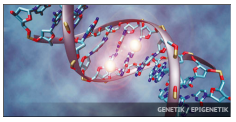
Computational methods for epigenome analysis (COMPEPIWS 2022)
Prof. Dr. Jörn Walter, Jun-Prof. Dr. Fabian Müller
Nihit Aggarwal , Gilles Gasparoni
Kathrin Kattler, Abdulrahman Salhab, Midhuna Maran, Sarath Kumar
Walter Lab & Müller Lab @ Saarland University



ChIP-seq

We are going to use the nf-core/chipseq pipeline that was introduced in the general task sheet. In the first part you will run the pipeline, then you will assess the quality of the raw and processed data and explore the results in the genome browser IGV. At the end you will use a segmentation tool to define chromatin states across the genome.

1 Nextflow ChIP-seq pipeline

⚠ Make sure you run this task on the master node inside a screen
⚠ We limited the Nextflow Java virtual machines memory into NXF_OPTS='-Xms200m -Xmx300m'. Do NOT change this.

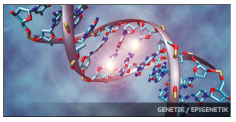
1. **Organize your raw data** it is a good practice to have the raw data placed in one folder:
   a) Enter your working directory `/vol/COMPEPIWS/groups/<group>/tasks`
   b) List the raw fastq files under `/vol/COMPEPIWS/data/reduced/ChIP-seq/`
   c) How many files per cell_type per time point per replicate are there?
   d) What are the histone marks you are dealing with?
   e) how many "control" per replicate do you have?
   f) Create a "data" folder in your group working directory and create symbolic links of the **relevant files according to your group** to the "data" folder as follow:
      - chipseq1: kidney samples
      - chipseq2: liver samples
      - chipseq3: kidney samples

2. **Generate the samplesheet** The pipeline is dependent on a samplesheet as input. Examples and details are in the documentation.
   a) Enter the task folder
   b) prepare the samplesheet. You can prepare it manually in any editor of your choice, but it is recommended to generate it on the command line taking the advantage of the file names, this is particularly useful when you have many samples. You can loop over the files and extract the information you need to create the samplesheet (hint: basename, cut).

```
for i in /vol/COMPEPIWS/data/reduced/ChIP-seq/*H3*gz
do
    fastq1=$i
    group=$(basename ${i}|cut -f 1-3 -d _)
    replicate=.........
    mark=........
    control=.........
    echo "$group,$replicate,$fastq1,,$mark,$control" (maybe replace this line with
    ``if'' clause to not print the 5,6th columns when the the group is control)
done > samplesheet.csv
```

   Don't forget to add the header to the samplesheet

3. **Running the pipeline** As you might noticed from the documentation, you can provide parameters on the command line or add them to a configuration file which you pass to (-c) option. We have prepared a config file for you, have a look to the parameters in /vol/COMPEPIWS/pipelines/configs/chipseq.config. We will use singularity as a profile.
   a) based on the config file:



i. is the data paired end or single end?
ii. which steps of the pipelines are skipped?
iii. why is the blacklist file used? What do these regions represent?
iv. which tool will be used for mapping?

b) Load Conda environment:

```
source /vol/COMPEPIWS/conda/miniconda3/bin/activate /vol/COMPEPIWS/conda/
miniconda3/envs/core
```

c) Run the pipeline (will take around 2 hours to complete):
⚠ Please talk to your mentor to check the samplesheet before you start running the pipeline

```
nextflow run nf-core/chipseq -r 1.2.2 -profile singularity -c
/vol/COMPEPIWS/pipelines/configs/chipseq.config --input samplesheet.csv
```

4. watch your jobs using *squeue --user=$ USER*.
5. The output folder will have the name "results"
6. List in order the main steps of the pipeline for processing ChIP-seq data?

2 Quality Control

In this part we will explore the data quality (raw and processed) using fastqc, MultiQC report generated by the pipeline and deepTools2 [1] software. But first we will get an idea about technical details related to the pipeline output.

2.1 Exploring the result folder

Explore the different sub-directories in the `results` folder. You will use most of them for downstream analyses.

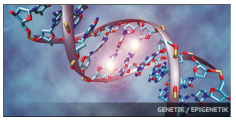
1. Have a look into the `pipeline_info/execution_report.html`:
   - How long did the pipeline take to finish?
   - Which step took the most CPU/Memory usage?
2. Where are the aligned reads located (*bam files)? You will use these files later
3. Where are the peaks files located (*narrowPeak)? which tool was used for peak calling?
4. Where are the consensus peaks located (*consensus_peaks.bed)?
5. Where are the signal tracks located (*bigWig)? You will use these files later

2.2 MultiQC

We will now inspect data quality of the input sequence data based on the MultiQC report generated by the pipeline.

1. Explore the different section of the MultiQC HTML report generated by the pipeline. You find it under `results/multiqc/narrowPeak` folder.
2. In the "LIB: FASTQC (raw)" section:
   - How many reads do you have for each sample?
   - How good is the quality of the bases in the reads?

[1] https://deeptools.readthedocs.io/en/develop/



- How good is the quality of the reads?
- What are the read lengths for the samples?
- Is there a sign for adapter contamination?
- compare the number of reads after trimming? do you loose many reads after trimming?

3. In the "MERGED LIB: SAMTools (unfiltered)":
   - How is the mapping efficiency?
   - What is the duplication rate?
4. In the "MERGED LIB: MACS2 PEAK COUNT" section:
   - Do you get the similar number of peaks for all histone marks?
   - Overall: which marks get the highest/lowest number of peaks?
5. In the "MERGED LIB: MACS2 FRIP SCORE" section:
   - What does FRiP score mean?
   - which marks get the highest/lowest FRiP scores? How is it related to the mark signal distribution (narrow/broad)?

2.3 deepTools2

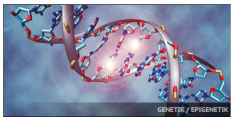
⚠ Make sure you run this task on one of the worker nodes
deepTools has different quality control tools that can be used to evaluate the ChIP-seq data quality. We will use some of them (read about the tools from the Documentation).

1. Activate the `core` environment.
2. Explore the batch effect between your samples (hint: multiBigwigSummary, plotPCA, plotCorrelation, recognize the different parameters of each tool)?
   - PCA plot might not be very informative due to the large number of samples, try to plot it in R with ggplot. There is a ready file to be read in R if you provide "--outFileNameData" option to "plotPCA" function.
3. Based on the plotCorrelation outputs, which marks are more similar to each other.
4. How well are the signals (marks) separated from the background. Since you have many samples it would be wise to do each histone mark of all samples separately. i.e, one plot for all H3K4me3, another for H3K4me1 ..... etc (hint: plotFingerprint)
5. Choose one of the replicates of one stage of one cell type and plot profiles of the 7 histone marks around TSS (+-1500bp) and across the gene body (+-1500bp). Use the gene file produced by your job under `results/genome/genome_genes.bed` (hint: plotProfile, plotHeatmap). Which marks are enriched at TSS and which are enriched at the gene body?

3 Exploratory Analysis using IGV

⚠ Make sure you run this task on one of the worker nodes
IGV is a genome browser that can view different types of data. Like coverage tracks (bigwig) and genomic regions (bed).

1. IGV is installed in /vol/COMPEPIWS/softwares/IGV_Linux_2.10.2/.
2. Start IGV by `bash /vol/COMPEPIWS/softwares/IGV_Linux_2.10.2/igv.sh`
3. Open the IGV session from `results/igv/narrowPeak/igv_session.xml`. It might be a bit slow. After loading, remove the consensus tracks.



4. Upload the gene tracks from `results/genome/genome_genes.bed`
5. Do you see clear peaks for all marks? which marks are sharp and which ones are broad(er)?
6. You can color the tracks as you want (e.g. per mark). Right click on the track name
7. Realize the local distribution of the different histone marks. Which one(s) overlap with TSS and which of them ovelap with gene body?
8. Do you always see an agreement between the called peaks of the two replicates of the same mark in the same cell type? To get a quantitative answer have a look to the Upset plots under `results/bwa/mergedLibrary/macs/narrowPeak/consensus/<mark>/*.boolean.intersect.plot.pdf`. What do you understand from this plot (choose two of the sharp marks)

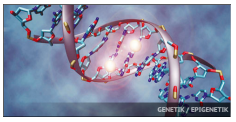
For integrative analysis later, your colleagues will need some data (signal and bam files) produced by your group. In order to keep consistency between the groups we envisage the following folder structure in "/vol/COMPEPIWS/groups/shared/ChIP-seq" folder where all the groups have access to. Make symbolic links to the corresponding sub-directories (make sure your files are protected from being overwritten by others before you do the linking):

```
/vol/COMPEPIWS/groups/shared/ChIP-seq
                                    |
                                    |__ chipseq1
                                    |     |__ alignments (bam files)
                                    |     |__ signals (bigwig files)
                                    |     |__ peaks (narrowPeak files)
                                    |     |__ segmentation (chromHMM runs, later for task 4.2)
                                    |
                                    |__ chipseq2
                                    |     |__ alignments (bam files)
                                    |     |__ signals (bigwig files)
                                    |     |__ peaks (narrowPeak files)
                                    |     |__ segmentation (chromHMM runs, later for task 4.2)
                                    |
                                    |__ chipseq3
                                          |__ alignments (bam files)
                                          |__ signals (bigwig files)
                                          |__ peaks (narrowPeak files)
                                          |__ segmentation (chromHMM runs, later for task 4.2)
```

4 Chromatin segmentation with ChromHMM

After aligning the reads and exploring the coverage tracks of the different histone marks across the different samples, you must have realized that the data is too big to be explored and find something interesting only using a genome browser. We will integrate the histone mark data into one tool called ChromHMM [2] [3] to define chromatin states (i.e., chromatin segmentation). To generate genome wide segmentation using ChromHMM we need to run two commands; BinzrizeBam and LearnModel. Refer to ChromHMM manual [4] for the detailed command usage. In this step you will use the bam files (aligned reads).

[2] https://www.nature.com/articles/nmeth.1906
[3] https://www.nature.com/articles/nprot.2017.124
[4] http://compbio.mit.edu/ChromHMM/ChromHMM_manual.pdf



4.1 ChromHMM Installation

We have installed and prepared references for ChromHMM in /vol/COMPEPIWS/softwares/ChromHMM/. So you do not need to install it. Use directly the jar file inside this path. The chromosome size file is also available in this directory "CHROMSIZES".

4.2 Running ChromHMM

1. Activate Core conda environment.
2. Create a folder called *segmentation* under your work group folder, and enter it
3. Create a folder called "inputs" and create symbolic links of all aligned read files to this folder (there should be 64 bam files).
4. To run BinarizeBam you need to prepare a tab delimited file "cellmarkfiletable.txt". Prepare this file in such away that the replicates of one stage per cell type are pooled together. i.e at the end we want to have one segmentation result per cell type per stage: liver_14.5/15.5 and kidney_14.5/15.5. Use the control samples in "cellmarkfiletable.txt" file to adjust the binarization threshold (hint: loop over the bam files, extract the cell type and histone mark names from the file name).
5. Run the **BinarizeBam** command with 200bp as a bin size (be careful in selecting the right chromosom-Lengthfile) ⚠ Make sure you run this task on one of the worker nodes

```
java -Xmx3000M -Djava.awt.headless=true -jar \
/vol/COMPEPIWS/softwares/ChromHMM/ChromHMM.jar BinarizeBam \
-b <binSize> \
/vol/COMPEPIWS/softwares/ChromHMM/CHROMSIZES/mm10.txt \
inputs \
cellmarkfiletable.txt \
output_folder_name
```

6. Run the **LearnModel** command with different number of states: 5-15 (submit the commands as jobs (sbatch) to SLURM, refer to the generalTask 4.4): ⚠ Make sure you run this task on one of the master node

   `Example: sbatch --job-name=job_name -o job_$state.out --wrap="Your learnModel command"`

7. Explore the output files called webpage_$state.html. Especially, emission heatmap, RefSeq_TSS_neighborhood and overlap figures. Comment on the "new" states you get.
8. Compare the different models to assess what is the proper minimum number of states. Use the model with the highest number of states as a reference (hint: CompareModels function). What is the minimum reasonable number of states you think is appropriate?
9. The "cellmarkfiletable" in step 5 was prepared to produce 4 segmentation files (1 per stage per cell type). Repeat step 5 in a way that you merge the stages of each cell type in order to produce only 2 segmentation files (1 for each cell type).
10. Repeat steps 6 with the new "cellmarkfiletable". Do it only with 15 states model
11. Based on the emission heatmap, overlap and enrichment of this new 15-state model, try to assign biological labels to the states. Use figure 2 as reference and use the shortcuts from table 1. In case you couldn't find the proper annotation from figure 2 try to be creative and give a label by yourself.
12. Rename the states in column 4 of the files *dense.bed (or *segments.bed) using the biological states from the previous question (consider the command *MakeBrowserFiles* from ChromHMM or *awk* in bash to do the task). **Note:** Redirect the output into new files in a subfolder called "relabeled" to avoid overwriting the original files.

state_name	label
Weak promoter	Pr_W
Flanking promoter	Pr_F
Active promoter	Pr_A
Bivalent promoter	Pr_B
Active Enhancer	Enh_A
Poised Enhancer	Enh_P
Weak Enahncer	Enh_W
Strong, Tss-distal Enhancer	Enh_ds
Transcription Initiation	Tx_I
Weak Transcription	Tx_W
Strong Transcription	Tx_S
Mixture	Mix
Weak Heterochromatin	Het_W
Strong Heterochromatin	Het_S
PolyComb	Het_P
No signal	NS

Figure 1: States and labels

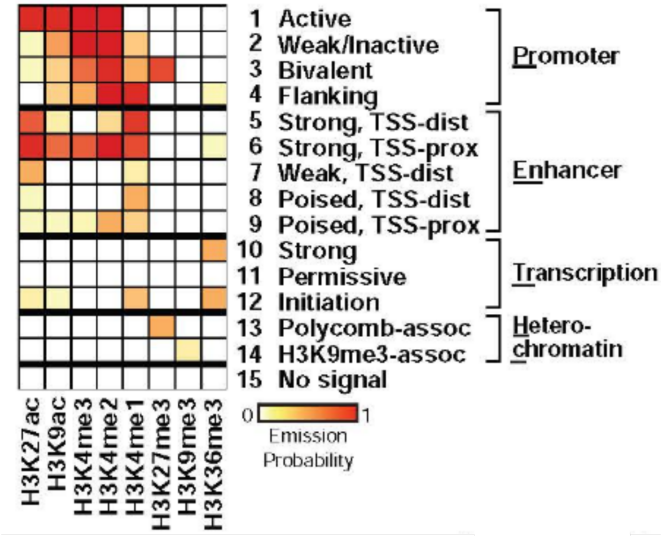


Figure 2: Reference ChromHMM

13. Run "reorder" command to generate the emission heatmap with the new labels.
14. Copy the relabeled segmentation files of Q12 and the emission heatmap of Q13 into the shared folder under your group sub-directory `/vol/COMPEPIWS/groups/shared/<group>/segmentation`

5 Exploration of chromatin state segmentation using IGV

Upload the coverage files of one replicate (7 bigwigs) of one cell type and the corresponding relabeled segmentation file (15 states) *dense.bed in IGV and start browsing the data.

1. Which state(s) is(are) more likely overlapping with TSS?
2. Upload the segmentation of the other cell type (one replicate). Can you pin point some differential states between the two cell types? save some snapshots.

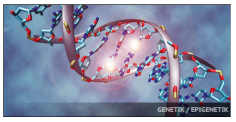
6 Exploratory Analysis using R

In this section we will explore the segmentation results of 15-state model (the relabeled) by looking to their length distributions, genome coverage percentages. We will do this in R and ggplot package for plotting. This task will teach how to visualize multiple layers of information in one plot. You will try to summarize all what you need in a data frame for a later easy plotting. The final data frame should look like:

```
chromosome   start   end   state     sample     length
chr18        12334   13034  2_Enh_A   liver       700
..           ....    .....  ......    .........  ...
..           ....    .....  ......    .........  ...
chr18        10250   11350  1_Het_P     kidney    1100
..           ....    .....  ......    .........  ...
..           ....    .....  ......    .........  ...
```

The steps 1-9 will help you to prepare such a table. However, feel free to do it in your own way.

1. Activate the “core” environment.
2. Start R



3. Read all *(reordered)dense.bed files from task 4.2.12 from "segmentation" directory results into R. (hint: use *list.files, lapply, read.table* functions)
4. Merge the results into one data frame (hint: *rbind* function).
5. Extract the sample names, they should be two unique names (hint: *strsplit*).
6. Make a vector of names with length equal to the data frame length and consider the different name numbers for each sample (hint: *for, rep*)
7. Add a sample name column to the data frame.
8. Calculate the segment length and add it as a new column to the data frame.
9. Calculate the segment percentage per sample (hint: melt, aggregate, dcast functions).
10. Plot and save the length distributions per sample (hint: ggplot() + geom_boxplot() + facet_wrap(), ggsave).
11. Plot and save the segment genomic percentage per sample as stacked bar plots (hint: ggplot() + geom_barplot(), ggsave).
12. Based on the plots from task 10 and 11, comment on the state distributions/lengths across the cell types and stages.