

mir181 binding sites - union of mir181 enriched binding sites and Ago binding sites targeted by mir181

Melina Klostermann

14 April, 2023

Contents

1	Libraries and settings	1
2	What was done?	2
3	Files	2
4	mir181 binding sites	3
5	Combine mir181 binding sites with differential binding sites	4
6	Characterise mir181 binding sites	7
7	Save output	10
8	Session Info	10

1 Libraries and settings

```
# -----  
# libraries  
# -----  
library(tidyverse)  
library(GenomicRanges)  
library(colorspace)  
library(eulerr)  
library(gghalves)  
  
# -----  
# settings  
# -----  
out <- "/Users/melinaklostermann/Documents/projects/AgoCLIP_miR181/R_github/miR181_paper/Figure1/mir181  
source("/Users/melinaklostermann/Documents/projects/R_general_functions/theme_paper.R")  
source("/Users/melinaklostermann/Documents/projects/AgoCLIP_miR181/mirko_files/mirECLIP/DifferentialBin  
  
# farben  
farbeneg <- "#B4B4B4"  
farbe1 <- "#0073C2FF" #WT farbe
```

```

farbe2 <- "#EFC000FF"
farbe3 <- "#CD534CFF" #mir181KO farbe
farbe4 <- "#7AA6DCFF"
farbe5 <- "#868686FF"
farbe6 <- "#003C67FF"
farbe7 <- "#8F7700FF"
farbe8 <- "#3B3B3BFF"
farbe9 <- "#A73030FF"
farbe10 <- "#4A6990FF"
farbe11 <- "#FF6F00FF"
farbe12 <- "#C71000FF"
farbe13 <- "#008EAOFF"
farbe14 <- "#8A4198FF"
farbe15 <- "#5A9599FF"
farbe16 <- "#FF6348FF"

```

2 What was done?

mir181 binding sites are defined as the union of - AGO binding sites that contain at least 2 chimirc mir181 crosslinks (from the IP_WT chimeric reads or the IP_mir181_WT chimeric reads) in a window from 10nt before till 10nt after a the AGO binding site - binding sites defined on enriched mir181 data (IP_mir181_WT)

- the two subgroups are plotted as a venn diagram (figure 1 XX)
- this is compared to the differntially regulated AGO binding sites from the mir181 KO condition
- the genotype and gene region of the mir 181 binding sites (union) are plotted (Figure2XX)

3 Files

```

# -----
# mir181 enriched binding sites
# -----
mir181_enriched <- readRDS("/Users/melinaklostermann/Documents/projects/AgoCLIP_miR181/R_github/miR181_p

# -----
# chimeric reads
# -----

chimeric_reads <- readRDS("/Users/melinaklostermann/Documents/projects/AgoCLIP_miR181/R_github/miR181_p

# -----
# AGO binding sites
# -----
ago_bs <- readRDS("/Users/melinaklostermann/Documents/projects/AgoCLIP_miR181/R_github/miR181_paper/Metl

# -----
# BS downregulated in mir181 KO
# -----
diff <- readRDS("/Users/melinaklostermann/Documents/projects/AgoCLIP_miR181/R_github/miR181_paper/Figur
diff$down <- (diff$resBs.padj < 0.05) & (diff$resBs.log2FoldChange < -log2(2))

```

4 mir181 binding sites

4.1 Get AGO binding sites with chimeric mir181

Here we define mir181 AGO binding sites by overlapping the AGO binding sites (see script Methods/02_AGO_binding_site_definition) with the chimeric mir181 reads (see script Figure1/Ago_targetome). AGO binding sites that contained at least 2 chimeric mir181 crosslinks in the binding site or within 10nt proximity to the binding site are selected as mir181 Ago binding sites.

```
# use region of bs +/-10nt for overlaps
ago_bs_10 <- ago_bs + 10

# use chimeric reads from both mir181 enriched and non-enriched data
chimeric_reads <- c(makeGRangesFromDataFrame(chimeric_reads$IP_WT, keep.extra.columns = T), makeGRangesFromDataFrame(chimeric_reads$IP_KO, keep.extra.columns = T))

# find overlaps of mir181 and AGO bs
idx <- findOverlaps(ago_bs_10, chimeric_reads)

# make a data frame from the ago bs
names(ago_bs) <- 1:NROW(ago_bs)
ago_bs <- as.data.frame(ago_bs)
ago_bs$BS_ID <- rownames(ago_bs)

# add mir info to ago bs
ago_bs_mir181_chi <- cbind(ago_bs[queryHits(idx),], mir_IP = chimeric_reads[subjectHits(idx),]$Name)

ago_bs_mir181_chi <- ago_bs_mir181_chi[grepl(ago_bs_mir181_chi$mir_IP,
                                           pattern = "miR-181"),]

# count chimerics
mir181_chi <- ago_bs_mir181_chi %>% group_by(BS_ID) %>%
  summarize(n_mir181 = sum(grepl(mir_IP, pattern = "miR-181")),
            n_mir181a = sum(grepl(mir_IP, pattern = "miR-181a")),
            n_mir181b = sum(grepl(mir_IP, pattern = "miR-181b")),
            n_mir181c = sum(grepl(mir_IP, pattern = "miR-181c")),
            n_mir181d = sum(grepl(mir_IP, pattern = "miR-181d")),
            .groups = "keep") %>% subset (n_mir181 > 0)

ago_bs_mir181_chi <- ago_bs_mir181_chi %>%
  subset(!duplicated(ago_bs_mir181_chi$BS_ID)) %>%
  left_join(., mir181_chi, by = "BS_ID") %>% makeGRangesFromDataFrame(keep.extra.columns = T)
```

4.2 Combine AGO binding sites with chimeric mir181 with mir181 enriched binding sites

I combine the mir181 Ago binding sites that we obtained above with the binding sites from the mir181 enriched Ago-eCLIP (see Methods/mir181-enriched_binding_site_definition). In order to do that, I first select binding sites from both conditions that do not overlap with any binding site from the other set. For the binding sites that overlap between the two conditions, I select the AGO mir181 binding sites and tag them as occurring in both sets. Then I combine the three subsets. The obtained union of mir181 binding sites from both conditions are our final mir181 binding sites.

```
# -----
# combine mir181 Ago BS and mir181 enriched BS
# -----
```

```

# get only Ago mir181 BS with now overlaps to enriched mir181 BS
only_ago_bs_mir181_chi <- subsetByOverlaps(ago_bs_mir181_chi, mir181_enriched, type = "any", invert = T)
only_ago_bs_mir181_chi$set <- "ago_bs_mir181_chi"

# get only enriched mir181 BS with now overlaps to Ago mir181 BS
only_mir181_enriched <- subsetByOverlaps(mir181_enriched, ago_bs_mir181_chi, type = "any", invert = T)
only_mir181_enriched$set <- "mir181_enriched"

# get only Ago mir181 BS overlapping with mir181 enriched BS
both_mir181_enriched_chi <- subsetByOverlaps(ago_bs_mir181_chi, mir181_enriched, type = "any")
both_mir181_enriched_chi$set <- "ago_bs_mir181_chi&mir181_enriched"

# combine all three sets
mir181_bs <- c(only_ago_bs_mir181_chi, only_mir181_enriched, both_mir181_enriched_chi)

```

5 Combine mir181 binding sites with differential binding sites

Next, I combine the obtained mir181 binding sites with the results we obtained from the differential binding between AGO binding sites and AGO binding sites with mir181 KO (see script Figure1/Differential_Binding_AGO_BS_mir181_KO).

```

# -----
# combine with differential BS
# -----
# get overlaps with diff binding
diff_overlap <- findOverlaps( mir181_bs, makeGRangesFromDataFrame(diff, keep.extra.columns = T) , type = "any")
# add differential information to mir181 binding sites
d <- diff[,9:49]
mcols(mir181_bs) <- cbind(mcols(mir181_bs), d[diff_overlap,])

# add only diff bs (these are Ago binding sites but not mir181 Binding sites)
diff_only <- subsetByOverlaps( makeGRangesFromDataFrame(diff, keep.extra.columns = T), mir181_bs , type = "any")
mir181_bs_diff <- c(mir181_bs, diff_only)

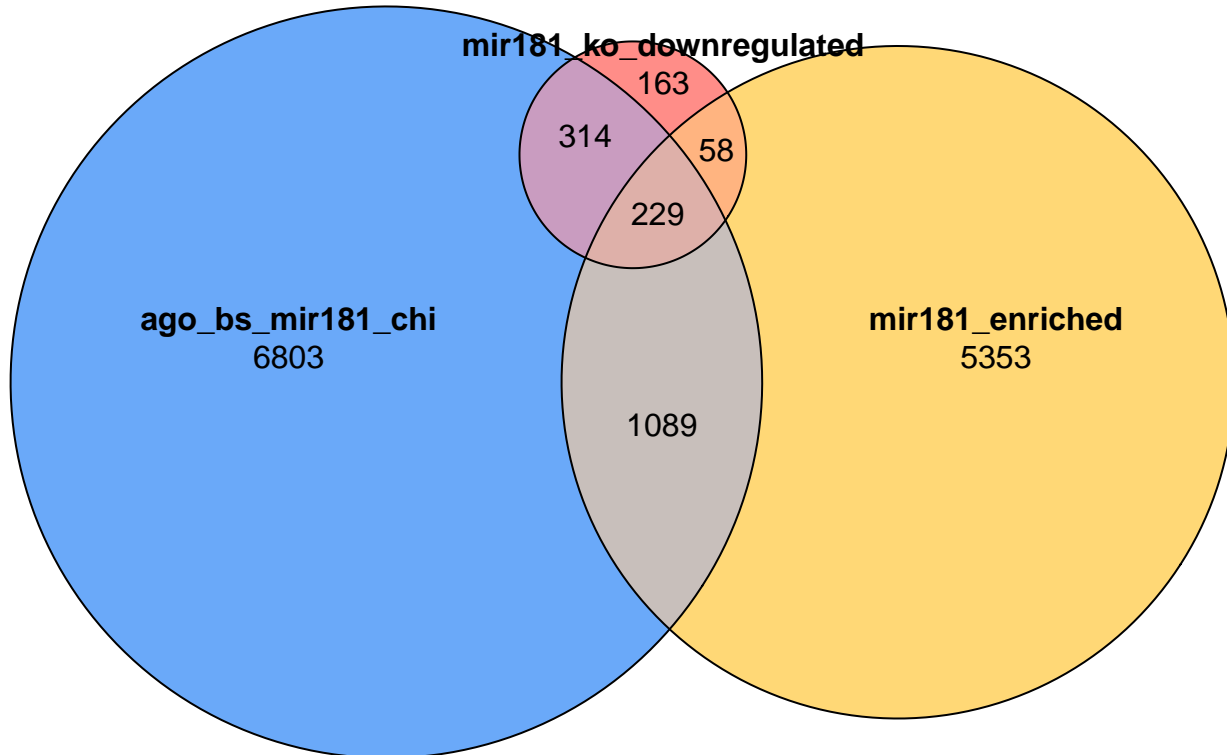
# -----
# make venn diagram
# -----

# select downregulated BS from differential BS
mir181_bs_diff_anygroup <- mir181_bs_diff %>% subset((down == T) | !is.na(set))

# make venn
venn_df <- data.frame(ago_bs_mir181_chi =
  (mir181_bs_diff_anygroup$set == "ago_bs_mir181_chi") | (mir181_bs_diff_anygroup$set == "mir181_enriched") &
  (mir181_bs_diff_anygroup$set == "mir181_enriched") | (mir181_bs_diff_anygroup$set == "mir181_ko_downregulated") &
  (mir181_ko_downregulated == (mir181_bs_diff_anygroup$down == T) )
) %>%
mutate(ago_bs_mir181_chi = case_when(is.na(ago_bs_mir181_chi) ~ F, T~ago_bs_mir181_chi),
  mir181_enriched = case_when(is.na(mir181_enriched) ~ F, T ~ mir181_enriched),
  mir181_ko_downregulated = case_when(is.na(mir181_ko_downregulated) ~ F, T ~ mir181_ko_downregulated)
)

```

```
venn <- eulerr::euler(venn_df)
p <- plot(venn, quantities = T, fills = c( lighten(farbe1, amount = 0.4), lighten(farbe2, amount = 0.4)
p
```



```
pdf(paste0(out, "Venn_overlaps_mirBs.pdf"), height = 3, width = 4 )
p
dev.off()
```

```
## pdf
## 2
```

5.1 Differential binding both sets

Here I look at the overall binding changes between mir181 KO and WT for the three subsets of the mir181 binding sites.

```
# -----
# Compare regulation of BS during mir181 KO of the different subgroups of mir181 binding sites
# -----
names(mir181_bs_diff) <- 1:NROW(mir181_bs_diff)

mir181_bs_diff <- as.data.frame(mir181_bs_diff)

p1 <- ggplot(mir181_bs_diff, aes(x = resBs.log2FoldChange, color = set))+
  geom_vline(xintercept = 0, color = "grey")+
  stat_ecdf()+
  scale_color_manual(values = c(farbe1, farbe14, farbe2, farbeneg))+
  theme_paper()+
```

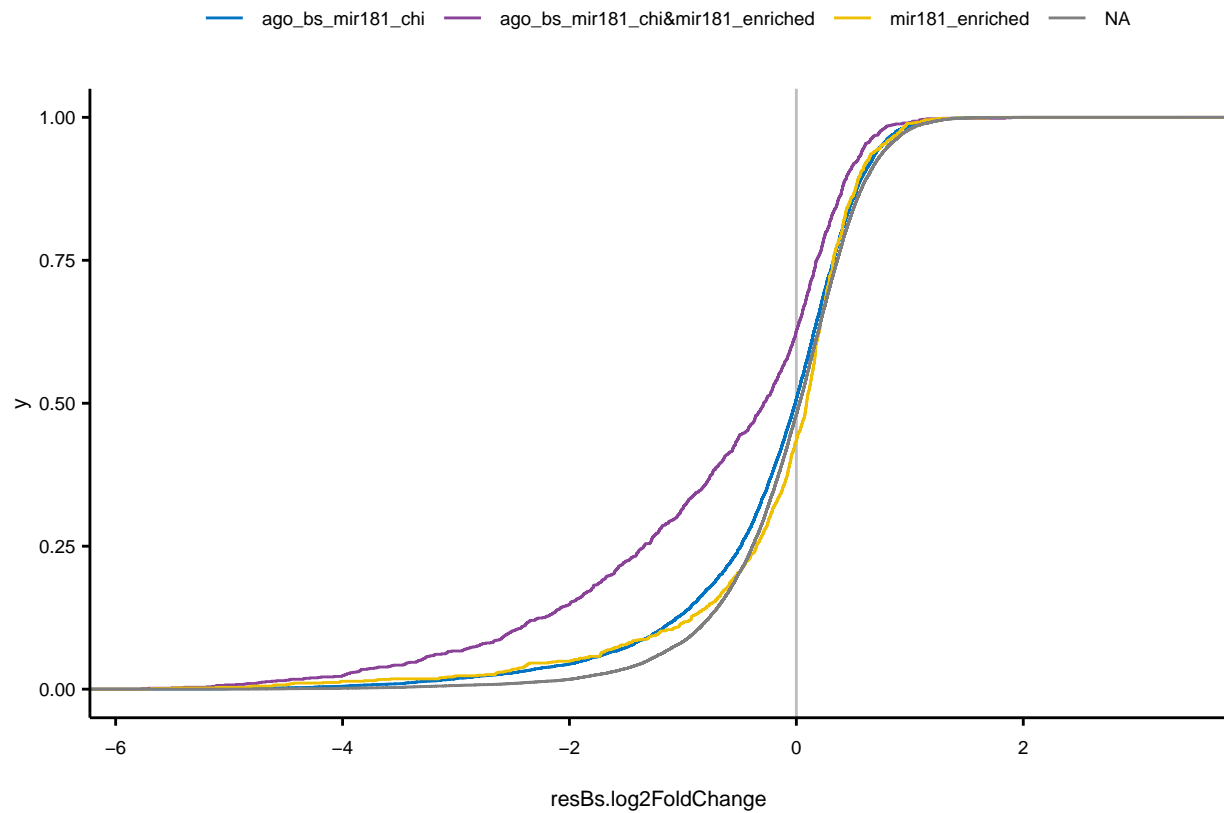
```

theme(legend.position = "top")

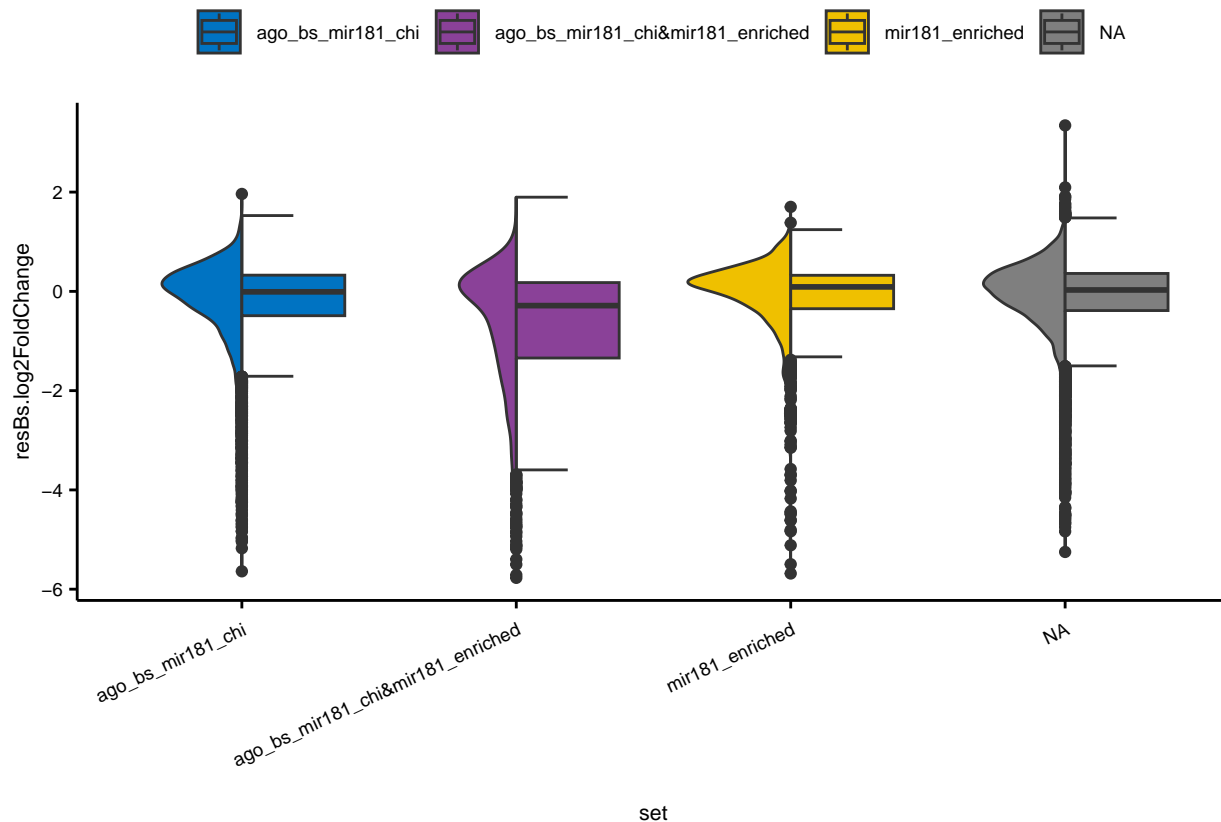
p2 <- ggplot(mir181_bs_diff, aes(y = resBs.log2FoldChange, x = set, fill = set))+
  geom_half_violin()+
  geom_half_boxplot(side = "r")+
  theme_paper()+
  scale_fill_manual(values = c(farbe1, farbe14, farbe2, farbeneg))+
  theme(legend.position = "top")+
  scale_x_discrete(guide = guide_axis(angle = 25))

```

p1



p2



```
ggsave(p1, filename = paste0(out, "ecdf_differntial_binding_vs_mir181BS.pdf"), width = 6, height = 7,
ggsave(p2, filename = paste0(out, "violin_differntial_binding_vs_mir181BS.pdf"), width = 10, height = 10)
```

6 Characterise mir181 binding sites

6.1 mir181 bound genes - Figure 2A?

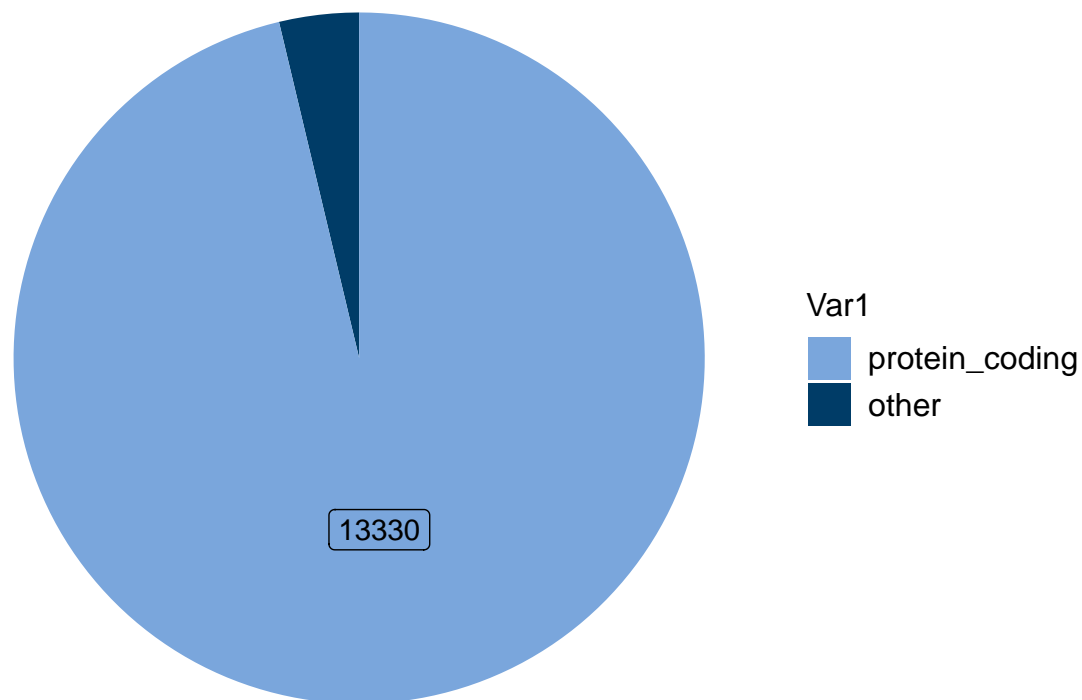
```
names(mir181_bs) <- 1:NROW(mir181_bs)
mir181_bs <- as.data.frame(mir181_bs)

gene_type_df <- mutate(mir181_bs, geneType = case_when(geneType != "protein_coding" ~ "other", T ~ "pro
gene_type_df <- table(gene_type_df$geneType) %>%
  as.data.frame(.)

p <- ggplot(gene_type_df, aes(y=Freq, x="", fill=Var1)) +
  geom_col()+
  coord_polar(theta="y") +
  # xlim(c(2, 4)) +
  geom_label(data = gene_type_df %>% subset(gene_type_df == "protein_coding"), aes(y=Freq, x="", fill=V
    position = position_stack(vjust = 0.5),
    show.legend = FALSE) +
  scale_fill_manual(values = c (farbe6, farbe4)) +
  theme_paper() +
  theme_nice_pie() +
  #theme(legend.position = "none") +
  guides(fill = guide_legend(reverse = TRUE)) +
  labs(y = NULL,
```

```
x = NULL)
```

p



```
ggsave(p, filename = paste0(out, "Figure2A_bound_gene_types_miR181_BS", Sys.Date(), ".pdf"), width = un
```

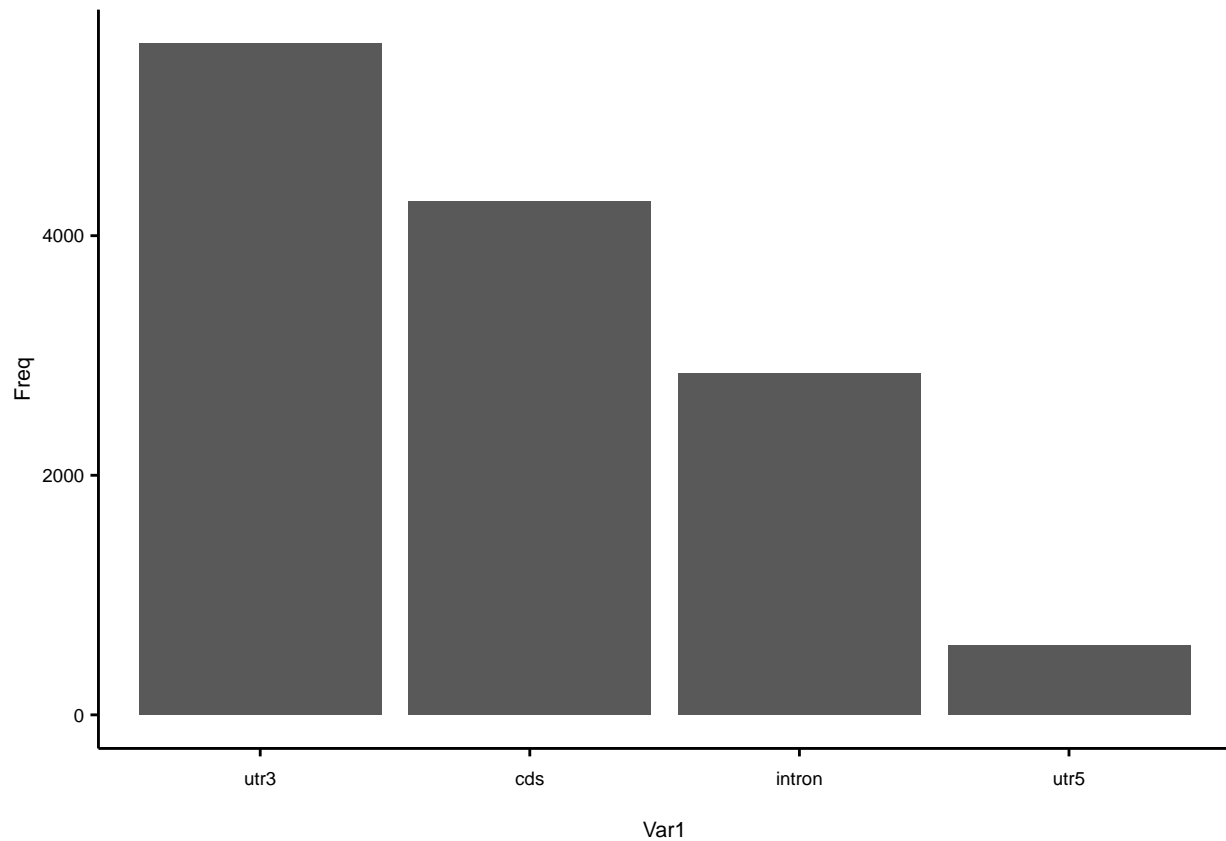
6.2 mir181 bound regions - Figure 2B?

```
gene_region_df <- table(mir181_bs$region) %>%
  as.data.frame(.) %>%
  arrange(desc(Freq))

gene_region_df$Var1 <- factor(gene_region_df$Var1, levels = gene_region_df$Var1)

p <- ggplot(gene_region_df %>% subset(Var1 != "outside"), aes(y=Freq, x=Var1)) +
  geom_col()+
  theme_paper()

p
```

```
ggsave(p, filename = paste0(out, "Figure2B_bound_gene_regions_miR181_BS", Sys.Date(), ".pdf"), width = 10, height = 10)
```

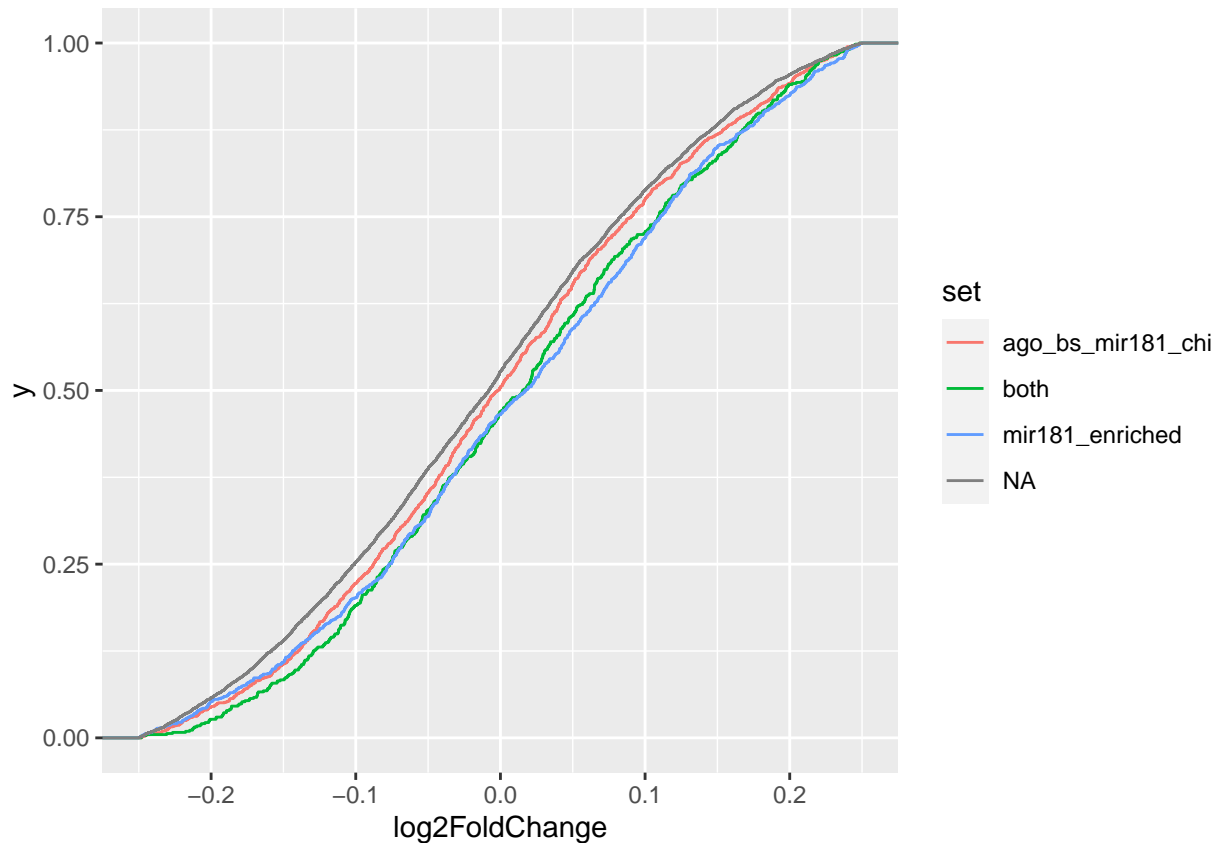
6.2.1 Venn bound genes from both sets

6.2.2 Ribofootprint both sets

```
# TODO need to be other script later
rpf <- read.csv("/Users/melinaklostermann/Documents/projects/AgoCLIP_miR181/files_RNaseq_RiboP/RPF_master.csv")
rpf <- mutate(rpf, Gene = substr(Gene,1,18))

rpf <- rpf %>% mutate(., set =
  case_when(Gene %in% both_mir181_enriched_chi$geneID ~ "both",
            (Gene %in% ago_bs_mir181_chi$geneID) & !(Gene %in% both_mir181_enriched_chi$geneID) ~ "ago",
            (Gene %in% mir181_enriched$geneID) & !(Gene %in% both_mir181_enriched_chi$geneID) ~ "mir",
            TRUE ~ "other")
)

ggplot(rpf, aes(x = log2FoldChange, color = set))+
  stat_ecdf()+
  xlim(-0.25, 0.25)
```



7 Save output

```
saveRDS(mir181_bs, paste0(out, "mir181_bs.rds"))
```

8 Session Info

```
sessionInfo()

## R version 4.2.2 (2022-10-31)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur ... 10.16
##
## Matrix products: default
## BLAS: /Library/Frameworks/R.framework/Versions/4.2/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.2/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats4      stats      graphics  grDevices  utils      datasets  methods
## [8] base
##
## other attached packages:
## [1] gghalves_0.1.4      eulerr_7.0.0        colorspace_2.1-0
```

```

## [4] GenomicRanges_1.50.2 GenomeInfoDb_1.34.7 IRanges_2.32.0
## [7] S4Vectors_0.36.1      BiocGenerics_0.44.0 forcats_0.5.2
## [10] stringr_1.5.0          dplyr_1.0.10        purrr_1.0.1
## [13] readr_2.1.3            tidyr_1.3.0          tibble_3.1.8
## [16] ggplot2_3.4.0          tidyverse_1.3.2      knitr_1.42
##
## loaded via a namespace (and not attached):
## [1] httr_1.4.4              jsonlite_1.8.4         carData_3.0-5
## [4] modelr_0.1.10           assertthat_0.2.1       highr_0.10
## [7] GenomeInfoDbData_1.2.9 googlesheets4_1.0.1    cellranger_1.1.0
## [10] yaml_2.3.7              pillar_1.8.1           backports_1.4.1
## [13] glue_1.6.2              digest_0.6.31          XVector_0.38.0
## [16] ggsignif_0.6.4          polyclip_1.10-4        rvest_1.0.3
## [19] htmltools_0.5.4         pkgconfig_2.0.3        broom_1.0.3
## [22] haven_2.5.1             zlibbioc_1.44.0        scales_1.2.1
## [25] tzdb_0.3.0              timechange_0.2.0       googledrive_2.0.0
## [28] farver_2.1.1            generics_0.1.3         car_3.1-1
## [31] ellipsis_0.3.2          ggpubr_0.5.0           withr_2.5.0
## [34] cli_3.6.0               magrittr_2.0.3         crayon_1.5.2
## [37] readxl_1.4.1            evaluate_0.20          fs_1.6.0
## [40] fansi_1.0.4             rstatix_0.7.1          xml2_1.3.3
## [43] textshaping_0.3.6       tools_4.2.2            hms_1.1.2
## [46] gargle_1.2.1            lifecycle_1.0.3        munsell_0.5.0
## [49] reprex_2.0.2            compiler_4.2.2         systemfonts_1.0.4
## [52] rlang_1.0.6             grid_4.2.2             RCurl_1.98-1.9
## [55] rstudioapi_0.14         labeling_0.4.2         bitops_1.0-7
## [58] rmarkdown_2.20          gtable_0.3.1           abind_1.4-5
## [61] DBI_1.1.3               R6_2.5.1               lubridate_1.9.1
## [64] fastmap_1.1.0           utf8_1.2.2             ragg_1.2.5
## [67] polylabelr_0.2.0        stringi_1.7.12         Rcpp_1.0.10
## [70] vctrs_0.5.2             dbplyr_2.3.0           tidysselect_1.2.0
## [73] xfun_0.36

```