### deepTools

The production of figures is an important step in any analysis workflow; well produced, descriptive figures can provide considerable evidence to support a finding or hypothesis, particularly when presented in a research paper (Springer 2019). The software suite deepTools is a Python built set of tools concerned with analysing high-throughput sequencing data, such as ATACseq and ChIPseq data. Importantly, deepTools contains a number of tools which allow for the production of plots designed to visualise datasets, such as heatmaps, plot profiles and enrichment plots. First created by researchers at the Max Planck Institute of Immunobiology and Epigenetics in 2014, deepTools2 was released as an update to the first package in 2016 – including new tools and enhancements for visualising and interpreting genetic data (Ramirez et al. 2016). Before they are able to be plotted as a figure, datasets must first go through some procedures to produce subsidiary analysis outputs, such as bam coverage files and compute matrices. deepTools version 2.5.4 was loaded onto the Hawk supercomputer, first by importing the Raven module list followed by loading the “deeptools” module. Below are the procedures that were carried out in the suite:

#### Step 0 – Creating a Padded Bed File

Before the deepTools analysis can take place, a bed file must be created containing a list of genes, the position of their Transcription Start Site (TSS) and the chromosome scaffold number must be created. This file is then padded (using `bedtools slop`) by adding and subtracting 1000bp from the TSS to create a region around the TSS to be visualised. Below are the steps taken to create this file.

##### Step 1 - Download from Biomart (Ensembl Genes 97 - Mouse Genes (GRCh38.p6))

1. Using the list of genes (in the select order, e.g. in order of p-value or Fold Change), select filters.

2. Input external references ID (Gene stable ID(s)) and upload the list of genes.

3. In attributes, select (IN THIS ORDER): Chromosome/scaffold name, Transcription Start Site (TSS) and Gene stable ID version.

4. Selecting results, download the file (unique results only) selecting file TSV.

5. Rename the downloaded file something memorable (for example genesFoldChange.txt).

##### Step 2 - Using awk to alter the file

1. Upload the file to the supercomputer.

2. Use the command `awk 'BEGIN{FS="\t"} {print $1"\t"$2"\t"$2"\t"$3.$2}' <GENES>.txt > <GENES>.bed` to produce the .bed file.

3. Check this altering has worked using `head <GENES>.bed` - You should see a file containing (IN THIS ORDER): Chromosome number, TWO columns for Transcription Start Site (TSS) and a unique column (a concatenation of the Gene stable ID and the TSS).

4. Remove the column headers (using vi + dd).

##### Step 3 - Creating the Gene file using samtools

1. On a pre-existing .bam file (any will do), run `samtools view <BAM FILE>.bam -H > <GENOME FILE NAME>.txt`.

2. Edit the newly created gene file by removing the first column, the first row and removing the "LN:" and "SN:" from the 2nd and 3rd column. Also, move the 2nd and 3rd column to become the 1st and 2nd.

##### Step 4 - Using bedtools slop to produce the padded .bed file

1. Run `bedtools slop -i <GENES>.bed -g <GENOME FILE NAME>.txt -b 1000 > <GENES PAD>.pad1000.bed`.

#### Step 1 – Producing a Bam Coverage file

bamCoverage --bam <PATH-TO-BAM>/wt1\_1\_trimmed.rmdup.bam \

--outFileName wt1coverage.bigWig

In this step, a coverage file (the same as those viewable in IGV Genome Browser) of the .bam alignment files produced by the NextFlow pipeline. The output is saved in `.bigWig` format, as it is an indexed binary file.

#### Step 2 – Producing a Compute Matrix file

computeMatrix reference-point –regionsFileName LogFold-Genes.pad1000.bed \--scoreFileName wt1coverage.bigWig \

--outFileName wt1LogMatrix.tab.gz --referencePoint center \

--beforeRegionStartLength 1000 --afterRegionStartLength 1000 \

--sortRegions keep

A compute matrix must then be produced containing scores per genome regions, using the previously created `.bigWig` file (`--scoreFileName`) and `.bed` file (`--regionsFileName`). The `computeMatrix reference-point` command is used, as the TSS is of focus (the reference point); the regions before (upstream, `--beforeRegionStartLength 1000`) and after (downstream, `--afterRegionStartLength 1000`) are also plotted. The compute matrix is saved in gzipped .tab format (`--outFIleName`), the reference point is centred in the plot (`--referencePoint center`), the original sorting is kept (as the .bed files are sorted by p-value and log fold change separately - `--sortRegions keep`).

set TRAPPATH=”<path-to-TRAP>”

set UCSCPATH=”<path-to-ucsctools>”

../TRAP/utilities/trap.sh -s GRCm38\_sm.fa -m TATA.psem -g GRCm38\_sm.fa.fai

computeMatrix reference-point --regionsFileName MouseGenes.bed \

--scoreFileName GRCm38\_sm\_M00471.bw --outFileName TATA-sites.tab.gz \

--referencePoint TSS -a 100 -b 100 --binSize 5

For creation of the TATA box enrichment site compute matrix, the `.bigWig` file was instead created by the Transcription Affinity Prediction (TrAP) package, installed on the Hawk supercomputer (version 3.05) via GitHub clone (`git clone https://github.com/maxplanck-ie/TRAP`). Also installed was the UCSC tools (version 1.04.00) package – a dependency of the TrAP package - in the same way (`git clone https://github.com/kevlim83/ucsctools`). The environment variables were then set as the path to each of the package folders in order for the `trap.sh` script to run correctly. The TrAP package was then run by executing the `../TRAP/utilities/trap.sh` shell script, using the mouse reference genome (`-s GRCm38\_sm.fa`), a piecewise Structural Equation Modeling matrix (PSEM, `-m TATA.psem`) - supplied by Dr Thomas Manke of the Max Planck Institute of Immunobiology and Epigenetics - and a indexed version of the mouse reference genome (`-g GRCm38\_sm.fa.fai`) which was created using `samtools faidx`.

The bigwig file created is then inputted (`--scoreFileName GRCm38\_sm\_M00471.bw`) into the compute matrix command, using a .bed file of all mouse genes (`--regionsFileName MouseGenes.bed`) and the centre point set as the TSS, with 100bp upstream and downstream also plotted with a bin size of 5 (`--referencePoint TSS -a 100 -b 100 --binSize 5`).

#### Step 3 – Plotting the Compute Matrix file as a Heatmap

plotHeatmap --matrixFile wt1LogMatrix.tab.gz \

--outFileName wt1LogHeatMap.png --sortRegions no

The final step is to plot the newly created compute matrix as a heatmap. The only customised flag in this scenario is the regions are not sorted, so that the order of p-value or Log Fold Change can be kept (`--sortRegions no`).

plotHeatmap --matrixFile TATA-sites.tab.gz \

--outFileName TATAsites.png \

--colorMap hot\_r --missingDataColor .4 --heatmapHeight 7 \

--plotTitle 'TATA motif' --whatToShow 'heatmap and colorbar' \

--sortRegions ascend

This is also done for the TATA box matrix, however a number of aesthetics are changed in order to visualise the TATA sites easier. The colours are set to hot (`--colorMap hot\_r`), missing data is coloured a dark grey (`--missingDataColor .4`), the height of the heatmap set to 7cm (`--heatmapHeight 7`), the title set to ‘TATA motif’ (`--plotTitle ‘TATA motif’`), regions sorted by ascending psem scores (`--sortRegions ascend`) and just the heat map and colorbar shown (`--whatToShow ‘heatmap and colorbar’`).