

# A Model to Predict Drivers of Adipogenesis

***Arun B. Dutta<sup>1</sup>, Bao H. Nguyen<sup>1</sup>, and Michael J. Guertin<sup>1</sup>***

<sup>1</sup>University of Virginia, Charlottesville, Virginia

**1 February 2021**

## Contents

1	Background . . . . .	4
2	Retrieving ATAC-seq fastq . . . . .	4
3	Processing ATAC-seq reads and aligning with Bowtie2 . . . . .	4
3.1	Installing fastq_pair . . . . .	4
3.2	Generate slurm file for each replicate and run in parallel . . . . .	4
3.3	Simplified version . . . . .	6
4	Performing seqOutBias . . . . .	7
4.1	Generating slurm file for each replicate . . . . .	7
4.2	Running seqOutBias . . . . .	8
4.3	Simplified version . . . . .	8
5	Calling ATAC peaks . . . . .	9
6	ATAC Preclustering . . . . .	10
6.1	Assign counts to peaks . . . . .	10
6.2	Plot PCA . . . . .	11
6.3	Dynamic peaks . . . . .	11
7	ATAC Clustering . . . . .	13
8	ATAC Postclustering . . . . .	14
8.1	Generate plot.df object . . . . .	14
8.2	Plot all clusters . . . . .	14
8.3	Plot dendrogram . . . . .	16
8.4	Plot clusters organized by supercluster . . . . .	17
8.5	Plot supercluster traces . . . . .	19
8.6	Plot individual traces . . . . .	20
8.7	Generating .bed file for each cluster . . . . .	22

9	FIMO motif enrichment . . . . .	23
9.1	Download required script . . . . .	23
9.2	Find motifs enriched in cluster . . . . .	23
10	MEME motif enrichment . . . . .	25
10.1	Generate slurm file for each replicate and run in parallel . . . . .	25
10.2	Simplified Version . . . . .	26
11	MEME-FIMO analysis . . . . .	27
11.1	Prepare MEME databases . . . . .	27
11.2	Query MEME motifs in TOMTOM . . . . .	28
11.3	Organize MEME superclusters . . . . .	29
11.4	Find matches between MEME and FIMO . . . . .	30
12	Defining motif families . . . . .	32
12.1	Query factors . . . . .	32
12.2	Visualize PSWM . . . . .	33
13	Generating composite motifs . . . . .	35
13.1	Download required scripts . . . . .	35
13.2	Query TOMTOM to calculate index and values for PSWM . . . . .	35
13.3	Generate composite motif and sequence logo . . . . .	36
14	Running FIMO on composite and splitting SP/KLF . . . . .	38
14.1	Download required scripts . . . . .	38
14.2	Generate slurm file for each replicate and run in parallel . . . . .	38
14.3	Preparing bed and bigWig files . . . . .	39
14.4	Split and generate individual composite motifs for SP and KLF . . . . .	41
14.5	Processing FIMO for SP and KLF . . . . .	44
15	Figure 1 . . . . .	45
15.1	Generate ‘fimo.scores.atac.Rdata’ object . . . . .	45
15.2	Plot all family peaks . . . . .	46
15.3	Plot all families bar plot, including no family category . . . . .	48
15.4	Plot traces of isolated peaks . . . . .	49
15.5	Plot bar plot of isolated peaks . . . . .	51
15.6	Plot family heatmaps . . . . .	52
15.7	Plot up/down split occurrences . . . . .	53
15.8	Plot FIMO scores box and whisker plots for isolated peaks . . . . .	55
15.9	Plot FIMO scores box and whisker plots for all peaks . . . . .	56
15.10	Plot split on FIMO scores for isolated peaks . . . . .	58

# A Model to Predict Drivers of Adipogenesis

15.11	Plot split on FIMO scores for all peaks . . . . .	59
15.12	Generate composites for all activated and repressed peaks . . . . .	61
15.13	Plot motif enrichment around summits . . . . .	63
16	Supplemental Figure 1 . . . . .	66
16.1	. . . . .	66

## 1 Background

---

The following pipeline is designed to compile on the UVA Rivanna Cluster. While most tools and packages are available through Rivanna, you will need to install some manually. To transfer files into and out of Rivanna, use [sftp](#). Remember to modify the directory path for all relevant scripts to match your /scratch/user path.

## 2 Retrieving ATAC-seq fastq

---

## 3 Processing ATAC-seq reads and aligning with Bowtie2

---

### 3.1 Installing fastq\_pair

[fastq\\_pair](#) rewrite paired-end fastq files to make sure all reads have a mate, which is required for Bowtie2 to align properly. Remember to move the fastq\_pair binary into the \$PATH. All files within the /scratch/user directory has a 90 days expiration limit (since last modified) so it is recommended to install packages in a permanent directory like /home/user.

```
mkdir /scratch/bhn9by/ATAC
cd /scratch/bhn9by/ATAC

#Retrieve and build fastq_pair binary
wget https://github.com/linsalrob/fastq-pair/archive/master.zip
unzip master.zip
cd fastq-pair-master
gcc -std=gnu99 main.c robstr.c fastq_pair.c is_gzipped.c -o fastq_pair
cp fastq_pair /home/bhn9by/bin
cd..
```

### 3.2 Generate slurm file for each replicate and run in parallel

To run slurm jobs for each replicate in parallel, we concatenate three slurm header files and two lines corresponding to the name of the replicate. Processing and aligning each replicate will take several hours so this slurm\_header method will be faster. The content of each header is provided in the next subsection.

[github raw](#)

```
#header_1 --> sbatch -n and -t
#temp.txt --> sbatch -o (name of replicate)
#header_2 --> sbatch -p, -A, and modules to load
#temp2.txt --> define variable $i as name of replicate
#header_3 --> actual script

for i in *_atac_PE1.fastq.gz
do
    name=$(echo $i | awk -F"/" '{print $NF}' | \
        awk -F"_atac_PE1.fastq.gz" '{print $1}')
    echo $name
    echo '#SBATCH -o' $name'.align.out' > temp.txt
```

```
echo 'i='$i > temp2.txt
cat align_slurm_header_1.txt temp.txt \
    align_slurm_header_2.txt temp2.txt \
    align_slurm_header_3.txt > $name.align.slurm
sbatch $name.align.slurm
rm temp.txt
rm temp2.txt
done
```

### 3.2.1 align\_slurm\_header\_1.txt

[github raw](#)

```
#!/bin/bash
#SBATCH -n 1
#SBATCH -t 96:00:00
```

### 3.2.2 align\_slurm\_header\_2.txt

[github raw](#)

```
#SBATCH -p standard
#SBATCH -A guertinlab

module load bioconda/py3.6 gcc/7.1.0 bowtie2/2.2.9 samtools/1.10
source activate myenv
```

### 3.2.3 align\_slurm\_header\_3.txt

[github raw](#)

```
name=$(echo $i | awk -F"/" '{print $NF}' | \
    awk -F"_atac_PE1.fastq.gz" '{print $1}')
echo $name
gunzip $name*.gz

echo 'pairing .fastq files'
fastq_pair ${name}_atac_PE1.fastq ${name}_atac_PE2.fastq
rm ${name}single.fq
echo 'align to mouse genome'

bowtie2 --maxins 500 -x \
    /project/genomes/Mus_musculus/UCSC/mm10/Sequence/Bowtie2Index/genome \
    -1 ${name}_atac_PE1.fastq.paired.fq \
    -2 ${name}_atac_PE2.fastq.paired.fq -S ${name}_atac_smp.sam

echo 'quality filter and remove duplicate amplicons'
samtools view -b -q 10 ${name}_atac_smp.sam | samtools sort -n - | \
    samtools fixmate -m - - | samtools sort - | \
    samtools markdup -r - ${name}_atac_rmdup.bam
```

```
rm ${name}_atac_smp.sam
gzip ${name}*fastq
```

### 3.3 Simplified version

For clarity we provided a simplified, sequential version of the processing and alignment script.

[github raw](#)

```
#align .fastq to 10mm genome
for i in *_atac_PE1.fastq.gz
do
    name=$(echo $i | awk -F"/" '{print $NF}' | \
        awk -F"_atac_PE1.fastq.gz" '{print $1}')
    echo $name
    gunzip $name*.gz

    echo 'pairing .fastq files'
    fastq_pair ${name}_atac_PE1.fastq ${name}_atac_PE2.fastq
    rm ${name}*single.fq

    echo 'align to mouse genome'
    bowtie2 --maxins 500 -x \
        /project/genomes/Mus_musculus/UCSC/mm10/Sequence/Bowtie2Index/genome \
        -1 ${name}_atac_PE1.fastq.paired.fq \
        -2 ${name}_atac_PE2.fastq.paired.fq -S ${name}_atac_smp.sam

    echo 'quality filter and remove duplicate amplicons'
    samtools view -b -q 10 ${name}_atac_smp.sam | samtools sort -n - | \
        samtools fixmate -m - - | samtools sort - | \
        samtools markdup -r - ${name}_atac_rmdup.bam
    rm ${name}_atac_smp.sam
    gzip ${name}*fastq
done
```

## 4 Performing seqOutBias

`seqOutBias` corrects for enzymatic sequence bias by scaling the aligned read counts by the ratio of genome-wide observed read counts to the expected sequence based counts for each k-mer. The sequence based k-mer counts take into account mappability at a given read length using Genome Tools' Tallymer program. We will also use `seqOutBias` to generate bigwig from the bam files.

### 4.1 Generating slurm file for each replicate

The content of each header is provided in the next subsection.

[github raw](#)

```
for bam in *_rmdup.bam
do
    name=$(echo $bam | awk -F"_atac_rmdup.bam" '{print $1}')
    echo $name
    echo '#SBATCH -o '$name'.bigwig.out' > temp.txt
    echo 'bam='$bam > temp2.txt
    cat bigwig_slurm_header_1.txt temp.txt \
        bigwig_slurm_header_2.txt temp2.txt \
        bigwig_slurm_header_3.txt > $name.convert.to.bigwig.slurm
done
```

#### 4.1.1 bigwig\_slurm\_header\_1.txt

[github raw](#)

```
#!/bin/bash
#SBATCH -n 1
#SBATCH -t 96:00:00
```

#### 4.1.2 bigwig\_slurm\_header\_2.txt

[github raw](#)

```
#SBATCH -p standard
#SBATCH -A guertinlab

module load genometools/1.5.10 wigtobigwig/2.8 gcc/7.1.0 seqoutbias/1.2.0
```

#### 4.1.3 bigwig\_slurm\_header\_3.txt

[github raw](#)

```
cd /scratch/bhn9by/ATAC

name=$(echo $bam | awk -F"_atac_rmdup.bam" '{print $1}')
echo $name

seqOutBias \
```

```
/project/genomes/Mus_musculus/UCSC/mm10/Sequence/WholeGenomeFasta/genome.fa \
$bam --skip-bed --no-scale --bw=${name}.bigWig \
--only-paired --shift-counts --read-size=38
```

## 4.2 Running seqOutBias

Run one slurm file to completion to generate requisite tallymer mappability file.

[github raw](#)

```
sbatch 3T3_20min_rep1.convert.to.bigwig.slurm
```

After this is done, start all others (and repeat the first one).

```
for slurm in *bigwig*slurm
do
    sbatch $slurm
done
```

## 4.3 Simplified version

For clarity we provided a simplified, sequential version of the seqOutBias script.

[github raw](#)

```
#perform seqOutBias and convert .bam to .bigwig
cd /scratch/bhn9by/ATAC

for bam in *rmdup.bam
do
    name=$(echo $bam | awk -F"_atac_rmdup.bam" '{print $1}')
    echo $name

    seqOutBias \
    /project/genomes/Mus_musculus/UCSC/mm10/Sequence/WholeGenomeFasta/genome.fa \
        $bam --skip-bed --no-scale --bw=${name}.bigWig \
        --only-paired --shift-counts --read-size=38
done
```

## 5 Calling ATAC peaks

MACS2 is a program for detecting regions of genomic enrichment. The input is .bam files and the output is .bed files. We will also remove blacklisted regions of the mm10 genome using Bedtools.

[github raw](#)

```
#!/bin/bash
#SBATCH -n 1
#SBATCH -t 96:00:00
#SBATCH -o old.peak.calling.out
#SBATCH -p standard
#SBATCH -A guertinlab

module load bioconda/py3.6 gcc/7.1.0 bedtools/2.26.0 samtools/1.10 macs2/2.2.7.1
source activate myenv

#Call peaks
echo 'Calling Peaks'
macs2 callpeak -t *rmdup.bam -f BAMPE -n 3T3_atac \
--outdir old_peak_calling_macs --keep-dup 50 -q 0.05

#Download blacklisted mm10 regions
wget https://www.encodeproject.org/files/ENCFF547MET/@download/ENCFF547MET.bed.gz
gunzip ENCFF547MET.bed.gz
mv ENCFF547MET.bed mm10.blacklist.bed

#Remove blacklisted regions
cd old_peak_calling_macs
bedtools subtract -a 3T3_atac_summits.bed -b ../mm10.blacklist.bed \
> 3T3_atac_summits_bl_removed.bed
awk '{OFS="\t";} {print $1,$2-99,$3+100,$4,$5}' 3T3_atac_summits_bl_removed.bed \
> old_peak_calling_summit_window.bed
cd ..
echo 'Done'
```

## 6 ATAC Preclustering

The following Rscript processes the ATAC peaks data through the DESeq2 workflow.

[github raw](#)

```
library(bigWig)
library(DESeq2)
library(lattice)
library(DEGreport)
source('https://raw.githubusercontent.com/mjg54/znf143_pro_seq_analysis/master/docs/ZNF143_functions.R')

directory = '/scratch/bhn9by/ATAC'
setwd(directory)
preadipo.file = read.table("/scratch/bhn9by/ATAC/old_peak_calling_macs/
    old_peak_calling_summit_window.bed", header = F, sep = "\t")
```

### 6.1 Assign counts to peaks

A custom function is used to get raw counts from the .bed file and DESeq2 is used to generate a count table for all ATAC peaks.

```
get.raw.counts.interval <- function(df, path.to.bigWig, file.prefix = 'H') {
  df = df[,1:5]
  vec.names = c()
  inten.df=data.frame(matrix(ncol = 0, nrow = nrow(df)))
  for (mod.bigWig in Sys.glob(file.path(path.to.bigWig,
                                         paste(file.prefix, "*.bigWig", sep = '')))) {
    factor.name = strsplit(strsplit(mod.bigWig, "/")[[1]][length(strsplit(
      mod.bigWig, "/"))[[1]]], '\\.')[[1]][1]
    print(factor.name)
    vec.names = c(vec.names, factor.name)
    loaded.bw = load.bigWig(mod.bigWig)
    mod.inten = bed.region.bpQuery.bigWig(loaded.bw, df[,1:3])
    inten.df = cbind(inten.df, mod.inten)
  }
  colnames(inten.df) = vec.names
  r.names = paste(df[,1], ':', df[,2], '-', df[,3], sep='')
  row.names(inten.df) = r.names
  return(inten.df)
}

df.preadipo = get.raw.counts.interval(preadipo.file, directory, file.prefix = '3')
save(df.preadipo, file= 'df.preadipo.Rdata')

sample.conditions = factor(sapply(strsplit(as.character(colnames(df.preadipo)), '_'), '[' , 2))

sample.conditions = factor(sample.conditions, levels=c("t0","20min","40min",
                                                       "60min","2hr","3hr","4hr"))

deseq.counts.table = DESeqDataSetFromMatrix(df.preadipo,
```

```
as.data.frame(sample.conditions), ~ sample.conditions)

dds = DESeq(deseq.counts.table)

#counts table
normalized.counts.atac = counts(dds, normalized=TRUE)
save(normalized.counts.atac, file='normalized.counts.atac.Rdata')
```

## 6.2 Plot PCA

Principal component analysis (PCA) deconvolutes the complex data into two dimensions. Replicates of the same treatment should cluster together.

```
#PCA
rld = rlog(dds, blind=TRUE)

x = plotPCA(rld, intgroup="sample.conditions", returnData=TRUE)
plotPCAlattice(x, file = 'PCA_atac.pdf')
```

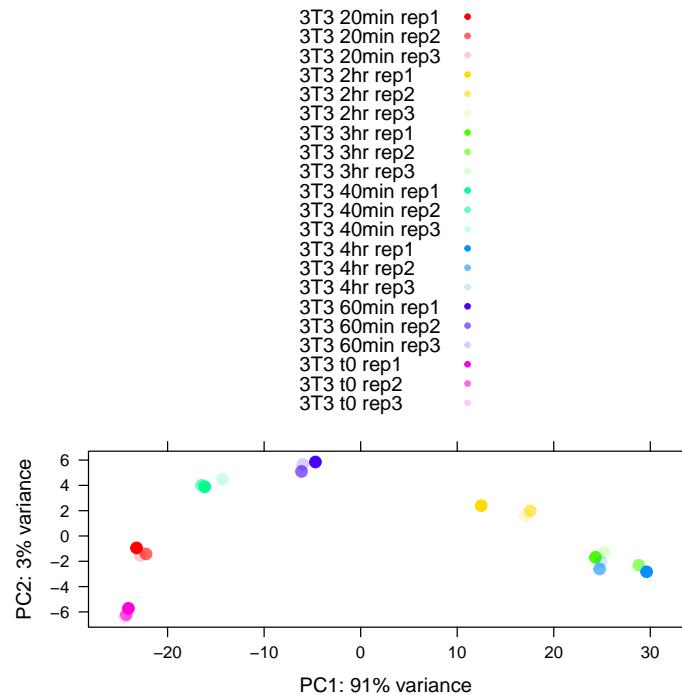


Figure 1: Expected output for PCA plot.

## 6.3 Dynamic peaks

Apply the differential functions of DESeq2 to extract dynamic peaks.

## A Model to Predict Drivers of Adipogenesis

```
dds.lrt = DESeq(dds, test="LRT", reduced = ~ 1)

res.lrt = results(dds.lrt)

padj.cutoff = 0.00000001 #1e-8

siglrt.re = res.lrt[res.lrt$padj < padj.cutoff & !is.na(res.lrt$padj),]

rld_mat <- assay(rld)
cluster_rlog = rld_mat[rownames(siglrt.re),]
meta = as.data.frame(sample.conditions)
rownames(meta) = colnames(cluster_rlog)
save(cluster_rlog, meta, sample.conditions, file = 'cluster_rlog_pval_1e8.Rdata')
```

## 7 ATAC Clustering

Cluster dynamic peaks using degPatterns. Rscript should be run as slurm job due to large memory requirement.

[github raw](#)

```
library(DESeq2)
library(DEGreport)
library(tibble)
library(lattice)

setwd('/scratch/bhn9by/ATAC')

load('cluster_rlog_pval_1e8.Rdata')

clusters.all.test.1e8 <- degPatterns(cluster_rlog, metadata = meta, minc = 100,
                                       time = "sample.conditions", col=NULL, eachStep = TRUE)

save(clusters.all.test.1e8, file = 'clusters.all.minc100.1e8.Rdata')
```

Use the following .slurm script to run the Rscript.

[github raw](#)

```
#!/bin/bash
#SBATCH -n 1
#SBATCH -t 96:00:00
#SBATCH -o atac.clustering.out
#SBATCH -p largemem
#SBATCH -A guertinlab

module load gcc/7.1.0 openmpi/3.1.4 R/4.0.0

cd /scratch/bhn9by/ATAC

Rscript atac.clustering.R
```

## 8 ATAC Postclustering

The following Rscript visualizes the ATAC clustering with plots of clusters, dendrogram, superclusters, etc.

[github raw](#)

```
library(lattice)
library(data.table)

source('plot.traces.R')

load('clusters.all.minc100.1e8.Rdata')
```

### 8.1 Generate plot.df object

```
plot.df = clusters.all.test.1e8$normalized

plot.df$sample.conditions = as.character(plot.df$sample.conditions)
plot.df$sample.conditions[plot.df$sample.conditions == 't0'] = 0
plot.df$sample.conditions[plot.df$sample.conditions == '20min'] = 20
plot.df$sample.conditions[plot.df$sample.conditions == '40min'] = 40
plot.df$sample.conditions[plot.df$sample.conditions == '60min'] = 60
plot.df$sample.conditions[plot.df$sample.conditions == '2hr'] = 120
plot.df$sample.conditions[plot.df$sample.conditions == '3hr'] = 180
plot.df$sample.conditions[plot.df$sample.conditions == '4hr'] = 240
plot.df$sample.conditions = as.numeric(plot.df$sample.conditions)
plot.df = plot.df[order(plot.df$genes),]
plot.df = plot.df[order(plot.df$sample.conditions),]

plot.df$cluster = paste('cluster', as.character(plot.df$cluster), sep = '')

plot.df$chr = sapply(strsplit(plot.df$genes, '[.)'), '[', 1)
plot.df$start = sapply(strsplit(plot.df$genes, '[.)'), '[', 2)
plot.df$end = sapply(strsplit(plot.df$genes, '[.)'), '[', 3)

save(plot.df,file='plot.df.Rdata')

write.table(cbind(plot.df$chr, plot.df$start, plot.df$end),
            file = 'dynamic_peaks.bed', quote = FALSE,
            sep = '\t', col.names=FALSE, row.names=FALSE)
```

### 8.2 Plot all clusters

```
for (i in unique(plot.df$cluster)) {
  print(i)
  write.table(plot.df[plot.df$cluster == i,
                     c('chr','start','end', 'value', 'cluster')][
                     !duplicated(plot.df[plot.df$cluster == i]$genes),],
```

```

file = paste0('cluster_bed_',
              gsub(" ", "", i, fixed = TRUE), '.bed'),
quote = FALSE, row.names = FALSE, col.names = FALSE, sep = '\t')
}

pdf('atac_clusters.pdf', width=11, height=15)

trellis.par.set(box.umbrella = list(lty = 1, col="black", lwd=1),
                box.rectangle = list( lwd=1.0, col="black", alpha = 1.0),
                plot.symbol = list(col="black", lwd=1.0, pch ='.'))
print(
xyplot(value ~ sample.conditions | cluster, group = genes, data = plot.df,
       type = c('l'),#type = c('l','p'),
       scales=list(x=list(cex=1.0,relation = "free", rot = 45),
                   y =list(cex=1.0, relation="free")),
       aspect=1.0,
       between=list(y=0.5, x=0.5),
       ylab = list(label = 'Normalized ATAC signal', cex =1.0),
       xlab = list(label = 'Time (minutes)', cex =1.0),
       par.settings = list(superpose.symbol = list(pch = c(16),
                                                    col=c('grey20'), cex =0.5),
                            strip.background=list(col="grey80"),
                            superpose.line = list(col = c('#99999980'), lwd=c(1),
                                                  lty = c(1))),
       panel = function(x, y, ...) {
         panel.xyplot(x, y, ...)
         panel.bwplot(x, y, pch = '|', horizontal = FALSE, box.width = 15,
                       do.out = FALSE)
         panel.spline(x, y, col = 'blue', lwd =2.0, ...)
       })
dev.off()

#up.flat - 9,23,17,11
#grad.up - 5,8,10,1
#up.down - 6,2,12
#grad.down - 7,3,4,13,14
#down.up - 24

```

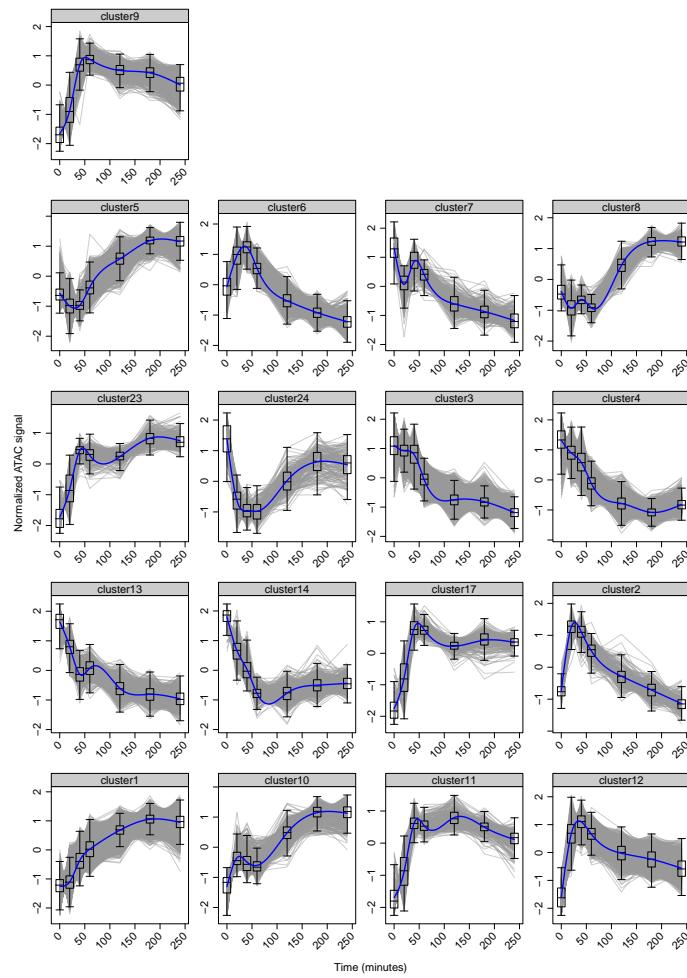


Figure 2: Plot for 17 ATAC clusters.

### 8.3 Plot dendrogram

```

x = as.data.table(plot.df)
plot.df.cluster = dcast(x, genes + cluster ~ sample.conditions, value.var="value")

avg.clusters = as.data.frame(matrix(nrow = 0, ncol = 7))
for (i in unique(plot.df.cluster$cluster)) {
  z = data.frame(matrix(colMeans(plot.df.cluster[plot.df.cluster$cluster == i,3:9]),
                        ncol = 7, nrow = 1))
  rownames(z) = c(i)
  colnames(z) = as.character(colnames(plot.df.cluster)[3:9])
  avg.clusters = rbind(avg.clusters, z)
}

dd = dist(avg.clusters)

```

```

hc = hclust(dd, method = "complete")

pdf('dendrogram.pdf', width=8, height=5)
plot(hc, xlab = "Clusters", main = ' ', hang = -1)
abline(h = 2, lty = 2)
dev.off()

```

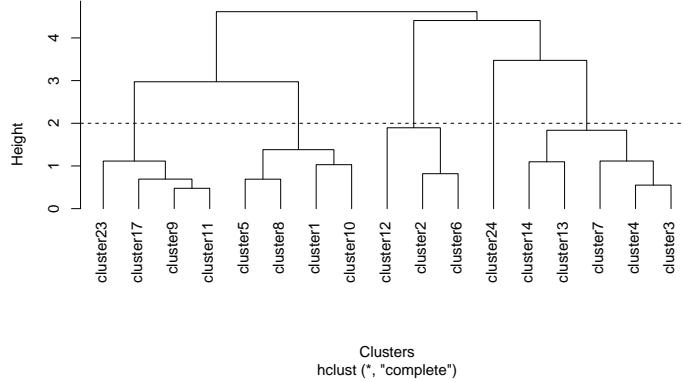


Figure 3: Hierarchical clustering groups ATAC clusters based on similarity in dynamics. Dendrogram shows a threshold of 2 which divides the 17 clusters into 5 superclusters.

## 8.4 Plot clusters organized by supercluster

```

df = data.frame(index=1:17,cluster=unique(plot.df$cluster)[order(unique(plot.df$cluster))])
df$cluster.num = as.integer(sapply(strsplit(df$cluster, 'cluster'), '[' , 2))
df = df[order(df$cluster.num),]
df = df[reorder(df$cluster.num,c(23,17,9,11,5,8,1,10,12,2,6,24,14,13,7,4,3)),]

pdf('atac_clusters_org_by_sc.pdf', width=11, height=15)

trellis.par.set(box.umbrella = list(lty = 1, col="black", lwd=1),
                box.rectangle = list( lwd=1.0, col="black", alpha = 1.0),
                plot.symbol = list(col="black", lwd=1.0, pch ='.'))
print(
  xyplot(value ~ sample.conditions | cluster, group = genes, data = plot.df,
  type = c('l'),#type = c('l','p'),
  scales=list(x=list(cex=1.0,relation = "free", rot = 45),
  y =list(cex=1.0, relation="free")),
  aspect=1.0,
  layout = c(5,5),
  between=list(y=0.5, x=0.5),
  index.cond=list(rev(df$index)),
  skip = c(F,F,F,F,F,
           F,T,T,T,T,
           F,F,F,T,T,
           F,F,F,F,T,

```

```

F,F,F,F,T),
ylab = list(label = 'Normalized ATAC signal', cex =1.0),
xlab = list(label = 'Time (minutes)', cex =1.0),
par.settings = list(superpose.symbol = list(pch = c(16),
                                             col=c('grey20'), cex =0.5),
                      strip.background=list(col="grey80"),
                      superpose.line = list(col = c('#99999980'), lwd=c(1),
                                            lty = c(1))),
panel = function(x, y, ...) {
  panel.xyplot(x, y, ...)
  panel.bwplot(x, y, pch = '|', horizontal = FALSE, box.width = 15,
               do.out = FALSE)
  panel.spline(x, y, col = 'blue', lwd =2.0, ...)
}
dev.off()

```

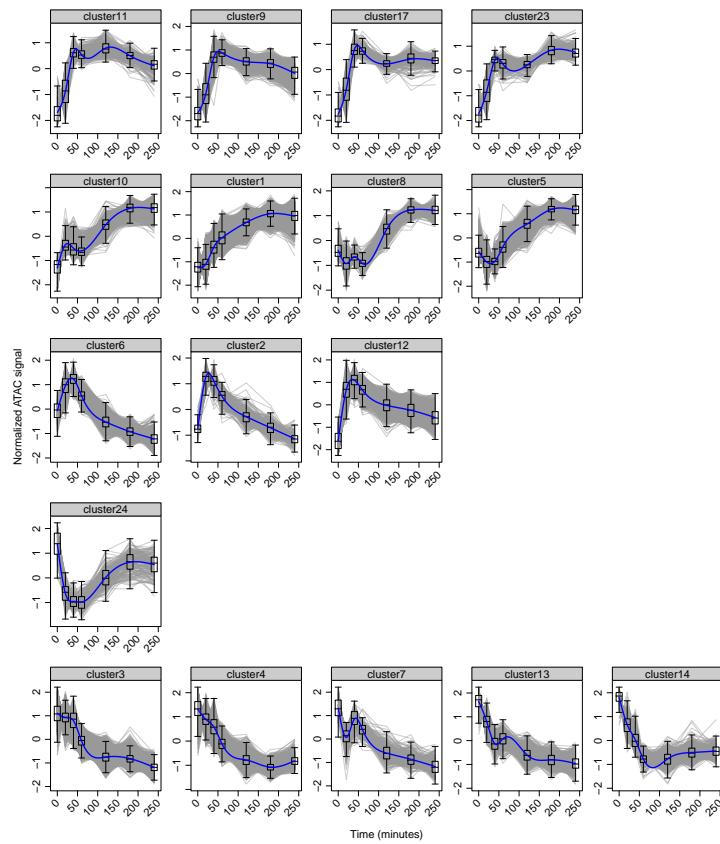


Figure 4: ATAC clusters are organized according to their superclusters.

## 8.5 Plot supercluster traces

```

gradual.down = plot.df[plot.df$cluster == 'cluster7' |
                      plot.df$cluster == 'cluster3' |
                      plot.df$cluster == 'cluster4' |
                      plot.df$cluster == 'cluster13' |
                      plot.df$cluster == 'cluster14',]

down.up = plot.df[plot.df$cluster == 'cluster24',]

gradual.up = plot.df[plot.df$cluster == 'cluster5' |
                      plot.df$cluster == 'cluster8' |
                      plot.df$cluster == 'cluster10' |
                      plot.df$cluster == 'cluster1',]

up.flat = plot.df[plot.df$cluster == 'cluster9' |
                      plot.df$cluster == 'cluster23' |
                      plot.df$cluster == 'cluster17' |
                      plot.df$cluster == 'cluster11',]

up.down = plot.df[plot.df$cluster == 'cluster6' |
                      plot.df$cluster == 'cluster12' |
                      plot.df$cluster == 'cluster2',]

gradual.down$supercluster = 'gradual.down'
down.up$supercluster = 'down.up'
gradual.up$supercluster = 'gradual.up'
up.flat$supercluster = 'up.flat'
up.down$supercluster = 'up.down'

nrow(gradual.down)/7
nrow(down.up)/7
nrow(gradual.up)/7
nrow(up.flat)/7
nrow(up.down)/7

plot.df.atac = rbind(gradual.down,
                     down.up,
                     gradual.up,
                     up.flat,
                     up.down)
plot.df.atac = plot.df.atac[, -(7:26)]
plot.df.atac$genes = paste0(plot.df.atac$chr, ':', plot.df.atac$start, '-', plot.df.atac$end)
save(plot.df.atac, file='plot.df.atac.Rdata')

pdf('atac_superclusters.pdf', width=6.83, height=5)

trellis.par.set(box.umbrella = list(lty = 1, col="black", lwd=1),
                box.rectangle = list( lwd=1.0, col="black", alpha = 1.0),
                plot.symbol = list(col="black", lwd=1.0, pch ='.'))
print(

```

```

xyplot(value ~ sample.conditions | supercluster, group = genes,
       data = plot.df.atac, type = c('l'), #type = c('l','p'),
       scales=list(x=list(cex=1.0,relation = "free", rot = 45),
                   y =list(cex=1.0, relation="free")),
       aspect=1.0,
       between=list(y=0.5, x=0.5),
       layout = c(5,1),
       ylab = list(label = 'Normalized ATAC signal', cex =1.0),
       xlab = list(label = 'Time (minutes)', cex =1.0),
       par.settings = list(superpose.symbol = list(pch = c(16),
                                                    col=c('grey20'), cex =0.5),
                            strip.background=list(col="grey80"),
                            superpose.line = list(col = c('#99999980'), lwd=c(1),
                                                  lty = c(1))),
       panel = function(x, y, ...) {
         panel.xyplot(x, y, ...)
         panel.bwplot(x, y, pch = '|', horizontal = FALSE, box.width = 15,
                       do.out = FALSE)
         panel.spline(x, y, col = 'blue', lwd =2.0, ...)
       })
dev.off()

```

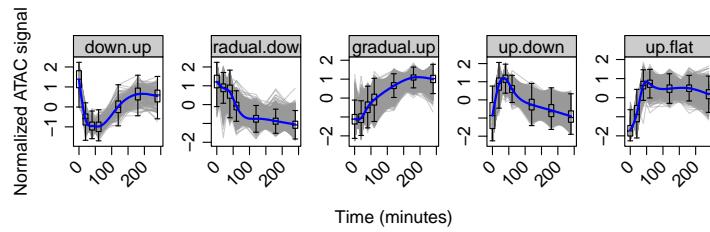


Figure 5: Clusters are weighed and averaged to plot supercluster traces.

## 8.6 Plot individual traces

```

df = data.frame(sc = c("down.up", "gradual.down", "gradual.up", "up.down", "up.flat"),
                col = c('orange1', '#4169E1', '#FF0000', 'green2', '#A020F0'))

```

```

for(sc in unique(plot.df.atac$supercluster)) {
  col = df[df$sc == sc,]$col

  y = plot.df.atac[plot.df.atac$supercluster == sc,]

  pdf(file=paste0(sc, '.traces.pdf'))

  trellis.par.set(box.umbrella = list(lty = 1, col="black", lwd=1),
                  box.rectangle = list( lwd=1.0, col="black", alpha = 1.0),
                  plot.symbol = list(col="black", lwd=1.0, pch ='.'))
  
  print(
    xyplot(value ~ sample.conditions, group = genes, data = y, type = c('l'),
           scales=list(x=list(cex=1.0,relation = "free", rot = 45),
                        y = list(cex=1.0, relation="free")),
           aspect=1.0,
           between=list(y=0.5, x=0.5),
           main = list(label = paste0(sc, ' traces'), cex = 1.5),
           ylab = list(label = 'Normalized ATAC signal', cex =1.0),
           xlab = list(label = 'Time (minutes)', cex =1.0),
           par.settings = list(superpose.symbol = list(pch = c(16),
                                                       col=c('grey20'), cex =0.5),
                                strip.background = list(col="grey80"),
                                superpose.line = list(col = c('#99999980'),
                                                       lwd=c(1),lty = c(1))),
           panel = function(x, y, ...) {
             panel.xyplot(x, y, ...)
             panel.bwplot(x, y, pch = '|', horizontal = FALSE, box.width = 15,
                           do.out = FALSE)
             panel.spline(x, y, col = 'blue', lwd = 3.5, ...)
             #replace col = 'blue' with col = col for different sc colors
           })
  )
}

dev.off()
}

```

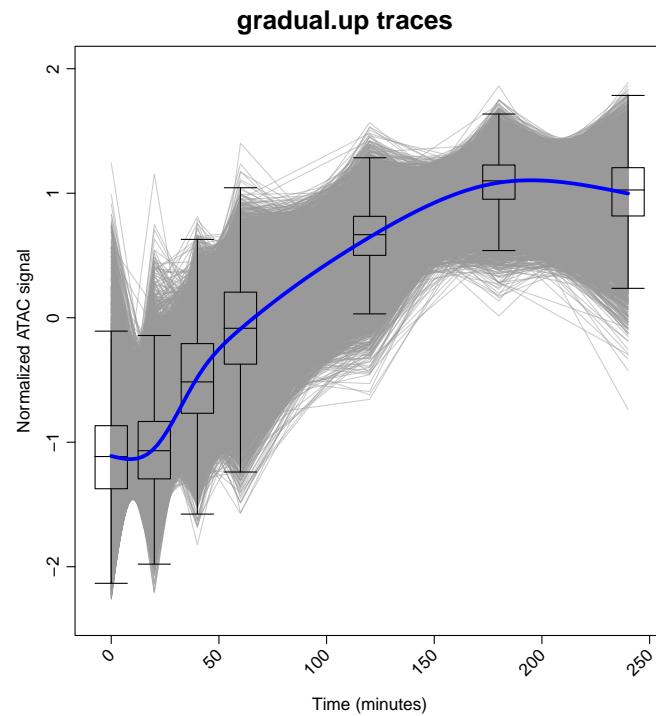


Figure 6: Example of output for one supercluster traces.

## 8.7 Generating .bed file for each cluster

We will generate .bed file for each cluster in order to perform motif enrichment analyses.

```
for (i in unique(plot.df.atac$supercluster)) {
  print(i)
  write.table(plot.df.atac[plot.df.atac$supercluster == i,
    c('chr','start','end', 'value', 'supercluster')][
      !duplicated(plot.df.atac[plot.df.atac$supercluster == i,]$genes),],
    file = paste0(i, '.bed'),
    quote = FALSE, row.names = FALSE, col.names = FALSE, sep = '\t')
}
```

## 9 FIMO motif enrichment

---

### 9.1 Download required script

[fimo\\_motif\\_enrichment.R](#): this Rscript performs a Chi-squared test on each motif to determine significance and outputs a barplot for enrichment in clusters.

github raw

```
#!/bin/bash

cd /scratch/bhn9by/ATAC
wget https://raw.githubusercontent.com/guertinlab/adipogenesis/master/Bao_code_chunks/misc_scripts/fimo_motif_enrichment.R
```

### 9.2 Find motifs enriched in cluster

github raw

```
#!/bin/bash
#SBATCH -n 1
#SBATCH -t 96:00:00
#SBATCH -o fimo.enrichment.out
#SBATCH -p standard
#SBATCH -A guertinlab

module load gcc/7.1.0 bedtools/2.26.0 openmpi/3.1.4 R/4.0.0

cd /scratch/bhn9by/ATAC

#collect ATAC peaks that were not sorted into clusters
#this will serve as a nondynamic control set of regions
intersectBed -v -a old_peak_calling_macs/old_peak_calling_summit_window.bed \
             -b cluster*bed > nondynamic_peaks.bed

tab=$'\t'

mkdir fimo_motif_enrichment
cd fimo_motif_enrichment

#loop through each motif
for motif in /scratch/bhn9by/ATAC/Top_motif/*bed
do
    motif_name=$(echo $motif | awk -F"/" '{print $NF}' | awk -F".bed" '{print $1}')
    echo $motif_name
    touch ${motif_name}.txt
    echo "name"$tab"with.motif"$tab"without.motif" >> ${motif_name}.txt

    #find how many nondynamic peaks contain the motif
    nondyn_with_motif=$(intersectBed -wa -a ../nondynamic_peaks.bed -b $motif | wc -l)
    nondyn_without_motif=$(intersectBed -v -a ../nondynamic_peaks.bed -b $motif | wc -l)
```

## A Model to Predict Drivers of Adipogenesis

```
echo "nondynamic$tab$nodyn_with_motif$tab$nodyn_without_motif" >> ${motif_name}.txt

#loop though each cluster
for bed in ../cluster*bed
do
name=$(echo $bed | awk -F"${cluster_bed_}" '{print $2}' | awk -F".bed" '{print $1}')
#find how many peaks in each cluster contain the motif

#calculate percentage of cluster peaks that have the motif
cluster_with_motif=$(intersectBed -wa -a $bed -b $motif | wc -l)
cluster_without_motif=$(intersectBed -v -a $bed -b $motif | wc -l)

echo "$name$tab$cluster_with_motif$tab$cluster_without_motif" >> ${motif_name}.txt

done

Rscript ../fimo_motif_enrichment.R ${motif_name}.txt

done
cd ..
```

## 10 MEME motif enrichment

---

### 10.1 Generate slurm file for each replicate and run in parallel

The content of each header is provided in the next subsection.

[github raw](#)

```
#!/bin/bash

#run MEME on each cluster to find enriched motifs
wget https://hgdownload-test.gi.ucsc.edu/goldenPath/mm10/bigZips/mm10.chrom.sizes
mkdir meme_motif_enrichment

for i in *cluster*bed
do
    name=$(echo $i | awk -F "_" '{print $NF}' | awk -F ".bed" '{print $1}')
    echo $name
    echo '#SBATCH -o '$name'.meme.out' > temp.txt
    echo 'i='\$i > temp2.txt
    cat meme_slurm_header_1.txt temp.txt \
        meme_slurm_header_2.txt temp2.txt \
        meme_slurm_header_3.txt > $name.meme.slurm
    sbatch $name.meme.slurm
    rm temp.txt
    rm temp2.txt
done
```

#### 10.1.1 meme\_slurm\_header\_1.txt

[github raw](#)

```
#!/bin/bash
#SBATCH -n 60
#SBATCH --mem-per-cpu=6000
#SBATCH -t 72:00:00
```

#### 10.1.2 meme\_slurm\_header\_2.txt

[github raw](#)

```
#SBATCH -p parallel
#SBATCH -A guertinlab

module load gcc/7.1.0 mvapich2/2.3.3 bedtools/2.26.0 meme/5.1.0
```

#### 10.1.3 meme\_slurm\_header\_3.txt

[github raw](#)

```
cd meme_motif_enrichment
```

```

name=$(echo $i | awk -F"\_" '{print $NF}' | awk -F".bed" '{print $1}')
echo $name

slopBed -i ../$i \
-g ../mm10.chrom.sizes -b -50 | \
fastaFromBed -fi /project/genomes/Mus_musculus/UCSC/mm10/Sequence/WholeGenomeFasta/genome.fa \
-bed stdin -fo ${name}.fasta
head ${name}.fasta
srun meme -p 59 -oc ${name}.meme_output -nmotifs 15 \
-objfun classic -evt 0.01 -searchsize 0 -minw 6 \
-maxw 18 -revcomp -dna -markov_order 3 -maxsize 100000000 \
${name}.fasta

cd ..

```

## 10.2 Simplified Version

For clarity we provided a simplified, sequential version of the MEME script.

[github raw](#)

```

#!/bin/bash
#SBATCH -n 60
#SBATCH --mem-per-cpu=6000
#SBATCH -t 72:00:00
#SBATCH -o meme.enrichment.out
#SBATCH -p parallel
#SBATCH -A guertinlab

module load gcc/7.1.0 mvapich2/2.3.3 bedtools/2.26.0 meme/5.1.0

wget https://hgdownload-test.gi.ucsc.edu/goldenPath/mm10/bigZips/mm10.chrom.sizes
mkdir meme_motif_enrichment
cd meme_motif_enrichment

#run MEME on each cluster to find enriched motifs
for i in ../*cluster*bed
do
    name=$(echo $i | awk -F"\_" '{print $NF}' | awk -F".bed" '{print $1}')
    echo $name

    slopBed -i ../$i \
    -g ../mm10.chrom.sizes -b -50 | \
    fastaFromBed -fi /project/genomes/Mus_musculus/UCSC/mm10/Sequence/WholeGenomeFasta/genome.fa \
    -bed stdin -fo ${name}.fasta
    head ${name}.fasta
    srun meme -p 59 -oc ${name}.meme_output -nmotifs 15 \
    -objfun classic -evt 0.01 -searchsize 0 -minw 6 \
    -maxw 18 -revcomp -dna -markov_order 3 -maxsize 100000000 \
    ${name}.fasta
done

```

## 11 MEME-FIMO analysis

---

### 11.1 Prepare MEME databases

[github raw](#)

```
#Generate homer_uniprobe_jaspar PSWM database

wget http://meme-suite.org/meme-software/Databases/motifs/motif_databases.12.19.tgz
tar -xzf motif_databases.12.19.tgz
#JASPAR
mv motif_databases/JASPAR/JASPAR2018_CORE_vertebrates_non-redundant.meme $PWD
#Uniprobe
mv motif_databases/MOUSE/uniprobe_mouse.meme $PWD

#Homer
#CAUTION: the HOMER_MEME_conversion.py was written for python2 so remember to specify python2.7 if you have python3 installed
wget https://raw.githubusercontent.com/mjg54/znf143_pro_seq_analysis/master/docs/HOMER_MEME_conversion.py
wget http://homer.ucsd.edu/homer/custom.motifs
python2.7 HOMER_MEME_conversion.py -i custom.motifs -o homer.motifs

#edit databases to work with tomtom
cp JASPAR2018_CORE_vertebrates_non-redundant.meme JASPAR_edited_meme.txt
grep MOTIF JASPAR_edited_meme.txt > motifs.txt
cat motifs.txt | while read motif
do
    name=$(echo $motif | awk -F" " '{print $NF}')
    temp=$(echo 'MOTIF' $name'_jaspar')
    echo $temp
    sed -i "s;${motif};${temp};g" JASPAR_edited_meme.txt
done
rm motifs.txt

#edit databases to work with tomtom
cp uniprobe_mouse.meme uniprobe_edited_meme.txt
grep MOTIF uniprobe_edited_meme.txt > motifs.txt
cat motifs.txt | while read motif
do
    name=$(echo $motif | awk -F" " '{print $NF}')
    temp=$(echo 'MOTIF' $name'_uniprobe')
    echo $temp
    sed -i "s;${motif};${temp};g" uniprobe_edited_meme.txt
done
#remove 'secondary' motifs
sed -i -e '4210,8365d;' uniprobe_edited_meme.txt

rm motifs.txt

#edit databases to work with tomtom
cp homer.motifs_meme.txt homer_edited_meme.txt
grep MOTIF homer_edited_meme.txt > motifs.txt
```

```

cat motifs.txt | while read motif
do
    name=$(echo $motif | awk -F" " '{print $NF}')
    name=$(echo $name | awk -F"/" '{print $1}')
    temp=$(echo 'MOTIF' $name'_homer')
    echo $temp
    sed -i "s;${motif};${temp};g" homer_edited_meme.txt
done
rm motifs.txt

#Collect all database motifs into one file
cat homer_edited_meme.txt \
    uniprobe_edited_meme.txt JASPAR_edited_meme.txt > homer_uniprobe_jaspar_edited.txt

```

## 11.2 Query MEME motifs in TOMTOM

[github raw](#)

```

module load gcc/9.2.0 mvapich2/2.3.3 meme/5.1.0

#run tomtom on meme output
mkdir tomtom
cd tomtom

echo 'Running TOMTOM'
for i in ../meme_motif_enrichment/*output
do
    name=$(echo $i | awk -F"/" '{print $NF}' | awk -F".meme" '{print $1}')
    echo $name
    tomtom -no-ssc -o $name.tomtom_output -verbosity 1 -min-overlap 5 \
        -dist ed -evalue -thresh 0.05 \
        $i/meme.txt ../homer_uniprobe_jaspar_edited.txt
    tomtom -no-ssc -o $name.tomtom_output -verbosity 1 -min-overlap 5 -dist \
        ed -text -evalue -thresh 0.05 \
        $i/meme.txt ../homer_uniprobe_jaspar_edited.txt > $name.tomtom_output/tomtom.txt
done

cd ..

mkdir motifid_clusters
cd motifid_clusters

#extract motif names from tomtom output
echo 'Extracting Motif Names'
for file in ../tomtom/*cluster*tomtom_output/*.txt
do
    name=$(echo $file | awk -F"/" '{print $(NF-1)}' | awk -F".tomtom_output" '{print $1}')
    echo $name
    motifid=$name
    #echo $file

```

```

linenum=$(awk 'END {print NR}' $file)
#echo $linenum
first=2
i=$first
if [[ $linenum != 5 ]]
then
mkdir $name
while [[ $i -le $linenum ]]
do
    head -$i $file | tail -1 > lastline
    mapid[$i]=$(awk 'END {print $2}' lastline | awk -F"_" '{print $1}')
    #echo mapid[$i]_${mapid[$i]}
    echo ${mapid[$i]} >> $name/motifidlist_$motifid.txt
    ((i = i + 1))
done
head -n $(( $(wc -l $name/motifidlist_$motifid.txt | awk '{print $1}') - 4 )) \
$name/motifidlist_$motifid.txt > $name/motifidlist_$motifid.final.txt
rm $name/motifidlist_$motifid.txt
mv $name/motifidlist_$motifid.final.txt $name/motifidlist_$motifid.txt
fi
done

rm lastline
cd ..

```

### 11.3 Organize MEME superclusters

[github raw](#)

```

#organize de novo motifs by supercluster
echo 'Sort MEME motifs by supercluster'
mkdir supercluster_meme_motifs
cd supercluster_meme_motifs

mkdir up.down
mkdir up.flat
mkdir grad.up
mkdir down.up
mkdir grad.down

#up.flat - 9,23,17,11
#grad.up - 5,8,10,1
#up.down - 6,2,12
#grad.down - 7,3,4,13,14
#down.up - 24

cp -r ../../motifid_clusters/cluster6 up.down
cp -r ../../motifid_clusters/cluster2 up.down
cp -r ../../motifid_clusters/cluster12 up.down

```

```
cp -r ./motifid_clusters/cluster9 up.flat
cp -r ./motifid_clusters/cluster23 up.flat
cp -r ./motifid_clusters/cluster17 up.flat
cp -r ./motifid_clusters/cluster11 up.flat

cp -r ./motifid_clusters/cluster5 grad.up
cp -r ./motifid_clusters/cluster8 grad.up
cp -r ./motifid_clusters/cluster10 grad.up
cp -r ./motifid_clusters/cluster1 grad.up

#no motifs tomtom'd out from cluster24 meme result
#cp -r ./motifid_clusters/cluster24 down.up
rm -r down.up

cp -r ./motifid_clusters/cluster7 grad.down
cp -r ./motifid_clusters/cluster3 grad.down
cp -r ./motifid_clusters/cluster4 grad.down
cp -r ./motifid_clusters/cluster13 grad.down
cp -r ./motifid_clusters/cluster14 grad.down

for dir in *.*
do
    echo $dir
    cd $dir
    for int_dir in cluster*
    do
        cd $int_dir
        mv motifidlist* ..
        cd ..
        done
        cat *txt > $dir.denovo.motifs.txt
        mv $dir.denovo.motifs.txt ..
        cd ..
    done
done

cd ..
```

## 11.4 Find matches between MEME and FIMO

```
github raw

module load gcc/7.1.0 openmpi/3.1.4 R/4.0.0

#organize fimo motifs by supercluster
echo 'Sort FIMO motifs by supercluster'
mkdir supercluster_fimo_motifs

Rscript sort_fimo_motifs_supercluster.R

#Match the de novo with the fimo identified motifs
```

```
echo 'Match MEME and FIMO motifs by supercluster'
mkdir fimo_denovo_match
cd fimo_denovo_match

for file in ../supercluster_meme_motifs/*.txt
do
    name=$(echo $file | awk -F"/" '{print $NF}' | awk -F".denovo" '{print $1}')
    echo $name
    awk 'FNR==NR{a[$1];next}{$1 in a}{print}' $file \
        ../supercluster_fimo_motifs/$name.fimo.motifs.txt \
        > ${name}_meme_fimo_match.txt
    wordcount=$(wc -l ${name}_meme_fimo_match.txt | awk 'END {print $1}')
    if [[ $wordcount == 0 ]]
    then
        rm ${name}_meme_fimo_match.txt
    fi
done

cat * > all_matched_motifs.txt
cp all_matched_motifs.txt ..

cd ..
```

## 12 Defining motif families

---

### 12.1 Query factors

[github raw](#)

```
#!/bin/bash

#pulls each de novo motif from all the clusters and saves each as its own file

#extract individual meme files from combined database
#CAUTION: you only need to run this once, even if you're working through the code again
mkdir individual_memes
cd individual_memes

#CAUTION: the MEME_individual_from_db.py was written for Python2 so remember to specify python2.7 when running
wget https://github.com/guertinlab/adipogenesis/blob/2fdf0bbab4fe6f368f5a60e42f7899b6570ff71c/motif_analysis/MEME_individual_from_db.py
python2.7 ./MEME_individual_from_db.py -i ../homer_uniprobe_jaspar_edited.txt

for file in *meme.txt
do
    name=$(echo $file | awk -F"homr_uniprobe_jaspar_edited.txt_" '{print $1}')
    mv $file ${name}meme.txt

done

cd ..

#tomtom all query factors against all others
mkdir tomtom_all_query_factors
cd tomtom_all_query_factors

cat ../all_matched_motifs.txt | while read factor
do
    echo $factor
    cp ../individual_memes/${factor}_meme.txt $PWD
done

#remove Ptfla motif b/c it's a forced palindrome
rm Ptfla_homer_meme.txt
#remove Tal1:Tcf3 motif b/c motif doesn't make biological sense - looks like neither Tal1 nor Tcf3
rm TAL1::TCF3_jaspar_meme.txt
#remove ZNF740 motif
rm ZNF740_jaspar_meme.txt
#remove Pitx1:Ebox b/c it's conflated with TWIST1
rm Pitx1:Ebox_homer_meme.txt

cat *meme.txt > ../all_query_factors_meme.txt

#define motif families
module load gcc/9.2.0  mvapich2/2.3.3  meme/5.1.0
```

```

for meme in *meme.txt
do
    name=$(echo $meme | awk -F".txt" '{print $NF}' | awk -F"_meme.txt" '{print $1}')
    #echo $name
    tomtom -no-ssc -o $name.tomtom_output -verbosity 1 -incomplete-scores -min-overlap 1 -dist ed -evalue -1
    cd $name.tomtom_output
    cut -f1,2,5 tomtom.tsv | tail -n +2 | sed '$d' | sed '$d' | sed '$d' | sed '$d' >> ../../3_col_combined_motif_db_pre.txt
    cd ..
done

grep -v '#' 3_col_combined_motif_db_pre.txt > 3_col_combined_motif_db.txt
rm 3_col_combined_motif_db_pre.txt
cp 3_col_combined_motif_db.txt ..
cd ..

```

## 12.2 Visualize PSWM

[github raw](#)

```

library(igraph)
library(dichromat)

setwd('/scratch/bhn9by/ATAC')

threecol=read.table("3_col_combined_motif_db.txt",header=F,stringsAsFactors = F,sep='\t')
colnames(threecol)=c('from','to','e_value')
threecol$weight=abs(log(threecol$e_value))

#create the graph variable
g=graph.data.frame(threecol,directed=F)
g=simplify(g)

cluster=clusters(g)

for(i in 1:length(groups(cluster))) {
  write.table(groups(cluster)[i],file=paste0('PSWM_family_',i,'.txt'), \
  col.names = F, row.names = F, quote = F, sep = '\t')
}

l=layout.fruchterman.reingold(g)
l=layout.norm(l,-1,1,-1,1)

colfunc<-colorRampPalette(c("red","yellow","springgreen","royalblue","purple"))
#pick a distinct color from the palette for each disease and save the list of colors as a vector
mycol = colfunc(length(groups(cluster)))

pdf(file='families.pdf',width=10,height=10)
plot(g,layout=l,rescale=F,vertex.label.cex=.5,xlim=range(l[,1]), ylim=range(l[,2]),
edge.width=E(g)$weight/20,vertex.size=degree(g,mode='out')/5,
edge.curved=T,vertex.label=NA,vertex.color=mycol[cluster$membership],

```

```
margin=0,asp=0)  
dev.off()
```

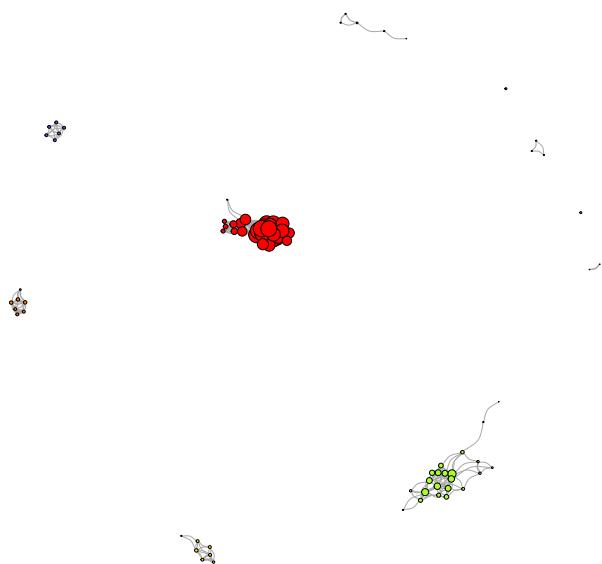


Figure 7: Clustering plot of motif families

## 13 Generating composite motifs

---

### 13.1 Download required scripts

[tomtom\\_output\\_to\\_composite.py](#): this Python script convert the .xml output of tomtom into a PSWM .txt and an offset/reverse complement index .txt, required for generating composite motif.

[generate\\_composite\\_motif.R](#): this Rscript aligns and averages the PSWM of each motif to generate a composite motif.

[meme\\_header.txt](#): this .txt contains the standard MEME suite PSWM header.

github raw

```
#!/bin/bash

cd /scratch/bhn9by/ATAC
wget https://raw.githubusercontent.com/guertinlab/adipogenesis/master/Bao_code_chunks/misc_scripts/tomtom_out
wget https://raw.githubusercontent.com/guertinlab/adipogenesis/master/Bao_code_chunks/misc_scripts/generate_
wget https://raw.githubusercontent.com/guertinlab/adipogenesis/master/Bao_code_chunks/misc_scripts/meme_head
```

### 13.2 Query TOMTOM to calculate index and values for PSWM

github raw

```
#!/bin/bash

cd /scratch/bhn9by/ATAC

mkdir composite_motifs
cd composite_motifs

#query tomtom for each factor against all others
module purge
module load gcc/9.2.0 mvapich2/2.3.3 meme/5.1.0
for txt in ../PSWM_family*.txt
do

    dir_name=$(echo $txt | awk -F'/' '{print $2}' | awk -F'.txt' '{print $1}')
    echo $dir_name
    mkdir $dir_name
    cd $dir_name

    cat ../$txt | while read line
    do
        echo $line
        cp ../../individual_memes/${line}_meme.txt $PWD
    done
    echo ''

    query_factor='head -1 ../$txt'
```

```

if [[ $(wc -l < ../$txt) -ge 2 ]]
then
cat ../$txt | { while read line
do
    query_factor=$line
    rm ref_factors_meme.txt
    mv ${query_factor}_meme.txt ..
    cat *_meme.txt > ref_factors_meme.txt
    mv ../${query_factor}_meme.txt $PWD
    tomtom -no-ssc -o ${query_factor}.tomtom_output -verbosity 1 -incomplete-scores \
        -min-overlap 1 -dist ed -evaluate -thresh 0.0005 \
        ${query_factor}_meme.txt ref_factors_meme.txt

    if [[ $(wc -l < ${query_factor}.tomtom_output/tomtom.tsv) -ge $max_motif ]]
    then
        max_motif=$(wc -l < ${query_factor}.tomtom_output/tomtom.tsv)
        final_query=$query_factor
    fi

done
echo FINAL_QUERY IS $final_query
wc -l ${final_query}.tomtom_output/tomtom.tsv
cd ${final_query}.tomtom_output
python2.7 ../../tomtom_output_to_composite.py -i tomtom.xml
mv tomtom.xml_test_index_pswm.txt ../composite.values.txt
mv tomtom.xml_test_index_rc_offset.txt ../composite.index.txt
cd ../../
}
fi

cd ..

done
module purge

```

### 13.3 Generate composite motif and sequence logo

[github raw](#)

```

#!/bin/bash

for family in PSWM*
do
    cd $family
    num=$(ls *txt | wc -l)

    if [[ $num -ge 2 ]]
    then

        #generate composite PSWM

```

```

module load gcc/7.1.0 openmpi/3.1.4 R/4.0.0
Rscript ../../generate_composite_motif.R $family
cat ../../meme_header.txt ${family}_composite_PWM.txt > ${family}_meme.txt
else
line=`grep MOTIF *meme.txt`
cp *meme.txt ${family}_meme.txt
sed -i "s;${line};MOTIF    Composite;g" ${family}_meme.txt
fi

#generate logo
module load gcc/7.1.0 meme/4.10.2
ceqlogo -i ${family}_meme.txt -m Composite -o ${family}.eps
ceqlogo -i ${family}_meme.txt -m Composite -o ${family}.rc.eps -r
cd ..

done

cd ..
module purge

```

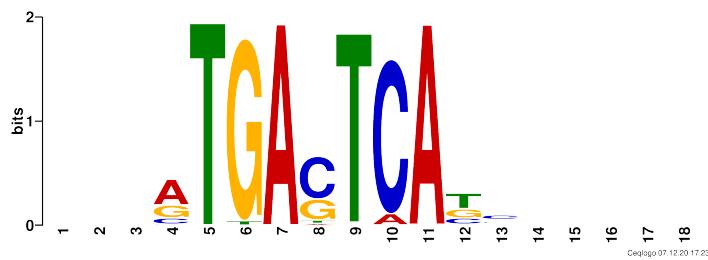


Figure 8: Sequence logo for the composite AP1 motif.

## 14 Running FIMO on composite and splitting SP/KLF

### 14.1 Download required scripts

[generate\\_composite\\_motif.SP.R](#): this Rscript is similar to the previous generate\_composite\_motif.R but is specific for SP.

[generate\\_composite\\_motif.KLF.R](#): this Rscript is similar to the previous generate\_composite\_motif.R but is specific for KLF.

[prep.SP.KLF.fimo.R](#): this Rscript [insert function]

[SP\\_KLF\\_split.R](#): this Rscript [insert function]

[github raw](#)

```
#!/bin/bash

cd /scratch/bhn9by/ATAC/
wget https://raw.githubusercontent.com/guertinlab/adipogenesis/master/Bao_code_chunks/misc_scripts/generate_
wget https://raw.githubusercontent.com/guertinlab/adipogenesis/master/Bao_code_chunks/misc_scripts/generate_
wget https://raw.githubusercontent.com/guertinlab/adipogenesis/master/Bao_code_chunks/misc_scripts/prep.SP.KL
wget https://raw.githubusercontent.com/guertinlab/adipogenesis/master/Bao_code_chunks/misc_scripts/SP_KLF_sp
```

### 14.2 Generate slurm file for each replicate and run in parallel

The content of each header is provided in the next subsection.

[github raw](#)

```
#!/bin/bash

#run fimo against PSWM and take top 2 million hits
rm -r fimo_composites
mkdir fimo_composites

for i in PSWM*.txt
do
    name=$(echo $i | awk -F".txt" '{print $1}')
    echo $name
    echo '#SBATCH -o '$name'.fimo.out' > temp.txt
    echo 'i=../composite_motifs/'$name'/'${name}_meme.txt > temp2.txt
    cat fimo_slurm_header_1.txt temp.txt \
        fimo_slurm_header_2.txt temp2.txt \
        fimo_slurm_header_3.txt > $name.fimo.slurm
    sbatch $name.fimo.slurm
    rm temp.txt
    rm temp2.txt
done
```

#### 14.2.1 fimo\_slurm\_header\_1.txt

[github raw](#)

```
#!/bin/bash  
#SBATCH -n 1  
#SBATCH -t 96:00:00
```

## 14.2.2 fimo slurm header 2.txt

github raw

```
#SBATCH -p standard  
#SBATCH -A guertinlab  
  
module load gcc/7.1.0  meme/4.10.2
```

### 14.2.3 fimo slurm header 3.txt

github raw

```

cd /scratch/bhn9by/ATAC/fimo_composites

name=$(echo $i | awk -F"/" '{print $NF}'| awk -F"_meme.txt" '{print $1}')

fimo --thresh 0.001 --text $i /project/genomes/Mus_musculus/UCSC/mm10/Sequence/WholeGenomeFasta/genome.fa \
> ${name}_composite_fimo.txt

#this takes top 2M
score=$(tail -n +2 ${name}_composite_fimo.txt | sort -nrk6,6 | awk 'FNR == 2000000 {print $6}')
echo $score
tail -n +2 ${name}_composite_fimo.txt | awk -v sco="$score" '{ if ($6 >= sco) { print } }' | \
awk '{OFS="\t";} {print $2,$3,$4,$7,$6,$5,$8}' > ${name}_2M.txt

#this was to get the order of conformity to consensus.
tomtom -no-ssc -oc ${name}_ranks.tomtom_output -verbosity 1 -incomplete-scores -min-overlap 1 \
-dist ed -evaluate -thresh 0.05 ../all_query_factors_meme.txt ${name}_composite_fimo.txt

echo 'DONE'

```

## 14.3 Preparing bed and bigWig files

github raw

```
#!/bin/bash

module load gcc/9.2.0 bedtools/2.29.2

cd /scratch/bhn9by/ATAC/fimo_composites

for i in *_2M.txt
do
    name=$(echo $i | awk -F"/" '{print $NF}' | awk -F"_2M.txt" '{print $1}')
    echo $name
```

```

echo $name
intersectBed -loj -a ../dynamic_peaks.bed -b $i > ${name}_fimo.bed
intersectBed -loj -a ../nondynamic_peaks.bed -b $i > ${name}_fimo_nondyn.bed
intersectBed -loj -a ../all_peaks.bed -b $i > ${name}_fimo_all.bed
cat $i | cut -f1-3,5 | sort -k1,1 -k2,2n > ${name}_2M.bed
done

mv PSWM_family_1_fimo_all.bed AP1_fimo_all.bed
mv PSWM_family_3_fimo_all.bed GR_fimo_all.bed
mv PSWM_family_5_fimo_all.bed CEBP_fimo_all.bed
mv PSWM_family_18_fimo_all.bed TWIST_fimo_all.bed
rm PSWM*all.bed

#transfer bed files for AP1, GR, CEBP, and TWIST into main figures directory
#check that family number matches up to corresponding motif
mkdir main_figure_beds
cp PSWM_family_1_fimo.bed main_figure_beds/AP1_fimo.bed
cp PSWM_family_3_fimo.bed main_figure_beds/GR_fimo.bed
cp PSWM_family_5_fimo.bed main_figure_beds/CEBP_fimo.bed
cp PSWM_family_18_fimo.bed main_figure_beds/TWIST_fimo.bed

cp PSWM_family_1_2M.bed main_figure_beds/AP1_2M.bed
cp PSWM_family_3_2M.bed main_figure_beds/GR_2M.bed
cp PSWM_family_5_2M.bed main_figure_beds/CEBP_2M.bed
cp PSWM_family_18_2M.bed main_figure_beds/TWIST_2M.bed

#prepare bigWigs for motif enrichment plot
cd /scratch/bhn9by/ATAC/fimo_composites/main_figure_beds

cat SP_unsorted.bed | sort -k1,1 -k2,2n > SP_2M.bed
cat KLF_unsorted.bed | sort -k1,1 -k2,2n > KLF_2M.bed

module load ucsc-tools/3.7.4 gcc/9.2.0 bedtools/2.29.2

intersectBed -loj -a ../../all_peaks.bed -b SP_2M.bed > ../../SP_fimo_all.bed
intersectBed -loj -a ../../all_peaks.bed -b KLF_2M.bed > ../../KLF_fimo_all.bed

intersectBed -loj -a ../../dynamic_peaks.bed -b SP_2M.bed > SP_fimo.bed
intersectBed -loj -a ../../dynamic_peaks.bed -b KLF_2M.bed > KLF_fimo.bed

for bed in *2M.bed
do
    name=$(echo $bed | awk -F"/" '{print $NF}' | awk -F"_2M.bed" '{print $1}')
    echo $name
    #summing scores of motifs w/in peak
    cat $bed | mergeBed -i stdin -c 4 -o sum > ${name}_merged_2M.bed
    bedGraphToBigWig ${name}_merged_2M.bed ../../mm10.chrom.sizes ${name}_mm10_instances.bigWig
    rm ${name}_merged_2M.bed
done

```

```
#extract up / down composite motifs from combined SP.KLF
cd /scratch/bhn9by/Adipogenesis/ATAC/SP_KLF_split

module load gcc/7.1.0 openmpi/3.1.4 R/4.0.0

Rscript ../extract.motifs.from.combined.family.R

module load gcc/7.1.0 meme/4.10.2

ceqlogo -i SP.KLF_activated.txt -m SP.KLF_activated -o SP.KLF.activated.eps
ceqlogo -i SP.KLF_activated.txt -m SP.KLF_activated -o SP.KLF.activated.rc.eps -r

ceqlogo -i SP.KLF_repressed.txt -m SP.KLF_repressed -o SP.KLF.repressed.eps
ceqlogo -i SP.KLF_repressed.txt -m SP.KLF_repressed -o SP.KLF.repressed.rc.eps -r
```

## 14.4 Split and generate individual composite motifs for SP and KLF

The following script is nearly identical to the previous composite steps but require minor name and directory changes.

[github raw](#)

```
cd /scratch/bhn9by/ATAC/

#generate new directory
rm -r SP_KLF_split
mkdir SP_KLF_split
cd SP_KLF_split

echo Generate composite SP and KLF motifs
```

```
#separating SP motifs
mkdir SP
cd SP

module load gcc/9.2.0 mvapich2/2.3.3 meme/5.1.0

motifs=$(grep -E 'SP|Sp' ../../PSWM_family_8.txt)

for motif in $motifs
do
    cp /scratch/bhn9by/ATAC/individual_memes/$motif*meme.txt $PWD
done

max_motif=0
final_query='

#query tomtom for each SP factor against all others
for line in *meme.txt
do
    name=$(echo $line | awk -F"_meme.txt" '{print $1}')
```

```

echo $name
query_factor=$line
mv ${query_factor} ..
rm ref_factors.txt
cat *_meme.txt > ref_factors.txt
mv ../${query_factor} $PWD
#increased e-value threshold to 0.05!
tomtom -no-ssc -oc $name.tomtom_output -verbosity 1 -min-overlap 5 -mi 1 \
-dist pearson -evalue -thresh 0.05 ${query_factor} ref_factors.txt
if [[ $(wc -l < $name.tomtom_output/tomtom.tsv) -ge $max_motif ]]
then
max_motif=$(wc -l < $name.tomtom_output/tomtom.tsv)
final_query=$name
fi
done

echo FINAL_QUERY IS $final_query
wc -l $final_query.tomtom_output/tomtom.tsv
cd $final_query.tomtom_output
python2.7 ../../tomtom_output_to_composite.py -i tomtom.xml
mv tomtom.xml_test_index_pswm.txt ../composite.values.txt
mv tomtom.xml_test_index_rc_offset.txt ../composite.index.txt
cd ..

module load gcc/7.1.0 openmpi/3.1.4 R/4.0.0

#generate composite SP PSWM
Rscript ../../generate_composite_motif.SP.R

cat ../../meme_header.txt SP_composite_PSWM.txt > SP_composite_meme.txt

module load gcc/7.1.0 meme/4.10.2

#generate composite SP logo
ceqlogo -i SP_composite_meme.txt -m Composite -o SP_composite.eps
ceqlogo -i SP_composite_meme.txt -m Composite -o SP_composite.rc.eps -r

cd ..

#separating KLF motifs
mkdir KLF
cd KLF

module load gcc/9.2.0 mvapich2/2.3.3 meme/5.1.0

motifs='grep -E 'KLF|Klf' ../../PSWM_family_8.txt'

for motif in $motifs
do
    cp /scratch/bhn9by/ATAC/individual_memes/$motif*meme.txt $PWD
done

```

```

max_motif=0
final_query='

#query tomtom for each KLF factor against all others
for line in *meme.txt
do
    name=$(echo $line | awk -F"_meme.txt" '{print $1}')
    echo $name
    query_factor=$line
    mv ${query_factor} ..
    rm ref_factors.txt
    cat *_meme.txt > ref_factors.txt
    mv ../../${query_factor} $PWD
    #increased e-value threshold to 0.05!
    tomtom -no-ssc -oc $name.tomtom_output -verbosity 1 -min-overlap 5 -mi 1 \
        -dist pearson -evalue 0.05 ${query_factor} ref_factors.txt
    if [[ $(wc -l < $name.tomtom_output/tomtom.tsv) -ge $max_motif ]]
    then
        max_motif=$(wc -l < $name.tomtom_output/tomtom.tsv)
        final_query=$name
    fi
done

echo FINAL_QUERY IS $final_query
wc -l $final_query.tomtom_output/tomtom.tsv
cd $final_query.tomtom_output
python2.7 ../../tomtom_output_to_composite.py -i tomtom.xml
mv tomtom.xml_test_index_pswm.txt ../../composite.values.txt
mv tomtom.xml_test_index_rc_offset.txt ../../composite.index.txt
cd ..

#generate composite KLF PSWM
module load gcc/7.1.0 openmpi/3.1.4 R/4.0.0

Rscript ../../generate_composite_motif.KLF.R

cat ../../meme_header.txt KLF_composite_PSWM.txt > KLF_composite_meme.txt

module load gcc/7.1.0 meme/4.10.2

#generate composite KLF logo
ceqlogo -i KLF_composite_meme.txt -m Composite -o KLF_composite.eps
ceqlogo -i KLF_composite_meme.txt -m Composite -o KLF_composite.rc.eps -r

cd ..

```

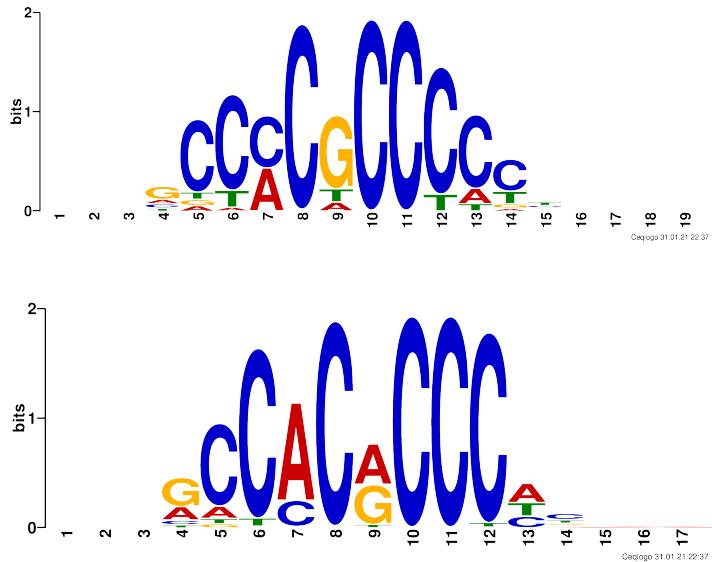


Figure 9: Verify successful splitting of SP(top) and KLF(bottom).

## 14.5 Processing FIMO for SP and KLF

```
github raw
cd /scratch/bhn9by/ATAC/SP_KLF_split

module load gcc/7.1.0 openmpi/3.1.4 R/4.0.0

echo Starting prep R script
Rscript ./prep.SP.KLF.fimo.R

module load gcc/9.2.0 mvapich2/2.3.3 meme/5.1.0

echo Starting SP FIMO
fimo --thresh 0.01 --text SP/SP_composite_meme.txt sp_fimo.txt > output_sp1.txt
echo Starting KLF FIMO
fimo --thresh 0.01 --text KLF/KLF_composite_meme.txt sp_fimo.txt > output_klf.txt

module load gcc/7.1.0 bedtools/2.26.0
cp /scratch/bhn9by/ATAC/fimo_composites/PSWM_family_8_2M.bed $PWD/sp_klf_2M.bed
bedtools getfasta -fi /project/genomes/Mus_musculus/UCSC/mm10/Sequence/WholeGenomeFasta/genome.fa \
-bed sp_klf_2M.bed > sp_klf_2M.fasta

module load gcc/9.2.0 mvapich2/2.3.3 meme/5.1.0

fimo --thresh 0.01 --text SP/SP_composite_meme.txt sp_klf_2M.txt > output_sp1_2M.txt
fimo --thresh 0.01 --text KLF/KLF_composite_meme.txt sp_klf_2M.txt > output_klf_2M.txt

echo Starting split R script
module load gcc/7.1.0 openmpi/3.1.4 R/4.0.0
```

```
Rscript ./SP_KLF_split.R
```

```
echo DONE
```

## 15 Figure 1

The following Rscripts generate plots showing that distinct transcription factors drive opening and closing of chromatin in early adipogenesis.

[github raw](#)

### 15.1 Generate ‘fimo.scores.atac.Rdata’ object

```
library(ggplot2)
library(lattice)
library(pheatmap)
library(fabricatr)
library(ggseqlogo)

load('/scratch/bhn9by/ATAC/plot.df.atac.Rdata')

dir = '/scratch/bhn9by/ATAC/fimo_composites/'
setwd(dir)

#generate 'fimo.scores.atac.Rdata' object
res = unique(read.table('/scratch/bhn9by/ATAC/dynamic_peaks.bed'))
colnames(res) = c('chr', 'start', 'end')
res$start = as.numeric(as.character(res$start))
res$end = as.numeric(as.character(res$end))
rownames(res) = paste0(res[,1], ':', res[,2], '-', res[,3])

#main figure families

for(bed.file in Sys.glob(file.path(paste0(dir,'main_figure_beds/*_fimo.bed')))) {

  factor.name = strsplit(bed.file, "/")[[1]]
  factor.name = strsplit(factor.name[length(factor.name)],
                        '_fimo.bed')[[1]][1]
  print(factor.name)
  x = read.table(bed.file, stringsAsFactors=FALSE)
  x = x[x[,6] != -1,]
  y = aggregate(as.numeric(V8)~V1+V2+V3, data=x, FUN=sum)
  colnames(y) = c('chr', 'start', 'end', factor.name)
  rownames(y) = paste0(y[,1], ':', y[,2], '-', y[,3])

  func <- function(peak) {
    if(peak %in% rownames(y)) {
      return(y[rownames(y) == peak,ncol(y)])
    } else {
```

```

        return(NA)
    }
}

res[,ncol(res)+1] = sapply(rownames(res),func)
colnames(res)[ncol(res)] = factor.name

}

res = res[,-c(1:3)]

#add in SP and KLF after separation
load('/scratch/bhn9by/ATAC/SP_KLF_split/sp.klf.scores.atac.Rdata')

x = sp.klf.scores.atac[,4:5]
colnames(x) = c('SP','KLF')

res = merge(res,x,by='row.names',all=TRUE)
rownames(res) = res[,1]
res = res[,-1]

fimo.scores.atac = res
save(fimo.scores.atac, file = '/scratch/bhn9by/ATAC/fimo.scores.atac.Rdata')

#save as bed files
for(i in 1:ncol(fimo.scores.atac)) {
  temp = fimo.scores.atac[!is.na(fimo.scores.atac[,i]),]
  chr = sapply(strsplit(rownames(temp), ':'), '[', 1)
  x = sapply(strsplit(rownames(temp), ':'), '[', 2)
  start = sapply(strsplit(x, '-'), '[', 1)
  end = sapply(strsplit(x, '-'), '[', 2)
  bed = data.frame(chr, start, end, score = temp[,i])
  write.table(bed, file = paste0(colnames(fimo.scores.atac)[i], '_instances.bed'),
              row.names=F,col.names=F,quote=F,sep='\t')
}

```

## 15.2 Plot all family peaks

```

plot.df = data.frame()
for(i in 1:ncol(fimo.scores.atac)) {
  temp = plot.df.atac[plot.df.atac$genes %in% rownames(fimo.scores.atac[!is.na(fimo.scores.atac[,i])]),]
  temp$family = colnames(fimo.scores.atac)[i]
  plot.df = rbind(plot.df,temp)
}

plot.df$genes = as.factor(plot.df$genes)
cat.colours = plot.df[plot.df$merge == 'one_group0', c(1,10)]
cat.colours$genes <- as.factor(cat.colours$genes)
cat.colours$supercluster <- as.factor(cat.colours$supercluster)

```

```

cat.colours$colour[cat.colours$supercluster == 'up.flat'] <- '#FF000008'
cat.colours$colour[cat.colours$supercluster == 'up.down'] <- '#FF000008'
cat.colours$colour[cat.colours$supercluster == 'gradual.up'] <- '#FF000008'
cat.colours$colour[cat.colours$supercluster == 'gradual.down'] <- '#0000FF08'
cat.colours$colour[cat.colours$supercluster == 'down.up'] <- '#0000FF08'

cat.colours$colour <- as.factor(cat.colours$colour)

cat.colours <- cat.colours[match(levels(plot.df$genes), cat.colours$genes), ]

pdf(file = 'all_families_dynamic_accessibility.pdf', width=14, height=14)

trellis.par.set(box.umbrella = list(lty = 1, col="black", lwd=1),
                box.rectangle = list( lwd=1.0, col="black", alpha = 1.0),
                plot.symbol = list(col="black", lwd=1.0, pch = '.'))

print(
  xyplot(value ~ sample.conditions | family, group = genes, data = plot.df,
         type = c('l'),#type = c('l','p'),
         scales=list(x=list(cex=1.0,relation = "free", rot = 45), y =list(cex=1.0, relation="free")),
         aspect=1.0,
         layout = c(3,2),
         between=list(y=0.5, x=0.5),
         index.cond=list(c(4:6,
                           1:3)),
         main = list(label = 'All Family Motifs in Dynamic Peaks', cex = 1.5),
         ylab = list(label = 'Normalized ATAC signal', cex =1.0),
         xlab = list(label = 'Time (minutes)', cex =1.0),
         par.settings = list(superpose.symbol = list(pch = c(16),col=c('grey20'), cex =0.5),
                             strip.background=list(col="grey80"),
                             superpose.line = list(col = as.character(cat.colours$colour),
                                                   lwd=c(1),lty = c(1))),
         panel = function(x, y, ...) {
           panel.xyplot(x, y, ...)
           #panel.bwplot(x, y, pch = '|', horizontal = FALSE, box.width = 15, do.out = FALSE)
           #panel.spline(x, y, col ='blue', lwd =2.0, ...)
         })
)
dev.off()

```

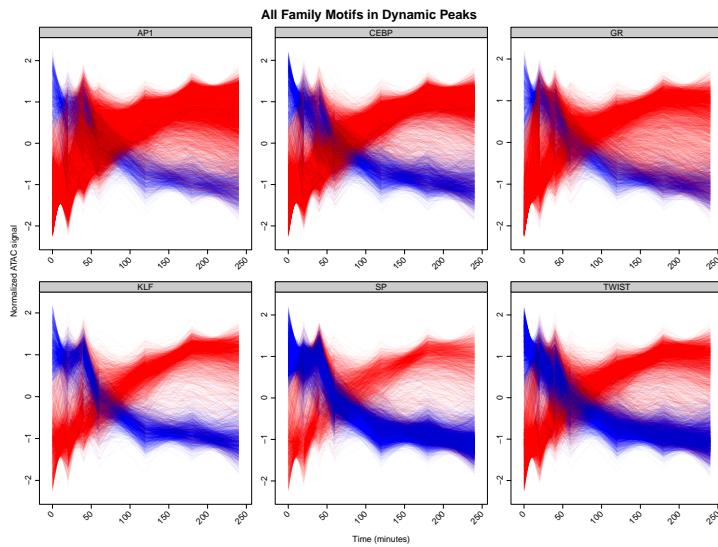


Figure 10: PLACEHOLDER

### 15.3 Plot all families bar plot, including no family category

```

plot.df = data.frame()
for(i in 1:ncol(fimo.scores.atac)) {
  temp = plot.df.atac[plot.df.atac$genes %in% rownames(fimo.scores.atac)!is.na(fimo.scores.atac[,i]),]
  temp$family = colnames(fimo.scores.atac)[i]
  plot.df = rbind(plot.df,temp)
}

plot.df$genes = as.factor(plot.df$genes)
no.fam = rownames(fimo.scores.atac[which(rowSums(is.na(fimo.scores.atac)) == ncol(fimo.scores.atac)),])
temp = plot.df.atac[plot.df.atac$genes %in% no.fam,]
temp$family = 'No Family'
plot.df = rbind(plot.df,temp)

plot.df$status = 'Activated'
plot.df[plot.df$supercluster == 'gradual.down' | plot.df$supercluster == 'down.up',]$status = 'Repressed'
activated = table(plot.df[plot.df$status == 'Activated',]$family) / 7
repressed = table(plot.df[plot.df$status == 'Repressed',]$family) / 7
df = data.frame(names = c(names(activated),names(repressed)),
                 num = c(as.vector(activated),as.vector(repressed)),
                 cond = c(rep('Activated',length(activated)),rep('Repressed',length(repressed))),
                 index = rep(as.vector(table(plot.df$family)),2))
df[df$names == 'No Family',]$index = min(df$index) - 1

```

```

pdf(file='all.peaks.with.no.fam.bar.pdf',width=12,height=7)
print(
  ggplot(df,aes(x = reorder(names,-index),y = num,fill=cond)) +
  geom_bar(stat='identity',position='stack',color='black') +
  #geom_text(aes(label=num),position=position_stack(vjust = 0.5),size=6) +
  labs(title = 'Number of Peaks For Each Motif Family', y = 'Number of Peaks', x = NULL, fill = NULL) -
  theme_minimal() +
  theme(panel.grid.minor = element_blank(),
        axis.ticks = element_blank(),
        axis.text.x = element_text(angle=45,size=20,hjust=.99,vjust=1,color='black',face='bold'),
        axis.text.y = element_text(size=20,face='bold',color='black'),
        axis.title.y = element_text(size=18,face='bold'),
        legend.text = element_text(size=18,face='bold'),
        plot.title = element_text(size=22,face='bold',hjust=0.5)) +
  scale_fill_manual(values = c("indianred1","dodgerblue"))
)
dev.off()

```

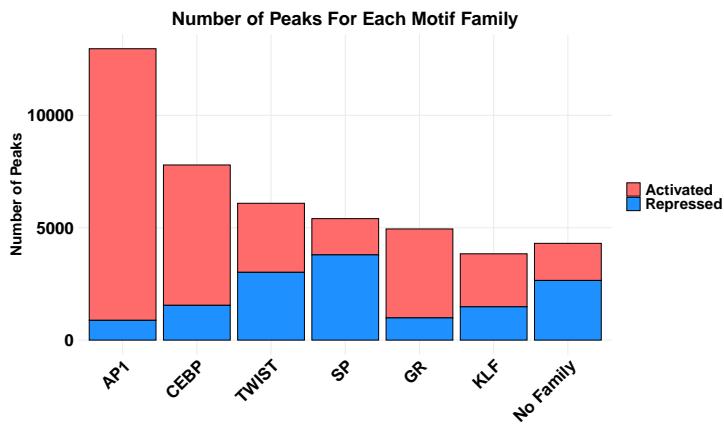


Figure 11: PLACEHOLDER

## 15.4 Plot traces of isolated peaks

```

plot.df = data.frame()

for(i in 1:ncol(fimo.scores.atac)) {
  scores.temp = fimo.scores.atac[!is.na(fimo.scores.atac[,i]),]
  scores.temp = scores.temp[,-i]
  scores.temp = scores.temp[which(rowSums(is.na(scores.temp)) == ncol(scores.temp)),]
  temp = plot.df$genes %in% rownames(scores.temp),
  temp$family = colnames(fimo.scores.atac)[i]
  plot.df = rbind(plot.df,temp)
}

plot.df$genes = as.factor(plot.df$genes)
cat.colours = plot.df[plot.df$merge == 'one_group0', c(1,10)]

```

```

cat.colours$genes <- as.factor(cat.colours$genes)
cat.colours$supercluster <- as.factor(cat.colours$supercluster)

cat.colours$colour[cat.colours$supercluster == 'up.flat'] <- '#FF00001A'
cat.colours$colour[cat.colours$supercluster == 'up.down'] <- '#FF00001A'
cat.colours$colour[cat.colours$supercluster == 'gradual.up'] <- '#FF00001A'
cat.colours$colour[cat.colours$supercluster == 'gradual.down'] <- '#0000FF1A'
cat.colours$colour[cat.colours$supercluster == 'down.up'] <- '#0000FF1A'

cat.colours$colour <- as.factor(cat.colours$colour)

cat.colours <- cat.colours[match(levels(plot.df$genes), cat.colours$genes), ]

pdf(file = 'all_isolated_families_dynamic_accessibility.pdf', width=14, height=14)

trellis.par.set(box.umbrella = list(lty = 1, col="black", lwd=1),
                 box.rectangle = list( lwd=1.0, col="black", alpha = 1.0),
                 plot.symbol = list(col="black", lwd=1.0, pch ='.'))

print(

  xyplot(value ~ sample.conditions | family, group = genes, data = plot.df, type = c('l'),#type = c('l', 'p'),
           scales=list(x=list(cex=1.5,relation = "free", rot = 45), y =list(cex=1.5, relation="free")),
           aspect=1.0,
           layout = c(3,2),
           between=list(y=0.5, x=0.5),
           index.cond=list(c(4:6,
                         1:3)),
           main = list(label = 'All Isolated Family Motifs in Dynamic Peaks', cex = 2.0),
           ylab = list(label = 'Normalized ATAC signal', cex = 2.0),
           xlab = list(label = 'Time (minutes)', cex = 2.0),
           par.settings = list(superpose.symbol = list(pch = c(16),col=c('grey20'), cex = 2.0),
                                 strip.background=list(col="grey80"),
                                 superpose.line = list(col = as.character(cat.colours$colour), lwd=c(1),lty = c(1)))
           panel = function(x, y, ...) {
             panel.xyplot(x, y, ...)
             #panel.bwplot(x, y, pch = '|', horizontal = FALSE, box.width = 15, do.out = FALSE)
             #panel.spline(x, y, col ='blue', lwd =2.0, ...)
           })

)
dev.off()

```

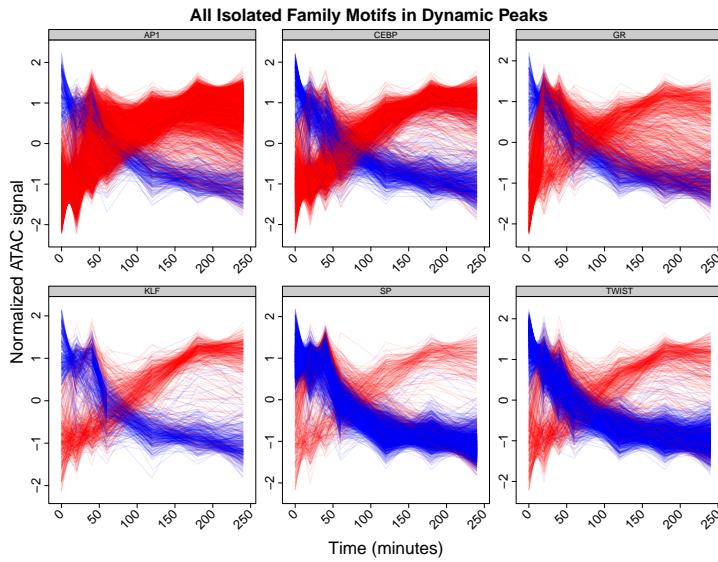


Figure 12: PLACEHOLDER

## 15.5 Plot bar plot of isolated peaks

```

plot.df$status = 'Activated'
plot.df[plot.df$supercluster == 'gradual.down' | plot.df$supercluster == 'down.up',]$status = 'Repressed'
activated = table(plot.df[plot.df$status == 'Activated',]$family) / 7
repressed = table(plot.df[plot.df$status == 'Repressed',]$family) / 7
df = data.frame(names = c(names(activated),names(repressed)),
                 num = c(as.vector(activated),as.vector(repressed)),
                 cond = c(rep('Activated',length(activated)),rep('Repressed',length(repressed))),
                 index = rep(as.vector(table(plot.df$family)),2))

pdf(file='isolated.peaks.bar.pdf',width=12,height=7)
print(
  ggplot(df,aes(x = reorder(names,-index),y = num,fill=cond)) +
  geom_bar(stat='identity',position='stack',color='black') +
  #geom_text(aes(label=num),position=position_stack(vjust = 0.5),size=6) +
  labs(title = 'Number of Isolated Peaks For Each Motif Family',
       y = 'Number of Peaks', x = NULL, fill = NULL) +
  theme_minimal() +
  theme(panel.grid.minor = element_blank(),
        axis.ticks = element_blank(),
        axis.text.x = element_text(angle=45,size=12,hjust=.99,vjust=1,color='black',face='bold'),
        axis.text.y = element_text(size=12,face='bold',color='black'),
        axis.title.y = element_text(size=14,face='bold'),
        legend.text = element_text(size=12,face='bold'))
)

```

```

        plot.title = element_text(size=16,face='bold',hjust=0.5)) +
      scale_fill_manual(values = c("indianred1","dodgerblue"))
    )
dev.off()

```

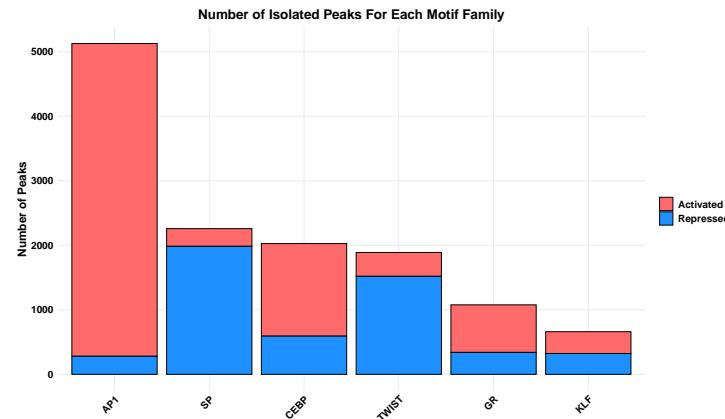


Figure 13: PLACEHOLDER

## 15.6 Plot family heatmaps

```

prop.fam.df = data.frame(matrix(nrow=ncol(fimo.scores.atac),ncol=ncol(fimo.scores.atac)),
                         row.names=colnames(fimo.scores.atac))
total.fam.df = data.frame(matrix(nrow=ncol(fimo.scores.atac),ncol=ncol(fimo.scores.atac)),
                         row.names=colnames(fimo.scores.atac))
colnames(prop.fam.df) = colnames(fimo.scores.atac)
colnames(total.fam.df) = colnames=colnames(fimo.scores.atac)

for(i in 1:ncol(fimo.scores.atac)) {

  x = fimo.scores.atac[!is.na(fimo.scores.atac[,i]),]

  for(j in 1:ncol(x)) {
    z = x[!is.na(x[,j]),]
    total.fam.df[j,i] = nrow(z)
    prop.fam.df[j,i] = 100*(nrow(z)/nrow(fimo.scores.atac[!is.na(fimo.scores.atac[,j])]))
  }
}

rownames(prop.fam.df) = colnames(fimo.scores.atac)
colnames(prop.fam.df) = colnames(fimo.scores.atac)

prop.fam.df = prop.fam.df[order(colnames(prop.fam.df)),]

pdf(file='prop.fam.heatmap.pdf')
pheatmap(prop.fam.df,cluster_rows=FALSE, show_rownames=TRUE,cluster_cols=FALSE,fontsize=12)
dev.off()

```

```
pdf(file='total.fam.heatmap.pdf')
pheatmap(total.fam.df,cluster_rows=FALSE, show_rownames=TRUE,cluster_cols=FALSE,fontsize=12)
dev.off()
```

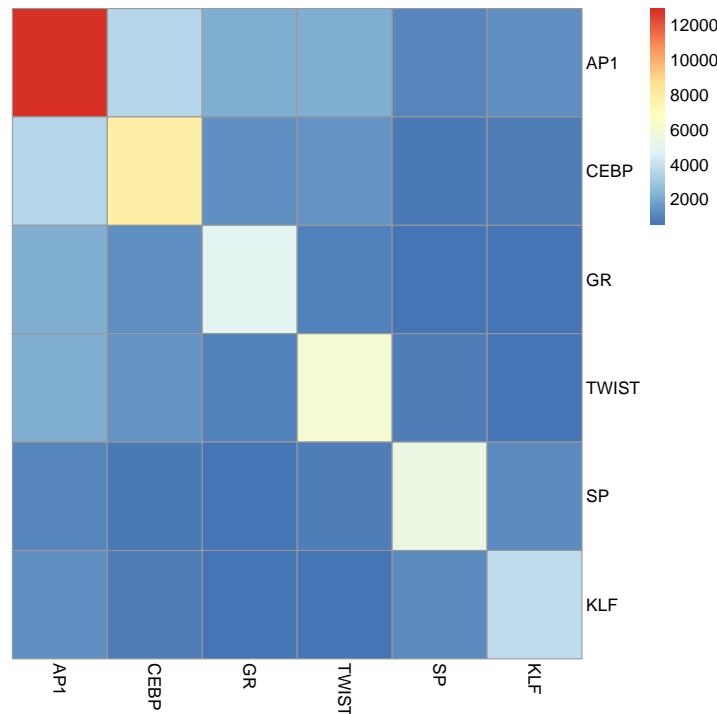


Figure 14: PLACEHOLDER

## 15.7 Plot up/down split occurrences

```
plot.df = plot.df.atac
plot.df$status = 'Activated'
plot.df[plot.df$supercluster == 'gradual.down' | plot.df$supercluster == 'down.up',]$status = 'Repressed'

for(i in 1:ncol(fimo.scores.atac)) {
  up.num = c()
  down.num = c()

  for(j in 1:ncol(fimo.scores.atac)) {

    x = fimo.scores.atac[!is.na(fimo.scores.atac[,j]),]
    y = plot.df[plot.df$genes %in% rownames(x),]
    up = fimo.scores.atac[rownames(fimo.scores.atac) %in% y[y$status == 'Activated',]$genes,]
    down = fimo.scores.atac[rownames(fimo.scores.atac) %in% y[y$status == 'Repressed',]$genes,]

    up.num = append(up.num, (nrow(up[!is.na(up[,i])])/nrow(up))*100)
    down.num = append(down.num, (nrow(down[!is.na(down[,i])])/nrow(down))*100)
  }
}
```

```

}

df = data.frame(names = rep(colnames(fimo.scores.atac),2),
                 #names = rep(c('NRF','TWIST','STAT','TFAP2','ZNF263','AP1','bHLH','GR',
                 #'CTCFL','SP/KLF','ELK/ETV','GFX/ZBTB33','Maz','NFY'),2),
                 num = c(up.num,down.num),
                 cond = c(rep('Activated',length(up.num)),rep('Repressed',length(down.num))))
                 #index = rep(c(10:14,1:9),2))

name = df[i,]$names

df = df[-which(df$names == name),]

df$names = as.factor(df$names)

pdf(file=paste0('percent.',name,'.in.other.families.pdf'),width=8,height=5)
print(
  ggplot(df,aes(x = names,y = num,fill=cond)) +
  geom_bar(stat='identity',position='dodge',color='black') +
  labs(title = paste0('% of ',name,' Occurrences in Other Family Motifs'),
       y = paste0('% of Occurrences with ',name),
       x = 'Motif Family',
       fill = 'Direction') +
  theme_minimal() +
  scale_x_discrete(labels= levels(df$names)) +
  theme(panel.grid.minor = element_blank(),
        plot.title = element_text(hjust = 0.5,face='bold'),
        axis.text.x = element_text(angle=45,size=12,hjust=.99,vjust=1,color='black',face='bold'),
        axis.title.x = element_text(size=14,face='bold'),
        axis.text.y = element_text(size=12,face='bold',color='black'),
        axis.title.y = element_text(size=14,face='bold'),
        legend.title = element_text(size=14,face='bold',color='black'),
        legend.text = element_text(size=12,face='bold',color='black'),
        axis.ticks = element_blank()) +
  scale_fill_manual(values = c("indianred1","dodgerblue"))
)
dev.off()
}

```

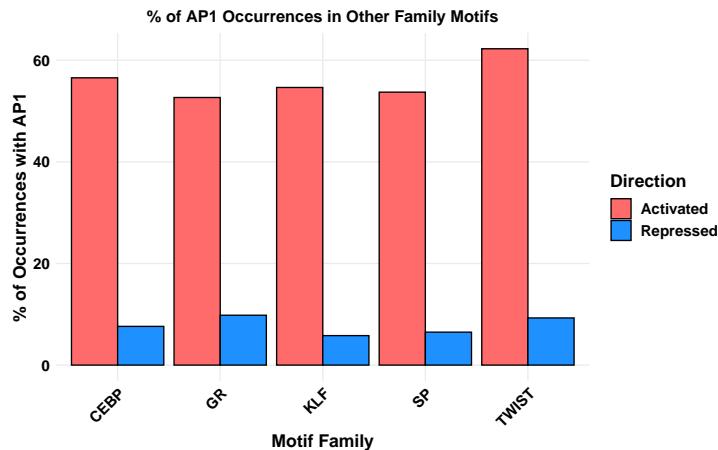


Figure 15: PLACEHOLDER

## 15.8 Plot FIMO scores box and whisker plots for isolated peaks

```

plot.df = data.frame()

for(i in 1:ncol(fimo.scores.atac)) {
  scores.temp = fimo.scores.atac[!is.na(fimo.scores.atac[,i]),]
  scores.temp = scores.temp[, -i]
  scores.temp = scores.temp[which(rowSums(is.na(scores.temp)) == ncol(scores.temp)),]
  temp = plot.df$atac[plot.df$atac$genes %in% rownames(scores.temp),]
  temp$family = colnames(fimo.scores.atac)[i]
  plot.df = rbind(plot.df,temp)
}

plot.df$status = 'Activated'
plot.df[plot.df$supercluster == 'gradual.down' | plot.df$supercluster == 'down.up',]$status = 'Repressed'
plot.df = unique(plot.df[,c(1,11,12)])

func <- function(peak) {
  family = plot.df[plot.df$genes == peak,]$family
  colnum = which(colnames(fimo.scores.atac) == family)
  score = fimo.scores.atac[rownames(fimo.scores.atac) == peak,colnum]
  return(score)
}

plot.df$score = sapply(plot.df$genes,func)

pdf(file = paste0('fimo.scores.isolated.peaks.bw.pdf'),width=12,height=8)

trellis.par.set(box.umbrella = list(lty = 1, col=c("red", "blue"), lwd=2),
                box.rectangle = list( lwd=2.0, col=c("red", "blue"), alpha = 1.0),
                plot.symbol = list(col=c("red", "blue"), lwd=2.0, pch ='.'))

print(

```

## A Model to Predict Drivers of Adipogenesis

```

bwplot(log(as.numeric(as.character(score))), base = 10) ~ status | family, data = plot.df,
horizontal = FALSE, pch = '|', do.out = FALSE,
scales=list(x=list(cex=1.0, relation = "free", rot = 45), y = list(cex=1.0, relation="free")),
aspect=2.0,
between=list(y=0.5, x=0.5),
index.cond=list(c(4:6,1:3)),
ylab = expression('log'[10]* paste(Sigma, '(motif scores)'),),
xlab = expression('ATAC Peak category'),
#manually setting to avoid outlier, since do.out = FALSE
#ylim = list(c(0.9, 1.9), c(0.9, 1.7), c(0.8, 1.8), c(1, 2.15), c(0.96, 1.4)),
par.settings = list(superpose.symbol = list(pch = c(16),col=c('grey20'), cex =0.5),
#I want to change the background strip to the corresponding motif color
strip.background=list(col=c("grey80"))))

)
dev.off()

```

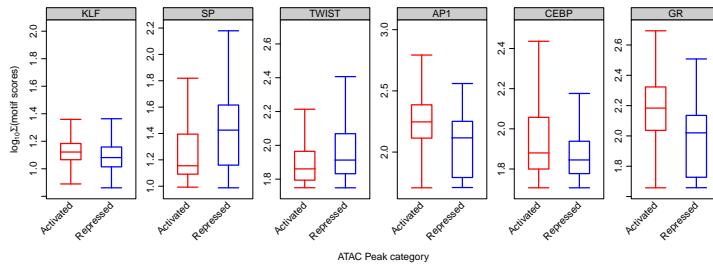


Figure 16: PLACEHOLDER

### 15.9 Plot FIMO scores box and whisker plots for all peaks

```

plot.df = data.frame()
for(i in 1:ncol(fimo.scores.atac)) {
  temp = plot.df$atac[plot.df$atac$genes %in% rownames(fimo.scores.atac)[!is.na(fimo.scores.atac[,i])],]
  temp$family = colnames(fimo.scores.atac)[i]
  plot.df = rbind(plot.df,temp)
}

plot.df$status = 'Activated'
plot.df[plot.df$supercluster == 'gradual.down' | plot.df$supercluster == 'down.up',]$status = 'Repressed'
plot.df = unique(plot.df[,c(1,11,12)])

func <- function(peak,family) {

```

```

#family = plot.df[plot.df$genes == peak,]$family
colnum = which(colnames(fimo.scores.atac) == family)
score = fimo.scores.atac[rownames(fimo.scores.atac) == peak,colnum]
return(score)
}

plot.df$score = mapply(func,plot.df$genes,family=plot.df$family)

pdf(file = paste0('fimo.scores.all.peaks.bw.pdf'),width=12,height=8)

trellis.par.set(box.umbrella = list(lty = 1, col=c("red", "blue"), lwd=2),
                box.rectangle = list( lwd=2.0, col=c("red", "blue"), alpha = 1.0),
                plot.symbol = list(col=c("red", "blue"), lwd=2.0, pch ='.'))

print(
  bwplot(log(as.numeric(as.character(score))), base = 10) ~ status | family, data = plot.df,
  horizontal = FALSE, pch = '|', do.out = FALSE,
  scales=list(x=list(cex=1.0, relation = "free", rot = 45), y = list(cex=1.0, relation="free")),
  aspect=2.0,
  between=list(y=0.5, x=0.5),
  index.cond=list(c(4:6,1:3)),
  ylab = expression('log'[10]* paste(Sigma,'(motif scores)'), 
  xlab = expression('ATAC Peak category'),
  #manually setting to avoid outlier, since do.out = FALSE
  #ylim = list(c(0.9, 1.9), c(0.9, 1.7), c(0.8, 1.8), c(1, 2.15), c(0.96, 1.4)),
  par.settings = list(superpose.symbol = list(pch = c(16),col=c('grey20'), cex =0.5),
  #I want to change the background strip to the corresponding motif color
  strip.background=list(col=c("grey80"))))

)
dev.off()

```

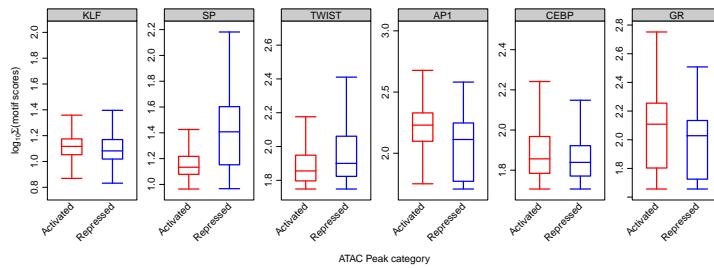


Figure 17: PLACEHOLDER

## 15.10 Plot split on FIMO scores for isolated peaks

```

plot.df = data.frame()

for(i in 1:ncol(fimo.scores.atac)) {
  scores.temp = fimo.scores.atac[!is.na(fimo.scores.atac[,i]),]
  scores.temp = scores.temp[,-i]
  scores.temp = scores.temp[which(rowSums(is.na(scores.temp)) == ncol(scores.temp)),]
  temp = plot.df.atac[plot.df.atac$genes %in% rownames(scores.temp),]
  temp$family = colnames(fimo.scores.atac)[i]
  plot.df = rbind(plot.df,temp)
}

plot.df$status = 'Activated'
plot.df[plot.df$supercluster == 'gradual.down' | plot.df$supercluster == 'down.up',]$status = 'Repressed'

func <- function(peak) {
  family = unique(plot.df[plot.df$genes == peak,]$family)
  colnum = which(colnames(fimo.scores.atac) == family)
  score = fimo.scores.atac[rownames(fimo.scores.atac) == peak,colnum]
  return(score)
}

plot.df$score = sapply(plot.df$genes,func)

for(fam in unique(plot.df$family)) {
  df = plot.df[plot.df$family == fam,]

  fam = gsub('/', '.', fam)
  print(fam)

  df$quantile = paste0('Quantile ',split_quantile(x = as.numeric(df$score),type = 5))

  df$genes = as.factor(df$genes)
  cat.colours = df
  cat.colours$genes <- as.factor(cat.colours$genes)
  cat.colours$status <- as.factor(cat.colours$status)

  cat.colours$colour[cat.colours$status == 'Activated'] <- '#FF000008'
  cat.colours$colour[cat.colours$status == 'Repressed'] <- '#0000FF08'

  cat.colours$colour <- as.factor(cat.colours$colour)

  cat.colours <- cat.colours[match(levels(df$genes), cat.colours$genes),]

  pdf(file = paste0(fam,'_isolated_split_on_FIMO_scores.pdf'),width=11,height=4)

  trellis.par.set(box.umbrella = list(lty = 1, col="black", lwd=1),
                  box.rectangle = list( lwd=1.0, col="black", alpha = 1.0),
                  plot.symbol = list(col="black", lwd=1.0, pch = '.'))

}

```

```

print(
  xyplot(value ~ sample.conditions | quantile, group = genes, data = df,
  type = c('l'),#type = c('l','p'),
  scales=list(x=list(cex=1.0,relation = "free", rot = 45), y = list(cex=1.0, relation="free")),
  aspect=1.0,
  layout = c(5,1),
  between=list(y=0.5, x=0.5),
  main = list(label = paste0(fam, ' Isolated Motifs Stratified By FIMO Score'), cex = 1.5),
  ylab = list(label = 'Normalized ATAC signal', cex = 1.0),
  xlab = list(label = 'Time (minutes)', cex = 1.0),
  par.settings = list(superpose.symbol = list(pch = c(16),col=c('grey20'), cex =0.5),
  strip.background=list(col="grey80"),
  superpose.line = list(col = as.character(cat.colours$colour),
  lwd=c(1),lty = c(1))),
  panel = function(x, y, ...) {
    panel.xyplot(x, y, ...)
    panel.bwplot(x, y, pch = '|', horizontal = FALSE, box.width = 15, do.out = FALSE)
    panel.spline(x, y, col = 'grey70', lwd =3.0, ...)
  })
)

dev.off()

}

```

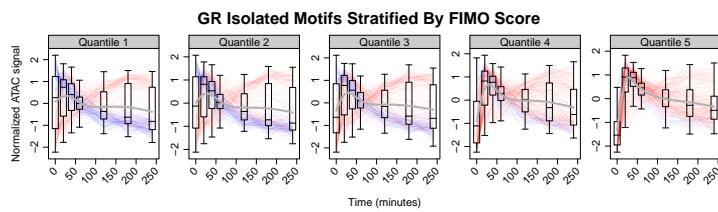


Figure 18: PLACEHOLDER

## 15.11 Plot split on FIMO scores for all peaks

```

plot.df = data.frame()

for(i in 1:ncol(fimo.scores.atac)) {
  temp = plot.df.atac[plot.df.atac$genes %in% rownames(fimo.scores.atac[!is.na(fimo.scores.atac[,i]),]),]
  temp$family = colnames(fimo.scores.atac)[i]
  plot.df = rbind(plot.df,temp)
}

plot.df$status = 'Activated'
plot.df[plot.df$supercluster == 'gradual.down' | plot.df$supercluster == 'down.up',]$status = 'Repressed'

```

```

func <- function(peak,family) {
  #family = plot.df[plot.df$genes == peak,]$family
  colnum = which(colnames(fimo.scores.atac) == family)
  score = fimo.scores.atac[rownames(fimo.scores.atac) == peak,colnum]
  return(score)
}

plot.df$score = mapply(func,plot.df$genes,family=plot.df$family)

for(fam in unique(plot.df$family)) {
  df = plot.df[plot.df$family == fam,]

  print(fam)

  df$quantile = paste0('Quantile ',split_quantile(x = as.numeric(df$score),type = 5))

  df$genes = as.factor(df$genes)
  cat.colours = df
  cat.colours$genes <- as.factor(cat.colours$genes)
  cat.colours$status <- as.factor(cat.colours$status)

  cat.colours$colour[cat.colours$status == 'Activated'] <- '#FF000008'
  cat.colours$colour[cat.colours$status == 'Repressed'] <- '#0000FF08'

  cat.colours$colour <- as.factor(cat.colours$colour)

  cat.colours <- cat.colours[match(levels(df$genes), cat.colours$genes),]

  pdf(file = paste0(fam,'_all_split_on_FIMO_scores.pdf'),width=11,height=4)

  trellis.par.set(box.umbrella = list(lty = 1, col="black", lwd=1),
                  box.rectangle = list( lwd=1.0, col="black", alpha = 1.0),
                  plot.symbol = list(col="black", lwd=1.0, pch = '.'))

  print(
    xyplot(value ~ sample.conditions | quantile, group = genes, data = df,
           type = c('l'),#type = c('l','p'),
           scales=list(x=list(cex=1.0,relation = "free", rot = 45), y = list(cex=1.0, relation="free")),
           aspect=1.0,
           layout = c(5,1),
           between=list(y=0.5, x=0.5),
           main = list(label = paste0(fam, ' All Motifs Stratified By FIMO Score'), cex = 1.5),
           ylab = list(label = 'Normalized ATAC signal', cex =1.0),
           xlab = list(label = 'Time (minutes)', cex =1.0),
           par.settings = list(superpose.symbol = list(pch = c(16),col=c('grey20'), cex =0.5),
                               strip.background=list(col="grey80"),
                               superpose.line = list(col = as.character(cat.colours$colour),
                                                     lwd=c(1),lty = c(1))),
           panel = function(x, y, ...) {
             panel.xyplot(x, y, ...)
```

```

        panel.bwplot(x, y, pch = '|', horizontal = FALSE, box.width = 15, do.out = FALSE)
        panel.spline(x, y, col = 'grey70', lwd = 3.0, ...)

    })
}

dev.off()

}

```

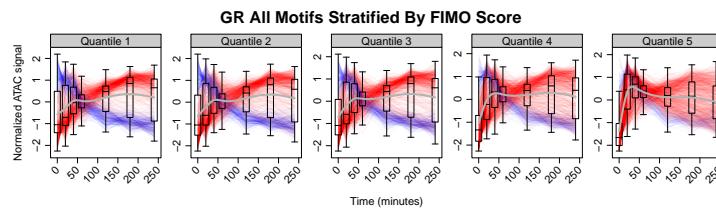


Figure 19: PLACEHOLDER

## 15.12 Generate composites for all activated and repressed peaks

```

pswm.fimo <- function(fimo.in, out = 'outfilename', nm = 'bHLH_Activated', rc = FALSE) {
  posnum = nchar(fimo.in$sequence[1])
  fimo.in$sequence = toupper(fimo.in$sequence)
  col.matrix = matrix()
  for (g in 1:posnum){
    itnum = lapply(strsplit(as.character(fimo.in$sequence), ''), "[", g)
    if (g == 1) {
      col.matrix = itnum
    } else {
      col.matrix = cbind(col.matrix, itnum)
    }
  }
  a.nuc = sapply(1:posnum, function(x) sum(col.matrix[,x] == "A"))
  t.nuc = sapply(1:posnum, function(x) sum(col.matrix[,x] == "T"))
  c.nuc = sapply(1:posnum, function(x) sum(col.matrix[,x] == "C"))
  g.nuc = sapply(1:posnum, function(x) sum(col.matrix[,x] == "G"))

  pswm = cbind(a.nuc, c.nuc, g.nuc, t.nuc)
  print(pswm)
  outfile = file(paste0(out, '.txt'))
  on.exit(close(outfile))
  writeLines(c("MEME version 4", "ALPHABET= ACGT", "strands: + -", " ",
             "Background letter frequencies (from uniform background):",
             "A 0.30000 C 0.20000 G 0.20000 T 0.30000", paste("MOTIF", out), " ",
             paste("letter-probability matrix: alength= 4 w=", posnum)), outfile)
  pswm = pswm/rowSums(pswm)
}

```

```

if (rc == "TRUE"){
  pswm<- pswm[nrow(pswm):1,ncol(pswm):1]
} else {
  write.table(pswm, file = paste0(out, '.txt'), append = TRUE,
              quote=FALSE, row.names =FALSE, col.names = FALSE)
  pswm = t(pswm)
  rownames(pswm) = c('A', 'C', 'G', 'T')
  return(pswm)
}

# system(paste0('/Users/guertinlab/meme/libexec/meme-5.1.1/ceqlogo -i ', out,
#               '.txt -m Composite -o ', nm, '.eps'))
# system(paste0('/Users/guertinlab/meme/libexec/meme-5.1.1/ceqlogo -i ', out,
#               '.txt -m Composite -o ', nm, '.rc.eps -r'))
}

plot.df = plot.df.atac
plot.df$status = 'Activated'
plot.df[plot.df$supercluster == 'gradual.down' | plot.df$supercluster == 'down.up',]$status = 'Repressed'

count = -1
vec.names = c()

df.seq = as.data.frame(matrix(ncol=5, nrow=0),stringsAsFactors = FALSE)
colnames(df.seq) = c('chr', 'start', 'end', 'sequence', 'factor')

for(bed.file in Sys.glob(file.path(paste0(dir,'main_figure_beds/*fimo.bed')))) {
  #print(bed.file)
  count = count + 1
  factor.name = strsplit(bed.file, "/")[[1]]
  factor.name = strsplit(factor.name[length(factor.name)],
                         '_fimo.bed')[[1]][1]
  print(factor.name)
  x = read.table(bed.file, stringsAsFactors=FALSE)
  x = x[x[,6] != -1,]
  x = x[,c(1,2,3,10)]
  x$factor = factor.name
  colnames(x) = c('chr', 'start', 'end', 'sequence', 'factor')
  x$re = paste0(x[,1], ':', x[,2], '-', x[,3])
  df.seq = rbind(df.seq, x)
}

#now add SP and KLF
sp = read.table('/scratch/bhn9by/ATAC/SP_KLF_split/output_sp1.txt', stringsAsFactors=FALSE,
                sep = '\t', header = TRUE)[,c(3,10)]

out.sp <- strsplit(as.character(sp[,1]), ':')
sp <- data.frame(do.call(rbind, out.sp, quote=FALSE), sp[,c(2)])
sp <- sp[,c(1:3,10)]
colnames(sp) <- c('chr','start','end','sequence')
sp$factor = 'SP'
sp$re = paste0(sp$chr,':',sp$start,'-',sp$end)

```

```

df.seq = rbind(df.seq,sp)

klf = read.table('/scratch/bhn9by/ATAC/SP_KLF_split/output_klf.txt', stringsAsFactors=FALSE,
                 sep = '\t', header = TRUE)[,c(3,10)]

out.klf <- strsplit(as.character(klf[,1]), ':')
klf <- data.frame(do.call(rbind, out.klf, quote=FALSE), klf[,c(2)])
klf <- klf[,c(1:3,10)]
colnames(klf) <- c('chr','start','end','sequence')
klf$factor = 'KLF'
klf$re = paste0(klf$chr,':',klf$start,'-',klf$end)

df.seq = rbind(df.seq,klf)

func <- function(peak) {
  return(unique(plot.df[plot.df$genes == peak,]$status))
}

df.seq$status = sapply(df.seq$re,func)

for(i in unique(df.seq$factor)) {
  act = df.seq[df.seq$factor == i & df.seq$status == 'Activated',]
  rep = df.seq[df.seq$factor == i & df.seq$status == 'Repressed',]
  pswm.fimo(act, out = paste0(i,'_activated'))
  pswm.fimo(rep, out = paste0(i,'_repressed'))
}

```

### 15.13 Plot motif enrichment around summits

[github raw](#)

```

library(lattice)
library(bigWig)

dir = '/scratch/bhn9by/ATAC/fimo_composites/'
setwd(dir)

plot.fimo.lattice <- function(dat, fact = 'Motif', summit = 'Hypersensitivity Summit', class= '',
                               num.m = -200, num.p = 90, y.low = 0, y.high = 0.2,
                               col.lines = c(rgb(0,0,1,1/2), rgb(1,0,0,1/2),
                                             rgb(0.1,0.5,0.05,1/2), rgb(0,0,0,1/2),
                                             rgb(1/2,0,1/2,1/2), rgb(0,1/2,1/2,1/2), rgb(1/2,1/2,0,1/2)),
                               fill.poly = c(rgb(0,0,1,1/4), rgb(1,0,0,1/4), rgb(0.1,0.5,0.05,1/4),
                                             rgb(0,0,0,1/4), rgb(1/2,0,1/2,1/4))) {

  pdf('motif_enrichment_around_summits.pdf')#, width=6.83, height=3.5)
  print(xyplot(density ~ range|tf, groups = category, data = dat, type = 'l',
               scales=list(x=list(cex=0.8,relation = "free"),
                           y =list(cex=0.8,axs = 'i',relation = "free")),
               xlim=c(num.m,num.p),

```

```

    col = col.lines,
    auto.key = list(points=F, lines=T, cex=0.8, columns = 2),
    par.settings = list(superpose.symbol = list(pch = c(16), col=col.lines, cex =0.7),
                         superpose.line = list(col = col.lines, lwd=c(2,2,2,2,2),
                                               lty = c(1,1,1,1,1,1,1,1))),
    cex.axis=1.0,
    par.strip.text=list(cex=0.9, font=1, col='black',font=2),
    aspect=1.0,
    between=list(y=0.5, x=0.5),
    index.cond = list(c(4:6,1:3)),
    lwd=2,
    ylab = list(label = "Weighted Motif Density", cex =1,font=2),
    xlab = list(label = 'Distance from ATAC-seq Peak Summit', cex =1,font=2),
    strip = function(..., which.panel, bg) {
      bg.col = 'grey'#c("blue","grey65","red")
      strip.default(..., which.panel = which.panel,
                    bg = rep(bg.col, length = which.panel)[which.panel])
    }
  ))
  dev.off()
}

load('/scratch/bhn9by/ATAC/plot.df.atac.Rdata')
load('/scratch/bhn9by/ATAC/fimo.scores.atac.Rdata')

plot.df = data.frame()
for(i in 1:ncol(fimo.scores.atac)) {
  temp = plot.df.atac[plot.df.atac$genes %in% rownames(fimo.scores.atac,!is.na(fimo.scores.atac[,i])),]
  temp$family = colnames(fimo.scores.atac)[i]
  plot.df = rbind(plot.df,temp)
}

plot.df$status = 'Activated'
plot.df[plot.df$supercluster == 'gradual.down' | plot.df$supercluster == 'down.up',]$status = 'Repressed'

all.fimo = data.frame(matrix(ncol = 4, nrow = 0))
colnames(all.fimo) = c('density', 'tf', 'category', 'range')

half.win = 600
file.suffix = '_mm10_instances.bigWig'
dir = '/scratch/bhn9by/ATAC/fimo_composites/main_figure_beds/'

decreased = plot.df[plot.df$status == 'Repressed',7:9]
decreased[,2] = as.numeric(decreased[,2])
decreased[,3] = as.numeric(decreased[,3])
decreased = bed.window(decreased,half.win)

increased = plot.df[plot.df$status == 'Activated',7:9]
increased[,2] = as.numeric(increased[,2])
increased[,3] = as.numeric(increased[,3])
increased = bed.window(increased,half.win)

```

```

not.different = read.table('/scratch/bhn9by/ATAC/nondynamic_peaks.bed')
not.different = not.different[not.different$V1 != 'chrM',]
not.different = bed.window(not.different,half.win)

all.fimo = data.frame()

for(i in 1:ncol(fimo.scores.atac)) {
  factor = colnames(fimo.scores.atac)[i]

  mod.bigWig = paste0(dir,factor,file.suffix)
  factor.name = factor
  print(factor.name)

  loaded.bw = load.bigWig(mod.bigWig)

  dec.inten = bed.step.probeQuery.bigWig(loaded.bw, decreased,
                                         gap.value = 0, step = 10, as.matrix = TRUE)
  dec.query.df = data.frame(cbind(colMeans(dec.inten), factor.name,
                                   'Closed', seq(-half.win, (half.win-10), 10)), stringsAsFactors=F)
  colnames(dec.query.df) = c('density', 'tf', 'category', 'range')

  inc.inten = bed.step.probeQuery.bigWig(loaded.bw, increased,
                                         gap.value = 0, step = 10, as.matrix = TRUE)
  inc.query.df = data.frame(cbind(colMeans(inc.inten), factor.name,
                                   'Opened', seq(-half.win,(half.win-10), 10)), stringsAsFactors=F)
  colnames(inc.query.df) = c('density', 'tf', 'category', 'range')

  ctrl.inten = bed.step.probeQuery.bigWig(loaded.bw, not.different,
                                         gap.value = 0, step = 10, as.matrix = TRUE)
  ctrl.query.df = data.frame(cbind(colMeans(ctrl.inten), factor.name,
                                   'Nondynamic', seq(-half.win, (half.win-10), 10)), stringsAsFactors=F)
  colnames(ctrl.query.df) = c('density', 'tf', 'category', 'range')

  tf.all = rbind(dec.query.df, inc.query.df, ctrl.query.df)

  all.fimo = rbind(all.fimo,tf.all)
}

all.fimo[,1] = as.numeric(all.fimo[,1])
all.fimo[,4] = as.numeric(all.fimo[,4])

plot.fimo.lattice(all.fimo, num.m = -500, num.p = 500,
                  col.lines = c('blue','grey65','red'))

```

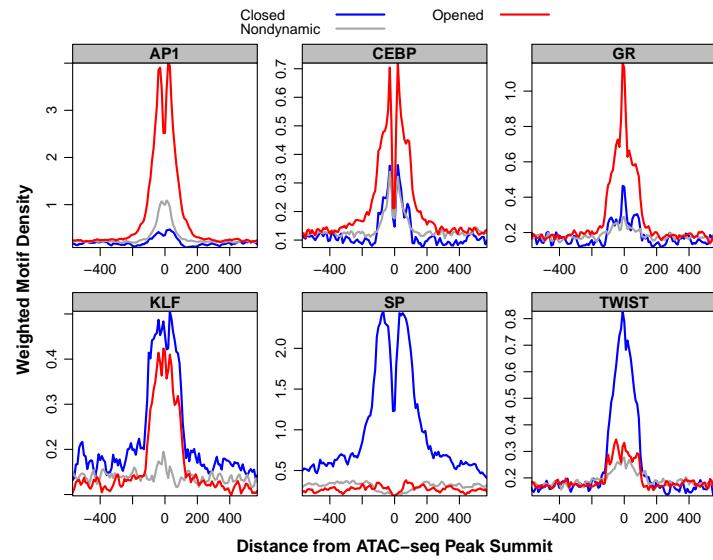


Figure 20: PLACEHOLDER

## 16 Supplemental Figure 1

### 16.1