Analysis_reproduced

Chandrima

2025-06-15

Load library

```
# Loading the packages required for preprocessing
suppressPackageStartupMessages({
  library(ggplot2)
  library(ggrepel)
  library(RColorBrewer)
  library(DESeq2)
  library(openxlsx)
  library(cowplot)
  library(readr)
  library(edgeR)
  library(rhdf5)
  library(stringr)
  library(sva)
  library(vsn)
  library(genefilter)
  library(TxDb.Mmusculus.UCSC.mm10.knownGene)
  library(rGREAT)
  library(org.Mm.eg.db)
  library(ChIPpeakAnno)
  library(tidyverse)})
```

```
## Warning: package 'GenomicFeatures' was built under R version 4.3.3
```

Sourcing Functions

```
source("C:/practice_bulk/pre_processing_data.R")
source("C:/practice_bulk/Quality_control.R")
source("C:/practice_bulk/Run_dge.R" )
source("C:/practice_bulk/gene_annotation.R")
```

Sourcing data from Deciphering the role of histone modifications in memory and exhausted CD8 T cells https://www.nature.com/articles/s41598-025-99804-0

```
path <- "C:/practice_bulk/"
dfCounts.H3K4me <- read_tsv(paste0(path, "GSE285245_RawRC_H3K4me3_cleanID.txt"))
```

Loading CUT&RUN data first from the paper

```
## Rows: 50322 Columns: 22
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr  (3): Geneid, Chr, Strand
## dbl (19): Start, End, Length, Arm_P14_Day30_H3K4me3_Rep1_S5, Arm_P14_Day30_H...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dfCounts.H3K9me3 <- read_tsv(paste0(path, "GSE285245_RawRC_H3K9me3_cleanID.txt"))
```

```
## Rows: 25434 Columns: 22
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr  (3): Geneid, Chr, Strand
## dbl (19): Start, End, Length, Arm_P14_Day30_H3K9me3_Rep1_S3, Arm_P14_Day30_H...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dfCounts.H3K27ac <- read_tsv(paste0(path, "GSE285245_RawRC_H3K27ac_cleanID.txt"))
```

```
## Rows: 38662 Columns: 22
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr  (3): Geneid, Chr, Strand
## dbl (19): Start, End, Length, Arm_P14_Day30_H3K27ac_Rep1_S5, Arm_P14_Day30_H...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
dfCounts.H3K27me3 <- read_tsv(paste0(path, "GSE285245_RawRC_H3K27me3_cleanID.txt"))
```

```
## Rows: 32659 Columns: 22
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr  (3): Geneid, Chr, Strand
## dbl (19): Start, End, Length, Arm_P14_Day30_H3K27me3_Rep1_S7, Arm_P14_Day30_...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# Bedfiles
bedData.H3K4me <- dfCounts.H3K4me[,2:6]
bedData.H3K9me3 <- dfCounts.H3K9me3[,2:6]
bedData.H3K27ac <- dfCounts.H3K27ac[,2:6]
bedData.H3K27me3 <- dfCounts.H3K27me3[,2:6]
```

```
#make the rownames as chromosome loci
dfCounts.H3K4me <- SetRow(dfCounts.H3K4me)
dfConditions.H3K4me <- SetCondition(dfCounts.H3K4me)

dfCounts.H3K9me3 <- SetRow(dfCounts.H3K9me3)
dfConditions.H3K9me3 <- SetCondition(dfCounts.H3K9me3)

dfCounts.H3K27ac <- SetRow(dfCounts.H3K27ac)
dfConditions.H3K27ac <- SetCondition(dfCounts.H3K27ac)

dfCounts.H3K27me3 <- SetRow(dfCounts.H3K27me3)
dfConditions.H3K27me3 <- SetCondition(dfCounts.H3K27ac)
```

Running basic pre-processing of the counts matrix In the paper no additional filtering was done for CUT&RUN sequencing data

So we direct run the deseq

```
deRNA.H3K4me <- RunDeseq(dfCounts.H3K4me, dfConditions.H3K4me)
```

```
## converting counts to integer mode

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

```
deRNA.H3K9me3 <- RunDeseq(dfCounts.H3K9me3, dfConditions.H3K9me3)
```

```
## converting counts to integer mode

## estimating size factors

## estimating dispersions
```

```
## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

```
deRNA.H3K27ac <- RunDeseq(dfCounts.H3K27ac,dfConditions.H3K27ac)
```

```
## converting counts to integer mode

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

```
deRNA.H3K27me3 <- RunDeseq(dfCounts.H3K27me3 , dfConditions.H3K27me3)
```

```
## converting counts to integer mode

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

Now we run PCA on the DESeq2 Normalized data

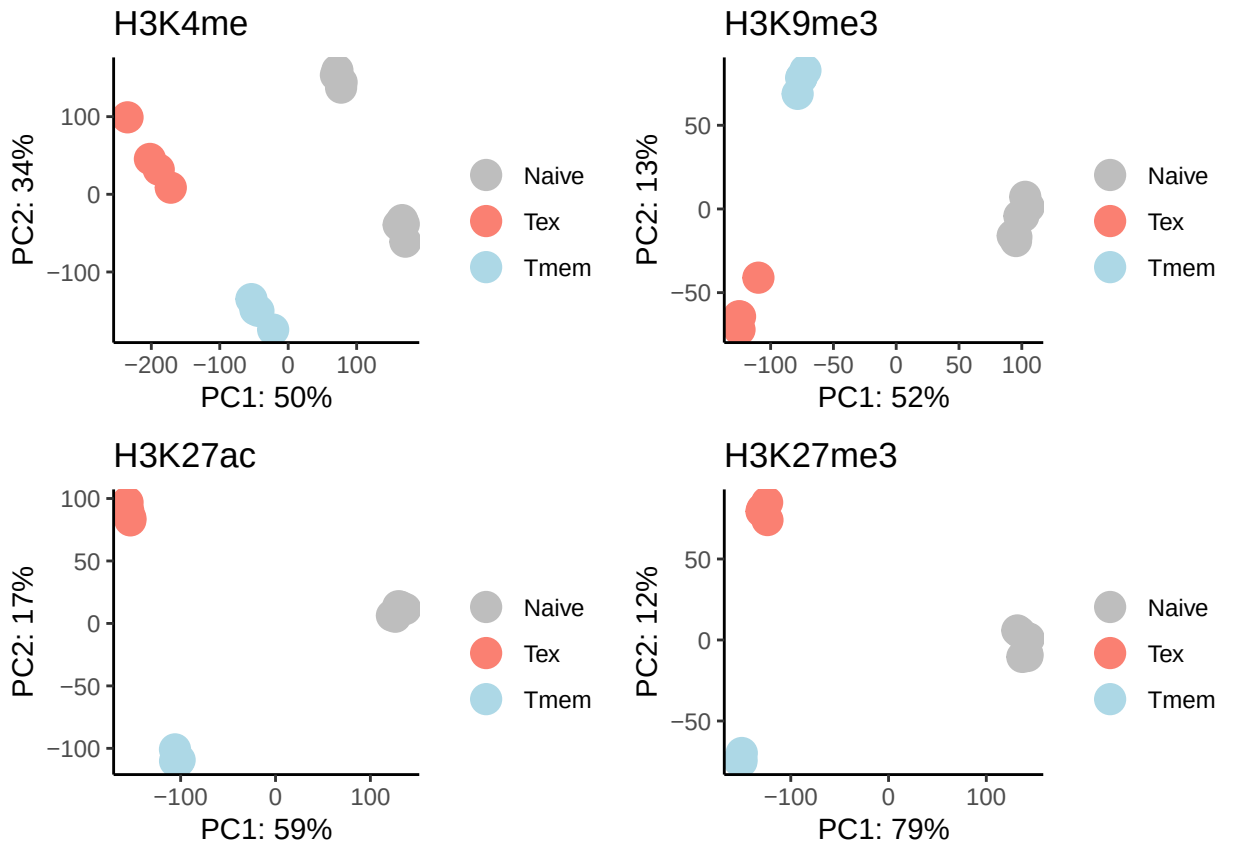
```r
# Normalized data
dfNormalizedCounts.H3K4me <- as.data.frame(counts(deRNA.H3K4me, normalized = T))
p.H3K4me <- RunPCA(dfNormalizedCounts.H3K4me, dfConditions.H3K4me)+ggtitle("H3K4me")

dfNormalizedCounts.H3K9me3<- as.data.frame(counts(deRNA.H3K9me3, normalized = T))
p.H3K9me3 <- RunPCA(dfNormalizedCounts.H3K9me3, dfConditions.H3K9me3)+ggtitle("H3K9me3")

dfNormalizedCounts.H3K27ac <- as.data.frame(counts(deRNA.H3K27ac, normalized = T))
p.H3K27ac <- RunPCA(dfNormalizedCounts.H3K27ac, dfConditions.H3K27ac)+ggtitle("H3K27ac")

dfNormalizedCounts.H3K27me3 <- as.data.frame(counts(deRNA.H3K27me3, normalized = T))
p.H3K27me3 <- RunPCA(dfNormalizedCounts.H3K27me3, dfConditions.H3K27me3)+ggtitle("H3K27me3")

plot_grid(ncol = 2, p.H3K4me,p.H3K9me3,p.H3K27ac, p.H3K27me3)
```



As is visible the analysis was reproduced to the papers figure 1 panel b. Now we compute differentially expressed peak regions for all the sample, we started with making pairwise comparison dataframe which will used to loop for the combinations

```r
# Taking in combinations of two samples to create pairwise samples
dfConditions.H3K27ac$celltyp <- as.factor(dfConditions.H3K27ac$celltyp)
dfPairwiseCond <- as.data.frame(combn(levels(as.factor(dfConditions.H3K27ac$celltyp)), 2))
# Making labels for plots
groupLabels <- sapply(dfPairwiseCond, function(x) {
  paste0(x[[1]], "vs", x[[2]])})
```

One the sample was not runned because batch correction issue

```
DPE.H3K9me3 <- RunDGE(deRNA.H3K9me3, dfConditions.H3K9me3)
```

```
## [1] "NaivevsTex"
## [1] "NaivevsTmem"
## [1] "TexvsTmem"
```

```
DPE.H3K27ac <- RunDGE(deRNA.H3K27ac, dfConditions.H3K27ac)
```

```
## [1] "NaivevsTex"
## [1] "NaivevsTmem"
## [1] "TexvsTmem"
```

```
DPE.H3K27me3 <- RunDGE(deRNA.H3K27me3, dfConditions.H3K27me3)
```

```
## [1] "NaivevsTex"
## [1] "NaivevsTmem"
## [1] "TexvsTmem"
```

Now we RNA seq Data pre processing and DGE Loading the counts matrix data

```
dfCounts <- read_tsv(paste0(path, "GSE285248_RawRC_RNA.txt"))
```

```
## Rows: 55471 Columns: 22
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr  (4): Chr, Geneid, Gene.Name, Strand
## dbl (18): Start, End, Naive_P14_Day30_RNA_Rep1_S1, Naive_P14_Day30_RNA_Rep2_...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# Saving the first few columns as bed files
bedData <- dfCounts[,1:6]

# Removing any duplicate gene names  before setting them as rownames
dfCounts <- dfCounts[!duplicated(dfCounts$Gene.Name),]

# Making the counts matrix
dfCounts <- dfCounts |>
  select(c(5,7:ncol(dfCounts))) |>
  column_to_rownames("Gene.Name")
```

Making the experiment design and removing low quality reads setting the cut more than 5 as per the paper

```
dfConditions <- SetCondition(dfCounts)
# Setting the design as factor
dfConditions$celltyp <- as.factor(dfConditions$celltyp)
dfConditions$time <- as.factor(dfConditions$time)

# Dataframe, margin(1 for row, 2 for column) in the apply function
dfCounts <- subset(dfCounts, subset = apply(dfCounts, 1,max) > 5)
```

Running DESeq

```
deRNA <- RunDeseq(dfCounts, dfConditions)
```

```
## converting counts to integer mode

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

```
DGE.rna <- RunDGE(deRNA, dfConditions)
```

Now we run DGE

```
## [1] "NaivevsTex"
## [1] "NaivevsTmem"
## [1] "TexvsTmem"
```

Now we will make the plot in Figure 1 panel c

We start with making the data frame with cutoffs of p_adj <0.05 and absolute logFC > 1.5 for memory vs exhausted rna expression

```
# Select Padj and FC columns of Tex vs Tmem comparison
dge_sig.rna <- as.data.frame(cbind(DGE.rna$qvals$TexvsTmem_padj,DGE.rna$fc$TexvsTmem_fc))
# Setting column names
colnames(dge_sig.rna) <- c("RNA_padj", "RNA_FC")
#Setting rownames
rownames(dge_sig.rna) <- rownames(DGE.rna$qvals)
# Subetting the DGE based upon the cut-off
dge_sig.rna <- subset(dge_sig.rna, RNA_padj < 0.05 & abs(RNA_FC) > 1.5)

# Saving the rownames as column gene
dge_sig.rna <- dge_sig.rna |> rownames_to_column("gene")
```

Now we do the same for H3K27ac and H3K27me3

```r
# For H3K27ac
dge_sig.H3K27ac <- as.data.frame(cbind(DPE.H3K27ac$qvals$TexvsTmem_padj,DPE.H3K27ac$fc$TexvsTmem_fc))
colnames(dge_sig.H3K27ac) <- c("H3K27ac_padj", "H3K27ac_FC")
rownames(dge_sig.H3K27ac) <- rownames(DPE.H3K27ac$qvals)
dge_sig.H3K27ac <- subset(dge_sig.H3K27ac, H3K27ac_padj < 0.05 & abs(H3K27ac_FC) > 1.5)
dge_sig.H3K27ac <- dge_sig.H3K27ac |> rownames_to_column("ID")
#H3K27me3
dge_sig.H3K27me3 <- as.data.frame(cbind(DPE.H3K27me3$qvals$TexvsTmem_padj,DPE.H3K27me3$fc$TexvsTmem_fc))
colnames(dge_sig.H3K27me3) <- c("H3K27me3_padj", "H3K27me3_FC")
rownames(dge_sig.H3K27me3) <- rownames(DPE.H3K27me3$qvals)
dge_sig.H3K27me3 <- subset(dge_sig.H3K27me3, H3K27me3_padj < 0.05 & abs(H3K27me3_FC) > 1.5)
dge_sig.H3K27me3 <- dge_sig.H3K27me3 |> rownames_to_column("ID")
```

Now we make bed data frame and get the gene annotation

```r
# For H3K27ac
H3K27ac.associated_genes <- GeneAnnot(dge_sig.H3K27ac)
```

```
## * TSS source: TxDb.

## * check whether TxDb package 'TxDb.Mmusculus.UCSC.mm10.knownGene' is installed.

## * gene ID type in the extended TSS is 'Entrez Gene ID'.

## * restrict chromosomes to 'chr1, chr2, chr3, chr4, chr5, chr6, chr7, chr8, chr9, chr10,
##     chr11, chr12, chr13, chr14, chr15, chr16, chr17, chr18, chr19, chrX, chrY, chrM'.

## * 20585/24515 protein-coding genes left.

## * update seqinfo to the selected chromosomes.

## * TSS extension mode is 'basalPlusExt'.

## * construct the basal domains by extending 5000bp to upstream and 1000bp to downsteram of TSS.

## * calculate distances to neighbour regions.

## * extend to both sides until reaching the neighbour genes or to the maximal extension.

## duplicated or NA names found. Rename all the names by numbers.

## * use GO:BP ontology with 15848 gene sets (source: org.Mm.eg.db).

## * check gene ID type in 'gene_sets' and in 'extended_tss'.

## * use whole genome as background.

## * remove excluded regions from background.
```

```
## * overlap ‘gr‘ to background regions (based on midpoint).

## * in total 3146 ‘gr‘.

## * overlap extended TSS to background regions.

## * check which genes are in the gene sets.

## * only take gene sets with size >= 5.

## * in total 9464 gene sets.

## * overlap ‘gr‘ to every extended TSS.

## * perform binomial test for each biological term.
```

```
# Inner join the data frames of Fold Change and annotated genes to the peak regions
dge_sig.H3K27ac <- inner_join(dge_sig.H3K27ac, H3K27ac.associated_genes, by = "ID")

# We split the multiple genes with each peak region and set the colname as gene
dge_sig.H3K27ac <- unnest(dge_sig.H3K27ac, annotated_genes)
colnames(dge_sig.H3K27ac)[4] <- "gene"

# For H3K27me3
H3K27me3.associated_genes <- GeneAnnot(dge_sig.H3K27me3)
```

```
## * TSS source: TxDb.

## * extended_tss is already cached, directly use it.

## duplicated or NA names found. Rename all the names by numbers.

## * use GO:BP ontology with 15848 gene sets (source: org.Mm.eg.db).

## * check gene ID type in ‘gene_sets‘ and in ‘extended_tss‘.

## * use whole genome as background.

## * remove excluded regions from background.

## * overlap ‘gr‘ to background regions (based on midpoint).

## * in total 1347 ‘gr‘.

## * overlap extended TSS to background regions.

## * check which genes are in the gene sets.

## * only take gene sets with size >= 5.
```

```
## * in total 9464 gene sets.

## * overlap 'gr' to every extended TSS.

## * perform binomial test for each biological term.
```

```
# Inner join the data frames of Fold Change and annotated genes to the peak regions
dge_sig.H3K27me3 <- inner_join(dge_sig.H3K27me3, H3K27me3.associated_genes, by = "ID")

# We split the multiple genes with each peak region and set the colname as gene
dge_sig.H3K27me3 <- unnest(dge_sig.H3K27me3, annotated_genes)
colnames(dge_sig.H3K27me3)[4] <- "gene"
```

Now join the the data to RNA DGE to make the plots

```
joined_df.rna.H3K27ac <- inner_join(dge_sig.H3K27ac, dge_sig.rna, by = "gene")
joined_df.rna.H3K27me3 <- inner_join(dge_sig.H3K27me3, dge_sig.rna, by = "gene")
```

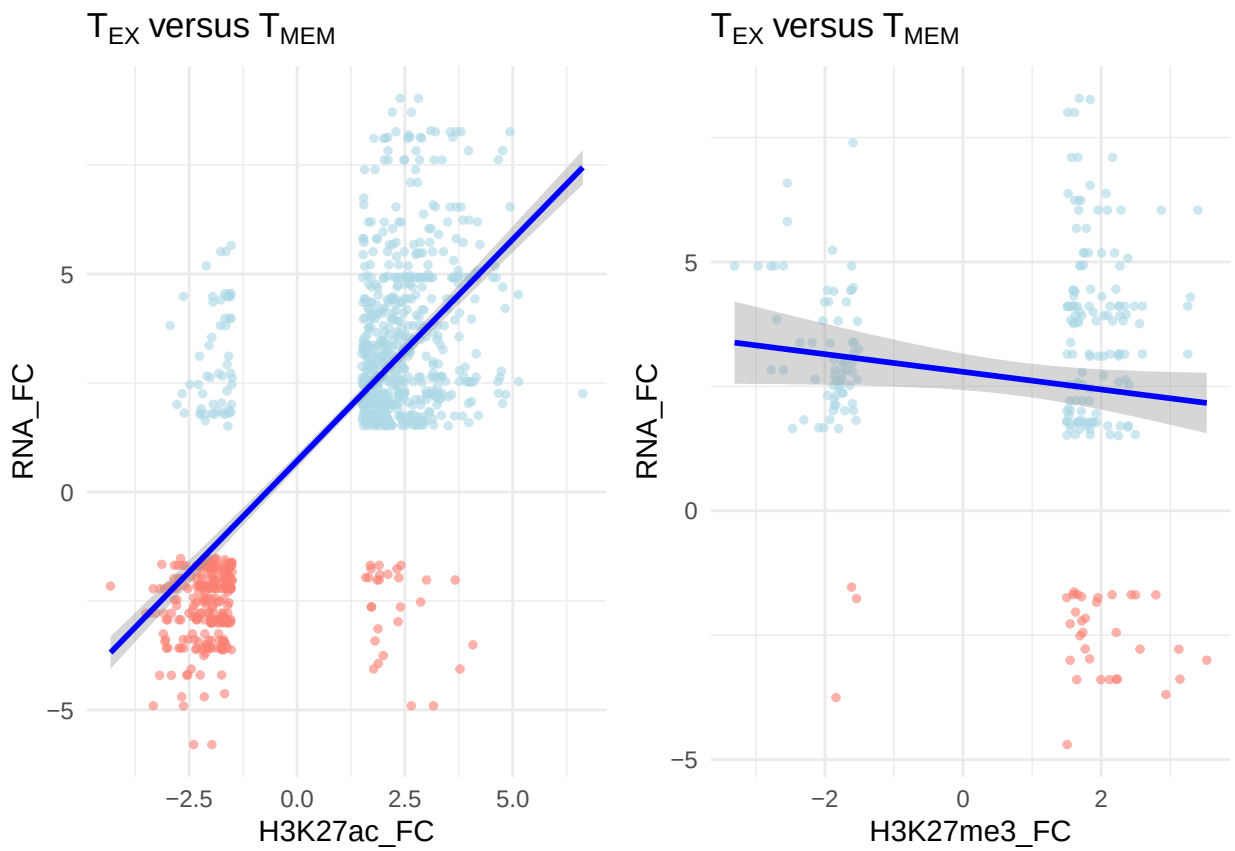
#Plots

```
# H3K27ac
joined_df.rna.H3K27ac <- ColorScheme(joined_df.rna.H3K27ac)
H3K27ac.corplot <- plotCorrelation(joined_df.rna.H3K27ac, "H3K27ac_FC", "RNA_FC", "color")

# H3K27me3
joined_df.rna.H3K27me3 <- ColorScheme(joined_df.rna.H3K27me3)
H3K27me3.corplot <- plotCorrelation(joined_df.rna.H3K27me3, "H3K27me3_FC", "RNA_FC", "color")

plot_grid(ncol = 2, H3K27ac.corplot,H3K27me3.corplot)
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```



From the figure 1 panel c only one them was reproducible.