

Figure 6 - Combined analysis of cRBP & ncRBP targets

Carlos Gallardo

16 May, 2025

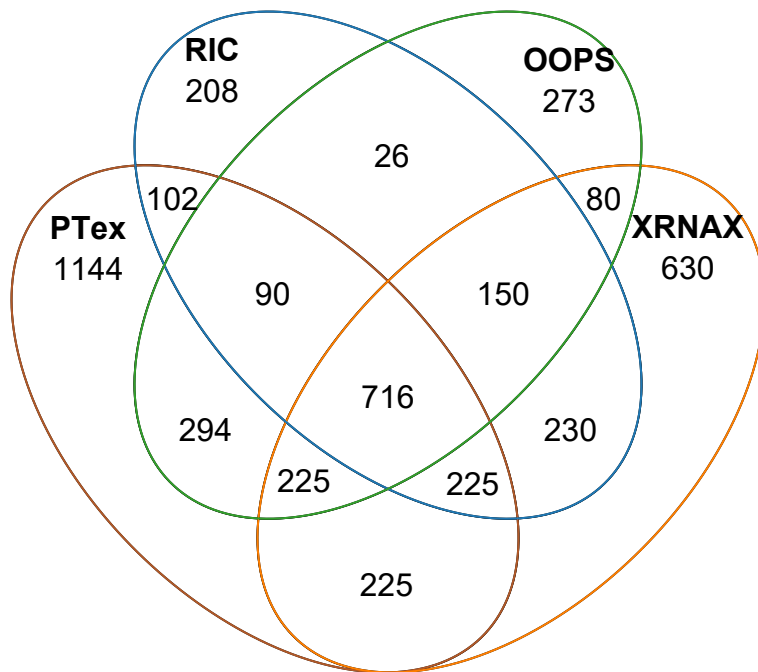
```
# source and library import
source('code/00_helper_functions.R')
library(tidyverse)
library(org.Hs.eg.db)
library(eulerr)
library(g3viz)
library(patchwork)
library(ComplexUpset)
library(clusterProfiler)
library(rrvgo)
library(ComplexHeatmap)
library(circlize)
library(fgsea)
library(ggrepel)
```

Figure S6A - RBPome overlaps

```
# RBPomes
RBPomes <- list(
  PTex = read.csv(file = "data/RBPomes/PTex.csv", stringsAsFactors = F, header = T, sep=";"),
  XRNAX = read.csv(file = "data/RBPomes/XRNAX_table.csv", stringsAsFactors = F, header = T, sep=";"),
  RIC = read.csv(file = "data/RBPomes/RIC_table.csv", stringsAsFactors = F, header = T, sep=";"),
  OOPS = read.csv(file = "data/RBPomes/OOPS.csv", stringsAsFactors = F, header = T, sep=";")
)

# extract RBPs
RBPomes$PTex <- RBPomes$PTex %>%
  pull(Gene.name) %>%
  unique()
RBPomes$XRNAX <- RBPomes$XRNAX %>%
  pull(Gene.name) %>%
  unique()
RBPomes$RIC <- RBPomes$RIC[rowSums(RBPomes$RIC[,seq(4,length(RBPomes$RIC))]>0,)] %>%
  pull(UNIQUE) %>%
  unique()
RBPomes$OOPS <- AnnotationDbi::select(org.Hs.eg.db, keys=RBPomes$OOPS$Uniprot_ID, columns = c('UNIPROT', 'SYMBOL'), keytype="UNIPROT") %>%
  pull(SYMBOL) %>%
  unique()

# plot venn diagram
plot(venn(RBPomes), fills = FALSE, edges=c("#B15928", "#FF7F00", "#1F78B4", "#33A02C"), quantities=TRUE)
```



```
# overlap of all 4 RBPome datasets
overlap <- table(unlist(RBPomes))[table(unlist(RBPomes))==4] %>%
  names()

# resources for canonical RBPs
RBDs <- read.table(file = "Data/RBPomes/RBDs_in_RBP_Census.txt", stringsAsFactors = F, header = T, sep="\t")
RBP_Census <- read.csv(file = "Data/RBPomes/Canonical_RBPs_Gerstberger_et_al.csv", stringsAsFactors = F, header = T, sep=";")
RBPDB <- read.table(file = "Data/RBPomes/RBPDB.utoronto.txt", stringsAsFactors = F, header = T, sep="\t")

pattern <- paste(RBDs$Pfam.RNA.binding.domains, collapse = "|")
RBP_Census <- RBP_Census[grepl(pattern, RBP_Census$domains.count.) | grepl("established", RBP_Census$supporting.evidence...pubmed.ID.),] %>%
  pull(gene.name) %>%
  unique()

RBPDB <- RBPDB$Annotation.ID %>%
  AnnotationDbi::select(org.Hs.eg.db, keys=., columns = c('ENSEMBL', 'SYMBOL'), keytype="ENSEMBL") %>%
  pull(SYMBOL) %>%
  unique()

# remove high confident RBPs with RBDs based on published annotations
ncRBPs_unfilt <- setdiff(overlap, unique(c(RBP_Census, RBPDB)))

ncRBPs_unfilt_ids <- AnnotationDbi::select(org.Hs.eg.db, keys=ncRBPs_unfilt, columns = c('SYMBOL', 'UNIPROT', 'GENENAME'), keytype="SYMBOL") %>%
  dplyr::rename(gene_name = SYMBOL, uniprot = UNIPROT, Description = GENENAME)

# extract Pfam domains and filter ncRBPs
ncRBPs <- lapply(ncRBPs_unfilt_ids$uniprot, uniprot2pfam) %>%
  bind_rows() %>%
  merge(ncRBPs_unfilt_ids, by="uniprot") %>%
  # dplyr::slice(grep(paste(unique(RBDs$Pfam.RNA.binding.domains), collapse = "|"), .$hmm.name, invert = T))
  filter(!grepl(paste(unique(.$gene_name[grepl(pattern, hmm.name)]), collapse = "|"), gene_name))

# manual curation to remove splicing factors, elongation factors, etc
ncRBPs <- ncRBPs %>%
  filter(!grepl(c("TCERG1|AQR|PRPF40A|PSIP1|GTF2F1|ERH|TCOF1|PRPF4B|TMA16|SCAF11|EIF3|ERH|NVL|RNH1|RPRD2|TCEA1|ZNF598"), gene_name)) %>%
  pull(gene_name) %>%
  unique()

# get canonical RBPs
cRBPs <- setdiff(overlap, ncRBPs)

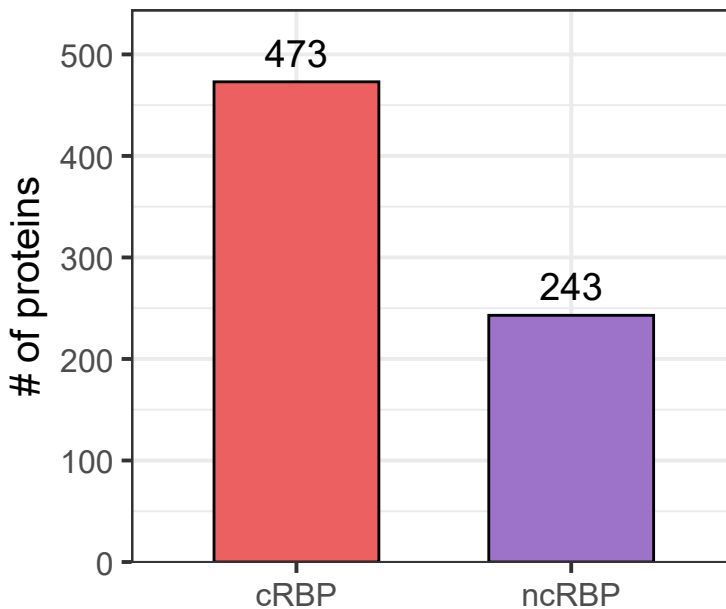
# plot cRBPs vs ncRBPs
df <- data.frame(
  Category = c("cRBP", "ncRBP"),
```

```

Count = c(length(cRBPs), length(ncRBPs))
)

ggplot(df, aes(x = Category, y = Count, fill = Category)) +
  geom_bar(stat = "identity", width = 0.6, color='black', show.legend = FALSE) +
  geom_text(aes(label = Count), vjust = -0.5, size = 5) +
  scale_fill_manual(values = c("cRBP" = "#ED6062", "ncRBP" = "#9C73C8")) +
  scale_y_continuous(expand = expansion(mult = c(.002, .15))) +
  labs(
    title = "",
    x = "",
    y = "# of proteins"
  ) +
  theme_bw(base_size = 15)

```



Load RAPseq peaks

```

all_RBP_peaks <- lapply(list.files("data/peaks/"), function(x) {
  read.table(paste("data/peaks/",x, sep=""), header = T, stringsAsFactors = F, sep="\t")
}) %>%
  bind_rows() %>%
  mutate(RBP = gsub("_final","",RBP)) %>%
  mutate(RBP = str_to_upper(RBP))

# split canonical and noncanonical RBP peaks
cRBP_peaks <- all_RBP_peaks %>%
  filter(str_detect(RBP, "RBF0X|HSHUR|PTBP|IGF2BP|HNRNP|YBX|IRP1|YTHDF1")) %>%
  filter(!str_detect(RBP, "HURPTBP1|IGF2BP1R|IGF2BP2B|IGF2BP3R|IGF2BP3I")) %>%
  mutate(RBP = str_replace(RBP, "HSHURHUMAN", "HUR")) %>%
  mutate(RBP = str_replace(RBP, "IGF2BP1WT", "IGF2BP1")) %>%
  mutate(RBP = str_replace(RBP, "IGF2BP3WT", "IGF2BP3"))

ncRBP_peaks <- all_RBP_peaks %>%
  filter(!str_detect(RBP, "RBF0X|HUR|PTBP|IGF2BP|HNRNP|YBX|IRP1|YTHDF1"))

RBP_peaks <- rbind(cRBP_peaks %>% mutate(RBP_type = 'cRBP'),
  ncRBP_peaks %>% mutate(RBP_type = 'ncRBP'))

```

Figure S6B - RBP binding sites

```
RBP_BS <- RBP_peaks %>%
  group_by(RBP) %>%
  summarise(n_sites = n(), RBP_type) %>%
  distinct() %>%
  arrange(desc(n_sites))

ggplot(RBP_BS, aes(x=RBP_type, y=n_sites, fill=RBP_type)) +
  geom_boxplot(color = "black", outlier.shape = NA) +
  scale_fill_manual(values = c("cRBP" = "#ED6062", "ncRBP" = "#9C73C8")) +
  labs(
    x = "",
    y = "# of Peaks"
  ) +
  ylim(0, 15000) +
  theme_classic(base_size = 14) +
  theme(
    axis.text.x = element_text(size = 10, angle = 90, hjust = 1, vjust = 0.5),
    plot.margin = margin(10, 10, 10, 10)
  )
```

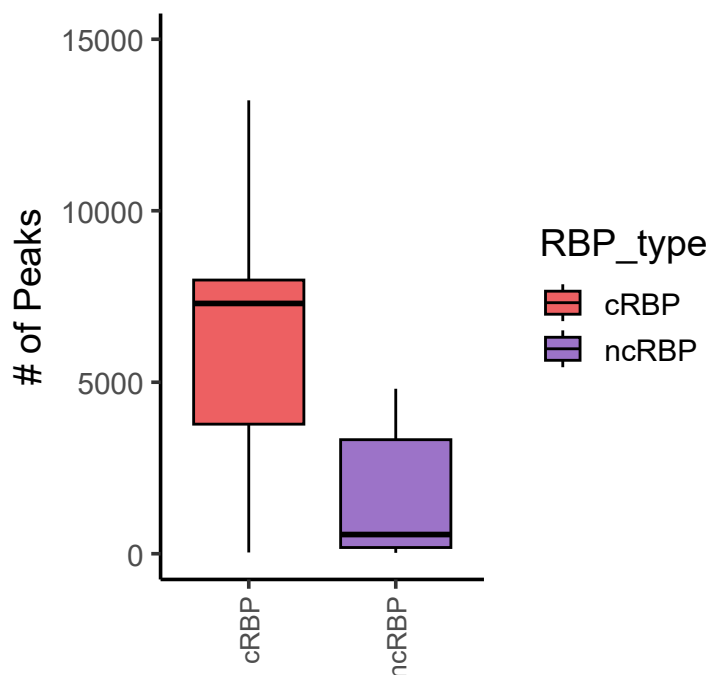


Figure S6C - RBP binding sites per RBP

```
# get binding sites for each RBP
RBP_BS <- RBP_peaks %>%
  group_by(RBP) %>%
  summarise(n_sites = n(), RBP_type) %>%
  distinct() %>%
  arrange(desc(n_sites))

## canonical RBP
cRBP_BS <- cRBP_peaks %>%
  group_by(RBP) %>%
  summarise(n_sites = n()) %>%
  arrange(desc(n_sites))

## noncanonical RBP
ncRBP_BS <- ncRBP_peaks %>%
  group_by(RBP) %>%
  summarise(n_sites = n()) %>%
  arrange(desc(n_sites))
```

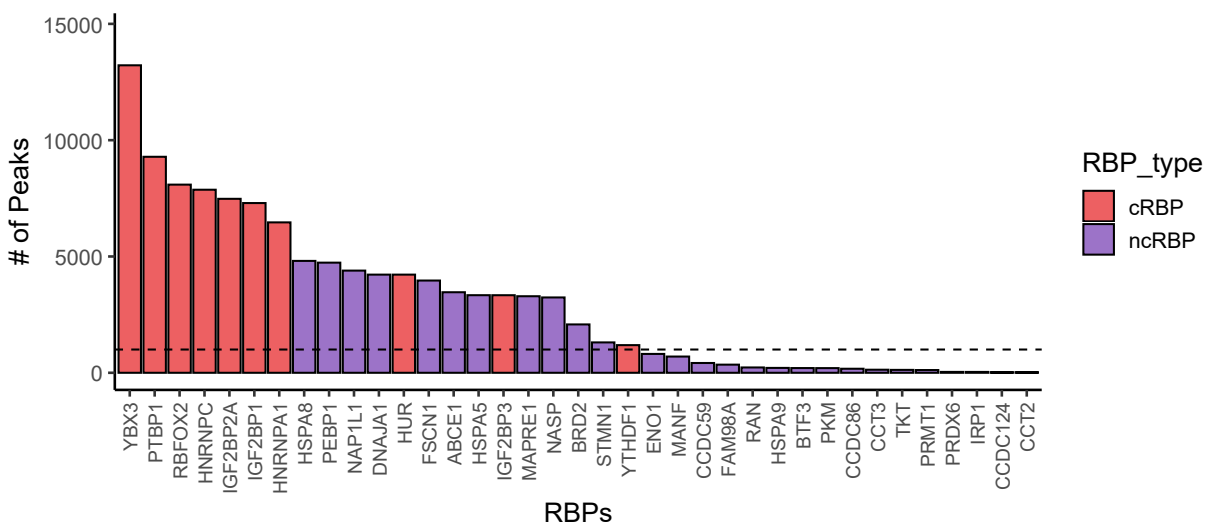
```

y_lim <- max(RBP_BS$n_sites)

p <- ggplot(RBP_BS, aes(x = reorder(RBP, -n_sites), y = n_sites, fill = RBP_type)) +
  geom_bar(stat = "identity", color = "black") +
  geom_hline(yintercept = 1000, linetype = "dashed", color = "black") +
  # geom_text(aes(label = n_sites),
  #           vjust = -0.5,
  #           size = 3) +
  scale_fill_manual(values = c("cRBP" = "#ED6062", "ncRBP" = "#9C73C8")) +
  labs(
    x = "RBPs",
    y = "# of Peaks"
  ) +
  ylim(0, y_lim + 1500) +
  theme_classic(base_size = 14) +
  theme(
    axis.text.x = element_text(size = 10, angle = 90, hjust = 1, vjust = 0.5),
    plot.margin = margin(10, 10, 10, 10)
  )

```

P



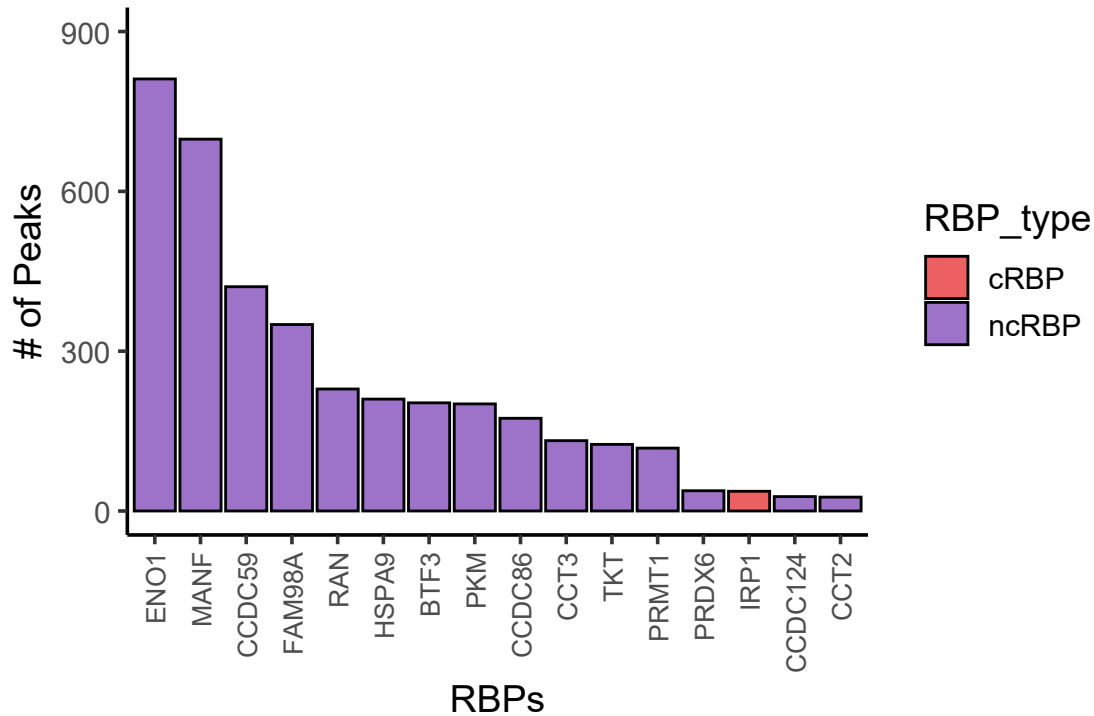
```

df <- RBP_BS %>%
  filter(n_sites < 1000)

p <- ggplot(df, aes(x = reorder(RBP, -n_sites), y = n_sites, fill = RBP_type)) +
  geom_bar(stat = "identity", color = "black") +
  scale_fill_manual(values = c("cRBP" = "#ED6062", "ncRBP" = "#9C73C8")) +
  labs(
    x = "RBPs",
    y = "# of Peaks"
  ) +
  scale_y_continuous(limits = c(0, 900), breaks = seq(0, 900, by = 300)) +
  theme_classic(base_size = 14) +
  theme(
    axis.text.x = element_text(size = 10, angle = 90, hjust = 1, vjust = 0.5),
    plot.margin = margin(10, 10, 10, 10)
  )

```

P



```
# filter RBPs with less than 1000 peaks
cRBP_names <- cRBP_BS %>%
  pull(RBP)
cRBP_filt_names <- cRBP_BS %>%
  filter(n_sites >= 1000) %>%
  pull(RBP)
cRBP_filt_peaks <- cRBP_peaks %>%
  filter(RBP %in% cRBP_filt_names)

ncRBP_names <- ncRBP_BS %>%
  pull(RBP)
ncRBP_filt_names <- ncRBP_BS %>%
  filter(n_sites >= 1000) %>%
  pull(RBP)
ncRBP_filt_peaks <- ncRBP_peaks %>%
  filter(RBP %in% ncRBP_filt_names)

RBP_filt_names <- c(cRBP_filt_names, ncRBP_filt_names)
RBP_filt_peaks <- rbind(cRBP_filt_peaks %>% mutate(RBP_type = 'cRBP'),
  ncRBP_filt_peaks %>% mutate(RBP_type = 'ncRBP'))
```

Figure 6A - RBP target distribution

```
# feature_colors <- brewer.pal(8, "Set3")
feature_colors <- c("intron" = "#08306B",
  "exon" = "#08519C",
  "CDS" = "#4292C6",
  "5UTR" = "#9ECAE1",
  "3UTR" = "#DEEBF7")

# get uniquely targeted genes
cRBP_filt_genes <- cRBP_filt_peaks %>%
  group_by(RBP) %>%
  summarise(n_genes = n_distinct(gene_name)) %>%
  arrange(desc(n_genes))

ncRBP_filt_genes <- ncRBP_filt_peaks %>%
  group_by(RBP) %>%
  summarise(n_genes = n_distinct(gene_name)) %>%
  arrange(desc(n_genes))

# get genomic feature distribution
```

```

cRBP_filt_features <- cRBP_filt_peaks %>%
  group_by(RBP, feature) %>%
  summarise(n = n()) %>%
  mutate(percentage = n / sum(n) * 100) %>%
  ungroup() %>%
  mutate(RBP = factor(RBP, levels = rev(cRBP_filt_genes$RBP)),
         feature = factor(feature, levels = rev(names(feature_colors))))

ncRBP_filt_features <- ncRBP_filt_peaks %>%
  group_by(RBP, feature) %>%
  summarise(n = n()) %>%
  mutate(percentage = n / sum(n) * 100) %>%
  ungroup() %>%
  mutate(RBP = factor(RBP, levels = rev(ncRBP_filt_genes$RBP)),
         feature = factor(feature, levels = rev(names(feature_colors))))

# get log2FC to Halo
cRBP_filt_FC <- cRBP_filt_peaks %>%
  mutate(log2_FC = log2(Mean_FCH)) %>%
  mutate(RBP = factor(RBP, levels = rev(cRBP_filt_genes$RBP)))

ncRBP_filt_FC <- ncRBP_filt_peaks %>%
  mutate(log2_FC = log2(Mean_FCH)) %>%
  mutate(RBP = factor(RBP, levels = rev(ncRBP_filt_genes$RBP)))

# define maximum x-axis limits
max_genes <- max(cRBP_filt_genes$n_genes, ncRBP_filt_genes$n_genes)
max_FC <- max(c(max(cRBP_filt_FC$log2_FC), max(ncRBP_filt_FC$log2_FC)))

# plot
plot_cRBP_genes <- ggplot(cRBP_filt_genes, aes(x = n_genes, y = reorder(RBP, n_genes))) +
  geom_bar(stat = "identity", fill = "#ED6062", width = 0.8, color = "black") +
  geom_text(aes(label = n_genes),
            hjust = -0.2,
            size = 3.5) +
  labs(
    x = NULL,
    y = "Canonical RBPs"
  ) +
  # xlim(0, max_genes + 1500) +
  scale_x_continuous(limits = c(0, max_genes + 1500), breaks = c(0, 3000, 6000)) +
  theme_classic(base_size = 12) +
  theme(
    axis.text.y = element_text(size = 10),
    axis.text.x = element_blank(),
    axis.ticks.x = element_blank(),
    plot.margin = margin(10, 10, 10, 10)
  )
plot_ncRBP_genes <- ggplot(ncRBP_filt_genes, aes(x = n_genes, y = reorder(RBP, n_genes))) +
  geom_bar(stat = "identity", fill = "#9C73C8", width = 0.8, color = "black") +
  geom_text(aes(label = n_genes),
            hjust = -0.2,
            size = 3.5) +
  labs(
    x = "# of Target Genes",
    y = "Noncanonical RBPs"
  ) +
  # xlim(0, max_genes + 1500) +
  scale_x_continuous(limits = c(0, max_genes + 1500), breaks = c(0, 3000, 6000)) +
  theme_classic(base_size = 12) +
  theme(
    axis.text.y = element_text(size = 10),
    plot.margin = margin(10, 10, 10, 10)
  )
plot_cRBP_features <- ggplot(cRBP_filt_features,
  aes(x = percentage, y = RBP, fill = feature)) +
  geom_bar(stat = "identity", position = "fill", width = 0.8, color = "black") +
  scale_fill_manual(values = feature_colors) +
  labs(
    x = NULL,
    y = NULL,
    fill = "Feature"
  ) +
  theme_classic(base_size = 12) +
  theme(
    legend.position = "none",
    axis.text.y = element_blank(),
    axis.text.x = element_blank(),
    axis.ticks.y = element_blank(),

```

```

    axis.ticks.x = element_blank(),
    plot.margin = margin(10, 10, 10, 10)
  )
plot_ncRBP_features <- ggplot(ncRBP_filt_features,
                             aes(x = percentage, y = RBP, fill = feature)) +
  geom_bar(stat = "identity", position = "fill", width = 0.8, color="black") +
  scale_fill_manual(values = feature_colors) +
  labs(
    x = "Feature Distribution (%)",
    y = NULL,
    fill = "Feature"
  ) +
  theme_classic(base_size = 12) +
  theme(
    legend.position="none",
    axis.text.y = element_blank(),
    axis.ticks.y = element_blank(),
    # axis.ticks.x = element_blank(),
    plot.margin = margin(10, 10, 10, 10)
  )
plot_cRBP_FC <- ggplot(cRBP_filt_FC, aes(x = log2_FC, y = RBP)) +
  geom_boxplot(fill = "#ED6062", color = "black", outlier.shape = NA) +
  labs(
    x = NULL,
    y = NULL
  ) +
  xlim(0, max_FC) +
  scale_x_continuous(breaks = c(0, 4, 8)) +
  theme_classic(base_size = 12) +
  theme(
    axis.text.y = element_blank(),
    axis.text.x = element_blank(),
    axis.ticks.x = element_blank(),
    plot.margin = margin(10, 10, 10, 10)
  )
plot_ncRBP_FC <- ggplot(ncRBP_filt_FC, aes(x = log2_FC, y = RBP)) +
  geom_boxplot(fill = "#9C73C8", color = "black", outlier.shape = NA) +
  labs(
    x = "log2(FC to Halo)",
    y = NULL
  ) +
  xlim(0, max_FC) +
  scale_x_continuous(breaks = c(0, 4, 8)) +
  theme_classic(base_size = 12) +
  theme(
    axis.text.y = element_blank(),
    axis.text.x = element_text(size = 10),
    plot.margin = margin(10, 10, 10, 10)
  )
)

combined_plot <- (plot_cRBP_genes | plot_cRBP_features | plot_cRBP_FC) / (plot_ncRBP_genes | plot_ncRBP_features | plot_ncRBP_FC)
print(combined_plot)

```

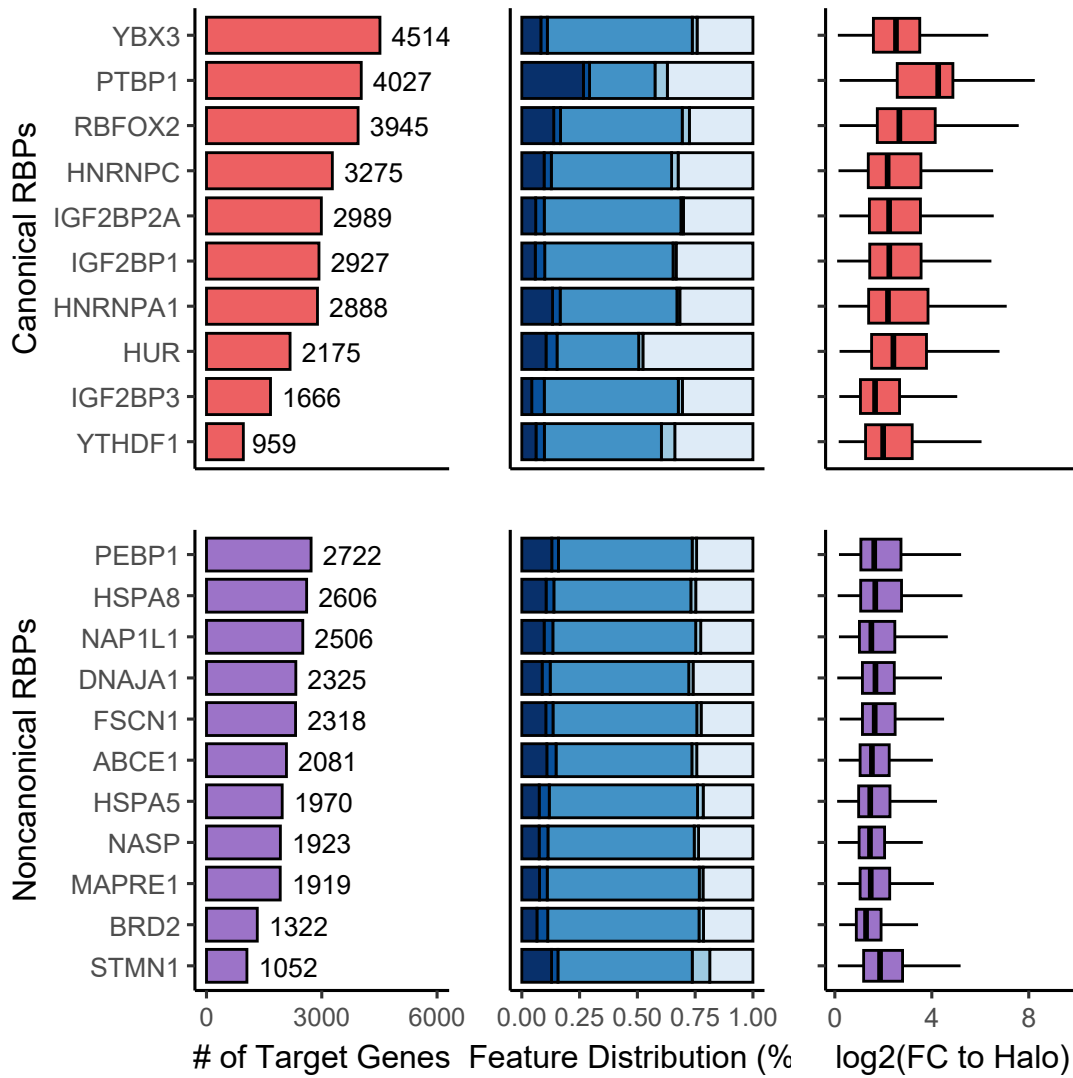



Figure S6D - RBP binding sites per gene

```
RBP_BS_per_gene_mean <- RBP_peaks %>%
  group_by(RBP, gene_name) %>%
  summarise(n_sites = n(), RBP_type) %>%
  distinct() %>%
  group_by(RBP) %>%
  summarise(mean_sites_per_gene = mean(n_sites), RBP_type) %>%
  distinct() %>%
  arrange(desc(mean_sites_per_gene))

p <- ggplot(RBP_BS_per_gene_mean, aes(x=RBP_type, y=mean_sites_per_gene, fill=RBP_type)) +
  geom_boxplot(color = "black", outlier.shape = NA) +
  scale_fill_manual(values = c("cRBP" = "#ED6062", "ncRBP" = "#9C73C8")) +
  labs(
    x = "",
    y = "Mean # of Peaks per Gene"
  ) +
  ylim(0, 3) +
  theme_classic(base_size = 14) +
  theme(
    axis.text.x = element_text(size = 10, angle = 90, hjust = 1, vjust = 0.5),
    plot.margin = margin(10, 10, 10, 10)
  )
```

p

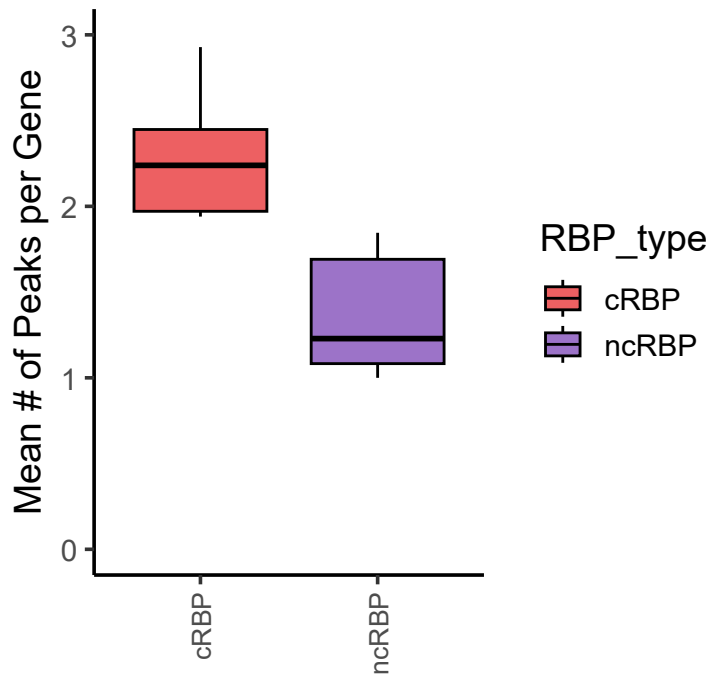
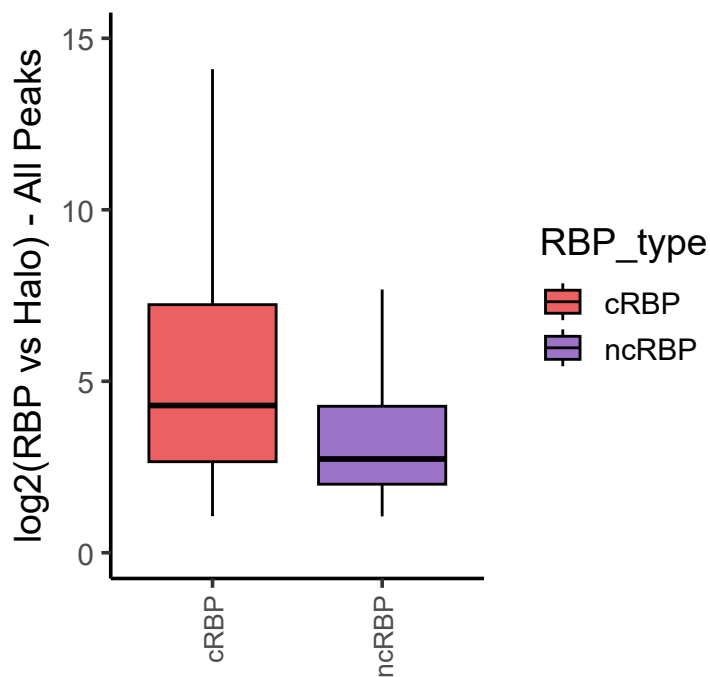


Figure S6E - RBP binding strength (downsampled)

```
RBP_FCH <- RBP_peaks

p <- ggplot(RBP_FCH, aes(x=RBP_type, y=Mean_FCH, fill=RBP_type)) +
  geom_boxplot(color = "black", outlier.shape = NA) +
  scale_fill_manual(values = c("cRBP" = "#ED6062", "ncRBP" = "#9C73C8")) +
  labs(
    x = "",
    y = "log2(RBP vs Halo) - All Peaks"
  ) +
  ylim(0, 15) +
  theme_classic(base_size = 14) +
  theme(
    axis.text.x = element_text(size = 10, angle = 90, hjust = 1, vjust = 0.5),
    plot.margin = margin(10, 10, 10, 10)
  )
```

p



```
RBP_FCH <- RBP_peaks %>%
  group_by(RBP_type) %>%
  sample_n(42601)

p <- ggplot(RBP_FCH, aes(x=RBP_type, y=Mean_FCH, fill=RBP_type)) +
  geom_boxplot(color = "black", outlier.shape = NA) +
  scale_fill_manual(values = c("cRBP" = "#ED6062", "ncRBP" = "#9C73C8")) +
  labs(
    x = "",
    y = "log2(RBP vs Halo) - Downsampled"
  ) +
  ylim(0, 15) +
  theme_classic(base_size = 14) +
  theme(
    axis.text.x = element_text(size = 10, angle = 90, hjust = 1, vjust = 0.5),
    plot.margin = margin(10, 10, 10, 10)
  )
```

p

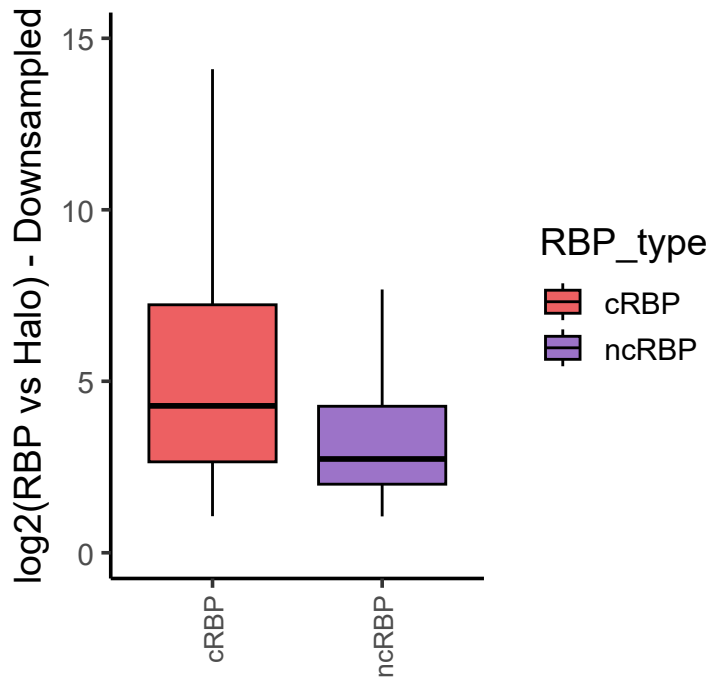


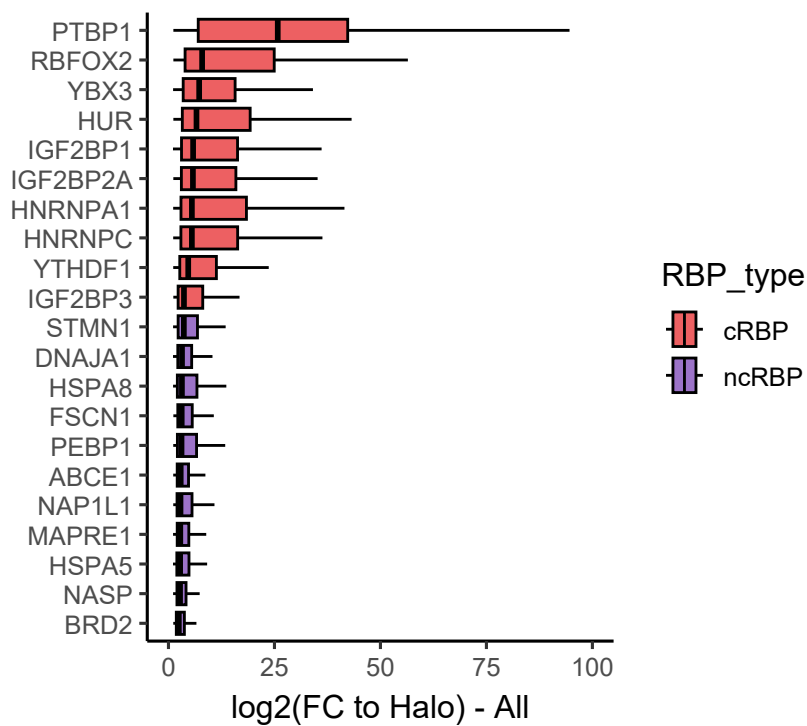
Figure S6F - Binding strength per RBP (downsampled)

```
RBP_filt_sorted_median_FCH <- RBP_filt_peaks %>%
  group_by(RBP) %>%
  summarise(median_FCH = median(Mean_FCH)) %>%
  arrange(desc(median_FCH)) %>%
  pull(RBP)

df <- RBP_filt_peaks %>%
  mutate(RBP = factor(RBP, levels = rev(RBP_filt_sorted_median_FCH)))

p <- ggplot(df, aes(x = Mean_FCH, y = RBP, fill = RBP_type)) +
  geom_boxplot(color = "black", outlier.shape = NA) +
  scale_fill_manual(values = c("cRBP" = "#ED6062", "ncRBP" = "#9C73C8")) +
  labs(
    x = "log2(FC to Halo) - All",
    y = NULL
  ) +
  xlim(0, 100) +
  theme_classic(base_size = 12) +
  theme(
    plot.margin = margin(10, 10, 10, 10)
  )
```

P



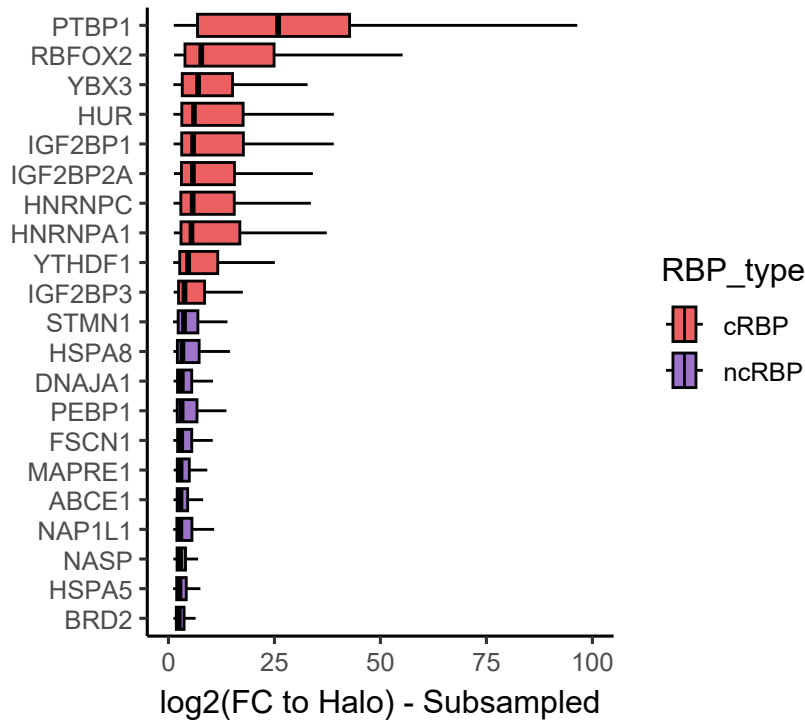
```
df <- RBP_filt_peaks %>%
  group_by(RBP) %>%
  sample_n(1000)

new_order <- df %>%
  group_by(RBP) %>%
  summarise(median_FCH = median(Mean_FCH)) %>%
  arrange(desc(median_FCH)) %>%
  pull(RBP)

df <- df %>%
  mutate(RBP = factor(RBP, levels = rev(new_order)))

p <- ggplot(df, aes(x = Mean_FCH, y = RBP, fill = RBP_type)) +
  geom_boxplot(color = "black", outlier.shape = NA) +
  scale_fill_manual(values = c("cRBP" = "#ED6062", "ncRBP" = "#9C73C8")) +
  labs(
    x = "log2(FC to Halo) - Subsampled",
    y = NULL
  ) +
  xlim(0, 100) +
  theme_classic(base_size = 12) +
  theme(
    plot.margin = margin(10, 10, 10, 10)
  )
```

p



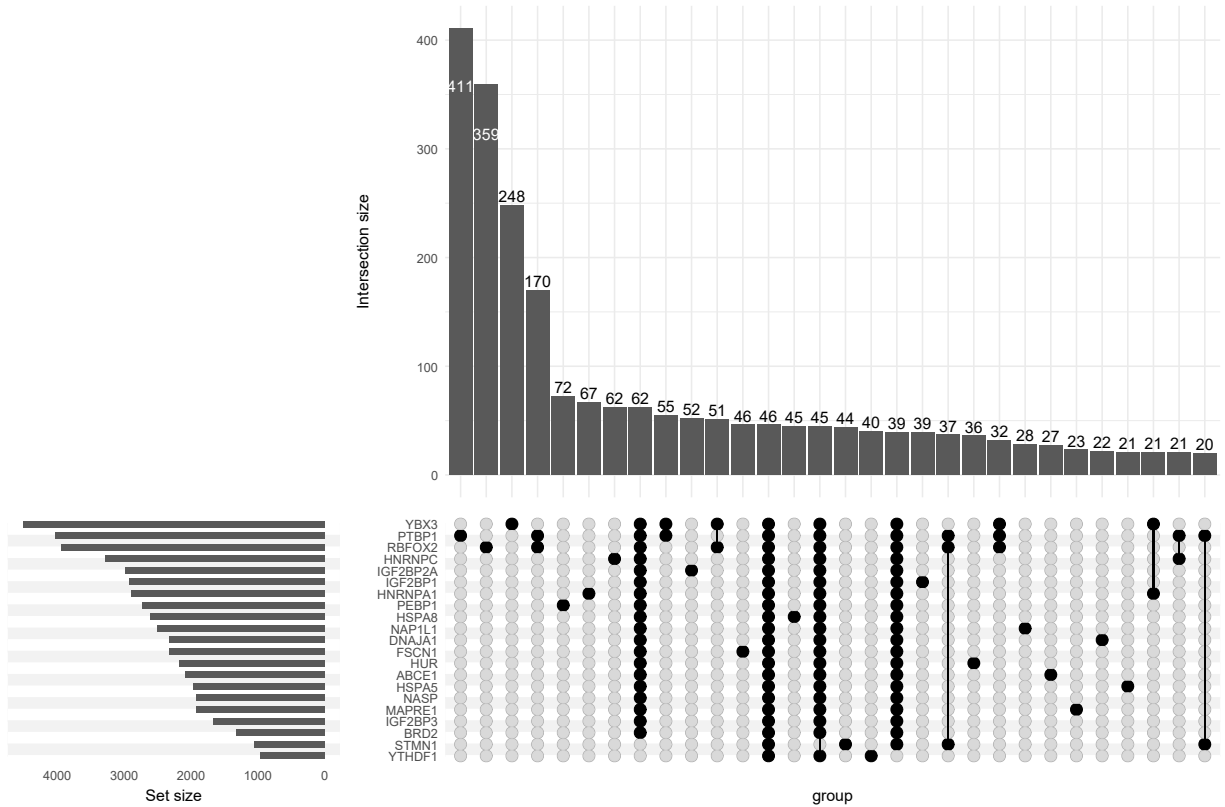
Overlap analysis of RBP targets

```
RBP_targets <- lapply(setNames(RBP_filt_names,RBP_filt_names), function(x){
  RBP_filt_peaks %>%
  filter(RBP == x) %>%
  pull(gene_name) %>%
  unique()
})

RBP_targets_all <- RBP_targets %>%
  unlist() %>%
  unique()

RBP_targets_overlap <- lapply(setNames(RBP_filt_names,RBP_filt_names), function(x){
  overlap <- c(x = RBP_targets_all %in% RBP_targets[[x]])
}) %>%
  bind_cols(.) %>%
  mutate(target = RBP_targets_all) %>%
  column_to_rownames(var = "target") %>%
  mutate(n_overlaps = rowSums(.),
         RBP_binders = apply(., 1, function(x) paste0(names(x)[x], collapse = '_'))))

# plotting intersections with at least 20 gene targets
upset(RBP_targets_overlap,
      RBP_filt_names,
      min_size = 20)
```



Generate RBP target network

```
RBP_targets_overlap_more20 <- RBP_targets_overlap$RBP_binders %>%
  table(.) %>%
  sort(decreasing = T) %>%
  as.data.frame() %>%
  filter(Freq>=20) %>%
  pull('.') %>%
  as.character()

RBP_targets_overlap_more20 <- setNames(RBP_targets_overlap_more20,RBP_targets_overlap_more20)

RBP_targets_overlap_more20_sets <- lapply(RBP_targets_overlap_more20, function(x){
  RBP_targets_overlap %>%
    rownames_to_column(var = "target") %>%
    filter(RBP_binders == x) %>%
    pull(target)
})

# establish nodes and edges of RBP target overlaps network
nodes <- data.frame(node_description = RBP_filt_names,
  node_type = "RBP",
  node_size = 100,
  id = RBP_filt_names) %>%
  bind_rows(data.frame(node_description = names(RBP_targets_overlap_more20_sets),
    node_type = "RBP_overlap",
    node_size = lapply(RBP_targets_overlap_more20_sets, length) %>% unlist()) %>%
    mutate(id = paste(node_description,node_size, sep="_"))
  )

edges <- nodes %>%
  filter(node_type == "RBP_overlap") %>%
  mutate(source = id) %>%
  add_column(target = .$node_description, .after = 'source') %>%
  separate_rows(target, sep="_") %>%
  mutate(interaction = 'link') %>%
  dplyr::select(source, interaction, target)
```

```
# export node and edge tables for Cytoscape
write.table(nodes, file = "results/RBP_target_network_nodes.txt", sep="\t", row.names = F, col.names = T, quote = F)
write.table(edges, file = "results/RBP_target_network_edges.txt", sep="\t", row.names = F, col.names = T, quote = F)
```

Figure 6D - Analysis of RBP target network subsets

GO enrichment analysis of RBP target subsets

```
gene_targets_subsets <- RBP_targets_overlap_more20_sets[c("PTBP1", "RBFOX2", "YBX3", "STMN1", "PTBP1_RBFOX2", "YBX3_PTBP1", "YBX3_RBFOX2", "PTBP1_
input <- read.table(file="data/INPUTs.TPMs.txt", stringsAsFactors = F, header=T)
input_genes <- input %>%
  filter(TPM_RAP >= 1) %>%
  pull(gene_ID) %>%
  gsub("\\..*", "", .) %>%
  unique() %>%
  mapIds(x=org.Hs.eg.db, keys=., column='SYMBOL', keytype = 'ENSEMBL') %>%
  unique()
RBP_bound_genes <- RBP_filt_peaks %>%
  pull(gene_name) %>%
  unique()
gene_background <- unique(c(input_genes, RBP_bound_genes))

# GO enrichment
gene_targets_subsets_GO <- lapply(setNames(names(gene_targets_subsets), names(gene_targets_subsets)), function(x) {
  res <- enrichGO(gene = gene_targets_subsets[[x]],
    universe = gene_background,
    keyType = "SYMBOL",
    OrgDb = org.Hs.eg.db,
    ont = "ALL",
    pAdjustMethod = "BH",
    pvalueCutoff = 0.1,
    qvalueCutoff = 0.1,
    minGSSize = 10,
    maxGSSize = 100,
    readable = TRUE)
  res@result <- res@result %>%
    add_column(subset_name = x, .before = 1)
  res
})
gene_targets_subsets_GO_combined <- lapply(gene_targets_subsets_GO, function(x) x@result) %>%
  bind_rows()

# split by BP, MF and CC
gene_targets_subsets_GO_combined_BP <- gene_targets_subsets_GO_combined %>%
  filter(ONTOLOGY == "BP")
gene_targets_subsets_GO_combined_MF <- gene_targets_subsets_GO_combined %>%
  filter(ONTOLOGY == "MF")
gene_targets_subsets_GO_combined_CC <- gene_targets_subsets_GO_combined %>%
  filter(ONTOLOGY == "CC")

# collapse GO terms
## BP
sim_mtx <- calculateSimMatrix(unique(gene_targets_subsets_GO_combined_BP$ID),
  orgdb="org.Hs.eg.db",
  ont="BP",
  method="Rel")
sim_mtx_reduced <- reduceSimMatrix(sim_mtx,
  threshold=0.9,
  orgdb="org.Hs.eg.db")
gene_targets_subsets_GO_combined_BP <- gene_targets_subsets_GO_combined_BP %>%
  left_join(sim_mtx_reduced, by=c("ID"="go")) %>%
  dplyr::filter(!is.na(parent))
gene_targets_subsets_GO_combined_BP_reduced <- gene_targets_subsets_GO_combined_BP %>%
  dplyr::group_by(parentTerm) %>%
  dplyr::summarise(ONTOLOGY, geneID = paste0(geneID, collapse="/")) %>%
  dplyr::mutate(geneID = unlist(lapply(geneID, function(x) paste0(unique(unlist(strsplit(x, split="/"))), collapse="/" )))) %>%
  dplyr::filter(!duplicated(parentTerm))

## MF
sim_mtx <- calculateSimMatrix(unique(gene_targets_subsets_GO_combined_MF$ID),
  orgdb="org.Hs.eg.db",
  ont="MF",
  method="Rel")
sim_mtx_reduced <- reduceSimMatrix(sim_mtx,
  threshold=0.9,
```



```

                                orgdb="org.Hs.eg.db")
gene_targets_subsets_GO_combined_MF <- gene_targets_subsets_GO_combined_MF %>%
  left_join(sim_mtx_reduced, by=c("ID"="go")) %>%
  dplyr::filter(!is.na(parent))
gene_targets_subsets_GO_combined_MF_reduced <- gene_targets_subsets_GO_combined_MF %>%
  dplyr::group_by(parentTerm) %>%
  dplyr::summarise(ONTOLOGY, geneID = paste0(geneID, collapse="/")) %>%
  dplyr::mutate(geneID = unlist(lapply(geneID, function(x) paste0(unique(unlist(strsplit(x, split="/"))), collapse="/" )))) %>%
  dplyr::filter(!duplicated(parentTerm))

## CC
sim_mtx <- calculateSimMatrix(unique(gene_targets_subsets_GO_combined_CC$ID),
                              orgdb="org.Hs.eg.db",
                              ont="CC",
                              method="Rel")
sim_mtx_reduced <- reduceSimMatrix(sim_mtx,
                                    threshold=0.9,
                                    orgdb="org.Hs.eg.db")
gene_targets_subsets_GO_combined_CC <- gene_targets_subsets_GO_combined_CC %>%
  left_join(sim_mtx_reduced, by=c("ID"="go")) %>%
  dplyr::filter(!is.na(parent))
gene_targets_subsets_GO_combined_CC_reduced <- gene_targets_subsets_GO_combined_CC %>%
  dplyr::group_by(parentTerm) %>%
  dplyr::summarise(ONTOLOGY, geneID = paste0(geneID, collapse="/")) %>%
  dplyr::mutate(geneID = unlist(lapply(geneID, function(x) paste0(unique(unlist(strsplit(x, split="/"))), collapse="/" )))) %>%
  dplyr::filter(!duplicated(parentTerm))

# merge collapsed enriched GO terms
gene_targets_subsets_GO_combined_reduced <- bind_rows(gene_targets_subsets_GO_combined_BP_reduced,
                                                       gene_targets_subsets_GO_combined_MF_reduced,
                                                       gene_targets_subsets_GO_combined_CC_reduced) %>%
  dplyr::mutate(ONTOLOGY = factor(ONTOLOGY, levels = c("BP", "MF", "CC"))) %>%
  dplyr::arrange(ONTOLOGY, parentTerm, .locale = "en")

```

Binding strength of RBP targets

```

# binding strength of RBPs per gene target
GO_enriched_genes <- gene_targets_subsets_GO_combined_reduced %>%
  pull(geneID) %>%
  str_split("\\/") %>%
  unlist() %>%
  unique()

df <- RBP_filt_peaks %>%
  filter(gene_name %in% GO_enriched_genes) %>%
  select(RBP, gene_name, Gene_BS) %>%
  filter(!duplicated(.)) %>%
  pivot_wider(names_from = RBP, values_from = Gene_BS, values_fill = 0.1) %>%
  mutate(gene_name = factor(gene_name, levels=GO_enriched_genes)) %>%
  arrange(gene_name) %>%
  column_to_rownames(var = "gene_name") %>%
  select("YBX3", "PTBP1", "RBF2", "STMN1")

p <- list()

p[[1]] <- Heatmap(log2(df+1),
  name = "Gene_BS",
  col = circlize::colorRamp2(c(0, max((log2(df))))), c("#BEBEBE", "#000000")),
  cluster_rows = F,
  cluster_columns = F,
  row_names_side = "left",
  column_names_side = "bottom",
  width = unit(ncol(df)*4, "mm"),
  height = unit(nrow(df)*4, "mm"),
  cell_fun = function(j, i, x, y, width, height, fill) {
    grid.rect(x, y, width, height,
      gp = gpar(col = "white", fill=NA, lwd = 1))
  },
  border = T)

```

Expression of RBP target genes in liver cancer stages

```

# median expression of gene targets in liver cancer stages (TCGA cohort)
TCGA_annot <- read.table(file="data/TCGA/Human_TCGA_LIHC_MS_Clinical_Clinical_01-28-2016_BI_Clinical_Firehose.tsi.txt",
  stringsAsFactors = F,
  sep = "\t",

```

```

      header=T) %>%
pivot_longer(cols = -attrib_name, names_to = "patient_id", values_to = "value") %>%
pivot_wider(names_from = attrib_name, values_from = value) %>%
dplyr::filter(!is.na(pathologic_stage)) %>%
mutate(pathologic_stage = factor(pathologic_stage, levels = c("stagei", "stageii", "stageiii", "stageiv"))) %>%
arrange(pathologic_stage)

TCGA_RNAseq <- read.table(file="data/TCGA/Human_TCGA_LIHC_UNC_RNAseq_HiSeq_RNA_01_28_2016_BI_Gene_Firehose_RSEM_log2.cct",
      stringsAsFactors = F,
      sep = "\t",
      header=T) %>%
column_to_rownames("attrib_name") %>%
dplyr::select(TCGA_annot$patient_id[TCGA_annot$patient_id %in% colnames(.)])

TCGA_RNAseq_scaled_median_by_stage <- groupTransform(TCGA_RNAseq,
      TCGA_annot$pathologic_stage[TCGA_annot$patient_id %in% colnames(TCGA_RNAseq)],
      function(x) apply(x, 1, median)) %>%
scaleData(., method = "zscore") %>%
rownames_to_column(var = "geneID")

GO_enriched_genes <- gene_targets_subsets_GO_combined_reduced %>%
pull(geneID) %>%
str_split("\\/") %>%
unlist() %>%
unique()

df <- data.frame(geneID = GO_enriched_genes) %>%
left_join(TCGA_RNAseq_scaled_median_by_stage, by=c("geneID"="geneID")) %>%
column_to_rownames("geneID")

divergent_pal <- c( "#2166AC", "#4393C3", "#92C5DE", "#D1E5F0", "#F7F7F7", "#FDDBC7", "#F4A582", "#D6604D", "#B2182B")

p[[2]] <- Heatmap(df,
      name = "HCC_expr",
      col = divergent_pal,
      cluster_rows = F,
      cluster_columns = F,
      row_names_side = "left",
      column_names_side = "bottom",
      width = unit(ncol(df)*4, "mm"),
      height = unit(nrow(df)*4, "mm"),
      cell_fun = function(j, i, x, y, width, height, fill) {
        grid.rect(x, y, width, height,
          gp = gpar(col = "white", fill=NA,lwd = 1))
      },
      border = T)

```

Presence of gene targets in enriched GO terms

```

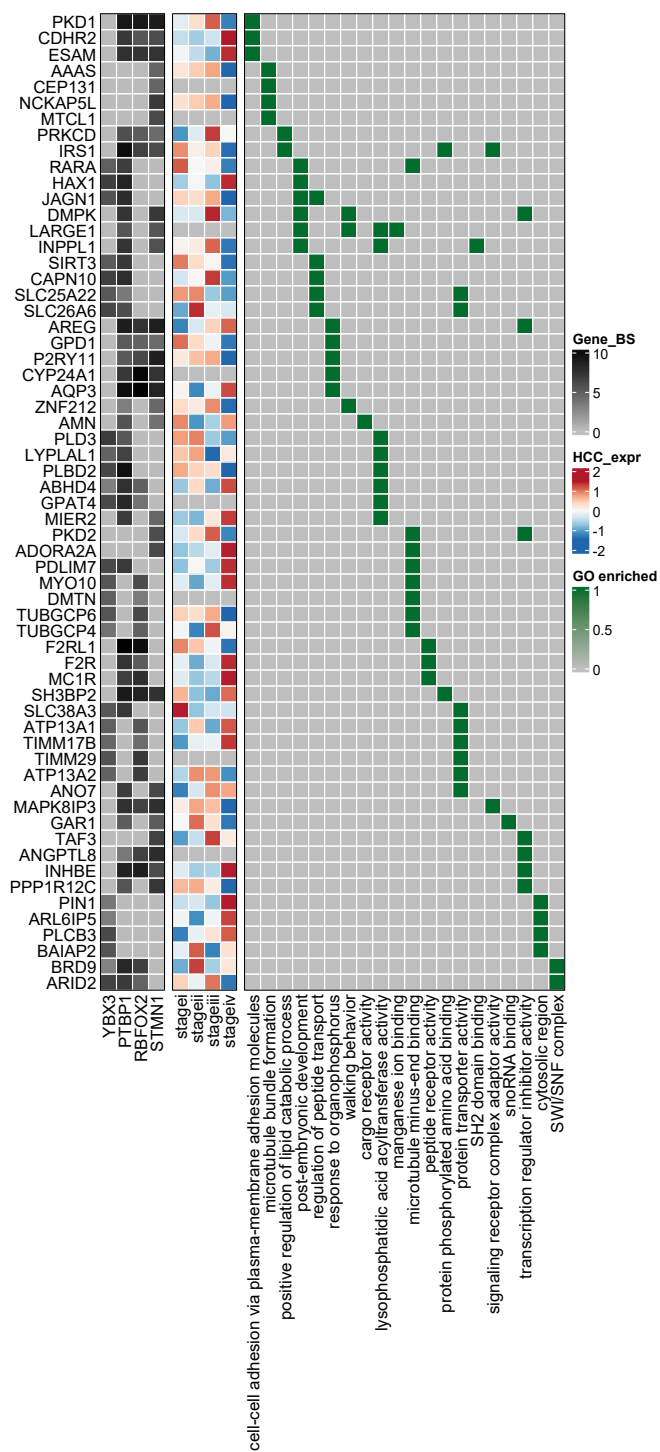
# presence of gene targets in enriched GO terms
GO_enriched_genes <- gene_targets_subsets_GO_combined_reduced %>%
pull(geneID) %>%
str_split("\\/") %>%
unlist() %>%
unique()

df <- gene_targets_subsets_GO_combined_reduced %>%
select(parentTerm, geneID) %>%
separate_longer_delim(col = geneID, delim = "/") %>%
pivot_wider(names_from = parentTerm, values_from = parentTerm,
      values_fn = length, values_fill = 0) %>%
group_by(geneID) %>%
summarise_all(sum) %>%
mutate(geneID = factor(geneID, levels=GO_enriched_genes)) %>%
arrange(geneID) %>%
column_to_rownames('geneID')

p[[4]] <- Heatmap(df,
      name = "GO enriched",
      col = circlize::colorRamp2(c(0, 1), c("#BEBEBE", "#006D2C")),
      cluster_rows = F,
      cluster_columns = F,
      row_names_side = "left",
      column_names_side = "bottom",
      width = unit(ncol(df)*4, "mm"),
      height = unit(nrow(df)*4, "mm"),

```

```
cell_fun = function(j, i, x, y, width, height, fill) {  
  grid.rect(x, y, width, height,  
            gp = gpar(col = "white", fill=NA,lwd = 1))  
},  
border = T)  
p[[1]] + p[[2]] + p[[4]]
```



Distribution of RBP target sites in target genes

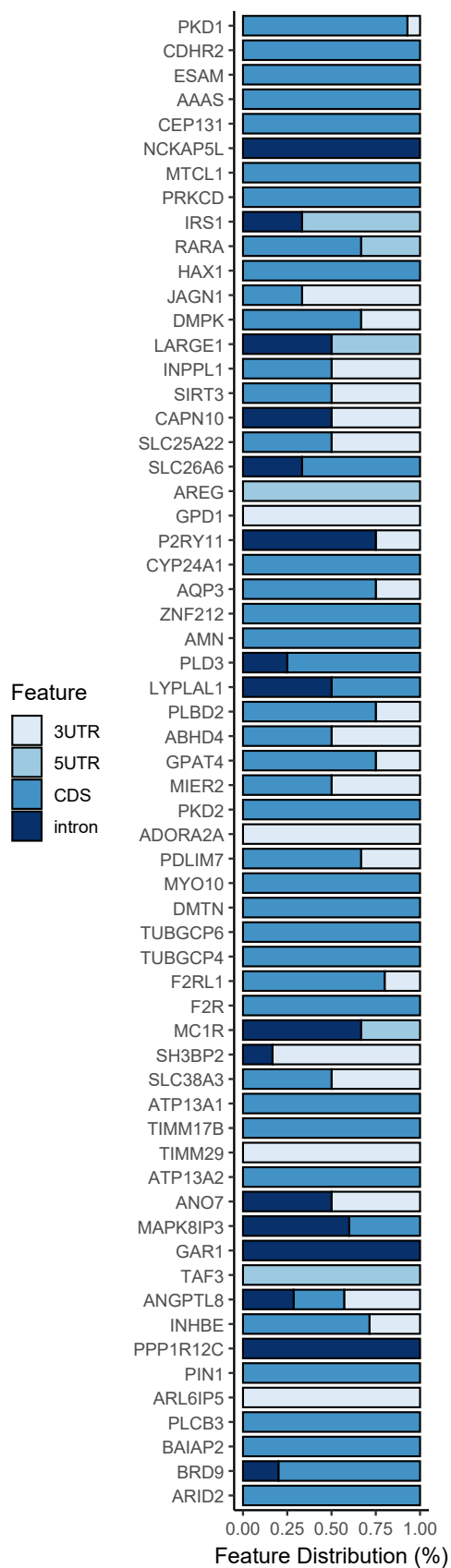
```
# distributions of RBP target sites in target genes
feature_colors <- c("intron" = "#08306B",
                   "exon" = "#08519C",
                   "CDS" = "#4292C6",
                   "5UTR" = "#9ECAE1",
                   "3UTR" = "#DEEBF7")

GO_enriched_genes <- gene_targets_subsets_GO_combined_reduced %>%
  pull(geneID) %>%
  str_split("\\\\") %>%
  unlist() %>%
  unique()

df <- RBP_filt_peaks %>%
  filter(gene_name %in% GO_enriched_genes) %>%
  group_by(gene_name, feature) %>%
  summarise(n_sites = n()) %>%
  mutate(percent = n_sites/sum(n_sites)*100) %>%
  mutate(gene_name = factor(gene_name, levels=rev(GO_enriched_genes))) %>%
  mutate(feature = factor(feature, levels=rev(names(feature_colors))))

p[[5]] <- ggplot(df, aes(x = percent, y = gene_name, fill = feature)) +
  geom_bar(stat = "identity", position = "fill", width = 0.8, color="black") +
  scale_fill_manual(values = feature_colors) +
  labs(
    x = "Feature Distribution (%)",
    y = NULL,
    fill = "Feature"
  ) +
  theme_classic(base_size = 12) +
  theme(
    legend.position="left",
    plot.margin = margin(10, 10, 10, 10)
  )

p[[5]]
```



```
# from Sondergaard et al. 2022
RBPkDs_Sondergaard2022 <- lapply(setNames(readxl::excel_sheets("data/RBPkD/table_s7_rbp_kd_rnaseq_deg.xlsx"),readxl::excel_sheets(readxl::read_excel("data/RBPkD/table_s7_rbp_kd_rnaseq_deg.xlsx", sheet = x) %>%
  dplyr::rename(ensembl_gene_id = ensembl_ID,
                external_gene_name = gene_name,
                log2FoldChange = logFC,
                padj = FDR,
                gene_biotype = gene_type)
}))
names(RBPkDs_Sondergaard2022) <- gsub("(.)_DEG", "\\1", names(RBPkDs_Sondergaard2022))

# from ENCODE Consortium (Van Nostrand et al. 2020)
RBPkDs_ENCODE_DESeq2 <- readRDS("data/RBPkD/RBPkDs_ENCODE_filt_DESeq2.rds")
```

```
rbp_names <- list(
  "YBX3" = "YBX3",
  "PTBP1" = "PTBP1",
  "RBFox2" = "RBFox2",
  "STMN1" = "STMN1"
)

feature_names <- list(
  "exon" = "exon",
  "intron" = "intron",
  "5UTR" = "5UTR",
  "3UTR" = "3UTR",
  "CDS" = "CDS",
  "all" = c("exon", "intron", "5UTR", "3UTR", "CDS")
)

rbp_targets <- lapply(rbp_names, function(x){
  lapply(feature_names, function(y){
    RBP_filt_peaks %>%
      filter(RBP == x & feature %in% y) %>%
      pull(gene_name) %>%
      unique()
  })
})

# gene background
input <- read.table(file="Data/INPUTs.TPMs.txt", stringsAsFactors = F, header=T)
input_genes <- input %>%
  filter(TPM_RAP >= 1) %>%
  pull(gene_ID) %>%
  gsub("\\..*", "", .) %>%
  unique() %>%
  mapIds(x=org.Hs.eg.db, keys=., column='SYMBOL', keytype = 'ENSEMBL') %>%
  unique()
RBP_bound_genes <- RBP_filt_peaks %>%
  pull(gene_name) %>%
  unique()
gene_background <- unique(c(input_genes, RBP_bound_genes))
random_sets_n100 <- lapply(setNames(seq(1,100), paste0("set_", seq(1,100))), function(x){
  sample(x = gene_background, size = 100)
})

genes_rnk_kd <- list(
  "YBX3" = RBPkds_ENCODE_DESeq2$HepG2_RNAi$HepG2_RNAi_YBX3_ENCF571NNW %>%
    dplyr::filter(!is.na(pvalue)) %>%
    dplyr::mutate(external_gene_name = mapIds(x=org.Hs.eg.db, keys=ensembl_gene_id, column='SYMBOL', keytype = 'ENSEMBL')) %>%
    dplyr::filter(!is.na(external_gene_name)) %>%
    dplyr::mutate(neg_log10Pval = -log10(pvalue)*sign(log2FoldChange)) %>%
    dplyr::mutate(neg_log10Pval = ifelse(is.na(neg_log10Pval), 0, neg_log10Pval)) %>%
    dplyr::arrange(dplyr::desc(neg_log10Pval)),
  "PTBP1" = RBPkds_ENCODE_DESeq2$HepG2_RNAi$HepG2_RNAi_PTBP1_ENCF044UQQ %>%
    dplyr::filter(!is.na(pvalue)) %>%
    dplyr::mutate(external_gene_name = mapIds(x=org.Hs.eg.db, keys=ensembl_gene_id, column='SYMBOL', keytype = 'ENSEMBL')) %>%
    dplyr::filter(!is.na(external_gene_name)) %>%
    dplyr::mutate(neg_log10Pval = -log10(pvalue)*sign(log2FoldChange)) %>%
    dplyr::mutate(neg_log10Pval = ifelse(is.na(neg_log10Pval), 0, neg_log10Pval)) %>%
    dplyr::arrange(dplyr::desc(neg_log10Pval)),
  "RBFox2" = RBPkds_ENCODE_DESeq2$HepG2_RNAi$HepG2_RNAi_RBFox2_ENCF166HLT %>%

```

```

dplyr::filter(!is.na(pvalue)) %>%
dplyr::mutate(external_gene_name = mapIds(x=org.Hs.eg.db, keys=ensembl_gene_id, column='SYMBOL', keytype = 'ENSEMBL')) %>%
dplyr::filter(!is.na(external_gene_name)) %>%
dplyr::mutate(neg_log10Pval = -log10(pvalue)*sign(log2FoldChange)) %>%
dplyr::mutate(neg_log10Pval = ifelse(is.na(neg_log10Pval), 0, neg_log10Pval)) %>%
dplyr::arrange(dplyr::desc(neg_log10Pval)),
"STMN1" = RBPkds_Sondergaard2022$STMN1 %>%
dplyr::filter(!is.na(external_gene_name)) %>%
dplyr::mutate(neg_log10Pval = -log10(PValue)*sign(log2FoldChange)) %>%
dplyr::mutate(neg_log10Pval = ifelse(is.na(neg_log10Pval), 0, neg_log10Pval)) %>%
dplyr::arrange(dplyr::desc(neg_log10Pval))
)

# run GSEA
gsea_res <- lapply(rbp_names, function(x) {
  print(x)
  runGSEA(gene.rnk = setNames(genes_rnk_kd[[x]]$neg_log10Pval,
                             genes_rnk_kd[[x]]$external_gene_name),
          gene.sets = rbp_targets[[x]],
          min.size = 1,
          max.size = 5000,
          nproc = 1)
})

```

```

## [1] "YBX3"
## |
##
## [1] "PTBP1"
## |
##
## [1] "RBFox2"
## |
##
## [1] "STMN1"
## |

```

```

gsea_res_rand_n100 <- lapply(rbp_names, function(x) {
  print(x)
  runGSEA(gene.rnk = setNames(genes_rnk_kd[[x]]$neg_log10Pval,
                             genes_rnk_kd[[x]]$external_gene_name),
          gene.sets = random_sets_n100,
          min.size = 1,
          max.size = 5000,
          nproc = 1)
})

```

```

## [1] "YBX3"
## |
##
## [1] "PTBP1"
## |
##
## [1] "RBFox2"
## |
##
## [1] "STMN1"
## |

```

```

df <- lapply(rbp_names, function(x) {
  gsea_res_NES = gsea_res[[x]]$all %>%
  dplyr::mutate(abs_NES = abs(NES),
               neg_log10pval = -log10(pval)) %>%
  dplyr::select(pathway, abs_NES, neg_log10pval) %>%
  dplyr::add_row(pathway = "rand_n100", abs_NES = mean(abs(gsea_res_rand_n100[[x]]$all$NES)), neg_log10pval = mean(-log10(gsea_res_rand_n100[[x]]$all$pvalue)))
  filter(!pathway %in% c("all")) %>%
  dplyr::mutate(pathway = factor(pathway, levels = c("intron", "exon", "CDS", "5UTR", "3UTR", "rand_n100")))
  if(!"exon" %in% unique(gsea_res_NES$pathway)){
    gsea_res_NES <- gsea_res_NES %>%
    add_row(pathway = "exon", abs_NES = 0, neg_log10pval = 0)
  }
  gsea_res_NES
})

lab_colors <- c("intron" = "#08306B",
               "exon" = "#08519C",
               "CDS" = "#4292C6",
               "5UTR" = "#9ECAE1",
               "3UTR" = "#DEEBF7",
               "rand_n100" = "#CCCCCC")

p <- list()

```



```

rbp <- "YBX3"
p[[1]] <- ggplot(df[[rbp]], aes(x = pathway, y = abs_NES, fill = pathway)) +
  geom_bar(stat = "identity", width = 0.8, color="black") +
  labs(
    x = "Genomic feature",
    y = "|Norm. enrich. score|"
  ) +
  scale_fill_manual(values = lab_colors) +
  ylim(0, 2) +
  ggtitle(rbp) +
  theme_classic(base_size = 14)

rbp <- "PTBP1"
p[[2]] <- ggplot(df[[rbp]], aes(x = pathway, y = abs_NES, fill = pathway)) +
  geom_bar(stat = "identity", width = 0.8, color="black") +
  labs(
    x = "Genomic feature",
    y = "|Norm. enrich. score|"
  ) +
  scale_fill_manual(values = lab_colors) +
  ylim(0, 2) +
  ggtitle(rbp) +
  theme_classic(base_size = 14)

rbp <- "RBFox2"
p[[3]] <- ggplot(df[[rbp]], aes(x = pathway, y = abs_NES, fill = pathway)) +
  geom_bar(stat = "identity", width = 0.8, color="black") +
  labs(
    x = "Genomic feature",
    y = "|Norm. enrich. score|"
  ) +
  scale_fill_manual(values = lab_colors) +
  ylim(0, 2) +
  ggtitle(rbp) +
  theme_classic(base_size = 14)

rbp <- "STMN1"
p[[4]] <- ggplot(df[[rbp]], aes(x = pathway, y = abs_NES, fill = pathway)) +
  geom_bar(stat = "identity", width = 0.8, color="black") +
  labs(
    x = "Genomic feature",
    y = "|Norm. enrich. score|"
  ) +
  scale_fill_manual(values = lab_colors) +
  ylim(0, 2) +
  ggtitle(rbp) +
  theme_classic(base_size = 14)

combined_plot <- (p[[1]] | p[[2]]) / (p[[3]] | p[[4]])

print(combined_plot)

```

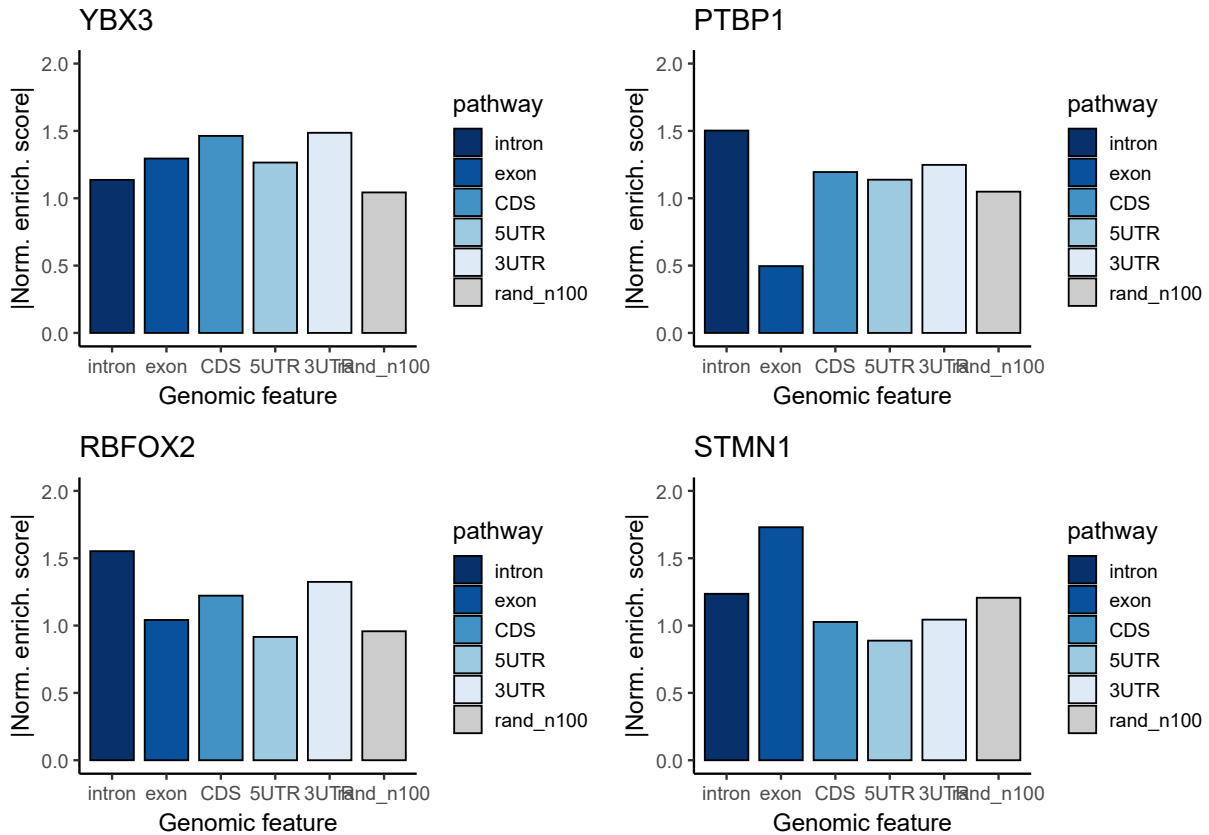


Figure 6C - Enrichment of RBP target subsets in RBPKD datasets

```
# target subsets for PTBP1, RBFOX2, YBX3, STMN1. Both shared and unique.
gene_targets_subsets <- RBP_targets_overlap_more20_sets(c("YBX3",
  "YBX3_PTBP1",
  "YBX3_RBFOX2",
  "YBX3_PTBP1_RBFOX2",
  "YBX3_HNRNPA1",
  "PTBP1",
  "PTBP1_RBFOX2",
  "PTBP1_STMN1",
  "PTBP1_RBFOX2_STMN1",
  "PTBP1_HNRNPC",
  "RBFOX2",
  "STMN1"])

input <- read.table(file="Data/INPUTs.TPMs.txt", stringsAsFactors = F, header=T)
input_genes <- input %>%
  filter(TPM_RAP >= 1) %>%
  pull(gene_ID) %>%
  gsub("\\.\\.", "", .) %>%
  unique() %>%
  mapIds(x=org.Hs.eg.db, keys=., column='SYMBOL', keytype = 'ENSEMBL') %>%
  unique()
RBP_bound_genes <- RBP_filt_peaks %>%
  pull(gene_name) %>%
  unique()
gene_background <- unique(c(input_genes, RBP_bound_genes))

genes_rnk_kd <- list(
  "YBX3" = RBPKDs_ENCODE_DESeq2$HepG2_RNAi$HepG2_RNAi_YBX3_ENCF571NNW %>%
    dplyr::filter(!is.na(pvalue)) %>%
    dplyr::mutate(external_gene_name = mapIds(x=org.Hs.eg.db, keys=ensembl_gene_id, column='SYMBOL', keytype = 'ENSEMBL')) %>%
    dplyr::filter(!is.na(external_gene_name)) %>%
    dplyr::mutate(neg_log10Pval = -log10(pvalue)*sign(log2FoldChange)) %>%
    dplyr::mutate(neg_log10Pval = ifelse(is.na(neg_log10Pval), 0, neg_log10Pval)) %>%
    dplyr::arrange(dplyr::desc(neg_log10Pval)),
```

```

"PTBP1" = RBPkDs_ENCODE_DESeq2$HepG2_RNAi$HepG2_RNAi_PTBP1_ENCF044UQQ %>%
  dplyr::filter(!is.na(pvalue)) %>%
  dplyr::mutate(external_gene_name = mapIds(x=org.Hs.eg.db, keys=ensembl_gene_id, column='SYMBOL', keytype = 'ENSEMBL')) %>%
  dplyr::filter(!is.na(external_gene_name)) %>%
  dplyr::mutate(neg_log10Pval = -log10(pvalue)*sign(log2FoldChange)) %>%
  dplyr::mutate(neg_log10Pval = ifelse(is.na(neg_log10Pval), 0, neg_log10Pval)) %>%
  dplyr::arrange(dplyr::desc(neg_log10Pval)),
"RBFOX2" = RBPkDs_ENCODE_DESeq2$HepG2_RNAi$HepG2_RNAi_RBFOX2_ENCF166HLT %>%
  dplyr::filter(!is.na(pvalue)) %>%
  dplyr::mutate(external_gene_name = mapIds(x=org.Hs.eg.db, keys=ensembl_gene_id, column='SYMBOL', keytype = 'ENSEMBL')) %>%
  dplyr::filter(!is.na(external_gene_name)) %>%
  dplyr::mutate(neg_log10Pval = -log10(pvalue)*sign(log2FoldChange)) %>%
  dplyr::mutate(neg_log10Pval = ifelse(is.na(neg_log10Pval), 0, neg_log10Pval)) %>%
  dplyr::arrange(dplyr::desc(neg_log10Pval)),
"STMN1" = RBPkDs_Sondergaard2022$STMN1 %>%
  dplyr::filter(!is.na(external_gene_name)) %>%
  dplyr::mutate(neg_log10Pval = -log10(PValue)*sign(log2FoldChange)) %>%
  dplyr::mutate(neg_log10Pval = ifelse(is.na(neg_log10Pval), 0, neg_log10Pval)) %>%
  dplyr::arrange(dplyr::desc(neg_log10Pval))
)

# run GSEA
gsea_res <- lapply(rbp_names, function(x) {
  print(x)
  runGSEA(gene.rnk = setNames(genes_rnk_kd[[x]]$neg_log10Pval,
                              genes_rnk_kd[[x]]$external_gene_name),
          gene.sets = gene_targets_subsets,
          min.size = 1,
          max.size = 5000,
          nproc = 1)
})

## [1] "YBX3"
## [1] "PTBP1"
## [1] "RBFOX2"
## |
##
## [1] "STMN1"
## |
df <- lapply(rbp_names, function(x) {
  gsea_res_NES = gsea_res[[x]]$all %>%
    dplyr::mutate(abs_NES = abs(NES),
                  neg_log10pval = -log10(pval)) %>%
    dplyr::select(pathway, abs_NES, neg_log10pval) %>%
    dplyr::mutate(pathway = factor(pathway, levels = c("YBX3",
                                                       "YBX3_PTBP1",
                                                       "YBX3_RBFOX2",
                                                       "YBX3_PTBP1_RBFOX2",
                                                       "YBX3_HNRNPA1",
                                                       "PTBP1",
                                                       "PTBP1_RBFOX2",
                                                       "PTBP1_STMN1",
                                                       "PTBP1_RBFOX2_STMN1",
                                                       "PTBP1_HNRNPC",
                                                       "RBFOX2",
                                                       "STMN1"))))

  gsea_res_NES
})

p <- list()
rbp <- "YBX3"
p[[1]] <- ggplot(df[[rbp]], aes(x = pathway, y = abs_NES)) +
  geom_bar(stat = "identity", width = 0.8, color="black", fill="#ED6062") +
  labs(
    x = "",
    y = "|Norm. enrich. score|"
  ) +
  ylim(0, 2) +
  ggtitle(rbp) +
  theme_classic(base_size = 14) +
  theme(axis.text.x = element_blank())

rbp <- "PTBP1"
p[[2]] <- ggplot(df[[rbp]], aes(x = pathway, y = abs_NES)) +
  geom_bar(stat = "identity", width = 0.8, color="black", fill="#ED6062") +
  labs(
    x = "",
    y = "|Norm. enrich. score|"
  ) +
  ylim(0, 2) +

```

```

ggtitle(rbp) +
theme_classic(base_size = 14) +
theme(axis.text.x = element_blank())

rbp <- "RBFOX2"
p[[3]] <- ggplot(df[[rbp]], aes(x = pathway, y = abs_NES)) +
  geom_bar(stat = "identity", width = 0.8, color="black", fill="#ED6062") +
  labs(
    x = "",
    y = "|Norm. enrich. score|"
  ) +
  ylim(0, 2) +
  ggtitle(rbp) +
  theme_classic(base_size = 14) +
  theme(axis.text.x = element_blank())

rbp <- "STMN1"
p[[4]] <- ggplot(df[[rbp]], aes(x = pathway, y = abs_NES)) +
  geom_bar(stat = "identity", width = 0.8, color="black", fill="#9C73C8") +
  labs(
    x = "RBP Target Subset",
    y = "|Norm. enrich. score|"
  ) +
  ylim(0, 2) +
  ggtitle(rbp) +
  theme_classic(base_size = 14) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5))

combined_plot <- p[[1]] / p[[2]] / p[[3]] / p[[4]]

print(combined_plot)

```

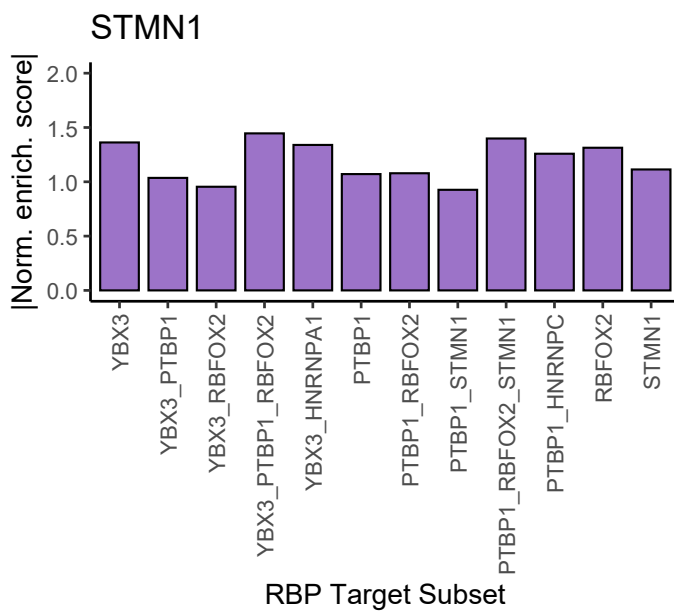
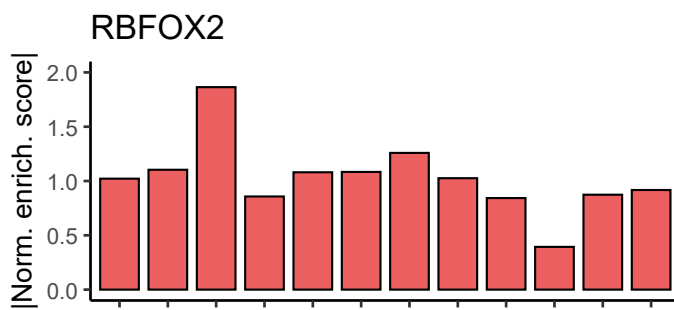
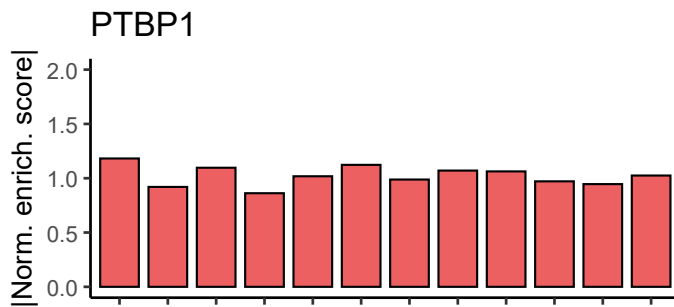
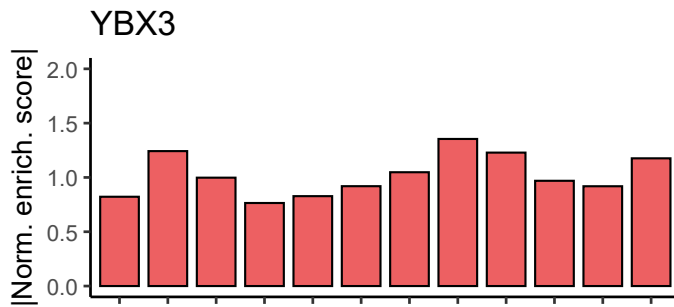


Figure S6H - Volcano plots of RBP target genes in RBPKD datasets

```
RBP_diff_expr <- list(
  "YBX3" = RBPKDs_ENCODE_DESeq2$HepG2_RNAi$HepG2_RNAi_YBX3_ENCFF571NNW %>%
    dplyr::filter(!is.na(padj)) %>%
    dplyr::mutate(external_gene_name = mapIds(x=org.Hs.eg.db, keys=ensembl_gene_id, column='SYMBOL', keytype = 'ENSEMBL')) %>%
    dplyr::filter(!is.na(external_gene_name)) %>%
    dplyr::mutate(padj = ifelse(is.na(padj), 1, padj)) %>%
    dplyr::mutate(neg_log10Pval = -log10(padj)) %>%
    dplyr::mutate(is_target = external_gene_name %in% RBP_targets[["YBX3"]]) %>%
    dplyr::mutate(is_target_in_subset = external_gene_name %in% intersect(RBP_targets[["YBX3"]], unlist(RBP_targets_overlap_more20_sets))) %>%
    dplyr::arrange(neg_log10Pval),

  "PTBP1" = RBPKDs_ENCODE_DESeq2$HepG2_RNAi$HepG2_RNAi_PTBP1_ENCFF044UQQ %>%
    dplyr::filter(!is.na(padj)) %>%
    dplyr::mutate(external_gene_name = mapIds(x=org.Hs.eg.db, keys=ensembl_gene_id, column='SYMBOL', keytype = 'ENSEMBL')) %>%
    dplyr::filter(!is.na(external_gene_name)) %>%
    dplyr::mutate(padj = ifelse(is.na(padj), 1, padj)) %>%
    dplyr::mutate(neg_log10Pval = -log10(padj)) %>%
    dplyr::mutate(is_target = external_gene_name %in% RBP_targets[["PTBP1"]]) %>%
    dplyr::mutate(is_target_in_subset = external_gene_name %in% intersect(RBP_targets[["PTBP1"]], unlist(RBP_targets_overlap_more20_sets))) %>%
    dplyr::arrange(neg_log10Pval),

  "RBFox2" = RBPKDs_ENCODE_DESeq2$HepG2_RNAi$HepG2_RNAi_RBFox2_ENCFF166HLT %>%
    dplyr::filter(!is.na(padj)) %>%
    dplyr::mutate(external_gene_name = mapIds(x=org.Hs.eg.db, keys=ensembl_gene_id, column='SYMBOL', keytype = 'ENSEMBL')) %>%
    dplyr::filter(!is.na(external_gene_name)) %>%
    dplyr::mutate(padj = ifelse(is.na(padj), 1, padj)) %>%
    dplyr::mutate(neg_log10Pval = -log10(padj)) %>%
    dplyr::mutate(is_target = external_gene_name %in% RBP_targets[["RBFox2"]]) %>%
    dplyr::mutate(is_target_in_subset = external_gene_name %in% intersect(RBP_targets[["RBFox2"]], unlist(RBP_targets_overlap_more20_sets))) %>%
    dplyr::arrange(neg_log10Pval),

  "STMN1" = RBPKDs_Sondergaard2022$STMN1 %>%
    dplyr::filter(!is.na(external_gene_name)) %>%
    dplyr::mutate(padj = ifelse(is.na(padj), 1, padj)) %>%
    dplyr::mutate(neg_log10Pval = -log10(padj)) %>%
    dplyr::mutate(is_target = external_gene_name %in% RBP_targets[["STMN1"]]) %>%
    dplyr::mutate(is_target_in_subset = external_gene_name %in% intersect(RBP_targets[["STMN1"]],
                                                                    unlist(RBP_targets_overlap_more20_sets))) %>%
    dplyr::arrange(neg_log10Pval)
)

p <- list()
# YBX volcano plot
rbp <- "YBX3"
df <- RBP_diff_expr[[rbp]] %>%
  dplyr::mutate(is_target = ifelse(is_target, "Target", "Non-target")) %>%
  dplyr::filter(is_target == 'Target' | external_gene_name == rbp) %>%
  dplyr::mutate(is_target_in_subset = ifelse(is_target_in_subset, "Target in subset", "Non-target in subset")) %>%
  dplyr::mutate(is_target_in_subset = ifelse(padj<0.05 & abs(log2FoldChange)>log2(1.5), is_target_in_subset, "Non significant")) %>%
  dplyr::mutate(is_target_in_subset = ifelse(external_gene_name == rbp, rbp, is_target_in_subset)) %>%
  dplyr::mutate(is_target_in_subset = factor(is_target_in_subset, levels = c("Non significant", "Non-target in subset", "Target in subset", rbp)))
  dplyr::arrange(is_target_in_subset)
to_label <- c(df %>% filter(external_gene_name == rbp) %>% pull(external_gene_name),
              df %>% filter(is_target_in_subset == "Non-target in subset" & padj<0.05 & abs(log2FoldChange)>log2(1.5)) %>% dplyr::arrange(padj) %>%
              df %>% filter(is_target_in_subset == "Target in subset" & padj<0.05 & abs(log2FoldChange)>log2(1.5)) %>% pull(external_gene_name)) %>%
  unique()
df <- df %>%
  mutate(label = ifelse(external_gene_name %in% to_label, external_gene_name, ""))

paste0(rbp, ":")

## [1] "YBX3:"
table(df$is_target_in_subset)

##
##      Non significant Non-target in subset      Target in subset
##      4067          39          6
##      YBX3
##      1
p[[1]] <- ggplot(df, aes(x = log2FoldChange, y = neg_log10Pval, color = is_target_in_subset)) +
  geom_point(size = 2) +
  scale_alpha_manual(values = c("Target in subset" = 0.8, "Non-target in subset" = 0.3, rbp = 1)) +
  scale_color_manual(values = c("Target in subset" = "#e31a1c", "Non-target in subset" = "#fb9a99", rbp = "#000000", "Non significant" = "#f0f0f0")) +
  geom_hline(yintercept = -log10(0.05), linetype = "dashed", color = "black") +
  geom_vline(xintercept = c(-log2(1.5), log2(1.5)), linetype = "dashed", color = "black") +
  geom_text_repel(aes(label = label), box.padding = 0.5, point.padding = 0.5, segment.color = "black", max.overlaps = Inf) +
  labs(
    x = "log2 Fold Change",
    y = "-log10(padj)"
  )
```

```

) +
ggtitle(rbp) +
theme_bw(base_size = 14) +
theme(panel.grid = element_blank(),
      legend.position = "none")

# PTBP1 volcano plot
rbp <- "PTBP1"
df <- RBP_diff_expr[[rbp]] %>%
  dplyr::mutate(is_target = ifelse(is_target, "Target", "Non-target")) %>%
  dplyr::filter(is_target == 'Target' | external_gene_name == rbp) %>%
  dplyr::mutate(is_target_in_subset = ifelse(is_target_in_subset, "Target in subset", "Non-target in subset")) %>%
  dplyr::mutate(is_target_in_subset = ifelse(padj<0.05 & abs(log2FoldChange)>log2(1.5), is_target_in_subset, "Non significant")) %>%
  dplyr::mutate(is_target_in_subset = ifelse(external_gene_name == rbp, rbp, is_target_in_subset)) %>%
  dplyr::mutate(is_target_in_subset = factor(is_target_in_subset, levels = c("Non significant", "Non-target in subset", "Target in subset", rbp)))
  dplyr::arrange(is_target_in_subset)
to_label <- c(df %>% filter(external_gene_name == rbp) %>% pull(external_gene_name),
              df %>% filter(is_target_in_subset == "Non-target in subset" & padj<0.05 & abs(log2FoldChange)>log2(1.5)) %>% dplyr::arrange(padj) %>%
              df %>% filter(is_target_in_subset == "Target in subset" & padj<0.05 & abs(log2FoldChange)>log2(1.5)) %>% dplyr::arrange(padj) %>% pull(external_gene_name))
unique()
df <- df %>%
  mutate(label = ifelse(external_gene_name %in% to_label, external_gene_name, ""))

paste0(rbp, ".")

## [1] "PTBP1:"
table(df$is_target_in_subset)

##
##      Non significant Non-target in subset      Target in subset
##              3169              406              114
##      PTBP1
##              1

p[[2]] <- ggplot(df, aes(x = log2FoldChange, y = neg_log10Pval, color = is_target_in_subset)) +
  geom_point(size = 2) +
  scale_alpha_manual(values = c("Target in subset" = 0.8, "Non-target in subset" = 0.3, rbp = 1)) +
  scale_color_manual(values = c("Target in subset" = "#e31a1c", "Non-target in subset" = "#fb9a99", rbp = "#000000", "Non significant" = "#BFBFBF"))
  geom_hline(yintercept = -log10(0.05), linetype = "dashed", color = "black") +
  geom_vline(xintercept = c(-log2(1.5), log2(1.5)), linetype = "dashed", color = "black") +
  geom_text_repel(aes(label = label), box.padding = 0.5, point.padding = 0.5, segment.color = "black", max.overlaps = Inf) +
  labs(
    x = "log2 Fold Change",
    y = "-log10(padj)"
  ) +
  ggtitle(rbp) +
  theme_bw(base_size = 14) +
  theme(panel.grid = element_blank(),
        legend.position = "none")

# RBFOX2 volcano plot
rbp <- "RBFOX2"
df <- RBP_diff_expr[[rbp]] %>%
  dplyr::mutate(is_target = ifelse(is_target, "Target", "Non-target")) %>%
  dplyr::filter(is_target == 'Target' | external_gene_name == rbp) %>%
  dplyr::mutate(is_target_in_subset = ifelse(is_target_in_subset, "Target in subset", "Non-target in subset")) %>%
  dplyr::mutate(is_target_in_subset = ifelse(padj<0.05 & abs(log2FoldChange)>log2(1.5), is_target_in_subset, "Non significant")) %>%
  dplyr::mutate(is_target_in_subset = ifelse(external_gene_name == rbp, rbp, is_target_in_subset)) %>%
  dplyr::mutate(is_target_in_subset = factor(is_target_in_subset, levels = c("Non significant", "Non-target in subset", "Target in subset", rbp)))
  dplyr::arrange(is_target_in_subset)
to_label <- c(df %>% filter(external_gene_name == rbp) %>% pull(external_gene_name),
              df %>% filter(is_target_in_subset == "Non-target in subset" & padj<0.05 & abs(log2FoldChange)>log2(1.5)) %>% dplyr::arrange(padj) %>%
              df %>% filter(is_target_in_subset == "Target in subset" & padj<0.05 & abs(log2FoldChange)>log2(1.5)) %>% pull(external_gene_name))
unique()
df <- df %>%
  mutate(label = ifelse(external_gene_name %in% to_label, external_gene_name, ""))

paste0(rbp, ".")

## [1] "RBFOX2:"
table(df$is_target_in_subset)

##
##      Non significant Non-target in subset      Target in subset
##              3624              12              3
##      RBFOX2
##              1

p[[3]] <- ggplot(df, aes(x = log2FoldChange, y = neg_log10Pval, color = is_target_in_subset)) +
  geom_point(size = 2) +

```

```

scale_alpha_manual(values = c("Target in subset" = 0.8, "Non-target in subset" = 0.3, rbp = 1)) +
scale_color_manual(values = c("Target in subset" = "#e31a1c", "Non-target in subset" = "#fb9a99", rbp = "#000000", "Non significant" = "#BFBFBF") +
geom_hline(yintercept = -log10(0.05), linetype = "dashed", color = "black") +
geom_vline(xintercept = c(-log2(1.5), log2(1.5)), linetype = "dashed", color = "black") +
geom_text_repel(aes(label = label), box.padding = 0.5, point.padding = 0.5, segment.color = "black", max.overlaps = Inf) +
labs(
  x = "log2 Fold Change",
  y = "-log10(padj)"
) +
ggtitle(rbp) +
theme_bw(base_size = 14) +
theme(panel.grid = element_blank(),
  legend.position = "none")

# STMN1 volcano plot
rbp <- "STMN1"
df <- RBP_diff_expr[[rbp]] %>%
  dplyr::mutate(is_target = ifelse(is_target, "Target", "Non-target")) %>%
  dplyr::filter(is_target == 'Target' | external_gene_name == rbp) %>%
  dplyr::mutate(is_target_in_subset = ifelse(is_target_in_subset, "Target in subset", "Non-target in subset")) %>%
  dplyr::mutate(is_target_in_subset = ifelse(padj<0.05 & abs(log2FoldChange)>log2(1.5), is_target_in_subset, "Non significant")) %>%
  dplyr::mutate(is_target_in_subset = ifelse(external_gene_name == rbp, rbp, is_target_in_subset)) %>%
  dplyr::mutate(is_target_in_subset = factor(is_target_in_subset, levels = c("Non significant", "Non-target in subset", "Target in subset", rbp)))
to_label <- c(df %>% filter(external_gene_name == rbp) %>% pull(external_gene_name),
  df %>% filter(is_target_in_subset == "Non-target in subset" & padj<0.05 & abs(log2FoldChange)>log2(1.5)) %>% dplyr::arrange(padj) %>%
  df %>% filter(is_target_in_subset == "Target in subset" & padj<0.05 & abs(log2FoldChange)>log2(1.5)) %>% pull(external_gene_name)) %>%
  unique()
df <- df %>%
  mutate(label = ifelse(external_gene_name %in% to_label, external_gene_name, ""))

paste0(rbp, ":",)

## [1] "STMN1:"
table(df$is_target_in_subset)

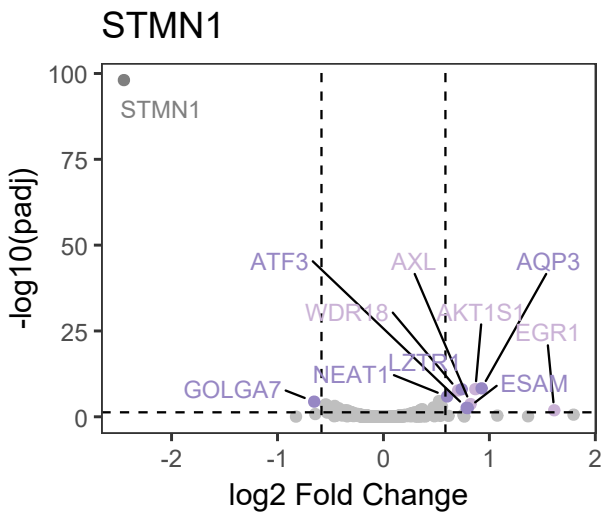
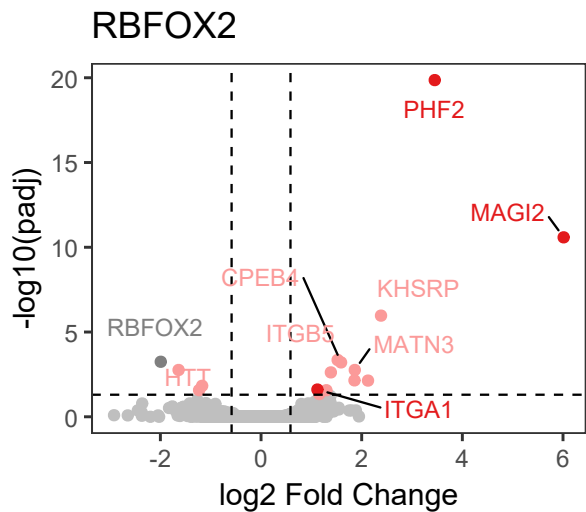
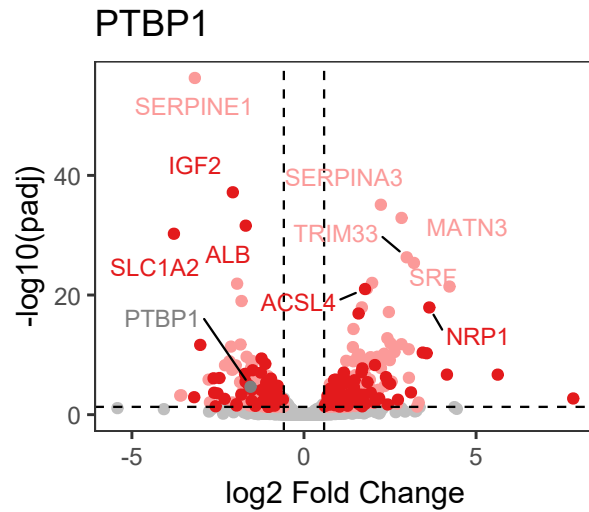
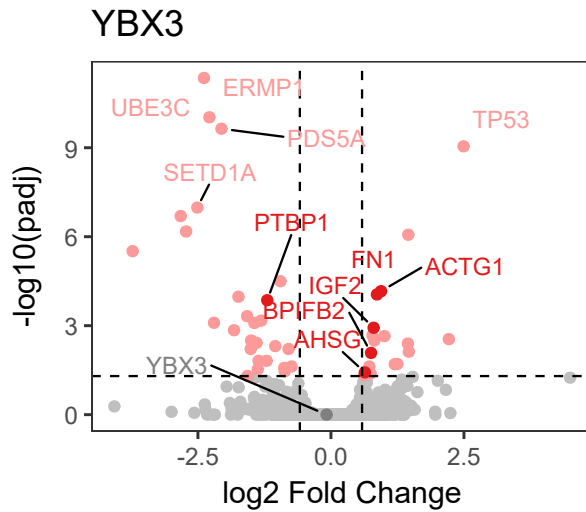
##
##      Non significant Non-target in subset      Target in subset
##              980              4              6
##      STMN1
##              1

p[[4]] <- ggplot(df, aes(x = log2FoldChange, y = neg_log10Pval, color = is_target_in_subset)) +
  geom_point(size = 2) +
  scale_alpha_manual(values = c("Target in subset" = 0.8, "Non-target in subset" = 0.3, rbp = 1)) +
  scale_color_manual(values = c("Target in subset" = "#9787c4", "Non-target in subset" = "#cab2d6", rbp = "#000000", "Non significant" = "#BFBFBF") +
  geom_hline(yintercept = -log10(0.05), linetype = "dashed", color = "black") +
  geom_vline(xintercept = c(-log2(1.5), log2(1.5)), linetype = "dashed", color = "black") +
  geom_text_repel(aes(label = label), box.padding = 0.5, point.padding = 0.5, segment.color = "black", max.overlaps = Inf) +
  labs(
    x = "log2 Fold Change",
    y = "-log10(padj)"
  ) +
  ggtitle(rbp) +
  theme_bw(base_size = 14) +
  theme(panel.grid = element_blank(),
    legend.position = "none")

combined_plot <- (p[[1]] | p[[2]]) / (p[[3]] | p[[4]])

print(combined_plot)

```

SessionInfo

```
sessionInfo()
```

```
## R version 4.4.1 (2024-06-14 ucrt)
## Platform: x86_64-w64-mingw32/x64
## Running under: Windows 10 x64 (build 19045)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## time zone: Europe/Stockholm
## tzcode source: internal
##
## attached base packages:
## [1] grid      stats4    graphics grDevices utils    datasets
## [8] methods  base
##
## other attached packages:
```

```

## [1] ggrepel_0.9.5          fgsea_1.30.0          circlize_0.4.16
## [4] ComplexHeatmap_2.20.0  rrvgo_1.16.0          clusterProfiler_4.12.0
## [7] ComplexUpset_1.3.3     patchwork_1.2.0       g3viz_1.2.0
## [10] eulerr_7.0.2           org.Hs.eg.db_3.19.1  AnnotationDbi_1.66.0
## [13] IRanges_2.38.1         S4Vectors_0.42.1     Biobase_2.64.0
## [16] BiocGenerics_0.50.0    lubridate_1.9.3       forcats_1.0.0
## [19] stringr_1.5.1          dplyr_1.1.4           purrr_1.0.2
## [22] readr_2.1.5            tidyr_1.3.1           tibble_3.2.1
## [25] ggplot2_3.5.1          tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
## [1] RColorBrewer_1.1-3     shape_1.4.6.1         rstudioapi_0.16.0
## [4] jsonlite_1.8.8        umap_0.2.10.0         magrittr_2.0.3
## [7] farver_2.1.2          rmarkdown_2.27        GlobalOptions_0.1.2
## [10] fs_1.6.4              zlibbioc_1.50.0       vctrs_0.6.5
## [13] memoise_2.0.1         askpass_1.2.0         ggtree_3.12.0
## [16] htmltools_0.5.8.1     cellranger_1.1.0     gridGraphics_0.5-1
## [19] htmlwidgets_1.6.4     plyr_1.8.9            httr2_1.0.1
## [22] cachem_1.1.0          igraph_2.0.3          iterators_1.0.14
## [25] mime_0.12             lifecycle_1.0.4       pkgconfig_2.0.3
## [28] Matrix_1.7-0          R6_2.5.1              fastmap_1.2.0
## [31] gson_0.1.0            clue_0.3-66           GenomeInfoDbData_1.2.12
## [34] shiny_1.8.1.1         digest_0.6.36         aplot_0.2.3
## [37] enrichplot_1.24.0     colorspace_2.1-0     RSpectra_0.16-1
## [40] RSQLite_2.3.7         labeling_0.4.3        fansi_1.0.6
## [43] timechange_0.3.0      httr_1.4.7            polyclip_1.10-6
## [46] compiler_4.4.1        doParallel_1.0.17    bit64_4.0.5
## [49] withr_3.0.0           BiocParallel_1.38.0  viridis_0.6.5
## [52] DBI_1.2.3             highr_0.11            ggforce_0.4.2
## [55] MASS_7.3-60.2         openssl_2.2.0        rappdirs_0.3.3
## [58] rjson_0.2.21          HD0.db_0.99.1         tools_4.4.1
## [61] ape_5.8               scatterpie_0.2.3      httpuv_1.6.15
## [64] glue_1.7.0            promises_1.3.0        nlme_3.1-164
## [67] GOSemSim_2.30.0       polylabelr_0.2.0     shadowtext_0.1.3
## [70] gridBase_0.4-7        cluster_2.1.6         reshape2_1.4.4
## [73] generics_0.1.3        gtable_0.3.5          tzdb_0.4.0
## [76] data.table_1.15.4     hms_1.1.3            xml2_1.3.6
## [79] tidygraph_1.3.1       utf8_1.2.4           XVector_0.44.0
## [82] foreach_1.5.2        pillar_1.9.0          yulab.utils_0.1.4
## [85] later_1.3.2           splines_4.4.1         tweenr_2.0.3
## [88] treeio_1.28.0         lattice_0.22-6        bit_4.0.5
## [91] tidyselect_1.2.1     GO.db_3.19.1         tm_0.7-13
## [94] Biostrings_2.72.1     knitr_1.48            gridExtra_2.3
## [97] NLP_0.2-1            xfun_0.45             graphlayouts_1.1.1
## [100] matrixStats_1.3.0    pheatmap_1.0.12      stringi_1.8.4
## [103] UCSC.utils_1.0.0     lazyeval_0.2.2        ggfun_0.1.5
## [106] yaml_2.3.9           evaluate_0.24.0       codetools_0.2-20
## [109] wordcloud_2.6        ggraph_2.2.1         qvalue_2.36.0
## [112] ggplotify_0.1.2      cli_3.6.3            reticulate_1.38.0
## [115] xtable_1.8-4         munsell_0.5.1         readxl_1.4.3
## [118] treemap_2.4-4        Rcpp_1.0.12           GenomeInfoDb_1.40.1
## [121] png_0.1-8            parallel_4.4.1        blob_1.2.4
## [124] DOSE_3.30.1          slam_0.1-50           viridisLite_0.4.2
## [127] tidytree_0.4.6       scales_1.3.0          crayon_1.5.3
## [130] GetoptLong_1.0.5     rlang_1.1.4           cowplot_1.1.3
## [133] fastmatch_1.1-4      KEGGREST_1.44.1

```