

ATAC-seq data analysis: read mapping, peak calling, and data visualization

This protocol is used to map ATAC-seq reads to the genome of origin, followed by peak calling with Homer to identify regions of the genome that are hypersensitive to transposase integration, and visualization of the data using deepTools. We also include a section on identification of transcription factor footprints using pyDNase. The protocol assumes that the following software packages are installed and are on your PATH:

- Bowtie2 (<http://bowtie-bio.sourceforge.net/bowtie2/index.shtml>)
- Samtools (<http://www.htslib.org/doc/samtools.html>)
- Homer (<http://homer.salk.edu/homer/>)
- deepTools (<https://deeptools.readthedocs.io/en/latest/>)
- pyDNase (<http://pythonhosted.org/pyDNase/>)

Command lines for each program are given in *italics* at each step. Homer, deepTools, and pyDNase have a variety of adjustable parameters that you should be aware of -- complete documentation for each can be found at the websites listed above. The commands given here have been optimized specifically for use with Arabidopsis ATAC-seq data and may need further tuning for other applications.

I. Mapping and processing of reads

Here we map reads to the genome of origin using Bowtie2 with default parameters. We then sort the file and filter it to retain only reads with a mapping quality score of at least 2. This filtered bam file is used for peak calling in the next step.

1. Map ATAC-seq reads to the desired genome using Bowtie2 with default parameters to generate a sam file:

```
bowtie2 -x genome.index.prefix -1 read1.fastq.gz -2 read2.fastq.gz -S file.sam
```

2. Convert sam file to bam format:

```
samtools view -bS -o file.bam file.sam
```

3. Sort bam file:

```
samtools sort file.bam -o file.sorted.bam
```

4. Index bam file:

```
samtools index file.sorted.bam
```

5. Filter bam file to retain only reads with a mapping quality (q score) of at least 2 using Samtools:

```
samtools view -q 2 -b file.sorted.bam > file.sorted.q2.bam
```

II. Calling peaks with Homer

Homer's peak calling tool (*findpeaks*) has many adjustable parameters that can be changed depending on the nature of the peaks that are expected. Here, we use the tool in "region" mode, which works well for ATAC-seq. The .bam file from the previous step must first be used to make a "tag directory" that Homer will use for calling peaks. The peak file generated by Homer is in a non-standard format, which we convert to a merged, sorted bed file in the final step of this section.

1. Make a Homer tag directory from the bam file of mapped ATAC-seq reads:

```
makeTagDirectory <tag.directory.folder.name> file.sorted.q2.bam
```

2. Call peaks with Homer in "region" mode:

```
findpeaks tag.directory.folder.name/ -o name.of.peak.file -gsize 1.2e8  
minDist 150 -region
```

→ **Note:** -gsize parameter is the effective genome size (Arabidopsis = 1.2e8, tomato = 8.2e8, rice = 3.7e8, Medicago = 4e8)

3. Convert Homer peak file to a merged, sorted bed file:

```
pos2bed.pl name.of.peak.file | bedtools sort | bedtools merge >  
name.of.peak.file.merged.bed
```

III. Visualization of ATAC-seq data using deepTools

Now that your reads are mapped and peaks are called, you may want to visualize your data in a genome browser or use a heatmap and/or average plot to visualize the ATAC-seq signals at your peak regions. You can accomplish both of these tasks using deepTools. For easy visualization in a genome browser and for making heatmaps you will need to first convert your processed .bam file into bigwig (.bw) format. This file is greatly reduced in size relative to the .bam and is therefore more amenable to genome scanning in a browser. The bigwig file, along with your .bed file of peak regions, will also be used to generate a data matrix of read counts over the peak regions. This data matrix can then be represented as a heatmap and an average plot.

1. Convert .bam file to a normalized bigwig using the *bamCoverage* tool in deepTools:

```
bamCoverage -b file.sorted.q2.bam -o file.sorted.q2.bw -bs=1  
--normalizeUsingRPKM -p=max
```

→ **Note:** In this case, we are making the bigwig file (*file.sorted.q2.bw*) using a bin size of 1 bp (-bs=1 parameter), normalizing using RPKM, and using all available processor power (-p=max parameter).

2. Generate a data matrix of read counts over your peak regions using the deepTools *computeMatrix* function:

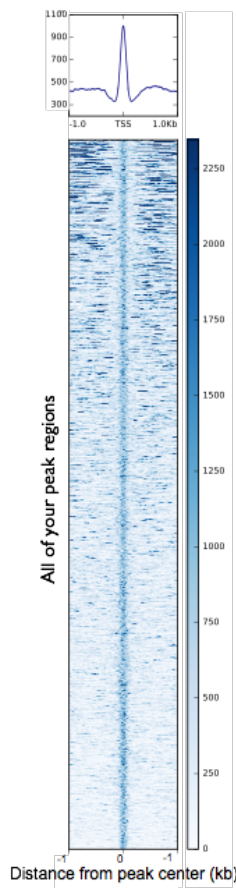
```
computematrix reference-point -S file.sorted.q2.bw -R  
name.of.peak.file.merged.bed --referencePoint center -b 1000 -a 1000  
-bs=1 -p=max -out new.data.matrix.gz
```

→ **Note:** Here we've created a data matrix of read counts centered on the middle of each peak region (using the parameter: `--referencePoint center`) and extended the regions of analysis to include 1000 bp upstream of each peak start (`-b 1000` parameter) and 1000 bp downstream of each peak end (`-a 1000` parameter). We have again used a bin size of 1 bp and maximum processor capacity. Our output file here is: `new.data.matrix.gz`.

3. Generate a heatmap and average plot from the data matrix using the `plotHeatmap` function in deepTools:

```
plotHeatmap -m new.data.matrix.gz -out new.data.plot.pdf  
--colorMap=Blues
```

→ **Note:** This command will generate a .pdf file containing a combined image of the heatmap and the average plot, as seen below. The `--colorMap=Blues` parameter gives the heatmap a color scale ranging from white to blue:



→ **Note:** All of the deepTools functions described above have a large number of variable parameters and can be used in very sophisticated ways. The deepTools website is a great resource for learning about all that you can do with these tools.

IV. Footprinting with pyDNase

If you have sequenced your sample to a sufficient depth (~100 million mapped reads for Arabidopsis) then you can use pyDNase to look for transcription factor footprints inside your peak regions. pyDNase will use your Homer peak bed file and your bam file of ATAC-seq reads as input, and will return a bed file indicating the coordinates of any footprints it identifies. The pyDNase package also has a variety of other scripts for analyzing footprints (see website). Since the software was originally made for DNaseI-seq, you must be sure to use ATAC-seq mode for all of these scripts (that’s the “-A” parameter in the commands below).

→ **Note:** For footprinting it is **very important** to include a “naked DNA” control ATAC-seq in order to account for the insertion bias of Tn5 transposase, which appears to be extensive. Many “footprints” are also seen in naked DNA ATAC-seq, so users should be fully aware of this caveat and proceed accordingly.

1. Search for footprints within peak regions using your processed ATAC-seq bam file:

```
wellington_footprints.py -A name.of.peak.file.merged.bed
file.sorted.q2.bam name.of.output.directory
```

→ **Note:** The output directory must be made before the command is issued. After running, the output directory will contain a bed file with the coordinates and scores for all footprints identified in your peak regions (a file called “**..WellingtonFootprints.default.FDR.0.01.bed**”). A separate folder within will also contain the footprints divided up by p value cutoffs.

→ **Note:** If your peak file contains peak regions shorter than 100 bp, which is often the case with deeply sequenced samples, pyDNase will return the error message, “*the interval you’re trying to footprint is smaller than the parameters you’ve passed*”. We fix this problem by opening the bed file in excel, calculate peak length, and then adding the required number of bases to short peaks to bring them up to 100 bp. If you do this, be sure to save the .bed file as “Windows formatted text”, or pyDNase will have problems with it.

2. Examine the average ATAC-seq cut profile over your footprints:

```
dnase_average_profile.py -A
your.WellingtonFootprints.default.FDR.0.01.bed file.sorted.q2.bam
name.for.output.pdf
```

→ **Note:** This command will generate a .pdf containing a plot of the average TN5 insertion events per base over the footprint regions discovered in step 1. An example is shown below:

