EpigenomicsTutorial-ISMB2017 Documentation

Release latest

Jul 21, 2017

Contents

If you are taking part in our tutorial at ISMB 2017, then you find the information here that you need. The documentation is split up into 4 parts. Make sure that you read and follow the **What you need** and **Installation** guides before you appear in Prague.

Contents:

CHAPTER 1

What you need

You need to bring your own **laptop** with the following software installed (see detailed instructions below)

- R version 3.2 or higher
- Python version 2.7
- histoneHMM
- HINT
- TEPIC
- IGV
- samtools

Note: that the individual softwares may have some other dependencies, e.g. bedtools which you should have installed.

Course Material on Github

You need to download (or clone) the git repository EpigenomicsTutorial-ISMB2017. This repository contains all data to be used in the course, as well as solutions and scripts for some of the problems.

CHAPTER 2

Installation

The following software packages need to be installed for running the tutorial:

R version 3.2 or higher.

histoneHMM

You might need to install the following dependencies before installing histoneHMM.

Unix libraries:

- lib-gcc
- lib-gsl / lib-gsl-dev
- openssh

R libraries:

- Rcpp
- optparse
- GenomicRanges (bioconductor.org)
- Rsamtools (bioconductor.org)
- mvtnorm

The package was developed and tested using a linux system and R. To install the latest version of the package, open an R terminal and type in the following commands (using the 'devtools' package):

In the R terminal type the following:

```
install.packages("devtools") # if devtools is not yet installed
devtools::install_github("matthiasheinig/histoneHMM")
```

Now the latest version of histoneHMM should be installed on your system. In the tutorial, we will use the command-line interface to histoneHMM. In order for this to work smoothly, it would be best if you add the path to the histoneHMM script files to your $PATH variable (otherwise you'd have to specify the full path each time you call histoneHMM). You can find out the path you need to add by starting an R terminal and typing:

```
system.file("bin/", package="histoneHMM")
```

Which should yield something like this:

```
[1] "/home/[user-path]/R/x86_64-redhat-linux-gnu-library/3.2/histoneHMM/bin/"
```

Now you can add the directory indicated above to your PATH variable by calling:

```
export PATH=$PATH:/home/[user-path]/R/x86_64-redhat-linux-gnu-library/3.2/histoneHMM/
↪bin/
```

If you want to make the histoneHMM command-line available to you everytime you log on to your system, make sure that the directory is added to the $PATH variable everytime you log on or create a new terminal (e.g. by modifying your ~/.bashrc).

HINT

To install HINT (RGT Suite), you are advised to use the Python package installer pip. First, make sure that the python version is 2.7 and download the pip installer get-pip.py and then install pip.

```
python get-pip.py
```

Next, install dependencies:

```
pip install --user cython numpy scipy
pip install --user https://github.com/fabioticconi/MOODS/tarball/pypi-ready
```

and finally install HINT and RGT suite.

```
pip install --user RGT
```

Alternatively, look at detailed instructions here.

You also need to download genome information for mouse genome mm10.

```
cd ~/rgtdata
python setupGenomicData.py --mm10
```

Note: for Mac user, we recommend to first install python and wget by using

```
brew install python
brew install wget
```

IGV

Instructions on installing IGV are available here. We advise you to download a binary distribution.

TEPIC

Dependencies

TEPIC requires:

- bedtools
- A C++ compiler supporting openmp, e.g. g++ (test with version 4.9.2)

To run the machine learning pipeline DYNAMITE, which is part of the TEPIC repository, we require the *R libraries:*

- glmnet
- doMC
- gplots
- ggplot2
- reshape2
- gridExtra

The TEPIC examples in the tutorial also require the mouse reference genome that was downloaded during the HINT setup.

Installation

Start a terminal and clone the TEPIC repository

```
git clone https://github.com/SchulzLab/TEPIC.git
```

Next, go to the folder

```
TEPIC/Code
```

and type

```
bash compile_TRAP_install_R_packages.sh
```

to build the C++ component of TEPIC and install missing R packages.

If all dependencies mentioned above are available, no further installation steps are required.

Testing

To test the core functionality of TEPIC, go to the folder:

```
TEPIC/Code/
```

and run the example with the command::

```
./TEPIC.sh -g ../Test/example_sequence.fa -b ../Test/example_regions.bed -o TEPIC-
↪Example -p ../PWMs/pwm_vertebrates_jaspar_uniprobe_original.PSEM -a ../Test/example_
↪annotation.gtf -w 3000 -e FALSE
```

There should be three result files generated:

- TEPIC-Example <date> Affinity.txt
- TEPIC-Example <date> amd.tsv
- TEPIC-Example <date> Peak_Features_Affinity_Gene_View_Filtered.txt

To test the logistic regression framework DYNAMITE, which will be used in the tutorial, go to the folder

```
/TEPIC/MachineLearningPipelines/DYNAMITE/
```

and run the provided example by entering the command

```
bash runDYNAMITE.sh ./DYNAMITE.cfg
```

This will generate all output files that are described in the DYNAMITE documentation.

For further information, please see the TEPIC repository .

CHAPTER 3

Practical I - Differential Histone peak calling

In the first part of the practical, we will have a look at histone modifications in different cell-types as measured by ChIP-seq experiments in B-cell, CD4-cell and LSK (MPP) cell data from Lara-Astiaso et al 2014. Specifically, we look at the H3K4me3 and H3K27ac histone modifications and will analyze how they change between blood progenitor and more differentiated cells. We will use histoneHMM for calling regions in the genome which show histone modifications as well as for identifying those regions, which show differential modification states between cell-types. The data used in this part of the practical can be found in your checked out tutorial directory under `/EpigenomicsTutorial-ISMB2017/session1/step1/input`

The final version of the practical will be available at 19.07.2017 at the latest.

Step 1: Checking read alignments

Before we look at any modifications patterns in our ChIP-seq experiments, we shall get an impression of how our sequencing data look.

NOTE: In the step1 input directory, we also provide experiment files for the H3k4me1 and H3K4me2 histone marks. Those will not be fully processed using the scripts on this page, but you can look at them if you have any spare time left.

1. First, change into the EpigenomicsTutorial-ISMB2017 directory and see which files are available as an input

```
cd EpigenomicsTutorial-ISMB2017/session1
ls -lh step1/input
```

You can see the *.bam and *.bai files for the three cell-lines and the two examined histone modifications. You also see some *.wig files which we'll use later when looking at our data using IGV, ignore them for now. Now we want to get a brief overview on the nature of the bam files.

2. Create summary statistics using samtools

```
mkdir -p step1/output/stats
for i in step1/input/*.bam ; do
```

```
  samtools flagstat $i > step1/output/stats/$(basename $i .bam).summary ;
done
```

This will create for each of the available *.bam files a short read summary in the step1/output/stats directory. Now check those files, what do you see? Also have a look at the header of the *.bam files, what can you observe?

3. Have a look at the data files using the IGV

Just open the IGV, then via `File->Load from File` open your *.bam file of choice and the corresponding *.wig file (also in the step1/input/ dierctory). Make sure that the correct Mouse genome (mm10) is selected in the upper left view of the browser, since this is the genome build which was used during read mapping. Look at the region around the `ZAP70 gene` (e.g. by using the IGV search bar): Examine the loaded tracks, what do you observe? Are there regions of high/low coverage? (hint: you might want to scale the *.wig tracks to get a nicer view)

Step 2: Calling modified regions

Now we know what we are dealing with and we are ready to begin the process of analyzing our histone marks using histoneHMM. The first step is to identify those regions in the genome which show histone modifications, i.e. to 'call regions'. histoneHMM works by first binning the genome into equally sized, non overlapping bins and then analyzing the number of reads falling into each of those bins. Necessary inputs for this step are 1) a chromosome lengths file, indicating length and name of chromosomes which are available and 2) the *.bam file from the ChIP-seq experiment of interest. Chromosome lengths files can easily be downloaded from 'UCSC goldenpath'. Alternatively, you can extract the information from the *.bam files directly, since it should always be encoded in the header. You can use either of the two following code sections to get your chromosome lengths file.

1.1. Getting chromosome lengths from UCSC

```
wget ftp://hgdownload.cse.ucsc.edu/goldenPath/mm10/database/chromInfo.txt.gz
gunzip chromInfo.txt.gz
# we filter the chr1 only, since we only have chr1 reads
grep -w chr1 chromInfo.txt > chromInfo.chr1.txt
```

1.2. Extracting chromosome lengths from *.bam files

```
samtools view -H step2/input/B_H3k27ac.bam | grep SN:chr1 | cut -f 2,3 | sed s/
↪[SL][NQ]://g > chromInfo.chr1.txt
```

With the chromosome lengths file in place, we now run the command-line version of histoneHMM to call the modified regions. We also use the tool's -b parameter to set the size of the bins in which the genome should be devided to 2000bp.

NOTE: Before going on, make sure that the histoneHMM 'bin' directory is contained in you PATH variable (see installation instructions)

2. Run histoneHMM's 'call_regions'

```
mkdir -p step2/output/regions
wget ftp://hgdownload.cse.ucsc.edu/goldenPath/mm10/database/chromInfo.txt.gz
gunzip chromInfo.txt.gz
# we filter the chr1 only, since we only have chr1 reads
grep -w chr1 chromInfo.txt > chromInfo.chr1.txt
for i in step2/input/*.bam ; do
  prefix=step2/output/regions/$(basename $i .bam)
  histoneHMM_call_regions.R -b 2000 -c chromInfo.chr1.txt -o ${prefix} $i &> ${prefix}
↪.debug
done
```

Now for each experiment, the script generated a set of files. Figure out what the different files are using the histoneHMM manual . histoneHMM fits a mixture model to the counts using an EM algorithm. The two components of the mixture reflect two parts of the histogram: one with very high signal (high counts) and one with low signal values (low counts). Now check the generated count histograms, do you observe the two parts of the mixture fit? How does the count histogram look, would you have expected something like this?

Step 3: Differential region calling

The next and last step in this pipeline is formed by the differential region calling. Here we will compare experiments of the same histone modification in different cell-lines. To perform the differential region calling with histoneHMM, we only need a file with binned count information as is created during the previous step for both experiments we want to compare.

NOTE: If you want you can redirect all output of histoneHMM using the '&>' operator as we did in the previous step.

1. Call differential regions

```
odir=step3/output/differential
mkdir -p ${odir}
idir=step3/input/regions/

# call differential analysis for all possible comparisons
# for H3K4me3
histoneHMM_call_differential.R --sample1 LSK_H3K4me3 --sample2 CD4_H3K4me3 --outdir $
↪{odir} ${idir}/LSK_H3K4me3.txt ${idir}/CD4_H3K4me3.txt
histoneHMM_call_differential.R --sample1 CD4_H3K4me3 --sample2 B_H3K4me3 --outdir $
↪{odir} ${idir}/CD4_H3K4me3.txt ${idir}/B_H3K4me3.txt
histoneHMM_call_differential.R --sample1 LSK_H3K4me3 --sample2 B_H3K4me3 --outdir $
↪{odir} ${idir}/LSK_H3K4me3.txt ${idir}/B_H3K4me3.txt

# for H3K27ac
histoneHMM_call_differential.R --sample1 LSK_H3K27ac --sample2 CD4_H3K27ac --outdir $
↪{odir} ${idir}/LSK_H3K27ac.txt ${idir}/CD4_H3K27ac.txt
histoneHMM_call_differential.R --sample1 CD4_H3K27ac --sample2 B_H3K27ac --outdir $
↪{odir} ${idir}/CD4_H3K27ac.txt ${idir}/B_H3K27ac.txt
histoneHMM_call_differential.R --sample1 LSK_H3K27ac --sample2 B_H3K27ac --outdir $
↪{odir} ${idir}/LSK_H3K27ac.txt ${idir}/B_H3K27ac.txt
```

histoneHMM again creates several output files (check the manual do get to know those files). The infividual *.gff files contain the regions which are modified in both, none or only one of the compared experiments. For further analysis, we will only consider those regions which show an average posterior probability of at least 0.8. Also we want to make the *.gff files somewhat more convenient to deal with and convert them into *.bed files. You can do this however you want, here we will use a straight forward method using only Unix commands.

2. Filter and convert differential calls

```
for i in step3/output/differential/*.gff ; do
  ofile=$(dirname $i)/$(basename $i .gff).post_08.bed
  awk '{split($9,arr,";"); split(arr[1],arr2,"="); }{if(arr2[2]>=0.8) print $1 "\t"
↪$4-1 "\t" $5}' ${i} > ${ofile}
done
```

The new *.bed files (with the .post_08 suffix) now contain the coordinates of the differential and modified/not modified regions for the analyzed experiment. To further get to know the results, check how many differential regions were discovered for each comparison after filtering. How many regions do you observe? Do the numbers differ between the individual histone marks? As a last step, open again IGV and load the *.bam files as before. But now also add a

few of the filtered *.bed files to add tracks which show e.g. the location of the differential peaks. Now again load the region around the `ZAP70 gene`. Can you visually discern the differential peaks in the *.bam tracks? Do you agree with the results from histoneHMM?

CHAPTER 4

Practical II - Footprint calling & Transcription factor prediction

In the second practical, we will perform a footprint analysis with HINT to identify cell specific binding sites from open chromatin data (ATAC-seq). Next, we will combine the footprints with the differential histone peaks detected by histoneHMM (c.f. practical 1). Thereby, we will find tissue specific TF binding sites, which are located in regions with cell specific histone peaks. These regulatory regions are used in a DYNAMITE analysis with the aim of inferring TFs that might be related to gene expression differences between the tissues of interest.

The final version of the practical will be available at 19.07.2017 at the latest.

Step1: Footprint calling

First, we will use HINT to find genomic regions (footprints) with cell specific TF binding sites. For this, HINT requires (1) a sorted bam file containing the aligned reads from the sequencing library (DNase-,ATAC- or histone ChIP-seq) (2) and a bed file including peaks detected in the same sequencing library provided in (1). These peak regions are used by HINT to reduce the search space and can be generated by any peak caller.

Here, we will analyze ATAC-seq data from LSK cells (equivalent to MPP cells), B cells and T CD4 cells obtained from Lara-Astiaso et al 2014. We have performed low level analysis steps including read alignment and peaks calling in chromosome 1, which can be found in this folder `/EpigenomicsTutorial-ISMB2017/session2/step1/input`. Check here for a script showing commands used to generate these files for B cell ATAC-seq experiments.

1. First, go to the EpigenomicsTutorial-ISMB2017 directory and generate an output folder for the result files:

```
cd EpigenomicsTutorial-ISMB2017
mkdir session2/step1/output
```

2. Execute the following commands to call footprints on B, LSK and CD4 cells for chromosome 1:

```
rgt-hint --atac-footprints --organism=mm10 --output-location=session2/step1/output/ --
↪output-prefix=B_ATAC_chr1_footprints session2/step1/input/B_ATAC_chr1.bam session2/
↪step1/input/B_ATACPeaks_chr1.bed
rgt-hint --atac-footprints --organism=mm10 --output-location=session2/step1/output/ --
↪output-prefix=LSK_ATAC_chr1_footprints session2/step1/input/LSK_ATAC_chr1.bam␣
↪session2/step1/input/LSK_ATACPeaks_chr1.bed
```

```
rgt-hint --atac-footprints --organism=mm10 --output-location=session2/step1/output/ --
↪output-prefix=CD4_ATAC_chr1_footprints session2/step1/input/CD4_ATAC_chr1.bam␣
↪session2/step1/input/CD4_ATACPeaks_chr1.bed
```

This will generate an output file, i.e `session2/step1/output/B_ATAC_chr1_footprints.bed`, containing the genomic locations of the footprints. HINT also produces a file with ending ".info", which has general statistics from the analysis as no. of footprints, total number of reads and so on.

We can use IGV to vizualise ATAC-seq signals and footprint predictions in particular loci. First, we can use a special HINT command to generate genomic profiles (bigWig files).

```
rgt-hint --print-signal --bc-signal --bigWig --organism=mm10 --reads-file=./session2/
↪step1/input/B_ATAC_chr1.bam --regions-file=./session2/step1/input/B_ATACPeaks_chr1.
↪bed --output-location=./session2/step1/output --output-prefix=B_ATAC_chr1
rgt-hint --print-signal --bc-signal --bigWig --organism=mm10 --reads-file=./session2/
↪step1/input/CD4_ATAC_chr1.bam --regions-file=./session2/step1/input/CD4_ATACPeaks_
↪chr1.bed --output-location=./session2/step1/output --output-prefix=CD4_ATAC_chr1
rgt-hint --print-signal --bc-signal --bigWig --organism=mm10 --reads-file=./session2/
↪step1/input/LSK_ATAC_chr1.bam --regions-file=./session2/step1/input/LSK_ATACPeaks_
↪chr1.bed --output-location=./session2/step1/output --output-prefix=LSK_ATAC_chr1
```

This bigwig file contains number of ATAC-seq (or DNAse-seq) reads at each genomic position as estimated by HINT after signal normalization and cleveage bias correction. This is therefore more accurate than simply looking a coverage profiles of a bam file.

Open all bigwig and footprint files (bed) generated above in IGV. Remember to set the genome version to mm10 beforehand. You can also enrich the data by opening bam files of histone modifications as H3K4me3, H3K4me1 and H3K27ac (provided in session 1). Check for example the genomic profiles around the gene Zp70, which is part of T cell receptor. We observe that this gene has several open chromatin regions and high levels of histone levels in the start of the gene, which are specific to CD4 T cells. There is also some open chromatin level around exon 3, which is supported by H3K4me1 modifications. This potential enhancer region is also present in distinct degrees in B and LSK cells.

3. An important question when doing footprint analysis is to evaluate which TF motifs overlap with footprints and evaluate the ATAC-seq profiles around these motifs. RGT suite also offers a tool for finding motif predicted binding sites (mpbs). For example, we analyze here motifs from factors SPI1 and ELK4, which were discussed in Lara-Astiaso et al. 2014 to be associated respectively associated to LSK and CD4 cells.

Execute the following commands to do motif matching inside footprints for chromosome 1:

```
rgt-motifanalysis --matching --organism=mm10 --output-location=session2/step1/output/␣
↪--use-only-motifs=session2/step1/input/motifs.txt session2/step1/output/B_ATAC_chr1_
↪footprints.bed
rgt-motifanalysis --matching --organism=mm10 --output-location=session2/step1/output/␣
↪--use-only-motifs=session2/step1/input/motifs.txt session2/step1/output/CD4_ATAC_
↪chr1_footprints.bed
rgt-motifanalysis --matching --organism=mm10 --output-location=session2/step1/output/␣
↪--use-only-motifs=session2/step1/input/motifs.txt session2/step1/output/LSK_ATAC_
↪chr1_footprints.bed
```

The file `session2/step1/motifs.txt` contains a list of JASPAR motif ids to be used in the analysis. Ignoring this option will search for all JASPAR motifs. The above commands will generate bed files (i.e. LSK_ATAC_footprints_mpbs.bed) containing mpbs overlapping with distinct footprint regions. The 4th column contains the motif name and the 5th column the bitscore of the motif match. If you open these files in IGV, you will observe that a ELK4 binding site near the 3rd exon of the gene Zp70.

4. Finally, we use HINT to generate average ATAC-seq profiles around binding sites of particular TF. This analysis allow us to inspect the chromatin chromatin accessibility and the underlying sequence. Moreover, by comparing the

cut profiles from two ATAC-seq libraries (i.s. LSK vs T CD4 cells), we can get insights on changes in binding in two cells. For this, execute the following commands:

```
mkdir session2/step1/output/LSK_B
rgt-hint --diff-footprints --organism=mm10 --mpbs-file=session2/step1/result/LSK_B_
↪ATAC_footprints_mpbs.bed --reads-file1=session2/step1/input/LSK_ATAC.bam --reads-
↪file2=session2/step1/input/B_ATAC.bam --output-location=session2/step1/output/LSK_B␣
↪--output-prefix=LSK_B

mkdir session2/step1/output/LSK_CD4
rgt-hint --diff-footprints --organism=mm10 --mpbs-file=session2/step1/result/LSK_CD4_
↪ATAC_footprints_mpbs.bed --reads-file1=session2/step1/input/LSK_ATAC.bam --reads-
↪file2=session2/step1/input/CD4_ATAC.bam --output-location=session2/step1/output/LSK_
↪CD4 --output-prefix=LSK_CD4
```

The above commands will generate eps files with a ATAC-seq profile for each of the motifs founds in the provided mpbs bed files. Let's check the profiles in the comparison LSK and CD4, you will see that ELK4 has higher number of ATAC-seq counts in CD4 cells, while SPI1 has more ATAC-seq in LSK cells. Higher ATAC counts indicates higher activity of the factor in that particular cell. This fits with the results discussed in Lara-Astiaso that SPI1 are more relevant/active in LSK, while ELK4 in CD4 cells.

Step2: Intersecting footprints with differential histone peaks

To derive candidate regions for TF binding, we combine (1) genome wide footprint calls and (2) genome wide differential histone peak calls using the active chromatin marks H3K4me3 and H3K27ac. In addition to default unix functions we use *bedtools* to combine the respective bed files.

All input files are available in the folder `/EpigenomicsTutorial-ISMB2017/session2/step2/input`.

1. Assure that you are in the directory `EpigenomicsTutorial-ISMB2017/session2/step2`, otherwise *cd* to that directory.

2. Generate an output folder for the resulting bed files and **enter the folder**:

```
mkdir output
cd output
```

3. Combine the Differential peak calls for H3K4me3 and H3K27ac in one, sorted bed file. This needs to be done for each pairwise comparison and each cell type individually:

```
cat ../input/Dif_Histone_Peaks/B_H3K27ac-vs-CD4_H3K27ac-B.bed ../input/Dif_Histone_
↪Peaks/B_H3K4me3-vs-CD4_H3K4me3-B.bed | sort -k1,1 -k2,2n > B_vs_CD4_H3K27ac_H3K4me3_
↪B_sorted.bed
cat ../input/Dif_Histone_Peaks/B_H3K27ac-vs-CD4_H3K27ac-CD4.bed ../input/Dif_Histone_
↪Peaks/B_H3K4me3-vs-CD4_H3K4me3-CD4.bed | sort -k1,1 -k2,2n > B_vs_CD4_H3K27ac_
↪H3K4me3_CD4_sorted.bed

cat ../input/Dif_Histone_Peaks/LSK_H3K27ac-vs-B_H3K27ac-LSK.bed ../input/Dif_Histone_
↪Peaks/LSK_H3K4me3-vs-B_H3K4me3-LSK.bed | sort -k1,1 -k2,2n > LSK_vs_B_H3K27ac_
↪H3K4me3_LSK_sorted.bed
cat ../input/Dif_Histone_Peaks/LSK_H3K27ac-vs-B_H3K27ac-B.bed ../input/Dif_Histone_
↪Peaks/LSK_H3K4me3-vs-B_H3K4me3-B.bed | sort -k1,1 -k2,2n > LSK_vs_B_H3K27ac_H3K4me3_
↪B_sorted.bed

cat ../input/Dif_Histone_Peaks/LSK_H3K27ac-vs-CD4_H3K27ac-LSK.bed ../input/Dif_
↪Histone_Peaks/LSK_H3K4me3-vs-CD4_H3K4me3-LSK.bed | sort -k1,1 -k2,2n > LSK_vs_CD4_
↪H3K27ac_H3K4me3_LSK_sorted.bed
```

```
cat ../input/Dif_Histone_Peaks/LSK_H3K27ac-vs-CD4_H3K27ac-CD4.bed ../input/Dif_
↪Histone_Peaks/LSK_H3K4me3-vs-CD4_H3K4me3-CD4.bed | sort -k1,1 -k2,2n > LSK_vs_CD4_
↪H3K27ac_H3K4me3_CD4_sorted.bed
```

The *cat* command aggregates the input files for H3K27ac and H3K4me3 and pipes them (using the | operator) to a sort function which sorts by chromosome (*k1,1*) and first genomic coordinate (*k2,2n*). The result is stored in a specified output bed file (using the > operator).

4. Merge overlapping histone peaks using *bedtools merge* and intersect the merged regions with HINT-BCs footprint calls using *bedtools intersect*:

```
bedtools merge -i B_vs_CD4_H3K27ac_H3K4me3_B_sorted.bed | bedtools intersect -a stdin␣
↪-b ../input/Footprints/B.bed > Footprints_B_vs_CD4_H3K27ac_H3K4me3_B.bed
bedtools merge -i B_vs_CD4_H3K27ac_H3K4me3_CD4_sorted.bed | bedtools intersect -a␣
↪stdin -b ../input/Footprints/CD4.bed > Footprints_B_vs_CD4_H3K27ac_H3K4me3_CD4.bed

bedtools merge -i LSK_vs_CD4_H3K27ac_H3K4me3_LSK_sorted.bed | bedtools intersect -a␣
↪stdin -b ../input/Footprints/LSK.bed > Footprints_LSK_vs_CD4_H3K27ac_H3K4me3_LSK.bed
bedtools merge -i LSK_vs_CD4_H3K27ac_H3K4me3_CD4_sorted.bed | bedtools intersect -a␣
↪stdin -b ../input/Footprints/CD4.bed > Footprints_LSK_vs_CD4_H3K27ac_H3K4me3_CD4.bed

bedtools merge -i LSK_vs_B_H3K27ac_H3K4me3_LSK_sorted.bed | bedtools intersect -a␣
↪stdin -b ../input/Footprints/LSK.bed > Footprints_LSK_vs_B_H3K27ac_H3K4me3_LSK.bed
bedtools merge -i LSK_vs_B_H3K27ac_H3K4me3_B_sorted.bed | bedtools intersect -a stdin␣
↪-b ../input/Footprints/B.bed > Footprints_LSK_vs_B_H3K27ac_H3K4me3_B.bed
```

The *bedtools merge* command combines two overlapping regions into one region. The result of the intersection is piped into the standard input stream (*stdin*) of the *bedtools intersect -a* argument, while the *-b* argument is result of the Footprint calling. The resulting files will contain only footprints that intersect with a differential H3K4me3 and/or H3K27ac peak. In the next step, we will use these regions as candidate regions for TF binding. Precomputed results are stored in `/EpigenomicsTutorial-ISMB2017/session2/step2/result`.

By combining both footprints and differential peak calls of active chromatin marks we obtain a collection of canidate binding sites for TFs that are unique for expressed genes in one of the two tissues of interest.

Step3: Deriving candidate transcriptional regulators using *DYNAMITE*

During a *DYNAMITE* analysis, two main computational tasks are undertaken:

1. We calculate TF binding affinities for an example data set of 93 TFs and aggregate those to gene-TF scores using *TEPIC*. TF affinities are a quantitative measure of TF binding to a distinct genomic region.
2. A logistic regression classifier is learned that uses changes in TF gene scores between two samples to predict which genes are up/down- regulated between them. Investigating the features of the model allows the inference of potentially interesting regulators that are correlated to the observed expression changes.

Please check the documentation for details on the method.

We provide a script that automatically performs steps (1) and (2) as well as necessary data processing and formatting steps (See DYNAMITE documentation for details). All files used in this step are available in `/EpigenomicsTutorial-ISMB2017/session2/step3/input`. Additionally, we require the mm10 reference genome, which you should have downloaded while installing *HINT*.

Note that we precomputed the differential gene expression estimates. Computing those is neither part of the actual

tutorial nor of the *DYNAMITE* workflow. However a tool you could use to compute differential gene/transcript expression is Cuffdiff.

1. Assure that you are in the directory `EpigenomicsTutorial-ISMB2017/session2/step3`, otherwise *cd* to that directory.

2. Generate an output folder for the resulting files:

```
mkdir output
```

3. To run the *DYNAMITE* script go to the *DYNAMITE* folder in the *TEPIC* repository `TEPIC/MachineLearningPipelines/DYNAMITE`. We provide three configuration files for the *DYNAMITE* analyses:

1. DYNAMITE-LSKvsB.cfg
2. DYNMAITE-LSKvsCD4.cfg
3. DYNAMITE-BvsCD4.cfg

The configuration files are stored in the directory `EpigenomicsTutorial-ISMB2017/session2/step3/input`. They list all parameters that are needed for a run of *DYNAMITE*. To help you customise these files for later usage, we explain the essential parameters here:

- open_regions_Group1: One ore more files containing candidate transcription factor binding sites for samples belonging to group 1
- open_regions_Group2: One ore more files containing candidate transcription factor binding sites for samples belonging to group 2
- differential_Gene_Expression_Data: Differential gene expression data denoted with log2 fold changes
- outputDirectory: Directory to write the results to
- referenceGenome: Path to the reference genome that should be used
- chrPrefix: Flag indicating whether the reference genome uses a chr prefix
- pwm: Path to the pwms that should be used
- cores_TEPIC: Number of cores that are used in the TEPIC analysis
- geneAnnotation: Gene annotation file that should be used
- window: Size of the window around a genes TSS that is screened for TF binding sites
- decay: Flag indicating whether TEPIC should be using exponential decay to downweight far away regions while computing gene-TF scores
- peakFeatures: Flag indicating whether TEPIC should compute features based on peaks, e.g. peak count, peak length, or signal intensity within a peak

In the scope of the tutorial, you do not have to change any of those. A full description of all parameters is provided here.

4. Run the individual pairwise comparisons for LSK vs B:

```
bash runDYNAMITE.sh $HOME/EpigenomicsTutorial-ISMB2017/session2/step3/input/DYNAMITE-
↪LSKvsB.cfg
```

LSK vs CD4:

```
bash runDYNAMITE.sh $HOME/EpigenomicsTutorial-ISMB2017/session2/step3/input/DYNAMITE-
↪LSKvsCD4.cfg
```

and B vs CD4:

```
bash runDYNAMITE.sh $HOME/EpigenomicsTutorial-ISMB2017/session2/step3/input/DYNAMITE-
↪BvsCD4.cfg
```

Note that you have to **replace** the prefix `$HOME` with the proper path to the tutorial repository, if you have not cloned it to your *home* directory. The results of the analysis will be stored separately for each run in `EpigenomicsTutorial-ISMB2017/session2/step3/output`. There are three subfolders for each comparison:

1. Affinities
2. IntegratedData
3. Learning_Results

The folder *Affinities* contains TF affinities calculated in the provided regions for both groups, gene TF scores for both groups, and a metadata file that lists all settings used for the TF annotation with *TEPIC* (subfolders *group1* and *group2*). The subfolder *mean* contains the mean gene TF scores computed for group1 and group2. This is needed if you analyze more than one biological replicate per group. The folder *ratio* contains the gene TF score ratios computed between the gene TF scores of group1 and group2.

The folder *IntegratedData* encloses matrices that are composed of (1) gene TF score ratios and (2) a measure of differential gene expression. In the folder *Log2* the differential gene expression is represented as the log2 expression ratio between group1 and group2. In the folder *Binary*, the differential gene expression is shown in a binary way. Here, a 1 means a gene is upregulated in group 1 compared to group 2, whereas a 0 means it is down-regulated in group1. The binary format is used as input for the classification.

The folder *Learning_Results* comprises the results of the logistic regression classifier. The following files should be produced if all R dependencies are available:

1. Performance_overview.txt
2. Confusion-Matrix_<1..6>_Integrated_Data_For_Classification.txt
3. Regression_Coefficients_Cross_Validation_Integrated_Data_For_Classification.txt
4. Regression_Coefficients_Entire_Data_Set_Integrated_Data_For_Classification.txt
5. Performance_Barplots.png
6. Regression_Coefficients_Cross_Validation_Heatmap_Integrated_Data_For_Classification.svg
7. Regression_Coefficients_Entire_Data_SetIntegrated_Data_For_Classification.png
8. Misclassification_Lambda_<1..6>_Integrated_Data_For_Classification.svg

The file *Performance_overview.txt* contains accuracy on Test and Training data sets as well as F1 measures. These values are visualized in *Performance_Barplots.png*. As the name suggests, the files *Confusion-Matrix_<1..6>_Integrated_Data_For_Classification.txt* contain the confusion matrix computed on the test data sets. They show model performance by reporting True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN) in the following layout:

Observed/Predicted	Positive	Negative
Positive	TP	FN
Negative	FP	TN

The heatmap *Regression_Coefficients_Cross_Validation_Heatmap_Integrated_Data_For_Classification.svg* shows the regression coefficients of all selected features in the outer cross validation. This is very well suited to find features that are stably selected in all outer cross validation folds. The raw data used to generate the figure is stored in *Regression_Coefficients_Cross_Validation_Integrated_Data_For_Classification.txt*. The stronger a regression coefficient, the more important it is in the model.

In addition to the heatmap showing the regression coefficients during the outer cross validation, we also show the regression coefficients learned on the full data set: *Regression_Coefficients_Entire_Data_SetIntegrated_Data_For_Classification.png* and *Regression_Coefficients_Entire_Data_Set_Integrated_Data_For_Classification.txt*.

The figures *Misclassification_Lambda_<1..6>_Integrated_Data_For_Classification.svg* are of technical nature. They show the relationship between the misclassification error and the lambda parameter of the logistic regression function.

5. In addition to the plots describing model performance and feature selection generated by *DYNAMITE* (as described here), you can create further Figures for a distinct feature of interest using the script `TEPIC/MachineLearningPipelines/DYNAMITE/Scripts/generateFeaturePlots.R`. This will provide you with density plots showing the distribution of the feature in the two cell types, scatter plots linking feature value to gene expression changes, and a mini heatmap visualising the features regression coefficients.

To use this script, go to the output folder of step 3 `EpigenomicsTutorial-ISMB2017/session2/step3/output` and use the command

```
Rscript $HOME/TEPIC/MachineLearningPipelines/DYNAMITE/Scripts/generateFeaturePlots.R␣
↪LSK-vs-CD4 HOXA3 LSK CD4
```

This command will generate a plot comparing HOXA3 in LSK against CD4. Feel free to look at further features as you wish. The figure will be stored in the specified directory that contains the results of the *DYNAMITE* analysis. Again, note that you have to **replace** the prefix `$HOME` with the proper path used on your system, if necessary. Precomputed results are stored in `/EpigenomicsTutorial-ISMB2017/session2/step3/result`.