Figure1

Riccardo Mosca

2025-05-12

This markdown show how to generate Figure 1B and Figure S1E-F

Figure 1B Loading packages and qPCR fold changes values

```
library(dplyr)
library(ggplot2)
library(ggpubr)
library(data.table)
```

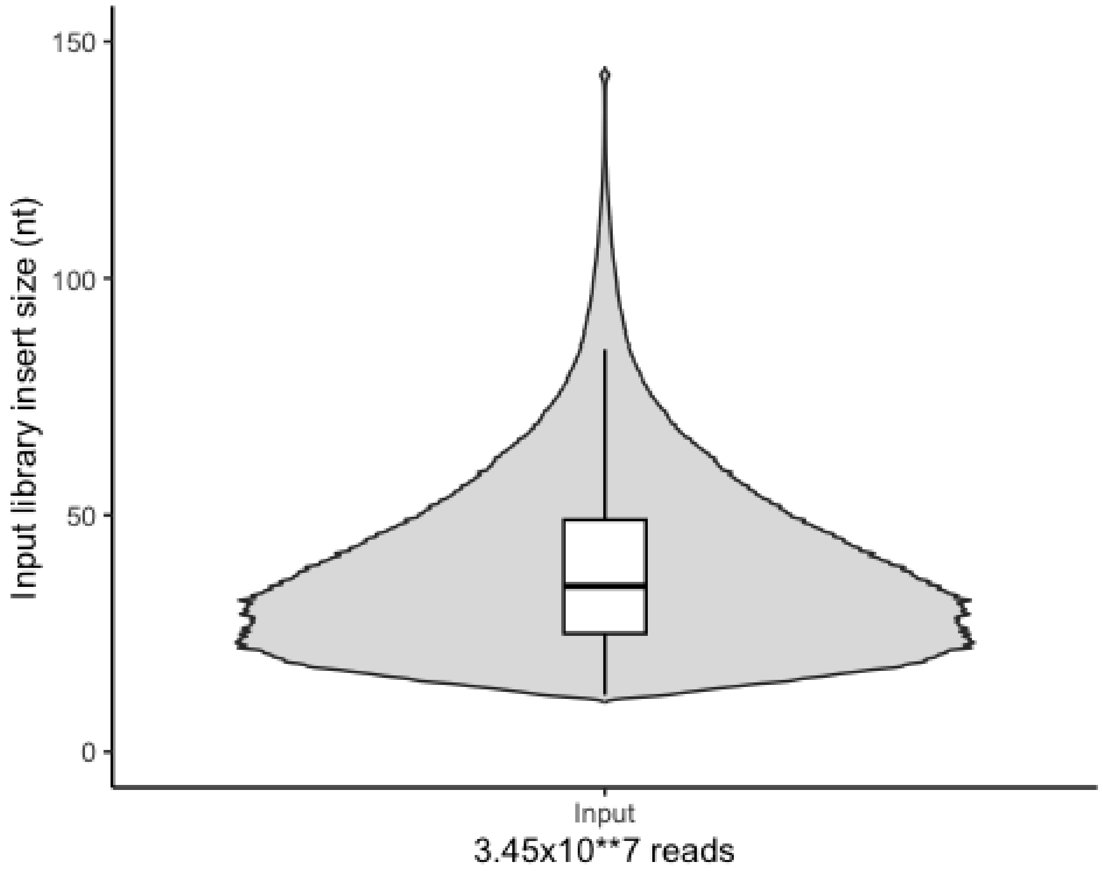
loading library size for fig S1A

```
read_length <- read.table(file =
"/Users/riccardomosca/Downloads/read_lengths.txt", header = F,
stringsAsFactors = F)
read_length$Name <- "Input"
colnames(read_length) <- c("length", "Name")
summary(read_length$length)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   12.00   25.00   35.00   39.43   49.00  143.00
```

plotting

```
p <- ggplot(read_length, aes(x=Name, y=length)) +
  geom_violin(trim=FALSE, fill="grey", alpha=0.5) +
  geom_boxplot(width=0.1, fill="white", color="black", outlier.shape=NA) +
  coord_cartesian(ylim=c(0,150)) +
  theme_classic() +
  ylab("Input library insert size (nt)") +
  xlab("3.45x10**7 reads")

p
```



```
ggsave("/Users/riccardomosca/Desktop/RAPseq_PAPER/FIGUREs/FIGURE1/NEW/FigS1A.
pdf", p, width = 3, height = 5)
```

Figure 1B Loading qPCR fold changes values

```
qPCR <- read.table(file =
"/Users/riccardomosca/Desktop/RAPseq_PAPER/FIGUREs/FIGURE1/NEW/HUR_qPCR",
header = T, stringsAsFactors = F)
qPCR_means <- qPCR %>% group_by(Substrate, Target) %>%
summarise(Mean_Fold_Change = mean(Fold_Change))

## `summarise()` has grouped output by 'Substrate'. You can override using
the
## `.groups` argument.

qPCR_means <- as.data.frame(qPCR_means)
head(qPCR_means)

##     Substrate Target Mean_Fold_Change
## 1 Full_Length     NT            2.385
## 2 Full_Length     T1            5.500
## 3 Full_Length     T3            0.000
## 4  fragmented     NT            1.210
## 5   fragmented      T1            47.500
## 6   fragmented      T3            81.000
```

plotting

```
Fig1B <- ggplot2::ggplot() + geom_bar(data = qPCR_means, aes(x = Target, y =
log2(Mean_Fold_Change +
    1), fill = Target), stat = "identity", color = "black", alpha = 0.9) +
geom_point(data = qPCR,
    aes(x = Target, y = log2(Fold_Change + 1), fill = Target), size = 3.5,
pch = 21,
    color = "black") + facet_wrap(~Substrate) + theme_pubr() +
theme(axis.text.x = element_text(angle = 90),
    legend.position = "none") + ylim(0, 7) + scale_fill_manual(values =
c("#C7E8BF",
    "#8FCF91", "#197B41"))

# pdf(file =
#
'/Users/riccardomosca/Desktop/RAPseq_PAPER/FIGUREs/FIGURE1/NEW/HUR_RAPqPCR_ba
rplot_Log2.pdf',2.5,5)
# dev.off()
Fig1B
```

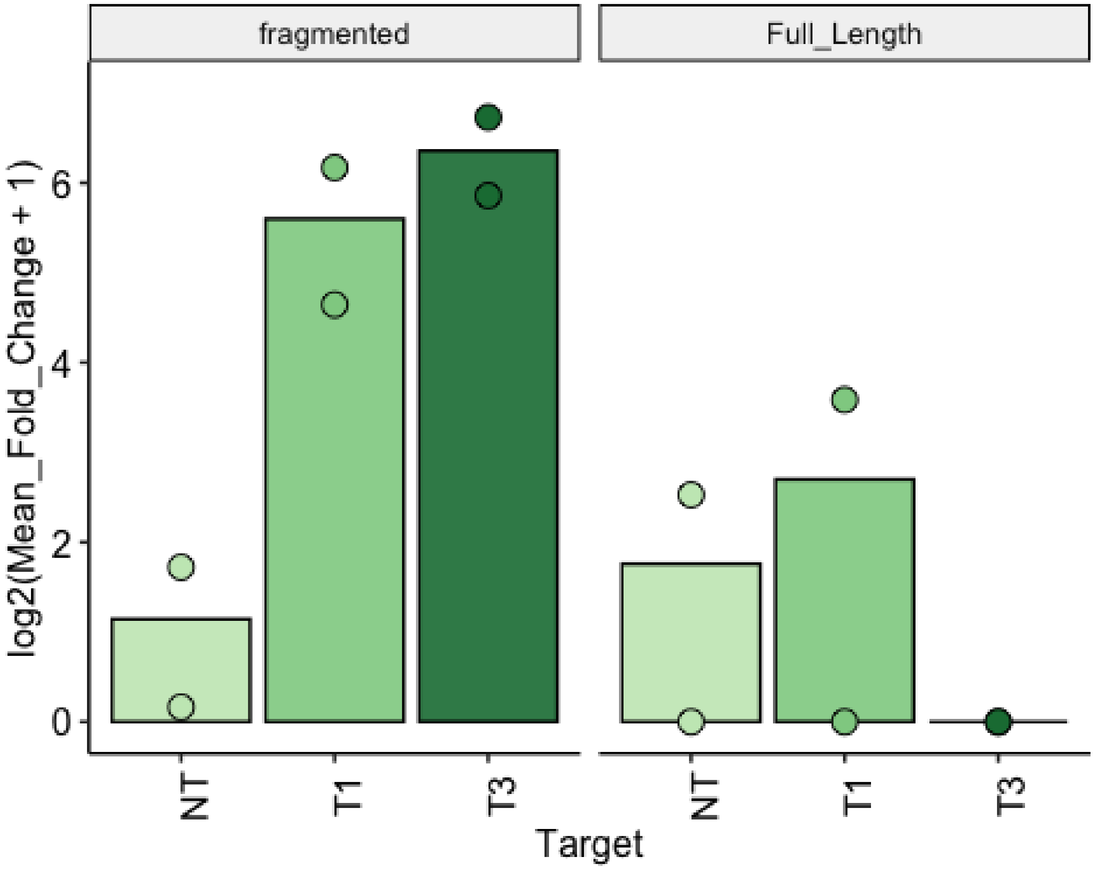
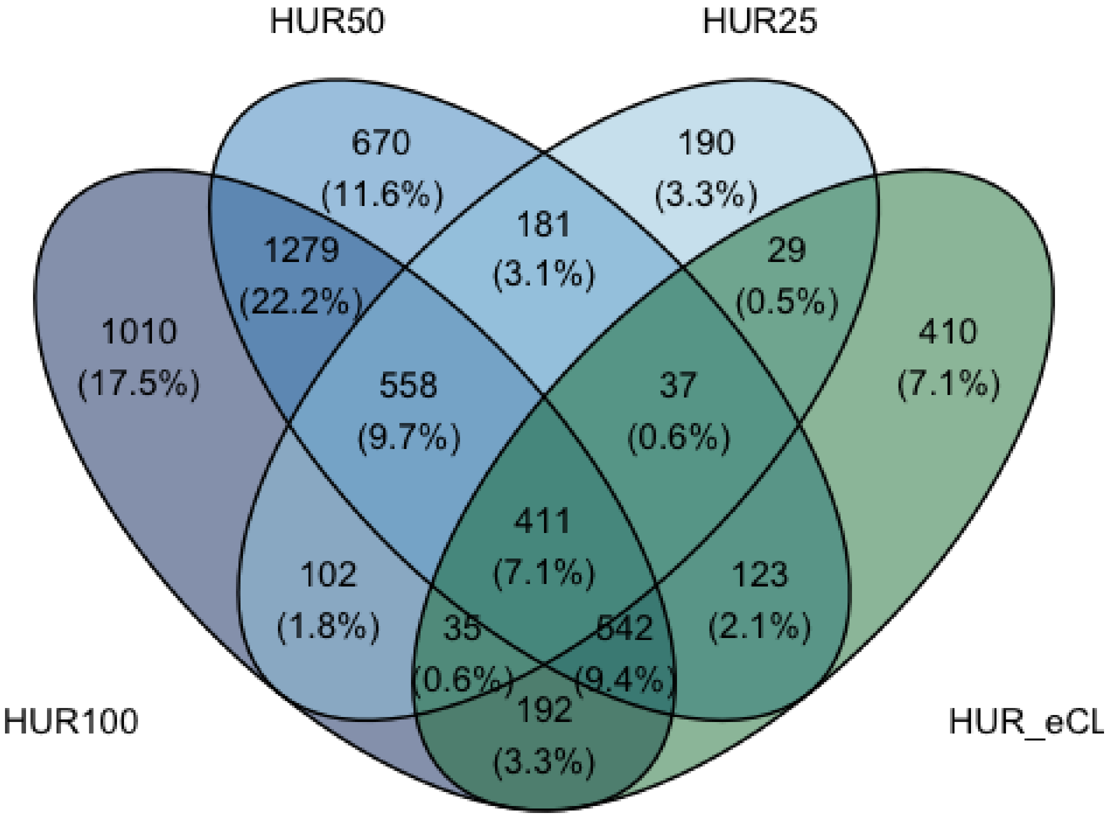


Figure S1E Loading packages and annotated peak files, keeping the unique targets

```
library(eulerr)
library(tidyverse)
library(clusterProfiler)
library(org.Hs.eg.db)
library(reshape2)
library(ggvenn)
library(dplyr)


path <-
"/Users/riccardomosca/Desktop/RAPseq_PAPER/FIGUREs/FIGURE1/NEW/HUR_hnRNPA1_ti
tration_revision_231016/DEEPER_SEQ/PEAKs"

# Different HUR titrations
types <- c("HUR100", "HUR50", "HUR25")

for (t in types) {
    # build the filename and read it in
    infile <- file.path(path, paste0(t, "_annotated.txt"))
    peaks <- read.delim(infile, sep = "\t", header = TRUE)

    # unique RNAs
    df <- peaks %>%
        dplyr::select(gene_name, Gene_BS, gene_ID, positive_fa, negative_fa)
%>%
        filter(!duplicated(gene_name)) %>%
        as.data.frame()

    # rename first column to 'gene'
    colnames(df)[1] <- "gene"

    assign(paste0("", t), df)
}

# Reading HUR eCLIP bed file from ENCODE
# https://www.encodeproject.org/experiments/ENCSR090LNQ/
eCLIP_HUR_peaks <-
read.delim("/Users/riccardomosca/Desktop/RAPseq_PAPER/FIGUREs/FIGURE1/NEW/HUR
_hnRNPA1_titration_revision_231016/PEAKs/HUR_eCLIP_annotated_peaks.txt",
    sep = "\t", header = T)
HUR_eCLIP <- eCLIP_HUR_peaks %>%
    dplyr::select(gene_name)
HUR_eCLIP <- HUR_eCLIP[!duplicated(HUR_eCLIP$gene_name), ]
HUR_eCLIP <- as.data.frame(HUR_eCLIP)
colnames(HUR_eCLIP)[1] <- "gene"
```

```
# Venn diagram
gene_HUR100 <- HUR100$gene
gene_HUR50 <- HUR50$gene
gene_HUR25 <- HUR25$gene
gene_HUR_eCLIP <- HUR_eCLIP$gene

ABCD <- list(HUR100 = gene_HUR100, HUR50 = gene_HUR50, HUR25 = gene_HUR25,
HUR_eCLIP = gene_HUR_eCLIP)

ggvenn(ABCD, fill_color = c("#09316A", "#4292C6", "#9ECAE0", "#187D41"),
stroke_size = 0.5,
    set_name_size = 4)
```



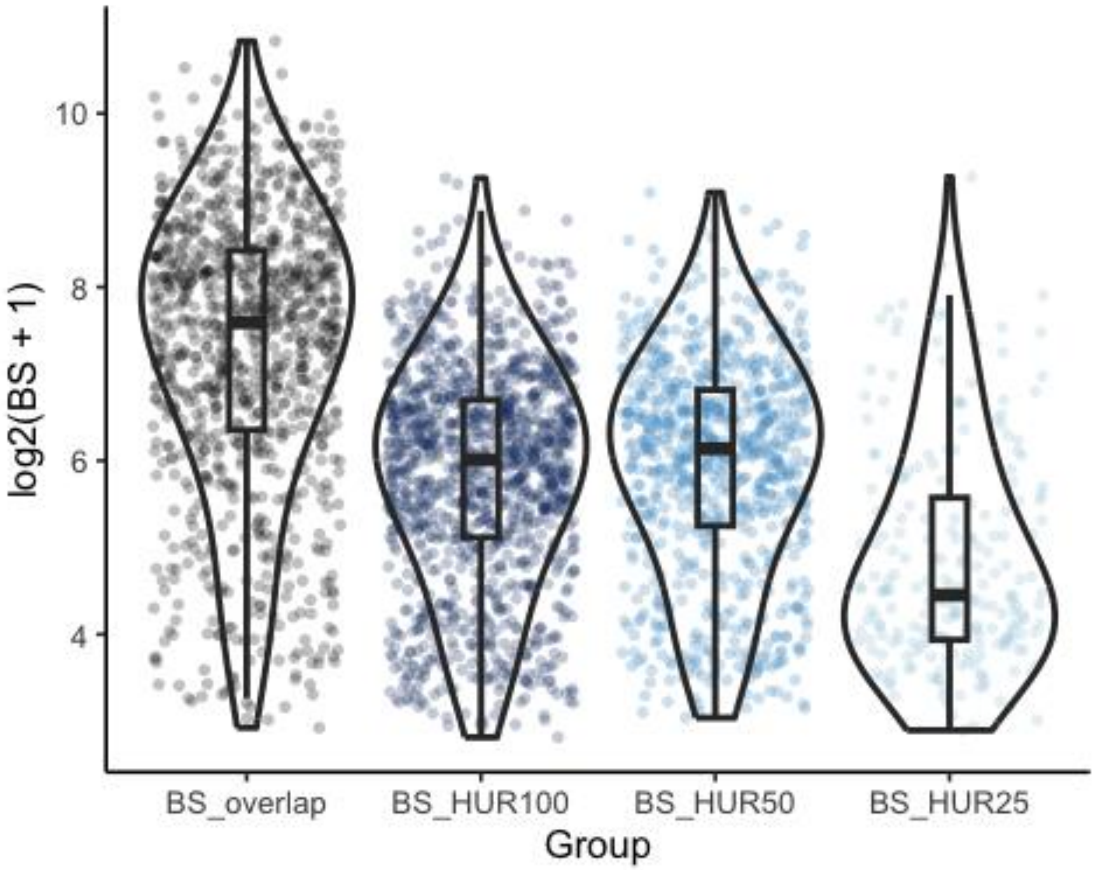
Computing the mean of Gene BS for the three titrations overlap, and each specific one.

```
overlap_HUR100_HUR50 <- subset(HUR100, gene %in% HUR50$gene)
overlap_HUR50_HUR25 <- subset(HUR50, gene %in% HUR25$gene)
overlap_HUR100_HUR50_HUR25 <- subset(overlap_HUR100_HUR50, gene %in%
overlap_HUR50_HUR25$gene)
specific_HUR100 <- HUR100[!as.character(HUR100$gene) %in%
as.character(HUR50$gene),
    ]
specific_HUR100 <- specific_HUR100[!as.character(specific_HUR100$gene) %in%
as.character(HUR25$gene),
    ]
specific_HUR50 <- HUR50[!as.character(HUR50$gene) %in%
as.character(HUR100$gene),
    ]
specific_HUR50 <- specific_HUR50[!as.character(specific_HUR50$gene) %in%
as.character(HUR25$gene),
    ]
specific_HUR25 <- HUR25[!as.character(HUR25$gene) %in%
as.character(HUR100$gene),
    ]
specific_HUR25 <- specific_HUR25[!as.character(specific_HUR25$gene) %in%
as.character(HUR50$gene),
    ]


# Determine the maximum length of the vectors to combine
max_length <- max(length(overlap_HUR100_HUR50_HUR25$Gene_BS),
length(specific_HUR100$Gene_BS),
    length(specific_HUR50$Gene_BS), length(specific_HUR25$Gene_BS))

# Create vectors of equal length by recycling shorter vectors
Gene_BS_HUR100_50_25 <- c(overlap_HUR100_HUR50_HUR25$Gene_BS, rep(NA,
max_length -
    length(overlap_HUR100_HUR50_HUR25$Gene_BS)))
Gene_BS_HUR100 <- c(specific_HUR100$Gene_BS, rep(NA, max_length -
length(specific_HUR100$Gene_BS)))
Gene_BS_HUR50 <- c(specific_HUR50$Gene_BS, rep(NA, max_length -
length(specific_HUR50$Gene_BS)))
Gene_BS_HUR25 <- c(specific_HUR25$Gene_BS, rep(NA, max_length -
length(specific_HUR25$Gene_BS)))

# Making an unique df
combined_BS <- data.frame(BS_overlap = Gene_BS_HUR100_50_25, BS_HUR100 =
Gene_BS_HUR100,
    BS_HUR50 = Gene_BS_HUR50, BS_HUR25 = Gene_BS_HUR25)
combined_BS <- melt(combined_BS, value.name = "BS", variable.name = "Group")
combined_BS <- na.omit(combined_BS)

head(combined_BS)

##        Group        BS
## 1 BS_overlap 252.89770
## 2 BS_overlap 424.43825
## 3 BS_overlap 623.91807
## 4 BS_overlap 674.67210
## 5 BS_overlap  52.20435
## 6 BS_overlap 109.50906
```

```
combined_BS %>%
    group_by(Group) %>%
    summarise(n = n())
```

```
## # A tibble: 4 × 2
##   Group          n
##   <fct>      <int>
## 1 BS_overlap   969
## 2 BS_HUR100   1202
## 3 BS_HUR50     793
## 4 BS_HUR25     219
```

```
aa <- ggplot(data = combined_BS, aes(x = Group, y = log2(BS + 1))) +
geom_jitter(aes(color = Group),
    pch = 16, alpha = 0.25) + scale_colour_manual(values = c("#000000",
"#0A316A",
    "#4292C6", "#9ECAE0")) + theme_classic(base_size = 12.5) +
theme(legend.position = "none")

FigS1E <- aa + geom_violin(trim = T, bw = 0.75, scale = "width", lwd = 1,
fill = NA) +
    geom_boxplot(outlier.shape = NA, width = 0.15, lwd = 1, fill = NA)

FigS1E
```



```
# collect specific Gene BS vectors into a named list
specific_list <- list(HUR100 = specific_HUR100$Gene_BS, HUR50 =
specific_HUR50$Gene_BS,
    HUR25 = specific_HUR25$Gene_BS)

# run wilcox.test() for each, comparing to the overlap vector
wt <- lapply(names(specific_list), function(nm) {
    y <- specific_list[[nm]]
    test <- wilcox.test(overlap_HUR100_HUR50_HUR25$Gene_BS, y)
    data.frame(comparison = paste0("overlap_vs_", nm), W =
as.numeric(test$statistic),
        p.value = test$p.value)
})

# combine into one df
wt_df <- do.call(rbind, wt)
```