figure8.pdf")
## graphique de dispersion de densité
DensityScatter(Singlecell,x = "nFeature_RNA", y= "percent.mt", quantiles = TRUE,
log_x=TRUE)
dev.off()
saveRDS(Singlecell,"/home/ocuvier/work/conda/eData/save_data/
Singlecell_percentmito.RDATA")
Singlecell = readRDS("/home/ocuvier/work/conda/eData/save_data/
Singlecell_percentmito.RDATA")

# Relation between nUMI and nGene detected
library(ggExtra)
library(cowplot)
## Création du premier graphique de dispersion entre nUMI et nGene détectés
avec une régression linéaire
p1 <- ggplot(Singlecell@meta.data, aes(x=nCount_RNA, y=nFeature_RNA)) +
  geom_point() + geom_smooth(method="lm")
p1 <- ggMarginal(p1, type = "histogram", fill="lightgrey") # Ajout
d'histogrammes marginaux
## Création du deuxième graphique de dispersion entre log(nUMI) et log(nGene)
détectés avec une régression linéaire
p2 <- ggplot(Singlecell@meta.data, aes(x=log10(nCount_RNA),y=log10(nFeature_RNA))) +
  geom_point() + geom_smooth(method="lm")
p2 <- ggMarginal(p2, type = "histogram", fill="lightgrey")
pdf("/home/ocuvier/work/conda/eData/figures/figure9.pdf")
## Assemblage des deux graphiques p1 et p2 dans une grille avec 2 colonnes
plot_grid(plotlist = list(p1,p2), ncol=2, align='h', rel_widths = c(1, 1))
dev.off()
## Set low and high thresholds on the number of detected genes

```

```

min.Genes.thr <- median(log10(Singlecell$nFeature_RNA)) - 3*mad(log10(Singlecell$nFeature_RNA))
max.Genes.thr <- median(log10(Singlecell$nFeature_RNA)) + 3*mad(log10(Singlecell$nFeature_RNA))
## Set high threshold on the number of transcripts
max.nUMI.thr <- median(log10(Singlecell$nCount_RNA)) + 3*mad(log10(Singlecell$nCount_RNA))
## Gene/UMI scatter plot before filtering
## Set low and high thresholds on the number of detected genes
min.Peaks.thr <- median(log10(Singlecell$nFeature_ATAC)) - 3*mad(log10(Singlecell$nFeature_ATAC))
max.Peaks.thr <- median(log10(Singlecell$nFeature_ATAC)) + 3*mad(log10(Singlecell$nFeature_ATAC))
## Set high threshold on the number of transcripts
max.ATAC_frag.thr <- median(log10(Singlecell$nCount_ATAC)) + 3*mad(log10(Singlecell$nCount_ATAC))
## Gene/UMI scatter plot before filtering
p1 <- ggplot(Singlecell@meta.data, aes(x=log10(nCount_RNA), y=log10(nFeature_RNA), col=Predicted_doublets)) +
  geom_point() +
  geom_smooth(method="lm") +
  geom_hline(aes(yintercept = min.Genes.thr), colour = "green", linetype = 2) +
  geom_hline(aes(yintercept = max.Genes.thr), colour = "green", linetype = 2) +
  geom_vline(aes(xintercept = max.nUMI.thr), colour = "red", linetype = 2)
p2 <- ggplot(Singlecell@meta.data, aes(x=log10(nCount_ATAC), y=log10(nFeature_ATAC), col=Predicted_doublets)) +
  geom_point() +
  geom_smooth(method="lm") +
  geom_hline(aes(yintercept = min.Peaks.thr), colour = "green", linetype = 2) +
  geom_hline(aes(yintercept = max.Peaks.thr), colour = "green", linetype = 2) +
  geom_vline(aes(xintercept = max.ATAC_frag.thr), colour = "red", linetype = 2)
pdf("/home/ocuvier/work/conda/eData/figures/figure10.pdf", width = 14, height = 8)
patchwork::wrap_elements(ggMarginal(p1, type = "histogram", fill="lightgrey"))+
  patchwork::wrap_elements(ggMarginal(p2, type = "histogram", fill="lightgrey"))
dev.off()

# filtration des cellules
-----  

## Seuils
nUMI.thrs = quantile(Singlecell$nCount_RNA, c(0.05,0.95))

```

```

nGenes.thrs = quantile(Singlecell$nFeature_RNA,c(0.05,0.95))
nFrags.thrs = quantile(Singlecell$nCount_ATAC,c(0.05,0.95))
nPeaks.thrs = quantile(Singlecell$nFeature_ATAC,c(0.05,0.95))
mito_thr = median(Singlecell$percent.mt)
tss.thr = quantile(Singlecell$TSS.enrichment,0.05)
nucleo.thr = quantile(Singlecell$nucleosome_signal,0.95)
# Filtration
Singlecell = subset(
  x= Singlecell,
  subset = nCount_RNA < nUMI.thrs[2] &
    nCount_RNA > nUMI.thrs[1] &
    nFeature_RNA < nGenes.thrs[2] &
    nFeature_RNA > nGenes.thrs[1] &
    percent.mt < mito_thr[1])
Singlecell=subset(
  x=Singlecell,
  subset = nCount_ATAC < nFrags.thrs[2] &
    nCount_ATAC > nFrags.thrs[1] &
    nFeature_ATAC < nPeaks.thrs[2] &
    nFeature_ATAC > nPeaks.thrs[1] &
    TSS.enrichment > tss.thr[1] &
    nucleosome_signal < 1.3
)
Singlecell = Singlecell[,which(Singlecell$Predicted_doublets=="Singlet")]
saveRDS(Singlecell,"/home/ocuvier/work/conda/eData/save_data/
  ↵Singlecell_captured.RDATA")
Singlecell = readRDS("/home/ocuvier/work/conda/eData/save_data/
  ↵Singlecell_captured.RDATA")

#####
# Reduction de dimensionnalité
#####

# Reduction de dimensionnalité Lineaire □
-----  

# RNA - PCA
DefaultAssay(Singlecell) = "RNA"
## Normalisation
Singlecell = SCTransform(Singlecell,vars.to.regress =
  ↵c("nCount_RNA","nFeature_RNA", "percent.mt", "percent.ribo"))
names(Singlecell)
## PCA
Singlecell = RunPCA(Singlecell)
names(Singlecell@reductions)
pdf("/home/ocuvier/work/conda/eData/figures/figure11.pdf")
ElbowPlot(Singlecell,ndims = 50,reduction = "pca")

```

```

dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figure12.pdf")
DepthCor(Singlecell,reduction = "pca",n = 20)
dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figure13.pdf", width = 15, height = 8)
VizDimLoadings(Singlecell,dims = 1:2)
dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figure14.pdf")
DimPlot(Singlecell, reduction = "pca") + NoLegend()
dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figure15.pdf")
DimHeatmap(Singlecell, dims = 1, cells = 500, balanced = TRUE,reduction =
  "pca",assays = "SCT")
dev.off()

# ATAC - LSI
DefaultAssay(Singlecell) = "ATAC"
Singlecell <- FindTopFeatures(Singlecell, min.cutoff = 5)
Singlecell <- RunTFIDF(Singlecell)
# If SVD throws an error:
# Error in irlba(A = t(x = object), nv = n, work = irlba.work, tol = tol) :
#   function 'as_cholmod_sparse' not provided by package 'Matrix'
#   install.packages("remotes")
#   remotes::install_version("Matrix", version = "1.6-1")
#   packageVersion("Matrix")
Singlecell <- RunSVD(Singlecell)
names(Singlecell)
pdf("/home/ocuvier/work/conda/eData/figures/figure16.pdf")
ElbowPlot(Singlecell,ndims = 50,reduction = "lsi")
dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figure17.pdf")
DepthCor(Singlecell,n = 20,reduction = "lsi") #20 definit avec l'elbowplot
dev.off() #on va enlever la premiere composante car ca correspond a qqchose
  ↪ d'anticorrelé a nos données
pdf("/home/ocuvier/work/conda/eData/figures/figure18.pdf", width = 15, height = 8)
VizDimLoadings(Singlecell,reduction = "lsi",dims = 1:2)
dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figure19.pdf")
DimPlot(Singlecell, reduction = "lsi") + NoLegend()
dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figure20.pdf")
DimHeatmap(Singlecell, dims = 1 , cells = 500, balanced = TRUE,reduction =
  "lsi",assays = "ATAC")
dev.off()

```

```

saveRDS(Singlecell,"/home/ocuvier/work/conda/eData/save_data/
↪Singlecell_reduction.RDATA")
Singlecell = readRDS("/home/ocuvier/work/conda/eData/save_data/
↪Singlecell_reduction.RDATA")
## RNA = SCT

# Non-linear dimensionality reduction
-----
```

```

# UMAP
set.seed(12345)
## RNA
DefaultAssay(Singlecell)="SCT"
Singlecell = RunUMAP(Singlecell,dims = 1:20,reduction = "pca",reduction.name =
↪"umap.rna",seed.use = 42)
## ATAC
DefaultAssay(Singlecell)="ATAC"
Singlecell = RunUMAP(Singlecell,dims = 2:20,reduction = "lsi",reduction.name =
↪"umap.atac",seed.use = 42)
## Plot
p1 <- DimPlot(Singlecell, label = TRUE, repel = TRUE,reduction = "umap.
↪rna",dims = 1:2)+NoLegend()+ ggtitle("RNA UMAP")
p2 <- DimPlot(Singlecell, label = TRUE, repel = TRUE,reduction = "umap.
↪atac",dims = 1:2)+NoLegend()+ ggtitle("ATAC UMAP")
pdf("/home/ocuvier/work/conda/eData/figures/figureUMAP_RNA.pdf")
p1
dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figureUMAP_ATAC.pdf")
p2
dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figureUMAP_RNA_ATAC.pdf", width =
↪11, height = 5)
p1 + p2
dev.off()

# TSNE
## RNA
DefaultAssay(Singlecell)="SCT"
Singlecell = RunTSNE(Singlecell,dims = 1:20,reduction = "pca",reduction.name =
↪"TSNE_RNA")
## ATAC
DefaultAssay(Singlecell)="ATAC"
Singlecell = RunTSNE(Singlecell,dims = 2:20,reduction = "lsi",reduction.name =
↪"TSNE_ATAC")
names(Singlecell@reductions)
```

```

## Plot
pdf("/home/ocuvier/work/conda/eData/figures/figure22.pdf", width = 12, height = 8)
DimPlot(Singlecell,reduction = "TSNE_RNA",dims = 1:2)+NoLegend()
dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figure22bis.pdf", width = 12, height = 8)
DimPlot(Singlecell,reduction = "TSNE_ATAC",dims = 1:2)+NoLegend()
dev.off()

# Integration des données RNA & ATAC, combinaison des ensembles de données
##FindMultiModalNeighbors() allows to apply the weighted nearest neighbor (WNN) method to
##compute a joint neighbor graph with both the DNA accessibility and gene expression information.
Singlecell = FindMultiModalNeighbors(Singlecell,
                                      reduction.list = list("pca","lsi"),
                                      dims.list = list(1:20,2:20),
                                      #this will be the column name for this values in metadata
                                      modality.weight.name = "wnn.weight",
                                      weighted.nn.name = "weighted.nn"
                                      )

# Expression des genes & localisation peaks ATAC dans la UMAP
-----


# RNA
DefaultAssay(Singlecell) = "SCT"
Singlecell <- RunUMAP(
  object = Singlecell,
  nn.name = "weighted.nn",
  assay = "RNA",
  verbose = TRUE
)
pdf("/home/ocuvier/work/conda/eData/figures/figure23.pdf")
DimPlot(Singlecell, label = TRUE, repel = TRUE, reduction = "umap") + NoLegend()
dev.off()
Loadings(Singlecell@reductions$pca) %>% as.data.frame() %>% arrange(desc(abs(PC_1))) %>% head()
##ImpL2
##CG15347
##Ldh
##lncRNA:Hsrromega
##Hsp26
##mt:CoII

```

```

featuresRNA <- c('ImpL2', 'CG15347', 'Ldh', 'lncRNA:HsrOmega', 'Hsp26', 'mt:
  ↪CoII')
pdf("/home/ocuvier/work/conda/eData/figures/figureTest.pdf", width = 15, height =
  ↪= 10)
RidgePlot(Singlecell, features = featuresRNA, ncol = 2)
dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figureTest2.pdf", width = 15, ↪
  ↪height = 10)
## Violin plot - Visualize single cell expression distributions in each cluster
VlnPlot(Singlecell, features = featuresRNA)
dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figure24.pdf", width = 15, height = ↪
  ↪10)
FeaturePlot(Singlecell,
             features = featuresRNA,
             reduction = "umap")
dev.off()

# ATAC
DefaultAssay(Singlecell) = "ATAC"
Singlecell <- RunUMAP(
  object = Singlecell,
  nn.name = "weighted.nn",
  assay = "ATAC",
  verbose = TRUE
)
pdf("/home/ocuvier/work/conda/eData/figures/figure22bis.pdf")
DimPlot(Singlecell,label = TRUE, repel = TRUE, reduction = "umap", dims = 1:
  ↪2)+NoLegend()
dev.off()
# To get the ATAC peaks with the most loadings
DefaultAssay(Singlecell) = "ATAC"
Loadings(Singlecell@reductions$lsi) %>% as.data.frame() %>% ↪
  ↪arrange(desc(abs(LSI_2))) %>% head()
#3L-3384985-3385870
#2L-13068132-13068965
#3L-2533101-2534015
#2R-6141175-6141978
#3L-3399318-3400143
#2L-14395932-14396814
featureATAC = ↪
  ↪c("3L-3384985-3385870", "2L-13068132-13068965", "3L-2533101-2534015", "2R-6141175-6141978", "3L-
  ↪")
pdf("/home/ocuvier/work/conda/eData/figures/figureexpressionn_featureATAC.pdf", ↪
  ↪width = 20, height = 15)
FeaturePlot(Singlecell,

```

```

        features = featureATAC,
        reduction = "umap")
dev.off()

#####
# Clustering
#####

# RNA

DefaultAssay(Singlecell) = "SCT"
Singlecell@neighbors$weighted.nn@nn.idx
Seurat::Neighbors(Singlecell)
Singlecell@graphs
##wsnn
##wknn
Singlecell = FindClusters(Singlecell,resolution = 0.8,graph.name = "wsnn")

## On essaye de trouver les marqueurs
Singlecell.RNAmarkers <- FindAllMarkers(object = Singlecell)
head(x = Singlecell.RNAmarkers)

## On cherche le top 10 des genes qui s'expriment le plus dans chaque cluster
## Pour ensuite les passer dans panther afin d'identifier a quels marqueurs
## cellulaires ils sont associés
Singlecell.RNAmarkers%>%top_n(wt=avg_log2FC,n=10)
Singlecell.RNAmarkers%>%group_by(cluster)%>%top_n(wt=avg_log2FC,n=10)
## pour sauvegarder dans un fichier tsv les top markers
##Singlecell.
  ↵RNAmarkers%>%group_by(cluster)%>%top_n(wt=avg_log2FC,n=10)%>%write_tsv("top10RNAmarkers_per
  ↵tsv")
## identifier les marqueurs et identifier quel type de cluster que c'est
### pour le cluster 0
Singlecell.
  ↵RNAmarkers%>%filter(cluster=="0")%>%top_n(wt=avg_log2FC,n=10)%>%pull(gene)
## cDIP, Tapdelta, SPARC, CG18088, CaBP1, CG1092, CG5885, Ama, CG18549, CG33307
### pour le cluster 1
Singlecell.
  ↵RNAmarkers%>%filter(cluster=="1")%>%top_n(wt=avg_log2FC,n=10)%>%pull(gene)
## path, daw, Gp150, ImpL2, Dgp-1, JhI-21, Thor, mnd, CG31324, Nep13
### pour le cluster 9 :
Singlecell.
  ↵RNAmarkers%>%filter(cluster=="9")%>%top_n(wt=avg_log2FC,n=10)%>%pull(gene)
## DnaJ-1 = Hsp
##Hsp26, Hsp23, Hsp27, Hsp70Bc, Hsp68, lncRNA:alphagamma-element:
  ↵CR32865, DnaJ-1, Hsp83, CG6770
## lien panther : https://pantherdb.org/ (autre : davidbioinformatics.nih.gov)

```

```

## lien flybase : https://flybase.org/
## https://www.flyrnai.org/tools/pangea/web/home/7227

## Echec de la mission, les cellules proviennent de structure trop similaires...
↪.

saveRDS(Singlecell.RNAmarkers, "/home/ocuvier/work/conda/eData/save_data/
↪Singlecell.RNAmarkers.RDATA")
Singlecell = readRDS("/home/ocuvier/work/conda/eData/save_data/
↪Singlecell_cluster.RDATA")

# ATAC
## Meme conclusion mais on a tenté quand même
## ATAC chipseqbeaf32 en bulk, on peut allez voir parmis les sites de fication
↪qui sont alterés (de novo, hyper variables )
##on s'attend a avoir des pics atac qui sont présent de novo quand on
##fait une depletion de beaf 32, les tad actif, on tendance a activer ou
↪inactiver les tad voisin
##et ca on peut le voir avec les pics atacs
DefaultAssay(Singlecell) = "ATAC"
Singlecell.ATACmarkers <- FindAllMarkers(object = Singlecell)
head(x = Singlecell.RNAmarkers)
## Echec de la mission, les cellules proviennent de structure trop similaires...
↪.

#####
# Expression Differentielle - BULK
#####

BiocManager::install("BSgenome.Dmelanogaster.UCSC.dm6")
library(GenomeInfoDb)
DefaultAssay(Singlecell) = "ATAC"
dm6_genome <- BSgenome.Dmelanogaster.UCSC.dm6::BSgenome.Dmelanogaster.UCSC.dm6
↪%>% `seqlevelsStyle<-`("ensembl")
Singlecell = RegionStats(Singlecell, genome=dm6_genome)

#install.packages("qlcMatrix")
#library(qlcMatrix)
#install.packages("devtools")
#devtools::install_github("cysouw/qlcMatrix")

# Coordonnées frontiere TAD
↪-----
```

```

library("TxDb.Dmelanogaster.UCSC.dm6.ensGene")
TAD_Regions = readRDS("/work/user/nschickele/PROJET_3D_VARIABILITY_2023/
↪test_package_HicAggR/DATA/dm6/S2/WT/TAD/RDS/tadRamirez_UCSC_dm6_gr.rds")
```

```

seqlevelsStyle(TAD_Regions) = "ensembl" #Pour que la version UCSC soit au
#format Ensembl
# sous la forme uc sc on change ca avec seqlevel au format ensembl pour faire la
#recherche des sites beafs32 au niveau des tads"
head(TAD_Regions)

DefaultAssay(Singlecell) = "ATAC"
#Version Ensembl :
beaf32_peaks_bed = rtracklayer::import.bed("/work/user/nschickele/
ANALYSES_SINGLECELL/DATA/CHIPSEQ/BED/Beaf32_trimmed_filt_sort_summits.bed")
head(beaf32_peaks_bed)

beaf32_peaks_bed = GenomeInfoDb::keepStandardChromosomes(beaf32_peaks_bed,
#species = "Drosophila_melanogaster", "coarse") #Enlève les chromosomes
#annexes, les régions pas sur les chromosomes de 2L 3L 4X 2R 3R Y.

# Je dois Comparer les TAD_Regions aux peaks de beaf32 pour trouver les gènes
#dans un rayon de 5 à 10kbp pour trouver les gènes qui sont en bordure TAD.

#d'abord on resize les regions tad, on les veut chevauchantes. Et on va séparer
#les starts et les ends.
#resize(x, width, fix="start", use.names = TRUE, ignore.strand = FALSE) avec x
#= notre objet GRRange et width la taille.

TAD_Regions_Start = resize(TAD_Regions, 1,fix="start", use.names = TRUE, ignore.
#strand = FALSE)+500
TAD_Regions_End = resize(TAD_Regions, 1,fix="end", use.names = TRUE, ignore.
#strand = FALSE)+500 #ça fait 500 de chaque côté donc 1000 en tout.
#rezise = je veux plus ou moins autour de ce tad
Liste_GR_TAD_Regions = c(TAD_Regions_Start, TAD_Regions_End)
Liste_TAD_Borders= GenomicRanges::reduce(Liste_GR_TAD_Regions)
#finOverlaps, il va trouver les overlaps entre beaf et tad.
#findOverlaps(query, subject) = le query on va garder ces informations et tout,
#donc notre query c'est beaf32
Overlaps_Beaf_TAD = subsetByOverlaps(beaf32_peaks_bed,Liste_TAD_Borders)
Overlaps_TAD_Beaf = subsetByOverlaps(Liste_TAD_Borders,beaf32_peaks_bed)
# pareil mais inverse
DefaultAssay(Singlecell) = "ATAC"
#On établi la liste des gènes
Liste_gene= Annotation(Singlecell)
Liste_gene= Liste_gene[Liste_gene$type == "gene"]
# toute l'annotation du genome (17807 genes qui font partie du genome via
#flybase)
DefaultAssay(Singlecell) = "SCT"
table(Liste_gene$gene_name %in% rownames(Singlecell))
FALSE TRUE

```

```

7791 10016
# 10016 gene au moins exprimé dans une cellule
Gene_Singlecell = Liste_gene[Liste_gene$gene_name %in% rownames(Singlecell)]  

↳#On sélectionne ceux qui sont présent dans au moins une cellule.
#resize les TAD overlapés avec des sites Beaf32 pour chercher les gènes
↳avoisinnants les TAD.
TAD_Beaf_Resize = resize(Overlaps_TAD_Beaf, 1,fix="center", use.names = TRUE,  

↳ignore.strand = FALSE)+5000
# 10000 pb autour des bordures tads
# reduce c'est pour enlever les regions qui s'overlap entre elles (qui se
↳surperposent ) on garde que une deqs deux
TAD_Beaf_Resize = GenomicRanges::reduce(TAD_Beaf_Resize)
#GRanges object with 1436 ranges and 0 metadata columns:
# on recap promo ceux auxquelson s'attend que le pic atac interagie
Promoteur_gene = promoters(Gene_Singlecell,upstream=500, downstream= 500) #On
↳trouve les promoteurs des gènes à peu près
# On cherche là où se recoupe la position des gènes detectés avec les beaf32
↳qui ont été autour de régions TAD : ça nous permettra de voir les gènes
↳autour des régons TAD.
Overlaps_Gene_TAD = subsetByOverlaps(Promoteur_gene,TAD_Beaf_Resize)
head(Overlaps_Gene_TAD$gene_name)
#[1] "DhpD"      "CG14641"   "abs"       "Gel"       "Vps24"     "MP1"

# Peaks to gene frontiere TAD
-----
Singlecell <- LinkPeaks(
  object = Singlecell,
  peak.assay = "ATAC",
  expression.assay = "SCT",
  genes.use = c("DhpD"      , "CG14641"   , "abs"       , "Gel"       , "Vps24"     , "MP1" )
)

feature_bordureTAD = c("DhpD"      , "CG14641"   , "abs"       , "Gel"       , "Vps24"     , 
↳"MP1")

p1 <- CoveragePlot(
  object = Singlecell,
  region = "DhpD",
  features = "DhpD",
  expression.assay = "SCT",
  group.by = "sample",
  extend.upstream = 1000,
  extend.downstream = 2000
)
## + Coverage plot pour tous les genes, ça donne rien ...
## On test autre chose

```

```

# Peaks to gene frontiere TAD - Trier par z score
-----
```

```

LinkPeak_Singlecell_Gene_Present_TAD = readRDS("/home/ocuvier/work/conda/MData/
  ↪save_data/LinkPeak_Singlecell_Gene_Present_TAD.RDATA")
# trier par z score
top_n(as.data.
  ↪frame(Links(LinkPeak_Singlecell_Gene_Present_TAD,assay="ATAC")),n=20,wt=zscore)
DefaultAssay(Singlecell) = "ATAC"
Annotation(Singlecell)$tx_id = Annotation(Singlecell)$transcript_id
# genes : "CG3036", "nAChRalpha6", ↪
  ↪"CG30456", "Drs", "Ids", "CG10863", "ImpL2", "form3", "CG32354"
p1 <- CoveragePlot(
  object = Singlecell,
  region = "CG3036",
  features = "CG3036",
  expression.assay = "SCT",
  group.by = "sample",
  extend.upstream = 5000,
  extend.downstream = 10000
)
## + Coverage plot pour tous les genes, ça donne rien ...

#####
# Verification de la depletion de BEAF32 - BULK
#####

# EST CE QUE la depletion de Beaf32 a bien fonctionné ?
#'BEAF-32'%in%rownames(Singlecell@assays$SCT@data)

pdf("/home/ocuvier/work/conda/eData/figures/figure27.pdf", width = 20, height = ↪
  ↪5)
FeaturePlot(Singlecell,
  features = "BEAF-32",
  split.by = "sample",
  reduction = "umap")
dev.off()

pdf("/home/ocuvier/work/conda/eData/figures/figure28.pdf", width = 10, height = ↪
  ↪5)
RidgePlot(Singlecell, features = "BEAF-32", group.by = "sample", ncol = 1)
dev.off()

pdf("/home/ocuvier/work/conda/eData/figures/figure29.pdf")
```

```

DotPlot(Singlecell, features = "BEAF-32", group.by = "sample") + RotatedAxis()
dev.off()

# LA REPONSE EST OUI
# la deplation de beaf32 a bien marché on voit bien sur la fig 29 que beaf kd1
# et d2
# ne ssont pas exprimé

# remarque :
# le gene même si il est bien exprimé on peut ne pas le detecter
# de base les niveaux d'expression de beaf32 n'est pas en norme comparé aux
# features Singlecell. Après depletion (les transcript sont comme en
# "competition", du au nombre de reads limité; partagé avec les fragments ATAC et
# les transcript (les genes bien exprimé sont facilement séquencé et détecté).
# donc en moyenne le nfeatures qu'on voit estime à env 10000 gene exprimé par
# cellule alors qu'on en détecte que 3 à 4 milles, du au fait que les genes de
# menage soient ultra abondant (c'est pour ça que pour faire ça bien on
# sequence bcp bcp de cellule, ou alors on réduit le nombre de cellule et on
# augmente le nombre de reads')))

#####
# Expression Differentielle - PSEUDOBULK
#####

Singlecell$condition = str_remove_all(Singlecell$sample, pattern = "[1|2]")

# Ordonner avec log2FC décroissant
# -----
# RNA, Identification des gènes différentiellement exprimés
DefaultAssay(Singlecell) = "SCT"
pseudo_rna_differential = FindMarkers(Singlecell, group.by = "condition", ident.
  ↪1="Beaf32-KD", ident.2="Luc-KD")
pseudo_rna_differential_avglog2FC = pseudo_rna_differential %>% dplyr::
  ↪filter(avg_log2FC > 0.3 & p_val_adj < 0.05) %>% dplyr::
  ↪arrange(desc(avg_log2FC))
pseudo_rna_differential_avglog2FC_names = rownames(pseudo_rna_differential %>%
  ↪dplyr::filter(avg_log2FC > 0.3 & p_val_adj < 0.05) %>% dplyr::
  ↪arrange(desc(avg_log2FC)))
head(pseudo_rna_differential_avglog2FC_names)
#[1] "CG7299" "CG16713" "CG10702" "CG32037" "CG42807" "Cyp12a4"
length(unique(pseudo_rna_differential_avglog2FC_names))
# [1] 94

# ATAC, Évaluation de la variabilité du signal ATAC

```

```

DefaultAssay(Singlecell) = "ATAC"
pseudo_atac_differential = FindMarkers(Singlecell, group.by = "condition",
  ↪ident.1="Beaf32-KD", ident.2="Luc-KD")
pseudo_atac_differential_avglog2FC = pseudo_atac_differential %>% dplyr::
  ↪filter(avg_log2FC> 0.3 & p_val_adj <0.05) %>% dplyr::
  ↪arrange(desc(avg_log2FC))
pseudo_atac_differential_avglog2FC_names = rownames(pseudo_atac_differential)
  ↪%>% dplyr::filter(avg_log2FC> 0.3 & p_val_adj <0.05)%>% dplyr::
  ↪arrange(desc(avg_log2FC)))
head(pseudo_atac_differential_avglog2FC_names)
#[1] "2R-14771639-14772572" "X-13713303-13714139" "2L-19069642-19070402"
#[4] "X-8554059-8554940"    "X-4129135-4129967"   "3R-15461777-15462655"
length(unique(pseudo_atac_differential_avglog2FC_names))
# [1] 32

# Voir Tableau :
pseudo_atac_differential %>% dplyr::filter(avg_log2FC> 0.3 & p_val_adj <0.05)
  ↪%>% dplyr::arrange(desc(avg_log2FC))

# seuil plus severe:
# pseudo_rna_differential %>% dplyr::filter(avg_log2FC> 0.3 & p_val_adj <0.05 &
#   ↪pct.1 > 0.2)%>% dplyr::arrange(desc(avg_log2FC))
#on garde pas c'est trop severe

# LinkPeaks
  ↪-----
# on refait linkpeaks avec les genes trouvés precedement

Singlecell <- LinkPeaks(
  object = Singlecell,
  peak.assay = "ATAC",
  expression.assay = "SCT",
  genes.use = pseudo_rna_differential_avglog2FC_names
)
# Testing 94 genes and 14727 peaks
# Évaluation de la corrélation pour examiner la relation entre le signal ATAC
  ↪et l'expression génique
unique(Singlecell@assays$ATAC@links$gene[which(Singlecell@assays$ATAC@links$peak%in%rownames(
  ↪& ↪
  ↪Singlecell@assays$ATAC@links$gene%in%rownames(pseudo_rna_differential_avglog2FC))])
# >
  ↪unique(Singlecell@assays$ATAC@links$gene[which(Singlecell@assays$ATAC@links$peak%in%rownames(
  ↪& ↪
  ↪Singlecell@assays$ATAC@links$gene%in%rownames(pseudo_rna_differential_avglog2FC))])
# [1] "CG10702" "AdamTS-A" "CG43759"

```

```

# "CG10702" "AdamTS-A" "CG43759" = gene différenciels exprimés significatifs avec des pics significatifs

# Visualisation
# [1] "CG10702" "AdamTS-A" "CG43759"

# Expression différentielle par rapport aux cluster trouvés précédemment
DefaultAssay(Singlecell) = "SCT"
featureRNA <- c("CG10702" , "AdamTS-A" , "CG43759")
pdf("/home/ocuvier/work/conda/eData/figures/figureexpressionn_featureRNA.pdf", width = 20, height = 15)
FeaturePlot(Singlecell,
            features = featureRNA,
            reduction = "umap")
dev.off()
# Ca donne rien
featureRNA <- c("CG10702" , "AdamTS-A" , "CG43759")
pdf("/home/ocuvier/work/conda/eData/figures/figuregene_cluster.pdf", width = 15, height = 10)
DotPlot(Singlecell, features = featureRNA) + RotatedAxis()
dev.off()
# "CG43759" semble plus s'exprimer dans le cluster 2

# CoveragePlot - "CG10702" "AdamTS-A" "CG43759"
DefaultAssay(Singlecell) = "ATAC"
p1 <- CoveragePlot(
  object = Singlecell,
  region = "CG10702" ,
  features = "CG10702" ,
  expression.assay = "SCT",
  group.by = "condition",
  extend.upstream = 1000,
  extend.downstream = 1000
)
p2 <- CoveragePlot(
  object = Singlecell,
  region = "AdamTS-A" ,
  features = "AdamTS-A" ,
  expression.assay = "SCT",
  group.by = "condition",
  extend.upstream = 10000,
  extend.downstream = 10000
)
p3 <- CoveragePlot(
  object = Singlecell,
  region = "CG43759" ,

```

```

features = "CG43759" ,
expression.assay = "SCT",
group.by = "condition",
extend.upstream = 10000,
extend.downstream = 10000
)

pdf("/home/ocuvier/work/conda/eData/figures/figurecoveragePlot.pdf", width = 10, height = 30)
patchwork::wrap_plots(p1,p2,p3, ncol = 1)
dev.off()

# Expression plot- "CG10702" "AdamTS-A" "CG43759"

p1bis <- ExpressionPlot(
  object = Singlecell,
  features = "CG10702",
  assay = "SCT",
  group.by = "condition"
)
p2bis <- ExpressionPlot(
  object = Singlecell,
  features = "AdamTS-A",
  assay = "SCT",
  group.by = "condition"
)
p3bis <- ExpressionPlot(
  object = Singlecell,
  features = "CG43759",
  assay = "SCT",
  group.by = "condition"
)

pdf("/home/ocuvier/work/conda/eData/figures/ExpressionPlot.pdf", width = 15, height = 5)
patchwork::wrap_plots(p1bis,p2bis,p3bis, ncol = 3)
dev.off()

# Comptage du nombre de cellule par condition beaf32KD & Luc KD
-----  

# matrice vide
cluster_means <- matrix(nrow = 14, ncol = 2)
for (i in 1:14) {
  cluster = table(Singlecell$condition[Singlecell@meta.data$seurat_clusters == i-1])

```

```

mean_beaf32 = cluster["Beaf32-KD"]
mean_luc = cluster["Luc-KD"]
somme = mean_beaf32+ mean_luc

cluster_means[i, 1] <- paste0(round(mean_beaf32 * 100 / somme), " %")
cluster_means[i, 2] <- paste0(round(mean_luc * 100 / somme), " %")
}

rownames(cluster_means) = c("0", "1" , "2", "3", "4" , "5" , "6" , "7", "8", "9", "10", "11", "12" , "13")
colnames(cluster_means) = c("Beaf32-KD", "Luc-KD")
saveRDS(Singlecell, "/home/ocuvier/work/conda/eData/save_data/
↪Singlecell_dataTest.RDATA")
Singlecell = readRDS("/home/ocuvier/work/conda/eData/save_data/
↪Singlecell_dataTest.RDATA")
save.image("17052024_sc.RDATA")
load("17052024_sc.RDATA")

#####
# Visualisation de la depletion de BEAF32 - PSEUDOBULK
#####

pdf("/home/ocuvier/work/conda/eData/figures/figureUMAP_condition3.pdf", width = 11, height = 5)
FeaturePlot(Singlecell,
            features = "BEAF-32",
            split.by = "condition",
            reduction = "umap")
dev.off()

p3 <- FeaturePlot(Singlecell,
                    features = "BEAF-32",
                    split.by = "condition",
                    reduction = "umap.rna")
p4 <- FeaturePlot(Singlecell,
                    features = "BEAF-32",
                    split.by = "condition",
                    reduction = "umap.atac")

pdf("/home/ocuvier/work/conda/eData/figures/figureUMAP_condition.pdf", width = 11, height = 5)
patchwork:::wrap_plots(p3, ncol = 1)
patchwork:::wrap_plots(p4, ncol = 1)
dev.off()
pdf("/home/ocuvier/work/conda/eData/figures/figureUMAP_condition2.pdf", width = 22, height = 5)

```

```
patchwork::wrap_plots(p3,p4,ncol = 2)
dev.off()

### LIMITE ET CONCLUSION DE NOTRE ANALYSE

#en single cell on capte pas tout le transcriptome, c'est les genes les plus
↪abondants en arn qui vont etre. si ya des interaction uniquement sur
↪certains genes on peut ne pas le voir...
#si on veut detecter des changements assez subtile dans ces conditions . les
↪genes qui vont avoir de la variabilite vont etre peu abondant en arn (car
↪c'est des genes pas forcement actif dans toutes les cellules)

#c'est connue que ca conduit pas enormement de difference d'expression de gene

***** les genes qui vont avoir de la variabilite vont etre peu abondant en arn
↪(car c'est des genes pas forcement actif dans toutes les cellules). en
↪single cell on capte pas tout le transcriptome, c'est les genes les plus
↪abondants en arn qui vont le plus ressortir et apparaitre en tant que
↪marqueur. Il si ya des interaction uniquement sur certains genes peut
↪abondant en arn, on peut ne pas le voir...
***** depletion beaf 32 normalement peut d'effet sur la regulation du
↪transcriptome (ctcf a le meme effet)
***** chez la droso ya d'autres proteine insulatrice telle que m1bp qui a le
↪meme motif que beaf donc potentiel compensation de la depletion beaf32 et
↪c'est pour ca qu'on verrait pas de difference d'expression
```



CHROMATIN DYNAMICS & CELL PROLIFERATION, CNRS
OLIVIER CUVIER

