1 ChIP-Seq Project 1: Sequencing depth and Chromatin Environment

The following tasks have been adapted from materials developed by Angela Goncalves, Myrto Kostadima, Steven Wilder and Maria Xenophontos.

1.1 Sequencing depth

One of the most frequent questions that come up in ChIP-seq experiments is whether the sequencing depth is sufficient.

The more we sequence a ChIP-seq library, the more peaks of low fold change we will identify. Therefore, the only way to answer that question is to look for the number of peaks identified when we down sample our library. To test for sufficient sequencing depth in our sample we will down sample our ChIP and Control datasets to 10%, 20%, .., 90% of the initial library size and call peaks. To do so, we will use the functions **randsample** and **callpeak** from macs2, respectively.

First, go to the group_projects folder.

```
cd /home/manager/course_data/group_projects
```

Check to see if the ChIPSeq-Project1 folder exists.

```
ls ChIPSeq-Project1
```

If this folder doesn't exist, please check with your course instructor.

Once you have the data, go into the ChIPSeq-Project1 directory.

```
cd ChIPSeq-Project1
```

Below are a series of commands which will downsample the ChIP and Control datasets to 10%, 20%, .., 90% of the initial library size and call peaks. To do this we use a while loop which will start counting from 10 (`i=10`), in intervals of 10 (e.g. 10, 20, 30...) (`(((i = i + 10)))`) up to 100 (`$i -lt 100`). The value which is being counted is stored in a variable called `i` that is then referenced in other commands using `$i`. Finally, to make sure we know where we are in the loop, we will print a message in the terminal `echo "Looking at ${i}% of the reads"`.

The command we are using to downsample the ChIP and Control datasets reads is `macs2 randsample` which we will be giving an input file `-t`, a percentage `-p`, an output file `-o`, an output directory `--outdir` and the format of the input file `-f`.

Look at the usage for `macs2 randsample`.

```
macs2 randsample -h
```

We then call the peaks for each set of downsampled reads using `macs2 callpeak` in a similar way to the ChIP-Seq tutorial.

Type the commands below into a file called `project1.sh`.

```
i=10

        while [[ $i -lt 100 ]]
        do

        echo "Looking at ${i}% of the reads"

        macs2 randsample -t PAX5.bam -p $i -o PAX5.perc${i}.bed \
        -outdir macs2_downsample -f BAM

        macs2 randsample -t Control.bam -p $i -o Control.perc${i}.bed \
        -outdir macs2_downsample -f BAM

        macs2 callpeak -t macs2_downsample/PAX5.perc${i}.bed \
        -c macs2_downsample/Control.perc${i}.bed -gsize 138000000 \
        -format BED -name macs2_downsample/PAX5.perc${i} \
        -pvalue 1e-3 -call-summits

        ((i = i + 10))

        done
```

Now, make your script executable.

```
chmod u+x project1.sh
```

And run the script.

```
./project1.sh
```

Next, we want to plot the number of peaks called in each of the downsampled datasets. To do this we will use `R`. First we tell R where to find our output files using `setwd`. We then use a for loop to extract only the peaks whose overall enrichment meets a threshold value (`fc.thres <- 4` and `peaks <- peaks[peaks[, 7] > fc.thres, ]`). We then count those peaks (`no.peaks <- c(no.peaks, nrow(peaks))`). Finally, we plot the number of peaks vs the percentage, saving the output to `peaks_vs_percentage.jpg`.

Type the following commands into a file called `script.R`.

```
rm(list = ls())

        options(stringsAsFactors=F)
        setwd( "/home/manager/course_data/group_projects/ChIPSeq-Project1" )

        fc.thres <- 4

        no.peaks <- c()
        for(row in seq(from=10, to = 90, by = 10))
        {
                print(row)

                peaks <- read.table(paste("macs2_downsample/PAX5.perc",
            row, "_peaks.narrowPeak", sep=""))

                peaks <- peaks[peaks[, 7] > fc.thres, ]

                no.peaks <- c(no.peaks, nrow(peaks))
        }

        peaks <- read.table("PAX5_peaks.narrowPeak")
        peaks <- peaks[peaks[, 7] > fc.thres, ]
        no.peaks <- c(no.peaks, nrow(peaks))


        jpeg('peaks_vs_percentage.jpg')

        plot(seq(from=10, to = 100, by = 10), no.peaks,
        type="o", col="blue", xlab="Percentage of reads", ylab="Number of peaks"

        dev.off()
```

Use `Rscript` to run your R script.

```
Rscript script.R
```

Let's take a look at the plot.

```
eog peaks_vs_percentage.jpg
```

Q1: Do you think that we have sequenced enough?

Close the file when you have finished looking at the plot.

1.2 Chromatin Environment

The goal of this hands-on session is to investigate the chromatin environment around gene features and regulatory elements. We will be plotting where the different histone modifications occur in relation to genes. We will also look at how we can distinguish different genes based on transcription and chromatin environment.

1.2.1 Prepare environment

We will be using ENCODE GM12878 histone modifications. The BAM alignment files have been downloaded from http://www.encodeproject.org. All ENCODE experiments had at least 2 technical replicates, but we will only be using the first replicate for simplicity.

We will be using **ngsplot** https://github.com/shenlab-sinai/ngsplot to visualise these datasets at transcription start sites. The ngsplot database for the human gene set has been generated from the Ensembl http://www.ensembl.org and RefSeq http://www.ncbi.nlm.nih.gov/refseq/ gene sets, by default using the Ensembl set.

Make sure that the location where we want to write the database to is writable.

```
sudo chmod -R 777 /usr/local/bioinf-recipes/ngsplot-2.63
```

When prompted, type the password for manager which is `manager`.

Next, make sure that the following R libraries are installed.

```
sudo R
```

In R, type the following commands. When prompted, type `n` so that other packages are not updated.

```
source("https://bioconductor.org/biocLite.R")
BiocInstaller::biocLite(c("ShortRead", "BSgenome", "doMC"))
```

To quit R type `q()` and enter `n` when prompted.

Now, install the database using `ngsplotdb.py`.

```
echo 'Y' | ngsplotdb.py install ngsplotdb_hg19_75_3.00.tar.gz
```

Look at the usage for `ngs.plot.r`.

```
ngs.plot.r
```

Further help on the ngsplot options can be found at https://github.com/shenlab-sinai/ngsplot/wiki/ProgramArguments101.

The histone modifications we will be looking at today are all thought to play different roles in the regulation of gene expression and the putative functions, according to the ENCODE Project, are summarised in the table. We will look at them individually in this exercise, but later will be integrating multiple histone modifications.

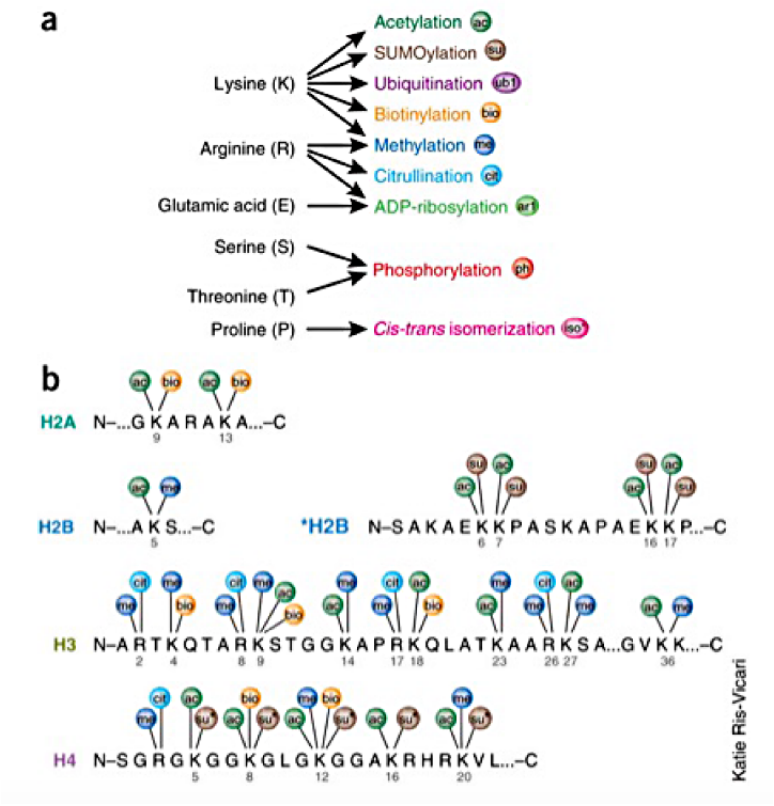

figure 1

Figure 1: The histone modifications are named by the histone tail, location in the protein sequence and the biochemical modification, e.g. H3K4me3, refers to the trimethylation of the lysine (K) in the fourth position in the protein sequence of histone 3 tail.

Histone modification or variant	Peak or Region	Putative functions
H2A.Z	Peak	Histone protein variant (H2A.Z) associated with regulatory elements with dynamic chromatin
H3K4me1	Peak/Region	Mark of regulatory elements associated with enhancers and other distal elements, but also enriched downstream of transcription starts
H3K4me2	Peak	Mark of regulatory elements associated with promoters and enhancers
H3K4me3	Peak	Mark of regulatory elements primarily associated with promoters/transcription starts
H3K9ac	Peak	Mark of activate regulatory elements with preference for promoters
H3K9me1	Region	Loosely associated with transcription, with preference for 5 end of genes
H3K9me3	Peak/Region	Repressive mark associated with constitutive heterochromatin, repetitive elements and certain broad repressive domains
H3K27ac	Peak	Mark of active regulatory elements; may distinguish active enhancers and promoters from their inactive counterparts
H3K27me3	Region	Repressive mark established by polycomb complex activity associated with repressive domains and silent developmental genes
H3K36me3	Region	Elongation mark associated with transcribed portions of genes, with preference for 3 regions after intron 1
H3K79me2	Region	Transcription-associated mark, with preference for 5 end of genes
H4K20me1	Region	Loosely associated with transcription, with preference for 5 end of genes

figure 2

Figure 2: A summary of the putative functions of various histone marks, according to the ENCODE Project.

1.2.2 Transcription Start Sites

It has been shown that the chromatin state, i.e. the combination of all regulatory variations at the chromatin level, in the neighbourhood of a gene has a large effect on its transcription. We will therefore start by looking at the genome-wide profiles of the selected histone marks around TSS.

Plot the distribution of H3K4me1 around protein-coding gene TSS (this may take a few minutes to run).

```
ngs.plot.r -G hg19 -O H3k4me1.tss -FL 150 -R cgi \
       -C H3k4me1.bam -T H3k4me1
```

Q2: You can now view the H3K4me1 aggregation plots and heatmaps in your folder. Where does the majority of the H3K4me1 signal appear in relation to TSS?

Q3: What proportion of genes display this pattern?

Repeat these plots for other histone modifications.

```
ngs.plot.r -G hg19 -O H3k4me3.tss -FL 150 -R tss \
       -C H3k4me3.bam -T H3k4me3
```

ngsplot also enables you to correct the ChIP sample using the control sample. This aims to correct for any genome biases in alignability, GC content etc.

Let's give this a try.

```
ngs.plot.r -G hg19 -O H3k4me3_Control.tss -FL 150 \
       -R tss -C H3k4me3.bam:Control.bam -T H3k4me3:Control
```

Note: If you get an error about the alignment not being sorted, try sorting your BAM file and using the sorted file instead.

You can now also plot the signal aroung specific annotated regions, such as Ensembl gene bodies. H3K36me3 is known to be enriched over transcribed genes.

Let's give this a try.

```
ngs.plot.r -G hg19 -O H3k36me3 -FL 150 -R genebody \
       -C H3k36me3.bam -T H3k36me3
```

Q4: Can you detect any patterns in this enrichment?

You can use configuration files in ngsplot to draw multiple graphs on the same plot. For instance, we will plot multiple histone modifications on the same plot, and also subset the genes, based on gene expression.

Using a text editor, create the following tab-separated file, called `multhist.txt` for drawing two histones modifications on the same plot. -1 corresponds to looking at the whole genome.

```
|-------------|----|---------|
| H3k4me1.bam | -1 | H3k4me1 |
| H3k4me3.bam | -1 | H3k4me3 |
```

Run the following command to plot the graphs:

```
ngs.plot.r -G hg19 -O H3k4me1.H3k4me3 -FL 150 -R tss \
       -C multhist.txt -T H3k4me1.H3k4me3
```

Q5: Are the patterns you observe consistent with the putative functions of the modification in the table given above?

1.2.3 Gene Expression

You will now use RNA-seq data form the same cell type results to separate all of the Ensembl genes into three categories, based on the expression (FPKM values) of the genes. The file containing the FPKMs for GM12878 whole cell RNA-seq is called `genes.fpkm_tracking`. These FPKM values have been computed by Cufflinks.

Read the first few lines of the GM12878 whole cell genes.fpkm_tracking file.

```
head -n 10 genes.fpkm_tracking
```

Remove the header line.

```
sed '1d' genes.fpkm_tracking > genes.fpkm.txt
```

Sort the genes by FPKM score.

```
sort -k10 -n -r genes.fpkm.txt > genes.fpkm.sorted.txt
```

Count the number of genes.

```
wc -l genes.fpkm.sorted.txt
```

Use `head`, `sed` and `awk` to pull out the Highest, Lowest and Intermediate expression genes.

```
head -n 5000 genes.fpkm.sorted.txt | \
       awk '{print $1}' > high_expressed_genes.txt
```

```
awk '{ if($10 ==0) {print $1} }' \
       genes.fpkm.sorted.txt > low_expressed_genes.txt
```

```
sed -n '10001,15000p' genes.fpkm.sorted.txt | \
       awk '{print $1}' > mid_expressed_genes.txt
```

Create a tab-delimited configuration file expression.txt in a text editor containing the following lines.

```
|--------------|--------------------------|--------|
| H3k27me3.bam | high_expressed_genes.txt | "High" |
| H3k27me3.bam | mid_expressed_genes.txt  | "Med"  |
| H3k27me3.bam | low_expressed_genes.txt  | "Low"  |
```

Then run the following command.

```
ngs.plot.r -G hg19 -R genebody -C expression.txt \
       -O H3k27me3.express.genebody -D ensembl \
       -FL 300 -T H3k27me3.expression
```

Q6: What patterns can you see?

If you made it to here, well done!