ATAC-seq data processing. From FASTQ files to peak calling

Author: Tomàs Montserrat Ayuso

Introduction

ATAC-seq experiments make it possible to study the state of the chromatin of the cells of a biological sample. These experiments make use of the transposase Tn5, which binds to DNA in positions of the genome where it is accessible. In this way, epigenetic mechanisms that regulate gene expression can be investigated.

In this post we will see how to process the data generated in an ATAC-seq experiment, from the *fastq* files with the reads from the fragments generated by the Tn5 transposase, to the identification of the accessible regions of the genome (peak calling). We will work with the data from the study entitled Chromatin accessibility dynamics and a hierarchical transcriptional regulatory network structure for plant somatic embryogenesis. The researchers of this work wanted to study the so-called somatic embryogenesis in *Arabidopsis thaliana* both at the transcriptional and epigenetic level. To do this, they performed ATAC-seq, ChIP-seq and RNA-seq experiments in somatic embryos induced at different time-points and in seedlings. To illustrate the analysis of ATAC-seq data, we will work with ATAC-seq data from embryos at 0 h and seedlings without induction medium. The *fastq* files can be downloaded from this website.

To do this, we will follow these steps:

- Quality control of the reads of the *fastq* file using the *fastqc* program.
- Trimming adapters and poor quality positions with *Trim Galore*.
- Filtering out aligned reads (those in the mitochondrial and chloroplast genome, in blacklist regions, duplicate reads and low-quality alignments) with *samtools*, *bedtools* and *picard*.
- Evaluation of the quality of the ATAC-seq data by studying the similarity of the biological replicates, generating enrichment graphs of reads in the open areas of the chromatin and in the TSS; and observing the length distribution of the sequenced fragments. We will do this step using tools from *deepTools* and the package for R *ATACseqQC*.
- Identification of the accessible regions of the chromatin (peak calling) in each sample.

Since we will generate many files during the analysis, it is important to be clear about the structure of our working directory. In our case we will work following this organization:

```
.
├── alignment
│   ├── CRR134566
│   ├── CRR134567
```

```
|   ├── CRR134582
|   ├── CRR134583
|   ├── reference
├── blacklist_regions
├── data
|   ├── CRR134566
|   ├── CRR134567
|   ├── CRR134582
|   ├── CRR134583
├── quality
|   ├── alignment
|   |   ├── bowtie2
|   |   ├── duplicates
|   |   ├── seq_depth
|   |   └── stats
|   ├── ATACseqQC
|   |   └── CRR134566
|   ├── correlation_plot
|   ├── fastqc
|   ├── fastqc_trimmed
|   ├── fingerprint_plot
|   ├── fragment_size_distribution
|   ├── multiqc
|   └── picard_markduplicates
├── results
├── r_objects
├── scripts
└── trimmed_data
    ├── CRR134566
    ├── CRR134567
    ├── CRR134582
    └── CRR134583
```

The initial *fastq* files can be found in the data directory. In this entry we will see how to process sample CRR124566 as an example.

Initial quality control

As in most bioinformatics analyses, the first thing we will do is the quality control of the reads generated during sequencing using the *fastqc* program. In the case of the ATAC-seq data, we will mainly be interested in the quality of the bases per position, the content in GC, the distribution of the length of the reads, the levels of duplication and the content of adapters. Let's run the *fastqc* program:

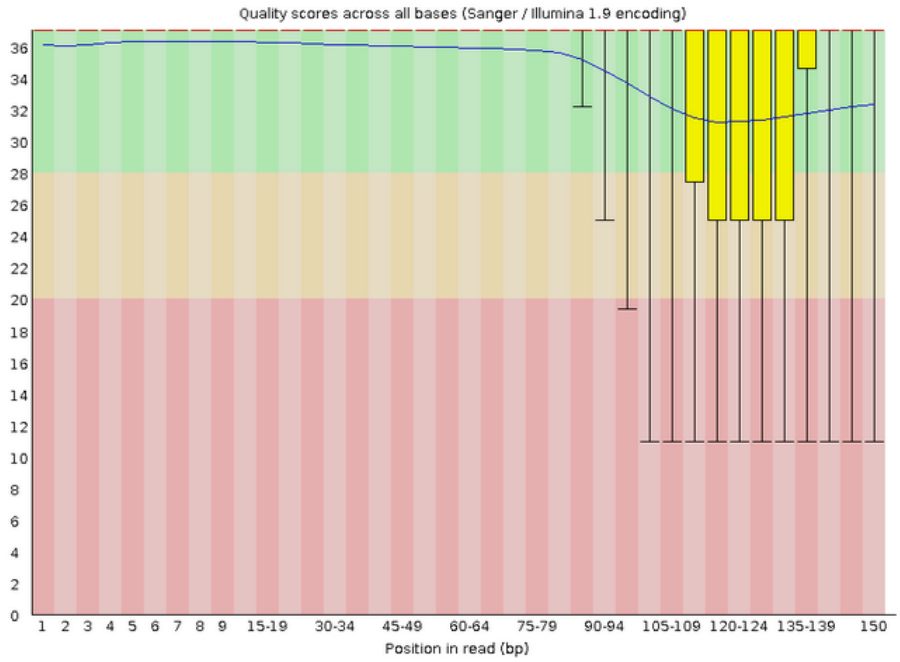
```
fastqc data/CRR124566/CRR124566_f1.fq.gz data/CRR124566/CRR124566_r2.fq.gz \
-o quality/fastqc/
```

The GC content of the reads does not show significant deviations from the theoretical distribution, so we can rule out any unexpected library contamination or any other bias. Let's discuss the most relevant details of the output:

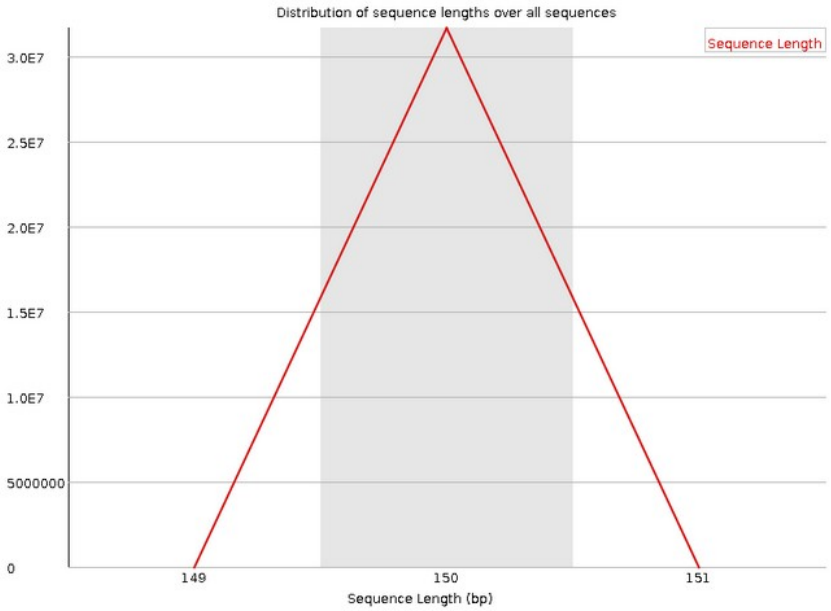
Base quality by position, read length distribution and adapter content

We grouped these three quality metrics because, in the case of ATAC-seq data, they are closely related. This is the *fastqc* output for the first sample (forward strand) for each of the three metrics:

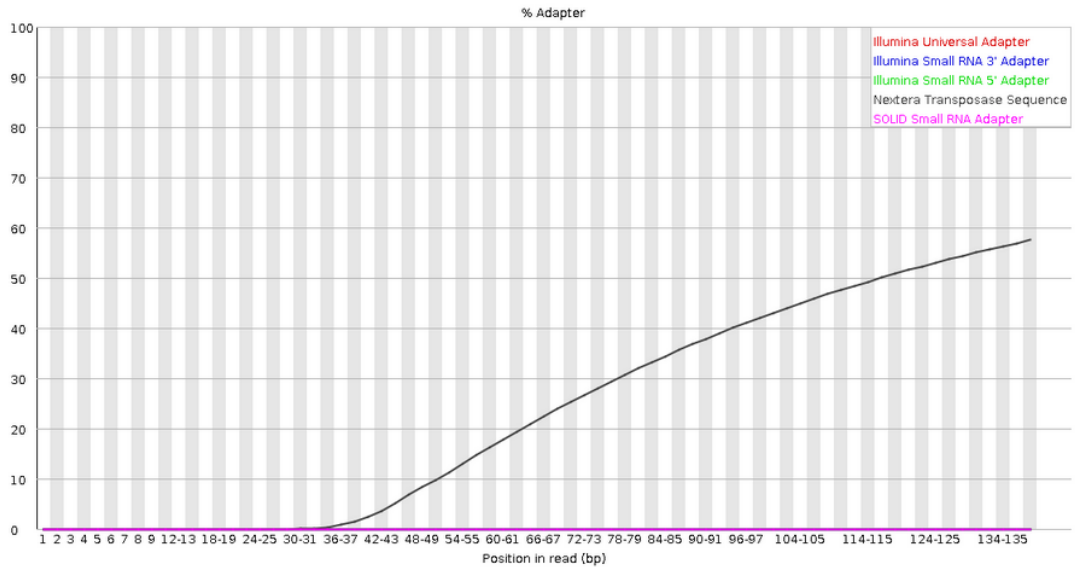
• Quality of bases by position:



- Distribution of the length of the reads:



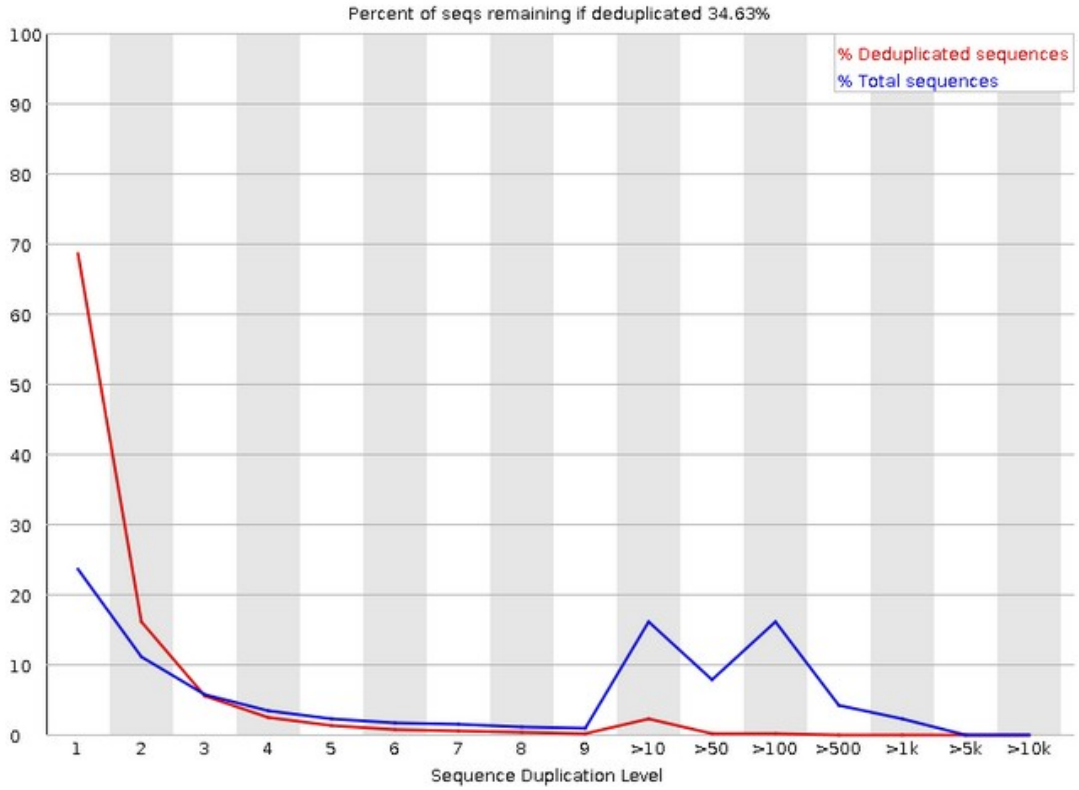
- Content of adapters



In the ATAC-seq experiments we expect an enrichment of short DNA fragments, below 100 base pairs. Since the obtained reads are 150 base pairs, many of the DNA fragments are shorter than the length of the read, so these short fragments will be sequenced completely and, when reaching the end, the adapter will also be sequenced. This is why we see that the proportion of long reads containing the adapter increases with length. If the read is short enough, not only will the adapter be sequenced, but the incorporation of bases will no longer be consistent with the complementary sequence, causing the degradation of the quality of the bases at the 3 end ' of the reads.

Duplication levels

The levels of duplication shown by this dataset are very high. Although the causes should be thoroughly investigated, one possible reason is that the genome of Arabidopsis thaliana is relatively small (135 million base pairs) compared to that of Homo sapiens (3054 million base pairs). Since it is recommended to generate more than 50 million base pairs for human to find the open regions of chromatin and we have more than 30 million reads for a much smaller genome, it is not unusual to find high duplication rates in these data because the same DNA fragment may have been captured several times.



Trimming of adapters and bases of poor quality

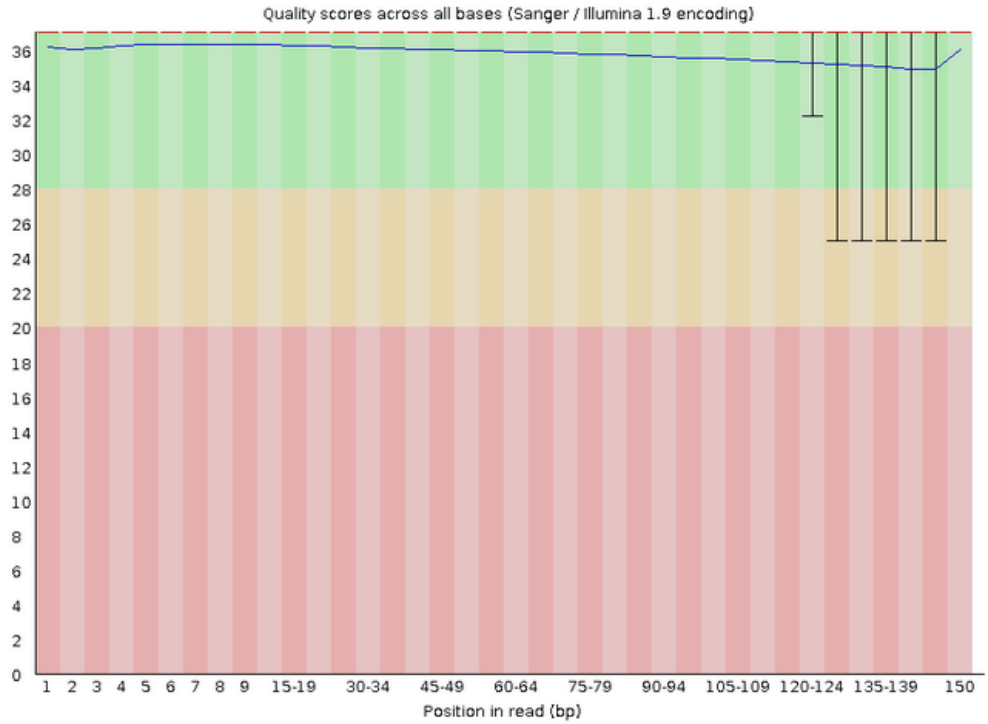
In the ATAC-seq data and as we have seen from the *fastqc* report, it is very important to remove adapter sequences and poor quality bases from the 3' end. These two operations can be easily done with the *Trim Galore* program, which is able to automatically detect the Nextera adapter used for ATAC-seq:

```
trim_galore \
--phred33 \
--quality 20 \
--paired \
--cores 6 \
--output_dir trimmed_data/CRR124566/ \
data/CRR124566/CRR124566_f1.fq.gz data/CRR124566/CRR124566_r2.fq.gz
```

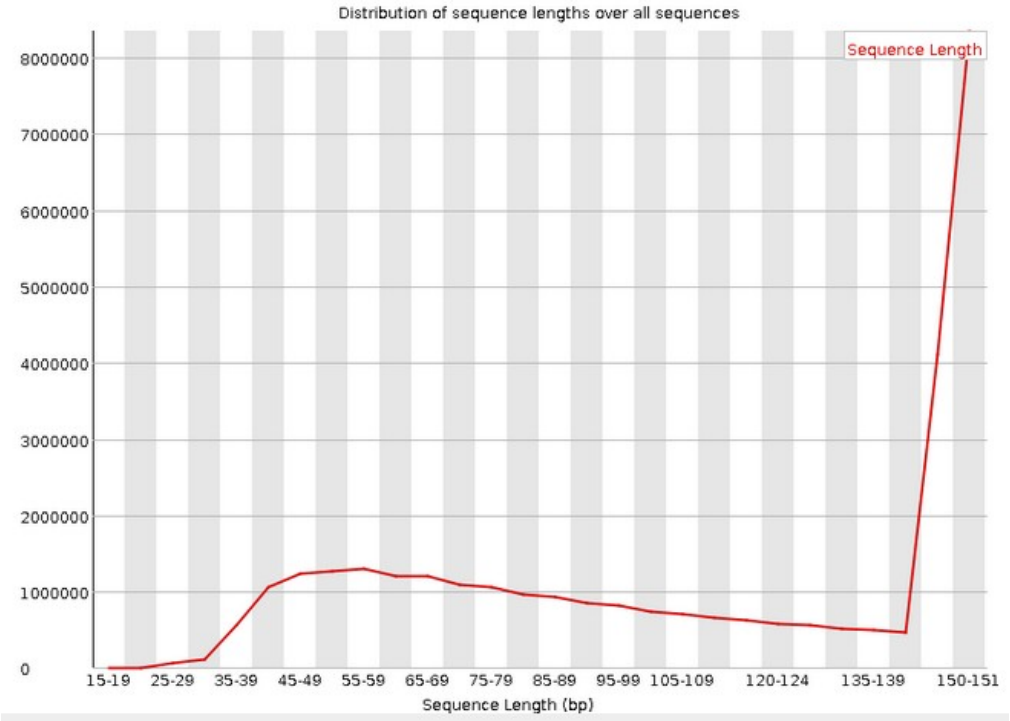
In the above command, we are telling *Trim Galore* that the base quality encoding is in Sanger format (`--phred33`), that we want to remove bases from the 3' end that are not of a quality greater than 20 (`--quality`), that we have paired reads (`--paired`) and that we want to use 6 processors (`--cores`).

Once we have the reads without adapters or bad quality bases at the 3' end, it is convenient to run the *fastqc* program in the new *fastq* file to check that the problems we mentioned before have disappeared:

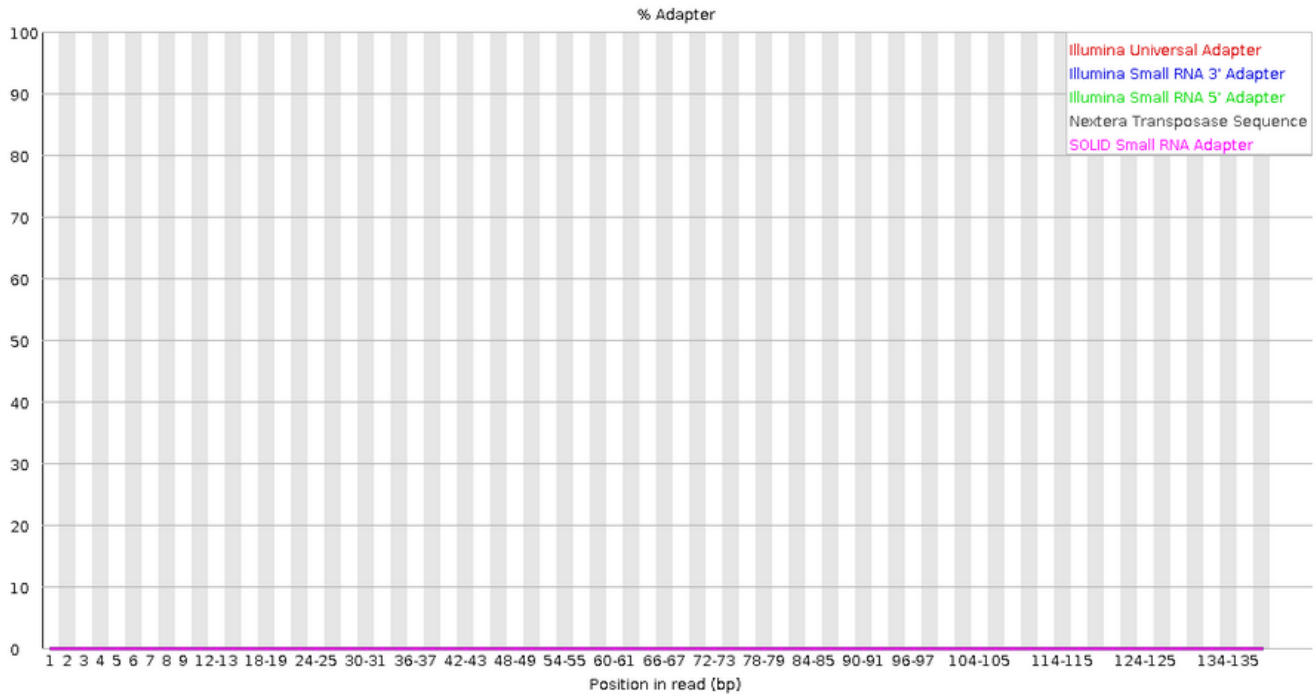
- Quality of the bases by position: thanks to the trimming of the poor quality bases of the 3' end, we no longer see this feature.



- Read length distribution: we no longer see all reads with the same length. In fact, most reads are less than 150 bp in length. We see an enrichment of short reads, between 35 and 100 bp, which is consistent with the expected fragment length in nucleosome-free regions (NFR; <100 bp), where there should be an enrichment of reads. It is obvious that there are also many reads that maintain the length of 150 bp. We also expect this bimodal distribution of read lengths, since, although many reads come from fragments in NFR, other fragments will be longer because they represent regions containing one or more nucleosomes.



- Adapter content: as expected, reads no longer contain adapter sequences.



Alignment with the reference genome

We can now align the reads with the reference genome to find out their origin. The first step is to download the genome in *fasta* format:

```
wget https://ftp.ensemblgenomes.ebi.ac.uk/pub/plants/release-57/fasta/arabidopsis_thaliana/dna/
Arabidopsis_thaliana.TAIR10.dna_sm.toplevel.fa.gz
gunzip Arabidopsis_thaliana.TAIR10.dna_sm.toplevel.fa.gz
```

We also need to create the index for the *bowtie2* aligner to use:

```
bowtie2-build alignment/reference/Arabidopsis_thaliana.TAIR10.dna_sm.toplevel.fa \
alignment/reference/tair10
```

Finally, we align the reads with the *Arabidopsis thaliana* genome:

```
bowtie2 -p 6 -q \
--very-sensitive \
-X 500 \
-x alignment/reference/tair10 \
-1 trimmed_data/${SAMPLE}/CRR124566_f1_val_1.fq.gz \
-2 trimmed_data/${SAMPLE}/CRR124566_r2_val_2.fq.gz \
2> quality/alignment/bowtie2/CRR124566_bowtie2_stats.txt \
| samtools view -u - \
| samtools sort - > alignment/CRR124566/CRR124566_sort.bam
```

The `--very-sensitive` option is a *preset* (a set of options) that causes the alignment to be slower, but also more accurate. `-X 500` is the maximum size of the fragments for which the pair of reads are considered valid. For

ATAC-seq, values of 1000 would also be suitable (because we expect there to be a small portion of the fragments containing multiple nucleosomes), but it slows down the alignment a lot.

We direct the output of the program directly to *samtools view* and *sort* to obtain the aligned reads ordered according to the coordinates and in *bam* format. The *-u* option changes the default resulting file to *bam* type. The hyphen (-) simply indicates that the program should use the standard out of the previous program.

Once we have the *bam* file, we'll create an index needed by many downstream programs:

```
samtools index -b alignment/CRR124566/CRR124566_sort.bam
```

We can take a look at the alignment statistics generated by *bowtie2*:

```
cat CRR134566_bowtie2_stats.txt
31691705 reads; of these:
  31691705 (100.00%) were paired; of these:
    555754 (1.75%) aligned concordantly 0 times
    19859907 (62.67%) aligned concordantly exactly 1 time
    11276044 (35.58%) aligned concordantly >1 times
    ----
    555754 pairs aligned concordantly 0 times; of these:
      83621 (15.05%) aligned discordantly 1 time
    ----
    472133 pairs aligned 0 times concordantly or discordantly; of these:
      944266 mates make up the pairs; of these:
        627737 (66.48%) aligned 0 times
        185753 (19.67%) aligned exactly 1 time
        130776 (13.85%) aligned >1 times
99.01% overall alignment rate
```

99.01% of the reads have been aligned, that is to say, practically all. 62.67% of the reads aligned with a single region of the genome, while only 15% of the reads discordantly aligned with their partner did so in a single region of the genome. The difference between concordant and discordant has to do with whether the two reads of an aligned pair meet the requirements. If, for example, the two reads were separated by more than 500 bp (specified in the `-X` option), they would be classified as discordant.

A percentage of 80% of reads aligned to the genome in a single position (uniquely mapped reads) is usually taken as a reference for good quality ATAC-seq data. Although this percentage is not quite the percentage of uniquely mapped reads due to the way *bowtie2* returns the alignment, we can take it as a reference. 62.67% is relatively far from 80%, so it seems that we have suboptimal quality data. However, we may continue to process the data.

Selection of aligned reads

In order to work with the appropriate reads for ATAC-seq analyzes of interest (starting with peak calling), we must first select them, filtering them properly. We will remove reads aligned to the mitochondrial and chloroplast genome; to regions of the genome known as blacklist regions; duplicates during PCR; not uniquely mapped.

- Removal of reads aligned to the mitochondrial and chloroplast genome:

Mitochondrial and chloroplast genomes are not the target of ATAC-seq studies. In addition, given that these genomes are more accessible than nuclear, a large part of the reads usually map to these genomes, which is why it is appropriate to remove them from the dataset. We will do this step with *samtools* and *grep*. We will generate a new intermediate *bam* file using *samtools view* with the `-b` option and redirecting the output:

```
samtools view -h alignment/CRR134566/CRR134566_sort.bam \
| grep -v "Mt" \
| grep -v "Pt" \
| samtools view -b - > alignment/CRR134566/CRR134566_sort_no_org.bam
```

- Removal of reads aligned to blacklist regions:

The blacklist regions are regions where it has been empirically found that a large number of reads are aligned in an artifactual way, impairing the interpretation of the results. We will use the list of blacklist regions created by Yin et al. in 2021 in the study H2AK121ub in Arabidopsis associates with a less accessible chromatin state at transcriptional regulation hotspots. We will use the *bedtools intersect* program for this task:

```
bedtools intersect \
-v -abam alignment/CRR134566/CRR134566_sort_no_org.bam \
-b blacklist_regions/blacklist.bed \
| samtools view -b - > alignment/CRR134566/CRR134566_sort_no_org_black.bam
```

The `-v` parameter causes the program to return the reads that are not in these regions. To use our *bam* file, we use `-abam`. As a second file we use `-b` and indicate where the file is located in *bed* format with the blacklist region.

- Removal of duplicate reads:

Duplicate reads are those that have mapped to the same genome coordinates and to the same DNA chain. These reads most likely come from duplications during PCR and are technical artefacts, so they should be removed from the analysis. We will do this with *MarkDuplicates* from the *picard* suite:

```
java -jar ${HOME}/picard/picard.jar MarkDuplicates \
--INPUT alignment/CRR134566/CRR134566_sort_no_org_black.bam \
--OUTPUT alignment/CRR134566/CRR134566_sort_no_org_black_dup.bam \
--METRICS_FILE quality/alignment/duplicates/CRR134566_duplicates.txt \
--REMOVE_DUPLICATES true
```

The `--REMOVE_DUPLICATES` option tells the program we want to return the *bam* file without duplicates.

- Removal of non-uniquely aligned reads:

In order to stay only with those reads aligned in a single position of the genome, we will use the map quality of each read returned by *bowtie2* in the *bam* file. This quality is proportional to the probability that that read comes from that region of the genome. In other words, the higher the map quality, the more unique that alignment is.

With *samtools* we will filter the aligned reads to stay only with those with a map quality of 30 or higher (on a scale from 0 to 42):

```
samtools view -h -b -q 30 alignment/CRR134566/CRR134566_sort_no_org_black_dup.bam \
| samtools view -b - > alignment/CRR134566/CRR134566_sort_no_org_black_dup_q.bam
samtools index -b alignment/CRR134566/CRR134566_sort_no_org_black_dup_q.bam
```

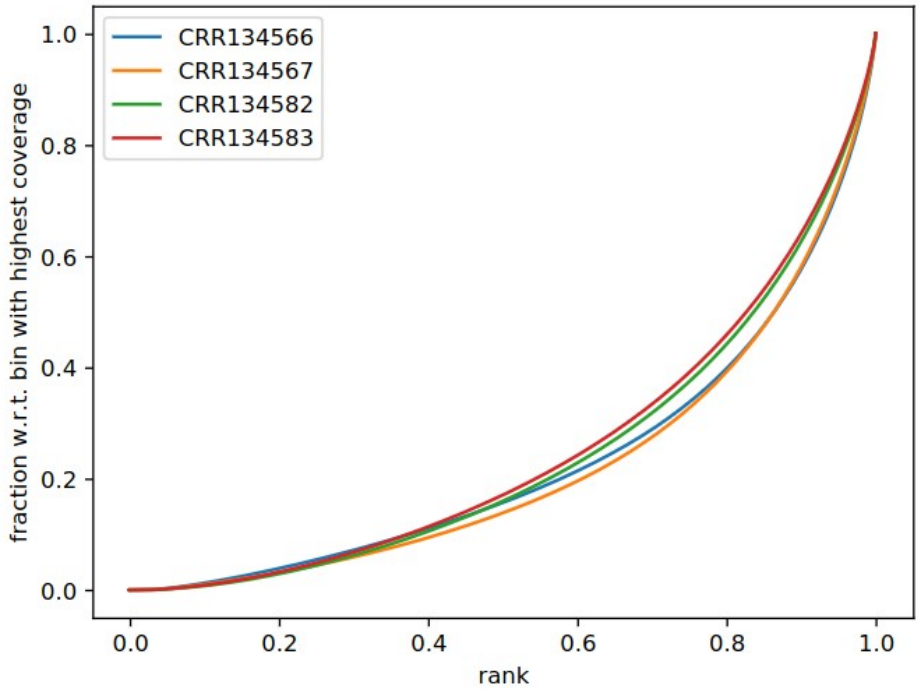
Assessment of ATAC-seq data quality

So far we have evaluated the quality of the reads and the alignment with the reference genome. In this section we will see how to study the quality of ATAC-seq data. That is, the signal-to-noise ratio. To do this, we will explore four indicators: the cumulative enrichment (or BAM fingerprint), the grouping of the replicates, the distribution of the length of the DNA fragments and the signal at the TSS (transcription start site) of the fraction of fragments corresponding to nucleosome free regions (NFR) and those corresponding to a single nucleosome.

BAM fingerprint

The *fingerprint plot* is a way of studying the signal quality of our ATAC-seq data. We can make this graph with the `plotFingerprint` function of the *deepTools* suite. By default, the program takes 500000 random regions of the genome of length 1000 (`--binSize`) and sums all overlapping reads. The values are sorted and the accumulated sum is displayed in a line graph. We will view this metric for samples CRR134566, CRR134566, CRR134582, and CRR134583. The first two correspond to embryos, while the last two to seedlings:

```
plotFingerprint --bamfiles \
alignment/CRR134566/CRR134566_sort_no_org_black_dup_q.bam \
alignment/CRR134567/CRR134567_sort_no_org_black_dup_q.bam \
alignment/CRR134582/CRR134582_sort_no_org_black_dup_q.bam \
alignment/CRR134583/CRR134583_sort_no_org_black_dup_q.bam \
 --binSize=1000 --plotFile quality/fingerprint_plot/samples_cumulative_enrichment.pdf \
 --labels CRR134566 CRR134567 CRR134582 CRR134583 -p 4
```



The more the curves resemble a rectangle, the stronger the signal from the ATAC-seq experiment. More reads are accumulated in a smaller fraction of the genome. Our ATAC-seq samples generate a fingerprint plot that is quite off the diagonal, so we expect to be able to easily detect the biological signal.

Assessment of similarity between biological replicates

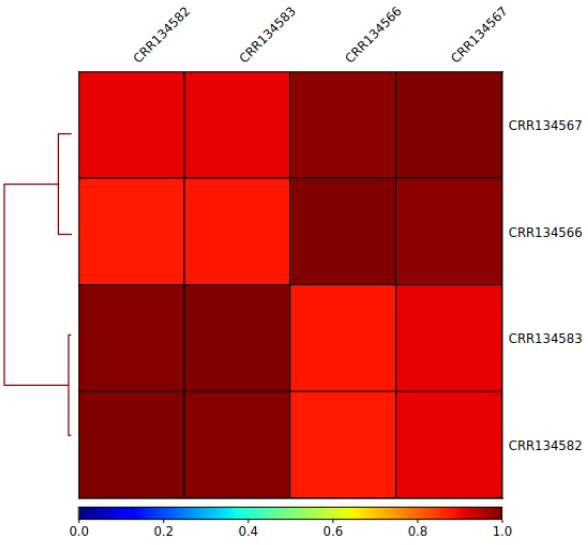
As in any experiment, we expect replicates from the same condition to be more similar to each other than to the rest of the replicates from the other conditions. One way to assess the similarity between replicates is to calculate the correlation between the samples and use this metric as a distance measure to perform hierarchical clustering of the samples. We will show the results for the samples in the previous section.

The `multiBamSummary bins` function of *deepTools*, divides the genome into the units specified in `--binSize` and counts the reads that come from each fraction in each sample:

```
multiBamSummary bins --bamfiles \
alignment/CRR134566/CRR134566_sort_no_org_black_dup_q.bam \
alignment/CRR134567/CRR134567_sort_no_org_black_dup_q.bam \
alignment/CRR134582/CRR134582_sort_no_org_black_dup_q.bam \
alignment/CRR134583/CRR134583_sort_no_org_black_dup_q.bam \
 --labels CRR134566 CRR134567 CRR134582 CRR134583 \
 --outFileName quality/correlation_plot/multiBamArray_replicates.npz --binSize 5000 -p 4
```

From this data, the correlation between samples can be calculated which will be used to do the clustering. The `plotCorrelation` function does this job and generates a graph with the results:

```
plotCorrelation --corData quality/correlation_plot/multiBamArray_replicates.npz \
 --plotFile quality/correlation_plot/replicates_correlation_bin.pdf --outFileCorMatrix
NKcellsATAC_chr14_correlation_bin.txt \
 --whatToPlot heatmap --corMethod spearman
```



The biological replicates clustered as we expected, suggesting good replicability of the experiment.

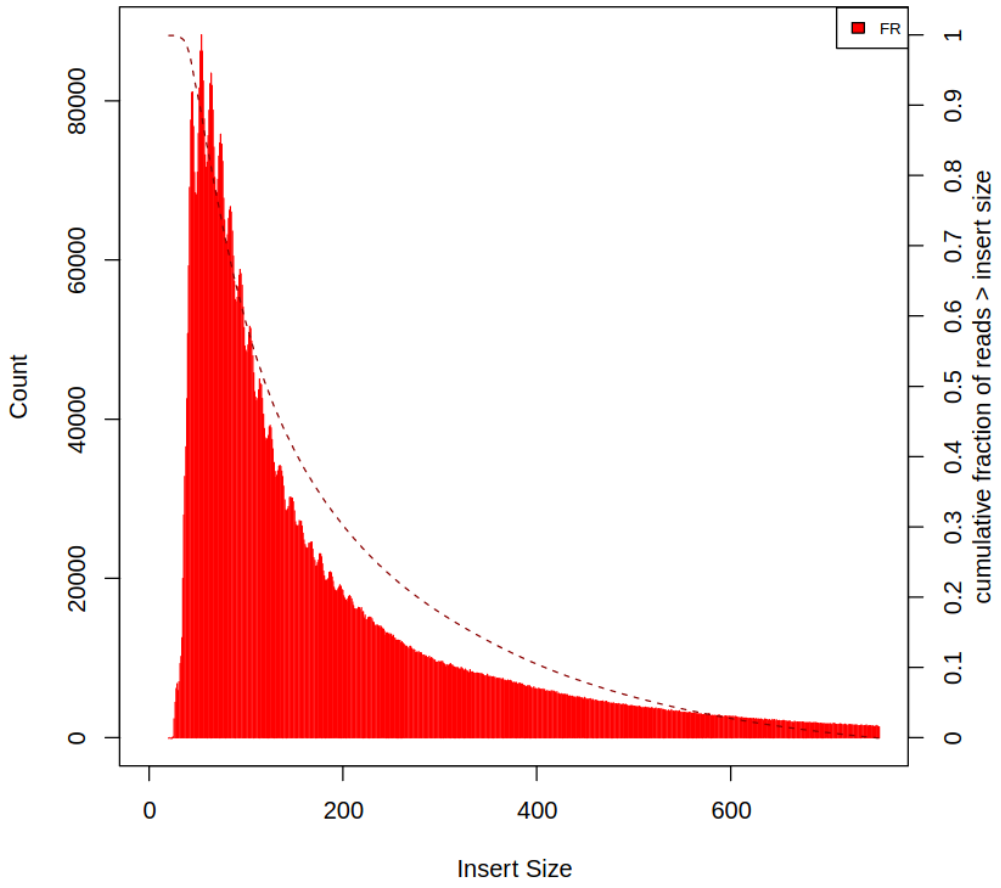
In the next two sections we will continue working with sample CRR134566 as we did at the beginning.

Distribution of the length of DNA fragments

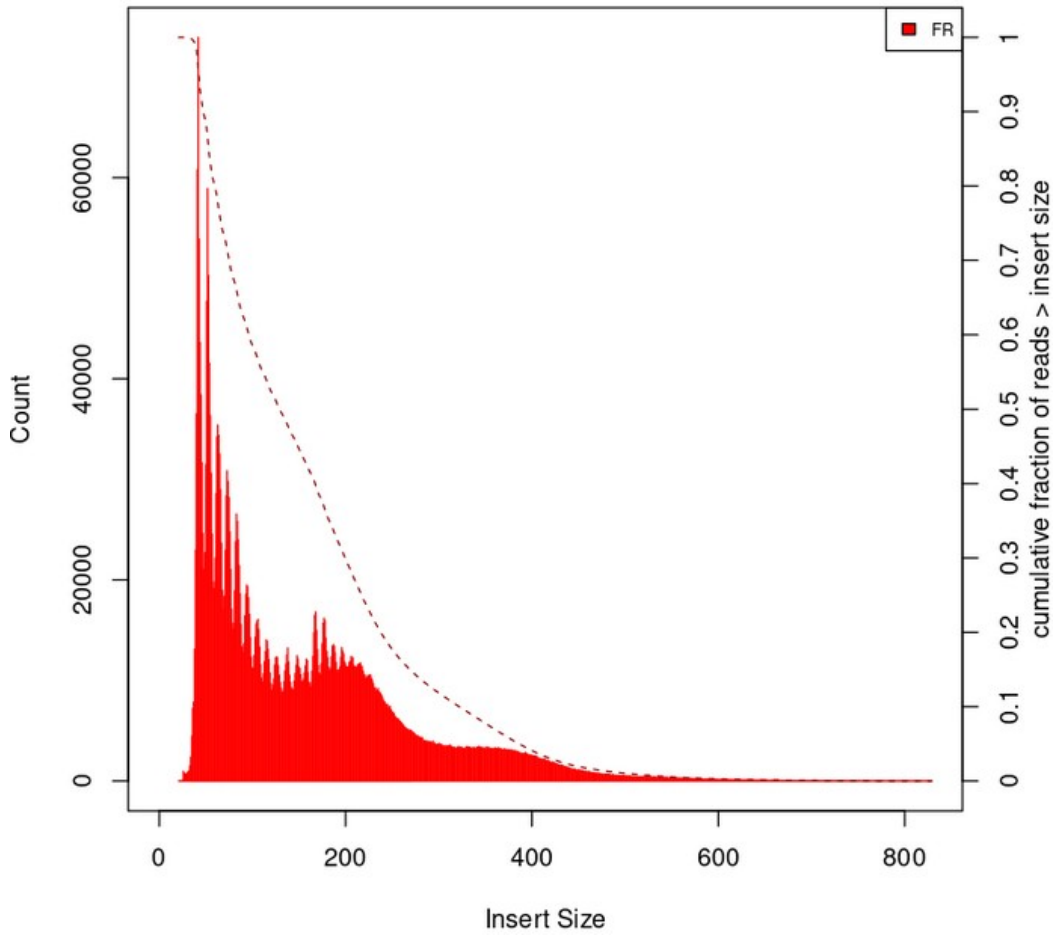
The length distribution of DNA fragments should be multimodal and periodic, because we expect fragments originating from NFRs, as well as fragments containing one, two, three or more nucleosomes. We'll start by generating a histogram for the fragment length distribution:

```
java -Xmx16G -jar ${HOME}/picard/picard.jar CollectInsertSizeMetrics \
-I alignment/CRR134566/CRR134566_sort_no_org_black_dup_q.bam \
-O quality/fragment_size_distribution/CRR134566_frag_length.stats \
-H quality/fragment_size_distribution/CRR134566_frag_length.pdf -M 0.5
```

The `-Xmx16G` option indicates the maximum memory that the program can use. The generated histogram is this:



This histogram is far from ideal for this metric. We should see a multimodal distribution with a clear periodic pattern with the highest peak at less than 100 bp (corresponding to nucleosome-free regions) and other increasingly lower peaks at approximately 200, 400, and 600 bp (corresponding to regions with one two and three nucleosomes respectively):



The image above has been taken from Epigenomics workshop 2023.

The pattern we observe is associated with ATAC-seq datasets with a lot of background noise, so we expect a small signal-to-noise ratio. However, other articles (Changes in chromatin accessibility between Arabidopsis stem cells and mesophyll cells illuminate cell type-specific transcription factor networks; Identification of Open Chromatin Regions in Plant Genomes Using ATAC-Seq) where ATAC-seq has been done in *Arabidopsis thaliana* have reported this distribution and have gone ahead with the study.

Signal to the TSS

To study the signal in the TSS we will use the package for R *ATACseqQC*. So, we'll start an R session and load all the libraries we'll need:

```
library("ATACseqQC")
library("BSgenome.Athaliana.TAIR.TAIR9")
library("TxDb.Athaliana.BioMart.plantsmart28")
library("ChIPpeakAnno")
library("Rsamtools")
```

The *Bsgenome.Athaliana.TAIR.TAIR9* library contains all *Arabidopsis thaliana* genome sequences. Even though it's TAIR9 and the version we've aligned with is TAIR10, we don't need to worry because TAIR10 only updates the annotations, not the sequences. The *ChIPpeakAnno* package contains specific functions to study the distribution of the data around the annotations of interest and *Rsamtools* will be used to manipulate the *bam* files.

Now we will define and save in different variables the working directory, the sample identifier and the path to the *bam* file and the directory where we will save the results:

```
wd <- "/media/tomas/DATA/biodatascience/atac_seq/"
bamFileLabels <- "CRR134566"
bamFile <- paste0(wd, "alignment/", bamFileLabels, "/", bamFileLabels, "_sort_no_org_black_dup_q.bam")
outPath <- paste0(wd, "quality/ATACseqQC/", bamFileLabels)
```

Next we need to perform a series of steps before generating any results. The reason for this is that before generating the visualizations we have to move the coordinates of the alignments and separate the reads into fragments corresponding to NFR and mono- di- or trinucleosomes.

The movement of the coordinates must be done because the Tn5 transposase introduces a 9 bp sequence at the 5' end of the fragments. To correct this and correctly identify the position where the Tn5 transposase has joined the DNA, we will move the positive strand (5' → 3') 4 bp in the 5' → 3' direction and the negative strand (3' → 5') we move it 5 bp in the direction 3' → 5'. In any case, this correction is only important for analysis that requires nucleotide resolution, such as transcription factor footprinting and motif analysis.

Let's start the preprocessing. First, we look for which tags (information about the alignment of each read) the *bam* file contains. We generate all possible tags:

```
possibleTag <- combn(LETTERS, 2)
possibleTag <- c(paste0(possibleTag[1, ], possibleTag[2, ]),
                 paste0(possibleTag[2, ], possibleTag[1, ]))
```

And then we retrieve those present in our *bam* file:

```
bamTop100 <- scanBam(
  file = BamFile(bamFile, yieldSize = 100),
  param = ScanBamParam(tag = possibleTag)
)[[1]]$tag
tags <- names(bamTop100)[lengths(bamTop100) > 0]
```

We will create a `GRanges` object of the *Arabidopsis thaliana* autosomes and another object with the alignments from our *bam* file. Object `GRanges` of the chromosomes:

```
seqlev <- c("Chr1", "Chr2", "Chr3", "Chr4", "Chr5")
which <- as(seqinfo(Athaliana)[seqlev], "GRanges")
```

The `Athaliana` object contains the *Arabidopsis thaliana* genome and comes from the previously loaded library *Bsgenome.Athaliana.TAIR.TAIR9*. If we evaluate the object `which` we will see that it is a `GRanges` with the coordinates of the chromosomes:

```
> which
GRanges object with 5 ranges and 0 metadata columns:
       seqnames     ranges strand
          <Rle>  <IRanges>  <Rle>
```

```
  Chr1     Chr1 1-30427671      *
  Chr2     Chr2 1-19698289      *
  Chr3     Chr3 1-23459830      *
  Chr4     Chr4 1-18585056      *
  Chr5     Chr5 1-26975502      *
  -------
  seqinfo: 5 sequences from TAIR9 genome
```

So that the chromosome name has the same format as the bam file, we will change the names with the `mapSeqlevels()` and `renameSeqlevels()` functions:

```
newStyle <- mapSeqlevels(seqlevels(which), "NCBI")
which <- renameSeqlevels(which, newStyle)
```

And now we read the *bam* file as a `GAlignmentsList` object:

```
gal <- readBamFile(
  bamFile = bamFile, tag = tags, which = which, asMates = TRUE, bigFile = TRUE
)
```

We can now move the coordinates. We'll either do this with the `shiftGAlignmentsList()` function:

```
shiftedBamFile <- paste0(wd, "alignment/",  bamFileLabels, "/", bamFileLabels, "_shifted.bam")
gal_shifted_gr <- shiftGAlignmentsList(gal, outbam = shiftedBamFile)
```

The function generates a new *bam* file with the corrected coordinates that we can use in the future for analyzes that require them.

The next part of the data preparation to study the signal at the TSS is to generate a `GRanges` object with the TSSs of the annotated transcripts. To do this, we will use the `transcripts()` function on the `TxDb.Athaliana.BioMart.plantsmart28` object from the previous package with the *Arabidopsis thaliana* annotations and select only the first coordinate of each of them:

```
txs <- transcripts(TxDb.Athaliana.BioMart.plantsmart28)
genome <- Athaliana
TSS <- promoters(txs, upstream = 0, downstream = 1)
TSS <- unique(TSS)
```

We will export the TSS coordinates in bed format in case we want to view them in IGV later:

```
export(TSS, paste0(outPath, "/tss.bed"))
```

We assume that shorter fragments correspond to NFRs, while increasingly longer fragments should reflect those containing one or more nucleosomes. The function `splitGAlignmentsByCut()` does this classification of the reads automatically and returns a list with the classified alignments:

```
objs <- splitGAlignmentsByCut(
  gal_shifted_gr, txs = txs, genome = genome, outPath = outPath
)
```

In the directory specified in `outPath` we will find the *bam* files for the NFR, mono-, di- and trinucleosome fragments. Now, let's estimate the size of each of the libraries (the number of fragments that each *bam* file contains). We will do this by counting the number of fragments in each of the fractions. Alternatively, we could do this with the `estLibSize()` function of the *ChIPpeakAnno* package. We need this data to perform a correct normalization when we calculate the enrichment around TSSs for each group of fragments:

```
librarySize <- c(
  length(objs$NucleosomeFree)/2,
  length(objs$mononucleosome)/2,
  length(objs$dinucleosome)/2,
  length(objs$trinucleosome)/2
)
names(librarySize) <- c("NucleosomeFree", "mononucleosome", "dinucleosome", "trinucleosome")
> librarySize
NucleosomeFree mononucleosome   dinucleosome  trinucleosome
       4857687        1041615         977402         150898
```

Now we have it ready to start analyzing the enrichment around the TSS. In a successful ATAC-seq experiment we expect the majority of NFR fragments to be at (or very close to) the TSS while mononucleosomes should be depleted at these positions but enriched in regions flanking these positions.

The function `enrichedFragments()` calculates the signal of the data around the TSS in each group of fragments. First we define the number of base pairs around the TSS and the amount of genomic intervals (`NTILE`) to generate:

```
ups <- 1010
dws <- 1010
NTILE <- 101
```

And now we can use the `enrichedFragments()` function. We will focus on the signal on chromosome 1, assuming that it will be homogeneous in each of them:

```
sigs <- enrichedFragments(
  gal = objs[c("NucleosomeFree", "mononucleosome", "dinucleosome", "trinucleosome")],
  TSS = TSS,
  librarySize = librarySize,
  seqlev = "1",
  TSS.filter = 0.5,
  n.tile = NTILE,
  upstream = ups,
  downstream = dws
)
```

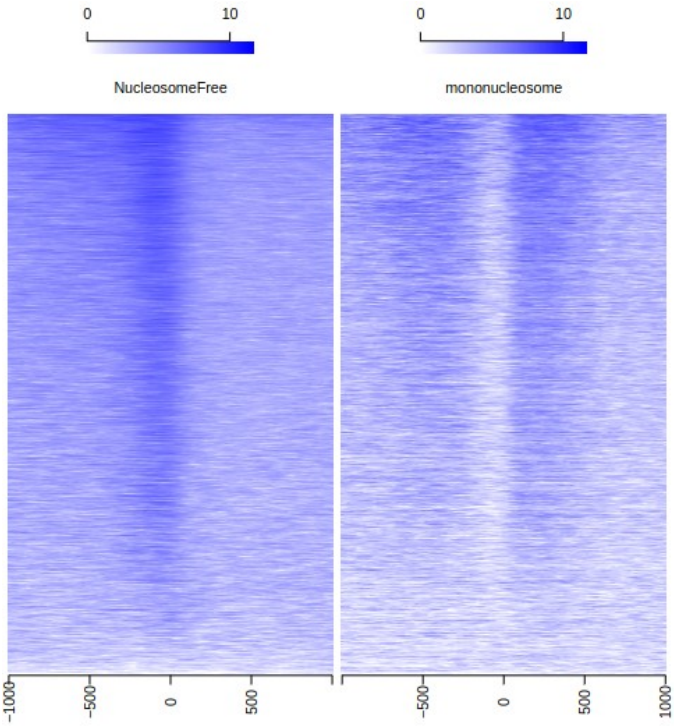
We transform the results by applying the logarithm to base 2:

```
sigs_logs2 <- lapply(sigs, function(.ele) log2(.ele + 1))
```

The first plot we will make will be a heatmap:

```
featureAlignedHeatmap(
  cvglists = sigs_logs2,
  feature.gr = reCenterPeaks(TSS, width=ups+dws),
  zeroAt = 0.5,
  n.tile=NTILE,
  color = colorRampPalette(c("white", "blue"))(50)
)
```

The argument `zeroAt = 0.5` specifies that the heatmap should be centered at the midpoint of the width of the genomic intervals entered in `feature.gr`.



Each row represents a gene and we see a 1000 bp window around the TSS (position 0). The more intense the color of the heatmap, the more overlap the fragments of that group have with that position. The heatmap suggests that the quality of the signal is as expected from an ATAC-seq experiment: NFR fragments mostly come from positions very close to the TSS, while mononucleosomal fragments come from flanking positions but not from these positions.

Finally, we will study the distributions of each fragment fraction around the TSS using a density plot. We will use the `featureAlignedDistribution()` function to calculate the distribution:

```
out <- featureAlignedDistribution(
  cvglists = sigs,
  feature.g = reCenterPeaks(TSS, width=ups+dws),
  zeroAt=0.5, n.tile=NTILE, type="l",
  ylab="Averaged coverage"
)
```

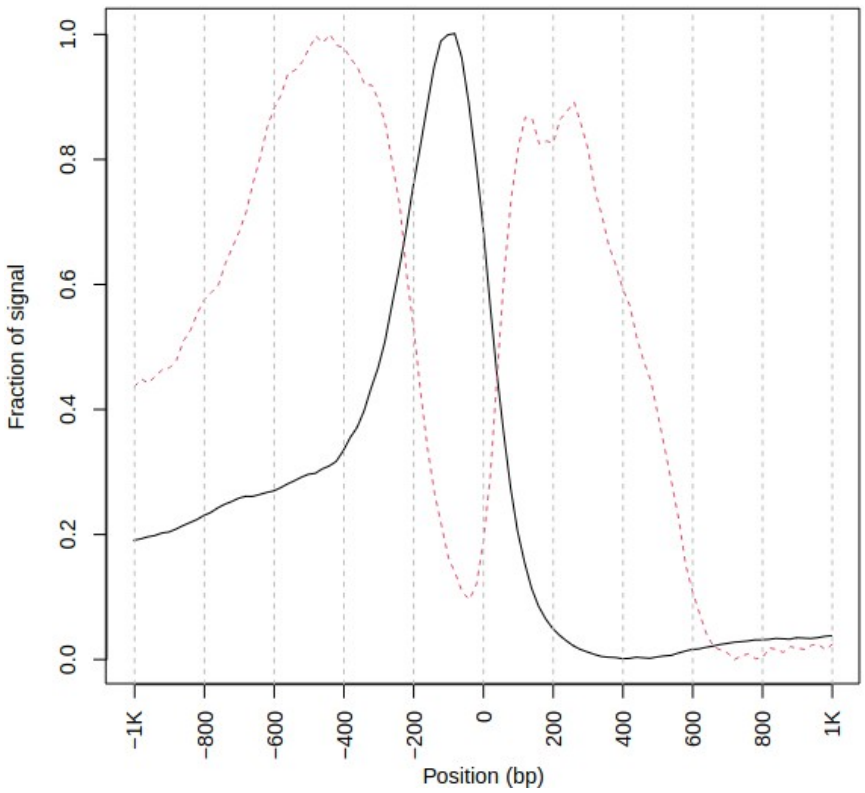
Now, we will perform a min/max normalization so that the values in each group are between 0 and 1:

```
range01 <- function(x) {
  (x-min(x))/(max(x)-min(x))
}
out <- apply(out, 2, range01)
> head(out)
     NucleosomeFree mononucleosome
[1,]      0.1889592      0.4387529
[2,]      0.1908510      0.4511787
[3,]      0.1941986      0.4452928
[4,]      0.1963801      0.4578554
[5,]      0.2005558      0.4672043
[6,]      0.2022263      0.4683198
```

We visualize the distributions by creating a line plot:

```
matplot(out, type="l", xaxt="n",
```

```
        xlab="Position (bp)",
        ylab="Fraction of signal")
axis(1, at=seq(0, 100, by=10)+1,
     labels=c("-1K", seq(-800, 800, by=200), "1K"), las=2)
abline(v=seq(0, 100, by=10)+1, lty=2, col="gray")
```



The results are consistent with the previous heatmap. The NFR fraction (solid line) is clearly enriched in fragments originating from positions belonging to or very close to the TSS. In contrast, the mononucleosomal fraction (dashed line) contains very few fragments at these positions, but is enriched at relatively close positions both upstream and downstream.

Peak calling

Once we have a clear idea of the quality of our data, we can start the most basic analysis of ATAC-seq data: peak calling. Peak calling is a computational method to identify open regions of chromatin peaks from aligned reads. We will use *macs2* compares the observed distribution of reads in the genome with a random background distribution:

```
macs2 callpeak --broad -t alignment/CRR134566/CRR134566_sort_no_org_black_dup_q.bam \
-n CRR134566_macs2 -f BAMPE \
-g 119481543 -q 0.05 --nomodel --keep-dup all \
--outdir results/peaks/
```

The `--broad` option is used to call wider peaks. We specified the `BAMPE` format, which deduces the length of the fragments from the information of each pair of reads. The `--nomodel` and `--keep-dup` options are associated with using `BAMPE` and having removed duplicates in a previous step, respectively.

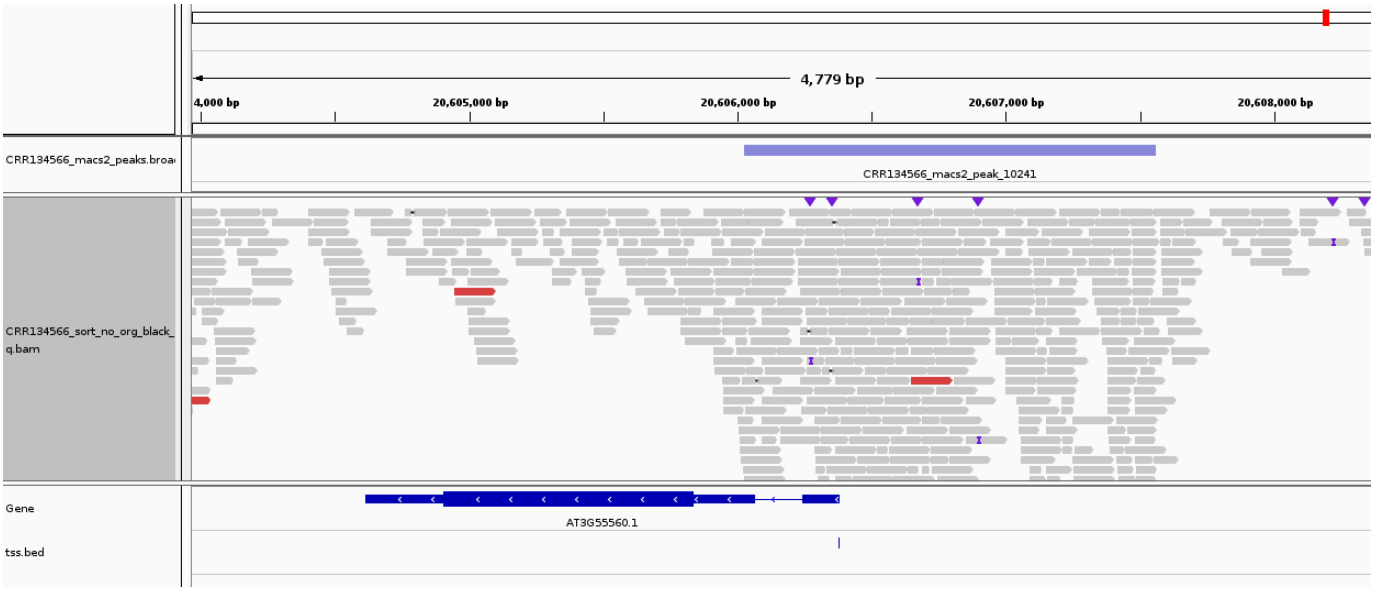
The program generates different result files. The *.broadPeaks* file is a bed file with the coordinates of all peaks found. The *.xls* file is a tabular file that also contains all the identified peaks as well as more information about them.

We can easily check the number of identified peaks:

```
wc -l CRR134566_macs2_peaks.broadPeak
17485 CRR134566_macs2_peaks.broadPeak
```

Data visualization and results in IGV

At this point, it is convenient to take a look at the reads aligned to IGV. We will view both the final *bam* file and the peaks, and the *bed* file with the TSSs:



The image shows the region Chr3:20,603,965-20,608,775. A peak overlapping with the TSS of the AT3G55560.1 gene is seen.

Conclusions

In this post we have seen how to manipulate ATAC-seq data from *fastq* files to peak calling. We performed the quality control of the reads, removed adapters and positions with poor quality base-calling. We also performed the alignment and filtered all the aligned reads until we were left with those of the highest quality and aligned with the nuclear chromosomes. We have also seen how to perform a quality control of the signal-to-noise ratio of the ATAC-seq experiment. Once all this processing and control of the data was done, we found the open areas of the chromatin thanks to the peak calling carried out with *macs2*. In a later post we will see how we can work with the results obtained.

The scripts with all the commands to reproduce the analysis can be found on the blog's GitHub.