Université de Nantes

UFR Sciences et Techniques

Pipeline d'analyse de données de ChIP-Seq

INSERM U892 -Equipe 11, Laboratoire d'Hématologie - UMGC

Victor Gaborit
Master 2 Bioinformatique

June 30, 2016

Contents

1 Présentation

Ce document est rédigé de façon à expliquer et détailler les différents outils utilisé au cours du stage pour l'analyse de données de ChIP-Seq pour des facteurs de transcription et des marques fines d'histones. Dans chaque partie, un pipeline sera présenté avec son utilisation théorique ainsi que dans des exemples concrets. Les différents outils utilisés pour la mise en place du pipeline seront également présentés.
L'ensemble des pipelines présentées par la suite à été testé sur des données de ChIP-Seq dans un environnement Linux.
L'ensemble des pipelines et des outils téléchargés sont disponibles sur le github :
https://github.com/victor2410/ChIPpipe
La documentation sur les différents outils existants utilisés peut être trouvée ici :

- fastqc documentation : http://www.bioinformatics.babraham.ac.uk/projects/fastqc/
- Trimmomatic documentation : http://www.usadellab.org/cms/?page=trimmomatic
- samtools documentation : http://samtools.sourceforge.net/
- bowtie2 documentation : http://bowtie-bio.sourceforge.net/bowtie2/index.shtml
- picard-tools documentation : https://broadinstitute.github.io/picard/
- python documentation : https://www.python.org/
- spp documentation : http://compbio.med.harvard.edu/Supplements/ChIP-seq/
- macs2 documentation : https://github.com/taoliu/MACS
- R documentation : https://www.r-project.org/other-docs.html
- ChromHMM documentation : http://compbio.mit.edu/ChromHMM/
- ENCODE IDR analyse : https://sites.google.com/site/anshulkundaje/projects/idr

2 Etape 1 - Installation des outils requis

2.1 Outils nécessaires pour pour le pipeline ChIPpipe

- Installation de fastQC (v0.10.1), requise pour trimQual

```
sudo apt-get install fastqc
```

- trimmomatic-0.35 est déja présent dans le dossier ChIPpipe/Scripts et est requis pour trimQual
- Installation de Samtools (v1.3.1), requise pour ChIPalign, CallPeaks et CallPeaks_norep

```
sudo apt-get install samtools
```

- Installation de Bowtie2 (v2.1.0), requise pour ChIPalign

```
sudo apt-get install bowtie2
```

- Installation de picard-tools (v1.95), requise pour ChIPalign

```
sudo apt-get install picard-tools
```

- Installation de python (¡=v2.7), requise pour le pipeline ChIPpipe

```
sudo apt-get install python
```

- Installation de R (v3.0.2), requise pour CallPeaks et CallPeaks_norep

```
sudo apt-get install r-base
sudo apt-get install r-base-core
```

- Installation de spp (v1.10.1), requise pour CallPeaks et CallPeaks_norep

```
cd path/to/ChIPpipe/Scripts
R CMD INSTALL spp_1.10.1.tar.gz
```

- Installation de macs2 (v2.1.1), requise pour CallPeaks_norep

```
sudo apt-get install macs2
```

2.2 Installation du pipeline ChIPpipe

Après avoir installé les outils nécessaires, déplacez vous dans le dossier ChIPpipe et lancer l'installation via python avec la commande suivante :

```
# se deplacer dans le dossier ChIPpipe:
cd chemin/vers/ChIPpipe
# exemple
cd /home/stage/ChIPpipe
# installation
sudo python setup.py build
sudo python setup.py install
```

Ensuite, modifiez le fichier .bashrc de façon à rajouter la variable d'environnement RCHIPpipe_PATH:

```
# se deplacer dans son repertoire racine
cd $HOME
# ouvrir le fichier .bashrc dans un editeur de texte
gedit .bashrc
# ajouter a la fin du fichier la ligne suivante:
# export RCHIPpipe_PATH = "/fullpath/to/ChIPpipe/Scripts"
# enrigsitrer et fermer l'editeur puis sur le terminal:
source ~.bashrc
```

2.3 Autres outils requis pour le déroulement de ce pipeline

- Installation de java:

```
sudo apt-get install default-jre
sudo apt-get install default-jdk
```

- Installation de chromHMM:
  Télécharger l'archive ChromHMM : http://compbio.mit.edu/ChromHMM/ChromHMM.zip et décompresser l'archive

3 Etape 2 - Analyse des FASTQ et Trimming

3.1 Description

La première étape lors de l'analyse de données de ChIP-Seq est le contrôle qualité et la filtration des reads (trimming) dans les fichiers fastq. Le fichier fastq est le fichier obtenu directement après séquençage des données.
Cette étape est impérative afin de s'assurer que les données utilisées sont de bonnes qualité afin de réaliser une bonne analyse (néanmoins cette étape est optionnelle si ces contrôle on déjà été effectués avant ce qui est le cas avec les données d'Active motif).

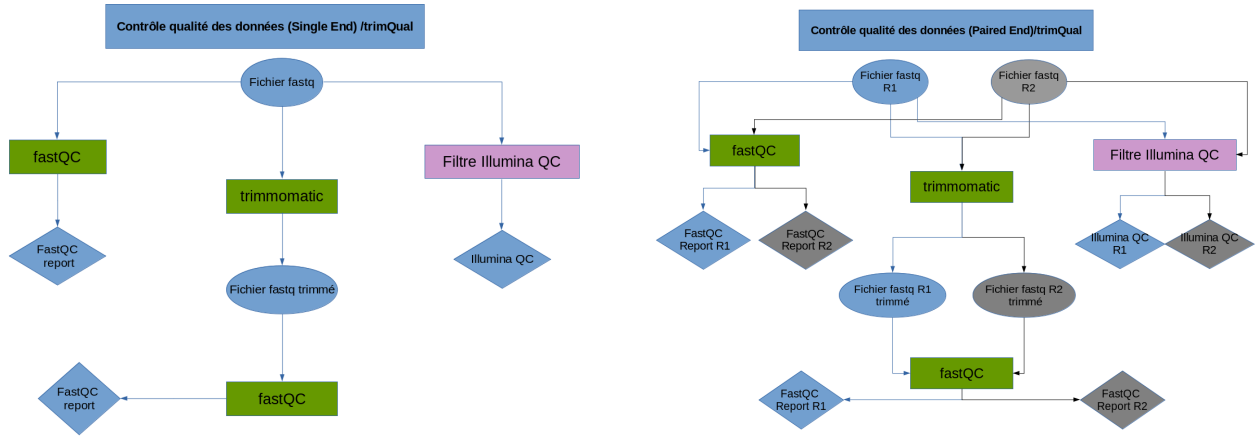
3.2 Pipeline d'analyse

Le pipeline à été écrit de façon à pouvoir être utilisé sur des données Single End (SE) ou Paired End (PE) en fonction du séquençage (Figure 1), deux fichiers fastq seront nécessaires (R1 et R2) pour le trimming de données Paired End. et les amorces pour les PCR seront également différentes selon la technique de séquençage (Hi-Seq ou autre) et le type de séquençage (SE ou PE). Ce pipeline se déroule en trois étapes et fait partie du gros pipeline ChIPpipe:

- Contrôle qualité des fichiers fastq (données brutes) tout juste sorties du séquenceur. On utilisera l'outil fastQC.
- Trimming (filtration) des données pour améliorer la qualité du fichier fastq. On utilisera trimmomatic.

- Contrôle qualité des fichiers fastq filtrés (permet de s'assurer que les données ont été correctement nettoyés). On utilisera à nouveau fastQC.

De façon optionnelle, on peut également contrôler le nombre de reads ayant échoués à passer le contrôle qualité du séquenceur (deux cluster overlappant par exemple).



(a) (b)

Figure 1: Pipeline trimQual pour le contrôle qualité et le trimming des fichiers FASTQ (a) Single End (b) Paired End

3.3 Utilisation

liste des options possibles pour ce pipeline :

- -f ou -1 et -2: Le nom et le chemin complet du fichier fastq à analyser (-f signifie que les données sont Single End, -1 et -2 pour des données Paired End), si -1 et -2 sont spécifiés, -1 est le fichier R1 et -2 le fichier R2. (REQUIS)
- –lib: librairie d'adaptateurs utilisée par le séquençeur (0 pour Illumina Genome analyzer et 1 pour Hi Seq 2000). (REQUIS)
- -o : Répertoire dans lequel inclure les fichiers sortis par ce pipeline (par défaut, le programme crée un nouveau répertoire "trimQual_out" dans le répertoire courant). (OPTIONNEL)
- –adapt: force le programme à rechercher des adaptateurs PE ou SE (par défaut il recherchera des adaptateurs SE si -f est spécifié et PE si -1 et -2 sont spécifiés). (OPTIONNEL)

Usage :

```
#Lancer la commande
ChIPpipe trimQual
#Affichage de l'usage de l'outil
TrimQual, quality check and trimming tools for ChIP-seq fastq files
Usage : ChIPpipe trimQual [-f path/file.fastq(.gz)> | -1 <path/fileR1.fastq(.gz)> -2 <path
    /fileR2.fastq(.gz)>] --lib INT [options]
  -h, --help   : print this usage message

REQUIRED ARGUMENTS
```

```
  -f FILE : full path and name of single end fastq file to analze (could be .fastq or .
    fastq.gz)
    or
  -1 FILE_R1 -2 FILE_R2 : full path and name of fastq file R1 (-1) and R2 (-2) for paired
    end data (could be .fastq or .fastq.gz)
\--lib INT  : select the adaptator library following used sequencer (0: Illumina Genome
    Analyzer IIx ; 1: Hi Seq 2000)

OPTIONNAL ARGUMENTS
  -o DIRECTORY  : output directory in which put all output files (default create
    trimQual_out in current directory)
  --adapt <PE|SE> : Sequencing library used for adaptators (default depend on -f (SE) or
    -1 (PE) options)
```

Exemple d'utilisation :

```
ChIPpipe trimQual -f /media/stage/Data_E/Victor/Stage/ENCODE/FOXM1/Fastq/ENCFF000NWE_trim.
    fastq.gz --lib 1 -o /media/stage/Data_E/Victor/Stage/ENCODE/FOXM1/trimQual_out/ --
    adapt SE
```

3.4 Résultats

Le programme crée deux sous-répertoires :

- fastqc_Report: Contient tous les fichiers crées par fastQC pour les fichiers fastq donnés. Ce répertoire contient donc un répertoire par fastq analysé qui contient lui même le fichier "fastqc_report.html" qui permet d'ouvrir une page internet présentant les résultats(Figure 2).

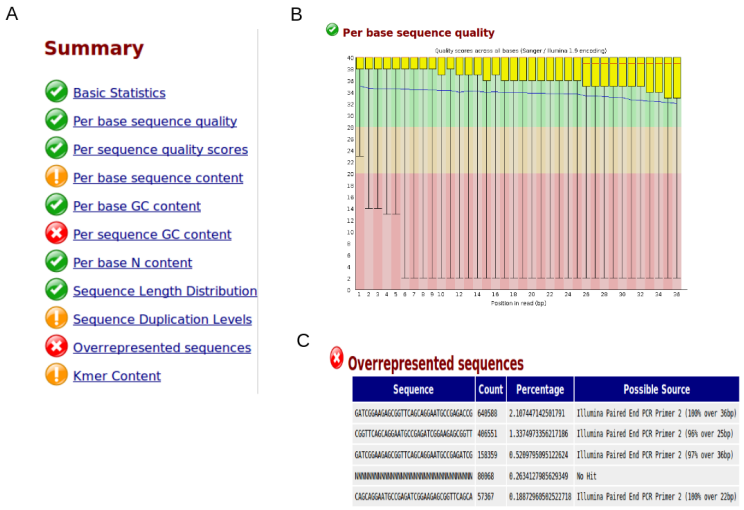


Figure 2: Exemple d'output donné par fastQC : (A) résumé des différents contrôles effectués et leur succès ou échec; (B) Exemple de contrôle qualité ayant réussi (score par base); (c) Exemple de contrôle qualité ayant échoué (séquences surreprésentées)

- fastq_trim: Contient les fichiers fastq filtrés par trimmomatic. Si les données sont single end, il n'y aura qu'un seul fichier (nomFastq_trimm.fastq.gz), si elles sont paired end, il y aura quatre fichiers fastq (un pour les reads conservés R1 et R2 et un pour les reads non conservés R1 et R2).

4 Etape 3 - Alignement des reads contre un génome de référence

4.1 Description

L'étape suivante consiste à aligner les reads du fichier fastq contre un génome de référence. Pour se faire, on utilisera une autre fonctionnalité du pipeline ChIPpipe : ChIPalign.

4.2 Pipeline d'analyse

Le pipeline à été écrit de façon à pouvoir être utilisé sur des données single end ou paired end en fonction du séquençage, deux fichiers fastq seront nécessaires (R1 et R2) pour l'alignement de données paired end. Ce pipeline se déroule en quatre étapes :

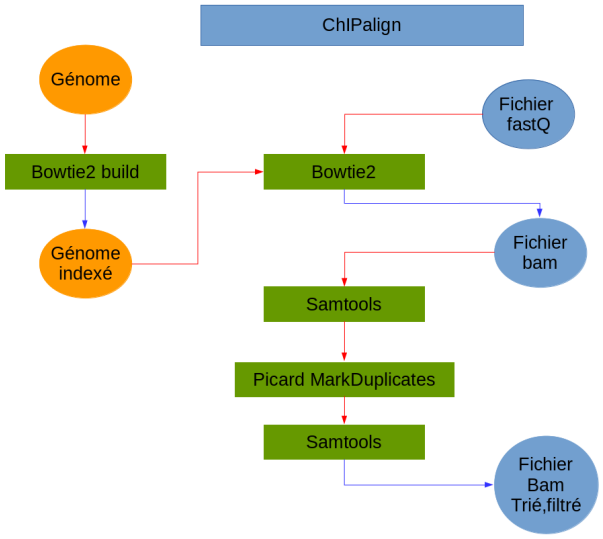


Figure 3: Pipeline *ChIPalign* crée pour aligner contre un génome de référence et filtrer les données de ChIP-Seq

- Indexation du génome avec Bowtie2.
- Alignement des reads contre le génome de référence avec Bowtie 2.
- Filtration des données avec Samtools et Picard .
- Tri et indexation du fichier aligné avec Samtools

Le but de ce pipeline est de sélectionner les différents filtres à appliquer au fichier d'alignement selon les critères désirés.

4.3 Utilisation

liste des options possibles pour ce pipeline :

- -f ou -1 et -2: Le nom et le chemin complet du fichier fastq à analyser (-f signifie que les données sont single end, -1 et -2 pour des données paired end), si -1 et -2 sont spécifiés, -1 est le fichier R1 et -2 le fichier R2. (REQUIS)

- -g: Préfixe du fichier fasta contenant les séquences de tous les chromosomes contre lesquels aligner les reads. ne pas mettre l'extension .fa ou .fasta. (REQUIS)
- -o: Répertoire dans lequel inclure les fichiers sortis par ce pipeline (par défaut, le programme crée un nouveau répertoire "ChIPalign_out" dans le répertoire courant). (OPTIONNEL)
- –index: Indexation nécessaire du génome pour l'alignement mais peut être effectuée avant de façon seule. (OPTIONNEL)
- -q: seuil minimum de mapping Quality qu'un reads doit avoir pour être conservé. (OPTIONNEL)
- -F: filtre les reads non mappés sur le génome. (OPTIONNEL)
- -L: filtre les reads situés dans les coordonnées d'un fichier (par exemple dans des régions à exclure type répétées). (OPTIONNEL)
- –rmdup: Enlève les duplicats de PCR et les reads alignés plusieurs fois (ne garde que le meilleur alignement). (OPTIONNEL)
- –sort: tri le fichier d'alignement en fin de filtration. (OPTIONNEL)
- –bamIndex: Indexation du fichier d'alignement. (OPTIONNEL)
- –name: nommer par un préfixe les fichiers de sorties

Usage :

```
#Lancer la commande
ChIPpipe ChIPalign
#Affichage de l'usage de l'outil
ChIPalign, ChIP-seq tool for alignment and filtration of reads
Usage : ChIPpipe ChIPalign [-f <path/file.fastq(.gz)> | -1 <path/fileR1.fastq(.gz)> -2 <
    path/file2.fastq(.gz)>] -g <path/GenomeDirectory/prefix> [options]
  -h, --help  : print this usage message

REQUIRED ARGUMENTS
  -f FILE : full path and name of single end fastq file to analyze(must be .fastq or .
    fastq.gz)
    or
  -1 FILE1 -2 FILE2 : full path and name of fastq file for reads 1 (-1) and reads 2 (-2)
    for paired end datas to analyze(must be .fastq or .fastq.gz)
  -g GENOMEPREFIX : full path and prefix of file containing all sequnces from reference
    chromosomes of genome used(without extension .fa or .fasta) (exemple : hg19_AllChr)

OPTIONNAL ARGUMENTS
  -o OUTPUTDIRECTORY  : full path and name of directory in wich writes all output files(
    default create a new repositorie in the current directory)
  --index : indexing genome files, must be select if the genome have not been indewed
    before(defaut : OFF)
  -q INT : reads filtering according to the minimum mapping quality specified
  -F : filter out unmapped reads
  -L FILE : remove reads mapped in coordinate files given(blacklist)
  --rmdup : remove PCR duplicates and non unique mappable reads
  --sort : sort final bam file
  --bamIndex : indexing final bam file (requiring --sort)
  --name NAME : prefix to give to output files
```

Exemple d'utilisation :

```
ChIPpipe ChIPalign -f /media/stage/Data_E/Victor/Stage/ENCODE/FOXM1/Fastq/ENCFF000NWE_trim
    .fastq.gz -g /media/stage/Data_E/Victor/Stage/Genome/allChr -o /media/stage/Data_E/
    Victor/Stage/ChIPalign_out --index -q 30 -F -L /media/stageData_E/Victor/Stage/
    blacklist.bed --rmdup --sort --bamIndex
```

4.4 Résultats

Le programme va produire deux fichiers Bam par Fastq : un premier qui sera le résultats de l'alignement et de la filtration et un autre qui contiendras les reads alignés dans les régions à exclure. En plus de cela, si l'option –bamIndex à été spécifiée, le script va crée un fichier .bam.bai qui correspondra au fichier bam indexé.

5 Etape 4a - PeakCalling avec utilisation de réplicats

5.1 Description

L'étape suivante consiste à aligner effectuer la recherche de pics correspondant à la fixation du facteur de transcription étudié. Si des réplicats biologiques ont étés réalisés pour les données, on utiliseras le protocole mis en place par ENCODE utilisant l'analyse IDR pour effectuer cette recherche de pics. Ce protocole est integré dans le pipelin ChIPpipe : CallPeaks.

5.2 Pipeline d'analyse

Le pipeline à été écrit de façon à suivre le protocole de PeakCalling d'encode, de ce fait deux fichiers fastq seront nécessaires (réplicat1 et réplicat2) ainsi qu'au moins un (maximum deux) fichier de contrôle appelé aussi fichier d'input. Ce pipeline se déroule en six étapes :

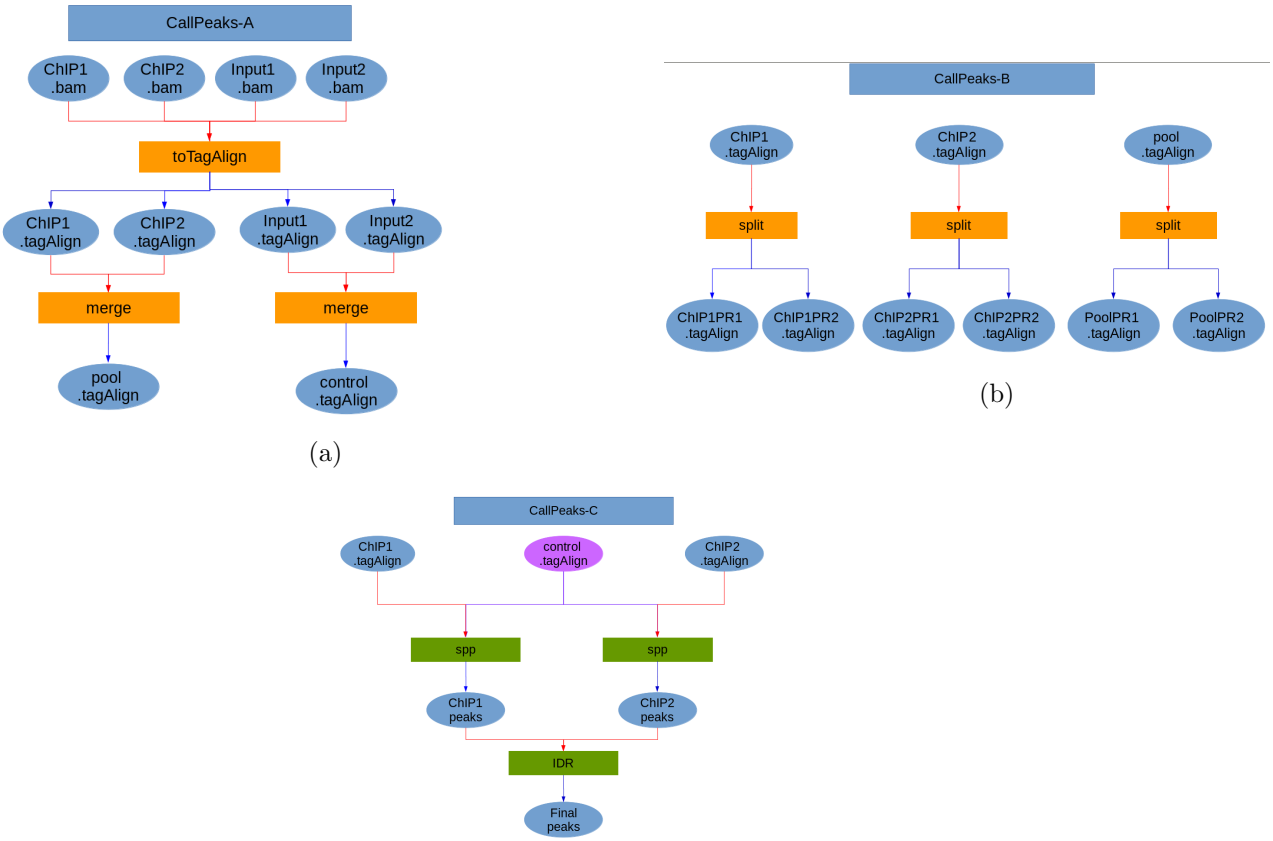


Figure 4: Pipeline CallPeaks pour la recherche de pics sur des fichiers d'alignement; (a) transformation des fichiers et création du pool de fichier ChIP et de fichier contrôle; (b) séparation aléatoire de chaque fichier en pseudo-réplicats; (c) entre chaque réplicat (ou pseudo-réplicat), on effectue la recherche de pics séparément (un réplicat versus le contrôle) puis on effectue l'analyse IDR entre les deux réplicats

- Transformation des fichiers d'alignement .bam en fichier .tagAlign (format demandé pour l'analyse)
- Création du pool de réplicats et si deux fichiers d'Input sont donnés, on les regroupes ensemble également

- On sépare aléatoirement chaque fichier de réplicat et le pool en deux pseudo-réplicats
- Pour chaque réplicats, le pool et chaque pseudo-réplicats, on effectue la recherche de pics versus le contrôle (pool Input) avec spp
- on effectue l'analyse IDR entre chaque réplicats et chaque pseudo-réplicats
- on sélectionne dans les pics obtenus du pool les pics qui correspondent à un seuil IDR fixé

5.3 Utilisation

liste des options possibles pour ce pipeline :

- -1 et -2: Le nom et le chemin complet des fichiers bam à analyser, -1 est le fichier réplicat1 et -2 le fichier réplicat2 (REQUIS)
- –c1: Le nom et le chemin complet du fichier bam pour l'Input (REQUIS)
- –c2: Le nom et le chemin complet du deuxième fichier bam pour l'Input si il existe (OPTIONNEL)
- -o: Répertoire dans lequel inclure les fichiers sortis par ce pipeline (par défaut, le programme crée un nouveau répertoire "ChIPalign_out" dans le répertoire courant). (OPTIONNEL)
- –idr: effectuer l'analyse IDR après la recherche de pics(OPTIONNEL)
- –sets: créer les sets de pics finaux (conservatifs et optimum) d'après l'analyse IDR(OPTIONNEL)
- –no-plots: Ne pas produire les graphes de résultats de l'analyse IDR(OPTIONNEL)
- –name: nommer par un préfixe les fichiers de sorties

Usage :

```
#Lancer la commande
ChIPpipe CallPeaks
#Affichage de l'usage de l'outil
Usage : ChIPpipe CallPeaks -1 <path/fileRep1.bam> -2 <path/fileRep2.bam> --c1 <path/
    fileControl.bam> [options]
  -h, --help  : print this usage message
REQUIRED ARGUMENTS
  -1 FILE_REP1  : full path and name of replicate 1 file (bam file)
  -2 FILE_REP2  : full path and name of replicate 2 file (bam file)
  --c1 FILE_CONTROL :  full path and name of control file (bam file)
OPTIONNAL ARGUMENTS
  --c2 FILE_CONTROL :  full path and name of second control file (if there is one) (bam
    file)
  -o <path/directory> : full path to directory in wich write all output files (default
    create a CallPeaks_out directory in the current directory)
  --idr FLOAT : Perform IDR analysis on output peakCalling files with a specified
    threshold (0.01 or 0.02 for transcription factor) (default : OFF)
  --sets  : Create final peak sets (conservative and optimum) corresponding to IDR
    threshold (require --idr option) (default :OFF)
  --no-plots  : Do not plot IDR results (default : ON)
  --name NAME : prefix to give to output files (default is CallPeaks)
```

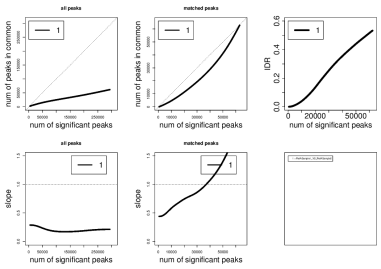
Exemple d'utilisation :

```
ChIPpipe CallPeaks -1 /media/stage/Data_E/Victor/Stage/ENCODE/FOXM1/Alignement/
    ENCFF000NWE_rep1.bam -2 /media/stage/Data_E/Victor/Stage/ENCODE/FOXM1/Alignement/
    ENCFF000NWE_rep2.bam --c1 /media/stage/Data_E/Victor/Stage/ENCODE/FOXM1/Alignement/
    Input.bam -o /media/stage/Data_E/Victor/Stage/CallPeaks_out --idr 0.01 --sets --name
    FOXM1
```

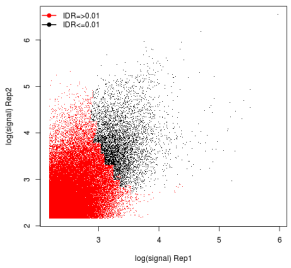
5.4 Résultats

Cet outil va créer quatre dossiers différents :

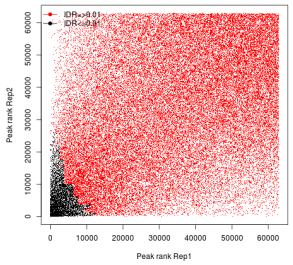
- Un dossier tagAlign qui va contenir tous les fichiers utilisés dans ce format (réplicats, pseudo-réplicats, pool...)
- Un dossier PeakCalling qui va contenir tous les fichiers issus de la recherche de pics par spp (y compris les graphes de cross-corrélation)
- Un dossier IDR (si l'option a été sélectionnée) qui va contenir tous les fichiers produit par cette analyse et un dossier qui va contenir tout les graphes de l'IDR (si l'option –no-plots n'a pas été utilisée)
- Un dossier finalsets (si l'option –sets à été utilisée) qui contient les deux sets de pics finaux (conservatif et optimum)



(a)



(b)



(c)

Figure 5: Graphes résultants de l'analyse IDR; (a) IDR globale; (b) log(signal.value) par rapport au seuil IDR; (c) rank(signalvalue) par rapport au seuil IDR

6 Etape 4b - PeakCalling sans utilisation de réplicats

6.1 Description

Cette étape ne doit être effectué que dans le cas où l'on ne dispose pas de réplicats biologiques. Si des réplicats biologiques ont étés réalisés, il est conseillé d'utiliser l'étape 4a. On utilisera une autre méthode pour effectuer la recherche de pics. Ce protocole est integré dans le pipelin ChIPpipe : CallPeaks_norep.

6.2 Pipeline d'analyse

Le pipeline à été écrit de façon à suivre le protocole de PeakCalling d'encode, de ce fait deux fichiers fastq seront nécessaires (réplicat1 et réplicat2) ainsi qu'au moins un (maximum deux) fichier de contrôle appelé aussi fichier d'input. Ce pipeline se déroule en trois étapes :

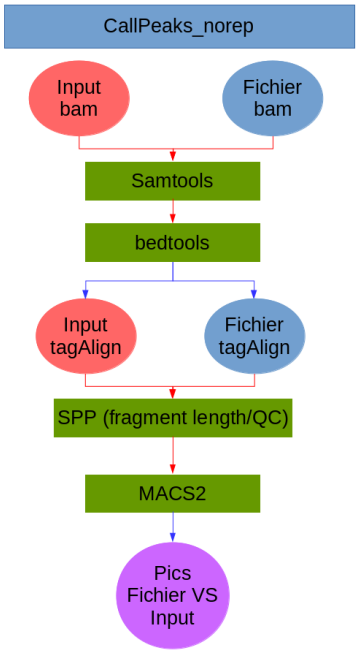


Figure 6: Pipeline CallPeaks_norep pour la recherche de pics sur des fichiers d'alignement sans réplicats biologiques disponibles

- Transformation du fichier d'alignement .bam en fichier .tagAlign (format demandé pour l'analyse)
- Contrôle qualité des données avant recherche de pics et estimation des paramètres de cette recherche avec spp
- Recherche de pics pour le fichier d'alignement versus le fichier de contrôle (Input)

6.3 Utilisation

liste des options possibles pour ce pipeline :

- -f: Le nom et le chemin complet du fichiers bam à analyser (REQUIS)

- -c: Le nom et le chemin complet du fichier bam pour l'Input (REQUIS)
- -o: Répertoire dans lequel inclure les fichiers sortis par ce pipeline (par défaut, le programme crée un nouveau répertoire "ChIPalign_out" dans le répertoire courant). (OPTIONNEL)
- –thresh: p-value seuil utilisée pour effectuer la recherche de pics(OPTIONNEL)
- –nomodel: estimer les paramètres de la recherche de pics via spp plutôt que de laisser macs2 construire le model(OPTIONNEL)
- –name: nommer par un préfixe les fichiers de sorties

Usage :

```
#Lancer la commande
ChIPpipe CallPeaks_norep
#Affichage de l'usage de l'outil
CallPeaks_norep, PeakCalling for sample without biological replicates related on MACS2
Usage : ChIPpipe CallPeaks_norep -f <path/file.bam>-c <path/control.bam> [options]
  -h, --help  : print this usage message
REQUIRED ARGUMENTS
  -f FILE : full path and name of alignment file to analyze (bam format)
  -c FILE : full path and name of Input alignment file to use (bam format)
OPTIONNAL ARGUMENTS
  -o OUTPUTDIRECTORY  : full path and name of directory in wich writes all output files (
    default create a new repositorie in the current directory)
  --thresh STR  : pvalue threshold for peak calling (ex : 1e-7 ; default = 1e-3)
  --nomodel : using estimated fragment length by phantomPeakQualtools for peakCalling than
     macs2 prediction model
  --name NAME : prefix to give to output files (default is CallPeaks_macs)
```

Exemple d'utilisation :

```
ChIPpipe CallPeaks_norep -f /media/stage/Data_E/Victor/Stage/ENCODE/FOXM1/Alignement/
    ENCFF000NWE.bam -c /media/stage/Data_E/Victor/Stage/ENCODE/FOXM1/Alignement/Input.bam
    -o /media/stage/Data_E/Victor/Stage/CallPeaks_norep_out --thresh 1e-7 --no-model --
    name FOXM1
```

6.4 Résultats

Cet outil va créer deux dossiers:

- Un dossier tagAlign qui va contenir tous les fichiers utilisés pour la recherche de pics au format .tagAlign (fichier ChIP-seq et Input)
- Un dossier PeakCalling qui va contenir tous les fichiers issus de la recherche de pics par macs2 (y compris les graphes de cross-corrélation produits par spp si l'option –no-model à été utilisée)

7 Etape 5 - Annotation des pics obtenus

7.1 Description

Pour effectuer l'annotation des pics pour des facteurs de transcription, nous avons besoin de données de marques d'histones afin de pouvoir annoter les différentes régions du génome (enhancer, promoteurs, région transcrites...). Ces marques d'histones sont spécifiques de la lignée cellulaire dans laquelle le ChIP-Seq a été réalisé. Cette étape se déroule en deux parties:

- L'annotation des régions sur le génome qui ne peut pas être automatisée car elle dépend de l'interprétation biologique de l'utilisateur mais ne devra être utilisée qu'une seule fois (certaines lignées cellulaires ont des données de ChromHMM publiques disponibles sur ucsc)
- L'annotation des pics obtenus lors des étapes précédentes au niveau des régions enhancers et promoteurs qui est inclue dans le pipeline d'analyse ChIPpipe : AnnoPeaks

7.2 Pipeline d'analyse

7.3 Utilisation de ChromHMM

Pour utiliser ChromHMM, il faut les fichiers d'alignement (bam) des marques d'histones spécifiques de la lignée cellulaire à étudier. Elle peuvent être produites par séquençage néanmoins, certaines marques sont disponibles sur ENCODE ou encore sur Blueprint. Dans notre cas, cinq marques d'histones ont été utilisées pour ChromHMM: H3K4me1, H3K4me3, H3K27ac, H3K27me3 et H3K36me3. L'utilisation de ChromHMM se fait en trois étapes :

- Binarisation des fichiers bam avec BinarizeBam qui requiert les éléments suivants:
  - Le dossier contenant les fichiers d'alignement (bam) à utiliser
  - Un fichier CellMark.txt qui est un fichier tabulé de quatre colonnes qui contiennent:
    1-le nom de la lignée cellulaire
    2-le nom de la marque d'histones
    3-le nom du fichier d'alignement correspondant
    4-le nom du fichier d'Input associé
  - Un fichier Chromsize.txt tabulé de deux colonnes qui contiennent:
    1-le nom du chromosome (ex: chr1)
    2-la taille du chromosme
  - Le dossier dans lequel écrire les fichiers de sorties

```
java -mx1600M -jar ChromHMM.jar BinarizeBam chemin/vers/dossiersBams/ CellMark.txt
    Chromsize.txt chemin/dossierSortie
```

- Apprentissage du modèle de Markhov avec LearnModel :
  - -p pour spécifier le nombre de processeurs à allouer à ChromHMM (0 signifie en utiliser autant que possible)
  - le dossier dans lequel ont été mis les fichiers bam binarisés
  - le dossier de sortie dans lequel mettre les résultats de la commande

– le nombre d'état que l'on souhaite rechercher (ici pour seulement 5 marques de disponible, 10 états est un maximum)

– le génome utilisé (ici hg19)

```
java -mx1600M -jar ChromHMM.jar LearnModel -p 0 chemin/DossierSortieBinarizeBam
    chemin/DossierSortieLearnModel 10 hg19
```

- L'étape d'après consiste en une importante interprétation biologique des résultats. Pour définir les états obtenus et les annoter correctement, nous allons utiliser essentiellement trois graphes ainsi que les connaissances biologiques:

  – Les paramètres d'émission

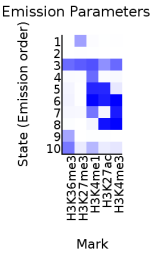


Figure 7: Paramètre d'émission du modèle produit par ChromHMM

  – L'enrichissement spécifiques de régions du génome

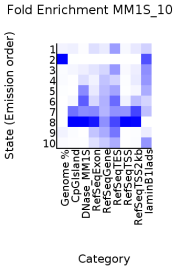


Figure 8: Enrichissement de régions selon les états produit par ChromHMM

  – La distribution des états autour du site de début de transcription

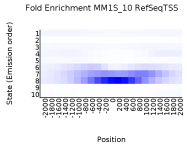


Figure 9: Distribution des états autour du TSS produit par ChromHMM

- Une fois chaque état associé à une annotation, on crée le fichier final d'annotation avec MakeBrowserFile qui requiert les arguments suivants:
  - l'option -c permet de spécifier un fichier pour donner la couleur à chaque états dans une genome browser il doit être composé de deux colonnes délimitées par une tabulation:
    1-le numero de l'état
    2-le code r,g,b de la couleur que l'on souhaite associer à l'état
  - l'option -m permet de spécifier un fichier pour nommer chaque états, il doit être composé de deux colonnes délimités par une tabulation:
    1-L'identifiant de l'état (ex: E1)
    2-L'annotation correspondante (ex: Promoteur_Actif)
  - l'option -n pour spécifier le nombre d'états utilisés
  - le fichier NomCellule_NbEtat_segment.bed qui peut être trouvé dans le dossier dans lequel ont été mis les fichiers de LearnModel
  - le préfixe à donner aux fichier de sortie

```
java -mx1600M -jar ChromHMM.jar MakeBrowserFile -c colorfile -m labbelfile -n
    nombreEtats segmentfile outputprefix
```

A la fin de ceci nous obtenons un fichier Prefix_dense.bed qui contient les positions et les annotations de tous les états. C'est ce fichier que nous allons utiliser ensuite pour notre pipeline

7.4 AnnoPeaks

7.5 Résultats

8 Exemple d'utilisation : Analyse de la sous-unité RelA du facteur nucléaire NF-κB

8.1 Etape1

Pour cette analyse, l'installation de tous les outils présentés dans l'étape 1 est requise

8.2 Etape2

Du fait que les fichiers fastq ont été produits par Active motif et que les contrôle qualité des reads et filtration des reads à été effectuée par Active Motif, cette étape n'est pas faite ici

8.3 Etape3

fichiers nécessaire :

- Le fichier fastq de séquençage pour RelA
- Un fichier fasta comprenant l'ensemble des séquences du génome de référence contre lequel on veut aligner les reads. Le génome de référence hg19 a été utilisé et seulement les séquences des chromosomes 1 à 22 et X et Y ont été utilisés. Les fichiers fasta pour chaque chromosomes peuvent être téléchargés sur le site ucsc. Ensuite on crée un fichier de référence globale.

```
# Pour telecharger les fichiers depuis le terminal:
for i in `seq 1 22`; do
wget 'ftp://hgdownload.cse.ucsc.edu/goldenPath/hg19/chromosomes/chr'$i'.fa.gz' -O "/home/stage/Genome/chr"$i".fa.gz";
done
wget 'ftp://hgdownload.cse.ucsc.edu/goldenPath/hg19/chromosomes/chrX.fa.gz' -O "/home/stage/Genome/chrX.fa.gz"
wget 'ftp://hgdownload.cse.ucsc.edu/goldenPath/hg19/chromosomes/chrY.fa.gz' -O /home/stage/Genome/chrY.fa.gz
# Regrouper les tous les chromosomes dans un seul fichier fasta:
cd /home/stage/Genome
for i in `ls`; do
zcat $i >> allChr.fa;
done
```

- le fichier consensus des régions blacklist définies par ENCODE qui est disponible ici à dézipper et à placer dans un dossier accessible

Commande utilisée:

```
ChIPpipe ChIPalign -f /home/stage/fastq/RelA.fastq.gz -g /home/stage/Genome/allChr --index -o /home/stage/Alignement -q 30 -F -L /home/stage/datas/consensusBlacklist.bed --rmdup --sort --bamIndex --name RelA
```

8.4 Etape4b

Il n'y a pas de réplicats biologiques de disponibles pour cette sous-unité, nous allons donc utiliser le pipeline CallPeaks_norep pour effectuer la recherche de pics.
fichiers nécessaires:

- Le fichier d'alignement pour RelA obtenu à l'étape précédente
- Le fichier d'Input aligné de la même façon que RelA qui a été fournit pour cette analyse

Commande utilisée:

```
ChIPpipe CallPeaks_norep -f /home/stage/Alignement/RelA.bam -c /home/stage/Alignement/
    Input.bam -o /home/stage/CallPeaks_norep_out --thresh 1e-7 --nomodel --name RelA_peaks
```

8.5 Etape5