Figure 4

Sergio Garcia-Moreno Alcantara (sga@stowers.org)

August 24, 2021

Aim

Analysis of SAGA occupancy at active core promoters in the Drosophila ovaries

Enviroment setup

Set working directory and load required libraries and lab functions

```
setwd("/n/projects/sga/analysis/SAGA/saga_publication/")
options(knitr.figure_dir = "plots/figure_4/")

# Standard packages
library(GenomicAlignments)
library(GenomicRanges)
library(Biostrings)
library(BSgenome.Dmelanogaster.UCSC.dm6)
library(TxDb.Dmelanogaster.UCSC.dm6.ensGene)
library(dplyr)
library(reshape2)
library(plyranges)
library(CAGEr)
library(magrittr)
library(ggplot2)
library(cowplot)
library(ggseqlogo)
library(gridExtra)
library(ggpubr)
library(DEGreport)
library(GGally)

# Lab sources
source("./shared_code/granges_common.r")
source("./shared_code/metapeak_common.r")
source("./shared_code/knitr_common.r")
```

Analysis

1. Loading samples and necessary data sets

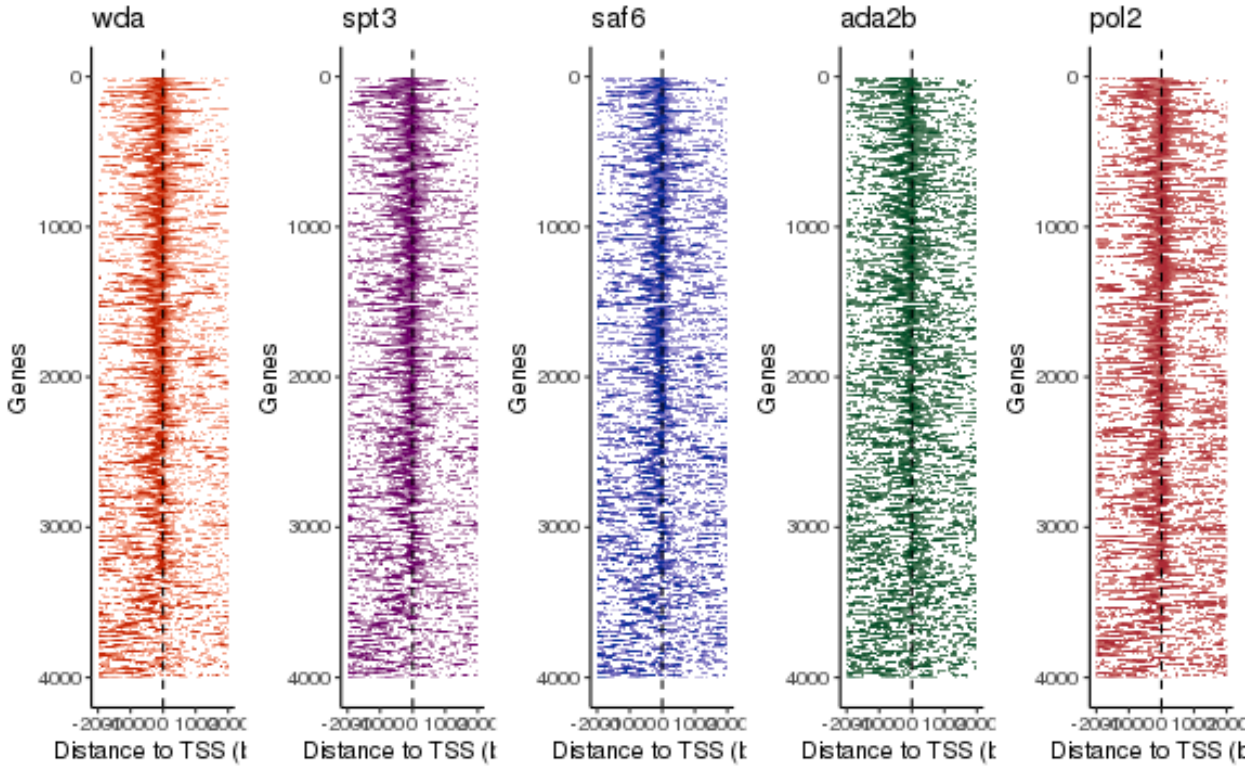
```
## Load sample list
sample_list <- read.csv("./chipseq_samples.csv", sep = ";")

## Define a function to pull samples from the sample list
load_bigwig <- function(sample_list) {

    bw_path <- function(path) {
        path = path
    }
    sample_list %>%
        mutate(list = purrr::map(as.character(path), bw_path)) %>%
        pull(list)
}

bw_list <- load_bigwig(sample_list)
names(bw_list) <- sample_list$short_name

## Load CAGE-seq data in ovaries
cage_gr <- get(load("./rdata/dm6_mrna_ovaries_tss.RData"))

## Load RNA-seq data in ovaries
rna_seq_df <- read.csv("./rdata/RSEM_TPM_table.csv")[, 1:5]
colnames(rna_seq_df)[c(1, 2)] <- c("fb_g_id", "gene_name")
rna_seq_df$rnaseq_tpm <- floor(rowMeans(rna_seq_df[, 3:5]))
rna_seq_df <- filter(rna_seq_df, rnaseq_tpm >= 3)  ## <- take genes with equal or more than 3 transcrip
rna_seq_df <- rna_seq_df[, c(1, 6)]

## Make a TSS object where there is active transcription based on both, RNA-seq
## and CAGE-seq data
tss <- merge(cage_gr, rna_seq_df, by = "fb_g_id") %>%
    makeGRangesFromDataFrame(., ignore.strand = F, keep.extra.columns = T)
```

2. Generate a occupancy heatmap for SAGA subunits at the top 4000 genes with the highest wda occupancy

```
tss$wda <- regionSums(resize(tss, 1001, "center"), bw_list$wda)
high_tss <- tss[order(tss$wda, decreasing = T)[1:4000]]

bp_signal_matrix_list <- mclapply(names(bw_list), function(x) {

    bw <- bw_list[[x]]
    bp_signal_matrix <- standard_metapeak_matrix(regions.gr = high_tss, sample.cov = bw,
        upstream = 2000, downstream = 2001)
    bp_signal_matrix

}, mc.cores = 5)
```

```
matrix_normalization <- function(matrix) {

    max.per.gene.pos <- apply(matrix, 1, function(x) {
        quantile(x, 0.95)
    })
    min.per.gene.pos <- apply(matrix, 1, function(x) {
        quantile(x, 0.5)
    })

    matrix.p <- matrix
    matrix.p[matrix.p <= min.per.gene.pos] <- NA  #Remove all values that are below 50th percentile
    matrix.p <- pmin(matrix.p/max.per.gene.pos, 1)  #Re-normalize values that remain.

    matrix.p

}

format_matrix <- function(matrix, sample_name) {

    df <- as.data.frame(matrix, sample_name)
    colnames(df) <- as.numeric(-2000:2000)
    df$fb_t_id <- 1:nrow(df)
    df$sample <- sample_name

    melted_df <- df %>%
        data.table %>%
        melt.data.table(df, id.vars = c("fb_t_id", "sample"), variable.name = "tss_distance",
            value.name = "signal", measure.vars = paste0(as.numeric(-2000:2000)))

    melted_df <- melted_df[!is.na(melted_df$signal), ]
    melted_df$tss_distance <- as.numeric(as.character(melted_df$tss_distance))
    melted_df

}

bp_signal_matrix_norm <- lapply(bp_signal_matrix_list, matrix_normalization)
names(bp_signal_matrix_norm) <- names(bw_list)

bp_signal_df <- lapply(names(bp_signal_matrix_norm), function(x) {

    mat <- bp_signal_matrix_norm[[x]]
    df <- format_matrix(mat, x)
    df

}) %>%
    bind_rows()

plot_heatmap <- function(df, col, name) {

    x <- ggplot(df, aes(tss_distance, fb_t_id, fill = signal)) + geom_raster(show.legend = F) +
        scale_fill_gradient(low = "white", high = col) + theme(legend.position = "none",
        axis.text.y = element_blank()) + geom_vline(xintercept = 0, linetype = 2) +
        ggtitle(name) + scale_y_reverse() + xlab("Distance to TSS (bp)") + ylab("Genes") +
```

```
        theme_classic()
    x
}

wda_htm <- plot_heatmap(filter(bp_signal_df, sample == "wda"), "#C62606", "wda")
saf6_htm <- plot_heatmap(filter(bp_signal_df, sample == "spt3"), "#6D0068", "spt3")
spt3_htm <- plot_heatmap(filter(bp_signal_df, sample == "saf6"), "#0B2E9B", "saf6")
ada2b_htm <- plot_heatmap(filter(bp_signal_df, sample == "ada2b"), "#044F2A", "ada2b")
pol2_htm <- plot_heatmap(filter(bp_signal_df, sample == "pol2"), "#AB2A35", "pol2")

plot_grid(wda_htm, saf6_htm, spt3_htm, ada2b_htm, pol2_htm, ncol = 5)
```

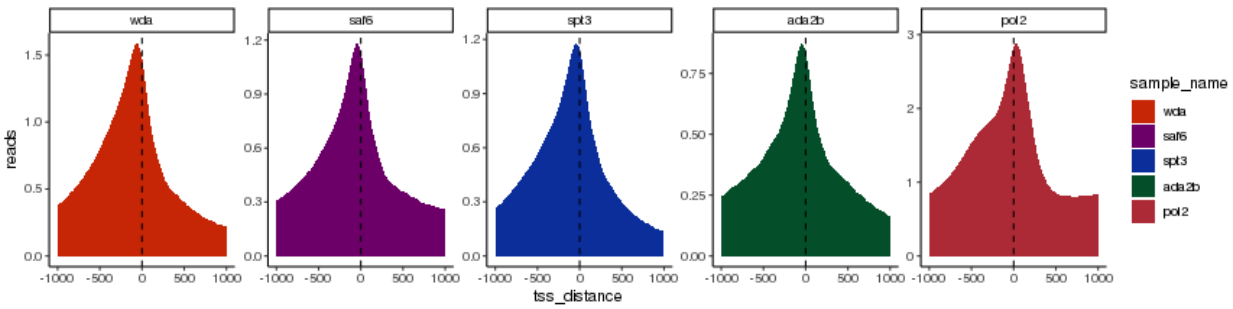


3. Generate a occupancy heatmap for SAGA subunits at the top 4000 genes with the highest expression

4. Plot metapeaks

```
metapeak_df <- mclapply(names(bw_list), function(x) {

    bw <- bw_list[[x]]
    df <- standard_metapeak(gr = high_tss, sample = bw, upstream = 1000, downstream = 1001,
        sample_name = x, smooth = NA)
    df

}, mc.cores = 5) %>%
    bind_rows()
```

```
metapeak_df$sample_name <- factor(metapeak_df$sample_name, levels = c("wda", "saf6",
    "spt3", "ada2b", "pol2"))

ggplot(metapeak_df, aes(tss_distance, reads, fill = sample_name)) + geom_area() +
    scale_fill_manual(values = c("#C62606", "#6D0068", "#0B2E9B", "#044F2A", "#AB2A35")) +
    geom_vline(xintercept = 0, linetype = 2) + facet_wrap(~sample_name, scales = "free",
    ncol = 5) + theme_classic()
```

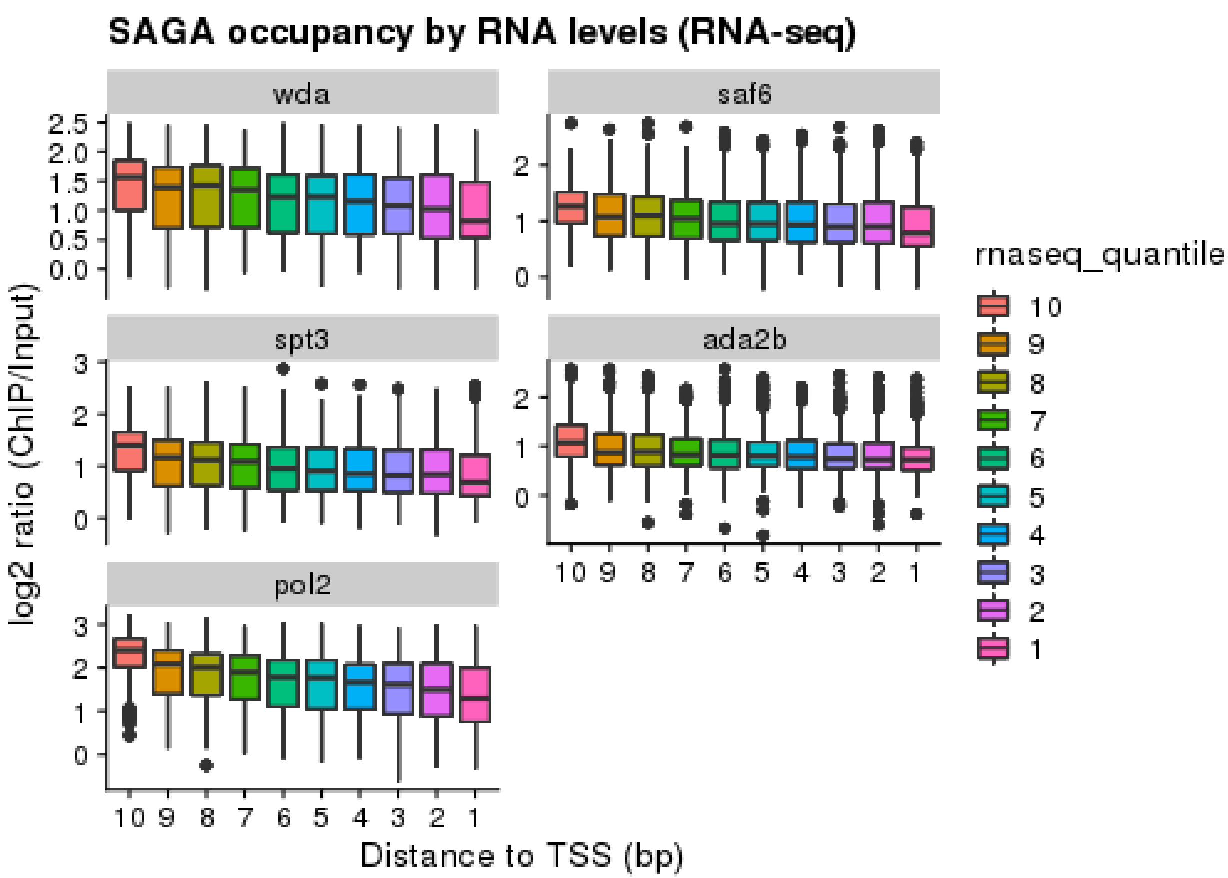


5. Plot the correlation between SAGA core subunits

6. Plot distribution of SAGA occupancy levels across quantiles of RNA-seq expression data

```
rna_tss <- tss[order(tss$rnaseq_tpm, decreasing = T)]
rna_tss$rnaseq_quantile <- ntile(rna_tss$rnaseq_tpm, 10)

# Make a data frame containing transcript ID and total signal per gene and
# promoter type

sig_df <- mclapply(levels(as.factor(rna_tss$rnaseq_quantile)), function(x) {

    quantile_gr <- subset(rna_tss, rnaseq_quantile == x)
    mclapply(names(bw_list), function(y) {
        bw <- bw_list[[y]]

        df <- data.frame(fb_t_id = quantile_gr$fb_t_id, signal = regionMaxs(resize(quantile_gr,
            501, "center"), bw), sample = y, rnaseq_quantile = x)
        df
    }, mc.cores = 5)
}, mc.cores = 5) %>%
    do.call(c, .) %>%
    bind_rows()

sig_df$rnaseq_quantile <- factor(sig_df$rnaseq_quantile, levels = c("10", "9", "8",
    "7", "6", "5", "4", "3", "2", "1"))
sig_df$sample <- factor(sig_df$sample, levels = c("wda", "saf6", "spt3", "ada2b",
    "pol2"))
# sig_df<-filter(sig_df,signal>=0)

boxplot <- ggplot(sig_df, aes(rnaseq_quantile, log2(signal + 1), fill = rnaseq_quantile)) +
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
```

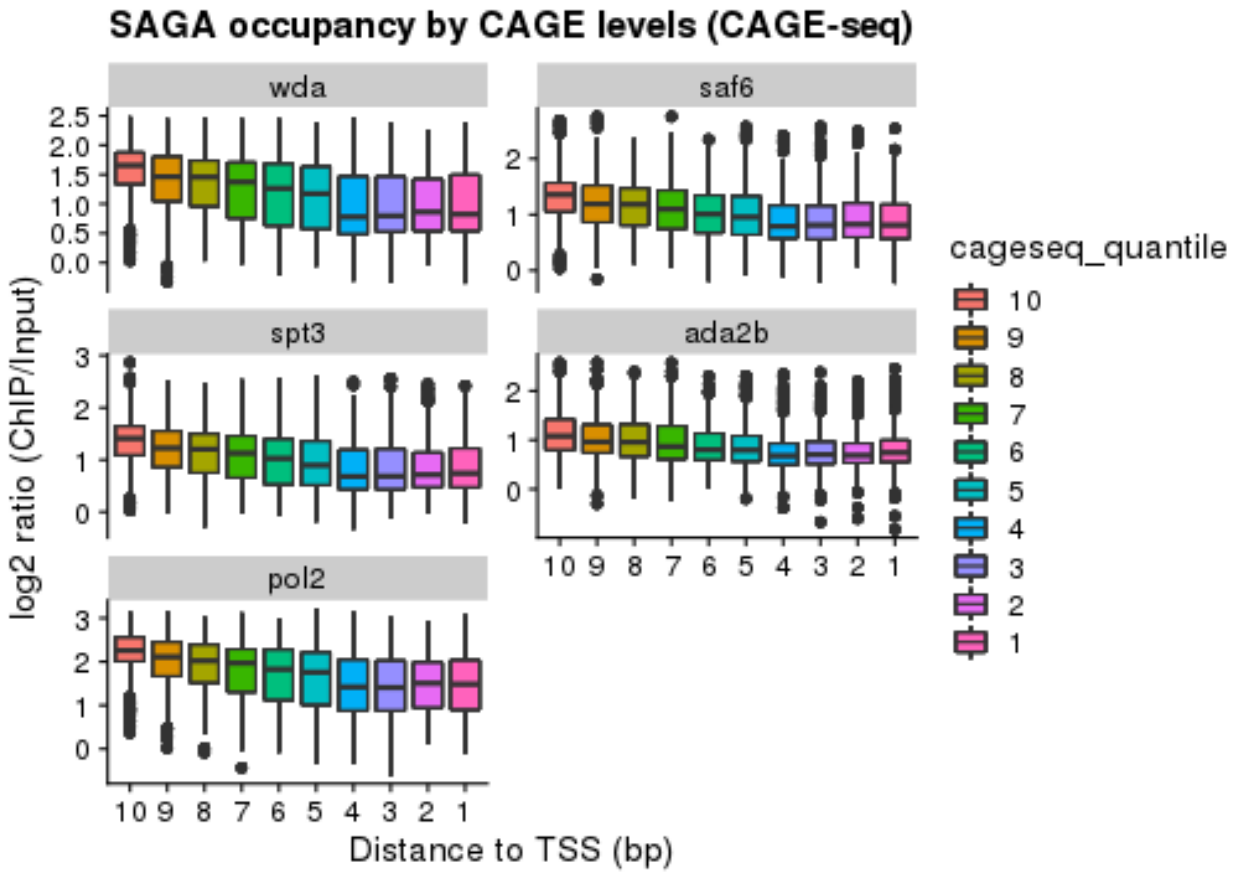
```
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
ggtitle("SAGA occupancy by RNA levels (RNA-seq)") + facet_wrap(~sample, scales = "free_y",
    ncol = 2) + theme(plot.title = element_text(size = 15, face = "bold")) + xlab("Distance to TSS (bp)
    ylab("log2 ratio (ChIP/Input)")

boxplot
```



7. Plot distribution of SAGA occupancy levels across quantiles of CAGE-seq expression data

```
cage_tss <- tss[order(tss$score, decreasing = T)]
cage_tss$cageseq_quantile <- ntile(cage_tss$score, 10)
```

```
# Make a data frame containing transcript ID and total signal per gene and
# promoter type

sig_df <- mclapply(levels(as.factor(cage_tss$cageseq_quantile)), function(x) {

    quantile_gr <- subset(cage_tss, cageseq_quantile == x)
    mclapply(names(bw_list), function(y) {
        bw <- bw_list[[y]]

        df <- data.frame(fb_t_id = quantile_gr$fb_t_id, signal = regionMaxs(resize(quantile_gr,
            501, "center"), bw), sample = y, cageseq_quantile = x)
        df
    }, mc.cores = 5)
}, mc.cores = 5) %>%
    do.call(c, .) %>%
    bind_rows()

sig_df$cageseq_quantile <- factor(sig_df$cageseq_quantile, levels = c("10", "9",
    "8", "7", "6", "5", "4", "3", "2", "1"))
sig_df$sample <- factor(sig_df$sample, levels = c("wda", "saf6", "spt3", "ada2b",
    "pol2"))
# sig_df<-filter(sig_df,signal>=0)

boxplot <- ggplot(sig_df, aes(cageseq_quantile, log2(signal + 1), fill = cageseq_quantile)) +
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
    geom_boxplot(alpha = 0.7) + theme_cowplot() + #scale_fill_manual(values = c('indianred3', '#EE962B'
ggtitle("SAGA occupancy by CAGE levels (CAGE-seq)") + facet_wrap(~sample, scales = "free_y",
    ncol = 2) + theme(plot.title = element_text(size = 15, face = "bold")) + xlab("Distance to TSS (bp)
    ylab("log2 ratio (ChIP/Input)")

boxplot
```



Session Info

```
sessionInfo()
```

```
## R version 4.1.0 (2021-05-18)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: CentOS Linux 7 (Core)
##
## Matrix products: default
## BLAS:   /n/apps/CentOS7/install/r-4.1.0/lib64/R/lib/libRblas.so
## LAPACK: /n/apps/CentOS7/install/r-4.1.0/lib64/R/lib/libRlapack.so
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats4    parallel  stats     graphics  grDevices utils     datasets
## [8] methods   base
```

```
##
## other attached packages:
##  [1] digest_0.6.27
##  [2] pander_0.6.3
##  [3] data.table_1.14.0
##  [4] lattice_0.20-44
##  [5] GGally_2.1.1
##  [6] DEGreport_1.28.0
##  [7] ggpubr_0.4.0
##  [8] gridExtra_2.3
##  [9] ggseqlogo_0.1
## [10] cowplot_1.1.1
## [11] ggplot2_3.3.3
## [12] magrittr_2.0.1
## [13] CAGEr_1.34.0
## [14] MultiAssayExperiment_1.18.0
## [15] plyranges_1.12.0
## [16] reshape2_1.4.4
## [17] dplyr_1.0.6
## [18] TxDb.Dmelanogaster.UCSC.dm6.ensGene_3.12.0
## [19] GenomicFeatures_1.44.0
## [20] AnnotationDbi_1.54.0
## [21] BSgenome.Dmelanogaster.UCSC.dm6_1.4.1
## [22] BSgenome_1.60.0
## [23] rtracklayer_1.52.0
## [24] GenomicAlignments_1.28.0
## [25] Rsamtools_2.8.0
## [26] Biostrings_2.60.1
## [27] XVector_0.32.0
## [28] SummarizedExperiment_1.22.0
## [29] Biobase_2.52.0
## [30] MatrixGenerics_1.4.0
## [31] matrixStats_0.59.0
## [32] GenomicRanges_1.44.0
## [33] GenomeInfoDb_1.28.0
## [34] IRanges_2.26.0
## [35] S4Vectors_0.30.0
## [36] BiocGenerics_0.38.0
##
## loaded via a namespace (and not attached):
##   [1] circlize_0.4.12             readxl_1.3.1
##   [3] backports_1.2.1             VGAM_1.1-5
##   [5] BiocFileCache_2.0.0         plyr_1.8.6
##   [7] ConsensusClusterPlus_1.56.0 splines_4.1.0
##   [9] operator.tools_1.6.3        BiocParallel_1.26.0
##  [11] foreach_1.5.1               htmltools_0.5.1.1
##  [13] fansi_0.5.0                 memoise_2.0.0
##  [15] cluster_2.1.2               doParallel_1.0.16
##  [17] openxlsx_4.2.3              limma_3.48.0
##  [19] annotate_1.70.0             ComplexHeatmap_2.8.0
##  [21] Nozzle.R1_1.1-1             formula.tools_1.7.1
##  [23] prettyunits_1.1.1           colorspace_2.0-1
##  [25] blob_1.2.1                  rappdirs_0.3.3
##  [27] ggrepel_0.9.1               haven_2.4.1
```

```
##  [29] xfun_0.23                 crayon_1.4.1
##  [31] RCurl_1.98-1.3            genefilter_1.74.0
##  [33] survival_3.2-11           iterators_1.0.13
##  [35] glue_1.4.2                gtable_0.3.0
##  [37] zlibbioc_1.38.0           GetoptLong_1.0.5
##  [39] DelayedArray_0.18.0       car_3.0-10
##  [41] shape_1.4.6               abind_1.4-5
##  [43] scales_1.1.1              DBI_1.1.1
##  [45] edgeR_3.34.0              som_0.3-5.1
##  [47] rstatix_0.7.0             Rcpp_1.0.6
##  [49] xtable_1.8-4              progress_1.2.2
##  [51] lasso2_1.2-21.1           tmvnsim_1.0-2
##  [53] clue_0.3-59               foreign_0.8-81
##  [55] bit_4.0.4                 httr_1.4.2
##  [57] RColorBrewer_1.1-2        ellipsis_0.3.2
##  [59] farver_2.1.0              pkgconfig_2.0.3
##  [61] reshape_0.8.8             XML_3.99-0.6
##  [63] dbplyr_2.1.1              locfit_1.5-9.4
##  [65] utf8_1.2.1                labeling_0.4.2
##  [67] tidyselect_1.1.1          rlang_0.4.11
##  [69] munsell_0.5.0             cellranger_1.1.0
##  [71] tools_4.1.0               cachem_1.0.5
##  [73] generics_0.1.0            RSQLite_2.2.7
##  [75] broom_0.7.6               ggdendro_0.1.22
##  [77] evaluate_0.14             stringr_1.4.0
##  [79] fastmap_1.1.0             yaml_2.2.1
##  [81] knitr_1.33                bit64_4.0.5
##  [83] zip_2.2.0                 beanplot_1.2
##  [85] purrr_0.3.4               KEGGREST_1.32.0
##  [87] nlme_3.1-152              formatR_1.11
##  [89] biomaRt_2.48.0            compiler_4.1.0
##  [91] filelock_1.0.2            curl_4.3.1
##  [93] png_0.1-7                 ggsignif_0.6.1
##  [95] geneplotter_1.70.0        tibble_3.1.2
##  [97] stringi_1.6.2             highr_0.9
##  [99] forcats_0.5.1             Matrix_1.3-4
## [101] psych_2.1.3               vegan_2.5-7
## [103] permute_0.9-5             vctrs_0.3.8
## [105] stringdist_0.9.6.3        pillar_1.6.1
## [107] lifecycle_1.0.0           GlobalOptions_0.1.2
## [109] bitops_1.0-7              R6_2.5.0
## [111] BiocIO_1.2.0              KernSmooth_2.23-20
## [113] rio_0.5.26                codetools_0.2-18
## [115] MASS_7.3-54               gtools_3.9.2
## [117] assertthat_0.2.1          DESeq2_1.32.0
## [119] rjson_0.2.20              withr_2.4.2
## [121] mnormt_2.0.2              GenomeInfoDbData_1.2.6
## [123] mgcv_1.8-36               hms_1.1.0
## [125] grid_4.1.0                tidyr_1.1.3
## [127] rmarkdown_2.8             carData_3.0-4
## [129] Cairo_1.5-12.2            logging_0.10-108
## [131] restfulr_0.0.13
```