

## 2017 Poznan Summer School of Bioinformatics

### Session: Next Generation Sequencing (NGS)

#### ChIP-seq and ATAC/DNase-seq data analysis

Rebecca Worsley Hunt

##### **OUTLINE:**

1. Mapping and pre-processing reads before peak calling (p.4-18)
2. Peak calling with MACS2 (p.19-24)
3. Data visualization in IGV (p.25-27)
4. READ-ONLY the Irreproducible Discovery Rate (IDR) for thresholding (p.28-29)
5. Peak proximity to genomic features, e.g. transcription start sites (TSS) (p.30-32)
6. Overlap of ChIP-seq peaks and ATAC-seq peaks (p.33-34)

If you have time (which may not happen) there is a Part 2 – to look at the intersection of GWAS SNPs in regulatory regions

##### **You may choose where to start:**

- 1) Start from the raw reads through alignment to the genome. Then move to peak calling.
- 2) Start with peak calling (files are provided), and go back to processing reads and alignment if you have time later

Please ask questions! Questions are how you learn.

**ATTENTION!!! Don't copy and paste commands!** Word document changes quotes and dashes, and these will often not be recognized on the command line. It is better to type the commands yourself to give yourself time to think about what you are doing. Figure out what the command is meant to do rather than using the command without understanding it.

## BIOINFORMATIC TOOLS TODAY:

- FastQC
- STAR aligner
- Picard Tools
- samtools
- bedtools
- IGV browser
- MACS2

### Path for the tools:

The paths for the tools have probably been put in the environmental variable `PATH`, whose contents you can see by using the `echo` command, if you are curious:

```
echo $PATH
```

Therefore you may not need to explicitly include a path when calling the tool. The system will check the variable `PATH` for the tool's location. However, if it isn't in `PATH`, such as when you install something yourself, then you will want to provide the tool's location.

A "path" is simply the hierarchy of directories that you follow when looking for your file.

### BEDTOOLS – you will be using it, so take a little look at it

In these exercises you will use some of the utilities that the *bedtools* suite provides.

*<http://bedtools.readthedocs.org/en/latest/content/bedtools-suite.html>*

It is worth looking over what *bedtools* offers, as it is full of useful things that one repeatedly uses in computational analyses, and so can save you from writing scripts that already exist.

*Example: look into what `bedtools intersect` does, and its options:*

```
bedtools intersect    # the command to view the help menu
OR go to: http://bedtools.readthedocs.org/en/latest/content/bedtools-suite.html
```

*Required:*

`-a file1`

`-b file2`

*Optional:*

`-wa` report original regions from file1 that overlap the file2

`-wb` report the original file2 regions that overlap file1

Bedtools manual provides nicely descriptive diagrams for what the tools accomplish.



## Section 1 – pre-processing of ChIP-seq and ATAC-seq before peak calling

POINTS 0-1 ARE FOR READING, NOT PRACTICE

The following is not comprehensive, it does not cover every scenario/software. Nor does it cover some of the simplifications possible on the command line. It is simply meant to be a starting place for those who haven't done this before.

For the alignment stage and some processing (up until extracting uniquely mapping reads) you will use CTCF ChIP-seq data. For the final clean-up of the data prior to the peak calling stage you can choose between the ChIP-seq or ATAC-seq data, as this is where their processing diverges.

The processing steps are provided as blocks of information along with the commands used. For daily use most of the commands would be placed into one or two scripting files as a pipeline, so that the output of one processing step would feed into the next processing step, without you having to run each step individually.

### DATA:

The home directory is accessible using `$HOME` (an environmental variable, therefore in capital letters). The data for today has put in the home space inside the directory `chipseq`. Copy this directory into your own directory:

```
cp -r $HOME/chipseq $HOME/YOUR_DIR/ # where YOUR_DIR is your name
```

The `cp -r` says “copy recursively, this directory and it's contents into the specified location”

All of your results should be sent to your directory that you created within the home space. e.g. you have created a directory with your name, which is where you store your data

```
echo $HOME # to see the full path to home. echo tells you the content of the variable $HOME
mydir=$HOME/YOUR_DIR # create a variable for the path of your own directory, where
                     YOUR_DIR is the name of your directory
```

You can use `ls -lh` to see what is in a directory or list information about a file (`-l` is long format, `-h` is human readable file size).

```
ls -lh # list what is in the directory that you are currently looking at
ls -lh $HOME # list what is in the home directory (it might be the same as above if you are already in
“home”)
ls -lh $mydir # list what is in your directory
```

### A) Location for today's files:

If you copied the data files to your directory then this is the path for the data going forward:

```
$mydir/chipseq/
```

**PLEASE: use full paths for files (data or results), this may sound painful, but the full path prevents errors. A path is simply the hierarchy of directories that you walk through to reach the file you are interested in.**

Examples of the output are in directories starting with "test"

```
chipseq_ctcf_chr21/  
  testfastqc  
  testcutadaptchr21  
  teststaralign  
  starchr21rep1/testfiltering
```

```
atacseq_chr21/testfiltering
```

## **B) Path for your results:**

You need to choose a place that your results will be written to. You probably don't want to put them in the same place as the data files unless you are very confident you won't overwrite an existing file. It's probably better you make a new directory for them, e.g.:

```
mkdir $mydir/your_results/          # change the "your_results" to whatever you want
```

Going forward, you will substitute your chosen results location where `$mydir/your_results/` is written. You may make multiple directories to keep results for different steps separated.

## **ChIP-seq data overview:**

**Sample** – CTCF ChIP'd DNA replicates (GM12878). Usually single-end.

### **Control types:**

**Input** – this is cross-linked DNA that is then sheared by sonication like the sample, but no IP is done. It generates a genomic background for how the DNA fragments when it is sheared. DNA does not fragment uniformly, which is not surprising given that there is structure to the chromatin.

This is the control you have for the CTCF ChIP data.

**IgG** – this is cross-linked DNA, that has been sheared like the sample, but is then run over an IP column with the non-specific IgG antibody. This is like the Input control except that it then further filters for those fragments of sheared DNA that are 'sticky' to the non-specific antibody, and thus might show up repetitively in the sample.

We do not have this control today.

**Matched condition** – you might have two conditions that you are interested in comparing, thus one condition will act as control for the other.

Ideally, one might have both input and IgG controls as they tell different stories with regards to the background, but that has never become a standard

## **ATAC-seq data overview:**

**Sample** – Transposase digested DNA, replicates (GM12878). Paired-end.

**No Controls:**

ATAC-seq protocol doesn't have controls to date.

DNase-seq is similar to ATAC-seq, except there is both single-end and paired-end available, so you change the processing to fit the read type. With DNase we typically haven't used fragment size the way it is done for ATAC-seq, so one wouldn't need that from section 1.7 either.

-----  
**RECOMENDATION:**  
-----

In all cases, you should have replicates of your samples. Journals likely won't accept papers that don't have replicates, as that's where the statistical power for your analyses will come from.

If you have many samples, you will want to consider sequencing a small subset of them to check that the data is satisfactory. Sequencing 17 samples at the same time only to find out your data is poor, is not a happy thing. Yes, I have seen that mistake happen.

**FOR SPEED, YOU WILL PROCESS ONLY ONE REPLICATE. NOT BOTH REPS AND NOT CONTROLS. PROCESSED FILES FOR ALL WILL BE GIVEN FOR THE PEAK CALLING SECTION.**

-----  
**Outline:**  
-----

**Processing from a fastq file of raw reads, through to the final filtered reads that are ready for peak calling.**

- 1.0 Demultiplexing READ-ONLY**
- 1.1 Single-end or paired-end fastq information READ-ONLY**
- 1.2 FastQC for sequenced read statistics**
- 1.3 Trim adaptors from your reads**
- 1.4 Mapping reads to a genome**
  - A) create genome index**
  - B) align reads to the genome**
- 1.5 De-duplicate your reads**
- 1.6 Select the uniquely mapping reads**
- 1.7 Final filtering and processing prior to peak calling**

---

## 1.0 READ-ONLY

### Demultiplexing – Illumina (you might not need to demultiplex)

---

You have sent your sample off for sequencing. If your sequencing sample(s) (from the Next-seq, Hi-seq or whichever sequencing platform) is combined with other samples, then your data will have been uniquely barcoded so that after the sequencing you can separate these samples again. Pulling the samples apart is called demultiplexing. Illumina has software and instructions to do this. The Illumina software (bcl2fastq2) that we have experience with was easy to run as you simply run the executable and provide your file from the sequencing. e.g. `bcl2fastq2 yourfile`

Often the sequencing facility will already have done the demultiplexing for you, and you will have been given a fastq file.

---

## 1.1 READ-ONLY

### Single-end or paired-end fastq

---

**You have a fastq file of sequenced reads (single-end or paired-end)**, perhaps from a public database or you have sequenced your own samples. You will have one of the following:

- if you have single-end data you should have a single fastq file
- If you have paired-end data you will likely have two fastq files – one for read1 one for read2. If the filenames contain the numbers 1 and 2, then 1 often is read-1 and 2 is often read-2. This is not always true though, so if your data doesn't seem to behave quite right, e.g. when trimming adaptors, try switching the files.
- If you have paired-end but only **one** file then it likely means that the two reads have been “interleaved” (if the read length is 50bp, then you will see the reported sequences in the file are 100bp). You can use the fastq program to split the reads. e.g. `fastq-dump --split-files /path/yourfile.fastq`

---

## 1.2 FastQC for sequenced read statistics

---

You have a fastq file(s). This is human-readable (not a compressed binary file) therefore you can use a `head` command on the file if you want to see some of the starting lines. They are not small files so do not try opening the file with a text editor on the command line!

You will want to get an overview of the quality of the reads, total number of reads, read length, %GC, overrepresented sequences etc. For this you can use the FastQC program. It doesn't take too long to run and will give you this information in a nice html format that so that you see the plots and information in your browser.

#make a directory for results, and then run fastqc. If the full genome is too slow then use the chr21 file

```
ls -lh $mydir/chipseq/chipseq_ctcf_chr21/

fileAll=wgEncodeBroadHistoneGm12878CtcfStdRawDataRep1.fastq
file21=wgEncodeBroadHistoneGm12878CtcfStdRawDataRep1.chr21.fastq

mkdir $mydir/your_results/fastqc_ctcf/
dirfastqc=$mydir/your_results/fastqc_ctcf
```

```
fastqc -o $dirfastqc -f fastq $mydir/chipseq/chipseq_ctcf_chr21/$fileAll
```

If you want to unzip the zip file to see the zipped directory:

```
cd $dirfastqc
```

```
unzip *fastqc.zip
```

```
ls -lh wgEncode*_fastqc/ # list contents of the dir. starting with wgEncode and ending fastqc
```

Your results will be in an html file, e.g. `yourInputFile_fastqc.html`, that you can view in a web browser. Results are also in a zip file containing the text files that make up the html.

**You have used IGV so you can open it as you did in the RNA-seq session, and ignore the below.**

```
# For the browser you need to know the full path
```

```
ls $dirfastqc/fastqc_ctcfrep1*.html # use ls to see what the full path is
```

```
# then put it into this form (substituting as necessary), and put in your browser window
```

```
file:///pathtoresults/fastqc_report.html
```

e.g. [file:///Volumes/rebecca/poznan/testbedfastq/test\\_fastqc.html](file:///Volumes/rebecca/poznan/testbedfastq/test_fastqc.html)

If the FastQC html report is not accessible from your computer, then you can use `scp` to do a secure copy from the server to your machine. Copy the directory to your `$mydir` space on the machine

```
cd $mydir
```

```
scp yourname@serveraddress:/pathToFastQC_results/* .
```

Examples of a **good** dataset and a **bad** dataset are provided by the authors of FastQC:

[https://www.bioinformatics.babraham.ac.uk/projects/fastqc/good\\_sequence\\_short\\_fastqc.html](https://www.bioinformatics.babraham.ac.uk/projects/fastqc/good_sequence_short_fastqc.html)

[https://www.bioinformatics.babraham.ac.uk/projects/fastqc/bad\\_sequence\\_fastqc.html](https://www.bioinformatics.babraham.ac.uk/projects/fastqc/bad_sequence_fastqc.html)

These are the 12 things that FastQC reports on. They have a coloured circle beside them in the html: green means everything ok, orange is a warning to look into the topic, and red is failure to pass FastQC's thresholds. It's up to you if a given evaluation matters to you. For the basics, look at highlighted headings:

- | Heading                        | useful   |
|--------------------------------|--|
| • Basic Statistics             | total sequences  |
| • Per base sequence quality    |  |
| • Per tile sequence quality    |  |
| • Per sequence quality scores  |  |
| • Per base sequence content    |  |
| • Per sequence GC content      |  |
| • Per base N content           |  |
| • Sequence Length Distribution |  |
| • Sequence Duplication Levels  | hopefully not too high   |
| • Overrepresented sequences    | useful for identifying adaptors or weirdness. Note 3 <sup>rd</sup> col is %, sometimes it's useful to add up the % because although it might be along list the influence on your data isn't worth worrying about. A number of the seqs are often similar with small changes. The 4 <sup>th</sup> column sometimes tells you the name of the adapter i.e. Illumina TruSeq |
| • Adapter Content              |  |
| • Kmer Content                 |  |



We will work with chr21 for speed going forward from here.

### 1.3 Trim adaptors from your reads – cutadapt, flexbar, etc

Sometimes your DNA fragments will be short enough that the sequencer will read through both the DNA sequence and part of the sequencing adaptors at the 3' end of the fragment. Thus there may be traces of adaptor information that you should remove before you align reads to a genome. The adaptor fragments will often show up as over-represented sequence in the FastQC report and the adaptor name may even be written in the report – this is convenient if you don't know what adaptor(s) was used.

If you don't know what adaptors were used, because you didn't generate the data, use the over-represented sequence information from FastQC together with the Illumina list of adaptors to figure it out:

[https://support.illumina.com/content/dam/illumina-support/documents/documentation/chemistry\\_documentation/experiment-design/illumina-adaptor-sequences\\_1000000002694-00.pdf](https://support.illumina.com/content/dam/illumina-support/documents/documentation/chemistry_documentation/experiment-design/illumina-adaptor-sequences_1000000002694-00.pdf)

You can use programs such as *cutadapt* and *flexbar* to trim off the adaptors. Even if adaptors don't show up in FastQC as overrepresented, it's still a good idea to do this if you already know the adaptors used. Sadly, many papers don't report the adaptors, so if you are using public data this could be a painful step.

**# ChIP-seq CTCF**, if you look at “Overrepresented sequences” in your FastQC output you will see nothing reported. We don't know what adapters were used (or maybe they were already removed). However, for the sake of practice you can use the provided adaptors. If you look in the cut-adapt log file you will see not much was done to the reads.

```
infile=$mydir/chipseq/chipseq_ctcf_chr21/wgEncodeBroadHistoneGm12878CtcfStdRawDataRep1.chr21.fastq
outfile=$mydir/your_results/ctcfchr21.cutadapt-m20.fastq

cutadapt -m 20 -a CTGTCTCTTATACACATCTGACGCTGCCGACGA -o $outfile $infile &>
$mydir/your_results/log_cutadapt.txt &
```

Look at the cutadapt report. You can see the number reads that had N number of base pairs trimmed. How many reads were processed and the % that were trimmed.

```
less $mydir/your_results/log_cutadapt.txt      # use q to quit
```

#### READ-ONLY

**# ATAC-seq (these adaptors are used for ATAC-seq)**

```
reads1=myfastqfile.read1.fastq
reads2=myfastqfile.read2.fastq
```

```
cutadapt -m 20 -a CTGTCTCTTATACACATCTGACGCTGCCGACGA -A
CTGTCTCTTATACACATCTCCGAGCCCACGAGAC -o myfile.read1.cutadapt-m20.fastq -p
myfile.read2.cutadapt-m20.fastq $reads1 $reads2
```

An example of a cutadapt report for the paired-end ATAC-seq data (genome-wide) is given. Both adapters are reported:

```
less $mydir/chipseq/atacseq_chr21/testcutadapt/log.atacfull.cutadapt-m20.txt
```

Look up the **options/parameters** given to *cutadapt* (v.1.9.1) in the software documentation.  
<https://cutadapt.readthedocs.org/en/stable/guide.html#basic-usage>

## 1.4 Mapping reads to a genome – STAR, Bowtie2 etc

### (A) Create genome index

### (B) Align reads to the genome

#### 1.4(A) Generate genome index – STAR

if you don't have a **genome index**, you will need to generate one. This can take some time on a full genome, especially with STAR, which considers splice junctions. You only need to do this once per genome assembly for a given read length (because we don't use splice junctions for ChIP-seq read length isn't really an issue, but maybe you would use it for RNA-seq later). We will do this for only chr21 here.

You need a genome fasta file, a genome annotation gtf file, and the length of your reads. e.g. GRCh37.p13.genome.chr21.**fa**, gencode.v19.annotation.chr21.**gtf**, and what you saw in the FastQC file (probably 36bp)

e.g. some STAR indexing options and parameters

```
--runThreadN 6
--runMode genomeGenerate
--genomeDir /path/yourindexDir
--genomeFastaFiles /path/organism.genome.fa
--sjdbGTFfile /path/annotationfile.gtf
--sjdbOverhang 36 # this one is the length of your reads
```

#### Generate STAR index:

#### Set the variables to your locations and files

```
yourStarindexDir=$mydir/your_results/starIndex36bp
fasta=$mydir/chipseq/star_indexing_files/GRCh37.p13.genome.chr21.fa
gtf=$mydir/chipseq/star_indexing_files/gencode.v19.annotation.chr21.gtf
lenOfRead=36
```

```
mkdir $yourStarindexDir
```

```
STAR --runMode genomeGenerate --runThreadN 1 --genomeDir $yourStarindexDir --
genomeFastaFiles $fasta --sjdbGTFfile $gtf --sjdbOverhang $lenOfRead
```

#### READ ONLY

e.g. Bowtie2 indexing options and parameters (simpler because splice junctions aren't involved)

```
bowtie2-build /path/GRCh37.p13.genome.fa hg19
```

#### 1.4(B) Align reads to the genome – STAR

There are many **aligner programs** out there. Oh soooo many. STAR performs very well in all aspects. It is also very fast. STAR is written to consider splice junctions, However, with ATAC-seq or ChIP-seq you aren't considering splice junctions thus you must suppress this. Bowtie2 is another good aligner used by many labs, it does not do splice junctions so you don't have to think about turning that aspect off.

The Ohler lab tested many aligners -- STAR, Bowtie2, TopHat2, BWA, BWA-PSSM, GEM, HISAT etc. -- evaluating unique alignments, errors in alignment, number of unmapped reads etc. STAR ranked 1<sup>st</sup> in more categories than the other aligners. Regardless, one often ends up with ~70% of your reads being uniquely mapped to the genome and the difference between STAR and Bowtie2 is not significant (although STAR often results in more uniquely mapped reads). So if you feel Bowtie2 is simpler to use, absolutely use that for your analysis, although we will use STAR for this tutorial. To use either STAR or Bowtie2 you will need to generate genome indexes for the aligners to use. For STAR you do this once for each combination of genome assembly and sample read length (if not concerned with splice junctions then read length is not important). For Bowtie2 you do this once per genome assembly.

ChIP-seq commonly obtains reads from both 5' ends of the pulled-down fragments, but the DNA is sequenced in a single-end fashion (not paired-end), which means it does not record which reads are paired together *i.e.* derived from the same fragment. Therefore you will have only one read file to submit for alignment. ATAC-seq is typically paired-end data, and you need to supply two read files to the aligner, so there is a slight change in options and parameters which you can read about in the respective aligner manuals.

Note: instead of using software like cutadapt to trim adapters, you can do this in STAR with *option* `--clip3pAdapterSeq`. I've found it works well.

**STAR 2.5.1 manual** (slightly overwhelming in terms of options)  
<https://github.com/alexdobin/STAR/blob/master/doc/STARmanual.pdf>

The choice of software *options* below is a place to start. As you read about the *options* you may want to test different ones to see how your results change.

**A few notes:**

If you want to submit gzip files you need: `--readFilesCommand zcat`

If you want to allow soft-clipping at the 3' end of the reads (the poor quality end) then you need to change `alignEndsType` to: `--alignEndsType Extend5pOfRead1`

You can shutdown STAR's splice junction search (done in the below commands) by setting the overhang minimum (`--alignSJDBoverhangMin` and `--alignSJoverhangMin`) to be the greatest length of your reads, or setting the maximum intron length to something small, like 10. There are other *options* you could potentially use to do this too. You can also filter out the spliced reads *after* mapping as these will contain "N" in the CIGAR string.

```
--alignSJoverhangMin 100 --alignSJDBoverhangMin 100
OR --alignIntronMax 10
```

**STAR options and parameters: ChIP-seq single-end data (using read length 36bp). Look up these options in the STAR manual so you get an idea of what they do.**

```
--genomeDir /path/genomeIndex
--runThreadN 6
--alignEndsType EndToEnd
--outSAMattributes NH HI AS nM MD NM
--outFilterMismatchNoverReadLmax 0.04
--outFilterMismatchNoverLmax 1
--alignSJoverhangMin 36
--alignSJDBoverhangMin 36
--outReadsUnmapped Fastx
--readFilesIn /path/reads1.fastq
--outFileNamePrefix #with this you can make a name for your files
```

## RUN STAR – ChIP-seq

→ use the genome index you generated, and the ChIP-seq chr21 fastq file (raw reads).

```
starindexDir=$mydir/your_results/starIndex36bp
readlen=36
fastq=$mydir/chipseq/chipseq_ctcf_chr21/wgEncodeBroadHistoneGm12878CtcfStdRawDataRep1.chr21.fastq
```

```
STAR --genomeDir $starindexDir --runThreadN 1 --alignEndsType EndToEnd --
outSAMAttributes NH HI AS nM MD NM --outFilterMismatchNoverReadLmax 0.04 --
outFilterMismatchNoverLmax 1 --alignSJoverhangMin $readlen --
alignSJDBoverhangMin 36 --outReadsUnmapped Fastx --readFilesIn $fastq
```

# Use `ls -lh` to list what is inside your directory now (-l is long format, -h is human readable file size)

# → take a look in STAR's log report for the data e.g. % mapped, multi-map, and unmapped

```
less $mydir/your_results/Log.final.out    # use q to quit
```

**Optional:** If you are curious, and want to use IGV to look at the aligned data, you can check **Section 3** for how to start IGV and load a file. Then come back here and go to the below coordinates.

→ to look at alignment in IGV browser, you will want a bam file (compressed)

→ generate a bam file from the alignment sam file, then sort, and index the bam.

Set the variables `samfile`, `bamfile` and `bamSrt` to equal your file names.

```
samfile=$mydir/your_results/Aligned.out.sam
bamfile=$mydir/your_results/Aligned.out.bam
bamSrt=$mydir/your_results/Aligned.out.srt.bam
```

```
samtools view -bS $samfile > $bamfile
samtools sort $bamfile > $bamSrt
samtools index $bamSrt
```

In IGV try going to: chr21:26,971,449-26,992,886 or chr21:38,343,093-38,368,682 (you can use any coord. you want)

## READ-ONLY – ATAC-seq

**STAR options and parameters: ATAC-seq paired-end data (using read length 50bp). Look up these options in the STAR manual to have an idea what they do (almost all are the same as ChIP-seq).**

```
--genomeDir /path/genomeIndex
--runThreadN 6
--alignEndsType EndToEnd
--outSAMAttributes NH HI AS nM MD NM
--outFilterMismatchNoverReadLmax 0.04
--outFilterMismatchNoverLmax 1
--alignSJoverhangMin 75
--alignSJDBoverhangMin 75
--outReadsUnmapped Fastx
--alignMatesGapMax 1500
--readFilesIn /path/reads1.fastq /path/reads2.fastq
--outFileNamePrefix #with this you can make a name for your files
```

With ATAC-seq we limit the fragment length *i.e.* the gap between the read mates (paired reads), to 1500bp or 2000bp, as we aren't really interested in larger fragment sizes (there usually isn't much that is longer).

## READ-ONLY

**Bowtie2** 2.2.6 manual

<http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml>

*e.g. single-end data*

```
bowtie2 -p 6 -x /path/genomeIndex -U /path/reads1.fastq -S  
/path/mydata_hg19_Aligned.sam
```

Look up the **options** in the Bowtie2 manual.

---

### 1.5 De-duplicate your reads (if that's advisable for your data type)

---

There will be duplicated reads, which is evaluated by the mapped coordinate of the 5' end. This might be real, but it is also commonly caused due to PCR amplification of reads. For ChIP-seq data one generally removes the duplicate reads and only keeps one copy of the read. This is true for ATAC-seq too. Yes, duplication can occur and be real, but it is considered the lesser evil to sacrifice those possibilities relative to the PCR issue. Some groups are working on how to distinguish real duplication from PCR duplication events. Other types of data, like GRO-cap for the start location of nascent transcripts, you keep duplicate reads as you expect the start of the transcript to frequently be the same single bp.

Example tools you can use for this are Picard, and samtools. Picard is the better option.

<https://broadinstitute.github.io/picard/>

<http://www.htslib.org/doc/samtools.html>

Picard can take either sam or bam files. We will use the sam file that STAR produced, but you could convert sam to bam to save space.

[We continue with chr21 for speed here. You may need the full Picard path here, but try it without first.](#)

#### Using Picard:

**First** sort reads by coordinate (not by name). Here we use Picard to sort (but you could use samtools sort).

Set the variables `samfile` and `outfile` to equal your file names.

```
samfile=$mydir/your_results/Aligned.out.sam  
outfile=$mydir/your_results/Aligned.out.coordSrt.sam
```

```
java -Xms1G -Xmx4G -jar SortSam.jar INPUT=$samfile OUTPUT=$outfile  
SORT_ORDER=coordinate VALIDATION_STRINGENCY=SILENT
```

#The `-Xms1G` and `-Xmx4G` control the range of memory java will have access to.

#### IF THE ABOVE COMMAND DOESN'T WORK TRY THIS WHEN AT THE SCHOOL COMPUTERS:

```
Picard SortSam.jar INPUT=$samfile OUTPUT=$outfile SORT_ORDER=coordinate  
VALIDATION_STRINGENCY=SILENT
```

### Second use Picard MarkDuplicates:

```
infile=$mydir/your_results/Aligned.out.coordSrt.sam
outfile=$mydir/your_results/Aligned.out.coordSrt.rmdup.sam
outmetric=$mydir/your_results/Aligned.out.coordSrt.rmdup.metrics.txt

java -Xms1G -Xmx4G -jar MarkDuplicates.jar INPUT=$infile OUTPUT=$outfile
METRICS_FILE=$outmetric REMOVE_DUPLICATES=TRUE VALIDATION_STRINGENCY=SILENT
```

### IF THE ABOVE COMMAND DOESN'T WORK TRY THIS WHEN AT THE SCHOOL COMPUTERS:

```
Picard MarkDuplicates.jar INPUT=$infile OUTPUT=$outfile
METRICS_FILE=$outmetric REMOVE_DUPLICATES=TRUE VALIDATION_STRINGENCY=SILENT
```

If you are curious to look at the top few lines of the sam file you can use samtools view:

```
samtools view -h $outfile | head
```

---

## 1.6 Select the uniquely mapping reads

---

There will be reads that do not map to the genome at all. There will also be reads that map to multiple locations throughout the genome. Finally, there are reads that map uniquely *i.e.* to a single location. Generally we tend to keep the uniquely mapping reads only, rather than trying to figure out which of the multi-mapping locations is the one we think most reasonable to keep for a given read. As prior mention, one keeps about 70% of the reads in the end (maybe a little less, sometimes a little more). If one is working with paired end data there are further considerations such as whether both paired-reads map uniquely, one of the reads in a pair maps uniquely, or both are multi-mapping.

STAR output does not give you the unmapped reads in the same file as those that mapped. Bowtie2 default output gives you every read and its status, whether it mapped to the genome or not.

When mapping with STAR you select for the flag `NH:i:1` to give you the unique reads. You must match `NH:i:1` exactly, thus require a tab after the "1" to avoid also matching `NH:i:10`. Keep lines that start with `@` because that is the sam file header.

# extract the uniquely mapped reads. Remember to change the file names.

```
reads=$mydir/your_results/Aligned.out.coordSrt.rmdup.sam
uniqreads=$mydir/your_results/Aligned.out.coordSrt.rmdup.uniq.sam
```

```
samtools view -h $reads | awk '{if($0 ~ /^@/ || $0 ~ /NH:i:1\t/) print $0 }' >
$uniqreads
```

# did the file count change?

```
samtools view -c $reads
samtools view -c $uniqreads
```

## READ-ONLY

If using Bowtie2 then you want to keep everything that mapped (AS:i:) and then to remove those reads which mapped to multiple locations (XS:i:). Keep lines that start with @ because that is the sam file header.

```
grep "AS:i:~|^@" myreads.rmdup.sam | grep -v "XS:i:" > myreads.rmdup.uniq.sam
```

### 1.7 Filtering: scaffolds and chrM, chrU etc. and for ATAC-seq do extra processing

Here you can choose to continue with the ChIP-seq or do the final processing of ATAC-seq (chr21), before starting peak calling. ChIP-seq is a bit boring, as you will see if you read below.

#### ChIP-seq

At the last step we had a sam file. Now we will convert this to a bed file to then remove unwanted data.

# Convert sam to bam, then bam to a bed file, and sort by coordinate

```
samfile=$mydir/your_results/Aligned.out.coordSrt.rmdup.uniq.sam
bamfile=$mydir/your_results/Aligned.out.coordSrt.rmdup.uniq.bam
outfile=$mydir/your_results/Aligned.out.coordSrt.rmdup.uniq.bed
filtered=$mydir/your_results/Aligned.out.coordSrt.rmdup.uniq.chrfilt.bed
```

```
samtools view -b $samfile > $bamfile
```

```
bedtools bamtobed -i $bamfile | sort -k1,4,1 -k2,2n -k3,3n > $outfile
```

# Then filter unwanted scaffolds/chromosomes (random, chrU, chrM) etc. In our case this isn't necessary because we only have chr21 in the example, but usually this step has an impact.

```
awk -F $'\t' '{if($1 ~ /^chr/ && $1 !~ /_/) print $0}' $outfile | grep -v
"chrM\|chrU\|random" > $filtered
```

#### ATAC-seq

If you chose ATAC-seq instead of ChIP-seq, you will use the provided chr21 files to do the processing on.

```
$mydir/chipseq/atacseq_chr21/
```

The final processing of paired-end ATAC-seq is more involved than single-end read ChIP-seq because you want to ensure you are only working with 1) concordant read pairs, 2) read pairs that are >38bp and <1500-2000bp apart, 3) the 1bp cut-sites of the transposase (or alternatively the region that the transposase occupies).

The 38bp is the estimated width of the transposase protein footprint on the DNA. The 1500 or 2000 bp is simply the limit of interest in terms of digested fragment length, but there usually isn't much that is longer than that.

Convert the sam file to a bam with index, sort by name, and then generate a bedpe file. In the bedpe format (created by bedTools) the read pairs will be on the same line in the file. This makes it simple for you to ensure that read pairs are concordant.

```
samfile=$mydir/chipseq/atacseq_chr21/rep1_hg19_Aligned.out.coordSrt.rmdup.uniq.chr21.sam
```

**#you will generate these files**

```
bamfile=$mydir/your_results/atacseq_chr21/rep1_hg19_Aligned.out.coordSrt.rmdup.uniq.chr21.bam
```

```
sortbam=$mydir/your_results/atacseq_chr21/rep1_hg19_Aligned.out.coordSrt.rmdup.uniq.chr21.nameSrt.bam
```

```
bedpefile=$mydir/your_results/rep1_hg19_Aligned.out.coordSrt.rmdup.uniq.chr21.nameSrt.bedpe
```

```
samtools view -Sb $samfile > $bamfile
```

```
samtools index $bamfile
```

```
samtools sort -n $bamfile -o $sortbam
```

```
bedtools bamtobed -bedpe -i $sortbam |sort -k1.4,1 -k2,2n -k3,3n > $bedpefile
```

**# look at the format of the bedpe file, you should see a pair of coordinates on each line: col.1-3 and col. 4-6**

```
head $bedpefile
```

**# are there lines where the read pairs are on the same strand? (the last two columns)**

```
grep "+"$'\t'"+" $bedpefile | wc -l
```

```
grep "+"$'\t'"+" $bedpefile | head
```

**# filter unwanted scaffolds/chromosomes (chrM, chrU etc), ensure read pairs are concordant (i.e. read1 +ve strand, read2 -ve strand), and select fragments with length 38-1500bp.**

**# You can visually split the below command on the pipes "|" to get five separate pieces. The first is filtering out scaffolds, the second is unwanted chr, the third checks concordance and prints fragment length, the fourth checks for the rare case when we will keep a fragment whose read1 is negative strand, and the fifth filters on fragment length.**

```
bedpefile=$mydir/your_results/rep1_hg19_Aligned.out.coordSrt.rmdup.uniq.chr21.nameSrt.bedpe
```

```
outfile=$mydir/your_results/rep1_hg19_Aligned.out.coordSrt.rmdup.uniq.chr21.filtered.bedpe
```

```
awk -F $'\t' '{OFS="\t"; if($1 ~ /^chr/ && $1 !~ /_/) print $0}' $bedpefile |  
grep -v "chrM|chrU|random" | awk '{OFS="\t"; if($9!=$10 && $1==$4 && $5>=$2)  
print $0, $6-$2-9}' | awk '{OFS="\t"; if($9=="+"){print $0}else{if($2==$5 &&  
$3==$6) print $1,$2,$3,$4,$5,$6,$7,$8,$10,$9,$11}}' | awk '{OFS="\t";  
if($11>=38 && $11<=2000) print $0}' > $outfile
```

**Did the read count change?**

Use **wc -l** on \$bedpefile and \$outfile

**Explanation of the awk commands piped together:**

OFS="\t" means the output will be tab delimited

1) The first column is the chromosome name. If the first column starts with "chr" \$1 ~ /^chr/ AND the first column does NOT contain an underscore \$1 !~ /\_/ then keep it.

```
awk -F $'\t' '{OFS="\t"; if($1 ~ /^chr/ && $1 !~ /_/) print $0}'
```

2) check for lines that do NOT (-v) contain chrM or chrU or random (do not want those chromosomes)



```
grep -v "chrM\|chrU\|random"
```

3) if the strands are different  $\$9 \neq \$10$  and the chromosomes are the same  $\$1 == \$4$  and read2 has a bigger coordinate than read1  $\$5 \geq \$2$ , then subtract read1 coordinate from read2 coordinate  $\$6 - \$2$  (the additional subtract 9 is because of how the transposase cuts) and put it in the last column of the file

```
awk '{OFS="\t"; if($9!= $10 && $1== $4 && $5>=$2) print $0, $6-$2-9}'
```

4) if read1 is positive  $\$9 == "+"$  then we keep it, but if read1 is negative then the only time we keep it is if the coordinates are exactly the same  $\$2 == \$5$  &&  $\$3 == \$6$  keep the fragment

```
awk '{OFS="\t"; if($9=="+") {print $0} else {if($2== $5 && $3== $6) print $1,$2,$3,$4,$5,$6,$7,$8,$10,$9,$11}}'
```

5) look at the fragment length from step 4) and if it is  $\geq 38$  and  $\leq 2000$ , then keep it

```
awk '{OFS="\t"; if($11>=38 && $11<=2000) print $0}'
```

```
# reduce reads to 1bp at the 5' end, as we are interested in the density of transposase cut sites in a region.
```

```
infile=$mydir/your_results/rep1_hg19_Aligned.out.coordSrt.rmdup.uniq.chr21.filtered.bedpe
outfile=$mydir/your_results/rep1_hg19_Aligned.out.coordSrt.rmdup.uniq.chr21.filtered.1bp.bedpe
```

```
awk '{OFS="\t"; print $1,$2,$2+1,$4,$6-1,$6,$7,$8,$9,$10,$11}' $infile > $outfile
```

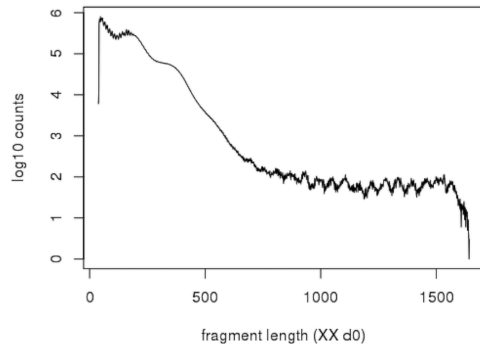
# convert the ATAC bedpe back to a bam file (reads will be separate lines), and index the bam  
A chromosome size file is needed. The one below was from UCSC. You can do a google search with terms: hg19 chromosome sizes

```
chrsize=$mydir/chipseq/hg19.chrom.sizes
bedpe=$mydir/your_results/rep1_hg19_Aligned.out.coordSrt.rmdup.uniq.chr21.filtered.1bp.bedpe
outbam=$mydir/your_results/rep1_hg19_Aligned.out.coordSrt.rmdup.uniq.chr21.filtered.1bp.bedpe.bam
```

```
bedtools bedpetobam -i $bedpe -g $chrsize | samtools sort - -o $outbam # the "-" is for the incoming data from the pipe
```

```
samtools index $outbam
```

**Optional:** from the bedpe you can count how often discrete fragment lengths occur i.e. the space between two cut sites (a simple unix command can do this for you), then if you are a little familiar with R you can plot the fragment length on the x-axis and log of the counts on the y-axis, to get see if your data produces the distribution that occurs due to fragments that span 0,1,2,3 etc nucleosomes. The highest point will be fragments <150bp (no nucleosome).



You probably won't get the above plot with only chr21, however you can look at the ratio of short vs long fragments lengths.

Use your `...filtered.1bp.bedpe` file

```
bedpe=$mydir/your_results/rep1_hg19_Aligned.out.coordSrt.rmdup.uniq.chr21.filtered.1bp.bedpe
```

#Note, bedpe files are 0-based, when you subtract numbers you don't need to add 1.

```
awk '{OFS="\t"; print $0, $2-$6}' $bedpe > tmpfile
```

#check inside your new file to see if it looks ok  
`head tmpfile`

# if the file looks ok, then replace the original file with this new one by using the move (`mv`) command.  
`mv tmpfile $bedpe`

Now you can assess the ratio of short to long fragments e.g. <100bp vs >150bp i.e. count how many lines are size <100bp, and count how many lines are >150bp. It should be approximately in the range of 2:1 or 1:2, not something extreme like 1:10

-----  
**Now you are ready for peak calling.**  
**Further notes:**  
 -----

If the peak caller you use wants a bam file, then convert your sam file to a bam file. If the peak caller wants bed files, then convert your sam file to a bam file and then the bam file to a bed file (yes, it is a little round-about).

Often a tool that accepts a bam file needs the bam file to be indexed, which is easily done with the `samtools index` command (the bam file should be sorted by coordinate, which is often the default for a bam file, but if necessary it can be so sorted with `samtools sort`)

`samtools` and `bedtools` are complimentary tools to accomplishing these file conversions.

`samtools`: <http://www.htslib.org/doc/samtools.html>

`bedtools`: <http://bedtools.readthedocs.org/en/latest/content/bedtools-suite.html>

## ----- **Data file formats** -----

Data file formats are an attempt to create a standard for how data is organized in a file. In theory researchers would follow the standards when creating data files e.g. bed files, and then every computational person would be happy because they know what is in the file without having to double check. However, this does not happen. It is always a good idea to double check that the data in the file matches what you expect, e.g. a q-value is meant to be column X, or coordinates are meant to be 0-based.

Descriptions of file formats such as bed, narrowPeak, broadPeak, *etc.* are easily found with a google search. UCSC also has a page that describes many of them:

<http://www.genome.ucsc.edu/FAQ/FAQformat.html>

If you want to understand the sam file format, you can read it from the following PDF.

A bam file is simply a compressed version of the human-readable sam file.

<https://samtools.github.io/hts-specs/SAMv1.pdf>

**End of section 1 (The boxes around command calls are not continued in the next Section)**

## Section 2 – Peak calling on ChIP-seq or ATAC-seq data

- 2.0. locate data and make results directories
- 2.1 use MACS2 to call peaks
- 2.2 Filtering peaks against the blacklist regions

Peak callers are used to find genomic regions of enrichment with the aligned reads derived from experiments such as ChIP-seq, DNase-seq/ATAC-seq etc. There are many peak finders published, each with their own strengths and weaknesses. We will use MACS2 for this tutorial. MACS2 is well established and is one of the most widely used peak callers (do not use MACS, as MACS2 has improvements).

MACS2 (PMID: 18798982; 2008 - <https://github.com/taoliu/MACS> ).

You will identify either CTCF ChIP-seq peaks (data from ENCODE) **OR** open chromatin regions from ATAC-seq data (data from Greenleaf lab, 2013) in GM12878. Genome assembly: hg19/GRCh37. For the sake of time, **we have restricted the analyses to chr21**, but the full dataset of peaks at open chromatin regions has been provided.

<https://github.com/taoliu/MACS/> # can find a README for MACS2 and download the software from here (although the install is a bit of a pain)

### - MACS2 overview

MACS scans the genome in sliding overlapping windows and uses a local Poisson distribution to model read counts locally. This allows MACS to take into account local biases in read counts, albeit not explicitly. With MACS2 replicates reproducibility is not considered by the software. Replicates can be run separately, or pooled together.

In the tutorial we use pooled replicates. Usually you would analyze replicates separately because you will be doing downstream analysis such as differential analysis. Or using software such as IDR to select the subset of peaks that are the most reproducible.

**Other peak finders** like ZINBA, HMcan and F-seq can explicitly model sources of bias like GC content, copy number variation and mappability. However, they can be complicated to run. JAMM is relatively easy to install and run, and is the only peak finder I am aware of that handles replicates separately to identify peaks of greatest confidence.

Examples of the output are in directories starting with "test"

```
chipseq_ctcf_chr21/testmacs2
atacseq_chr21/testmacs2
```

**NOTE:** there are no longer boxes placed around the commands you will do.

## S2.0 Locate your data

You have processed one replicate of your data. For the sake of time, the second processed replicate, and controls if they exist, have been provided.

## ChIP-seq

---

Replicate 1 is from your processed results, use the final filtered bed file (or you can use the provided file)

```
$mydir/your_results/Aligned.out.coordSrt.rmdup.uniq.chrfilt.bed
```

OR provided:

```
$mydir/chipseq/chipseq_ctcf_chr21/starchr21rep1/Aligned.out.coordSrt.rmdup.uniq.chrfilt.bed
```

Replicate 2 is given:

```
$mydir/chipseq/chipseq_ctcf_chr21/starchr21rep2/Aligned.out.coordSrt.rmdup.uniq.chrfilt.bed
```

Both controls are given:

```
$mydir/chipseq/chipseq_ctcf_chr21/starchr21control1/Aligned.out.coordSrt.rmdup.uniq.chrfilt.bed
```

```
$mydir/chipseq/chipseq_ctcf_chr21/starchr21control2/Aligned.out.coordSrt.rmdup.uniq.chrfilt.bed
```

If you are tired of repeating the same path all the time, you can assign it to a variable (this assumes you are typing things out and not using copy/paste):

*e.g.*

```
givenchip=$mydir/chipseq/chipseq_ctcf_chr21/  
ls -lh $givenchip/starchr21rep2/
```

---

## ATAC-seq

---

Replicate 1 is from your processed results, use the final filtered bed file (or you can use the provided file)

```
$mydir/your_results/
```

OR provided:

```
$mydir/chipseq/atacseq_chr21/rep1_hg19_Aligned.out.coordSrt.rmdup.uniq.chr21.nameSrt.filtered.1bp.bedpe.bam
```

Replicate 2 is given:

```
$mydir/chipseq/atacseq_chr21/rep2_hg19_Aligned.out.coordSrt.rmdup.uniq.chr21.nameSrt.filtered.1bp.bedpe.bam
```

If you are tired of repeating the same path all the time, you can assign it to a variable (this assumes you are typing things out and not using copy/paste):

```
givenatac=$mydir/chipseq/atacseq_chr21/  
ls -lh $givenatac
```

---

## both data types

---

Reminder: your home directory is accessible using `$mydir`. All of your results should be sent to directories you create within your home space.

Create a new directory (or directories) to save your future results to (name the directory whatever you want)

```
mkdir $mydir/peak_results/
```

If you want a 'shortcut' to your directory for later use, you can create a variable (use any name you want, `myresults` is just an example, but don't use a word that is an existing command).

```
myresults=$mydir/peak_results/  
echo $myresults  
ls -lh $myresults          # your directory is probably empty
```

## S2.1 Using MACS2 to call peaks

### - MACS2 view options and data formats

Look at MACS2's help menu:

```
macs2 -h
```

OR because MACS2 is in your `PATH` environmental variable (your computer knows where to find it) you can use

```
macs2 -h
```

To view MACS2 `callpeak` options use the help menu, and note which options are used in the MACS2 run below.

```
macs2 callpeak -h
```

MACS2 can take aligned reads as either BAM or BED formats.

### Run MACS2:

```
-----  
ChIP-seq  
-----
```

chr21 isn't big so don't expect too many peaks (just a few hundred)

```
# This will be your bed file of final filtered reads from the processing you did  
rep1=$mydir/your_results/YOURFILE.bed    # put your file name here YOURFILE
```

OR you can use the provided file

```
rep1=$mydir/chipseq/chipseq_ctcf_chr21/starchr21rep1/Aligned.out.coordSrt.rmdup  
.uniq.chrfilt.bed
```

# Provided files

```
rep2=$mydir/chipseq/chipseq_ctcf_chr21/starchr21rep2/Aligned.out.coordSrt.rmdup  
.uniq.chrfilt.bed
```

```
ctrl1=$mydir/chipseq/chipseq_ctcf_chr21/starchr21control1/Aligned.out.coordSrt.
rmdup.uniq.chrfilt.bed
ctrl2=$mydir/chipseq/chipseq_ctcf_chr21/starchr21control2/Aligned.out.coordSrt.
rmdup.uniq.chrfilt.bed
# this is the path you want your output to go, and the prefix you want to use to name the output files
outpref=$mydir/peak_results/macs2_chip
```

```
macs2 callpeak -t $rep1 $rep2 -c $ctrl1 $ctrl2 -f BED -g hs -n $outpref -B -q
0.01 &> $mydir/peak_results/log_macs2-chip.txt &
```

If you list both replicate files after the `-t` (or both controls after the `-c`) MACS2 will pool the replicates together. If you had a downstream analysis requiring 2 reps, then you would run them separately.

## ----- ATAC-seq -----

chr21 isn't big so don't expect too many peaks (just a few hundred)

These MACS2 options and parameters detect focused peaks *i.e.* the strongest signals in an open chromatin region, not necessarily the full breadth of the open region. You could try the `--broad` parameter if you wanted broader regions (I have not needed to try this myself yet):

```
rep1=$mydir/chipseq/atacseq_chr21/rep1_hg19_Aligned.out.coordSrt.rmdup.uniq.chr
21.nameSrt.filtered.1bp.bedpe.bam
rep2=$mydir/chipseq/atacseq_chr21/rep2_hg19_Aligned.out.coordSrt.rmdup.uniq.chr
21.nameSrt.filtered.1bp.bedpe.bam
outpref=$mydir/peak_results/macs2_atac      # change the prefix name to what you want
shift=75
ext=150
```

```
macs2 callpeak -t $rep1 $rep2 -f BAM -g hs -q 0.01 -n $outpref --nomodel --
shift -$shift --extsize $ext &> $mydir/peak_results/log_macs2-atac.txt &
```

If you list both replicate files after the `-t` MACS2 will pool the replicates together. If you had a downstream analysis requiring multiple reps (*e.g.* differential analysis), then you would run them separately.

### Explanation of a few MACS2 options/parameters:

# you should read about the MACS2 options yourself, and find what we used and what is available.

# `-q 0.01` - is the q-value cutoff (MACS2 default is 0.05 but that is often not a good choice. Do not fall into the trap of "more peaks is the best", always look into your data)

# `--shift -75 --extsize 150` - the `nomodel/shift/extsize` refuses the fragment modeling option and places the 1bp ATAC-seq cut-site (the 1bp coordinate in your data file) at the centre of a pseudo-fragment, which in this case is 150bp long. With a smaller pseudo-fragment you will get improved peak edge resolution but you will get an increased number of low quality peaks. With a larger pseudo-fragment you may get single peaks called that consist of two peaks in close proximity, but less low quality peaks. So your choice of size is a balance, you can decide by looking at your data. MACS2 is fast so it doesn't take long to test multiple options/parameters. Usually 75/150 or 100/200 seems fairly good. Do not fall into the trap of "more peaks is the best", always look into your data.

# ChIP-seq generally does not use `shift / extsize` because MACS2 will try to create a model for fragment length based on the bimodal distribution of negative and positive reads. In ATAC-seq we are interested in the cut-site instead of the bimodal distribution.

## MACS2 output

At least three result files are generated in the MACS2 results directory (the number can change with your choice of options):

- 1) \*\_peaks.narrowPeak - a list of the called peaks in narrowPeak format (an extended bed format, the same format JAMM uses)
- 2) \*\_peaks.xls - a report on the data, the options and parameters used and calculated, and the full list of peaks with information not present in the narrowPeak format such as column names, peak summit and fold enrichment
- 3) \*\_summits.bed - the peak summit for each peak *i.e.* the coordinate with the greatest signal (number of sequenced reads)

*Note:* bed files are 0-based, thus the first coordinate number is NOT the true start. Software asking for bed format knows that this is part of the bed file format. If you use or create a bed file, you must follow the 0-based format.

### What does the peak data look like ?

*e.g.*

Count how many peaks were called -> use `wc -l`

Take a look inside the file (use `head` and `tail`). Is it sorted by coordinate or by peak score?

If you don't remember the bed file format, then google search "bed file format".

What is the highest peak score and the lowest? (use `head` and `tail`)

If it's not sorted by peak score you'll have to use `sort` and use the option for numeric sort

If you are comfortable plotting in R, plot a distribution of the peak scores.

How wide are the peaks? Again you could plot the peak width distribution

get width using coordinate columns: `awk '{OFS="\t"; print $0, $3-$2 }' yourfile.narrowPeak > tmpfile`

### Recommendation:

The following papers are useful to read about peak callers:

PMID: 22522655. Systematic evaluation of factors influencing ChIP-seq fidelity. 2012

PMID: 25223640 JAMM: a peak finder for joint analysis of NGS replicates. 2015

PMID: 18798982 Model-based Analysis of ChIP-seq (MACS). 2008

There hasn't been a paper specific to MACS2.

## S2.2 Filtering peaks against the blacklist regions

There are regions in the genome found by ENCODE and modENCODE to have artificially high signal *i.e.* "excessive unstructured anomalous reads mapping". They are believed to be artifacts and have been assembled into a "blacklist" by ENCODE. It is relatively standard protocol to filter out peaks that intersect with these black list regions, unless you have specific reason not to.

*Google search:* blacklist encode; *or search with extra terms:* blacklist encode anshul kundaje

Change the path and file names where needed *e.g.* if you chose ATAC-seq then you will use `macs2_atac_peaks.narrowPeak` instead of `macs2_chip_peaks.narrowPeak`:

```
peaks=$mydir/peak_results/macs2_chip_peaks.narrowPeak
```



```
blacklist=$mydir/chipseq/encode_blacklist_hg19_wgEncodeDacMapabilityConsensusEx  
cludable.bed  
outfile=$mydir/peak_results/macs2_ chip _peaks.noblacklist.narrowPeak
```

```
bedtools intersect -v -a $peaks -b $blacklist > $outfile
```

You will look at these again in the next section

End of section 2.

## Section 3 – using IGV to visualize data

### 3.0. Open IGV

#### 3.1. Tips on using IGV

Using IGV (Integrated Genome Viewer) to visualize data from the aligned reads (bam files) through to the peaks called in the previous section (narrowPeak files).

Data visualization is used by researchers as a sanity check for their data, to discover errors and/or biases in data processing, and sometimes to discover trends that become interesting research questions.

Loading large datasets into IGV usually requires some level of compression, therefore the raw aligned reads will be loaded as BAM files (or bigWig format, but we don't have those today) to show the read pileups as smoothed densities.

Here you will use the provided full CTCF ChIP-seq and/or ATAC-seq datasets *i.e.* all chromosomes

## S3.0 Open IGV

**You will see the IGV graphical user interface (GUI) loading, give it a little time to open and load the genomes.**

To open IGV:

If your computer has IGV installed on it, you might be able to open it with a desktop link.

Alternatively, from the command line there is a script called `igv.sh` that contains the command to start the java based IGV. You can see how simple the script is with `less igv.sh` or `head -15 igv.sh`

### Open from command line

`sh igv/igv.sh &` # the "&" at the end puts IGV into the background so you can keep working in the terminal you have open. You can use the `jobs` command to see you have something running in the background.

OR send IGV output to a log file

`sh igv/igv.sh &> igv.log &` # put the IGV "report" into a file to read later.

**OR to control the memory that java can use: *i.e.* -Xmx5G**

`java -Xmx5G -jar igv/igv.jar &> igv.log &`

*# calling IGV with `java -Xmx5G` allows you to tell IGV the maximum memory you will let it have, in this case 5G. If IGV gets very slow, you will want to restart it with more memory. If you look inside the `igv.sh` file you will see this*

The very last `&` at the end of the line puts the running of IGV into the background so that you do not see it. If you had forgotten the `&` then you will be watching all the IGV output and can't work. To continue working on command line, open a new Terminal tab (a new shell) using `[ctrl][T]`. You can continue your work in the new shell, but you probably need to change to your working directory.

NOTE: if you want to stop IGV from running before it's even really started you can kill the command that you started with `[ctrl][c]`. Once IGV has started, be "polite" and use *File -> quit* from the IGV menu.

## S3.1 Using IGV

In the top bar, left side, make sure "Human hg19" is selected. Beside that you could also select a chr e.g. "chr21".

Load the GENCODE gtf annotation file, which has more transcripts than the RefSeq genes already loaded in IGV.

From IGV menu bar:

*File -> Load from file -> /path/genencode.v19.annotation.gtf*

For IGV you will use the provided full CTCF ChIP-seq and/or ATAC-seq datasets *i.e.* all chromosomes

```
ls $mydir/chipseq/chipseq_ctcf_full_datasets/*
```

```
ls $mydir/chipseq/atacseq_full_datasets/*
```

```
ls $mydir/chipseq/*blacklist*
```

Files for ChIP-seq (you don't have to load all):

<code>wgEncodeAwgTfbsBroadGm12878CtcfUniPk.narrowPeak</code>	<code># peaks on all chr</code>
<code>wgEncodeBroadHistoneGm12878CtcfStdAlnRep1.bam</code>	<code># aligned reads for 1 ChIP-seq CTCF</code>
<code>wgEncodeBroadHistoneGm12878ControlStdAlnRep1.bam</code>	<code># aligned reads for 1 ChIP-seq control</code>

Files for ATAC-seq:

<code>atacseq_mac2_peaks.narrowPeak</code>	<code># peaks on all chr</code>
<code>rep1_hg19_Aligned.out.coordSrt.1bp.reads.bam</code>	<code># aligned reads for ATAC-seq</code>
<code>rep2_hg19_Aligned.out.coordSrt.1bp.reads.bam</code>	

File of blacklist regions:

```
encode_blacklist_hg19_wgEncodeDacMapabilityConsensusExcludable.bed
```

### Load files into IGV:

Load the BAM files (the index files are already made for you, they don't need to be loaded into IGV):

From IGV menu bar: *File -> Load from file -> XXX.bam* # where XXX.bam is a specific bam

Load the MACS2 peak files (extension: narrowPeak or broadPeak), and blacklist file.

Take a look around, below are some tips for using IGV:

**Jump to a gene you know the name of e.g. CTCF or XIST, by putting the gene name in the IGV search bar. The search bar will start to list possible matches.**

**You might not see some data until you zoom in, as bam file data often doesn't show up until you narrow down to a 40-50kb region.**

### **Are the peaks overlapping regions with high signal as expected?**

\* For some types of data (for instance .bam files) you won't see anything in the data track if you look at a region bigger than ~46kb. You need to zoom in.

\* Zoom in and out with the + and – buttons.

\* Select a portion of the chromosome by right clicking and dragging your mouse on the coordinate line in the top bar.

**\* Right click on the left column in the viewer beside a track and you can try the following options:**

- 1) For the tracks that display the aligned reads from a BAM file, try the “Collapsed”, “Expanded”, and “Squished” options.
- 2) Rename your tracks to something more meaningful to you
- 3) Change the colour of one of the tracks

**\* Locally browse through the viewer by clicking in the middle of the window displaying the reads and dragging the window left or right.**

**\* Remove a track by right clicking on the track name in the left column, and choose “Remove track”**

**\* The \*.bam files will have both forward and reverse reads present. Display the forward and reverse reads as different colours:**

Right click on the track and choose “Colour alignments by...” -> read strand.

Try the other two options to see if they are useful:

“Sort alignments by -> read strand”

“Group alignments by -> read strand”.

**\* Keep an eye on the data range (tiny numbers on left upper side of track).** If there is low signal in the region (small data range) it might vertically fill up the track due to “autoscale”. You need to be aware of it so that you don't think “oh lots of signal !!” when actually there is a very low data range.

**\* Look at some of the ENCODE blacklist regions. Take some coordinates from the blacklist file and zoom out to see the local region.**

e.g.

head

```
$mydir/chipseq/encode_blacklist_hg19_wgEncodeDacMapabilityConsensusExcludable.bed
```

**\* If you have several data sets open in IGV and you want to keep this session for future use, you can save it:**

File -> Save session...

When you come back another day you can open it again: File -> Open session...

However, if you move your data to another location, the session won't be able to open it

**\* For future use: explore other options that change how IGV displays data from the menu bar:**

View -> Preferences -> Alignments

End of section 3.

**Of the following three sections you will only read about IDR so that you know it exists. The last two are small ways to start looking at your peak data.**

Section 4 – Irreproducible Discovery Rate – thresholding using replicates

Section 5 – Peaks relative to genomic features (using bedtools)

Section 6 – Does ChIP-seq intersect ATAC-seq

## READ-ONLY

### Section 4 – The Irreproducible Discovery Rate (IDR) for thresholding

#### 4.1 IDR – what it's about – READ ONLY

The number of peaks called by peak calling software changes dependent on the options and parameters used. A location can be considered a region of interest or not depending on your method of analysis, your chosen cutoffs, and whether you have consistently good replicates or not.

**One way to threshold your peaks is to consider those that are reproducible between your replicates.** Here is an overview of how to do this using the IDR framework (Irreproducible Discovery Rate). This is a statistical approach, for choosing peaks of confidence vs simply keeping all peaks that occur in both replicates regardless of how well the seq. reads support the peaks.

## READ-ONLY

### S4.1 Irreproducible Discovery Rate (IDR)

- **IDR (Irreproducible Discovery Rate):** <https://github.com/nboley/idr>

*“The IDR (Irreproducible Discovery Rate) framework is a unified approach to measure the reproducibility of findings identified from replicate experiments and provide highly stable thresholds based on reproducibility.”*

For ATAC-seq, this means that IDR tries to provide an estimate of which peaks are reproducible between the replicates. In that case, one would need to call peaks separately for each replicate and call peaks on the replicates combined (“combined” means pooled together). What the IDR framework will do is use the peak scores from the replicates to see whether peaks that overlap in the replicates have similar ranks. IDR program will tell you for example that at a threshold of 0.02 irreproducible discovery rate (2% of the peaks might be irreproducible), you should use “n” number of peaks. You can use this number to threshold your “combined” peak list. Optionally you can also use the reproducible peaks and intersect with them with your combined peak list, meaning take the peaks in the combined peak list that intersect your reproducible peaks. IDR results are stringent. You can choose how stringent you want to be.

(older version <https://sites.google.com/site/anshulkundaje/projects/idr> ; a more recent version and more complex: <https://github.com/nboley/idr> )

To run the IDR program you need a large set of peak calls for each replicate, so that you have true and false positives. This is because IDR needs to see examples of peaks that are irreproducible and have low ranks.

MACS2 should be run with a relaxed p-value:  $-p\ 0.05$  to get false positives. For IDR, it is recommended to use the top 100-125K peaks from MACS2 (as described on the IDR website), although 150K seems ok in data I have run.

If you want to use IDR, the old website for IDR is worth reading as it has fuller explanations for the method and is found here: <https://sites.google.com/site/anshulkundaje/projects/idr>

This is the IDR used in much of ENCODE and is downloadable from that page.

End of section 4.

Sections 5 and 6 are single examples for preliminary assessment of your peaks, now that you have them. Distributions of values associated with your data can give you a feel for how your data looks across the genome. *i.e.* how close are your peaks to the transcription start of genes, are they in introns or near splice sites, are they in promoters or always distal, do your ChIP-seq peaks fall within open chromatin regions.

## Section 5 – Where are peaks relative to genome annotation

### S5 How close are your peaks to genomic features?

You have data, and now you want to know something more about it. Do the peaks tend to be proximal or distal relative to gene **transcription start sites** (TSSs)? For ChIP-seq this value will change depending on the ChIP'd protein. For ATAC-seq approximately 1/3 of your peaks will be within 500bp of a GENCODE TSS.

Here you will again use the provided full CTCF ChIP-seq or ATAC-seq datasets *i.e.* all chromosomes

**Peak proximity to genomic features, here we use the TSS as an example feature:**

GTF files contain annotations of transcripts, isoforms, lncRNA, miRNA, snRNA, pseudogenes, and every other sequence level annotation you can think of.

You can see a description of the GTF format here from GENCODE:  
[http://www.genencodegenes.org/data\\_format.html](http://www.genencodegenes.org/data_format.html)

You should later take a look into the FANTOM CAT annotation for human. It appears to be very good.

<http://fantom.gsc.riken.jp/cat/>  
<http://www.nature.com/nature/journal/v543/n7644/full/nature21374.html>

# Provided is a processed GENCODE annotation file for you to work with (processed using Unix commands). It is no longer in GTF format. It has only those lines that represent a transcript, and the coordinates were reduced to the 1<sup>st</sup> base-pair of the transcript *i.e.* the transcript start. The file was then processed to make the first 3 columns the **chr** **start** and **end** coordinates, and a bunch of columns were removed that we don't need. You will use this file to find the TSS closest to each set of peaks.

```
anno=$mydir/chipseq/gencode.v19.annotation.col3Transcript.sorted.uniqtstart.bed
```

Take a quick look inside the annotation file to see what's there (it's a small enough file to do this);

```
less $anno          # use q to quit
#OR
head $anno
```

#### 5.1) Find the closest TSS annotation to the peaks of interest:

If you can't remember what's in the data directories, list what is in there:

```
ls $mydir/chipseq/chipseq_ctcf_full_datasets/  
ls $mydir/chipseq/atacseq_full_datasets/
```

ChIP-seq CTCF peaks:

```
wgEncodeAwgTfbsBroadGm12878CtcfUniPk.narrowPeak # all chr
```

ATAC-seq peaks:

```
atacseq_macs2_peaks.narrowPeak # all chr
```

bedtools closest expects files to be sorted, and it will complain if the files are not sorted (likely will happen with the below... next step will address that)

```
bedtools closest -h # look at the options for closest
```

# Use your data of choice, and **put the path and filename** in the command below

#ChIP

```
bedtools closest -a peakfile -b $anno -D b >  
$mydir/peak_results/chipseq_peaks.closestTSS
```

#ATAC

```
bedtools closest -a peakfile -b $anno -D b >  
$mydir/peak_results/atacseq_peaks.closestTSS
```

# IF you get a warning, then this time sort the file (using a subshell *i.e.* `<(sort stuff)` ) and direct the sorted data to bedtools.

You can use either the ChIP-seq and/or ATAC-seq. I've only written the ChIP-seq file.

*e.g.*

```
chip=$mydir/chipseq/wgEncodeAwgTfbsBroadGm12878CtcfUniPk.narrowPeak
```

```
bedtools closest -a <(sort $chip) -b $anno -D b >  
$mydir/peak_results/chipseq_peaks.closestTSS
```

## 5.2) How many of the peaks are within 500bp or 3kb of a TSS?

**A)** first find out how many columns there are in the file. The distance-to-TSS value is in the last column, so the count of columns will tell us the number of the distance column.

# use awk to count the columns and assign the value to a variable

```
distcol=$(awk -F $'\t' '{print NF; exit}'  
$mydir/peak_results/chipseq_peaks.closestTSS )
```

```
echo $distcol
```

*Explanation:* `-F $'\t'` tells awk that the data is tab delimited. `print NF` tells awk to print the number of fields (NF) in each line. Including `exit` forces awk to stop immediately after it parses the first line.



**B) now to find how many peaks are within 500bp or 3kb of a TSS**

# give awk the column number from above, and then ask if below 3000 or 500

```
awk -v n=$distcol '{d=$n; if(d<0){d=d*-1}; if(d<=3000) print $0}'  
$mydir/peak_results/chipseq_peaks.closestTSS >  
$mydir/peak_results/chipseq_peaks.closestTSS.3kb
```

Repeat the above but using 500 instead of 3000 (remember to change output file name)

*Explanation of awk:* `if(d<0){d=d*-1};` we want the absolute value of the distance, this is one way to do it. If your number is less than 0 then multiply it by -1 to make it positive.

How many peaks are within 3kb of a TSS? 500bp?

```
wc -l $mydir/peak_results/chipseq_peaks.closestTSS.3kb
```

You could upload `chipseq_peaks.closestTSS.3kb` into R and plot a density or histogram distribution of the distances.

You can use a full GENCODE GTF file to assess (or filter by) peak proximity relative to any genomic annotation, such as intronic, exonic, TSS, 3' UTR, protein-coding genes etc. GENCODE is not in bed file format, so if your tools don't handle GTF (e.g. bedtools), your first step will be to extract whatever annotation you are interested in, to the bed format (most minimal is: chr start end).

End of section 5.

## Section 6 – Intersection of ChIP-seq peaks with ATAC-seq

### S6 Are CTCF ChIP-seq peaks within open chromatin regions?

Here you will continue to use the provided full CTCF ChIP-seq or ATAC-seq datasets i.e. all chromosomes

```
ls $mydir/chipseq/chipseq_ctcf_full_datasets/
```

```
file: wgEncodeAwgTfbsBroadGm12878CtcfUniPk.narrowPeak # all chr
```

```
ls $mydir/chipseq/atacseq_full_datasets/
```

```
file: atacseq_macs2_peaks.narrowPeak # all chr
```

#the path+filenames are long, so you can assign to a variable if you want e.g.

```
ctcf=$mydir/chipseq/chipseq_ctcf_full_datasets/wgEncodeAwgTfbsBroadGm12878CtcfUniPk.narrowPeak
```

```
atac=$mydir/chipseq/atacseq_full_datasets/atacseq_macs2_peaks.narrowPeak
```

#### # Are the locations called from CTCF ChIP-seq in agreement with open chromatin ?

```
bedtools intersect -wa -a $ctcf -b $atac >
$mydir/peak_results/intersect_chipVsAtac.txt
```

Number peaks in original ChIP-seq peak file

```
wc -l $ctcf
```

*How many of CTCF ChIP-seq peaks intersect ATAC-seq peaks?*

*→ Cut first 3 columns because they are chr start end, and then count number of unique entries.*

*(Sometimes a peak can intersect something more than once, so you want unique counts of peaks)*

```
cut -f1-3 $mydir/peak_results/intersect_chipVsAtac.txt | sort | uniq | wc -l
```

Does this seem reasonable to you? (just to think about)

*Extra:*

You could ask how often there is more than one CTCF in a single ATAC-seq peak

How would you do this? *Hint:* start as above with the intersect but use uniq with an option

#### # reverse the question (although we don't really have a reason to do so), how many open chromatin regions contain a CTCF peak?

```
bedtools intersect -wa -a $atac -b $ctcf >
$mydir/peak_results/intersect_atacVsChip.txt
```

```
wc -l $atac
```

```
cut -f1-3 $mydir/peak_results/intersect_chipVsAtac.txt | sort | uniq | wc -l
```

### Extras (to figure out yourself):

# You might want to do a random shuffle of the CTCF peak locations, and intersect this with the ATAC-seq regions to see if the intersection count changes. Instead of random across the entire genome you could specify another file that represents promoter regions (perhaps derived from a GTF annotation file), or one that provides negative regions that the shuffle should **not** go to e.g. blacklist regions, repetitive regions.

```
bedtools shuffle  
chrsize=$mydir/chipseq/hg19.chrom.sizes
```

# if you want to extend your peaks a bit to catch ChIP-seq and ATAC-seq regions that are very close but not quite overlapping you could use *bedtools slop* to create a new bedfile, or simply use *awk* on the command line to adjust the left and right coordinates (rarely will you go beyond the edge of a chromosome, but be aware you might).

```
bedtools slop  
chrsize=$mydir/chipseq/hg19.chrom.sizes
```

OR

```
head $ctcf  
awk '{OFS="\t"; print ..... }' $ctcf > $ctcf".wider.bed"
```

End of section 6.

-----

This is a small introduction. There is so much more one can do! From this tutorial you hopefully have an idea about how to process your own data and some basic skills to continue exploring your data before going on to deeper questions.

*Aside:*

If you want to generate smoothed bigWig files of your data's genome coverage, instead of bam files, look into *deepTools* -> *bamCoverage*. This tool has several normalizations. If you have one sample or several with the same sequencing depth then you can choose as you will e.g. *RPKM*. To visually compare multiple samples with different sequencing depths, then use *normalizeTo1x*. *deepTools* also has *bamCompare* for the case of normalizing to controls e.g. ChIP-seq.

The smoothing approach essentially looks at a bin of chosen size, e.g. 20bp, and then a bigger window that surrounds the bin, e.g. 90bp, called the smoothing length. The bins are side-by-side across the genome. The windows will end up overlapping. The value of the bin is the average read count within the window normalized by whichever approach you chose.

There are other tools that will give a smoothed bigWig, this is one to start with. However, bam files should tell the same story if you keep your eye on the signal to noise ratio when dealing with multiple samples.

**Remember:** Google is your friend. So many web forums with people asking exactly the questions you have, you will always find an answer somewhere.