ChIP-seq Pipeline Explanation

This document will explain the ChIP-seq pipeline including the scripts used, arguments passed, and visualizations produced. The pipeline is split up into three separate sections which comprise multiple Bash, Python, and R scripts: upstream processing, peak & motif analysis, and the peak signal analysis. The following sections will address each section, refer to corresponding files in folder that this document is in. A visual depiction of the pipeline is attached at the end of this document.

Upstream Processing

The upstream processing portion of the pipeline demultiplexes and trims, runs QC on the FASTQ files, merges the files, maps the FASTQ files, and aligns them to a reference genome. The succeeding sections will describe the preceding steps.

Demultiplexing [/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/relacs] (see demultiplexing folder and readme.txt)]

The demultiplexing_relacs.py script trims the renames the 8-nucleotide RELACS barcode and 4-nucleotide long UMI and stores them in the header of the FASTQ file [1]. The script renames the FASTQ files according to the barcode, which indicates the phenotype and genotype of the sample. The sampleTable.tsv file is an example of the tab-separated sample table that the script uses to perform the demultiplexing. The first column is the name of the output folder, the second column is barcode, and the third column indicates the sample identifier. If the barcode is not found in the FASTQ file, it is named as unknown.

Script used:
/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/relacs/demultiplexing/sub.sh

Merging [secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/relacs] (see merge folder and readme.txt)]]

The merge_server.py script creates multiple submission scripts which can be submitted as an array job on the PBS queuing system. It assumes that all unmerged FASTQ files have the same name. It merges the files with the highest quality setting, n9. All inputs and outputs are commented in the Python script. The submit.sh script submits the submission scripts on the PBS system.

Scripts Used:
/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/relacs/merge/combined2/merge_server.py
/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/relacs/merge/combined2/scripts2/submit2.sh

Alignment [snakePipes]

The demultiplexed and merged FASTQ files are then aligned using the DNA-mapping function a part of the snakePipes workflow [2]. See the DNA-mapping.sub2.sh for specific usage of function and the documentation for explanation of what the function is doing.

Deduplication [umi-tools],
[/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/deduplication2/]

The filtered bam files are deduplicated using the UMI tools with a UMI length of four nucleotides [3]. The dedup_script.py write individual submission scripts for each bam file so that it can be submitted as an array job on the PBS system. Its inputs are the directory containing the bam files and the outputs are the script and output directory for the deduplicated bam files. The Python script passes arguments using the argument parser. The submit.sh shell script submits the individual scripts as an array job.

Scripts Used:
/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/deduplication2/scripts/dedup_script.py
/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/deduplication2/scripts/submit.sh

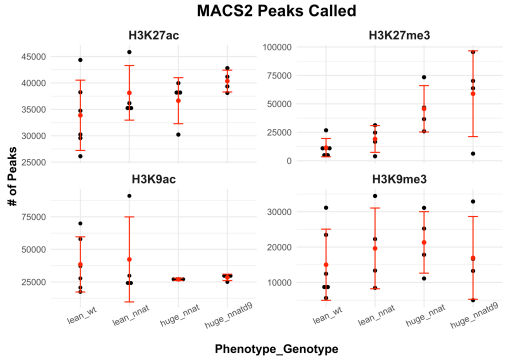
Peak Calling [snakePipes]

The ChIP-seq function in snakePipes is used to call the peaks from the deduplicated bam files [2]. The function requires a ChIP dictionary yaml file (ChIPdict.yaml) and the make.dic.py script creates a yaml file in the format required. The submission script shows how the ChIP-seq function was called for the RELACS protocol.

Peak and Motif Analysis

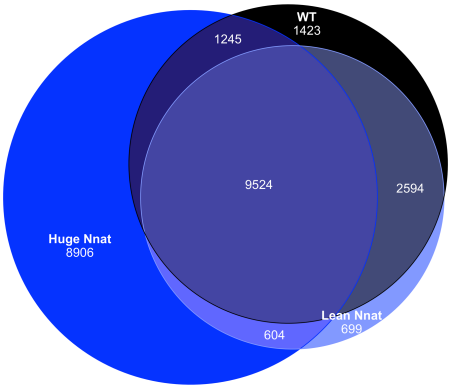
Peak Counting and Comparison
[/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/ChIPseq2/Peak_Counting_Compare]

The peakcount.R script creates a plot that shows how many peaks are called given a directory with bam files by counting the number of lines in each file. It assumes that the file names is as follows: animalnumber_phenotype_genotype_sex_antibody. The Rscript has arguments that can be parsed and the help file is shown when passing "Rscript peakcount.R -h" to the Linux command line.



The peak_compare.R script finds the intra and inter phenotype variability and saves a set of peaks according to each phenotype. It also outputs a Venn diagram that depicts the peaks that are conserved and different between phenotypes. This script assumes that there are six replicates for the name "lean_wt" and four replicates for the names "huge_nnat" and "lean_nnat". The code for this is changeable but may take a considerable amount of time. If we were to pursue the change, we would need to make sure that the ChIPpeakAnno package is ok to use (need to check the

makeVennDiagram for p-values if you are going to use). It also assumes that there are three phenotype_genotype combinations.



Scripts Used:
/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/ChIPseq2/Peak_Counting_Compare/peakcount.R

/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/ChIPseq2/Peak_Counting_Compare/peak_compare.R

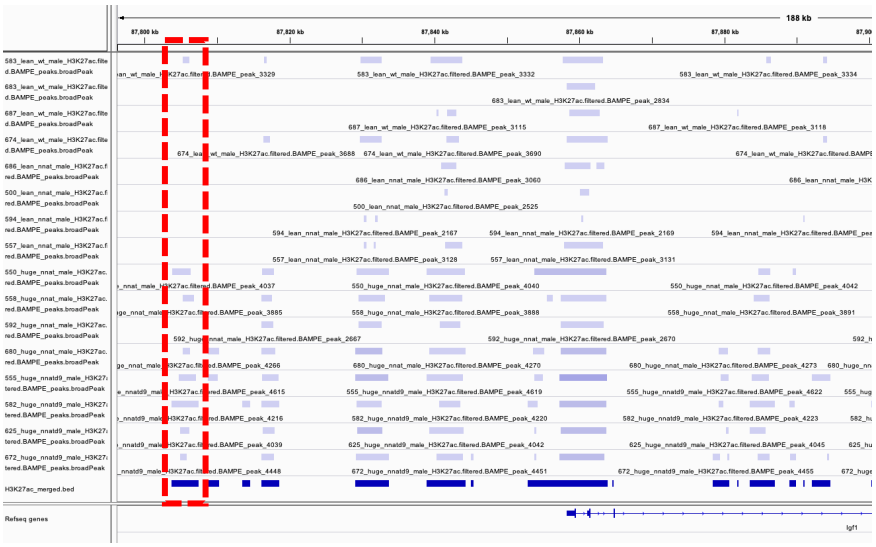
Motif Analysis [/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/ChIPseq2/H3K27ac_PeakCompare_Motif]

The motif analysis portion of the pipeline takes peaks specific to a group and searches for consensus motifs among the set. It's inputs are a primary and control bed file and outputs motif discovery and enrichment results from MEME and Homer. The motif6.py is parsable and shows the needed arguments with it's help message. It requires that the output directory do not have folders named "homer" or "meme" in it, because it can not overwrite them. The motif6.sh script is an example of how it can be ran in a shell script. The get_homer_gene.R script parses the transcription factor names produced by the top Homer results which can be used in a GO analysis.

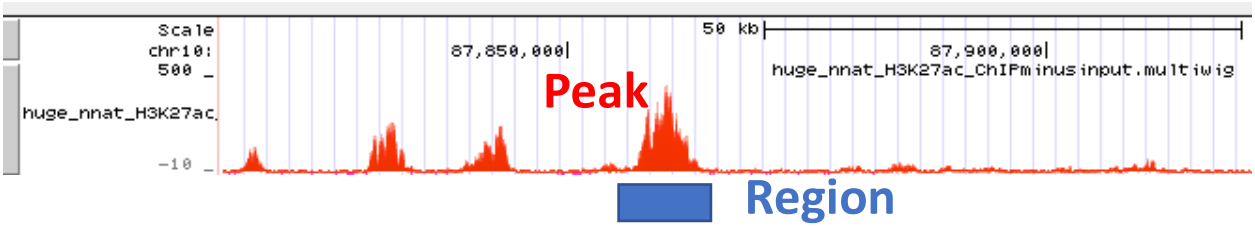
Peak Signal Analysis

Peak Merging and Signal Quantification [/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/ChIPseq2/Peak_Counting_Compare]

The peaks called from the MACS2 broad peak caller are filtered and merged with a given minimum overlap to create a set of regions across all samples that are used to quantify signal, which is accomplished by the bedmerge_filter.R script. It's inputs are a directory to the MACS2 regions, a threshold to filter the regions based on FDR, the minimum overlap threshold, a vector of antibody names, and number of cores. Its output is a directory where the merged bedfiles for each antibody is written. See below for visual depiction of peak merging.



The signal of the peaks are quantified by taking the sum of the product of the length and width of each peak within the bounds defined by a bed file. The signal files are programmed to be in bigwig format. This part of the pipeline utilizes the Get.enrichment.for.unions.R script by submitting an array job for each sample. The enrichment_script_maker.py script makes individual submission scripts for the array job, the script has input arguments that are appropriately commented. The enrichment.sh script submits the array job.



Scripts Used:

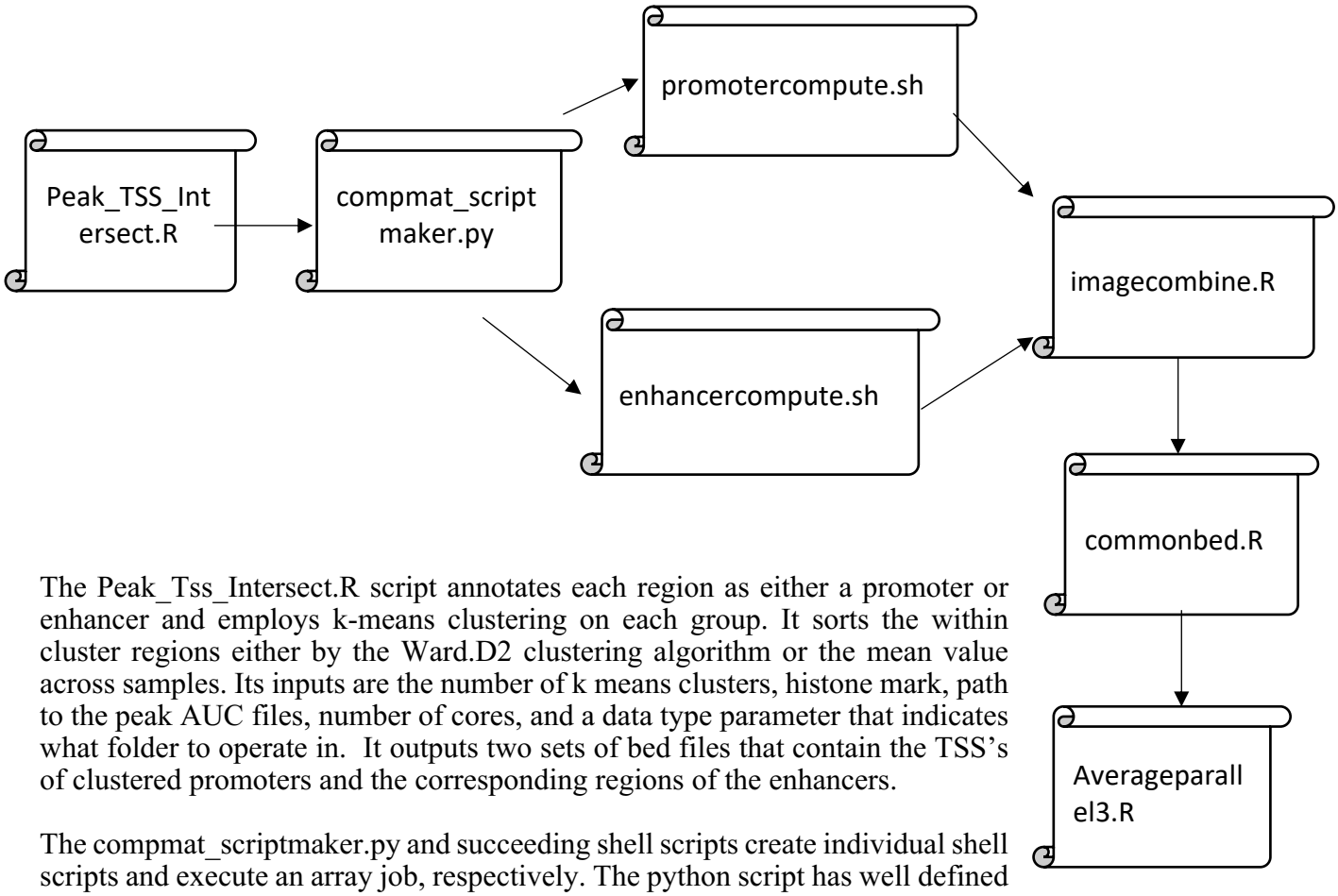
/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/ChIPseq2/Peak_Counting_Compare/bedmerge_filter.R

/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/ChIPseq2/H3K27ac_seqdepthnorm/enrichment_script_maker.py

/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/ChIPseq2/H3K27ac_seqdepthnorm/enrichment.sh

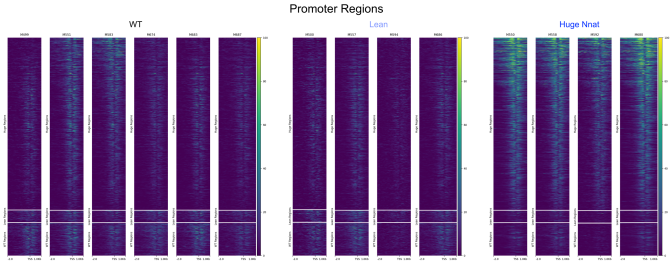
Annotation, Clustering, Heatmaps, and Average Plots
[/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/ChIPseq2/H3K27ac_seqdepthnorm]

This series of scripts clusters the peak AUCs, annotates which regions are promoter or enhancer regions, and produces visualization that depict the signal within a window of either the TSS or peak center, respectively. A promoter is defined as a TSS exisiting within a region and all other regions are classified as an enhancer. Below is a depiction of the order of how the scripts are ran.

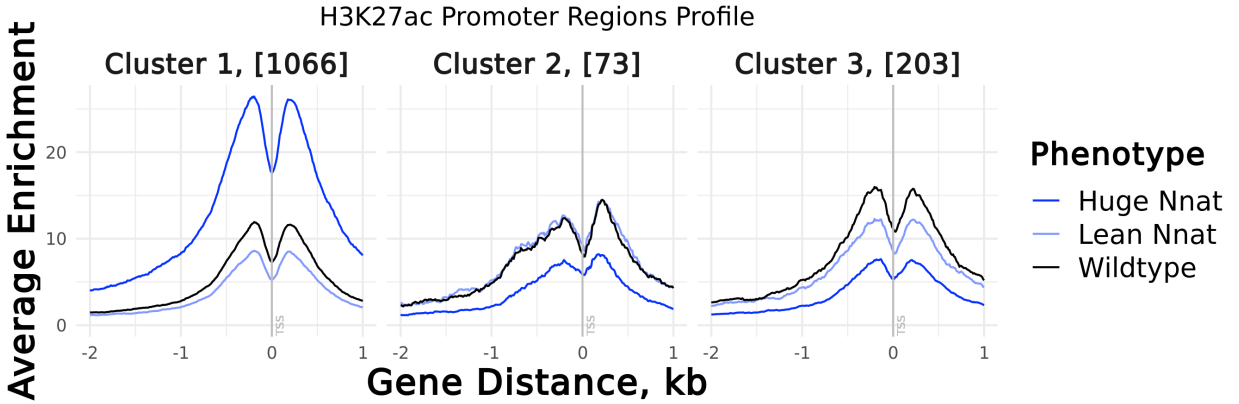


The Peak_Tss_Intersect.R script annotates each region as either a promoter or enhancer and employs k-means clustering on each group. It sorts the within cluster regions either by the Ward.D2 clustering algorithm or the mean value across samples. Its inputs are the number of k means clusters, histone mark, path to the peak AUC files, number of cores, and a data type parameter that indicates what folder to operate in. It outputs two sets of bed files that contain the TSS's of clustered promoters and the corresponding regions of the enhancers.

The compmat_scriptmaker.py and succeeding shell scripts create individual shell scripts and execute an array job, respectively. The python script has well defined inputs and out that are shown when opening the script in the comments. Each individual submission script runs the deepTools computeMatrix and plotHeatmap commands sequentially with select parameters. The regions label parameter needs to be adjusted to the number of clusters produced by the previous step and does not automatically detect how many bed files. A heatmap is produced for each sample for each set of regions. The heatmaps are combined into one plot each for promoters or enhancers by using the imagecombine.R script.



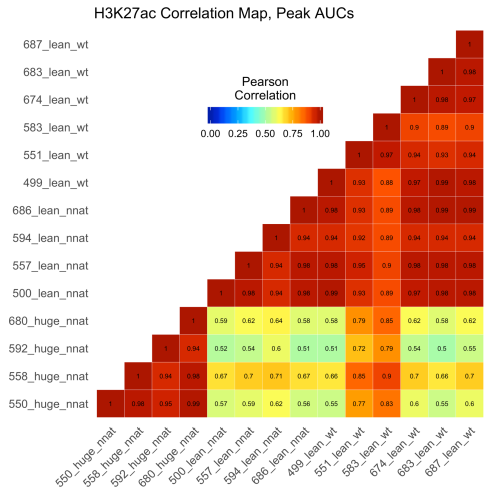
The commonbed.R script creates common bed file for either the enhancers or promoters that is inclusive of all regions. The common bedfile allows for an average to be calculated consistently across all samples. The averageplot.R script calculates an average signal across each phenotype using the common bed file. Its inputs are shown under the parameters portion of the script and it outputs an average plot for each cluster.

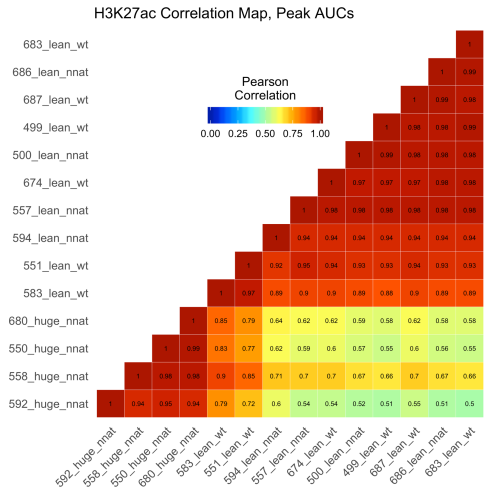


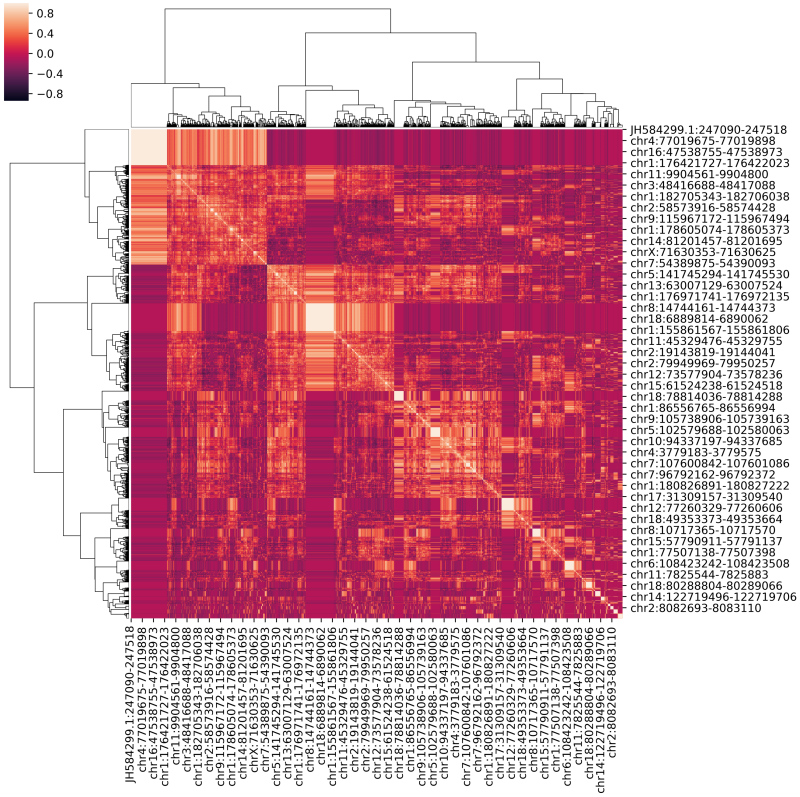
Scripts used:
/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/ChIPseq2/H3K27ac_seqdepthnorm/enrichment_script_maker.py

Correlation Maps

The scripts in this folder create correlation maps between samples and features. The cormat.R creates a correlation map for the samples. It's inputs are two Rdata objects, one that contains the feature by sample matrix and the other is a matrix that contains metadata regarding the samples. It outputs two plots, one ordered by phenotype and the other ordered by the hierarchical clustering algorithm Ward.D2. The feature_correlation.py script creates a visualization of the correlation between features. Its input is a feature by sample matrix and outputs a clustered correlation matrix. Python is used because it is less memory intensive. The script approximately takes 8 hours for a 24,000 by 18 matrix.

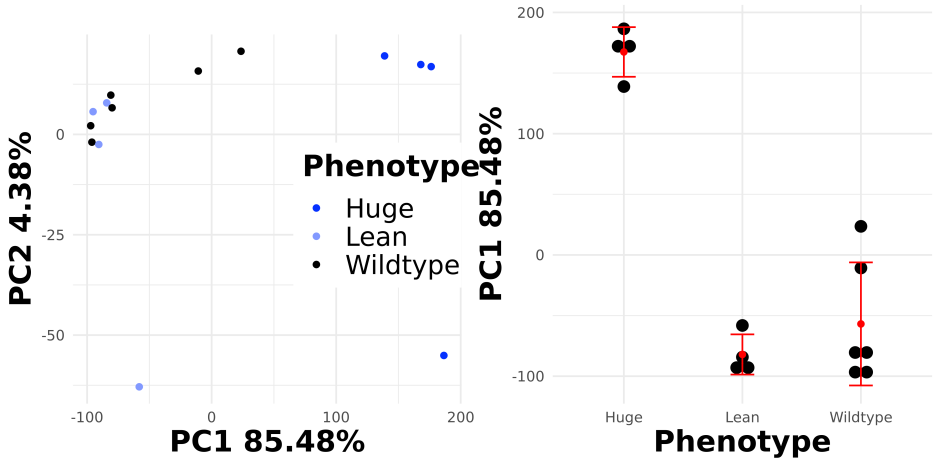






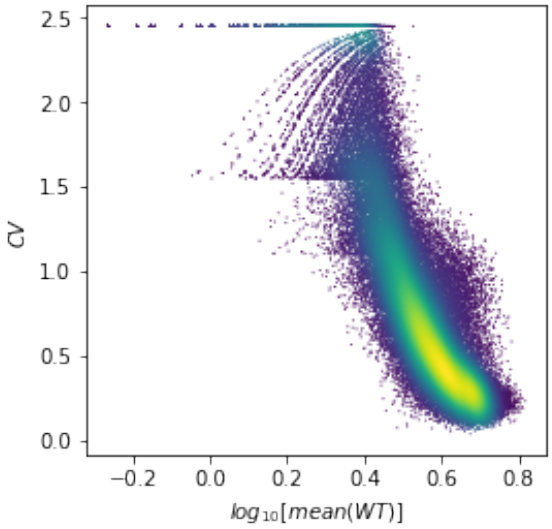
Dimensionality Reduction

The filter_PCA.R script performs a pca on z-score scaled peak AUC values. It's inputs are a Rdata object that contains the feature by sample data frame, a tab separated file that has phenotypic data for the samples, and corresponding colors to desired phenotype groups. The output is a plot that displays PC1 and PC2 values for each sample as well as a dotplot that shows the variation on PC1. The loadings can be obtained using the svd command in R on the matrix which is useful for a GO analysis.



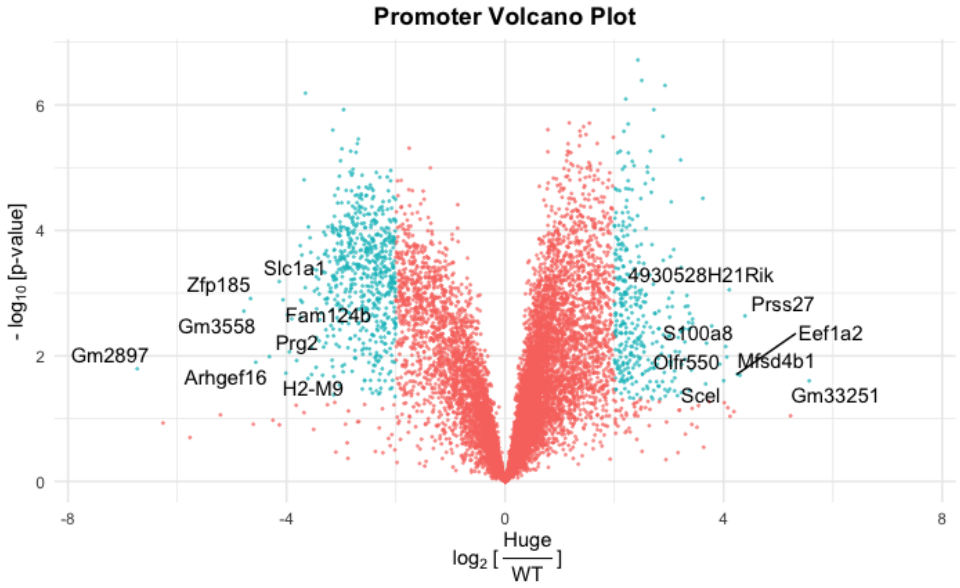
Feature Plots

The peak AUCs between samples are visualized to observe patterns of signal. The cv_signal_group.py script produces semi-log plots that depict the coefficient of variation and the mean signal of a phenotype group. Its input is a tab-delimited matrix. The CV is defined as the standard deviation divided by the mean. Below is an example plot of unfiltered matrix. The patterns seen in the upper portion of the plot could be artifacts of dividing by a number close to zero—it is recommended to do some sort of filtering of regions before proceeding with data analysis. This script can be altered to produce density plots of different types.

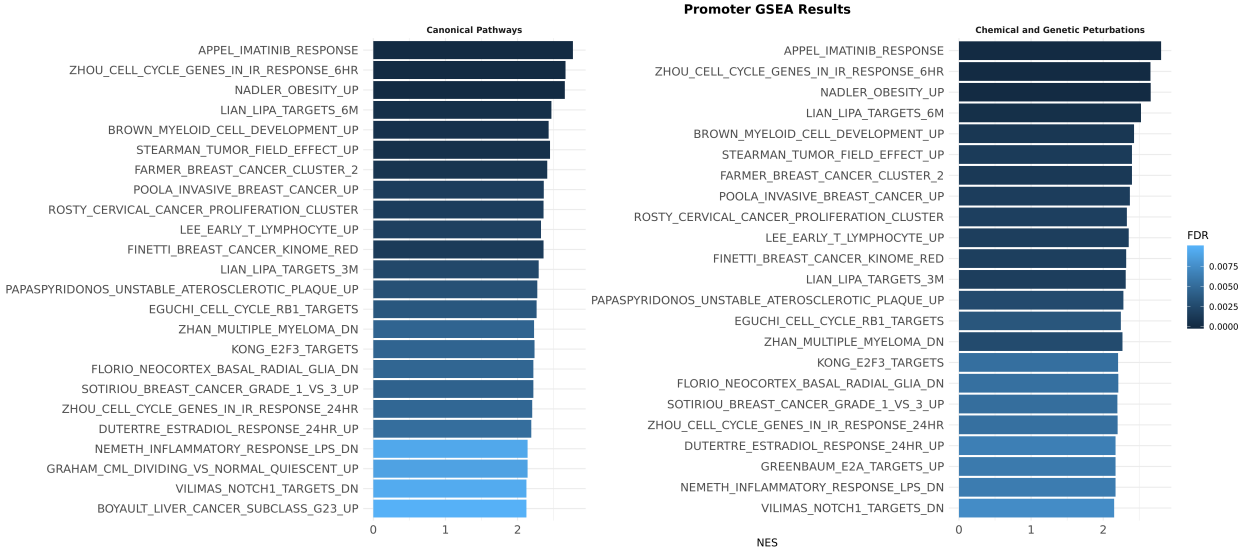


Differential Testing and GSEA

The diffanalysis.Rmd markdown document executes a two independent sample t-test on two groups of interest in both the promoter and enhancer regions. The markdown displays a volcano plot that shows the TSS(s) that are within the promoter regions and the nearest gene for the enhancer regions. P-value and $\log_2$ fold change cutoffs are implemented as parameters within the document.



The make_dup_df.R script makes duplicate features if there are multiple TSS's within a promoter region. Its outputs a .txt and .cls file needed for the GSEA software. The gseaplot.R script combines results that are obtained from the GSEA analyses. Its input is a directory that contains results from the GSEA software and assumes the following naming format for the analysis: Testtype_genesetname_permutationtype_groups_regiontype(ex:s2n_c2cp_geneset_hugewt_promoter).



UCSC Scripting (/secondary/projects/pospisilik/UCSC)

Three commands need to be ran in order to synchronize data to the UCSC website. The trackhub_org.py script writes and links all pertinent files into one folder (.super), with multiple multiwigs in it. The trackHub_generator.py script outputs the files and trackDb in the folder to be synchronized to AWS. An example of the third command is shown below:

aws s3 sync mm10_upload/ s3://pospisiliklab.aws.vai.org/trackhub/mm10/ --grants read=uri=http://acs.amazonaws.com/groups/global/AllUsers

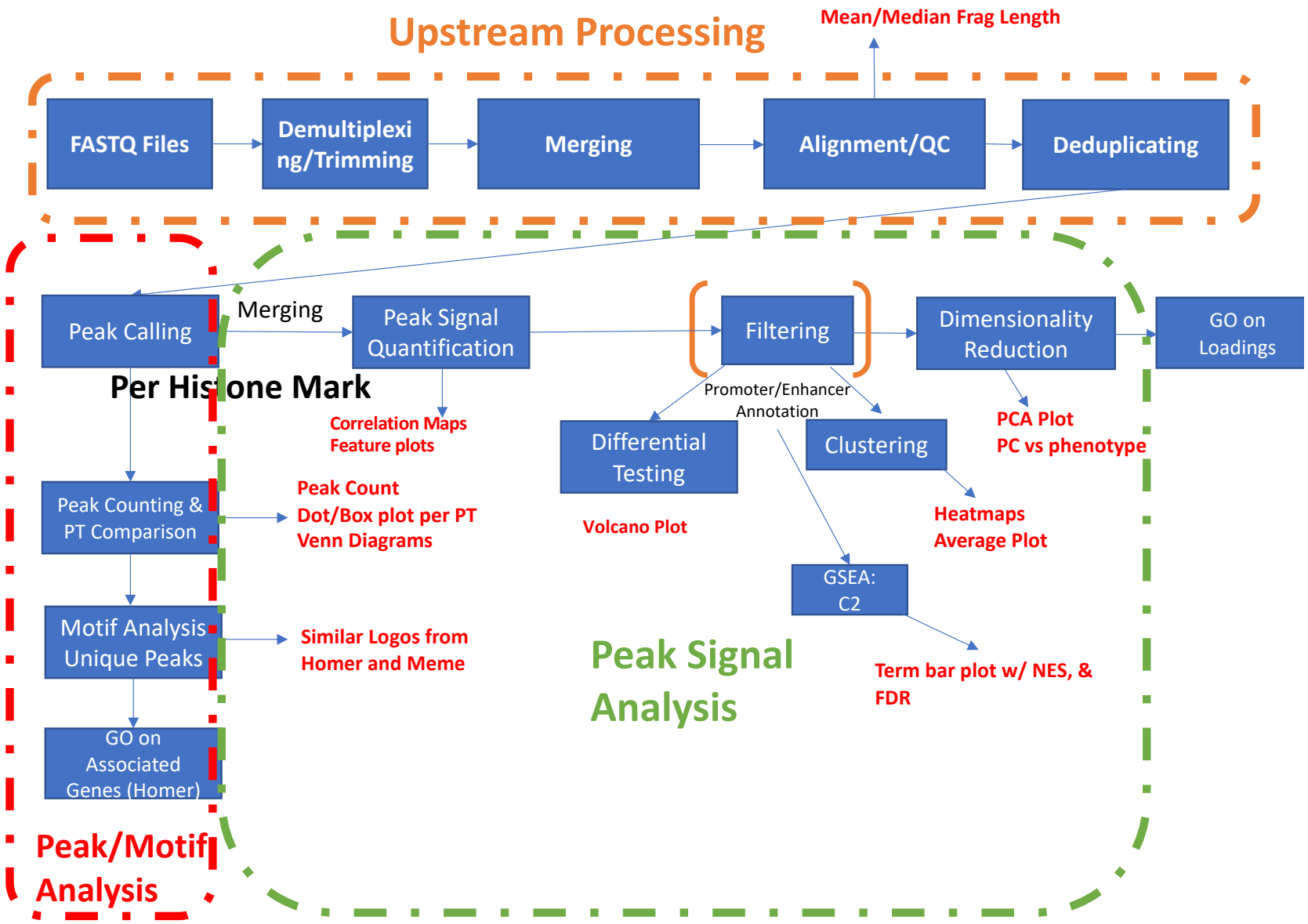
The readme.txt file in the given folder contains an example of how each command is used. The trackhub_org.py script may need adjusting depending on file names / projects (there are two in the folder, one for Luca and one for Yang). See argument parser help for further explanation of arguments.

Important Directories

Deduplicated sequencing depth normalized files for Yang's project : /secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/deduplication2/output/bws2/

Yang's Project:
/secondary/projects/mnp/nick/Leslie.Yang/adipocyte_data/

Ilaria's Project (still need to call ChIP-seq from snakePipes):
/secondary/projects/mnp/nick/Ilaria.Panzeri/Trim28_EedxD9


Upstream Processing
Mean/Median Frag Length
FASTQ Files
Demultiplexing/Trimming
Merging
Alignment/QC
Deduplicating
Peak Calling
Merging
Peak Signal Quantification
Filtering
Dimensionality Reduction
GO on Loadings
Per Histone Mark
Correlation Maps
Feature plots
Promoter/Enhancer Annotation
PCA Plot
PC vs phenotype
Differential Testing
Clustering
Peak Counting & PT Comparison
Peak Count
Dot/Box plot per PT
Venn Diagrams
Volcano Plot
Heatmaps
Average Plot
GSEA: C2
Motif Analysis Unique Peaks
Similar Logos from Homer and Meme
Peak Signal Analysis
Term bar plot w/ NES, & FDR
GO on Associated Genes (Homer)
Peak/Motif Analysis