Day 9 Worksheet – Preprocessing ChIP-Seq Data

Author: Rutendo Sigauke

Introduction: This worksheet goes over how to preprocess ChIP-seq data prior to peak calling. ChIP-seq is an assay genome wide binding of protein to DNA, so the coverage profile is different from RNA-seq and as such the data needs to preprocessed differently. This worksheet will go over assessing quality of ChIP-seq data and mapping the reads to the genome. The tools we will use are the same for other genome sequencing data (RNA-seq, ATAC-seq), BUT the flags used will be different.

QC tools:

- *fastqc : Assess the read quality in samples.*
- *preseq : Get read complexity (asses how reads are distrubuted in the genome after mapping).*
- *multiqc : Summarizing all the QC metrics in a single document.*

Mapping reads:

- *hisat2 : Mapping reads to genome with ChIP-seq friendly commands.*

! Note: The directory and username used in the screenshot will be for my working directory and username and will be different than yours. Here we will be working on the server and editing the script in **`vim`**.

Make working directories

Make the necessary working directories for **`day9`** in your scratch folder.

1. Use command **`pwd`** to determine what directory you are in and if necessary, **`cd`** to the directory that you want to place your new directory.

- Use **`mkdir`** to generate **`day9`** directory and in that folder, generate a **`scripts`** directory (**`/scratch/Users/<username>/day9`** and **`/scratch/Users/<username>/day9/scripts`**).
- For this script, the sub-directories will be generated within the script, so the only folder you will generate manually is the **`scripts`** folder.

2. Copy **`d9_preprocessing_chipseq.sbatch`** to your **`scripts`** folder. You can copy the script using the **`rsync <input><output>`** or **`cp`** commands.

Quality Control

1. Edit the sbatch script by using **`vim <script>`** to open a text editor on your sbatch script. Type **i** to toggle into edit/insert mode.

a. Similar to the previous exercise you will need to change the job name, user email, and the standard output and error log directories. Change the **`–job-name=<JOB ID>`** to a name related to the job you will be running, for example 'chip_qc'. Additionally you will want to change the **`–mail- user=<YOUR_EMAIL>`** to your email, as well as the path to your eofiles directory for the standard output (**`--output`**) and error log (**`--error`**). The **`%x`** will be replaced by your **`-job- name`** and the **`%j`** will be replace by the job id that will be assigned by the job manager when you run your sbatch script.

```
#!/bin/bash

#SBATCH --job-name=<JOB ID>                                    # Job name
#SBATCH --mail-type=ALL                                       # Mail events (NONE, BEGIN, END, FAIL, ALL)
#SBATCH --mail-user=<email>                                   # Where to send mail
#SBATCH --nodes=1                                             # Numbers of nodes
#SBATCH --ntasks=4                                            # Number of CPU (tasks)
#SBATCH --time=00:25:00                                       # Time limit hrs:min:sec
#SBATCH --partition=compute                                   # Partition/queue requested on server
#SBATCH --mem=20mb                                           # Memory limit
#SBATCH --output=/scratch/Users/<username>/eofiles/%x_%j.out
#SBATCH --error=/scratch/Users/<username>/eofiles/%x_%j.err
```

b. Change the working directory **`WD`** in the sbatch script. Within this working directory, other directories will be generated in this script (**`BAM`**, **`SAM`** and **`QC`**).

- After changing the **`WD`** and the top header of the script, we can submit the job. Please continue to the following sections for details on running each tool and summaries of the output.

```
################################################################################################
################ SET REQUIRED VARIABLES ########################################################
################################################################################################

INDIR=/scratch/Shares/public/sread2022/data_files/day9/fastq
WD=/scratch/Users/<username>/day9  <----------
BAM=${WD}/bam
SAM=${WD}/sam
QC=${WD}/qc
```

2. The next section loads the necessary modules for running the script. Just to reiterate, these programs are **`fastqc`**, **`preseq`**, **`hisat2`** and **`samtools`**.

```
################################################################################
############## LOAD REQUIRED MODULES ###########################################

module load fastqc/0.11.5
module load preseq/2.0.3
module load hisat2/2.1.0
module load samtools/1.8
```

3. We also run **`fastqc`** on ChIP-seq data to get a summary of read quality per each sample in our analysis. The output is a report for the read quality for each sample, and these will be located in the **`${QC}/fastqc`** folder.

! Note: For this example, will process more than one sample at a time. Therefore, we process the samples using a **`for loop`** in bash as shown below. This will process all the **`FILENAME`** sample in the **`$INDIR`** directory one at time. In the in-class example we will only process **`'BACH1'`** samples and select for them using **`grep`**.

```
################################################################################################
################## RUN JOB #####################################################################

#########################################################
#Use fastqc to check read quality                       ##
#########################################################
## the fastq files will be used as input to fastqc.
## output will be a fastqc file used to assess quality

mkdir -p ${QC}/fastqc

for FILENAME in `ls $INDIR | grep 'BACH1'`; do

    fastqc \
        ${INDIR}/${FILENAME} \
        -o ${QC}/fastqc

done
```

4. Next, we will align reads to the reference genome using **`hisat2`**. The main difference between mapping ChIP-seq reads to the genome is that we do not have to use the splice alignment. This feature is turned off using **`--no-spliced-alignment`** flag. The alignment output is **`bam`** files and alignment summary (reported if **`--new-summary`** flag is used).

! Note: The map statistics are being ouputted in to the **`QC`** folder (**`${QC}/hisat_mapstats`**), while the bam files go into the **`BAM`** folder.

```
############################################################
#align reads to reference genome using HISAT2          ##
############################################################
#here we are mapping reads to the genome, but since this is chip-seq
#we will ignore splicing aware features for the mapper
mkdir -p ${QC}/hisat_mapstats

for FILENAME in `ls $INDIR | grep 'BACH1' | tr '.' '\t' | cut -f 1`; do

    hisat2 -p 4 \
        --very-sensitive \
        --no-spliced-alignment \
        -x ${INDICES} \
        -U ${INDIR}/${FILENAME}.fastq \
        --new-summary \
        > ${SAM}/${FILENAME}.sam \
        2> ${QC}/hisat_mapstats/${FILENAME}.hisat2_mapstats.txt

    ###convert SAM to BAM
    samtools view -@ 4 -h -bS -o ${BAM}/${FILENAME}.bam ${SAM}/${FILENAME}.sam

    ###sort BAM file
    samtools sort -@ 4 ${BAM}/${FILENAME}.bam > ${BAM}/${FILENAME}.sorted.bam

    ###index sorted bam file
    samtools index ${BAM}/${FILENAME}.sorted.bam  ${BAM}/${FILENAME}.sorted.bam.bai

    ###detele intermediate and large files
    ###i.e. the unsorted bam file and the sam file
    rm ${BAM}/${FILENAME}.bam
    rm ${SAM}/${FILENAME}.sam

done
```

5. Once the alignment is complete, we can assess mapped read distribution on the genome using **`preseq`**. Preseq estimates the complexity of a library and it also estimates how many additional unique reads are sequenced with an increasing total read count.

! Note: The output is going into the **`QC`** folder as well (**`${QC}/preseq`**).

```
############################################################
#check for complexity of files                            ##
############################################################
#this program takes as input all your mapped bam files

mkdir -p ${QC}/preseq

for FILENAME in `ls $BAM | grep 'BACH1' | grep -v 'bai'`; do

    # generate the complexity curve
    # -s is the step size (for larger samples the default is 1million)
    preseq c_curve -B -s 10000 \
        -o ${QC}/preseq/${FILENAME}.c_curve.txt \
        ${BAM}/${FILENAME}

    # predict the complexity curve of a sequencing library using an initial experiment
    preseq lc_extrap -B \
        -o ${QC}/preseq/${FILENAME}.lc_extrap.txt \
        ${BAM}/${FILENAME}

done
```

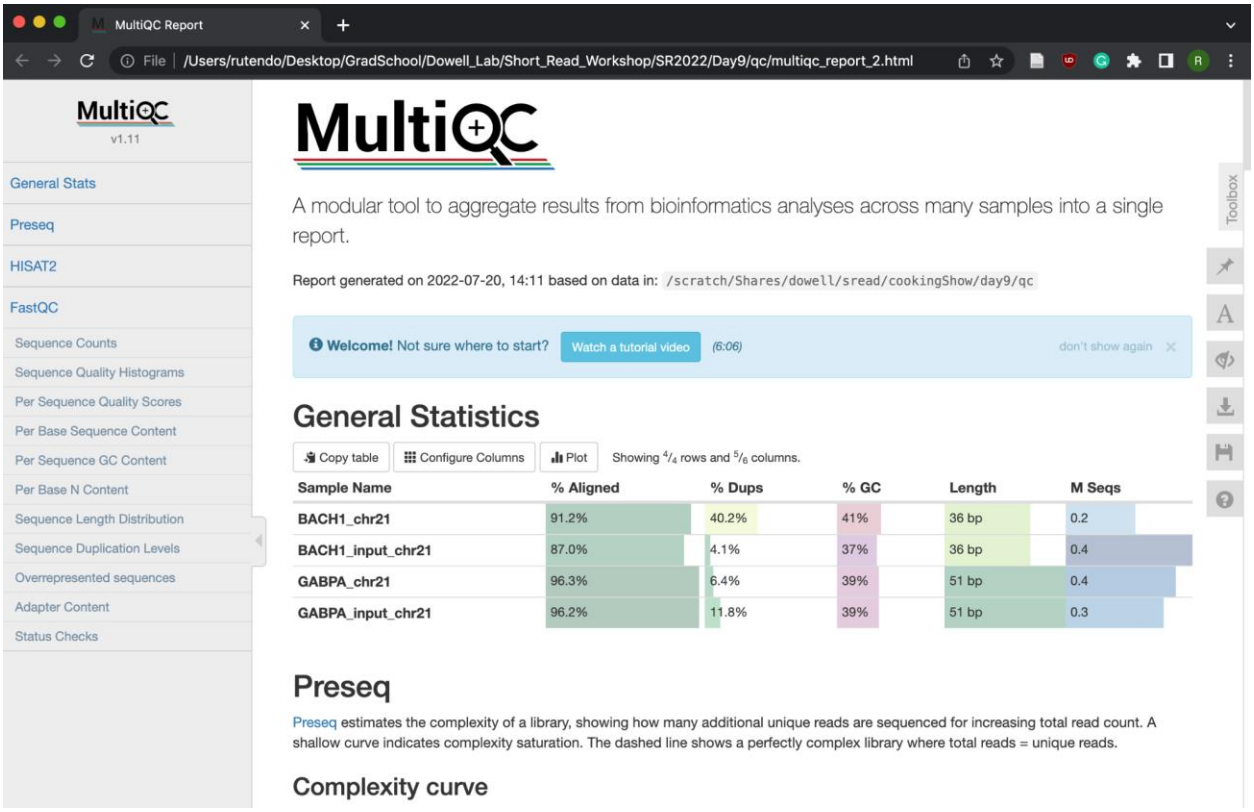
6. Lastly, we can summarize all the QC output using **`multiqc`**. This tool summarizes all the QC metrics within a specified folder and shows all the samples summaries side by side. As shown below, the command for running multqc only requires the folder that the program will summarize over (i.e. the **`${QC}`** folder).

```
############################################################
#Run multiqc                                              ##
############################################################

##this will summarize all the QC analysis we have done above in a single document

multiqc ${QC} #this is commented
```

Below is an example output from **`multiqc`**. There is a summary table for all the quality control metrics reported, additionally, several tabs for each of the QC metrics can be explored in an interactive manner. You can copy an example of the **`multiqc`** output from **`/scratch/Shares/public/sread2022/cookingShow/day9/scripts`**, you will need to move both the **folder** **`multiqc_data`** and the **html file** **`multiqc_report.html`** to your local computer. You can open the html file in a web browser to interact with the page.


MultiQC Report
File | /Users/rutendo/Desktop/GradSchool/Dowell_Lab/Short_Read_Workshop/SR2022/Day9/qc/multiqc_report_2.html
MultiQC
v1.11
General Stats
Preseq
HISAT2
FastQC
Sequence Counts
Sequence Quality Histograms
Per Sequence Quality Scores
Per Base Sequence Content
Per Sequence GC Content
Per Base N Content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Adapter Content
Status Checks
MultiQC
A modular tool to aggregate results from bioinformatics analyses across many samples into a single report.
Report generated on 2022-07-20, 14:11 based on data in: /scratch/Shares/dowell/sread/cookingShow/day9/qc
Welcome! Not sure where to start? Watch a tutorial video (6:06)
don't show again
Toolbox
General Statistics
Copy table
Configure Columns
Plot
Showing 4/4 rows and 5/6 columns.
Sample Name
% Aligned
% Dups
% GC
Length
M Seqs
BACH1_chr21
91.2%
40.2%
41%
36 bp
0.2
BACH1_input_chr21
87.0%
4.1%
37%
36 bp
0.4
GABPA_chr21
96.3%
6.4%
39%
51 bp
0.4
GABPA_input_chr21
96.2%
11.8%
39%
51 bp
0.3
Preseq
Preseq estimates the complexity of a library, showing how many additional unique reads are sequenced for increasing total read count. A shallow curve indicates complexity saturation. The dashed line shows a perfectly complex library where total reads = unique reads.
Complexity curve