BIF7104

Bioinformatique et sciences de la santé

Intégration de données transcriptomiques et épigénétiques publiques afin de mieux décrire l'origine et le rôle d'une sous-population de cellules T CD8+ essentielle pour le contrôle d'infections chroniques

Par Salix Boulet et Wanlin Li

Plan de travail soumis à la Dre Virginie Calderon, le 30 avril 2021

Introduction

Les cellules T $CD8^+$ sont centrales à la réponse du système immunitaire adaptatif [1]. Elles reconnaissent un fragment peptidique d'un antigène à l'aide de leur récepteur T (*T cell receptor* ou TCR). Lorsque la cellule T $CD8^+$ reconnait son antigène, le processus d'activation est enclenché. Pour que celui-ci soit optimal, il est classiquement reconnue que trois signaux sont requis : le signal TCR, la co-stimulation fournie par la cellule présentatrice de l'antigène, ainsi que l'inflammation. S'ensuit alors une prolifération massive des cellules T activées afin d'en avoir un nombre suffisant pour contrôler l'agent infectieux, ainsi qu'un processus de différenciation qui permet aux cellules T d'acquérir les caractéristiques nécessaires à l'élimination du pathogène, tel que la cytotoxique et la capacité de migrer aux tissus infectés. Une fois l'agent infectieux éliminé, la grande majorité des cellules T $CD8^+$ meurent par apoptose alors que certaines cellules survivront et se maintiendront afin de former un pool de cellules qui fournira la mémoire immunologique.

Toutefois, dans le contexte d'une infection chronique, tel que le VIH ou l'hépatitce C, ou encore dans les cancers, la persistance de l'antigène et de l'inflammation à long terme a des conséquences sur la différenciation et la fonction des cellules T $CD8^+$ [2]. Ces cellules accumulent en effet à leur surface une série de récepteurs inhibiteurs tel que PD-1, Lag3, 2B4 et CD160 et perdent leur capacité fonctionnelle. On dit de ces cellules qu'elles sont épuisées (*T exhausted* ou T_{EX}). Il est possible de réactiver les cellules épuisées en bloquant les interactions des récepteurs inhibiteurs, tel que PD-1, avec leur ligand, en injectant des anticorps bloquants[2]. En clinique, l'injection d'anticorps anti-PD1 est utilisée contre les mélanomes avec un certain succès[3], même s'il reste

certains défis tel qu'améliorer le taux de réponse et contrer le développement de résistance.

Des études récentes ont démontrés l'hétérogénéité parmi les Tex (voir Fig. 1) [4-11].

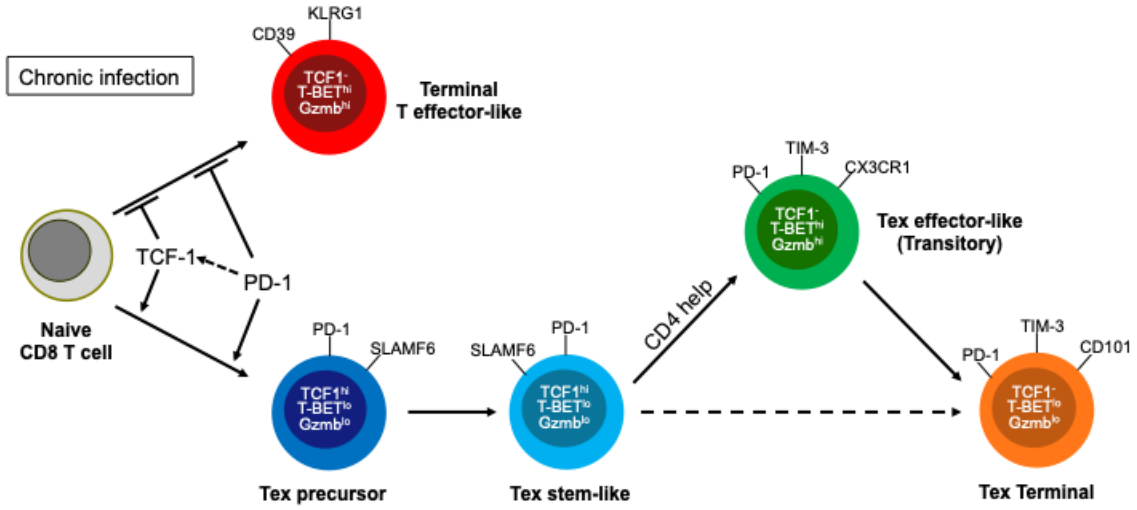


Figure 1. Modèle de différenciation des Tex dans un modèle d'infection chronique. Lors d'une infection chronique, la cellule T CD8+ fait une première différenciation binaire, soit en Tex précurseur ou en T effecteur au profil non-épuisé. Les cellules Tex précurseurs suivront une différenciation de TexP (Tex stem-like) vers TexTrans (Tex effector-like) et TexTerm). Adapté de Chen *et. al.*, Immunity, 2019; Hudson *et. al*., Immunity, 2019 et Zander *et. al*., Immunity, 2019.

Les Tex progéniteurs (TexP) sont moins différenciées et moins épuisées. Elles ont une capacité de d'auto-renouvèlement et sont celles qui répondent au traitement anti-PD1. Elles dépendent du facteur de transcription TCF1 et peuvent être identifiées à l'aide des marqueurs Slamf6, Tim3 et CD101 (Slamf6+Tim3-CD101-). Il a récemment été démontré que les TexP se différencient en cellules Tex transitoires (TexTrans)[8,10,11]. La présence de ces cellules corrèle avec le contrôle d'une infection chronique ou d'une tumeur. Les cellules TexTrans expriment le récepteur de chimiokine CX3CR1 ainsi que Tim3, mais pas CD101 ni Slamf6 (CX3CR1+Tim3+CD101-Slamf6-). Une autre

population de Tex, dites terminalement épuisées (TexTerm) est peu fonctionnelle et a peu de potentiel prolifératif. Celle-ci est Slamf6-CX3CR1-Tim3+CD101+.

L'avancé des technologies et la diminution des couts de séquençages, combinés au fait que les publications scientifiques exigent que les données de séquençage soient déposées et rendues publiques font en sorte la quantité de jeux de donnée de transcriptomique et d'épigénétique a augmenté de façon exponentielle lors des dernières années[12]. La fouille des bases de données comme Gene Expression Omnibus (GEO) et Sequence Reads Archive (SRA) afin de réanalyser ou d'intégrer différentes données peut contribuer à enrichir les analyses déjà publiées ou offrir de nouvelles interprétations.

Puisque la présence de TexP dans la tumeur corrèle avec la réponse aux traitements anti-récepteurs inhibiteurs, plusieurs études se sont attardées à élucider leur biologie[13]. Par contre, bien que certains éléments phénotypiques soient connus, il y a peu d'information cohérente sur les évènements moléculaires qui permettent la différenciation des TexP en TexTrans. Puisque les TexTrans jouent un rôle dans le contrôle d'infection ou de tumeur, il est important d'élucider ces mécanismes. Le profile transcriptomique des populations TexP et TexTrans est disponible, dans un modèle murin d'infection chronique (soit Lymphocytic Choriomenengitis Virus clone 13 ou LCMV cl13) à travers certaines publications[8,10], mais il n'a pas encore été combiné aux évènements dynamiques de régulation de répression de la chromatine ainsi qu'à l'activité des enhancers et promoteurs. En effectuant une recherche pour le terme 'CX3CR1', le marqueur définissant la population TexTrans, dans la base de donnée de GEO, nous avons trouvé des analyses de chromatine non-publiées, mais publiquement disponible pour ces mêmes populations de cellule, aussi dans le modèle LCMV cl13. Afin

d’approfondir notre connaissance sur la transition de TexP à TexTrans dans l’infection chronique, nous allons donc réanalyser et combiner les données d’une étude transcriptomique publiée[10] à celles, non-publiées (#GSE149810), de la modification des marques des histones, de ces deux populations. Des pipelines d’analyse pour RNA-seq et ChIPseq seront appliqués et les informations intégrées. De plus, nous allons comparer nous analyses aux résultats obtenus lors de l’utilisation de pipeline offert sur le web.

Les données

Les données pour l'analyse transcriptomique des populations de T CD8+ épuisées progéniteurs (T_{prog}) et transitoires (T_{trans}) seront téléchargées de la publication du groupe de R. Ahmed, Hudson *et al.*, Immunity, 2019 (PMID 31810882). Les données ont été déposées dans le NCBI Sequence Read Archive sous le BioProject #PRJNA497086. On y trouve 24 différents #SRA, mais nous analyserons les données obtenues au jour 45 post-infection avec LCMV cl13, lorsque la chronicité est bien établie. Nous nous concentrerons sur les cellules TexProg (CD101-Tim3-, n=3, SRR8065107, SRR8065110, SRR8065114) et TexTrans (CD101-Tim3+, n=3, SRR8065108, SRR8065117, SRR8065118). Les données sont le résultat d'un séquençage de fragments de 100 paires de base en *single-end*.

Les données pour l'analyse épigénétique des mêmes populations ont été brièvement rendues disponibles sur GEO (GSE 149810), mais ne le sont plus depuis le 18 mars 2021. L'étude du groupe de W. Cui n'est pas encore publiée. Celle-ci définit les TexProg comme Ly108+ (aussi connu comme Slamf6) et les TexTrans comme CX3CR1+. Elle dispose des échantillons suivants pour les TexProg: IgG (SRR11684812), la marque d'enhancer H3K27ac (SRR11684795, SRR11684796), la marque de répresseion H3K27Me3 (SRR11684801, SRR11684802), la marque associée aux promoteurs H3K4Me3 (SRR11684807, SRR11684808). Les échantillons similaires ont été séquencé pour les TexTrans: IgG (SRR11684811), H3K27ac (SRR11684793, SRR11684794), H3K27Me3 (SRR11684799, SRR11684800), H3K4Me3 (SRR11684805, SRR11684806).

Toutes les données ont été importées avec la commande fastq-dump du module sratoolkit, par exemple : `fastq-dump SRR11684807`

Analyse RNAseq

L'analyse des données de transcriptomiques se fera selon le pipeline suivant (Fig. 2) : contrôle de qualité des séquences (FASTQC) et pré-processing (TRIMMOMATIC) si nécessaire, alignement des séquences sur un génome de référence (STAR), résumer et dénombrer et les transcrits (FeatureCounts) et, finalement, faire une analyse d'expression différentielle (DESEQ). Une analyse biologique plus poussée pourra être ensuite faite à l'aide de logiciel disponible sur le web (ex. gProfiler, GSEA).

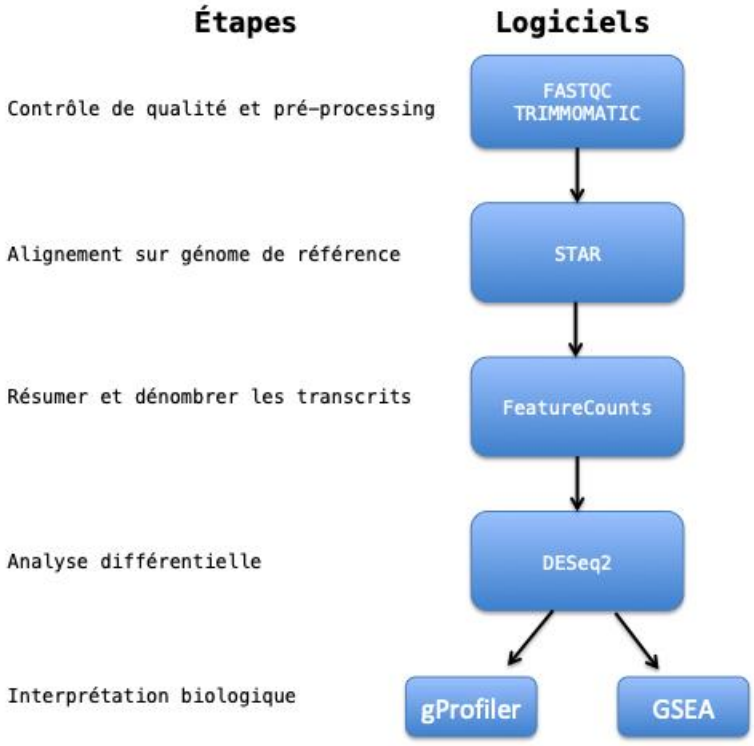


Figure 2. Pipeline d'analyse RNA-seq

Contrôle de qualité et pré-processing

La première étape de l'analyse consiste à vérifier la qualité du séquençage à partir des données brutes, les fichiers FASTQ. Ceci se fait à l'aide du logiciel FASTQC v.

0.11.5[14]. Cette étape est essentielle afin d'éviter d'inclure dans les analyses de transcriptomique des biais issus de la préparation de la libraire ou encore du processus de séquençage. Le logiciel FASTQC prend un sous-ensemble de séquences d'un échantillon afin d'évaluer, en moyenne, différents paramètres qui permettent d'évaluer la qualité de l'échantillon. Ceux-ci incluent la qualité par base, qualité par séquence, % de représentativité des bases, % GC, présence d'adapteurs, etc. Ces analyses se fondent, entre autre, sur les codes de qualité inclus dans le fichier fastq et sont représentée de façon intuitive à l'aide de score ou de pourcentage. Afin de faire un résumé des métriques des différents échantillons en un seul rapport, il est possible de combiner les résultats de FASTQC un utilisant MultiQC (multiqc .)[15]. Un exemple de script FASTQC est disponible en Annexe IA.

Lors de la première analyse de qualité des échantillons de TexP (SRR8065107, SRR8065110, SRR8065114) et de TexTrans (SRR8065108, SRR8065117, SRR8065118), on note que le nombre de séquences pour l'échantillon TexP #3 est moins important (12.5M versus plus de 17.5M, Fig.3A) et que son %GC est légèrement différent comparativement aux autres échantillons (Fig. 3B). Toutefois, la qualité de séquençage est bonne (plus de 28 sur 40) à chaque position (jusqu'à 100pb, Fig. 3C). Par contre, dans les premières 10-15pb des séquences, il y a une irrégularité dans le contenu en nucléotide, ce que peut révéler le séquençage d'adapteurs (Fig. 3D). Pour cette raison, les 17 premiers nucléotides (HEADCROP) ont été enlevés de l'analyse à l'aide de TRIMMOMATIC v. 0.39[16] (voir script Annexe IB). Suite à un deuxième contrôle de qualité avec FASTQC et MultiQC sur les fichier fastq obtenu après TRIMMOMATIC

(Fig. 3D), il est possible de procéder à la prochaine étape puisque les séquences problématiques ont été retirées.

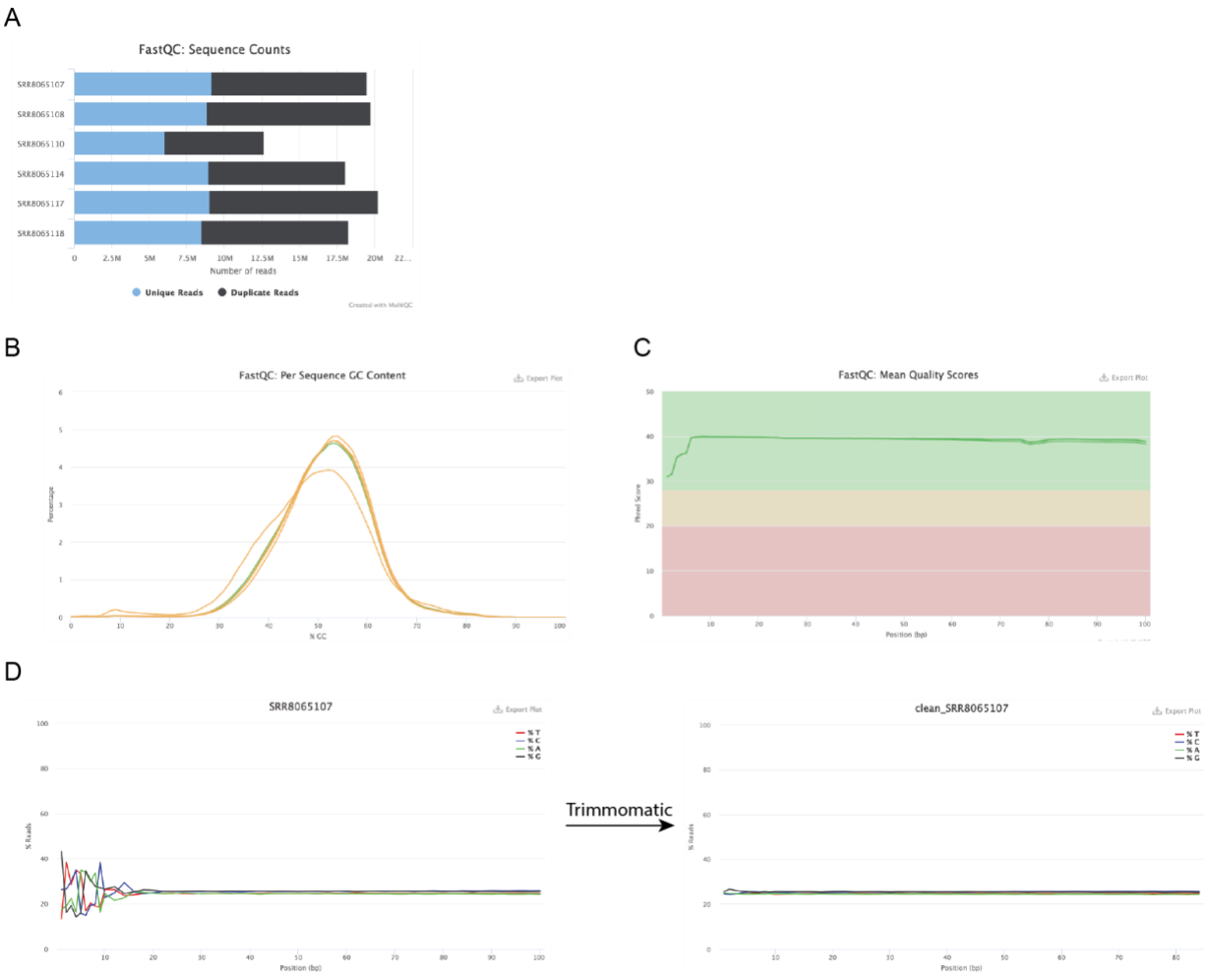


Figure 3. Résultat de contrôle de qualité des données brutes de RNA-seq. (A) Résultat de FastQC, nombre de séquences. (B) Résultat de FastQC, %GC. (C) Résultat de FastQC, score de qualité. (D) Résultat de FastQC, % de contenu en paire de base pré- (panneau de gauche) et post- (panneau de droite) traitement avec Trimmomatic.

Alignement sur génome de référence

Afin de déterminer où les séquences obtenues se situent sur le génome, il est ensuite nécessaire de procéder à l'alignement de celles-ci sur un génome de référence. Plusieurs outils sont disponibles pour cette étape, mais puisqu'il s'agit de données de RNA-seq, le logiciel doit prendre en compte les évènements d'épissage requis pour

générer l'ARN, pour cette raison, nous utilisons STAR v.2.7.3a[17] pour aligner sur le génome murin GRCm38 (mm10). Le logiciel prend en entrée des fichiers fastq et on obtient des fichiers BAM trié par les coordonnées du génome grâce à la commande `outSAMtype BAM SortedByCoordinate`.

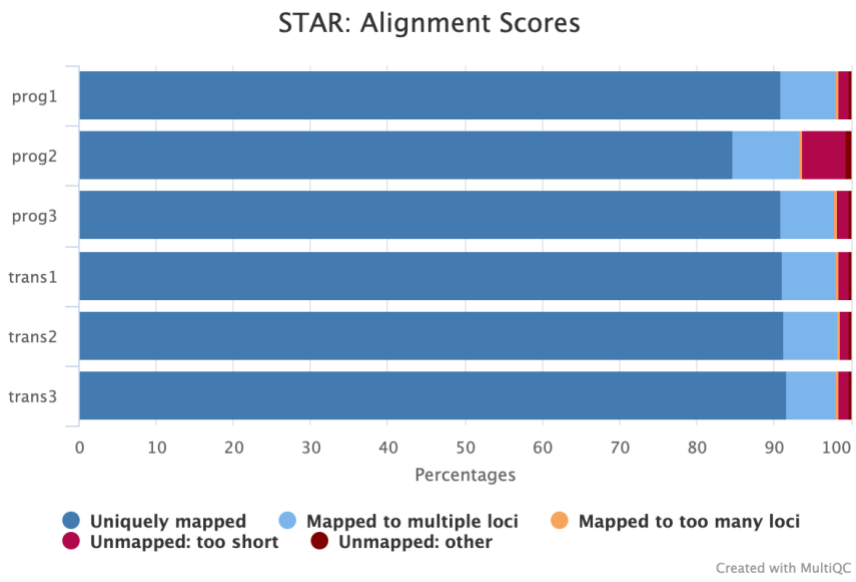


Figure 4. Résumé MultiQC de l'aligement des séquences par STAR.

En utilisant encore une fois MultiQC pour évaluer la qualité de l'alignement, on remarque que plus de 84% des séquences sont alignées de façon unique, écartant la possibilité qu'il y ait une contamination de l'échantillon et renforçant sa qualité (Fig.4).

Résumer et dénombrer les transcrits

La prochaine étape consiste à assigner les séquences alignées à un méta-élément génomique (*feature*) et les dénombrer. Dans le cas d'un RNA-seq, il s'agit d'identifier à quel transcrit sont associés les séquences. FeatureCounts v.1.6.4 permet d'effectuer ces fonctions pour des séquences d'ARN ou d'ADN[18]. En entrée, FeatureCounts prend un

fichier d'alignement des séquences (SAM ou BAM) et retourne, pour chaque échantillon, une matrice des comptes assigné à chaque *feature* du génome/transcriptome (script Annexe I-D). Afin de résumer la proportion ou le nombre de séquences assignées ou non par FeatureCount, il est possible d'encore une fois utiliser la fonction MultiQC (Fig 5).

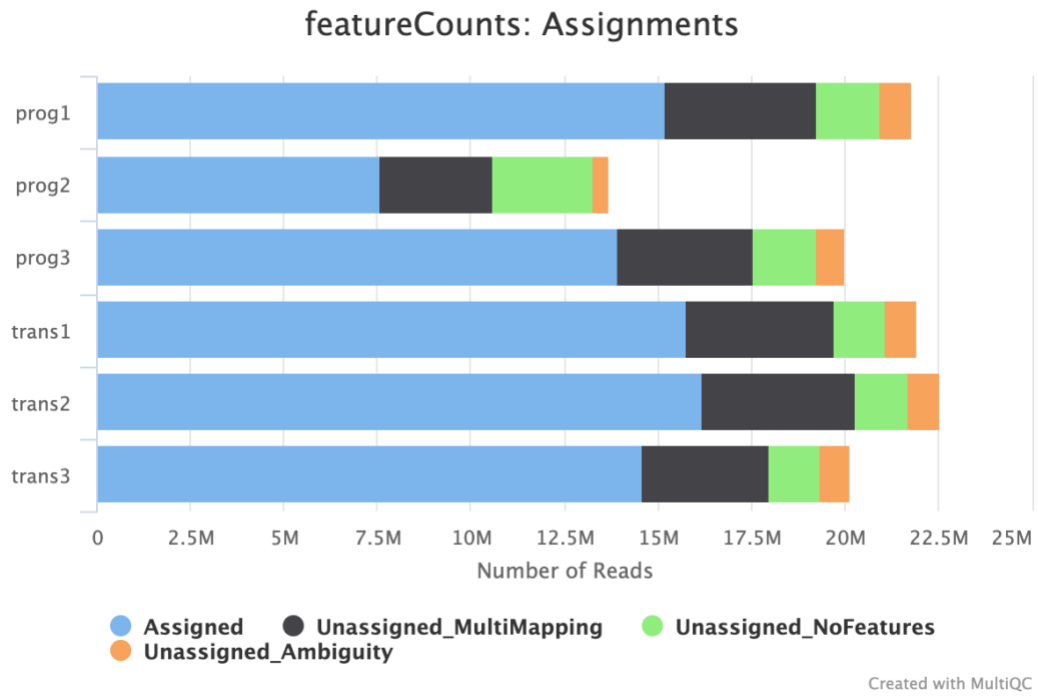


Figure 5. Résumé MultiQC du dénombrement avec featureCounts

Analyse différentielle

Une fois les éléments génomiques comptés, il est nécessaire d'évaluer quels gènes sont différentiellements exprimés dans nos conditions observées. Pour cela, nous utilisons le module DESEQ2, dans R[19]. À partir des matrices de compte, on doit créer un objet DESeqDataSet avec la fonction `DESeqDataSetFromMatrix`. Ensuite, l'expression différentielle entre les cellules TexP et TexTrans est calculée à partir de la fonction `DESeq`, après laquelle on peut obtenir des tableaux avec les valeurs statistiques, les

expressions normalisées et les différences d'expression (log2foldChange) entre les groupes, pour chaque gène. Avant de procéder, il est toutefois nécessaire de vérifier la qualité des données en vérifiant leur distance par dendrogramme et en appliquant une

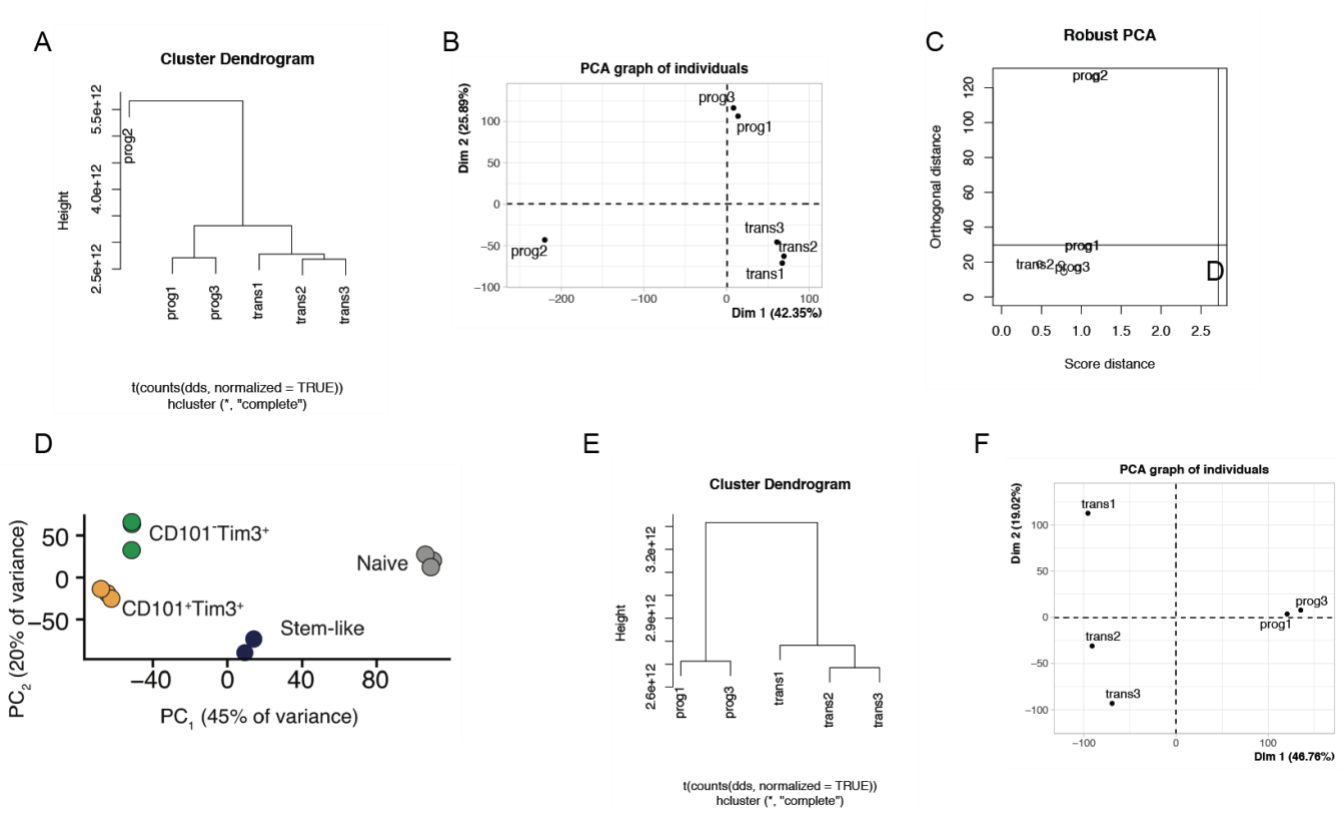


Figure 6. Hiérarchie et clusters des échantillons de TexP et TexTrans suite à l'analyse des données RNAseq. (A) Dendrogramme représentant la distance entre les échantillons. PCA (B) et rPCA (C) des échantillons. (C) Donnée adaptée de la Fig. 3B de la publication de Hudson et. al. d'où les données brutes ont été tirées. Dendrogramme (E) et PCA (F) après exclusion de prog2 (TexP#2, SRR8065110).

PCA afin de mieux observer la dispersion entre les échantillons. À l'aide des fonctions `hcluster` ainsi que `PCA` et `PcaGrid` (Fig. 6A-C), celles-ci appliquées sur des données transformées, l'échantillon TexP#2 (prog2 sur les graphiques) est identifié comme étant aberrant et est exclue de l'analyse différentielle dans une deuxième itération de la fonction DESeq, `dds[,-2]`. Fait à noter, dans l'article d'où ces données ont été tirées, une

donnée TexP à été retirée de l'analyse PCA présentée dans la Fig. 3B (reprise ici dans la Fig. 6D), mais a été gardée pour l'analyse d'expression différentielle[10]. Bien qu'une mention soit faite dans le matériel et méthode, cela est irrégulier et nous préférons poursuivre nos analyses en excluant cet échantillon. Lorsque les fonctions sont ré-appliquées une fois TexP#2 exclu, la similitude entre les échantillons d'un même groupe est satisfaisante (Fig. 6E-F).

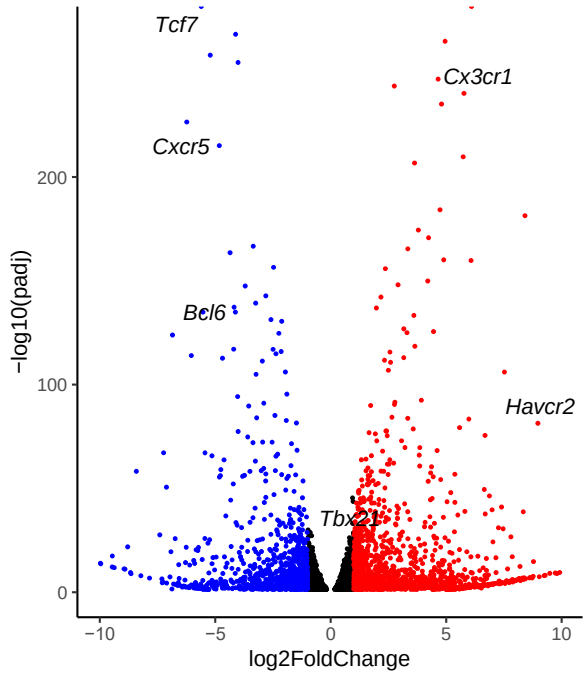


Figure 7. Volcano plot des gènes différentiellement exprimés entre les TexP et les TexTrans. Les gènes plus exprimés dans les TexTrans sont en rouge, ceux dans les TexP, en bleu.

En appliquant les critères de p<0.05 et |log2Foldchange| >1, on obtient que 2371 gènes sont différentiellement exprimés entre les TexP et les TexTrans, 1359 étant plus exprimés dans les TexTrans et 1012 plus dans les TexP (Fig. 7). Le fait que le gène *Cx3cr1* soit plus dans le TexTrans alors que *Slamf6* et *Tcf7* (le gène pour TCF-1) soient plus dans les TexP valide en partie l'analyse. Tel que publié par Hudson *et. al.*, les Bcl6 et Cxcr5 sont plus exprimé dans le groupe TexP alors que Tbx21 (codant pour le facteur de

transcription T-bet) et Havcr2 (codant pour Tim3) sont plus exprimés dans les TexTrans[10].

Le code R de cette section est disponible à l'Annexe I-E.

Interprétation biologique

Afin de faire une interprétation biologique des listes de gènes différentiellement exprimés entre les TexTrans et les TexP, il y a plusieurs logiciels d'analyse. Nous allons utiliser gProfiler et GSEA[20,21]. gProfiler est une plateforme web qui permet l'analyse d'enrichissement de fonction d'une liste de gène à travers la Gene Ontology (GO), les pathways biologique tel que Reactome ou KEGG, ou encore les motifs régulateurs trouvés dans l'ADN (TRANSFAC). gProfiler prend donc en entrée une liste de gènes. GSEA, développé par le Broad Institute est un logiciel qui prend plutôt les résultats d'une expérience de transcriptomique, créé une liste classé en ordre d'expression dans les groupes et vérifie si les gènes appartenant à une liste prédéterminée (*gene set*) apparaissent vers le début ou la fin de ce classement. Les *gene sets* peuvent être issus d'expérience de transciptomique effectuée dans d'autres études ou encore déterminés par des experts ou à travers d'autres bases de données. Les paramètres utilisés pour cette analyse GSEA effectué sont conséquents avec le peu d'échantillons (permutation type : gene_set).

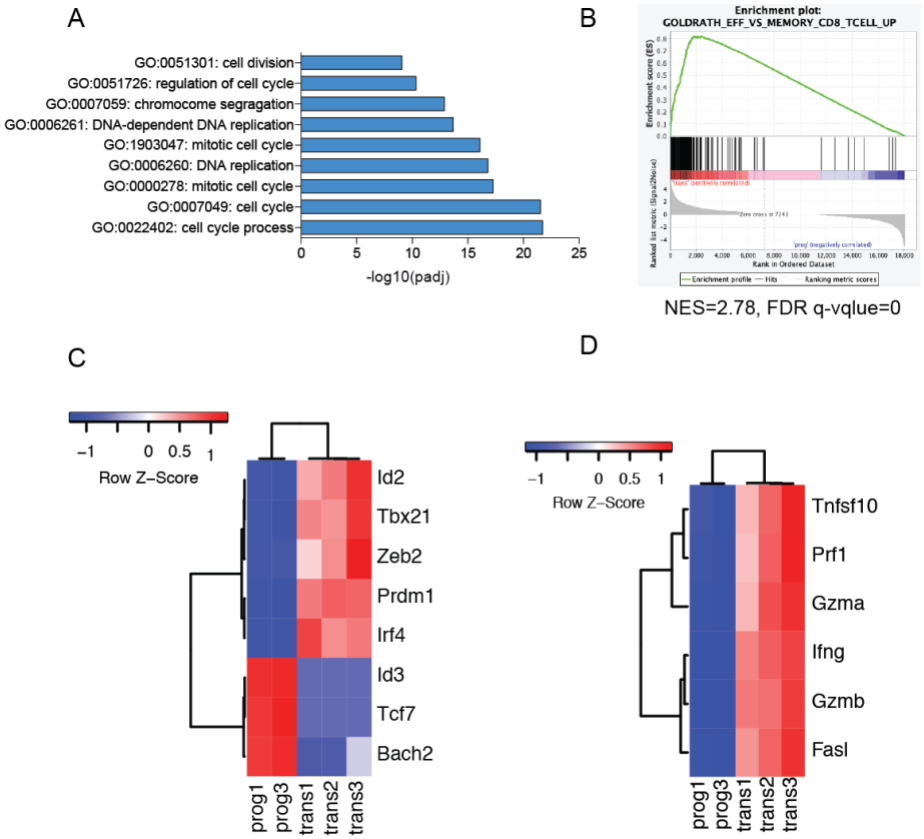


Figure 8. Interprétation biologique des gènes différentiellement exprimés entre les TexP et les TexTrans. (A) Top 10 des termes de la GO :BP obtenus avec gProfiler. (B) *Enrichment plot* pour la liste de gène sur-exprimé dans les T effecteurs versus les mémoires suite à l'analyse GSEA. Heatmap (voir Annexe I-F) de l'expression de gène codant pour des facteurs de transcription (C) et des molécules effectrices (D) importants dans la biologie des cellules T CD8+.

En examinant les gènes plus exprimés dans les TexTrans, on observe par gProfiler que les 10 premiers termes GO :BP associés à ces cellules ont un lien avec le cycle cellulaire (Fig. 8A). De plus, l'analyse par GSEA indique un enrichissement d'une liste de gène plus exprimé dans les CD8 effecteurs versus les cellules mémoires (Goldrath_effector_vs_memory_CD8_Tcell_UP) (Fig. 8B). Cela est conséquent avec le fait que plusieurs facteurs de transcription requis pour la formation efficace des CD8 T effecteurs (*Id2, Tbx21, Zeb2, Prdm1, Irf4*) sont plus exprimés dans les TexTrans alors que ceux associés au maintien homéostatique et à la mémoire immunologique le sont plus dans les TexP (*Id3, Tcf7, Bach2*) (Fig. 8C)[1,22]. De plus, le profil effecteur des TexTrans est relevé par l'expression de gène codant pour des cytokines ou des molécules effectrices (Fig. 8D). Il est donc possible de postuler que la différenciation d'une cellule

TexP en TexTrans soit accompagnée de prolifération et de l'acquisition de fonction effectrice, des processus possiblement régulé par un ou plusieurs facteurs de transcription associés aux cellules T CD8 effectrices retrouvées lors d'une infection aiguë.

Analyse ChIP-seq

Pour l'analyse du ChIP-seq des échantillons de l'étude non-publiée déposés brièvement sur GEO (GSE 149810), un contrôle de qualité FastQC est d'abord appliqué (Fig. 9). Malgré que les séquences semblent de bonne qualité, les pourcentages des nucléotides démontrent des irrégularités. (Fig. 9). Ceci, combiné au fait que l'alignement obtenu suite à l'utilisation de Bowtie2 (cette fois, STAR n'est pas utilisé puisqu'il n'est pas nécessaire de tenir compte de l'épissage) est en deçà de 1% suggère un problème avec les données brutes. Puisque celles-ci ne sont pas publiées encore et qu'elles sont redevenues privées, il nous est impossible d'avoir plus d'information et d'être certain de leur qualité, ou même du fait qu'elles aient étés correctement identifiées lors de leur dépôt sur GEO. Pour ces raisons, il est impossible de poursuivre plus avant avec la stratégie d'associer les marques de chromatine d'activation et de répression avec les états TexT et TexTrans. Par contre, puisque nos analyses de RNAseq démontrent l'importance des facteurs de transcription dans la transition d'un état à l'autre, nous allons intégrer les données RNAseq avec celles d'un ATACseq sur ces mêmes population de cellules d'une étude publiée (GSE#149879) [8]. Dans cette étude, bien qu'un RNAseq ait été fait aussi, peu d'intégration des donnés a été fait. Nous allons 1) pousser l'intégration des données RNAseq analysées ci-haut et avec celles de l'ATACseq et 2) utiliser le pipeline Taiji[23] qui permet l'analyse intégrée de RNAseq et ATACseq à partir d'une simple commande afin de comparer les résultats avec une analyse plus classique.

Analyse ATAC-seq

Afin d'analyse l'état de la chromatine des cellules TexP et TexTrans, les données seront importées d'une publication récente (TexP :SRR11687541, SRR11687542, SRR11687543; TexTrans :SRR11687544, SRR11687545, SRR11687546)[8]. Ces échantillons sont *Paired End*, comme il est recommandé pour les expériences d'ATACseq afin d'améliorer l'alignement. Tel que l'analyse de RNAseq, il est nécessaire de commencer par une analyse de la qualité du séquençage (FASTQC/TRIMMOMATIC). Ensuite le pipeline sera le suivant (Fig. 9) : alignement sur génome de référence, retirer les alignements sur ADN mitochondrial, identifier et enlever les duplicats de PCR, vérifier la taille des inserts, retirer les alignements de mauvaise qualité, décaler les séquences et retirer les éléments sur la 'liste noire' d'ENCODE, identifier les régions de chromatine ouverte (peak calling),

FASTQC et TRIMMOMATIC

Les séquençages d'ADN enrichie, que ce soit par immunoprécipitation ou encore la chromatine ouverte, peuvent être intrinsèquement biaisé. Pour cette raison, lors de l'analyse des résultats de FASTQC, il est plus important de se concentrer sur la présence d'adapteur et la qualité du séquençage que sur les biais potentiels (% de GC par ex.). De l'analyse FASTQC, qu'il y a un échantillon par groupe avec moins de séquences (Fig. 10). De plus, malgré le fait que la qualité des séquences soit généralement bonne, il y a un biais dans la distribution des nucléotides en début des échantillons qui peut être associé à la présence d'adapteur. Pour cette raison, Trimmomatic a été appliqué sur le début des séquences. Il aurait été aussi possible de l'appliquer sur la fin de séquence, qui

est légèrement moins bonne, mais puisqu'elle était passable, cela n'a pas été fait (Fig. 10).

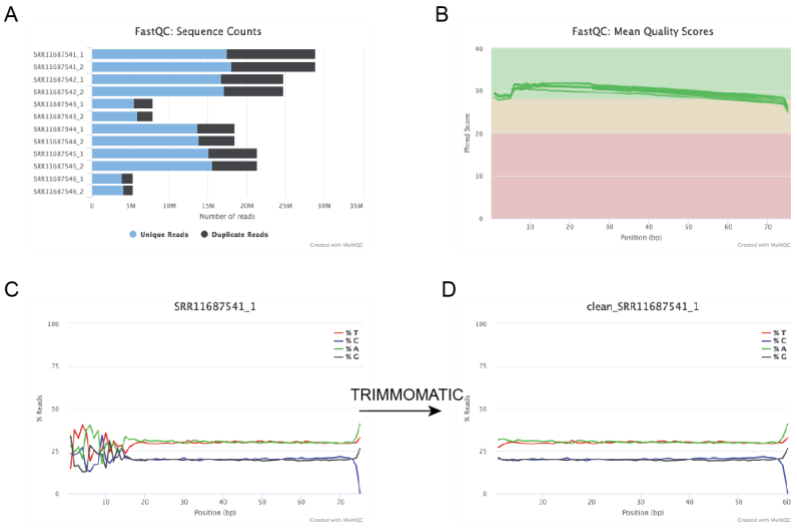


Figure 10. Résultat de contrôle de qualité des données brutes de l'ATACseq. (A) Résultat de FastQC, nombre de séquences. (B) Résultat de FastQC, score de qualité. (C) Résultat de FastQC, % de contenu en paire de base pré- (C) et post- (D) traitement avec Trimmomatic.

Alignement sur génome de référence

Pour aligner les séquences d'ADN sur un génome, d'autres logiciels que STAR sont généralement utilisés. STAR est rapide et permet de résoudre les éléments d'épissage joignant deux séquences d'ADN pour créer l'ARN, par contre les éléments de filtrage de qualité d'alignement de STAR ne sont pas optimisé pour l'ADN et peut causer des difficultés pour détecter les polymorphismes d'un nucléotide (SNP). Pour ces raisons, Bowtie2 (script Annexe III-A)[24]. Après l'alignement, le fichier SAM est convertit en fichier BAM (dans le script Bowtie2), les séquences sont triées sur l'ordre du génome et indexées pour les identifier à l'aide de samtools (script Annexe III-B). L'indexation sera nécessaire pour certaines étapes subséquentes, comme retirer l'ADN mitochondrial, et

aussi pour les logiciels qui permettent de visualiser (*Genome Browser*). Afin d'élever la confiance de l'alignement, puisqu'en ATACseq il n'y a pas de contrôle négatif à proprement parler, l'option `--very-sensitive \` est utilisée.

Nettoyer les alignements

Les prochaines étapes servent à s'assurer que l'analyse des régions de la chromatine ouverte se fasse sur des séquences de haute qualité. Pour se faire il faut d'abord retirer les

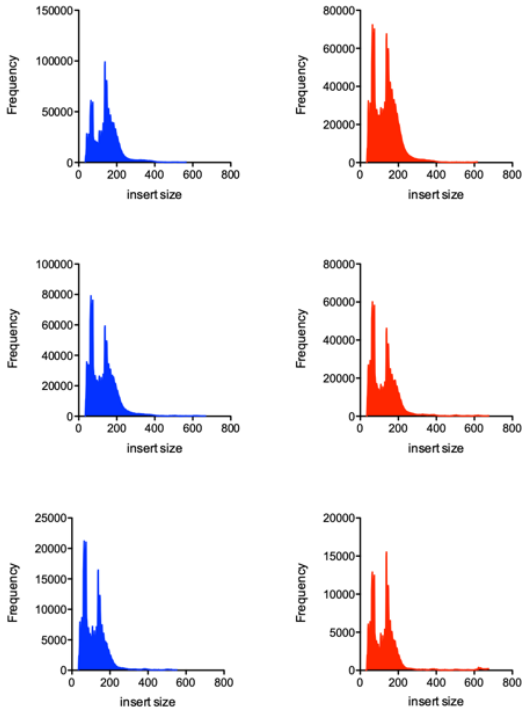


Figure 11. Taille des inserts. La fréquence des inserts est représentée en fonction de la taille des inserts. En bleu, les échantillons TexP, en rouge les TexTrans.

alignements sur ADN mitochondrial, puisque celui-ci n'est pas à l'étude (Annexe III-C). Cela est possible grâce à samtools où on utilise l'index des séquences, créé une liste des séquences alignées sur le chromosome mitochondrial (`indexstat` et `grep`), sam et les coupe

du fichier BAM pour en faire un nouveau (`cut` et `xargs samtools`). Ensuite, il faut identifier et enlever les duplicatas de PCR (Annexe III-D), des artéfacts de l'étape d'amplification qui peuvent biaiser l'enrichissement des séquences. Cela est possible grâce au logiciel Picard (http://broadinstitute.github.io/picard/) et à l'outil `MarkDuplicates` et à la fonction `Remove_Duplicates = TRUE`. En contrôle de qualité, on doit aussi vérifier la taille des inserts (Annexe III-E), à l'aide de l'outil `CollectInsertSizeMetrics` de Picard. Ceci mesure la distance entre deux séquences pairées. Les séquences à 50pb (insertion par la transposase dans une région sans nucléosomes), 200pb (1 nucléosome), 400 pb (2 nucléosomes) et 600 pb (3 nucléosomes) sont attendues à une fréquence plus ou moins décroissante. L'absence claire de ses tailles peut être révélateur d'un manque d'intégrité de la chromatine. Les données dans la Fig. 11 montrent bien ces inserts. Aussi, il est

nécessaire de filtrer les séquences de mauvaises qualitées (ici `–q` à 10) avec samtools view (Annexe III-F) Finalement, dans cette étape de nettoyage des séquences, Deeptools sera nécessaire pour décaler les séquences et retirer les éléments de la liste noire d'ENCODE (script Annexe III-G). Lors de la réaction catalysée par le dimère de la transposase, celle-ci insère les adapteurs séparés par 9pb (+4 sur le brin ADN+ et -5 sur le brin -), pour cette raison, il peut être recommandé de décaler les séquences dans les cas où la précision de l'alignement doit être optimale, lors de la recherche de motifs, par exemple. Les régions de la liste noire (DAC exclusion list) sont des zones de la chromatine qui ont tendance à donner des résultats de séquençage anormalement élevé[25]. Pour cette raison, les alignements dans ces régions sont exclus. Cela est possible grace à la fonction

alignmentSieve. À la fin de ce processus, on obtient un fichier BEDPE avec seulement les séquences alignées de hautes qualités.

Identifier les régions de chromatine ouverte : call peaks

Une fois les séquences nettoyées, il est possible de passer à l'étape d'identifier les régions enrichies. Cela est fait ici avec MACS2, qui détermine les régions enrichies statistiquement [26]. Puisque nous avons déjà exclu les duplicatas de PCR, ici nous utilisons la commande --keep-dup all (Annexe III-H).

Variante Genrich

Les étapes décrites ci-dessus sont relativement encombrantes, mais nécessaire pour préparer un fichier de premier ordre pour MACS2. Il existe d'autres logiciels pour identifier les enrichissements de séquence (peakcaller). Genrich (https://github.com/jsh58/Genrich) est un autre logiciel qui permet de déterminer les pics de séquence, mais qui offre des paramètres pouvant combiner les différentes étapes. Ainsi il y a des options de filtrage pour les duplicatats de PCR ou l'ADN aligné sur le génome mitochondrial (-r, -e), des options pour gérer les ajustements des insertions de la transposase (-D) et des options pour l'identification des pics (-p, -q, …) en une seule étape. Nous avons donc exécuté ce script afin d'en comparer les résultats avec MACS2 (voir script, Annexe III-I).

```
Usage: ./Genrich  -t <file>  -o <file>  [optional arguments]
Required arguments:
  -t  <file>       Input SAM/BAM file(s) for experimental sample(s)
  -o  <file>       Output peak file (in ENCODE narrowPeak format)
Optional I/O arguments:
  -c  <file>       Input SAM/BAM file(s) for control sample(s)
  -f  <file>       Output bedgraph-ish file for p/q values
  -k  <file>       Output bedgraph-ish file for pileups and p-values
  -b  <file>       Output BED file for reads/fragments/intervals
  -R  <file>       Output file for PCR duplicates (only with -r)
Filtering options:
  -r               Remove PCR duplicates
  -e  <arg>        Comma-separated list of chromosomes to exclude
  -E  <file>       Input BED file(s) of genomic regions to exclude
  -m  <int>        Minimum MAPQ to keep an alignment (def. 0)
  -s  <float>      Keep sec alns with AS >= bestAS - <float> (def. 0)
  -y               Keep unpaired alignments (def. false)
  -w  <int>        Keep unpaired alns, lengths changed to <int>
  -x               Keep unpaired alns, lengths changed to paired avg
Options for ATAC-seq:
  -j               Use ATAC-seq mode (def. false)
  -d  <int>        Expand cut sites to <int> bp (def. 100)
  -D               Skip Tn5 adjustments of cut sites (def. false)
Options for peak-calling:
  -p  <float>      Maximum p-value (def. 0.01)
  -q  <float>      Maximum q-value (FDR-adjusted p-value; def. 1)
  -a  <float>      Minimum AUC for a peak (def. 200.0)
  -l  <int>        Minimum length of a peak (def. 0)
  -g  <int>        Maximum distance between signif. sites (def. 100)
Other options:
  -X               Skip peak-calling
  -P               Call peaks directly from a log file (-f)
  -z               Option to gzip-compress output(s)
  -v               Option to print status updates/counts to stderr
```

Figure 12. Paramètres d'utilisation de Genrich.

Analyses différentielles

Que ce soit en utilisant MACS2 ou Genrich, une fois les pics de séquences déterminés, il est possible d'identifier les zones différentiellement accessible (*DAR : diferentially accessible regions*) entre les conditions TexP et TexTrans à l'aide de l'outil Diffbind de R[27] (voir script modifié d'un tutoriel, Annexe III-J). Pour exécuter Diffbind, on doit créer, à l'image de DESeq2, un objet DiffBind décrivant les échantillons à analyser (conditions, échantillons, …) avec les pics identifié à la sortie de Genrich ou MACS2 (narrowPeaks) ainsi que les fichiers bam. Une fois les données normalisées,

l'analyse différentielle est faite en utilisant DESeq2. En appliquant cette méthode sur les données obtenues via Genrich ou MACS2, on peut constater certaines différences (Fig. 13). En analysant la PCA, dans les deux cas il y a une distinction entre les TexTrans et les TexP, par contre le nombre de DAR diffère légèrement entre les logiciels (Genrich : 1769 DAR, 494 TexP et 982 TexTrans; MACS2 1261 DAR, 321 TexP et 940 TexTrans). Toutefois, dans les deux cas, les ouvertures de la chromatine sont plus grandes dans les TexTrans.

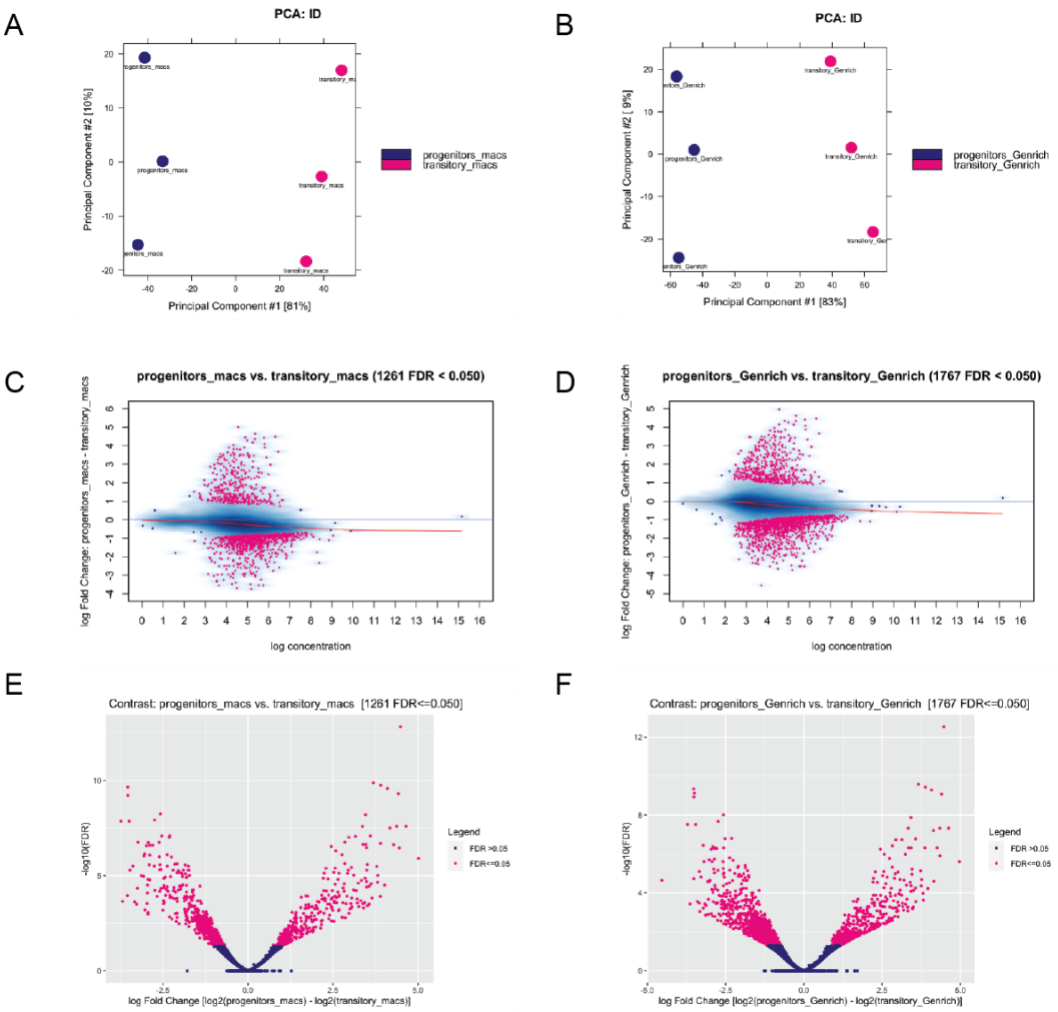


Figure 13. Résultats d'ouverture de chromatine différentielle en utilisant MACS2 our Genrich pour identifier les pics. PCA, MA et volcano plot pour MACS2 (A, C, E) et Genrich (B, D, F).

Homer

Afin de pouvoir faire une interprétation biologique des résultats, la dernière étape consiste à utiliser le logiciel Homer pour identifier les pics, c'est-à-dire associer ceux-ci à son gène le plus près, et analyser les séquences dans ces zones différentiellements ouvertes pour des motifs associés à des facteurs de transcription (script Annexe III-K et – L). Cela pourrait indiquer par quel processus biologiques ces cellules sont

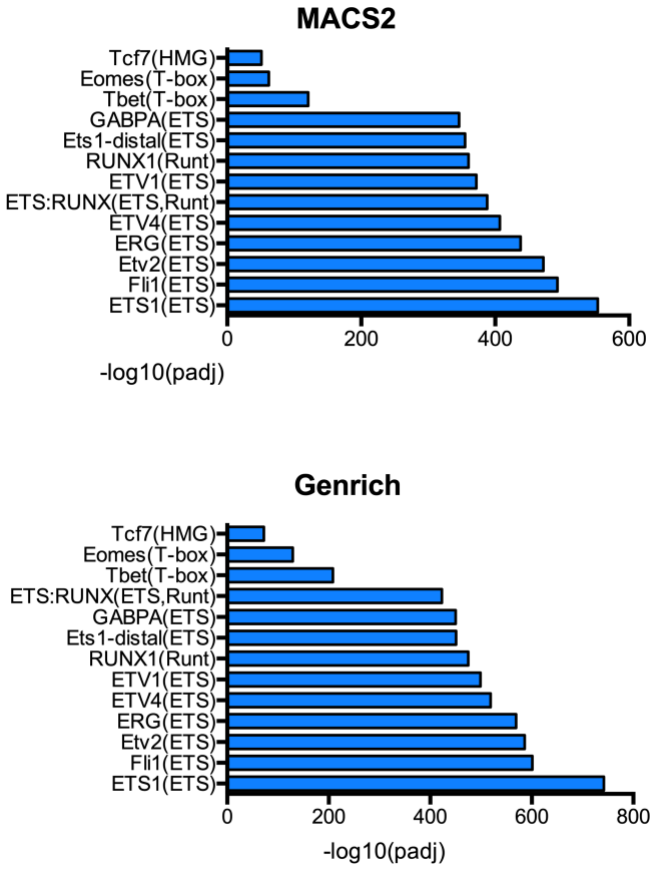


Figure 14. Analyses d'enrichissement des motifs de facteur de transcription via Homer avec les données obtenues par la méthode 'Macs2' ou 'Genrich'.

régulées[28]. Homer permet aussi d'analyser les annotation pour de possible enrichissement de termes dans les bases de données tel que GEO, KEGG et Reactome. Par souci d'économie, nous avons analysé l'ensemble des DAR pour l'enrichissement de motif

(Fig. 14), mais il est aussi possible d'analyser, par exemple les zones plus ouvertes dans les TexTrans seulement. Les résultats obtenus par MACS2 ou Genrich sont similaires. Quoi qu'il y a certaines différences statistiques, cela a peu d'impact sur les conclusions biologiques : les motifs pour la famille de facteurs de transcription ETS sont particulièrement présent dans les DAR entre TexP et TextTrans. C'est la conclusion principale que permet l'analyse des ces échantillons d'ATACseq.

Taiji

L'expression du facteur de transcription seule ne prédit pas nécessairement son activité. Par exemple, un facteur de transcription peut subir des modifications post-translationnelles qui modulent son activité. De plus, afin de pouvoir agir, un facteur de transcriptions doit pouvoir avoir accès à ses sites de liaison sur l'ADN. Le logiciel Taiji (https://taiji-pipeline.github.io/index.html) a été développé afin de favoriser l'intégration des données 'omiques pour prédire et classer *in silico* l'activité des facteurs de transcription dans des conditions données[23]. Bien que le logiciel puisse être utilisé pour une analyse simple de RNAseq, de ChIPseq, d'ATACseq ou encore de scRNAseq/ATACseq, il est plutôt conçu pour combiner deux méthodes.

Taiji : principes

Taiji intègre les différentes informations des expériences de séquençage à haut débit afin de construire un réseau de régulation génique (Fig. 15). Le logiciel détermine d'abord les zones actives de la chromatine à partir de données d'ATACseq ou de ChIPseq, y cherchant des motifs associés aux facteurs de transcription à l'aide de la suite CIS-BP[29]. Ensuite, le facteur de transcription est associé à ses gènes cibles, menant à la construction du réseau d'influence. Finalement, les valeurs d'expression

obtenue par RNAseq sont intégrées et l'algorithme PageRank, un algorithme couramment

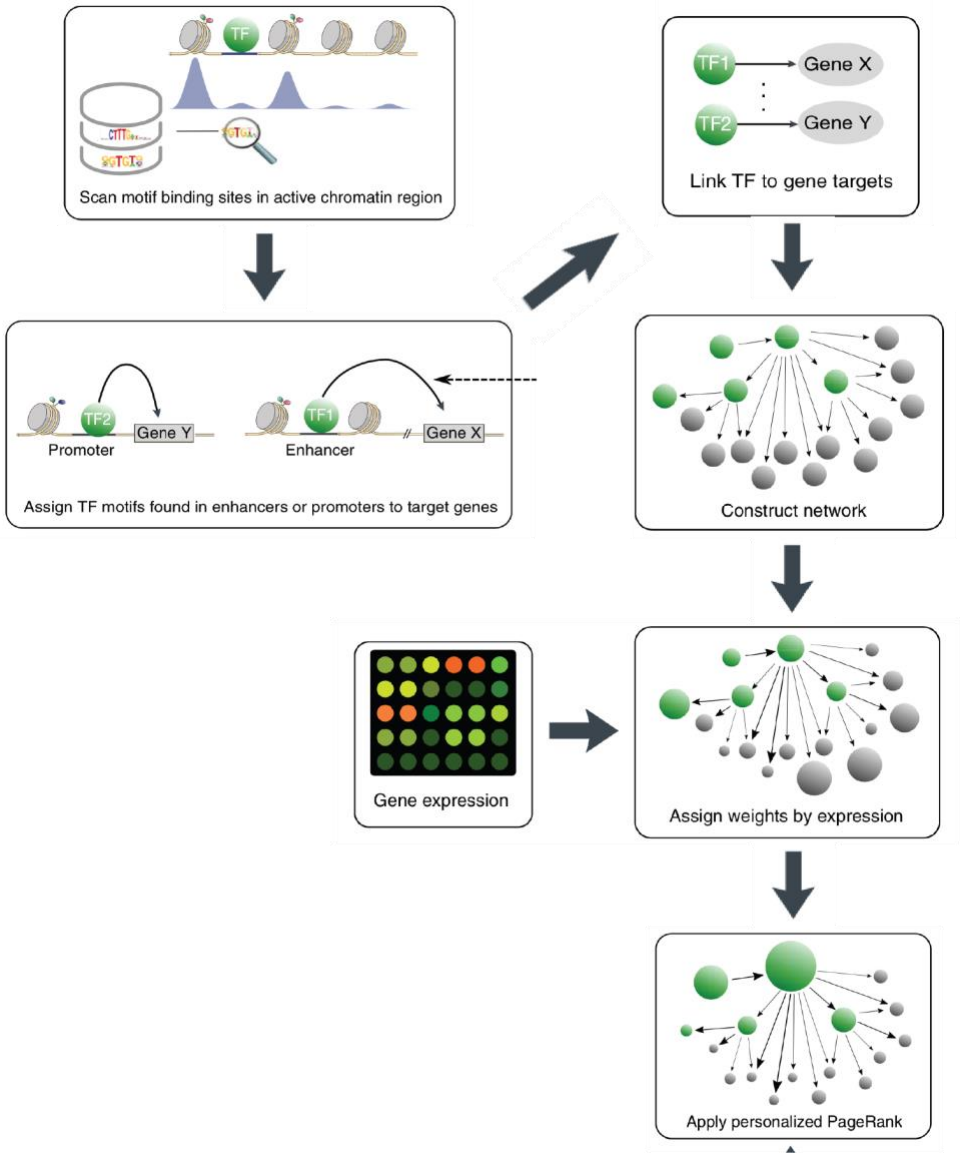


Figure 15. Design de l'algorithme Taiji. Adapté de Zhang *et.al.*, 2019

employé pour analyser les liens entre les pages web afin de les classer, est appliqué afin de déterminer l'influence globale des facteurs de transcription sur le génome. Cet algorithme a été utilisé afin de découvrir l'importance de nouveaux facteurs de transcription dont l'expression varie peu dans le processus de différenciation des lymphocytes T, mais dont les motifs deviennent plus ou moins accessible selon le stade de la réponse immunitaire[30].

Le pipeline de Taiji se base sur une série d'étapes généralement appliquées lors de l'analyse de données de séquençages (voir Annexe III). Par exemple, pour l'analyse d'ATACseq, les étapes inclues : alignement sur génome de référence (BWA), filtrer les données de mauvaises qualités (samtools), enlever les duplicats de PCR, identifier les peaks avec MACS2. Par contre, un des atouts de Taiji est sa simplicité d'opération. Pour l'installation (modifier le système d'exploitation au besoin) :

```
curl -L https://github.com/Taiji-pipeline/Taiji/releases/latest/download/taiji-CentOS-x86_64 -o taiji
chmod +x taiji
./taiji --help
```

Ensuite, selon les analyses, il est nécessaire d'installer d'autres logiciels tel que fastq-dump, samtools, MACS2, bwa, bedGraphToBedWig. Finalement, deux documents doivent être préparés, soit input.tsv et config.yml (voir Annexe IV pour les documents utilisés dans cette étude, config.yml *minimal setting*). Dans le document input, les informations sont entrées afin d'identifier le type de données, les groupes, les réplicats. Ils est possible d'importer des données déposées si le format SRA est indiquée et le #SRR ajouté dans la colonne *path*. Le document config.yml permet certaines spécifications, tel que les paramètres statistiques ou les réglages pour un ordonnanceur, mais celles-ci sont relativement limitées. Pour lancer Taiji, une seule commande suffit :

```
taiji run --config config.yml -n 3 +RTS -N3
```

Où –n 3 +RTS –N3 indique le nombre de cœur à utiliser.

Résultats Taiji

Nous avons découverts que les informations disponibles pour bien exécuter Taiji sont largement incomplètes. Une liste partielle des dépendances et un manque de

spécification des versions à utiliser (pour samtools par exemple) à mené à plusieurs retour d'erreur. Il nous a aussi été impossible, avec ce jeu d'essai erreur, de trouver la combinaison pour exécuter l'analyse sur les serveurs de Calcul Canada. Pour cette raison, nous avons analysé les résultats d'ATAC-seq seulement afin de tester l'opérabilité de Taiji sur un ordinateur personnel.

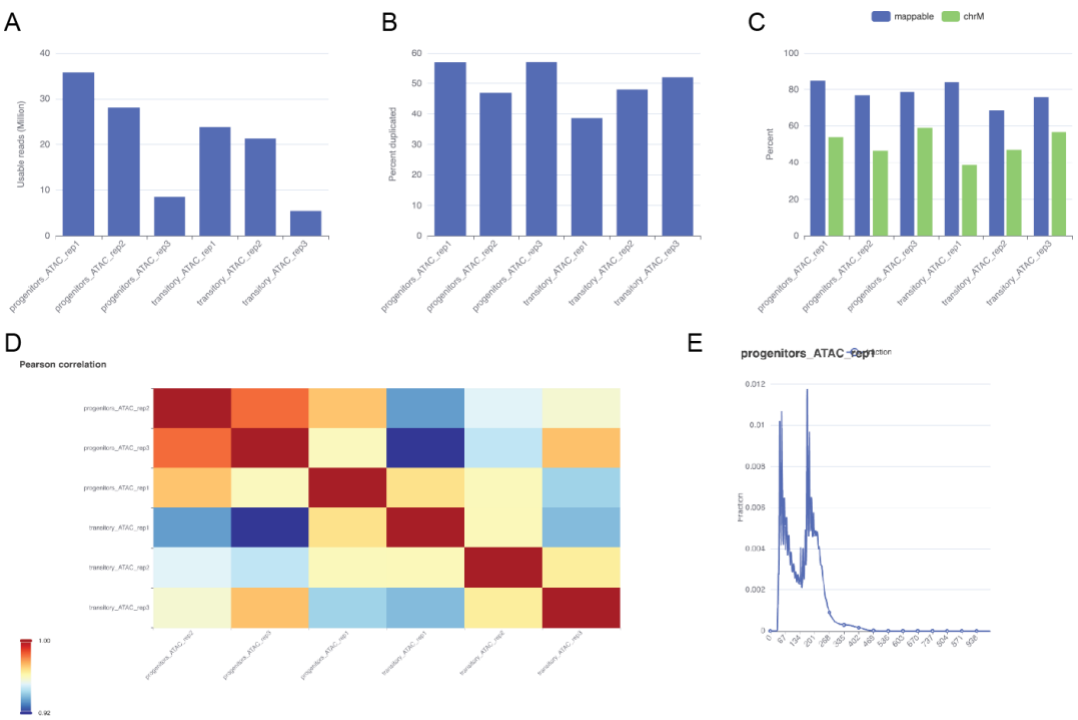


Figure 16. Contrôle de qualité Taiji. Pour tous les échantillons, Taiji produit un graphique représentant (A) le nombres de bonnes séquences (B) le pourcentage de duplicata (C) la proportion d'ADN mitochondrial (D) une corrélation entre les échantillons et (E) la taille des inserts.

En sortie, Taiji produit des analyses contrôles de qualité (Fig. 16), des fichier BED avec les sites d'interaction prédit des facteurs de transcription, des fichiers BW identifiant les zones d'enrichissement, un classement des facteurs de transcription dans les différents groupes – Pagerank – et une analyse des réseaux associé aux facteurs de transcription dans les groupes (.csv). L'analyse Pagerank entre les TexP et les TexTrans a permis d'identifier certains facteurs de transcription associés à une condition ou l'autre (Fig. 17). Ainsi, on peut confirmer le rôle de *Tcf7* (codant pour TCF1) pour les TexP. Le facteur de transcription *Eomes*, détecté ci-haut dans l'analyse Homer, est enrichi dans les TexTrans. Par contre, on note l'absence de facteur de transcription de la famille ETS dans

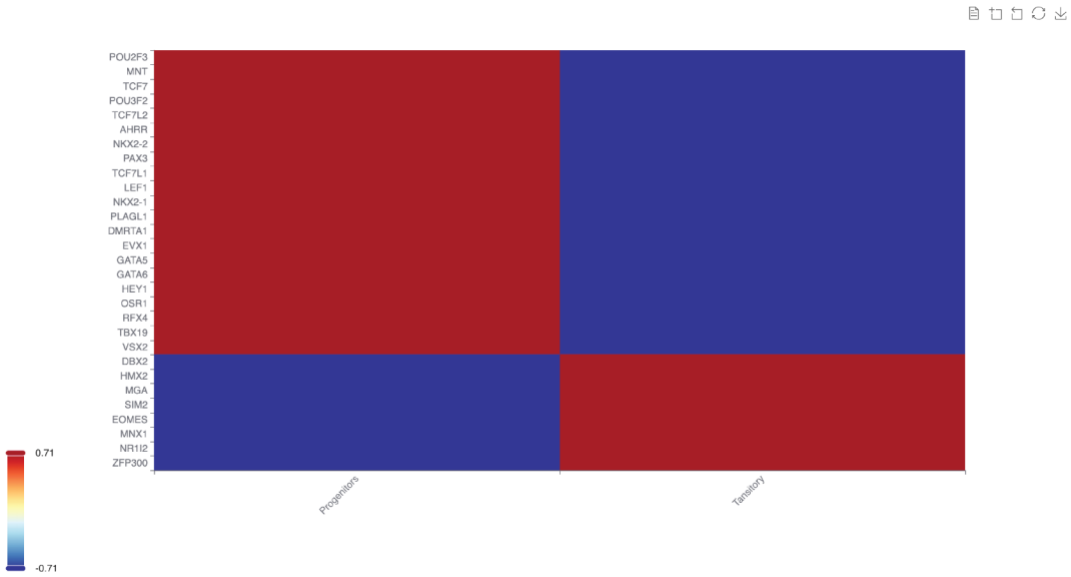


Figure 17. Pagerank Facteurs de transcription ayant les meilleurs scores Pagerank entre les TexP et les TexTrans.

la liste produite par Taiji, qui pourrait résulté de l'algorithme de CIS-BP. Afin d'approfondir le rôle de TCF1 dans la biologie des cellules Tex, à partir des analyses produites par Taiji, il nous a été possible d'identifier des sites différentiellement ouvert entre les TexP et les TexTrans, avec des motifs prédits de TCF1, à proximité de gènes

ayant un rôle important dans la biologie de l'épuisement des cellules T CD8, comme *Nr4a3* (Figure 18)[31].

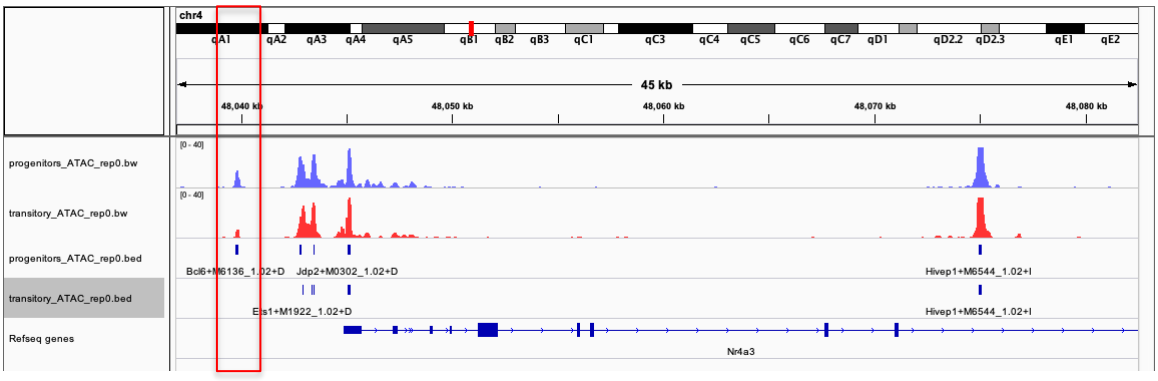


Figure 18. Visualisation d'un motif Tcf7 dans IGV. Une région différentiellement accessible entre les TexP et TexTrans à proximité du gène *Nr4a3* (encadré rouge) contient un motif Tcf7.

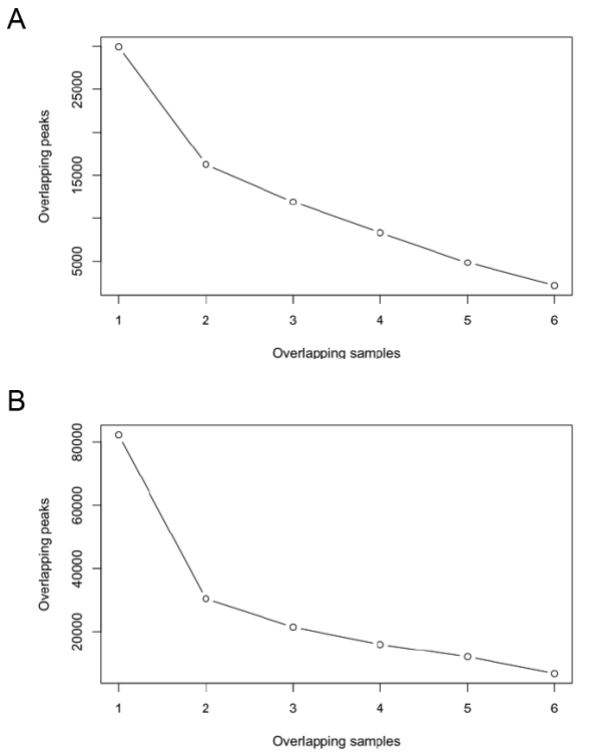
Conclusions

Nous avons tenté par ce travail d'apporter un élément nouveau sur la biologie qui sous-tend la différenciation des TexP en TexTrans en analysant des données publiquement disponible de RNAseq et d'ATACseq sur ces populations. Nous avons de plus comparer différentes approches pour l'analyse d'ATACseq.

Initialement, le plan était d'intégrer des données publiées de RNAseq avec des données publiquement disponibles mais non publiées de ChIP-seq de marques de la chromatine. Cela n'ayant réellement jamais été fait pour ces populations de cellule, nous aurions sans doute contribué de la nouveauté au domaine. Lors du contrôle de qualité, mais plus encore, lors de l'alignement des séquences nous avons réalisé que les donnés ne pouvaient être utilisées. Puisque, depuis, celles-ci sont redevenues privées, il nous est impossible de savoir exactement ce qu'il en est. Est-ce que les séquences ont été mal identifiées lorsque déposé? Cela démontre toutefois l'importance d'effectuer le contrôle de qualité lorsqu'on effectue une fouille de données.

MACS2 est l'un des logiciels servant à l'identification des pics les plus populaire. Initialement conçu pour l'analyse de ChIP-seq, son utilisation dans le contexte d'ATACseq requiert plusieurs étapes pour préparer les séquences et les 'nettoyer' de tout bruit, tel que l'ADN mitochondrial, les séquences de la 'liste noire', les duplicatats de PCR, … Plusieurs scripts sont donc nécessaires pour y parvenir, mais cela est essentiel puisqu'un contrôle négatif tel qu'un IgG ou l'input n'est pas disponible en ATACseq. Genrich est une alternative à MACS2 qui peut aussi servir pour l'identification des pics des expériences d'ATACseq ou de ChIPseq. Celui-ci n'est pas disponible dans l'environnement de Béluga à Calcul Québec, mais peut être cloné de Github (`$ git clone`).

Son principal atout est de regrouper en une seule étape le nettoyage des séquences et l'identification des pics. Nous avons trouvé une différence importante entre le nombre



total de pic appelé par MACS2 versus Genrich. Genrich a identifié plus de 80 000 régions

Figure 19. L'algorithme utilisé pour identifier les pics d'un ATACseq affecte le nombre de région d'intérêt. Nombre de régions identifiées en fonction du nombre d'échantillons dans lequel celles-ci se retrouvent selon le pipeline 'MACS2' (A) ou Genrich (B).

d'intérêt alors que MACS2 30 000 (Fig .19). Cela explique sans doute la différence du nombre de DAR entre les deux approches (1767 vs 1261). Les raisons derrières ces différences découler des algorithmes qui détectent les régions ouvertes de la chromatine. Par contre, il semble plus probable que cela découle des paramètres utilisés entre les deux approches. Il serait utile de modifier ceux-ci afin d'évaluer s'il est possible d'éventuellement obtenir des résultats similaires. Il est important de noter, toutefois que

l'impact des ces différences sur l'interprétation biologiques des données ne semble pas important dans ce cas-ci.

Taiji est conçu pour être une solution complète pour l'analyse et l'intégration de données de RNAseq et ATACseq, entre autres. Il offre en plus une approche unique – Pagerank – pour identifier les facteurs de transcription d'intérêt. Parmi ces avantages, Taiji est conçu avec une option d'auto-recouvrement (fichier sciflow.db), ce qui permet d'interrompre et de reprendre l'analyse en cours. Toutefois, nous avons trouvé des omissions importantes dans la documentation offerte sur le site web, qui ont mené à plusieurs échecs d'analyse. Par exemple, la liste complète des dépendances du logiciel n'est pas offerte, il est nécessaire d'en faire la recherche soi-même. De plus, sans que cela soit mentionné, nous avons découvert qu'il faille éviter certaines versions des logiciels de support. Ainsi, il est nécessaire d'utiliser la dernière version de samtools, mais d'utiliser une ancienne version de sra-toolkit sur le mac. Ces discordances causent des pertes de temps importantes et il est donc important de bien consulter leur page sur *Slack* avant de commencer. De plus, comme beaucoup de solution intégrée, les étapes précises de pipeline de Taiji reste nébuleuses. Il est parfois difficile de savoir comment les différentes étapes sont effectuées, et donc d'être certain de la qualité des données obtenues. Finalement, la philosophie de solution 'tout-en-un' de Taiji, pour but de simplicité, fait en sorte que les paramètres d'utlisation sont très peu flexibles, ce qui peut s'avérer limitant.

L'analyse de RNAseq devait initialement servir de point de comparaison pour les résultats de Taiji. Nous n'avons malheureusement pas été en mesure d'utiliser ce logiciel sur Béluga et l'analyse intégrée des RNAseq et ATACseq était impossible sur un ordinateur personnel. Néanmoins, l'analyse des données RNAseq a largement récapitulé

les découvertes concernant les cellules TexP et TexTrans, tel que l'expression différentielle de certains facteurs de transcription, le profile effecteur des TexTrans et leur capacité à proliférer[8,10], cela suggère que l'analyse a été faite correctement. Puisque l'enrichissement des *GO-term* dans les TexTrans suggère que la transition de TexP à TexTrans requiert une étape de prolifération et que celui des motifs d'ATACseq pointe vers la famille ETS, une famille ayant de rôle important dans prolifération, la différenciation et la survie, l'étude du rôle d'un ou des membres de cette famille de facteur de transcription dans la différenciation des TexTrans serait sans doute l'étape suivant l'analyse présentée dans ce travail. Pour l'instant, l'attention scientifique est portée sur TCF1 et TBET, dont les motifs se trouvent bel et bien dans les DAR entre TexP et TexTrans (voir Fig. 14), mais la proportion importante de membre ETS dans les résultats les plus significatifs suggèrent qu'ils pourraient être important, même si cela reste encore à démontrer.

Références

1. Kaech, S. M. & Cui, W. Transcriptional control of effector and memory CD8+ T cell differentiation. *Nat Rev Immunol* **12,** 749–761 (2012).
2. Wherry, E. J. & Kurachi, M. Molecular and cellular insights into T cell exhaustion. *Nat Rev Immunol* **15,** 486–499 (2015).
3. Robert, C. *et al.* Pembrolizumab versus Ipilimumab in Advanced Melanoma. *N. Engl. J. Med.* **372,** 2521–2532 (2015).
4. Im, S. J. *et al.* Defining CD8+ T cells that provide the proliferative burst after PD-1 therapy. *Nature* **537,** 417–421 (2016).
5. Wu, T. *et al.* The TCF1-Bcl6 axis counteracts type I interferon to repress exhaustion and maintain T cell stemness. *Sci Immunol* **1,** (2016).
6. Philip, M. *et al.* Chromatin states define tumour-specific T cell dysfunction and reprogramming. *Nature* **545,** 452–456 (2017).
7. Yao, C. *et al.* Single-cell RNA-seq reveals TOX as a key regulator of CD8+ T cell persistence in chronic infection. *Nat Immunol* **20,** 890–901 (2019).
8. Beltra, J.-C. *et al.* Developmental Relationships of Four Exhausted CD8+ T Cell Subsets Reveals Underlying Transcriptional and Epigenetic Landscape Control Mechanisms. *Immunity* (2020). doi:10.1016/j.immuni.2020.04.014
9. Chen, Z. *et al.* TCF-1-Centered Transcriptional Network Drives an Effector versus Exhausted CD8 T Cell-Fate Decision. *Immunity* (2019). doi:10.1016/j.immuni.2019.09.013
10. Hudson, W. H. *et al.* Proliferating Transitory T Cells with an Effector-like Transcriptional Signature Emerge from PD-1+ Stem-like CD8+ T Cells during Chronic Infection. *Immunity* **51,** 1043–1058.e4 (2019).
11. Zander, R. *et al.* CD4+ T Cell Help Is Required for the Formation of a Cytolytic CD8+ T Cell Subset that Protects against Chronic Infection and Cancer. *Immunity* (2019). doi:10.1016/j.immuni.2019.10.009
12. Wang, Z., Lachmann, A. & Ma'ayan, A. Mining data and metadata from the gene expression omnibus. *Biophys Rev* **11,** 103–110 (2019).
13. Sade-Feldman, M. *et al.* Defining T Cell States Associated with Response to Checkpoint Immunotherapy in Melanoma. *Cell* **175,** 998–1013.e20 (2018).
14. Andrews, S. Babraham Bioinformatics - FastQC A Quality Control tool for High Throughput Sequence Data. *bioinformatics.babraham.ac.uk* Available at: https://www.bioinformatics.babraham.ac.uk/projects/fastqc/. (Accessed: 22nd April 2021)
15. Ewels, P., Magnusson, M., Lundin, S. & Käller, M. MultiQC: summarize analysis results for multiple tools and samples in a single report. *Bioinformatics* **32,** 3047–3048 (2016).
16. Bolger, A. M., Lohse, M. & Usadel, B. Trimmomatic: a flexible trimmer for Illumina sequence data. *Bioinformatics* **30,** 2114–2120 (2014).
17. Dobin, A. *et al.* STAR: ultrafast universal RNA-seq aligner. *Bioinformatics* **29,** 15–21 (2013).
18. Liao, Y., Smyth, G. K. & Shi, W. featureCounts: an efficient general purpose

program for assigning sequence reads to genomic features. *Bioinformatics* **30,** 923–930 (2014).
19. Love, M. I., Huber, W. & Anders, S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biol* **15,** 550 (2014).
20. Subramanian, A. *et al.* Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proc Natl Acad Sci USA* **102,** 15545–15550 (2005).
21. Raudvere, U. *et al.* g:Profiler: a web server for functional enrichment analysis and conversions of gene lists (2019 update). *Nucleic Acids Res* **47,** W191–W198 (2019).
22. Chen, Y., Zander, R., Khatun, A., Schauder, D. M. & Cui, W. Transcriptional and Epigenetic Regulation of Effector and Memory CD8 T Cell Differentiation. *Front Immunol* **9,** 749 (2018).
23. Zhang, K., Wang, M., Zhao, Y. & Wang, W. Taiji: System-level identification of key transcription factors reveals transcriptional waves in mouse embryonic development. *Sci Adv* **5,** eaav3262 (2019).
24. Langmead, B. & Salzberg, S. L. Fast gapped-read alignment with Bowtie 2. *Nature Methods* **9,** 357–359 (2012).
25. Amemiya, H. M., Kundaje, A. & Boyle, A. P. The ENCODE Blacklist: Identification of Problematic Regions of the Genome. *Sci. Rep.* **9,** 9354 (2019).
26. Zhang, Y. *et al.* Model-based analysis of ChIP-Seq (MACS). *Genome Biol* **9,** R137 (2008).
27. Ross-Innes, C. S. *et al.* Differential oestrogen receptor binding is associated with clinical outcome in breast cancer. *Nature* **481,** 389–393 (2012).
28. Heinz, S. *et al.* Simple combinations of lineage-determining transcription factors prime cis-regulatory elements required for macrophage and B cell identities. *Mol. Cell* **38,** 576–589 (2010).
29. Weirauch, M. T. *et al.* Determination and inference of eukaryotic transcription factor sequence specificity. *Cell* **158,** 1431–1443 (2014).
30. Yu, B. *et al.* Epigenetic landscapes reveal transcription factors that regulate CD8+ T cell differentiation. *Nat Immunol* **18,** 573–582 (2017).
31. Chen, J. *et al.* NR4A transcription factors limit CAR T cell function in solid tumours. *Nature* **567,** 530–534 (2019).

Annexe I – scripts analyses RNAseq

A. FASTQC

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --mem=8Gb

module purge 2>/dev/null
module load mugqic/fastqc/0.11.5
module load mugqic/java
cd $SLURM_SUBMIT_DIR

fastqc --outdir /home/sboulet/projects/def-makarenk/sboulet/binome/rnaseq/results/trim_fastqc \
/home/sboulet/projects/def-makarenk/wanlin/binome/rnaseq/results/trim/clean_SRR8065108.fq.gz \
&> fastqc_clean_prog1.sh.log

fastqc --outdir /home/sboulet/projects/def-makarenk/sboulet/binome/rnaseq/results/trim_fastqc \
/home/sboulet/projects/def-makarenk/wanlin/binome/rnaseq/results/trim/clean_SRR8065117.fq.gz \
&> fastqc_clean_prog2.sh.log

fastqc --outdir /home/sboulet/projects/def-makarenk/sboulet/binome/rnaseq/results/trim_fastqc \
/home/sboulet/projects/def-makarenk/wanlin/binome/rnaseq/results/trim/clean_SRR8065118.fq.gz \
&> fastqc_clean_prog3.sh.log
```

B. TRIMMOMATIC

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --mem=8Gb

module purge 2>/dev/null
module load  StdEnv/2020
module load  trimmomatic/0.39
cd $SLURM_SUBMIT_DIR
```

```
java -jar $EBROOTTRIMMOMATIC/trimmomatic-0.39.jar SE -phred33 \
/home/wanlin/projects/def-makarenk/wanlin/binome/rnaseq/data/SRR8065108.fastq \
/home/wanlin/projects/def-makarenk/wanlin/binome/rnaseq/results/trim/clean_SRR8065108.fq.gz \
HEADCROP:17

java -jar $EBROOTTRIMMOMATIC/trimmomatic-0.39.jar SE -phred33 \
/home/wanlin/projects/def-makarenk/wanlin/binome/rnaseq/data/SRR8065117.fastq \
/home/wanlin/projects/def-makarenk/wanlin/binome/rnaseq/results/trim/clean_SRR8065117.fq.gz \
HEADCROP:17

  java -jar $EBROOTTRIMMOMATIC/trimmomatic-0.39.jar SE -phred33 \
/home/wanlin/projects/def-makarenk/wanlin/binome/rnaseq/data/SRR8065118.fastq \
/home/wanlin/projects/def-makarenk/wanlin/binome/rnaseq/results/trim/clean_SRR8065118.fq.gz \
HEADCROP:17
```

C. STAR

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --mem=60G
#SBATCH --ntasks-per-node=12
#SBATCH --time=4:00:00

module purge 2>/dev/null
module load mugqic/star/2.7.3a

cd $SLURM_SUBMIT_DIR

STAR \
--runMode alignReads \
--runThreadN 12 \
--genomeDir
/cvmfs/soft.mugqic/CentOS6/genomes/species/Mus_musculus.GRCm38/genome/star_index/Ensembl9
7.sjdbOverhang99 \
```

```
--sjdbOverhang 99 \
--sjdbGTFfile
/cvmfs/soft.mugqic/CentOS6/genomes/species/Mus_musculus.GRCm38/annotations/Mus_musculus.G
RCm38.Ensembl97.gtf \
--readFilesIn \
/home/sboulet/projects/def-makarenk/wanlin/binome/rnaseq/results/trim/clean_SRR8065107.fq.gz \
--readFilesCommand zcat \
--outFileNamePrefix /home/sboulet/projects/def-
makarenk/sboulet/binome/rnaseq/results/star/prog1/prog1 \
--outSAMtype BAM SortedByCoordinate \
--limitBAMsortRAM 32400000000 \
&> star_prog1.sh.log
```

D. featureCounts

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --mem=8192
#SBATCH --ntasks-per-node=2

module purge 2>/dev/null
module load subread/1.6.4

cd $SLURM_SUBMIT_DIR

featureCounts \
-T 2 \
-a /cvmfs/soft.mugqic/CentOS6/genomes/species/Mus_musculus.GRCm38/annotations/M$
-o /home/sboulet/projects/def-makarenk/sboulet/binome/rnaseq/results/featurecou$
/home/sboulet/projects/def-makarenk/sboulet/binome/rnaseq/results/star/prog1/pr$
&> featurecounts_prog1.sh.log
```

E. DESeq2

```
### install R & Rstudio
### install packages
### create samples.txt
### download featurecounts results

setwd("/Users/salixboulet/Documents/Biostatistiques/DESS_Uqam/BIF7104/Projet_session/rnaseq/des
eq")

install.packages("data.table","dplyr", "stringr")
if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")
BiocManager::install("DESeq2")

library(data.table)
library(DESeq2)
library(dplyr)
library(stringr)

samplesFile <- fread("samples.txt")
conditions <- factor(samplesFile$condition)
samples <- samplesFile$sample

# Read the featureCount files.
countData.list <- lapply(samples, function(sample) {
  file <-
Sys.glob(file.path("/Users/salixboulet/Documents/Biostatistiques/DESS_Uqam/BIF7104/Projet_session/r
naseq/deseq/featurecounts/", paste0(sample, ".txt")))
  dd <- fread(file, skip=1)
  dd <- dd %>% select(Geneid, Length, contains("bam"))
  setnames(dd, 3, sample)
  return(dd)
})

# Convert the list of dataframes to one dataframe
countData <- as.data.frame(do.call(cbind, countData.list))
```

```
# Remove the duplicate columns
countData <- countData[, unique(colnames(countData))]
colData <- data.frame(conditions, row.names=colnames(countData)[3:ncol(countData)])

# Create a DESeqDataSet object.
dds <- DESeqDataSetFromMatrix(countData=subset(countData, select=-Length), colData=colData,
design= ~ conditions, tidy=TRUE)
dds <- DESeq(dds)

# Give the gene lengths to DESeq
mcols(dds)$basepairs <- countData$Length

# Calculate and save normalized counts
normalized.counts <- counts(dds, normalized=TRUE)
write.table(normalized.counts, "counts_normalized_relative_to_library_size.tsv", quote=FALSE, sep="\t",
row.names=FALSE)

# Calculate and save FPKM counts
fpkm.counts <- fpkm(dds)
write.table(fpkm.counts, "fpkm_counts.tsv", quote=FALSE, sep="\t", row.names=FALSE)

#####################################
# Differential expression analysis. #
#####################################

condition.numerator <- "trans"
condition.denominator <- "prog"

deseq.results <- results(dds, contrast=c("conditions",condition.numerator,condition.denominator))
dim(deseq.results)

normalized.counts.with.deseq.results <- merge(normalized.counts, as.data.frame(deseq.results),
by="row.names")
#can use write table function here to get results
write.table(normalized.counts.with.deseq.results, "normalized.counts.with.deseq.results.tsv",
quote=FALSE, sep="\t", row.names=FALSE)
```

```
####################################
# Plots #
####################################

#install.packages("amap")
library(amap)
library(FactoMineR)

h<-hcluster(t(counts(dds, normalized=TRUE)), method="spearman")
plot(h)

rld <- rlog(dds)
assay.rld.t <- t(assay(rld))
pca <- PCA(assay.rld.t, graph=FALSE)
plot(pca)


library(rrcov) # detecter les outliers
rPCA_rlog <- PcaGrid(assay.rld.t, k=2)
plot(rPCA_rlog)

####################################rPCA-rlog idicates outlier, exclude
prog2########################################

# Create a DESeqDataSet object.
dds <- dds[,-2]
dds <- DESeq(dds)

# Give the gene lengths to DESeq
mcols(dds)$basepairs <- countData$Length

# Calculate and save normalized counts
normalized.counts <- counts(dds, normalized=TRUE)
write.table(normalized.counts, "counts_normalized_relative_to_library_size2.tsv", quote=FALSE,
sep="\t", row.names=FALSE)
```

```
# Calculate and save FPKM counts
fpkm.counts <- fpkm(dds)
write.table(fpkm.counts, "fpkm_counts2.tsv", quote=FALSE, sep="\t", row.names=FALSE)

#####################################
# Differential expression analysis. #
#####################################

condition.numerator <- "trans"
condition.denominator <- "prog"

deseq.results <- results(dds, contrast=c("conditions",condition.numerator,condition.denominator))
dim(deseq.results)

normalized.counts.with.deseq.results <- merge(normalized.counts, as.data.frame(deseq.results),
by="row.names")
#can use write table function here to get results
write.table(normalized.counts.with.deseq.results, "normalized.counts.with.deseq.results2.tsv",
quote=FALSE, sep="\t", row.names=FALSE)

#####################################
# Plots #
#####################################


h<-hcluster(t(counts(dds, normalized=TRUE)), method="spearman")
plot(h)

rld <- rlog(dds)
assay.rld.t <- t(assay(rld))
pca <- PCA(assay.rld.t, graph=FALSE)
plot(pca)

rPCA_rlog <- PcaGrid(assay.rld.t, k=2)
plot(rPCA_rlog)

#get gene names
```

```
if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")
BiocManager::install("org.Mm.eg.db")
library(org.Mm.eg.db)
ens <-DEG_full$Row.names
geneid<-mapIds(org.Mm.eg.db, keys = ens,
               column = c('SYMBOL'), keytype = 'ENSEMBL')
DEG_full= subset(normalized.counts.with.deseq.results)
DEG_full<-cbind(DEG_full, geneid)
DEG_trans_prog= subset(DEG_full, padj<0.05 & abs(log2FoldChange)>1)


#subset data
#DEG_trans_prog= subset(normalized.counts.with.deseq.results, padj<0.05)
#DEG_trans_prog= subset(normalized.counts.with.deseq.results, padj<0.05 & abs(log2FoldChange)>1)


#install.packages("gplots")
library(gplots)


tfinal_select=t(as.matrix(log2(DEG_trans_prog[,2:6]+1))) # le +1 évite les valeurs infinis avec les 0
ztfinal_select=scale(tfinal_select, center=T, scale = T)
tztfinal_select=t(ztfinal_select)
rownames(tztfinal_select)=DEG_trans_prog$geneid

heatmap.2(as.matrix(tztfinal_select),main =paste(nrow(DEG_trans_prog),"DEGs \n padj<0.05 \n z-
score"),
          cexRow=0.6,cexCol=1,scale = "none", trace="none", density.info="none",Colv = T,
          col =colorRampPalette(c("blue","white","red"))(60), symm=F,symkey=F,symbreaks=F, lhei =
c(3,12),margins=c(7,12))


##### Volcano BIF7104 ##########

if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")
```

```
BiocManager::install("EnhancedVolcano")

library(EnhancedVolcano)

EnhancedVolcano(deseq.results,x = "log2FoldChange",y = "padj",
                lab=rownames(deseq.results), pCutoff=0.05,
                cutoffLineType = "blank", subtitle = "", legendPosition = "top")



############ customized volcano ##############
library(ggplot2)

subset<-subset(DEG_full, padj<=.05 & abs(log2FoldChange)>1)
DEG_positive<-subset(DEG_full, padj<=.05 & log2FoldChange>1)
DEG_negative<-subset(DEG_full, padj<=.05 & log2FoldChange < -1)
DEG_NS<-subset(DEG_full, padj>.05 | abs(log2FoldChange) < 1)

#using ggplot
require("ggrepel")
set.seed(42)
ggplot(DEG_NS, aes(log2FoldChange, -log10(padj), label=geneid))+
  geom_point(shape=21, size=1, color="black", fill="black")+
  geom_point(data=DEG_positive, shape=21, size=0.5, color="red", fill="red")+
  geom_point(data=DEG_negative, shape=21, size=0.5, color="blue", fill="blue")+
  geom_text_repel(data=subset(DEG_positive, geneid%in%c('Cx3cr1','Havcr2', 'Tbx21') &
abs(log2FoldChange)>1), size=4, segment.size  = 0.2, segment.color = "grey50", fontface="italic")+
  geom_text_repel(data=subset(DEG_negative, geneid%in%c('Tcf7','Cxcr5','Bcl6') &
abs(log2FoldChange)>1),nudge_x=-0.7, size=4,segment.size  = 0.2, segment.color = "grey50",
fontface="italic")+
  theme_classic()
```

F. Heatmap

```
#load libraries
library(readxl) #load libraries
library(edgeR)
library(grid)
library(limma)
```

```
library(Glimma)
library(gplots)
library(org.Mm.eg.db)
library(RColorBrewer)
library(ggplot2)
library(dplyr)

DEG <-
read.delim("~/Documents/Biostatistiques/DESS_Uqam/BIF7104/Projet_session/rnaseq/deseq/DEG.tsv",
header=TRUE)
View(DEG)

#convert to dataframe
as.data.frame(DEG)

DEG<-DEG[, 2:13]
names <- DEG[,12]

#define genes of interest
TF <- c('Tbx21', 'Zeb2', 'Prdm1', 'Id2', 'Id3', 'Tcf7', 'Bach2', 'Irf4')
Func <- c('Gzma', 'Gzmb', 'Ifng', 'Prf1', 'Fasl', 'Tnfsf10')

#select from data FT
FT_DEG<-subset(DEG, DEG$geneid %in% TF, select=c(1:5, 12))

#Subset matrix and create matrix with good column names FT
xFT<-FT_DEG
rnames_FT <- xFT[,6]
yFT<-xFT[, 1:5]
zFT<-cbind(rnames_FT, yFT)
row.names(zFT)<-zFT[,1]
zFT<-zFT[,2:6]
mat_FT <- data.matrix(zFT)



#draw heatmap helpless day 8
my_palette <- colorRampPalette(c("blue","white","red"))(n = 299)
```

```
rowv8help <- as.dendrogram(hclust(as.dist(1-cor(t(mat_FT))))) # clusturing methods
colv8help <- as.dendrogram(hclust(as.dist(1-cor(mat_FT))))
heatmap8help<-heatmap.2 (mat_FT,
                    #breaks=col_breaks,    # enable color transition at specified limits
                    scale = "row", # normalize data to row for better viewing
                    Rowv=rowv8help, # indic livia
                    Colv=colv8help, # indic livia
                    key = "TRUE",# indic livia
                    keysize = 0.2,# indic livia
                    key.title = NA,# indic livia
                    density.info="none",# indic livia
                    trace="none",         # turns off trace lines inside the heat map
                    cexRow = 0.6,         #size of characters row
                    cexCol = 0.6,         #column
                    margins = c(3,6),
                    lhei = c(2,6), #set object size height
                    lwid = c(2,2), # width
                    col=my_palette,       # use on color palette defined earlier
                    dendrogram="both"    # only draw a column/samples dendrogram (clustering) (possible
to do it for row)
)

#select from data Func
Func_DEG<-subset(DEG, DEG$geneid %in% Func, select=c(1:5, 12))

#Subset matrix and create matrix with good column names FT
xFunc<-Func_DEG
rnames_Func <- xFunc[,6]
yFunc<-xFunc[, 1:5]
zFunc<-cbind(rnames_Func, yFunc)
row.names(zFunc)<-zFunc[,1]
zFunc<-zFunc[,2:6]
mat_Func <- data.matrix(zFunc)


#draw heatmap helpless day 8
my_palette <- colorRampPalette(c("blue","white","red"))(n = 299)
```

```
rowvFunc <- as.dendrogram(hclust(as.dist(1-cor(t(mat_Func))))) # clusturing methods
colvFunc <- as.dendrogram(hclust(as.dist(1-cor(mat_Func))))
heatmap8help<-heatmap.2 (mat_Func,
                    #breaks=col_breaks,    # enable color transition at specified limits
                    scale = "row", # normalize data to row for better viewing
                    Rowv=rowvFunc, # indic livia
                    Colv=colvFunc, # indic livia
                    key = "TRUE",# indic livia
                    keysize = 0.2,# indic livia
                    key.title = NA,# indic livia
                    density.info="none",# indic livia
                    trace="none",         # turns off trace lines inside the heat map
                    cexRow = 0.6,         #size of characters row
                    cexCol = 0.6,         #column
                    margins = c(3,6),
                    lhei = c(2,6), #set object size height
                    lwid = c(2,2), # width
                    col=my_palette,       # use on color palette defined earlier
                    dendrogram="both"    # only draw a column/samples dendrogram (clustering) (possible
to do it for row)
)

dev.off()
```

Annexe II – scripts analyses ChIP seq

A. Bowtie2

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --mem=60G
#SBATCH --ntasks-per-node=5
#SBATCH --time=4:00:00module purge 2>/dev/null

module load mugqic/bowtie2/2.3.5
module load mugqic/java
cd $SLURM_SUBMIT_DIRbowtie2 \

-p 5 \
-x
/cvmfs/ref.mugqic/genomes/species/Mus_musculus.GRCm38/genome/bowtie2_index/Mus_musculus.
GRCm38 \
-U /home/wanlin/projects/def-makarenk/wanlin/binome/chipseq/data/SRR11684793.fastq \
-S /home/wanlin/projects/def-makarenk/wanlin/binome/chipseq/results/bowtie2/793_test.sam \
&> test_bowtie.log
```

Annexe III-scripts ATAC-seq

A. Bowtie2

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --mem=100G
#SBATCH --ntasks-per-node=5
#SBATCH --time=10:00:00

module purge 2>/dev/null
module load mugqic/bowtie2/2.3.5
module load mugqic/samtools/1.10
module load mugqic/java
cd $SLURM_SUBMIT_DIR

bowtie2 \
--very-sensitive \
-X 1000 \
-x /cvmfs/ref.mugqic/genomes/species/Mus_musculus.GRCm38/genome/bowtie2_index/Mus_musculus.GRCm38 \
-1 /home/wanlin/projects/def-makarenk/wanlin/atacseq/results/trim/clean_SRR11687541_1.fq.gz \
-2 /home/wanlin/projects/def-makarenk/wanlin/atacseq/results/trim/clean_SRR11687541_2.fq.gz \
1> /home/wanlin/projects/def-makarenk/wanlin/atacseq/results/bowtie2/541.sam \
2> 541.sh.log

samtools view -hbS /home/wanlin/projects/def-makarenk/wanlin/atacseq/results/bowtie2/541.sam \
1> /home/wanlin/projects/def-makarenk/wanlin/atacseq/results/bowtie2/541.bam \
2>> 541.sh.log

rm /home/wanlin/projects/def-makarenk/wanlin/atacseq/results/bowtie2/541.sam \
2>> 541.sh.log
```

B. Samtool sort and index

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
```

```
#SBATCH --mem=16Gb
#SBATCH --time=6:00:00

module purge 2>/dev/null
module load samtools/1.10
cd $SLURM_SUBMIT_DIR

samtools sort \
-o ../../results/bowtie2/541_sorted.bam \
../../results/bowtie2/541.bam \
&> samtools_sort_541.sh.log

samtools index ../../results/bowtie2/541_sorted.bam \
2>> samtools_sort_541.sh.log

samtools sort \
-o ../../results/bowtie2/542_sorted.bam \
../../results/bowtie2/542.bam \
&> samtools_sort_542.sh.log

samtools index ../../results/bowtie2/542_sorted.bam \
2>> samtools_sort_542.sh.log

samtools sort \
-o ../../results/bowtie2/543_sorted.bam \
../../results/bowtie2/543.bam \
&> samtools_sort_543.sh.log

samtools index ../../results/bowtie2/543_sorted.bam \
2>> samtools_sort_543.sh.log

samtools sort \
-o ../../results/bowtie2/544_sorted.bam \
../../results/bowtie2/544.bam \
&> samtools_sort_544.sh.log

samtools index ../../results/bowtie2/544_sorted.bam \
```

```
2>> samtools_sort_544.sh.log

samtools sort \
-o ../../results/bowtie2/545_sorted.bam \
../../results/bowtie2/545.bam \
&> samtools_sort_545.sh.log

samtools index ../../results/bowtie2/545_sorted.bam \
2>> samtools_sort_545.sh.log

samtools sort \
-o ../../results/bowtie2/546_sorted.bam \
../../results/bowtie2/546.bam \
&> samtools_sort_546.sh.log

samtools index ../../results/bowtie2/546_sorted.bam \
2>> samtools_sort_546.sh.log
```

C. Remove mtDNA :samtools

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --mem=6Gb
#SBATCH --time=2:00:00

module purge 2>/dev/null
module load samtools/1.10
cd $SLURM_SUBMIT_DIR

samtools idxstats ../../results/bowtie2/541_sorted.bam | cut -f -1 | grep -v MT | xargs samtools view -b ../../results/bowtie2/541_sorted.bam > ../../results/removeMT/541_sorted_rmMT.bam

samtools idxstats ../../results/bowtie2/542_sorted.bam | cut -f -1 | grep -v MT | xargs samtools view -b ../../results/bowtie2/542_sorted.bam > ../../results/removeMT/542_sorted_rmMT.bam

samtools idxstats ../../results/bowtie2/543_sorted.bam | cut -f -1 | grep -v MT | xargs samtools view -b ../../results/bowtie2/543_sorted.bam > ../../results/removeMT/543_sorted_rmMT.bam
```

samtools idxstats ../../results/bowtie2/544_sorted.bam | cut -f -1 | grep -v MT | xargs samtools view -b ../../results/bowtie2/544_sorted.bam > ../../results/removeMT/544_sorted_rmMT.bam

samtools idxstats ../../results/bowtie2/545_sorted.bam | cut -f -1 | grep -v MT | xargs samtools view -b ../../results/bowtie2/545_sorted.bam > ../../results/removeMT/545_sorted_rmMT.bam

samtools idxstats ../../results/bowtie2/546_sorted.bam | cut -f -1 | grep -v MT | xargs samtools view -b ../../results/bowtie2/546_sorted.bam > ../../results/removeMT/546_sorted_rmMT.bam

D. Enlever les duplicats de PCR : Picard

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --mem=30Gb
#SBATCH --time=8:00:00

module purge 2>/dev/null
module load mugqic/java
module load StdEnv/2020
module load mugqic/picard/1.123
cd $SLURM_SUBMIT_DIR

java -jar /cvmfs/soft.mugqic/CentOS6/software/picard/picard-tools-1.123/MarkDuplicates.jar \
INPUT=../../results/removeMT/541_sorted_rmMT.bam \
OUTPUT=../../results/picard/541_sorted_rmMT_rmDupli.bam \
METRICS_FILE=../../results/picard/541_picard_metrics.txt \
REMOVE_DUPLICATES=true \
&> 541_picard.sh.log

java -jar /cvmfs/soft.mugqic/CentOS6/software/picard/picard-tools-1.123/MarkDuplicates.jar \
INPUT=../../results/removeMT/542_sorted_rmMT.bam \
OUTPUT=../../results/picard/542_sorted_rmMT_rmDupli.bam \
METRICS_FILE=../../results/picard/542_picard_metrics.txt \
REMOVE_DUPLICATES=true \
&> 542_picard.sh.log

java -jar /cvmfs/soft.mugqic/CentOS6/software/picard/picard-tools-1.123/MarkDuplicates.jar \
```

```
INPUT=../../results/removeMT/543_sorted_rmMT.bam \
OUTPUT=../../results/picard/543_sorted_rmMT_rmDupli.bam \
METRICS_FILE=../../results/picard/543_picard_metrics.txt \
REMOVE_DUPLICATES=true \
&> 543_picard.sh.log

java -jar /cvmfs/soft.mugqic/CentOS6/software/picard/picard-tools-1.123/MarkDuplicates.jar \
INPUT=../../results/removeMT/544_sorted_rmMT.bam \
OUTPUT=../../results/picard/544_sorted_rmMT_rmDupli.bam \
METRICS_FILE=../../results/picard/544_picard_metrics.txt \
REMOVE_DUPLICATES=true \
&> 544_picard.sh.log

java -jar /cvmfs/soft.mugqic/CentOS6/software/picard/picard-tools-1.123/MarkDuplicates.jar \
INPUT=../../results/removeMT/545_sorted_rmMT.bam \
OUTPUT=../../results/picard/545_sorted_rmMT_rmDupli.bam \
METRICS_FILE=../../results/picard/545_picard_metrics.txt \
REMOVE_DUPLICATES=true \
&> 545_picard.sh.log

java -jar /cvmfs/soft.mugqic/CentOS6/software/picard/picard-tools-1.123/MarkDuplicates.jar \
INPUT=../../results/removeMT/546_sorted_rmMT.bam \
OUTPUT=../../results/picard/546_sorted_rmMT_rmDupli.bam \
METRICS_FILE=../../results/picard/546_picard_metrics.txt \
REMOVE_DUPLICATES=true \
&> 546_picard.sh.log
```

E. Vérifier la taille des inserts

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --mem=30Gb
#SBATCH --time=8:00:00

module purge 2>/dev/null
module load mugqic/java
```

```
module load StdEnv/2020
module load mugqic/picard/1.123
cd $SLURM_SUBMIT_DIR

java -jar /cvmfs/soft.mugqic/CentOS6/software/picard/picard-tools-1.123/CollectInsertSizeMetrics.jar \
I=../../results/picard/541_sorted_rmMT_rmDupli.bam \
O=../../results/taille_insert/541_insert_size_metrics.txt \
H=../../results/taille_insert/541_insert_size_histogram.pdf \
M=0.5 \
&> 541_insertSize.sh.log

java -jar /cvmfs/soft.mugqic/CentOS6/software/picard/picard-tools-1.123/CollectInsertSizeMetrics.jar \
I=../../results/picard/542_sorted_rmMT_rmDupli.bam \
O=../../results/taille_insert/542_insert_size_metrics.txt \
H=../../results/taille_insert/542_insert_size_histogram.pdf \
M=0.5 \
&> 542_insertSize.sh.log

java -jar /cvmfs/soft.mugqic/CentOS6/software/picard/picard-tools-1.123/CollectInsertSizeMetrics.jar \
I=../../results/picard/543_sorted_rmMT_rmDupli.bam \
O=../../results/taille_insert/543_insert_size_metrics.txt \
H=../../results/taille_insert/543_insert_size_histogram.pdf \
M=0.5 \
&> 543_insertSize.sh.log

java -jar /cvmfs/soft.mugqic/CentOS6/software/picard/picard-tools-1.123/CollectInsertSizeMetrics.jar \
I=../../results/picard/544_sorted_rmMT_rmDupli.bam \
O=../../results/taille_insert/544_insert_size_metrics.txt \
H=../../results/taille_insert/544_insert_size_histogram.pdf \
M=0.5 \
&> 544_insertSize.sh.log

java -jar /cvmfs/soft.mugqic/CentOS6/software/picard/picard-tools-1.123/CollectInsertSizeMetrics.jar \
I=../../results/picard/545_sorted_rmMT_rmDupli.bam \
O=../../results/taille_insert/545_insert_size_metrics.txt \
H=../../results/taille_insert/545_insert_size_histogram.pdf \
M=0.5 \
```

```
&> 545_insertSize.sh.log

java -jar /cvmfs/soft.mugqic/CentOS6/software/picard/picard-tools-1.123/CollectInsertSizeMetrics.jar \
I=../../results/picard/546_sorted_rmMT_rmDupli.bam \
O=../../results/taille_insert/546_insert_size_metrics.txt \
H=../../results/taille_insert/546_insert_size_histogram.pdf \
M=0.5 \
&> 546_insertSize.sh.log
```

F. Retirer les alignements de mauvaise qualité : samtools

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --mem=16Gb
#SBATCH --time=6:00:00

module purge 2>/dev/null
module load samtools/1.10
cd $SLURM_SUBMIT_DIR

samtools view \
-b \
-q 10 \
../../results/picard/541_sorted_rmMT_rmDupli.bam                                    >
../../results/filterLowQuality/541_sorted_rmMT_Dupli_lowQ.bam

samtools view \
-b \
-q 10 \
../../results/picard/542_sorted_rmMT_rmDupli.bam                                    >
../../results/filterLowQuality/542_sorted_rmMT_Dupli_lowQ.bam

samtools view \
-b \
-q 10 \
```

```
../../results/picard/543_sorted_rmMT_rmDupli.bam >
../../results/filterLowQuality/543_sorted_rmMT_Dupli_lowQ.bam

samtools view \
-b \
-q 10 \
../../results/picard/544_sorted_rmMT_rmDupli.bam >
../../results/filterLowQuality/544_sorted_rmMT_Dupli_lowQ.bam

samtools view \
-b \
-q 10 \
../../results/picard/545_sorted_rmMT_rmDupli.bam >
../../results/filterLowQuality/545_sorted_rmMT_Dupli_lowQ.bam

samtools view \
-b \
-q 10 \
../../results/picard/546_sorted_rmMT_rmDupli.bam >
../../results/filterLowQuality/546_sorted_rmMT_Dupli_lowQ.bam
```

G. Décaler les séquences et retirer les éléments de la liste noire : Deeptools

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --mem=30Gb
#SBATCH --time=8:00:00

module purge 2>/dev/null
module load mugqic/deepTools/3.5.0
cd $SLURM_SUBMIT_DIR

alignmentSieve -b ../../results/filterLowQuality/541_sorted_rmMT_Dupli_lowQ.bam \
--BED \
--ATACshift \
--blackListFileName mm10.blacklist.bed.gz \
-o ../../results/shiftAndRmblackList/541_bienFilter.bedpe \
```

```
--filterMetrics ../../results/shiftAndRmblackList/541_mstrics_log.txt

alignmentSieve -b ../../results/filterLowQuality/542_sorted_rmMT_Dupli_lowQ.bam \
--BED \
--ATACshift \
--blackListFileName mm10.blacklist.bed.gz \
-o ../../results/shiftAndRmblackList/542_bienFilter.bedpe \
--filterMetrics ../../results/shiftAndRmblackList/542_mstrics_log.txt

alignmentSieve -b ../../results/filterLowQuality/543_sorted_rmMT_Dupli_lowQ.bam \
--BED \
--ATACshift \
--blackListFileName mm10.blacklist.bed.gz \
-o ../../results/shiftAndRmblackList/543_bienFilter.bedpe \
--filterMetrics ../../results/shiftAndRmblackList/543_mstrics_log.txt

alignmentSieve -b ../../results/filterLowQuality/544_sorted_rmMT_Dupli_lowQ.bam \
--BED \
--ATACshift \
--blackListFileName mm10.blacklist.bed.gz \
-o ../../results/shiftAndRmblackList/544_bienFilter.bedpe \
--filterMetrics ../../results/shiftAndRmblackList/544_mstrics_log.txt

alignmentSieve -b ../../results/filterLowQuality/545_sorted_rmMT_Dupli_lowQ.bam \
--BED \
--ATACshift \
--blackListFileName mm10.blacklist.bed.gz \
-o ../../results/shiftAndRmblackList/545_bienFilter.bedpe \
--filterMetrics ../../results/shiftAndRmblackList/545_mstrics_log.txt

alignmentSieve -b ../../results/filterLowQuality/546_sorted_rmMT_Dupli_lowQ.bam \
--BED \
--ATACshift \
--blackListFileName mm10.blacklist.bed.gz \
-o ../../results/shiftAndRmblackList/546_bienFilter.bedpe \
--filterMetrics ../../results/shiftAndRmblackList/546_mstrics_log.txt
```

H. MACS2 callpeaks

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --mem=30Gb
#SBATCH --time=10:00:00

module purge 2>/dev/null
module load mugqic/python/2.7.14
module load mugqic/MACS2/2.1.1.20160309

cd $SLURM_SUBMIT_DIR

macs2 callpeak \
--bdg \
--gsize mm \
--keep-dup all \
-f BEDPE \
--outdir ../../results/macs2/ \
-n 541_callpeak \
-t ../../results/filterLowQuality/541_sorted_rmMT_Dupli_lowQ.bam \
&> 541_callpeak.sh.log

macs2 callpeak \
--bdg \
--gsize mm \
--keep-dup all \
-f BEDPE \
--outdir ../../results/macs2/ \
-n 542_callpeak \
-t ../../resultss/filterLowQuality/542_sorted_rmMT_Dupli_lowQ.bam \
&> 542_callpeak.sh.log

macs2 callpeak \
--bdg \
--gsize mm \
--keep-dup all \
```

```
-f BEDPE \
--outdir ../../results/macs2/ \
-n 543_callpeak \
-t ../../results/filterLowQuality/543_sorted_rmMT_Dupli_lowQ.bam \
&> 543_callpeak.sh.log

macs2 callpeak \
--bdg \
--gsize mm \
--keep-dup all \
-f BEDPE \
--outdir ../../results/macs2/ \
-n 544_callpeak \
-t ../../results/filterLowQuality/544_sorted_rmMT_Dupli_lowQ.bam \
&> 544_callpeak.sh.log

macs2 callpeak \
--bdg \
--gsize mm \
--keep-dup all \
-f BEDPE \
--outdir ../../results/macs2/ \
-n 545_callpeak \
-t ../../results/filterLowQuality/545_sorted_rmMT_Dupli_lowQ.bam \
&> 545_callpeak.sh.log

macs2 callpeak \
--bdg \
--gsize mm \
--keep-dup all \
-f BEDPE \
--outdir ../../results/macs2/ \
-n 546_callpeak \
-t ../../results/filterLowQuality/546_sorted_rmMT_Dupli_lowQ.bam \
&> 546_callpeak.sh.log
```

I. Genrich

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --mem=100G
#SBATCH --ntasks-per-node=5
#SBATCH --time=10:00:00

module purge 2>/dev/null
module load StdEnv/2020
module load mugqic/samtools/1.10
module load mugqic/java
cd $SLURM_SUBMIT_DIR

./Genrich -t ../541_sorted_byname.sam \
-o ../541_narrowPeak  \
-j \
-f ../541.log  -r  -x  -q 0.05  -a 20.0  -v  \
-e MT \
-E /home/wanlin/projects/def-makarenk/wanlin/atacseq/scripts/shiftAndRmblackList/mm10.blacklist.bed.gz

./Genrich -t ../542_sorted_byname.sam \
-o ../542_narrowPeak  \
-j \
-f ../542.log  -r  -x  -q 0.05  -a 20.0  -v  \
-e MT \
-E /home/wanlin/projects/def-makarenk/wanlin/atacseq/scripts/shiftAndRmblackList/mm10.blacklist.bed.gz

./Genrich -t ../543_sorted_byname.sam \
-o ../543_narrowPeak  \
-j \
-f ../543.log  -r  -x  -q 0.05  -a 20.0  -v  \
-e MT \
-E /home/wanlin/projects/def-makarenk/wanlin/atacseq/scripts/shiftAndRmblackList/mm10.blacklist.bed.gz

./Genrich -t ../544_sorted_byname.sam \
-o ../544_narrowPeak  \
-j \
-f ../544.log  -r  -x  -q 0.05  -a 20.0  -v  \
-e MT \
-E /home/wanlin/projects/def-makarenk/wanlin/atacseq/scripts/shiftAndRmblackList/mm10.blacklist.bed.gz

./Genrich -t ../545_sorted_byname.sam \
-o ../545_narrowPeak  \
-j \
-f ../545.log  -r  -x  -q 0.05  -a 20.0  -v  \
-e MT \
-E /home/wanlin/projects/def-makarenk/wanlin/atacseq/scripts/shiftAndRmblackList/mm10.blacklist.bed.gz

./Genrich -t ../546_sorted_byname.sam \
-o ../546_narrowPeak  \
-j \
-f ../546.log  -r  -x  -q 0.05  -a 20.0  -v  \
-e MT \
-E /home/wanlin/projects/def-makarenk/wanlin/atacseq/scripts/shiftAndRmblackList/mm10.blacklist.bed.gz
```

J. Diffbind

making sample sheet

```
df_samples <- data.frame(     SampleID=c("prog541_Genrich","prog542_Genrich","prog543_Genrich","trans544_Genrich","trans545_Genrich","trans546_Genrich"),      Condition = c("progenitors_Genrich","progenitors_Genrich","progenitors_Genrich","transitory_Genrich","transitory_Genrich","transitory_Genrich"),      Replicate = c(1,2,3,1,2,3),     bamReads = c("bamfiles/541_sorted.bam","bamfiles/542_sorted.bam","bamfiles/543_sorted.bam","bamfiles/544_sorted.bam","bamfiles/545_sorted.bam","bamfiles/546_sorted.bam"),     Peaks = c("results_Genrich/541_narrowPeak","results_Genrich/542_narrowPeak","results_Genrich/543_narrowPeak","results_Genrich/544_narrowPeak","results_Genrich/545_narrowPeak","results_Genrich/546_narrowPeak"),
PeakCaller = c("bed","bed","bed","bed","bed","bed") )  df_samples

write.csv(df_samples,"our_samples_Genrich.csv", row.names = FALSE)
```

create Diffbind object

```
tam.peaks <- dba(sampleSheet=df_samples)

tam.peaks
```

consensus peak sets

```
olap.rate <- dba.overlap(tam.peaks, mode=DBA_OLAP_RATE) olap.rate

plot(olap.rate, xlab="Overlapping samples", ylab="Overlapping peaks", type="b")

consensus.peaks <- dba.peakset(tam.peaks, bRetrieve=TRUE) consensus.peaks[,0]

tam.counts <- dba.count(tam.peaks)

tam.counts
```

###plots for batch effect

```
plot(tam.counts)

dba.plotPCA(tam.counts,DBA_REPLICATE, label=DBA_CONDITION,labelSize=0.6)
```

###normalization default

```
dba.plotMA(tam.counts, bNormalized=FALSE, sub="Non-Normalized", contrast=list(progenitors_macs=tam.counts$masks$progenitors_Genrich, transitory_macs=tam.counts$masks$transitory_Genrich))

dba.plotMA(tam.counts, sub="Normalized (Default)", contrast=list(progenitors_macs=tam.counts$masks$progenitors_Genrich, transitory_macs=tam.counts$masks$transitory_Genrich))
```

###Default model

```
tam.model <- dba.contrast(tam.counts, reorderMeta=list(Condition="transitory_Genrich"))

tam.model
```

###Fitting and testing

```
tam.model <- dba.analyze(tam.model)

dba.show(tam.model,bContrasts=TRUE)
```

###obtaining differential sites

```
tam.db <- dba.report(tam.model) tam.db

sum(tam.db$Fold>0)

sum(tam.db$Fold<0)

write.table(tam.db, file = paste("report_diff_Genrich.csv"),sep="\t",row.names = FALSE)
```

###MA plot

```
dba.plotMA(tam.model)
```

###Volcano plot

```
dba.plotVolcano(tam.model)
```

###cluster heatmanp

```
plot(tam.model, contrast=1)
```

###PCA plot

```
dba.plotPCA(tam.model, contrast=1, label=DBA_CONDITION,labelSize=0.6)
```

###Reads concentration heatmap

```
hmap <- colorRampPalette(c("red", "black", "green"))(n = 13) readscores <- dba.plotHeatmap(tam.model, co
ntrast=1, correlations=FALSE,                    scale="row", colScheme = hmap)
```

K. Homer

```
#!/bin/bash
#SBATCH --account=def-makarenk
#SBATCH --output=%x.o%j
#SBATCH --error=%x.e%j
#SBATCH --mem=30G
#SBATCH --ntasks-per-node=3
#SBATCH --time=6:00:00

module purge 2>/dev/null
module load mugqic/homer/4.11

cd $SLURM_SUBMIT_DIR

annotatePeaks.pl \
../../results/macs3/541_callpeak_peaks.narrowPeak \
mm10 -gsize mm10 -cons -CpG \
-go ../../results/homer_annotatePeaks_Macs/541/gene_ontology \
-genomeOntology ../../results/homer_annotatePeaks_Macs/541/genome_ontology \
1> ../../results/homer_annotatePeaks_Macs/541_peaks_Macs.csv \
2> 541_macs_peaksAnnot.log

annotatePeaks.pl \
../../results/macs3/542_callpeak_peaks.narrowPeak \
mm10 -gsize mm10 -cons -CpG \
-go ../../results/homer_annotatePeaks_Macs/542/gene_ontology \
-genomeOntology ../../results/homer_annotatePeaks_Macs/542/genome_ontology \
1> ../../results/homer_annotatePeaks_Macs/542_peaks_Macs.csv \
2> 542_macs_peaksAnnot.log
```

```
annotatePeaks.pl \
../../results/macs3/543_callpeak_peaks.narrowPeak \
mm10 -gsize mm10 -cons -CpG \
-go ../../results/homer_annotatePeaks_Macs/543/gene_ontology \
-genomeOntology ../../results/homer_annotatePeaks_Macs/543/genome_ontology \
1> ../../results/homer_annotatePeaks_Macs/543_peaks_Macs.csv \
2> 543_macs_peaksAnnot.log

annotatePeaks.pl \
../../results/macs3/544_callpeak_peaks.narrowPeak \
mm10 -gsize mm10 -cons -CpG \
-go ../../results/homer_annotatePeaks_Macs/544/gene_ontology \
-genomeOntology ../../results/homer_annotatePeaks_Macs/544/genome_ontology \
1> ../../results/homer_annotatePeaks_Macs/544_peaks_Macs.csv \
2> 544_macs_peaksAnnot.log

annotatePeaks.pl \
../../results/macs3/545_callpeak_peaks.narrowPeak \
mm10 -gsize mm10 -cons -CpG \
-go ../../results/homer_annotatePeaks_Macs/545/gene_ontology \
-genomeOntology ../../results/homer_annotatePeaks_Macs/545/genome_ontology \
1> ../../results/homer_annotatePeaks_Macs/545_peaks_Macs.csv \
2> 545_macs_peaksAnnot.log

annotatePeaks.pl \
../../results/macs3/546_callpeak_peaks.narrowPeak \
mm10 -gsize mm10 -cons -CpG \
-go ../../results/homer_annotatePeaks_Macs/546/gene_ontology \
-genomeOntology ../../results/homer_annotatePeaks_Macs/546/genome_ontology \
1> ../../results/homer_annotatePeaks_Macs/546_peaks_Macs.csv \
2> 546_macs_peaksAnnot.log
```

L.

Annexe IV – pipeline partiel d’analyse Taiji

Voir : https://taiji-pipeline.github.io/static/other/taiji.html

SCRNA_Read_Input
SCRNA_Demultiplex
SCRNA_Make_Index
SCRNA_Align
SCRNA_Filter_Bam
SCRNA_Quantification
SCRNA_QC
ATAC_Read_Input
ATAC_Download_Data
ATAC_Make_Index
ATAC_Align
ATAC_Filter_Bam
ATAC_Remove_Duplicates
ATAC_Bam_To_Bed
ATAC_Get_Bed
ATAC_Merge_Bed
ATAC_Call_Peak
ATAC_Merge_Peaks
ATAC_Make_BigWig
ATAC_Compute_TE
ATAC_Compute_Peak_Signal
ATAC_Align_Stat
ATAC_QC
ATAC_Fragment_Size_Distr
ATAC_Find_TFBS_Prep
ATAC_Find_TFBS_Union
ATAC_Get_TFBS
ATAC_Gene_Count
ATAC_Make_Expr_Table
HiC_Read_Input
Create_Linkage_Prep
Create_Linkage
Compute_Ranks
Output_Ranks
RNA_Read_Input
RNA_Download_Data
RNA_Make_Index
RNA_Align
RNA_Quant
RNA_Convert_ID_To_Name
RNA_Make_Expr_Table
SCATAC_Filter_Bam

Annexe IV-documents Taiji

A. input.tsv

type	id	group	rep	path	tags	format
ATAC-seq	progenitors_	Progenitors	1	SRR1168754	PairedEnd	SRA
ATAC-seq	progenitors_	Progenitors	2	SRR1168754	PairedEnd	SRA
ATAC-seq	progenitors_	Progenitors	3	SRR1168754	PairedEnd	SRA
ATAC-seq	transitory_A	Transitory	1	SRR1168754	PairedEnd	SRA
ATAC-seq	transitory_A	Transitory	2	SRR1168754	PairedEnd	SRA
ATAC-seq	transitory_A	Transitory	3	SRR1168754	PairedEnd	SRA
RNA-seq	TexProg1	Progenitors	1	SRR8065107	GeneQuant	SRA
RNA-seq	TexProg2	Progenitors	2	SRR8065110	GeneQuant	SRA
RNA-seq	TexProg3	Progenitors	3	SRR8065114	GeneQuant	SRA
RNA-seq	TexTrans1	Transitory	1	SRR8065108	GeneQuant	SRA
RNA-seq	TexTrans2	Transitory	2	SRR8065117	GeneQuant	SRA
RNA-seq	TexTrans3	Transitory	3	SRR8065118	GeneQuant	SRA

B. config.yml

```
##############################################################################
# The meaning of each field is explained at:
# https://taiji-pipeline.github.io/documentation/options.html
##############################################################################

##############################################################################
# Minimal settings
##############################################################################

input: "input.tsv"

output_dir: "output/"

# Optional if `genome`, `annotation` and `motif_file` are set.
assembly: "mm10"

##############################################################################
# Advanced settings
##############################################################################
```

```
#genome: "/home/kai/genome/GRCh38/Sequence/genome.fa"

#annotation: "/home/kai/genome/GRCh38/Annotation/gencode.v25.annotation.gtf"

#motif_file: "/home/kai/motif_databases/cisBP_human.meme"

#genome_index: "/home/kai/genome/GRCh38/Sequence/GRCh38.index"

#bwa_index: "/home/kai/genome/GRCh38/Sequence/BWAIndex/genome.fa"

#star_index: "/home/kai/genome/GRCh38/Sequence/STAR_index/"

#rsem_index: "/home/kai/genome/GRCh38/Sequence/RSEM_index/genome"

#callpeak_fdr: 0.01

#callpeak_genome_size: "hs"

#tss_enrichment_cutoff: 5

################################################################################
# Settings for job schedulers
################################################################################

# Example settings for SGE

#submit_params: "-q glean"
#submit_command: "qsub"
#submit_cpu_format: "-l nodes=1:ppn=%d"
#submit_memory_format: "-l mem=%dG"

# Example settings for slurm

#submit_params: "-p glean"
#submit_command: "sbatch"
#submit_cpu_format: "--ntasks-per-node=%d"
#submit_memory_format: "--mem=%d000"
```

```
################################################################################
# Computing resource settings
################################################################################

#resource:
#  SCATAC_Remove_Duplicates:
#    parameter: "-q home -l walltime=24:00:00"

#  SCATAC_Merged_Reduce_Dims:
#    parameter: "-q home-ren -l walltime=24:00:00"
#    cpu: 4
#    memory: 80
```