irCLIPv2 dataset from 4 RBPs and 3 RDAPs in HEK293T and HepG2 from three RNP subzones

Luca Ducoli

This is the pipeline used to analyze the irCLIPv2 datasets for 4 RBPs and 3 RDAPs from three different gel sections (RNP subzone #1-3). The experiment was performed in HEK293T. Mapping, counting, and Differential enrichment analysis between RNP subzones and no-UV samples was performed using a custom snakemake pipeline. Please see this Github link for more information.

1. Load the data

After DEWseq analysis against noUV using a custom snakemake pipeline, we first characterized the significant regions.

```
#Load the libraries
library(RIdeogram)
library(RColorBrewer)
library(scales)
library(ggplot2)
library(tidyverse)
library(plyranges)
library(Repitools)
library(GenomicRanges)
library(grid)
library(gridExtra)
library(ComplexHeatmap)
library(ChIPpeakAnno)
library(eulerr)
library(ggridges)
library(AnnotationDbi)
library(gintools)
library(EnrichedHeatmap)
library(rtracklayer)
library(caTools)
library(memes)
library(universalmotif)
library(magrittr)
library(BSgenome.Hsapiens.UCSC.hg38)
library(paletteer)
library(gUtils)
library(circlize)
library(multcomp)
```

```
#Load the binding regions for each RBP
FUS <- read.delim("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/1_Regions_DEW-Seq/
    deseq2_norm/2_Subzone_sum/3_pvalue_0.05_fc1/FUS_DEWSeq_sign_regions_sum_categorization.txt",
    header = TRUE)
FUS$rbp <- "FUS"
colnames(FUS)[5] <- "region_strand"
FUS <- subset(FUS, chromosome != "chrM")
```

```
HNRNPC <- read.delim("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/1_Regions_DEW-
    Seq/deseq2_norm/2_Subzone_sum/3_pvalue_0.05_fc1/HNRNPC_DEWSeq_sign_regions_sum_categorization
    .txt", header = TRUE)
HNRNPC$rbp <- "HNRNPC"
colnames(HNRNPC)[5] <- "region_strand"
HNRNPC <- subset(HNRNPC, chromosome != "chrM")

HNRNPM <- read.delim("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/1_Regions_DEW-
    Seq/deseq2_norm/2_Subzone_sum/3_pvalue_0.05_fc1/HNRNPM_DEWSeq_sign_regions_sum_categorization
    .txt", header = TRUE)
HNRNPM$rbp <- "HNRNPM"
colnames(HNRNPM)[5] <- "region_strand"
HNRNPM <- subset(HNRNPM, chromosome != "chrM")

HNRNPU <- read.delim("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/1_Regions_DEW-
    Seq/deseq2_norm/2_Subzone_sum/3_pvalue_0.05_fc1/HNRNPU_DEWSeq_sign_regions_sum_categorization
    .txt", header = TRUE)
HNRNPU$rbp <- "HNRNPU"
colnames(HNRNPU)[5] <- "region_strand"
HNRNPU <- subset(HNRNPU, chromosome != "chrM")

ELAVL1 <- read.delim("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/1_Regions_DEW-
    Seq/deseq2_norm/2_Subzone_sum/3_pvalue_0.05_fc1/ELAVL1_DEWSeq_sign_regions_sum_categorization
    .txt", header = TRUE)
ELAVL1$rbp <- "ELAVL1"
colnames(ELAVL1)[5] <- "region_strand"
ELAVL1 <- subset(ELAVL1, chromosome != "chrM")

HNRNPFH <- read.delim("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/1_Regions_DEW-
    Seq/deseq2_norm/2_Subzone_sum/3_pvalue_0.05_fc1/HNRNPFH_DEWSeq_sign_regions_sum_
    categorization.txt", header = TRUE)
HNRNPFH$rbp <- "HNRNPFH"
colnames(HNRNPFH)[5] <- "region_strand"
HNRNPFH <- subset(HNRNPFH, chromosome != "chrM")

KHSRP <- read.delim("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/1_Regions_DEW-
    Seq/deseq2_norm/2_Subzone_sum/3_pvalue_0.05_fc1/KHSRP_DEWSeq_sign_regions_sum_categorization.
    txt", header = TRUE)
KHSRP$rbp <- "KHSRP"
colnames(KHSRP)[5] <- "region_strand"
KHSRP <- subset(KHSRP, chromosome != "chrM")

THOC4 <- read.delim("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/1_Regions_DEW-
    Seq/deseq2_norm/2_Subzone_sum/3_pvalue_0.05_fc1/THOC4_DEWSeq_sign_regions_sum_categorization.
    txt", header = TRUE)
THOC4$rbp <- "THOC4"
colnames(THOC4)[5] <- "region_strand"
THOC4 <- subset(THOC4, chromosome != "chrM")
```

2. Region characterization

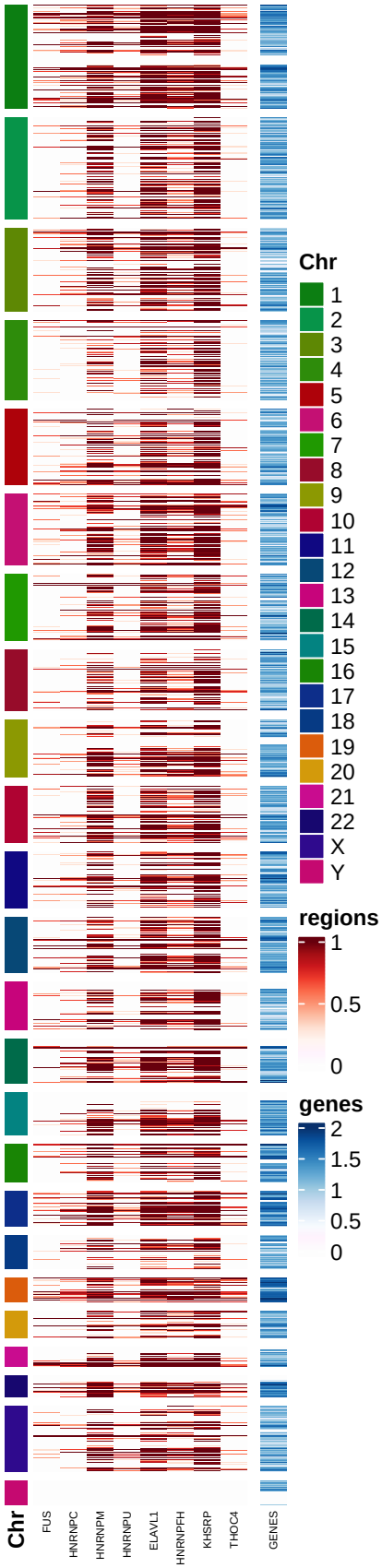
Heatmap of binding density throughout the genome

The first visualization is the distribution of the significant regions across the genome.

```
#Get gene density and human karyotype from the package RIdeogram
data(gene_density, package="RIdeogram")
gene_density2 <- gene_density[,-4]
data(human_karyotype, package="RIdeogram")
human_karyotype$Chr <- paste0("chr", human_karyotype$Chr, sep = "")

gd_function <- function (br, gd_all, rbp) {
```

```
  rbp.gd <- br %>% dplyr::select(chromosome,region_begin,region_end)
  rbp.gd <- unique(rbp.gd)
  colnames(rbp.gd) <- c("Chr", "Start", "End")
  rbp.gd$Chr <- gsub("chr", "", rbp.gd$Chr)
  rbp.gd2 <- bedtoolsr::bt.intersect(gd_all, rbp.gd, C = T)
  colnames(rbp.gd2) <- c("Chr", "Start", "End", rbp)
  return(rbp.gd2)
}

FUS.gd <- gd_function(FUS, gene_density2, "FUS")
HNRNPC.gd <- gd_function(HNRNPC, gene_density2, "HNRNPC")
HNRNPM.gd <- gd_function(HNRNPM, gene_density2, "HNRNPM")
HNRNPU.gd <- gd_function(HNRNPU, gene_density2, "HNRNPU")
ELAVL1.gd <- gd_function(ELAVL1, gene_density2, "ELAVL1")
HNRNPFH.gd <- gd_function(HNRNPFH, gene_density2, " HNRNPFH")
KHSRP.gd <- gd_function(KHSRP, gene_density2, "KHSRP")
THOC4.gd <- gd_function(THOC4, gene_density2, "THOC4")

#Combine the region densities
all.rbp <- cbind(FUS.gd[4], HNRNPC.gd[4], HNRNPM.gd[4], HNRNPU.gd[4], ELAVL1.gd[4], HNRNPFH.gd
    [4], KHSRP.gd[4], THOC4.gd[4])
rownames(all.rbp) <- paste0("row_", seq(nrow(all.rbp)))
all.rbp <- log10(all.rbp)
all.rbp[all.rbp==-Inf] <- 0

#Create annotation for the Heatmap
annotation <- FUS.gd[1]
annotation$Chr <- factor(annotation$Chr, levels = c(1:22,"X","Y"))
rownames(annotation) <- row.names(all.rbp)

my.breaks <- c(seq(0, 1, by=0.0125))
my.colors <- c(colorRampPalette(colors = c("#fdfdfd", "#fff3fd", brewer.pal(9, "Reds")))(length(
    my.breaks)))

#Generate Heatmap
p <- pheatmap(
  mat               = all.rbp,
  color = my.colors,
  breaks = my.breaks,
  name = "regions",
  annotation_row = annotation,
  show_colnames     = TRUE,
  show_rownames     = FALSE,
  drop_levels       = TRUE,
  fontsize          = 5,
  cluster_rows      = FALSE,
  cluster_cols      = FALSE,
  scale             = "none",
  gaps_row = head(as.numeric(cumsum(table(annotation$Chr))), -1)
)

#Add the density of the genes to the generated heatmap
gene_anno <- GFFex(input = "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/2_
    Characterization/gencode.v39.annotation.gff3.gz", karyotype = "/Users/lducoli/Documents/
    Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/2_Characterization/human_karyotype.txt",
    feature = "gene", window = 1000000)
colnames(gene_anno) <- c("Chr", "Start", "End", "GENES")
gene_anno$GENES <- log10(gene_anno$GENES)
gene_anno$GENES[is.infinite(gene_anno$GENES)] <- 0
gene_anno$Chr <- gsub("chr", "", gene_anno$Chr)
gene_anno$Chr <- factor(gene_anno$Chr, c(1:22,"X","Y"), ordered=TRUE)
gene_anno <- gene_anno[do.call(order, gene_anno[, c("Chr", "Start", "End", "GENES")]), ]
genes <- gene_anno[4]
rownames(genes) <- paste0("row_", seq(nrow(genes)))

#Genes
annotation2 <- gene_anno[1]
rownames(annotation2) <- row.names(genes)
```

```
my.breaks <- c(seq(0, 2, by=0.025))
my.colors <- c(colorRampPalette(colors = c("#fdfdfd", "#fff3fd", brewer.pal(9, "Blues")))(length(
    my.breaks)))

p2 <- pheatmap(
  mat               = genes,
  name = "genes",
  color = my.colors,
  breaks = my.breaks,
  show_colnames     = TRUE,
  show_rownames     = FALSE,
  drop_levels       = TRUE,
  fontsize          = 5,
  cluster_rows      = FALSE,
  cluster_cols      = FALSE,
  scale             = "none",
  gaps_row = head(as.numeric(cumsum(table(annotation$Chr))), -1)
)

p + p2
```

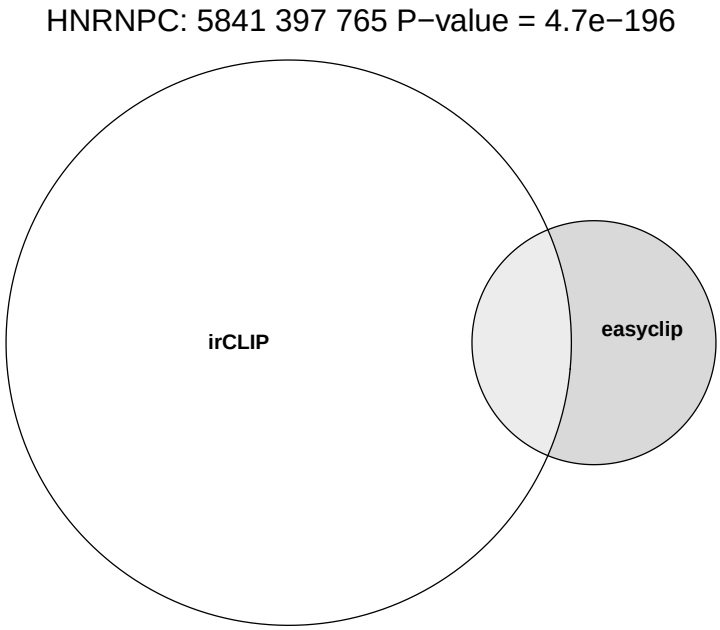

Chr
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
X
Y
regions
1
0.5
0
genes
2
1.5
1
0.5
0
Chr
FUS
HNRNPC
HNRNPM
HNRNPU
ELAVL1
HNRNPFH
KHSRP
THOC4
GENES

```
# Save the heatmap
pdf("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/2_Characterization/3_pvalue_0.05
    _fc1/HM_Gene_density.pdf", height = 20, width = 5)
p + p2
dev.off()
```

Comparison to easyCLIP

Next, we compared the regions of HNRNPC to the regions identified by Porter et al. using the easyCLIP protocol (PMID: 33692367). These regions were determined using the MACS2 tool.

```
#Easyclip data from Porter et al.
HNRNPC.EC <- read.delim("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones
    /2_Characterization/easyclip/HNRNPC_genomics_10perc_peaks.narrowPeak.txt", header = TRUE)

overlap_function <- function (br, EC) {
  rbp <- br %>% dplyr::select(chromosome,region_begin,region_end)
  rbp$region_begin <- round(((rbp$region_begin+rbp$region_end)/2)-250,0)
  rbp$region_end <- rbp$region_begin+500
  rbp <- GRanges(rbp[,grep("chromosome", colnames(rbp))], IRanges(start=rbp[,grep("region_begin",
      colnames(rbp))], end=rbp[,grep("region_end", colnames(rbp))]))
  rbp <- unique(rbp)
  EC <- EC %>% dplyr::select(chrom, start, end)
  EC$start <- round(((EC$start+EC$end)/2)-250,0)
  EC$end <- EC$start+500
  EC <- GRanges(EC[,grep("chrom", colnames(EC))], IRanges(start=EC[,grep("start", colnames(EC))],
      end=EC[,grep("end", colnames(EC))]))
  EC <- unique(EC)
  gr.list <- list(rbp,EC)
  return(gr.list)
}

HNRNPC.cp <- overlap_function(HNRNPC, HNRNPC.EC)

HNRNPC_res <- makeVennDiagram(Peaks=HNRNPC.cp, NameOfPeaks=c("HNRNPC", "easyclip"), plot = FALSE)

#HNRNPC
HNRNPC_high_venn <- eulerr::euler(c(irCLIP = length(HNRNPC$regionID)-HNRNPC_res$vennCounts
    [,3][4], easyclip = HNRNPC_res$vennCounts[,3][2], "irCLIP&easyclip" = HNRNPC_res$vennCounts
    [,3][4]), shape = "ellipse")
HNRNPC_plot <- plot(HNRNPC_high_venn, main = paste("HNRNPC:", length(HNRNPC$regionID)-HNRNPC_res
    $vennCounts[,3][4], HNRNPC_res$vennCounts[,3][4], HNRNPC_res$vennCounts[,3][2], "P-value =",
    signif(HNRNPC_res$p.value[[3]], digits=2)))

HNRNPC_plot
```




```
pdf("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/2_Characterization/3_pvalue_0.05
    _fc1/Easyclip_comparison.pdf", width = 7, height = 5)
HNRNPC_plot
dev.off()
```

Piechart of the binding distribution

Finally, the proportion of binding regions of the RBPs was visualized.

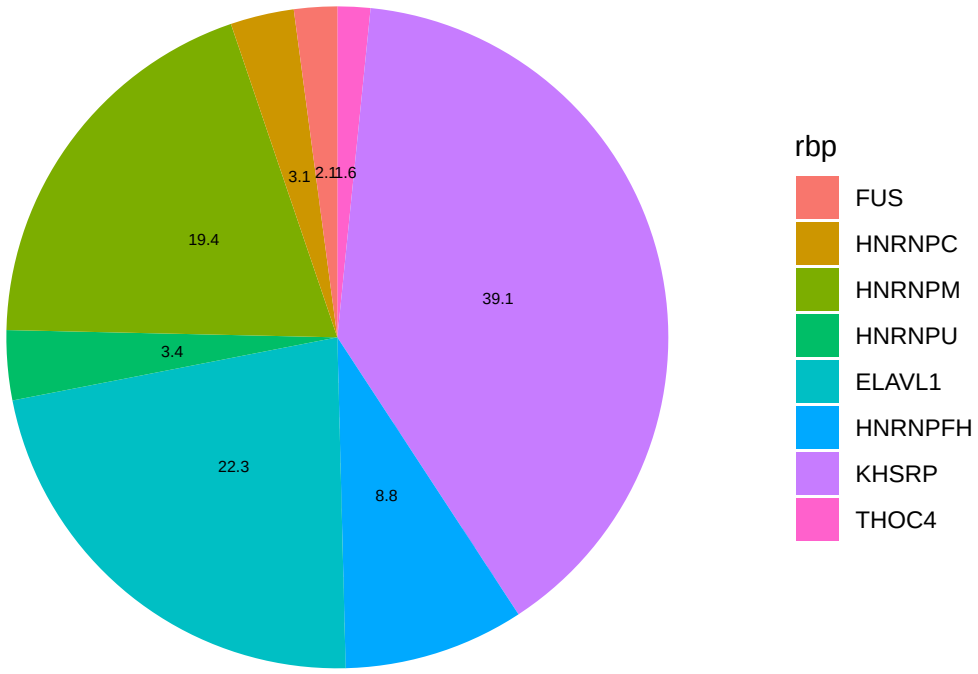
```
#Store regions as Granges
FUS.gr <- GRanges(FUS[,grep("chromosome", colnames(FUS))],
                  IRanges(start=FUS[,grep("region_begin", colnames(FUS))], end=FUS[,grep("region_
    end", colnames(FUS))]),
                  names = FUS[,grep("regionID", colnames(FUS))], strand = FUS[,grep("region_
    strand", colnames(FUS))],
                  rbp = "FUS")

HNRNPC.gr <- GRanges(HNRNPC[,grep("chromosome", colnames(HNRNPC))],
                  IRanges(start=HNRNPC[,grep("region_begin", colnames(HNRNPC))], end=HNRNPC[,grep
    ("region_end", colnames(HNRNPC))]),
                  names = HNRNPC[,grep("regionID", colnames(HNRNPC))], strand = HNRNPC[,grep("
    region_strand", colnames(HNRNPC))],
                  rbp = "HNRNPC")

HNRNPM.gr <- GRanges(HNRNPM[,grep("chromosome", colnames(HNRNPM))],
                  IRanges(start=HNRNPM[,grep("region_begin", colnames(HNRNPM))], end=HNRNPM[,grep
    ("region_end", colnames(HNRNPM))]),
                  names = HNRNPM[,grep("regionID", colnames(HNRNPM))], strand = HNRNPM[,grep("
    region_strand", colnames(HNRNPM))],
                  rbp = "HNRNPM")

HNRNPU.gr <- GRanges(HNRNPU[,grep("chromosome", colnames(HNRNPU))],
                  IRanges(start=HNRNPU[,grep("region_begin", colnames(HNRNPU))], end=HNRNPU[,grep
    ("region_end", colnames(HNRNPU))]),
                  names = HNRNPU[,grep("regionID", colnames(HNRNPU))], strand = HNRNPU[,grep("
    region_strand", colnames(HNRNPU))],
                  rbp = "HNRNPU")
```

```
ELAVL1.gr <- GRanges(ELAVL1[,grep("chromosome", colnames(ELAVL1))],
                    IRanges(start=ELAVL1[,grep("region_begin", colnames(ELAVL1))], end=ELAVL1[,grep
    ("region_end", colnames(ELAVL1))]),
                    names = ELAVL1[,grep("regionID", colnames(ELAVL1))], strand = ELAVL1[,grep("
    region_strand", colnames(ELAVL1))],
                    rbp = "ELAVL1")

HNRNPFH.gr <- GRanges(HNRNPFH[,grep("chromosome", colnames(HNRNPFH))],
                    IRanges(start=HNRNPFH[,grep("region_begin", colnames(HNRNPFH))], end=HNRNPFH[,
    grep("region_end", colnames(HNRNPFH))]),
                    names = HNRNPFH[,grep("regionID", colnames(HNRNPFH))], strand = HNRNPFH[,grep("
    region_strand", colnames(HNRNPFH))],
                    rbp = "HNRNPFH")

KHSRP.gr <- GRanges(KHSRP[,grep("chromosome", colnames(KHSRP))],
                    IRanges(start=KHSRP[,grep("region_begin", colnames(KHSRP))], end=KHSRP[,grep("
    region_end", colnames(KHSRP))]),
                    names = KHSRP[,grep("regionID", colnames(KHSRP))], strand = KHSRP[,grep("region
    _strand", colnames(KHSRP))],
                    rbp = "KHSRP")

THOC4.gr <- GRanges(THOC4[,grep("chromosome", colnames(THOC4))],
                    IRanges(start=THOC4[,grep("region_begin", colnames(THOC4))], end=THOC4[,grep("
    region_end", colnames(THOC4))]),
                    names = THOC4[,grep("regionID", colnames(THOC4))], strand = THOC4[,grep("region
    _strand", colnames(THOC4))],
                    rbp = "THOC4")

HNRNPC.gr
```

```
## GRanges object with 6238 ranges and 2 metadata columns:
##          seqnames              ranges strand |                names
##             <Rle>           <IRanges>  <Rle> |          <character>
##      [1]     chr1 100016040-100016110      + | chr1_100016040_10001..
##      [2]     chr1 100016057-100016127      + | chr1_100016057_10001..
##      [3]     chr1 100018855-100018905      + | chr1_100018855_10001..
##      [4]     chr1 100018861-100018911      + | chr1_100018861_10001..
##      [5]     chr1     1010359-1010409      + | chr1_1010359_1010409..
##      ...      ...                 ...    ... .                   ...
##   [6234]     chrX   74586715-74586765      - | chrX_74586715_745867..
##   [6235]     chrX   74588275-74588345      - | chrX_74588275_745883..
##   [6236]     chrX   74588435-74588505      - | chrX_74588435_745885..
##   [6237]     chrX   77903736-77903786      + | chrX_77903736_779037..
##   [6238]     chrX   80324393-80324483      - | chrX_80324393_803244..
##                rbp
##          <character>
##      [1]      HNRNPC
##      [2]      HNRNPC
##      [3]      HNRNPC
##      [4]      HNRNPC
##      [5]      HNRNPC
##      ...         ...
##   [6234]      HNRNPC
##   [6235]      HNRNPC
##   [6236]      HNRNPC
##   [6237]      HNRNPC
##   [6238]      HNRNPC
##   -------
##   seqinfo: 23 sequences from an unspecified genome; no seqlengths
```

```
#Generate a reduced data.frame of the regions - if you want to run it uncomment it (take aroung
    40min...)
# u = unique(c(FUS.gr, HNRNPC.gr, HNRNPM.gr, HNRNPU.gr, ELAVL1.gr, HNRNPFH.gr, KHSRP.gr, THOC4.gr
    ))
```

```
# geneRange <- sortSeqlevels(u)
# geneRange <- sort(geneRange)
# geneReduce <- GenomicRanges::reduce(geneRange, drop.empty.ranges = TRUE, with.revmap = TRUE,
    min.gapwidth = 1)
# mcols(geneReduce)$rbp <- "Undefined"
#
# pb <- txtProgressBar(min = 0, max = length(geneReduce), style = 3)
# for (i in seq_len(length(geneReduce))) {
#     mergeInd <- unlist(mcols(geneReduce)[i, 1])
#     mcols(geneReduce)[i, "rbp"] <- paste(mcols(geneRange)[mergeInd, "rbp"], collapse = ", ")
#     setTxtProgressBar(pb = pb, value = i)
# }
#
# regionRes <- as.data.frame(geneReduce)
# regionRes <- regionRes[,-6]
# regionRes$regions <- paste("Region", 1:length(regionRes$seqnames), sep = "_")
# write.table(regionRes, "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/2_
    Characterization/3_pvalue_0.05_fc1/Confident_region.txt", quote = F, sep = "\t", row.names =
    F)

regionRes <- read.delim("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/2_
    Characterization/3_pvalue_0.05_fc1/Confident_region.txt", header = TRUE)

#Generate Piechart
regionRes2 <- regionRes %>% mutate(rbp = str_split(rbp, ", ")) %>% unnest(rbp)
regionRes3 <- regionRes2 %>% dplyr::count(rbp)
regionRes3$rbp <- factor(regionRes3$rbp)
regionRes3$rbp <- factor(regionRes3$rbp, levels = c("FUS", "HNRNPC", "HNRNPM", "HNRNPU", "ELAVL1"
    , "HNRNPFH", "KHSRP", "THOC4"))

regionRes3 <- regionRes3 %>%
  arrange(desc(rbp)) %>%
  mutate(prop = round((n / sum(regionRes3$n))*100,1)) %>%
  mutate(ypos = cumsum(prop)- 0.5*prop )

ggplot(regionRes3, aes(x="", y=prop, fill=rbp))+
geom_bar(width = 1, stat = "identity") +
    coord_polar("y", start=0) +
  theme_minimal()+
  theme(
  axis.title.x = element_blank(),
  axis.title.y = element_blank(),
  panel.border = element_blank(),
  panel.grid=element_blank(),
  axis.ticks = element_blank(),
  plot.title=element_text(size=14, face="bold")
  )+
  theme(axis.text.x=element_blank()) +
  geom_text(aes(y = ypos, label = prop), color = "black", size=2)
```




```
pdf("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/2_Characterization/3_pvalue_0.05
    _fc1/Pie_chart.pdf", width = 5, height = 5)
ggplot(regionRes3, aes(x="", y=prop, fill=rbp))+
geom_bar(width = 1, stat = "identity") +
    coord_polar("y", start=0) +
  theme_minimal()+
  theme(
  axis.title.x = element_blank(),
  axis.title.y = element_blank(),
  panel.border = element_blank(),
  panel.grid=element_blank(),
  axis.ticks = element_blank(),
  plot.title=element_text(size=14, face="bold")
  )+
  theme(axis.text.x=element_blank()) +
  geom_text(aes(y = ypos, label = prop), color = "black", size=2)
dev.off()
```

3. Profile plot of RBPs around the highest nucleotide in the lowest section of the gel

Here, we looked whether irCLIPv2 could generate a 3'shift that can be reconnected to the presence of another RBP.

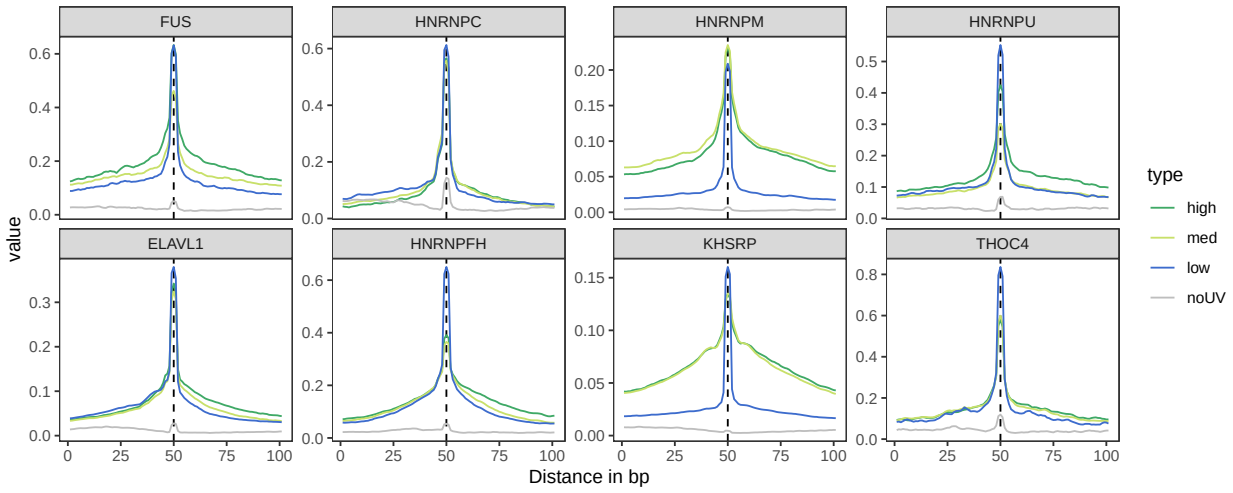
```
#Function for the generation of the profile plots
shift3p <- function(data, rbp) {
  cat("Prepare granges\n")
```

```
p <- subset(data, region_strand == "+" & chromosome != "chrM")
gr.p <- GRanges(p[,grep("chr_S1", colnames(p))],
                IRanges(start=p[,grep("begin_S1", colnames(p))],
                        end=p[,grep("end_S1", colnames(p))]),
                names = p[,grep("max_S1_id", colnames(p))],
                strand = p[,grep("strand_S1", colnames(p))],
                category = p[,grep("category", colnames(p))])
m <- subset(data, region_strand == "-" & chromosome != "chrM")
gr.m <- GRanges(m[,grep("chr_S1", colnames(m))],
                IRanges(start=m[,grep("begin_S1", colnames(m))],
                        end=m[,grep("end_S1", colnames(m))]),
                names = m[,grep("max_S1_id", colnames(m))],
                strand = m[,grep("strand_S1", colnames(m))],
                category = m[,grep("category", colnames(m))])
start(gr.p) <- (start(gr.p)+end(gr.p))/2
end(gr.p) <- start(gr.p)

start(gr.m) <- (start(gr.m)+end(gr.m))/2
end(gr.m) <- start(gr.m)

cat("Load bigwig files\n")
setwd("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/1_Bigwig_sum")
bw.S1.r1.p <- import.bw(paste(rbp, "293T_UVC_S1_R1.p.scaleddewseq.bw", sep = "_"))
bw.S1.r1.m <- import.bw(paste(rbp, "293T_UVC_S1_R1.m.scaleddewseq.bw", sep = "_"))
bw.S1.r2.p <- import.bw(paste(rbp, "293T_UVC_S1_R2.p.scaleddewseq.bw", sep = "_"))
bw.S1.r2.m <- import.bw(paste(rbp, "293T_UVC_S1_R2.m.scaleddewseq.bw", sep = "_"))
bw.S2.r1.p <- import.bw(paste(rbp, "293T_UVC_S2_R1.p.scaleddewseq.bw", sep = "_"))
bw.S2.r1.m <- import.bw(paste(rbp, "293T_UVC_S2_R1.m.scaleddewseq.bw", sep = "_"))
bw.S2.r2.p <- import.bw(paste(rbp,"293T_UVC_S2_R2.p.scaleddewseq.bw", sep = "_"))
bw.S2.r2.m <- import.bw(paste(rbp, "293T_UVC_S2_R2.m.scaleddewseq.bw", sep = "_"))
bw.S3.r1.p <- import.bw(paste(rbp, "293T_UVC_S3_R1.p.scaleddewseq.bw", sep = "_"))
bw.S3.r1.m <- import.bw(paste(rbp, "293T_UVC_S3_R1.m.scaleddewseq.bw", sep = "_"))
bw.S3.r2.p <- import.bw(paste(rbp,"293T_UVC_S3_R2.p.scaleddewseq.bw", sep = "_"))
bw.S3.r2.m <- import.bw(paste(rbp, "293T_UVC_S3_R2.m.scaleddewseq.bw", sep = "_"))
noUV.bw.r1.p <- import.bw(paste(rbp, "293T_noUV_R1.p.scaleddewseq.bw", sep = "_"))
noUV.bw.r1.m <- import.bw(paste(rbp, "293T_noUV_R1.m.scaleddewseq.bw", sep = "_"))
noUV.bw.r2.p <- import.bw(paste(rbp, "293T_noUV_R2.p.scaleddewseq.bw", sep = "_"))
noUV.bw.r2.m <- import.bw(paste(rbp, "293T_noUV_R2.m.scaleddewseq.bw", sep = "_"))

cat("Create profile matrices\n")
S1.mat1.p = normalizeToMatrix(bw.S1.r1.p, gr.p, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
S1.mat1.m = normalizeToMatrix(bw.S1.r1.m, gr.m, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
S1.mat2.p = normalizeToMatrix(bw.S1.r2.p, gr.p, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
S1.mat2.m = normalizeToMatrix(bw.S1.r2.m, gr.m, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)

S2.mat1.p = normalizeToMatrix(bw.S2.r1.p, gr.p, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
S2.mat1.m = normalizeToMatrix(bw.S2.r1.m, gr.m, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
S2.mat2.p = normalizeToMatrix(bw.S2.r2.p, gr.p, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
S2.mat2.m = normalizeToMatrix(bw.S2.r2.m, gr.m, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)

S3.mat1.p = normalizeToMatrix(bw.S3.r1.p, gr.p, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
S3.mat1.m = normalizeToMatrix(bw.S3.r1.m, gr.m, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
S3.mat2.p = normalizeToMatrix(bw.S3.r2.p, gr.p, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
S3.mat2.m = normalizeToMatrix(bw.S3.r2.m, gr.m, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)

noUV.mat1.p = normalizeToMatrix(noUV.bw.r1.p, gr.p, value_column = "score", extend = 200, mean_
```

```
    mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
noUV.mat1.m = normalizeToMatrix(noUV.bw.r1.m, gr.m, value_column = "score", extend = 200, mean_
    mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
noUV.mat2.p = normalizeToMatrix(noUV.bw.r2.p, gr.p, value_column = "score", extend = 200, mean_
    mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
noUV.mat2.m = normalizeToMatrix(noUV.bw.r2.m, gr.m, value_column = "score", extend = 200, mean_
    mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)

cat("Merge replicates\n")
S1.mat.p = (S1.mat1.p[]+S1.mat2.p[])/2
rownames(S1.mat.p) <- gr.p$names

S2.mat.p = (S2.mat1.p[]+S2.mat2.p[])/2
rownames(S2.mat.p) <- gr.p$names

S3.mat.p = (S3.mat1.p[]+S3.mat2.p[])/2
rownames(S3.mat.p) <- gr.p$names

noUV.mat.p = (noUV.mat1.p[]+noUV.mat2.p[])/2
rownames(noUV.mat.p) <- gr.p$names

S1.mat.m = (S1.mat1.m[]+S1.mat2.m[])/2
rownames(S1.mat.m) <- gr.m$names

S2.mat.m = (S2.mat1.m[]+S2.mat2.m[])/2
rownames(S2.mat.m) <- gr.m$names

S3.mat.m = (S3.mat1.m[]+S3.mat2.m[])/2
rownames(S3.mat.m) <- gr.m$names

noUV.mat.m = (noUV.mat1.m[]+noUV.mat2.m[])/2
rownames(noUV.mat.m) <- gr.m$names

cat("Center on S1 max\n")
s.sub.p <- S1.mat.p
s.sub.m <- S1.mat.m

get_centered_region <- function(x, y) {
  mat = list()
  for(i in 1:length(rownames(x))){
    center <- which.max(x[i, 175:225])
    shift <- y[i, (175+center-50):(175+center+50)]
    mat[[i]] <- shift
  }
  mat <- matrix(unlist(mat), ncol = length(rownames(x)))
  mat <- t(mat)
  rownames(mat) <- rownames(x)
  return(mat)
}

S1.p <- get_centered_region(s.sub.p, S1.mat.p)
S2.p <- get_centered_region(s.sub.p, S2.mat.p)
S3.p <- get_centered_region(s.sub.p, S3.mat.p)
noUV.p <- get_centered_region(s.sub.p, noUV.mat.p)

S1.m <- get_centered_region(s.sub.m, S1.mat.m)
S2.m <- get_centered_region(s.sub.m, S2.mat.m)
S3.m <- get_centered_region(s.sub.m, S3.mat.m)
noUV.m <- get_centered_region(s.sub.m, noUV.mat.m)

cat("Create table for profile plot\n")
S1.mat <- rbind(S1.p, S1.m)
S2.mat <- rbind(S2.p, S2.m)
S3.mat <- rbind(S3.p, S3.m)
noUV.mat <- rbind(noUV.p, noUV.m)

k = 3
S1.mat2 = t(runmean(t(S1.mat), k))
```

```
  S2.mat2 = t(runmean(t(S2.mat), k))
  S3.mat2 = t(runmean(t(S3.mat), k))
  noUV.mat2 = t(runmean(t(noUV.mat), k))

  S1.gg <- data.frame("value" = colMeans(S1.mat2, na.rm=TRUE))
  S1.gg$type <- "low"
  S1.gg$xlab <- c(1:dim(S1.mat2)[2])
  S2.gg <- data.frame("value" = colMeans(S2.mat2, na.rm=TRUE))
  S2.gg$type <- "med"
  S2.gg$xlab <- c(1:dim(S2.mat2)[2])
  S3.gg <- data.frame("value" = colMeans(S3.mat2, na.rm=TRUE))
  S3.gg$type <- "high"
  S3.gg$xlab <- c(1:dim(S3.mat2)[2])
  noUV.gg <- data.frame("value" = colMeans(noUV.mat2, na.rm=TRUE))
  noUV.gg$type <- "noUV"
  noUV.gg$xlab <- c(1:dim(noUV.mat2)[2])

  all <- rbind(S1.gg,S2.gg,S3.gg,noUV.gg)
  all$type <- factor(all$type)
  all$type <- factor(all$type, levels = c("high", "med", "low", "noUV"))
  return(all)
}

#If you want to run our code make sure to uncomment the two paragraphs below! It takes a lot of
    time
# FUS.plot <- shift3p(FUS, "FUS")
# HNRNPC.plot <- shift3p(HNRNPC, "HNRNPC")
# HNRNPM.plot <- shift3p(HNRNPM, "HNRNPM")
# HNRNPU.plot <- shift3p(HNRNPU, "HNRNPU")
# ELAVL1.plot <- shift3p(ELAVL1, "ELAVL1")
# HNRNPFH.plot <- shift3p(HNRNPFH, "HNRNPFH")
# KHSRP.plot <- shift3p(KHSRP, "KHSRP")
# THOC4.plot <- shift3p(THOC4, "THOC4")
#
# saveRDS(FUS.plot, "/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/4_
    Peak_shift/3_pvalue_0.05_fc1/FUS_3pshift.rds")
# saveRDS(HNRNPC.plot, "/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/
    4_Peak_shift/3_pvalue_0.05_fc1/HNRNPC_3pshift.rds")
# saveRDS(HNRNPM.plot, "/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/
    4_Peak_shift/3_pvalue_0.05_fc1/HNRNPM_3pshift.rds")
# saveRDS(HNRNPU.plot, "/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/
    4_Peak_shift/3_pvalue_0.05_fc1/HNRNPU_3pshift.rds")
# saveRDS(ELAVL1.plot, "/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/
    4_Peak_shift/3_pvalue_0.05_fc1/ELAVL1_3pshift.rds")
# saveRDS(HNRNPFH.plot, "/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones
    /4_Peak_shift/3_pvalue_0.05_fc1/HNRNPFH_3pshift.rds")
# saveRDS(KHSRP.plot, "/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/4
    _Peak_shift/3_pvalue_0.05_fc1/KHSRP_3pshift.rds")
# saveRDS(THOC4.plot, "/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/4
    _Peak_shift/3_pvalue_0.05_fc1/THOC4_3pshift.rds")

FUS.plot <- readRDS("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/4_
    Peak_shift/3_pvalue_0.05_fc1/FUS_3pshift.rds")
HNRNPC.plot <- readRDS("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/
    4_Peak_shift/3_pvalue_0.05_fc1/HNRNPC_3pshift.rds")
HNRNPM.plot <- readRDS("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/
    4_Peak_shift/3_pvalue_0.05_fc1/HNRNPM_3pshift.rds")
HNRNPU.plot <- readRDS("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/
    4_Peak_shift/3_pvalue_0.05_fc1/HNRNPU_3pshift.rds")
ELAVL1.plot <- readRDS("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/
    4_Peak_shift/3_pvalue_0.05_fc1/ELAVL1_3pshift.rds")
HNRNPFH.plot <- readRDS("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones
    /4_Peak_shift/3_pvalue_0.05_fc1/HNRNPFH_3pshift.rds")
KHSRP.plot <- readRDS("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/4
    _Peak_shift/3_pvalue_0.05_fc1/KHSRP_3pshift.rds")
THOC4.plot <- readRDS("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/4
    _Peak_shift/3_pvalue_0.05_fc1/THOC4_3pshift.rds")
```

```
FUS.plot$rbp <- "FUS"
HNRNPC.plot$rbp <- "HNRNPC"
HNRNPM.plot$rbp <- "HNRNPM"
HNRNPU.plot$rbp <- "HNRNPU"
ELAVL1.plot$rbp <- "ELAVL1"
HNRNPFH.plot$rbp <- "HNRNPFH"
KHSRP.plot$rbp <- "KHSRP"
THOC4.plot$rbp <- "THOC4"

all <- rbind(FUS.plot, HNRNPC.plot, HNRNPM.plot, HNRNPU.plot, HNRNPFH.plot, KHSRP.plot, ELAVL1.
    plot, THOC4.plot)
all$rbp <- factor(all$rbp, levels = c("FUS", "HNRNPC", "HNRNPM", "HNRNPU", "ELAVL1", "HNRNPFH", "
    KHSRP", "THOC4"))

cols <- c("high" = "#3fa966", "med" = "#c8e06a", "low" = "#3d6ccd", "noUV" = "grey")

all.plot <- all %>%
  ggplot( aes(x=xlab, y=value, group = type, color = type)) +
  geom_vline(xintercept = c(50), linetype="dashed")+
  # geom_area(aes(fill = type), alpha = 0.05, position = 'identity') +
  geom_line() +
  scale_color_manual(values = cols)+
  # xlim(c(145, 245))+
  xlab("Distance in bp") +
  theme_bw() +
  theme( panel.grid.major = element_blank(),
         panel.grid.minor = element_blank(),
         panel.background = element_blank(),
         axis.line = element_blank(),
         plot.title = element_text(hjust = 0.5)) +
  facet_wrap(vars(rbp), ncol = 4, nrow = 2,
             scales = "free_y")

all.plot
```



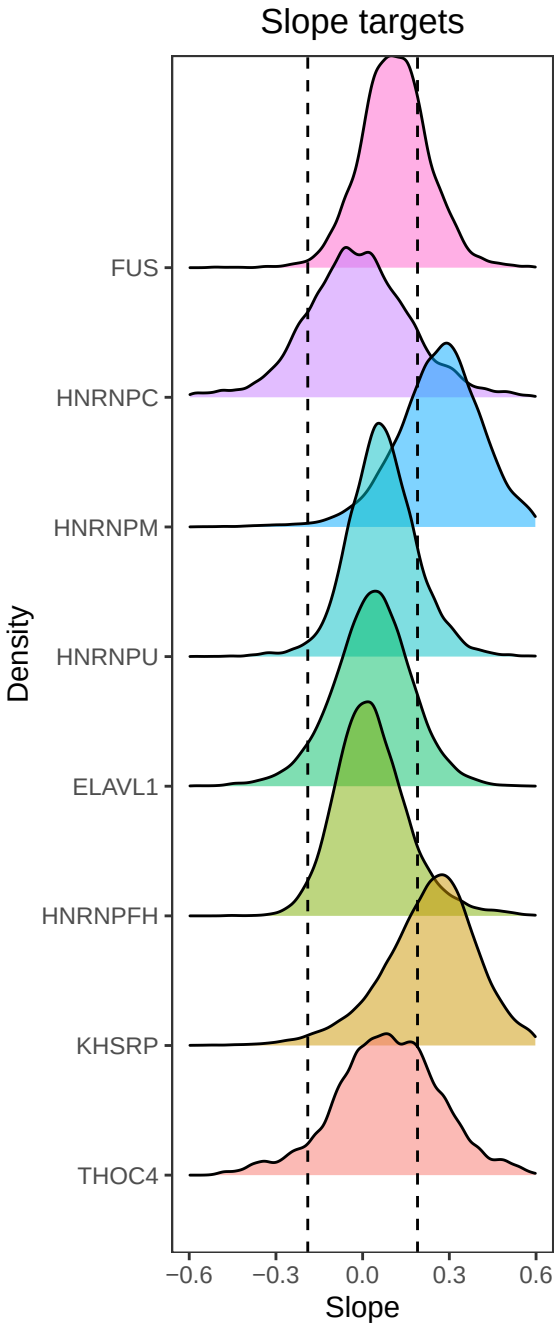

```
pdf("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/4_Peak_shift/3_pvalue_0.05_fc1/
    Profileplot_regions.pdf", width = 10, height = 4)
all.plot
dev.off()
```

4. Slope categorization

Slope calculation

The regions were categorized according to their slope calculated from the total normalized RT stop counts. Please see the snakemake at this Github link for more information.

```
datam <- rbind(FUS,HNRNPC,HNRNPM,HNRNPU,ELAVL1,HNRNPFH,KHSRP,THOC4)
datam$rbp <- factor(datam$rbp, levels = c("THOC4","KHSRP","HNRNPFH","ELAVL1","HNRNPU","HNRNPM","
    HNRNPC","FUS"))

datam.l <- subset(datam, lmslope < -sd(datam$lmslope, na.rm = TRUE))
datam.l$category_all <- "low"
datam.m <- subset(datam, lmslope > -sd(datam$lmslope, na.rm = TRUE) & lmslope < sd(datam$lmslope,
     na.rm = TRUE))
datam.m$category_all <- "med"
datam.h <- subset(datam, lmslope > sd(datam$lmslope, na.rm = TRUE))
datam.h$category_all <- "high"
datam <-  rbind(datam.l, datam.m, datam.h)

write.table(datam, "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/3_Categorization/
    3_pvalue_0.05_fc1/All_RBP_categorization.txt", quote = F, row.names = F, sep = "\t")

ggplot(data=datam, aes(x = lmslope, fill = rbp, y = rbp))+
  geom_density_ridges(alpha = 0.5)+
  geom_vline(aes(xintercept=0+sd(lmslope, na.rm = TRUE)), color="black", linetype="dashed", size
    =0.5)+
  geom_vline(aes(xintercept=0-sd(lmslope, na.rm = TRUE)), color="black", linetype="dashed", size
    =0.5)+
  xlab("Slope") + ylab("Density") + xlim(-0.6, 0.6) +
  ggtitle("Slope targets") +
  theme_bw() +
  theme(legend.position = "none", panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_blank(),
        plot.title = element_text(hjust = 0.5))
```



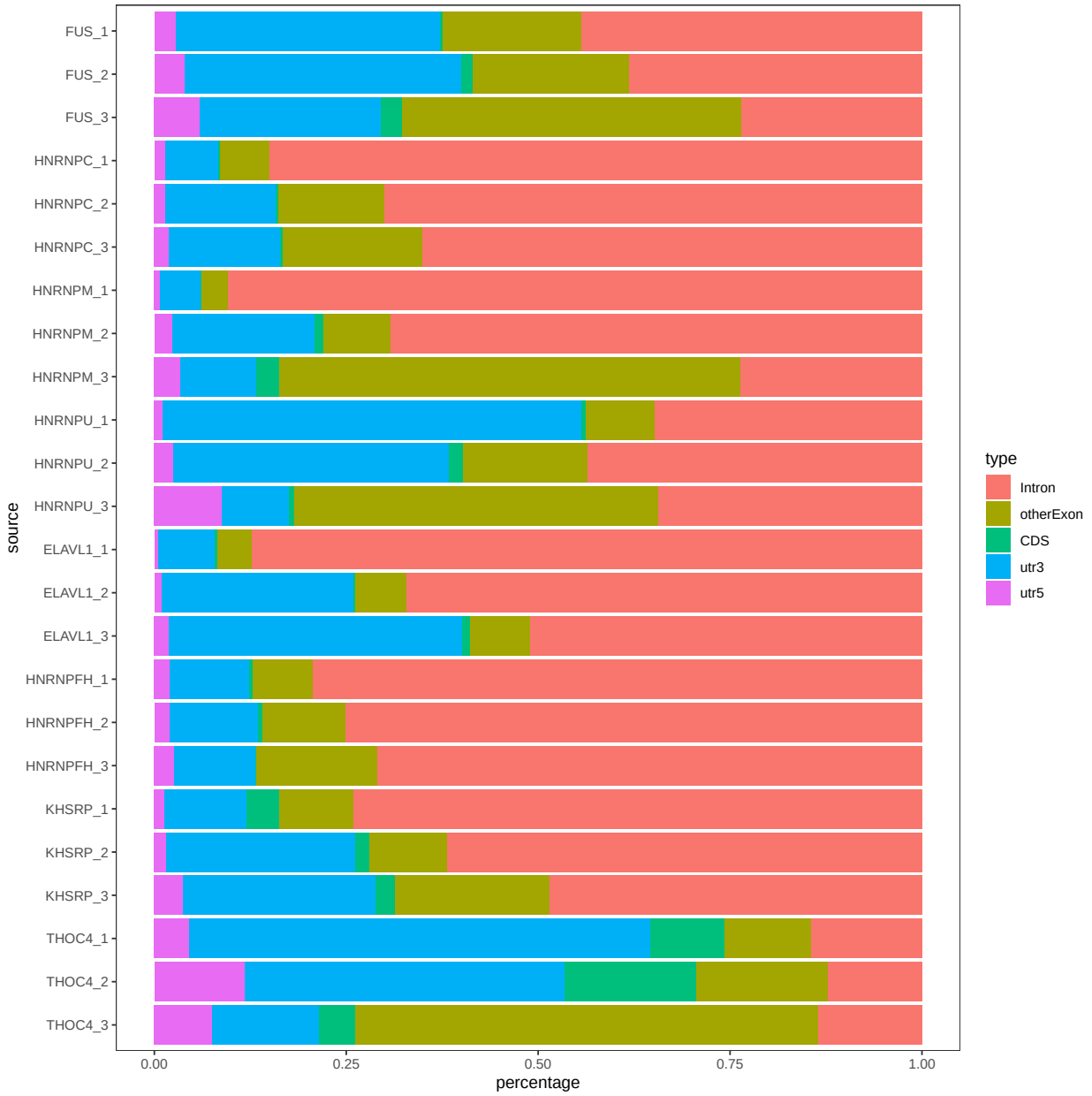

```
# Save the heatmap
pdf("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/3_Categorization/3_pvalue_0.05_
    fc1/Region_slope_ridgeline.pdf", height = 7, width = 3)
ggplot(data=datam, aes(x = lmslope, fill = rbp, y = rbp))+
  geom_density_ridges(alpha = 0.5)+
  geom_vline(aes(xintercept=0+sd(lmslope, na.rm = TRUE)), color="black", linetype="dashed", size
    =0.5)+
  geom_vline(aes(xintercept=0-sd(lmslope, na.rm = TRUE)), color="black", linetype="dashed", size
    =0.5)+
  xlab("Slope") + ylab("Density") + xlim(-0.6, 0.6) +
  ggtitle("Slope targets") +
  theme_bw() +
  theme(legend.position = "none", panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
```

```
        panel.background = element_blank(),
        axis.line = element_blank(),
        plot.title = element_text(hjust = 0.5))
dev.off()
```

Genomic annotation of categorized regions

```
#Load annotation
txdb <- loadDb("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/0_Annotation/gencode.
    v39.annotation")

#Create a granges for all RBPs
datam.gr <- GRanges(datam[,grep("chromosome", colnames(datam))],
                    IRanges(start=datam[,grep("region_begin", colnames(datam))], end=datam[,grep("
    region_end", colnames(datam))]),
                    names = datam[,grep("regionID", colnames(datam))], strand = datam[,grep("region
    _strand", colnames(datam))],
                    rbp = datam[,grep("rbp", colnames(datam))],
                    category = datam[,grep("category_all", colnames(datam))])

datam.grl <- GRangesList("FUS_3" = datam.gr %>% filter(rbp == "FUS" & category == "low"),
                         "FUS_2" = datam.gr %>% filter(rbp == "FUS" & category == "med"),
                         "FUS_1" = datam.gr %>% filter(rbp == "FUS" & category == "high"),
                         "HNRNPC_3" = datam.gr %>% filter(rbp == "HNRNPC" & category == "low"),
                         "HNRNPC_2" = datam.gr %>% filter(rbp == "HNRNPC" & category == "med"),
                         "HNRNPC_1" = datam.gr %>% filter(rbp == "HNRNPC" & category == "high"),
                         "HNRNPM_3" = datam.gr %>% filter(rbp == "HNRNPM" & category == "low"),
                         "HNRNPM_2" = datam.gr %>% filter(rbp == "HNRNPM" & category == "med"),
                         "HNRNPM_1" = datam.gr %>% filter(rbp == "HNRNPM" & category == "high"),
                         "HNRNPU_3" = datam.gr %>% filter(rbp == "HNRNPU" & category == "low"),
                         "HNRNPU_2" = datam.gr %>% filter(rbp == "HNRNPU" & category == "med"),
                         "HNRNPU_1" = datam.gr %>% filter(rbp == "HNRNPU" & category == "high"),
                         "ELAVL1_3" = datam.gr %>% filter(rbp == "ELAVL1" & category == "low"),
                         "ELAVL1_2" = datam.gr %>% filter(rbp == "ELAVL1" & category == "med"),
                         "ELAVL1_1" = datam.gr %>% filter(rbp == "ELAVL1" & category == "high"),
                         "HNRNPFH_3" = datam.gr %>% filter(rbp == "HNRNPFH" & category == "low"),
                         "HNRNPFH_2" = datam.gr %>% filter(rbp == "HNRNPFH" & category == "med"),
                         "HNRNPFH_1" = datam.gr %>% filter(rbp == "HNRNPFH" & category == "high"),
                         "KHSRP_3" = datam.gr %>% filter(rbp == "KHSRP" & category == "low"),
                         "KHSRP_2" = datam.gr %>% filter(rbp == "KHSRP" & category == "med"),
                         "KHSRP_1" = datam.gr %>% filter(rbp == "KHSRP" & category == "high"),
                         "THOC4_3" = datam.gr %>% filter(rbp == "THOC4" & category == "low"),
                         "THOC4_2" = datam.gr %>% filter(rbp == "THOC4" & category == "med"),
                         "THOC4_1" = datam.gr %>% filter(rbp == "THOC4" & category == "high"))

#Generate the plot for the annotation
dist <- genomicElementDistribution(datam.grl, TxDb = txdb, promoterRegion=c(upstream=2000,
    downstream=0), geneDownstream=c(upstream=0, downstream=2000), ignore.strand = FALSE, plot =
    FALSE)

data <- dist$plot$data %>% filter(category == "Exon level") %>% mutate(across('type', str_replace
    , 'undefined', 'Intron'))
data$type <- factor(data$type, levels = c("Intron", "otherExon", "CDS", "utr3", "utr5"))

ggplot(data, aes_string(x = "source", y = "percentage", fill = "type")) + geom_bar(stat = "
    identity") + coord_flip() +
          theme_bw() + theme(panel.grid.major = element_blank(), panel.grid.minor = element_
    blank(), panel.background = element_blank(),
                             axis.line = element_blank(), plot.title = element_text(hjust =
    0.5)) +
          scale_x_discrete(limits = c("THOC4_3", "THOC4_2","THOC4_1", "KHSRP_3", "KHSRP_2","
    KHSRP_1",
                                      "HNRNPFH_3", "HNRNPFH_2","HNRNPFH_1", "ELAVL1_3", "ELAVL1
    _2","ELAVL1_1",
```

```
    ,"HNRNPM_1",                    "HNRNPU_3", "HNRNPU_2","HNRNPU_1", "HNRNPM_3", "HNRNPM_2"
                                    "HNRNPC_3","HNRNPC_2","HNRNPC_1", "FUS_3","FUS_2","FUS_1"
    ))
```




```
pdf("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/3_Categorization/3_pvalue_0.05_
    fc1/Genomic_distribution_subzones.pdf", width = 3, height = 5)
ggplot(data, aes_string(x = "source", y = "percentage", fill = "type")) + geom_bar(stat = "
    identity") + coord_flip() +
          theme_bw() + theme(legend.position = "none", panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), panel.background = element_blank(),
                             axis.line = element_blank(), plot.title = element_text(hjust =
    0.5)) +
          scale_x_discrete(limits = c("THOC4_3", "THOC4_2","THOC4_1", "KHSRP_3", "KHSRP_2","
    KHSRP_1",
```

```
                                                    "HNRNPFH_3", "HNRNPFH_2","HNRNPFH_1", "ELAVL1_3", "ELAVL1
    _2","ELAVL1_1",
                                                    "HNRNPU_3", "HNRNPU_2","HNRNPU_1", "HNRNPM_3", "HNRNPM_2"
    ,"HNRNPM_1",
                                                    "HNRNPC_3","HNRNPC_2","HNRNPC_1", "FUS_3","FUS_2","FUS_1"
    ))
dev.off()
```

5. Motif analysis

To understand which motifs are enriched 3' downstream from the highest peak, we performed enrichment analysis using AME. A tool from the MEME suite.

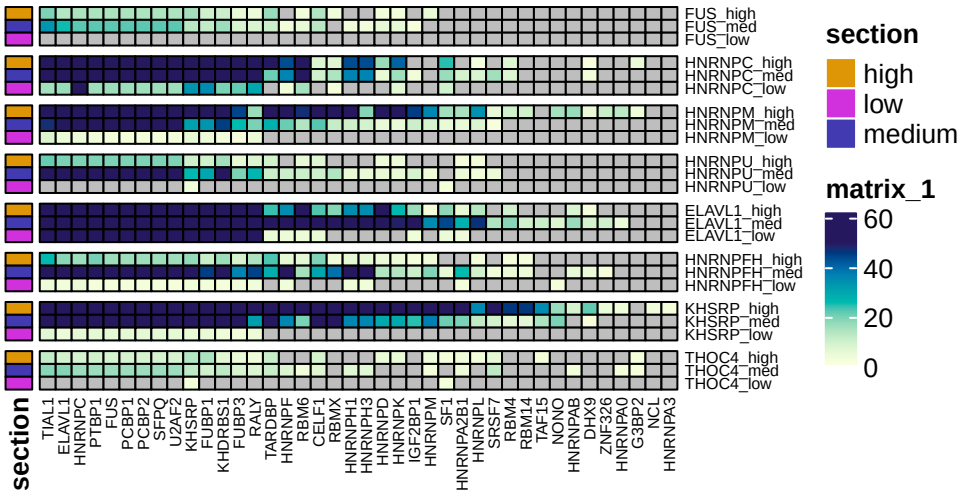
```
#Load the binding regions and genome
all <- read.delim("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/3_
    Categorization/3_pvalue_0.05_fc1/All_RBP_categorization.txt", header = TRUE)
all$rbp_category <- paste(all$rbp, all$category_all, sep = "_")
human.genome <- BSgenome.Hsapiens.UCSC.hg38

expandRange = function(x, upstream, downstream) {
  strand_is_minus = strand(x) == "-"
  on_plus = which(!strand_is_minus)
  on_minus = which(strand_is_minus)
  start(x)[on_plus] = start(x)[on_plus] - upstream
  start(x)[on_minus] = start(x)[on_minus] - downstream
  end(x)[on_plus] = end(x)[on_plus] + downstream
  end(x)[on_minus] = end(x)[on_minus] + upstream
  return(x)
}

#Subset
h <- all %>% filter(category_all == "high")
m <- all %>% filter(category_all == "med")
l <- all %>% filter(category_all == "low")

#Generate granges
h.gr <- GRanges(h[,grep("chr_S3", colnames(h))],
                          IRanges(start=h[,grep("begin_S1", colnames(h))], end=h[,grep("end_S1",
    colnames(h))]),
                          names = h[,grep("regionID", colnames(h))],  strand = h[,grep("strand_S1
    ", colnames(h))],
                        section = h[,grep("category_all", colnames(h))], rbp_category = h[,grep("
    rbp_category", colnames(h))])
h.gr.exp <- h.gr %>% expandRange(upstream = 0, downstream = 225)

m.gr <- GRanges(m[,grep("chr_S2", colnames(m))],
                        IRanges(start=m[,grep("begin_S1", colnames(m))], end=m[,grep("end_S1",
    colnames(m))]),
                        names = m[,grep("regionID", colnames(m))],  strand = m[,grep("strand_S1",
    colnames(m))],
                        section = m[,grep("category_all", colnames(m))], rbp_category = m[,grep("
    rbp_category", colnames(m))])
m.gr.exp <- m.gr %>% expandRange(upstream = 0, downstream = 125)

l.gr <- GRanges(l[,grep("chr_S1", colnames(l))],
                        IRanges(start=l[,grep("begin_S1", colnames(l))], end=l[,grep("end_S1",
    colnames(l))]),
                        names = l[,grep("regionID", colnames(l))],  strand = l[,grep("strand_S1",
    colnames(l))],
                        section = l[,grep("category_all", colnames(l))], rbp_category = l[,grep("
    rbp_category", colnames(l))])
l.gr.exp <- l.gr %>% expandRange(upstream = 0, downstream = 50)
```

```
#Get sequence for motif analysis
gr <- bind_ranges(h.gr.exp, m.gr.exp, l.gr.exp)
# gr.seq.all <- unique(gr) %>% get_sequence(human.genome)
gr.seq <- gr %>% split(mcols(.)$rbp_category) %>% get_sequence(human.genome)
gr.seq
```

```
## BStringSetList of length 24
## [["ELAVL1_high"]] chr1:100002555-100002830=TTTATTTCATCCTTTGTATTTTTGTTTAGTTTCT...
## [["ELAVL1_low"]] chr1:100094646-100094746=CTACAGGTGCCTGTCACTGCACCAGCTAATTTTTG...
## [["ELAVL1_med"]] chr1:100000855-100001030=CATCTTTAATCCATTTTGGGTTTTTTTTTTATATG...
## [["FUS_high"]] chr1:108696396-108696671=TTTTATTTATTGGTGGTTTAATGTCTTCAAGAACTTC...
## [["FUS_low"]] chr1:173865319-173865419=GATTTTAAAATTAAAGCAGATGGGAATCTCTCTGAGAA...
## [["FUS_med"]] chr1:103553673-103553848=TATCTGGTTTAGGTATTTATAATGTGTTTAAAATTTGG...
## [["HNRNPC_high"]] chr1:100016040-100016315=TTAAAAGCAGGTAGGATACTTCTATTTTTTTTTT...
## [["HNRNPC_low"]] chr1:11024083-11024183=ATAGCAGTTAATTCCCCTTTTTTGACCCTTTTGAGAT...
## [["HNRNPC_med"]] chr1:100018855-100019030=TTTTTATATTGAACATTGACGGTTTTAATTCTTTT...
## [["HNRNPFH_high"]] chr1:10045754-10046029=TTGAGAGGGGGAGAAAGGGTCAAACACTAGATGGT...
## ...
## <14 more elements>
```

```
#Load and prepare database
meme_db <- read_meme("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/0_Annotation/
    motif/all.meme")
meme_db_df <- meme_db %>% to_df()
meme_db_df$altname <- sapply(strsplit(meme_db_df$name, "_"), function(x) x[1])
gene <- unique(meme_db_df$altname)

motif_list <- list()
for (i in 1:length(gene)) {
meme <- meme_db_df %>%  dplyr::filter(altname == gene[i]) %>% remove_duplicate_motifs()
motif_list[[i]] <- meme
}
meme_db_df_dedup <- motif_list %>% dplyr::bind_rows(.id = "altname")
meme_db_df_dedup$altname <- sapply(strsplit(meme_db_df_dedup$name, "_"), function(x) x[1])
meme_db_new <- meme_db_df_dedup %>% to_list()

#Run AME -if you want to run the code uncomment the next two lines (it takes around 4h on a M1
    Mac)
# ame_all <- gr.seq %>% runAme(database = meme_db_new, silent = FALSE) %>% dplyr::bind_rows(.id =
     "rbp_category")

#Save AME results
# saveRDS(ame_all, "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/5_motif_analysis/
    3_pvalue_0.05_fc1/AME_all_shuffleCtrl_bothsides.rds")

#Read RDS
ame_all <- readRDS("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/5_motif_analysis/
    3_pvalue_0.05_fc1/AME_all_shuffleCtrl_bothsides.rds")

#Create heatmap of AME results
levels <- c("THOC4_low", "THOC4_med", "THOC4_high",
            "KHSRP_low", "KHSRP_med", "KHSRP_high", "HNRNPFH_low", "HNRNPFH_med", "HNRNPFH_high",
            "ELAVL1_low", "ELAVL1_med", "ELAVL1_high",
            "HNRNPU_low",  "HNRNPU_med", "HNRNPU_high", "HNRNPM_low", "HNRNPM_med", "HNRNPM_high"
    ,
            "HNRNPC_low",  "HNRNPC_med", "HNRNPC_high", "FUS_low", "FUS_med", "FUS_high")

ame_all$rbp_category <- factor(ame_all$rbp_category, levels = levels)

heatmap_mat <- ame_all %>%  dplyr::group_by(rbp_category, motif_alt_id, consensus) %>% mutate(adj
    .pvalue = -log10(adj.pvalue)) %>% dplyr::filter(adj.pvalue == max(adj.pvalue), motif_alt_id !
    = "NA")
```

```
heatmap_mat$motif_gene <- paste(heatmap_mat$motif_alt_id, heatmap_mat$consensus, sep = "_")
heatmap_mat$rbp <- sapply(strsplit(as.character(heatmap_mat$rbp_category), "_"), function(x) x
    [1])
heatmap_mat2 <- heatmap_mat %>% dplyr::group_by(rbp, motif_gene) %>% dplyr::summarise_at(vars(
    adj.pvalue), list(adj.pvalue = mean))
heatmap_mat2$protID <- sapply(strsplit(as.character(heatmap_mat2$motif_gene), "_"), function(x) x
    [1])
heatmap_mat3 <- heatmap_mat2 %>% dplyr::group_by(rbp, protID) %>% dplyr::filter(adj.pvalue == max
    (adj.pvalue))
heatmap_mat <- heatmap_mat %>% dplyr::group_by(rbp_category, motif_alt_id) %>% dplyr::filter(
    motif_gene %in% heatmap_mat3$motif_gene) %>% dplyr::filter(adj.pvalue == max(adj.pvalue))
heatmap_mat <- heatmap_mat %>% dplyr::select(rbp_category, motif_alt_id, adj.pvalue) %>% unique()
     %>% pivot_wider(names_from = motif_alt_id, values_from = adj.pvalue)

heatmap_hm <- as.matrix(heatmap_mat[-1])
rownames(heatmap_hm) <- heatmap_mat$rbp_category
heatmap_hm <- heatmap_hm[match(levels(heatmap_mat$rbp_category), rownames(heatmap_hm)),]

#Load TMT data
FUS.TMT <- read.delim("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/TMT_analysis/
    3_DEP/FUS_TMT_twofactor_results_sub.txt", header = TRUE)
FUS.TMT <- subset(FUS.TMT, order_sign != "FALSE_FALSE_FALSE_FALSE")
HNRNPC.TMT <- read.delim("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/TMT_
    analysis/3_DEP/hnC_TMT_twofactor_results_sub.txt", header = TRUE)
HNRNPC.TMT <- subset(HNRNPC.TMT, order_sign != "FALSE_FALSE_FALSE_FALSE")
HNRNPM.TMT <- read.delim("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/TMT_
    analysis/3_DEP/hnM_TMT_twofactor_results_sub.txt", header = TRUE)
HNRNPM.TMT <- subset(HNRNPM.TMT, order_sign != "FALSE_FALSE_FALSE_FALSE")
HNRNPU.TMT <- read.delim("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/TMT_
    analysis/3_DEP/hnU_TMT_twofactor_results_sub.txt", header = TRUE)
HNRNPU.TMT <- subset(HNRNPU.TMT, order_sign != "FALSE_FALSE_FALSE_FALSE")

genes <- unique(c(FUS.TMT$gene, HNRNPC.TMT$gene, HNRNPM.TMT$gene, HNRNPU.TMT$gene))

heatmap_hm <- heatmap_hm[, colnames(heatmap_hm) %in% genes]

heatmap_hm <- heatmap_hm[length(rownames(heatmap_hm)):1, order(colSums(-heatmap_hm, na.rm=TRUE))]
rownames(heatmap_hm)[3] <- "FUS_low"
annotation_col <- data.frame(section = rep(c("high", "medium", "low"), length(rownames(heatmap_hm
    ))/3))
rownames(annotation_col) <- rownames(heatmap_hm)

# Heatmap color range
my.breaks <- c(seq(0, 50, by=0.01))#, seq(60, 150, by=1))
my.colors <- c("white", rev(paletteer_c("grDevices::YlGnBu" , length(my.breaks)-1)))

#Heatmap
heatmap_ame <- pheatmap(
  mat               = heatmap_hm,
  border_color = "black",
  annotation_row = annotation_col,
  cellwidth = 6,
  cellheight = 6,
  fontsize_number=5.5,
  color = my.colors,
  breaks = my.breaks,
  show_colnames     = TRUE,
  show_rownames     = TRUE,
  drop_levels       = TRUE,
  fontsize          = 5.5,
  cluster_rows      = FALSE,
  cluster_cols      = FALSE,
  gaps_row = c(3,6,9,12,15,18,21,24),
  na_col = "lightgrey"
)

heatmap_ame
```




```
pdf("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/5_motif_analysis/3_pvalue_0.05_
    fc1/Heatmap_AME_bothsides.pdf", width = 10, height = 4)
heatmap_ame
dev.off()
```

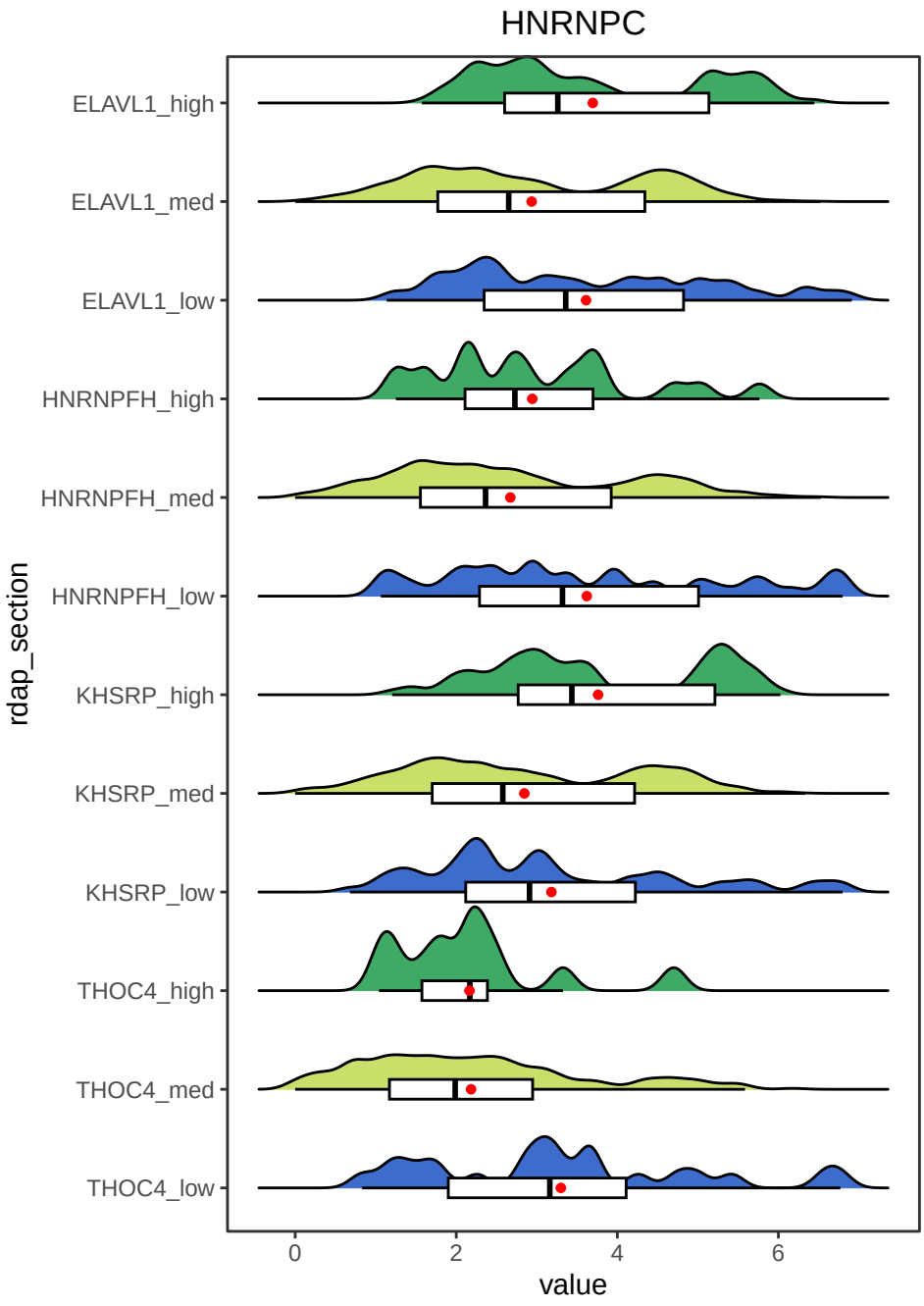
#6. Density plot between RBP-RDAP nearest peaks We reasoned that the logFC calculated by DEWSeq in the regions should be greater in the higher regions for the RDAPs that we know are coming in the higher subzones.

```
all <- read.delim("/Users/lducoli/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/3_
    Categorization/3_pvalue_0.05_fc1/All_RBP_categorization.txt", header = TRUE)
all$rbp_category <- paste(all$rbp, all$category_all, sep = "_")
HNRNPC.dds <- readRDS("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/1_Regions_DEW-
    Seq/deseq2_norm/1_Subzone_separate/3_pvalue_0.05_fc1/HNRNPC_DEWseq_res.rds")

#Subset
h <- all %>% filter(category_all == "high")
m <- all %>% filter(category_all == "med")
l <- all %>% filter(category_all == "low")

#Generate granges
h.gr <- GRanges(h[,grep("chr_S3", colnames(h))],
```

```
                          IRanges(start=h[,grep("begin_S3", colnames(h))], end=h[,grep("end_S3",
    colnames(h))]),
                          names = h[,grep("regionID", colnames(h))],  strand = h[,grep("strand_S3
    ", colnames(h))],
                        section = h[,grep("category_all", colnames(h))], rbp_category = h[,grep("
    rbp_category", colnames(h))],
                         max_fc = h[,grep("max_S3_fc", colnames(h))])
h.gr <- gr.mid(h.gr)


m.gr <- GRanges(m[,grep("chr_S2", colnames(m))],
                        IRanges(start=m[,grep("begin_S2", colnames(m))], end=m[,grep("end_S2",
    colnames(m))]),
                        names = m[,grep("regionID", colnames(m))],  strand = m[,grep("strand_S2",
    colnames(m))],
                        section = m[,grep("category_all", colnames(m))], rbp_category = m[,grep("
    rbp_category", colnames(m))],
                         max_fc = m[,grep("max_S2_fc", colnames(m))])
m.gr <- gr.mid(m.gr)

l.gr <- GRanges(l[,grep("chr_S1", colnames(l))],
                        IRanges(start=l[,grep("begin_S1", colnames(l))], end=l[,grep("end_S1",
    colnames(l))]),
                        names = l[,grep("regionID", colnames(l))],  strand = l[,grep("strand_S1",
    colnames(l))],
                        section = l[,grep("category_all", colnames(l))], rbp_category = l[,grep("
    rbp_category", colnames(l))],
                         max_fc = l[,grep("max_S1_fc", colnames(l))])
l.gr <- gr.mid(l.gr)

logfc_table <- function(rbp, rdap, section1, section2, section3, gr1, gr2, gr3) {
distance_genomic <- function(rbp, rdap, section, gr) {
RBP_RDAP.dist <- gr.dist(gr[gr$rbp_category == paste(rbp, section, sep = "_")], gr[gr$rbp_
    category == paste(rdap, section, sep = "_")], ignore.strand=FALSE)
rownames(RBP_RDAP.dist) <- gr[gr$rbp_category == paste(rbp, section, sep = "_")]$names
colnames(RBP_RDAP.dist) <- gr[gr$rbp_category == paste(rdap, section, sep = "_")]$names

RBP_RDAP.dist <- as.data.frame(RBP_RDAP.dist)
RBP_RDAP.dist$RBP_bdgsites <- rownames(RBP_RDAP.dist)
rownames(RBP_RDAP.dist) <- NULL
RBP_RDAP.dist <- RBP_RDAP.dist[c(length(gr[gr$rbp_category == paste(rdap, section, sep = "_")]$
    names)+1,1:length(gr[gr$rbp_category == paste(rdap, section, sep = "_")]$names))]

dist <- list()
for(i in 1:length(RBP_RDAP.dist$RBP_bdgsites)){
  gene <- sapply(strsplit(sapply(strsplit(RBP_RDAP.dist$RBP_bdgsites[i], "_"), function(x) x[5]),
     ":"), function(x) x[1])
  data_columns <- c(1,grep(gene, colnames(RBP_RDAP.dist)))
  if( length(data_columns) == 1 ) {
    dist[[i]] <- NA
  } else {
  a <- RBP_RDAP.dist[,data_columns]
  b <- data.frame(dist=t(a[i,]), RBP_bdgsites = RBP_RDAP.dist$RBP_bdgsites[i])
  b <- b[-1,]
  b$RDAP_bdgsites <- rownames(b)
  rownames(b) <- NULL
  colnames(b)[1] <- "dist"
  b <- b[,c(3,1:2)]
  dist[[i]] <- b
  }
}

RDAP <- do.call(rbind, dist)
RDAP <- subset(RDAP, dist != "NA")
RDAP$RBP_bdgsites <- factor(RDAP$RBP_bdgsites)
RDAP$dist <- as.numeric(RDAP$dist)
RDAP$logdist <- log10(RDAP$dist)
RDAP$kbdist <- RDAP$dist/1000
```

```
RDAP$logdist[is.infinite(RDAP$logdist)] <- 0
RDAP$section <- section
RDAP <- RDAP %>% arrange(dist, RDAP_bdgsites) %>% group_by(RDAP_bdgsites) %>% dplyr::slice(1)
RDAP <- RDAP %>% arrange(dist, RBP_bdgsites) %>% group_by(RBP_bdgsites) %>% dplyr::slice(1)
return(RDAP)
}

RDAP_dist_high <- distance_genomic(rbp, rdap, section1, gr1)
RDAP_dist_med <- distance_genomic(rbp, rdap, section2, gr2)
RDAP_dist_low <- distance_genomic(rbp, rdap, section3, gr3)
RDAP_dist_high <- RDAP_dist_high %>% filter(dist < 250)
RDAP_dist_med <- RDAP_dist_med %>% filter(dist < 150)
RDAP_dist_low <- RDAP_dist_low %>% filter(dist < 75)

RDAP_high <- gr1[gr1$rbp_category == paste(rbp, section1, sep = "_")]
RDAP_high <- RDAP_high[RDAP_high$names %in% RDAP_dist_high$RBP_bdgsites]

RDAP_med <- gr2[gr2$rbp_category == paste(rbp, section2, sep = "_")]
RDAP_med <- RDAP_med[RDAP_med$names %in% RDAP_dist_med$RBP_bdgsites]

RDAP_low <- gr3[gr3$rbp_category == paste(rbp, section3, sep = "_")]
RDAP_low <- RDAP_low[RDAP_low$names %in% RDAP_dist_low$RBP_bdgsites]

RBP_RDAP <- data.frame(names = c(RDAP_high$names, RDAP_med$names, RDAP_low$names), value = c(RDAP
    _high$max_fc, RDAP_med$max_fc, RDAP_low$max_fc), section = c(rep(section1, length(RDAP_high$
    max_fc)), rep(section2, length(RDAP_med$max_fc)), rep(section3, length(RDAP_low$max_fc))),
    rdap = rep(rdap, length(c(RDAP_high$max_fc, RDAP_med$max_fc, RDAP_low$max_fc))))

return(RBP_RDAP)
}

#If you want to run the code you need to uncomment the two paragraph below (it takes around 10min
    )
# RBP_RDAP1 <- logfc_table("HNRNPC", "KHSRP", "high", "med", "low", h.gr, m.gr, l.gr)
# RBP_RDAP2 <- logfc_table("HNRNPC", "ELAVL1", "high", "med", "low", h.gr, m.gr, l.gr)
# RBP_RDAP3 <- logfc_table("HNRNPC", "HNRNPFH", "high", "med", "low", h.gr, m.gr, l.gr)
# RBP_RDAP4 <- logfc_table("HNRNPC", "THOC4", "high", "med", "low", h.gr, m.gr, l.gr)

# saveRDS(RBP_RDAP1, "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/6_LogFC_plot/3_
    pvalue_0.05_fc1/HNRNPC_KHSRP_logFCtable.rds")
# saveRDS(RBP_RDAP2, "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/6_LogFC_plot/3_
    pvalue_0.05_fc1/HNRNPC_ELAVL1_logFCtable.rds")
# saveRDS(RBP_RDAP3, "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/6_LogFC_plot/3_
    pvalue_0.05_fc1/HNRNPC_HNRNPFH_logFCtable.rds")
# saveRDS(RBP_RDAP4, "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/6_LogFC_plot/3_
    pvalue_0.05_fc1/HNRNPC_THOC4_logFCtable.rds")

RBP_RDAP1 <- readRDS("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/6_LogFC_plot/3_
    pvalue_0.05_fc1/HNRNPC_KHSRP_logFCtable.rds")
RBP_RDAP2 <- readRDS("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/6_LogFC_plot/3_
    pvalue_0.05_fc1/HNRNPC_ELAVL1_logFCtable.rds")
RBP_RDAP3 <- readRDS("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/6_LogFC_plot/3_
    pvalue_0.05_fc1/HNRNPC_HNRNPFH_logFCtable.rds")
RBP_RDAP4 <- readRDS("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/6_LogFC_plot/3_
    pvalue_0.05_fc1/HNRNPC_THOC4_logFCtable.rds")

RBP_RDAP <- rbind(RBP_RDAP1, RBP_RDAP2, RBP_RDAP3, RBP_RDAP4)
RBP_RDAP$rdap_section <- paste(RBP_RDAP$rdap, RBP_RDAP$section, sep = "_")

write.table(RBP_RDAP, "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/6_LogFC_plot/3
    _pvalue_0.05_fc1/HNRNPC_LogFC_nearest.txt", quote = F, sep = "\t", row.names = F)

RBP_RDAP$rdap_section <- factor(RBP_RDAP$rdap_section, levels= c("THOC4_low","THOC4_med","THOC4_
    high","KHSRP_low","KHSRP_med", "KHSRP_high",
                                                                "HNRNPFH_low","HNRNPFH_med","
    HNRNPFH_high",
                                                                "ELAVL1_low","ELAVL1_med","
    ELAVL1_high"))
```

```
my_pal <- c("THOC4_high" = "#3fa966", "THOC4_med" = "#c8e06a", "THOC4_low" = "#3d6ccd",
            "KHSRP_high" = "#3fa966", "KHSRP_med" = "#c8e06a", "KHSRP_low" = "#3d6ccd",
            "HNRNPFH_high" = "#3fa966", "HNRNPFH_med" = "#c8e06a", "HNRNPFH_low" = "#3d6ccd",
            "ELAVL1_high" = "#3fa966", "ELAVL1_med" = "#c8e06a", "ELAVL1_low" = "#3d6ccd")

logFCplot <- ggplot(RBP_RDAP, aes(x = value , y = rdap_section, group = rdap_section, fill = rdap
    _section)) +
  stat_density_ridges(scale = .85,geom = "density_ridges_gradient",
                      alpha = 0.7, bandwidth = 0.15) +
  geom_boxplot(aes(x = value, y = rdap_section), color = "black", inherit.aes = FALSE, width =
    0.2, outlier.shape = NA) +
  stat_summary(fun=mean, geom="point", shape=20, size=2, color="red", fill="red") +
  coord_cartesian(clip = "off") +
  scale_fill_manual(values = my_pal, guide = "none") +
  scale_y_discrete(expand = c(0.04, 0)) +
  ggtitle("HNRNPC") + theme_bw() +
  theme( panel.grid.major = element_blank(),
         panel.grid.minor = element_blank(),
         panel.background = element_blank(),
         axis.line = element_blank(),
         plot.title = element_text(hjust = 0.5))

logFCplot
```




```
RBP_RDAP$section <- factor(RBP_RDAP$rdap_section)
HNRNPC.res_aov <- aov(value ~ rdap_section, data = RBP_RDAP)
post_test <- glht(HNRNPC.res_aov, linfct = mcp(rdap_section = "Tukey"), alternative = "greater")
summary(post_test)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = value ~ rdap_section, data = RBP_RDAP)
```

```
##
## Linear Hypotheses:
##                                              Estimate Std. Error t value Pr(>t)
## THOC4_med - THOC4_low <= 0                 -1.115943   0.313655  -3.558 1.0000
## THOC4_high - THOC4_low <= 0                -1.134790   0.497519  -2.281 1.0000
## KHSRP_low - THOC4_low <= 0                 -0.118089   0.346453  -0.341 1.0000
## KHSRP_med - THOC4_low <= 0                 -0.454068   0.305993  -1.484 1.0000
## KHSRP_high - THOC4_low <= 0                 0.462354   0.339789   1.361 0.8827
## HNRNPFH_low - THOC4_low <= 0                0.321144   0.379210   0.847 0.9923
## HNRNPFH_med - THOC4_low <= 0               -0.628176   0.306382  -2.050 1.0000
## HNRNPFH_high - THOC4_low <= 0              -0.355435   0.439397  -0.809 1.0000
## ELAVL1_low - THOC4_low <= 0                 0.312809   0.336831   0.929 0.9864
## ELAVL1_med - THOC4_low <= 0                -0.363014   0.304602  -1.192 1.0000
## ELAVL1_high - THOC4_low <= 0                0.395955   0.320828   1.234 0.9290
## THOC4_high - THOC4_med <= 0                -0.018847   0.402528  -0.047 1.0000
## KHSRP_low - THOC4_med <= 0                  0.997854   0.185832   5.370  <0.01 ***
## KHSRP_med - THOC4_med <= 0                  0.661875   0.090197   7.338  <0.01 ***
## KHSRP_high - THOC4_med <= 0                 1.578297   0.173091   9.118  <0.01 ***
## HNRNPFH_low - THOC4_med <= 0                1.437087   0.241462   5.952  <0.01 ***
## HNRNPFH_med - THOC4_med <= 0                0.487767   0.091506   5.330  <0.01 ***
## HNRNPFH_high - THOC4_med <= 0               0.760508   0.327983   2.319 0.2706
## ELAVL1_low - THOC4_med <= 0                 1.428752   0.167209   8.545  <0.01 ***
## ELAVL1_med - THOC4_med <= 0                 0.752929   0.085360   8.821  <0.01 ***
## ELAVL1_high - THOC4_med <= 0                1.511898   0.132038  11.450  <0.01 ***
## KHSRP_low - THOC4_high <= 0                 1.016702   0.428578   2.372 0.2418
## KHSRP_med - THOC4_high <= 0                 0.680722   0.396587   1.716 0.6762
## KHSRP_high - THOC4_high <= 0                1.597145   0.423209   3.774  <0.01 **
## HNRNPFH_low - THOC4_high <= 0               1.455934   0.455466   3.197 0.0271 *
## HNRNPFH_med - THOC4_high <= 0               0.506614   0.396886   1.276 0.9149
## HNRNPFH_high - THOC4_high <= 0              0.779356   0.506674   1.538 0.7922
## ELAVL1_low - THOC4_high <= 0                1.447599   0.420838   3.440 0.0125 *
## ELAVL1_med - THOC4_high <= 0                0.771776   0.395515   1.951 0.5074
## ELAVL1_high - THOC4_high <= 0               1.530745   0.408142   3.751  <0.01 **
## KHSRP_med - KHSRP_low <= 0                 -0.335980   0.172587  -1.947 1.0000
## KHSRP_high - KHSRP_low <= 0                 0.580443   0.227181   2.555 0.1603
## HNRNPFH_low - KHSRP_low <= 0                0.439232   0.282762   1.553 0.7825
## HNRNPFH_med - KHSRP_low <= 0               -0.510087   0.173274  -2.944 1.0000
## HNRNPFH_high - KHSRP_low <= 0              -0.237346   0.359477  -0.660 1.0000
## ELAVL1_low - KHSRP_low <= 0                 0.430898   0.222732   1.935 0.5198
## ELAVL1_med - KHSRP_low <= 0                -0.244926   0.170109  -1.440 1.0000
## ELAVL1_high - KHSRP_low <= 0                0.514044   0.197698   2.600 0.1441
## KHSRP_high - KHSRP_med <= 0                 0.916422   0.158786   5.771  <0.01 ***
## HNRNPFH_low - KHSRP_med <= 0                0.775212   0.231422   3.350 0.0162 *
## HNRNPFH_med - KHSRP_med <= 0               -0.174108   0.060215  -2.891 1.0000
## HNRNPFH_high - KHSRP_med <= 0               0.098634   0.320664   0.308 1.0000
## ELAVL1_low - KHSRP_med <= 0                 0.766877   0.152353   5.034  <0.01 ***
## ELAVL1_med - KHSRP_med <= 0                 0.091054   0.050387   1.807 0.6111
## ELAVL1_high - KHSRP_med <= 0                0.850023   0.112635   7.547  <0.01 ***
## HNRNPFH_low - KHSRP_high <= 0              -0.141210   0.274557  -0.514 1.0000
## HNRNPFH_med - KHSRP_high <= 0              -1.090530   0.159533  -6.836 1.0000
## HNRNPFH_high - KHSRP_high <= 0             -0.817789   0.353059  -2.316 1.0000
## ELAVL1_low - KHSRP_high <= 0               -0.149545   0.212218  -0.705 1.0000
## ELAVL1_med - KHSRP_high <= 0               -0.825368   0.156089  -5.288 1.0000
## ELAVL1_high - KHSRP_high <= 0              -0.066399   0.185773  -0.357 1.0000
## HNRNPFH_med - HNRNPFH_low <= 0             -0.949320   0.231936  -4.093 1.0000
## HNRNPFH_high - HNRNPFH_low <= 0            -0.676578   0.391144  -1.730 1.0000
## ELAVL1_low - HNRNPFH_low <= 0              -0.008335   0.270887  -0.031 1.0000
## ELAVL1_med - HNRNPFH_low <= 0              -0.684158   0.229581  -2.980 1.0000
## ELAVL1_high - HNRNPFH_low <= 0              0.074811   0.250709   0.298 1.0000
## HNRNPFH_high - HNRNPFH_med <= 0             0.272741   0.321035   0.850 0.9922
## ELAVL1_low - HNRNPFH_med <= 0               0.940985   0.153131   6.145  <0.01 ***
## ELAVL1_med - HNRNPFH_med <= 0               0.265162   0.052695   5.032  <0.01 ***
## ELAVL1_high - HNRNPFH_med <= 0              1.024131   0.113686   9.008  <0.01 ***
## ELAVL1_low - HNRNPFH_high <= 0              0.668244   0.350213   1.908 0.5384
## ELAVL1_med - HNRNPFH_high <= 0             -0.007580   0.319338  -0.024 1.0000
## ELAVL1_high - HNRNPFH_high <= 0             0.751390   0.334849   2.244 0.3110
## ELAVL1_med - ELAVL1_low <= 0               -0.675823   0.149540  -4.519 1.0000
## ELAVL1_high - ELAVL1_low <= 0               0.083146   0.180305   0.461 0.9998
```

```
## ELAVL1_high - ELAVL1_med <= 0     0.758969    0.108800    6.976   <0.01 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

```
pdf("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/6_LogFC_plot/3_pvalue_0.05_fc1/
    HNRNPC_LogFC_plot.pdf", width = 5, height = 7)
logFCplot
dev.off()
```

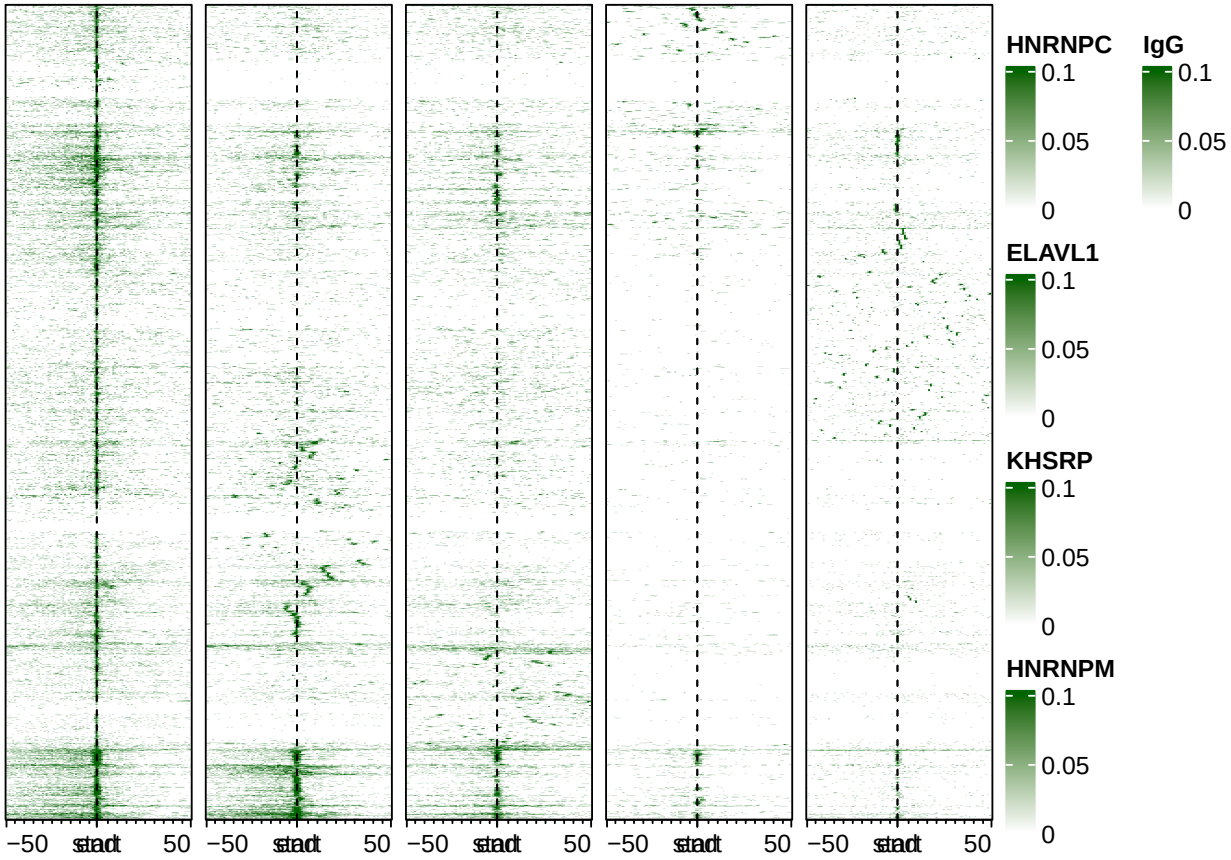
#7. Re-CLIP integration Here, we integrated the results from the Re-CLIP experiment where we sequentially pull down HNRNPC and several RDAPs followed by RNA-seq. The experiment was performed in HEK293T. These data went through the same custom snakemake pipeline found here: link.

```
#Load dds from DEWSeq
HNRNPC.bdg <- read.delim("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/1_Subzones/1_Regions_
    DEW-Seq/deseq2_norm/2_Subzone_sum/3_pvalue_0.05_fc1/HNRNPC_DEWSeq_sign_regions_sum_
    categorization.txt", header = TRUE)
HNRNPC.bdg$rbp <- "HNRNPC"
colnames(HNRNPC.bdg)[5] <- "region_strand"
HNRNPC.bdg <- subset(HNRNPC.bdg, chromosome != "chrM")

data <- HNRNPC.bdg
rbp <- "HNRNPC"

#Prepare granges
p <- subset(data, region_strand == "+" & chromosome != "chrM")
gr.p <- GRanges(p[,grep("chr_S1", colnames(p))],
                IRanges(start=p[,grep("begin_S1", colnames(p))],
                        end=p[,grep("end_S1", colnames(p))]),
                names = p[,grep("max_S1_id", colnames(p))],
                strand = p[,grep("strand_S1", colnames(p))],
                category = p[,grep("category", colnames(p))])
m <- subset(data, region_strand == "-" & chromosome != "chrM")
gr.m <- GRanges(m[,grep("chr_S1", colnames(m))],
                IRanges(start=m[,grep("begin_S1", colnames(m))],
                        end=m[,grep("end_S1", colnames(m))]),
                names = m[,grep("max_S1_id", colnames(m))],
                strand = m[,grep("strand_S1", colnames(m))],
                category = m[,grep("category", colnames(m))])
start(gr.p) <- (start(gr.p)+end(gr.p))/2
end(gr.p) <- start(gr.p)

start(gr.m) <- (start(gr.m)+end(gr.m))/2
end(gr.m) <- start(gr.m)

#Load bigwig files
setwd("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/2_293T_reclip/1_Bigwig_sum")
bw.ELAVL1.r1.p <- import.bw("HuR_293T-hnC_110-350_R1.p.scaleddewseq.bw")
bw.ELAVL1.r1.m <- import.bw("HuR_293T-hnC_110-350_R1.m.scaleddewseq.bw")
bw.ELAVL1.r2.p <- import.bw("HuR_293T-hnC_110-350_R2.p.scaleddewseq.bw")
bw.ELAVL1.r2.m <- import.bw("HuR_293T-hnC_110-350_R2.m.scaleddewseq.bw")

bw.IgG.r1.p <- import.bw("IgG_A431-hnC_EGF0_R1.p.scaleddewseq.bw")
bw.IgG.r1.m <- import.bw("IgG_A431-hnC_EGF0_R1.m.scaleddewseq.bw")
bw.IgG.r2.p <- import.bw("IgG_A431-hnC_EGF0_R2.p.scaleddewseq.bw")
bw.IgG.r2.m <- import.bw("IgG_A431-hnC_EGF0_R2.m.scaleddewseq.bw")

bw.KHSRP.r1.p <- import.bw("KHSRP_293T-hnC_140-350_R1.p.scaleddewseq.bw")
bw.KHSRP.r1.m <- import.bw("KHSRP_293T-hnC_140-350_R1.m.scaleddewseq.bw")
bw.KHSRP.r2.p <- import.bw("KHSRP_293T-hnC_140-350_R2.p.scaleddewseq.bw")
bw.KHSRP.r2.m <- import.bw("KHSRP_293T-hnC_140-350_R2.m.scaleddewseq.bw")
```

```
bw.HNRNPC.r1.p <- import.bw("hnRNPC_293T-hnC_all_R1.p.scaleddewseq.bw")
bw.HNRNPC.r1.m <- import.bw("hnRNPC_293T-hnC_all_R1.m.scaleddewseq.bw")
bw.HNRNPC.r2.p <- import.bw("hnRNPC_293T-hnC_all_R1.p.scaleddewseq.bw")
bw.HNRNPC.r2.m <- import.bw("hnRNPC_293T-hnC_all_R1.m.scaleddewseq.bw")

bw.HNRNPM.r1.p <- import.bw("hnRNPM_293T-hnC_140-350_R1.p.scaleddewseq.bw")
bw.HNRNPM.r1.m <- import.bw("hnRNPM_293T-hnC_140-350_R1.m.scaleddewseq.bw")
bw.HNRNPM.r2.p <- import.bw("hnRNPM_293T-hnC_140-350_R2.p.scaleddewseq.bw")
bw.HNRNPM.r2.m <- import.bw("hnRNPM_293T-hnC_140-350_R2.m.scaleddewseq.bw")

#Create profile matrices#
ELAVL1.mat1.p = normalizeToMatrix(bw.ELAVL1.r1.p, gr.p, value_column = "score", extend = 200,
    mean_mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
ELAVL1.mat1.m = normalizeToMatrix(bw.ELAVL1.r1.m, gr.m, value_column = "score", extend = 200,
    mean_mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
ELAVL1.mat2.p = normalizeToMatrix(bw.ELAVL1.r2.p, gr.p, value_column = "score", extend = 200,
    mean_mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
ELAVL1.mat2.m = normalizeToMatrix(bw.ELAVL1.r2.m, gr.m, value_column = "score", extend = 200,
    mean_mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)

IgG.mat1.p = normalizeToMatrix(bw.IgG.r1.p, gr.p, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
IgG.mat1.m = normalizeToMatrix(bw.IgG.r1.m, gr.m, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
IgG.mat2.p = normalizeToMatrix(bw.IgG.r2.p, gr.p, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
IgG.mat2.m = normalizeToMatrix(bw.IgG.r2.m, gr.m, value_column = "score", extend = 200, mean_mode
    = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)

KHSRP.mat1.p = normalizeToMatrix(bw.KHSRP.r1.p, gr.p, value_column = "score", extend = 200, mean_
    mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
KHSRP.mat1.m = normalizeToMatrix(bw.KHSRP.r1.m, gr.m, value_column = "score", extend = 200, mean_
    mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
KHSRP.mat2.p = normalizeToMatrix(bw.KHSRP.r2.p, gr.p, value_column = "score", extend = 200, mean_
    mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
KHSRP.mat2.m = normalizeToMatrix(bw.KHSRP.r2.m, gr.m, value_column = "score", extend = 200, mean_
    mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)

HNRNPC.mat1.p = normalizeToMatrix(bw.HNRNPC.r1.p, gr.p, value_column = "score", extend = 200,
    mean_mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
HNRNPC.mat1.m = normalizeToMatrix(bw.HNRNPC.r1.m, gr.m, value_column = "score", extend = 200,
    mean_mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
HNRNPC.mat2.p = normalizeToMatrix(bw.HNRNPC.r2.p, gr.p, value_column = "score", extend = 200,
    mean_mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
HNRNPC.mat2.m = normalizeToMatrix(bw.HNRNPC.r2.m, gr.m, value_column = "score", extend = 200,
    mean_mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)

HNRNPM.mat1.p = normalizeToMatrix(bw.HNRNPM.r1.p, gr.p, value_column = "score", extend = 200,
    mean_mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
HNRNPM.mat1.m = normalizeToMatrix(bw.HNRNPM.r1.m, gr.m, value_column = "score", extend = 200,
    mean_mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
HNRNPM.mat2.p = normalizeToMatrix(bw.HNRNPM.r2.p, gr.p, value_column = "score", extend = 200,
    mean_mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)
HNRNPM.mat2.m = normalizeToMatrix(bw.HNRNPM.r2.m, gr.m, value_column = "score", extend = 200,
    mean_mode = "w0", w = 1, smooth = FALSE, background = 0, limit = NA)

#Merge replicates#
ELAVL1.mat_list.p = list(ELAVL1.mat1.p, ELAVL1.mat2.p)
ELAVL1.mat.p = getSignalsFromList(ELAVL1.mat_list.p)
rownames(ELAVL1.mat.p) <- gr.p$names

IgG.mat_list.p = list(IgG.mat1.p, IgG.mat2.p)
IgG.mat.p = getSignalsFromList(IgG.mat_list.p)
rownames(IgG.mat.p) <- gr.p$names

KHSRP.mat_list.p = list(KHSRP.mat1.p, KHSRP.mat2.p)
KHSRP.mat.p = getSignalsFromList(KHSRP.mat_list.p)
rownames(KHSRP.mat.p) <- gr.p$names
```

```
HNRNPC.mat_list.p = list(HNRNPC.mat1.p, HNRNPC.mat2.p)
HNRNPC.mat.p = getSignalsFromList(HNRNPC.mat_list.p)
rownames(HNRNPC.mat.p) <- gr.p$names

HNRNPM.mat_list.p = list(HNRNPM.mat1.p, HNRNPM.mat2.p)
HNRNPM.mat.p = getSignalsFromList(HNRNPM.mat_list.p)
rownames(HNRNPM.mat.p) <- gr.p$names

ELAVL1.mat_list.m = list(ELAVL1.mat1.m, ELAVL1.mat2.m)
ELAVL1.mat.m = getSignalsFromList(ELAVL1.mat_list.m)
rownames(ELAVL1.mat.m) <- gr.m$names

IgG.mat_list.m = list(IgG.mat1.m, IgG.mat2.m)
IgG.mat.m = getSignalsFromList(IgG.mat_list.m)
rownames(IgG.mat.m) <- gr.m$names

KHSRP.mat_list.m = list(KHSRP.mat1.m, KHSRP.mat2.m)
KHSRP.mat.m = getSignalsFromList(KHSRP.mat_list.m)
rownames(KHSRP.mat.m) <- gr.m$names

HNRNPC.mat_list.m = list(HNRNPC.mat1.m, HNRNPC.mat2.m)
HNRNPC.mat.m = getSignalsFromList(HNRNPC.mat_list.m)
rownames(HNRNPC.mat.m) <- gr.m$names

HNRNPM.mat_list.m = list(HNRNPM.mat1.m, HNRNPM.mat2.m)
HNRNPM.mat.m = getSignalsFromList(HNRNPM.mat_list.m)
rownames(HNRNPM.mat.m) <- gr.m$names

HNRNPC.mat <- rbind(HNRNPC.mat.p, HNRNPC.mat.m)
rownames(HNRNPC.mat) <- c(gr.p$names, gr.m$names)
KHSRP.mat <- rbind(KHSRP.mat.p, KHSRP.mat.m)
rownames(KHSRP.mat) <- c(gr.p$names, gr.m$names)
ELAVL1.mat <- rbind(ELAVL1.mat.p, ELAVL1.mat.m)
rownames(ELAVL1.mat) <- c(gr.p$names, gr.m$names)
HNRNPM.mat <- rbind(HNRNPM.mat.p, HNRNPM.mat.m)
rownames(HNRNPM.mat) <- c(gr.p$names, gr.m$names)
IgG.mat <- rbind(IgG.mat.p, IgG.mat.m)
rownames(IgG.mat) <- c(gr.p$names, gr.m$names)

#Center on max nucleotide
s.sub.p <- HNRNPC.mat
get_centered_region <- function(x, y) {
  mat = list()
  for(i in 1:length(rownames(x))){
    center <- which.max(x[i, 175:225])
    shift <- y[i, (175+center-50):(175+center+50)]
    mat[[i]] <- shift
  }
  mat <- matrix(unlist(mat), ncol = length(rownames(x)))
  mat <- t(mat)
  rownames(mat) <- rownames(x)
  attr(mat, "upstream_index") = 1:49
  attr(mat, "target_index") = 50
  attr(mat, "downstream_index") = 51:101
  attr(mat, "target_is_single_point") = TRUE
  attr(mat, "extend") = c(50, 50)
  class(mat) = c("normalizedMatrix", "matrix")
  return(mat)
}

HNRNPC <- get_centered_region(s.sub.p, HNRNPC.mat)
KHSRP <- get_centered_region(s.sub.p, KHSRP.mat)
ELAVL1 <- get_centered_region(s.sub.p, ELAVL1.mat)
HNRNPM <- get_centered_region(s.sub.p, HNRNPM.mat)
IgG <- get_centered_region(s.sub.p, IgG.mat)

#Perform clustering
```

```
dist <- dist(HNRNPC) + dist(HNRNPM) + dist(KHSRP) + dist(ELAVL1) + dist(IgG)
hc <- hclust(dist, method = "ward.D2")
clusters <- cutree(hc, k = 60)

orders <- data.frame(max_S1_id = hc$labels, order = hc$order, cluster = clusters)
orders <- merge(orders, HNRNPC.bdg[,c(1:11,18)], by = "max_S1_id")
write.table(orders, "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/2_293T_reclip/2_Heatmap/3_
    pvalue_0.05_fc1/Heatmap_order.txt", quote = F, sep = "\t", row.names = F)

col_fun = colorRamp2(c(0, 0.1), c("white", "darkgreen"))

#Generate heatmap
EH <- EnrichedHeatmap(HNRNPC, name = "HNRNPC", col = col_fun, row_order = hc$order, top_
    annotation = NULL, raster_quality = 10) + EnrichedHeatmap(ELAVL1, name = "ELAVL1", col = col_
    fun, top_annotation = NULL, raster_quality = 10) + EnrichedHeatmap(KHSRP, name = "KHSRP", col
     = col_fun, top_annotation = NULL, raster_quality = 10) + EnrichedHeatmap(HNRNPM, name = "
    HNRNPM", col = col_fun, top_annotation = NULL, raster_quality = 10) + EnrichedHeatmap(IgG,
    name = "IgG", col = col_fun, top_annotation = NULL, raster_quality = 10)

EH
```




```
pdf("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/2_293T_reclip/2_Heatmap/3_pvalue_0.05_fc1/
    Heatmap_reclip_notrev.pdf", width = 5, height = 5)
EH
dev.off()

pdf("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/Seq/2_293T_reclip/2_Heatmap/3_pvalue_0.05_fc1/
    dendo_reclip_notrev.pdf", width = 5, height = 5)
plot(hc, labels=FALSE, hang = -1)
dev.off()
```

All the visualizations were saved as pdf and modified in illustrator.

```
sessionInfo()
```

```
## R version 4.2.1 (2022-06-23)
## 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] grid      stats4    stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] GenomicFeatures_1.48.4            multcomp_1.4-25
##  [3] TH.data_1.1-2                     MASS_7.3-60.0.1
##  [5] survival_3.5-8                    mvtnorm_1.2-4
##  [7] circlize_0.4.16                   gUtils_0.2.0
##  [9] data.table_1.15.2                 paletteer_1.6.0
## [11] BSgenome.Hsapiens.UCSC.hg38_1.4.4 BSgenome_1.64.0
## [13] Biostrings_2.66.0                 XVector_0.38.0
## [15] magrittr_2.0.3                    universalmotif_1.14.1
## [17] memes_1.4.1                       caTools_1.18.2
## [19] rtracklayer_1.56.1                EnrichedHeatmap_1.26.0
## [21] gintools_0.1.3                    AnnotationDbi_1.60.2
## [23] Biobase_2.58.0                    ggridges_0.5.6
## [25] eulerr_7.0.1                      ChIPpeakAnno_3.30.1
## [27] ComplexHeatmap_2.14.0             gridExtra_2.3
## [29] Repitools_1.42.0                  plyranges_1.16.0
## [31] GenomicRanges_1.50.2              GenomeInfoDb_1.34.9
## [33] IRanges_2.32.0                    S4Vectors_0.36.2
## [35] BiocGenerics_0.44.0               lubridate_1.9.3
## [37] forcats_1.0.0                     stringr_1.5.1
## [39] dplyr_1.1.4                       purrr_1.0.2
## [41] readr_2.1.5                       tidyr_1.3.1
## [43] tibble_3.2.1                      tidyverse_2.0.0
## [45] ggplot2_3.5.0                     scales_1.3.0
## [47] RColorBrewer_1.1-3                RIdeogram_0.2.2
##
## loaded via a namespace (and not attached):
##   [1] utf8_1.2.4                  tidyselect_1.2.1
##   [3] RSQLite_2.3.5               BiocParallel_1.32.6
##   [5] munsell_0.5.0               codetools_0.2-19
##   [7] preprocessCore_1.60.2       withr_3.0.0
##   [9] colorspace_2.1-0            filelock_1.0.3
##  [11] highr_0.10                  knitr_1.45
##  [13] rstudioapi_0.15.0           labeling_0.4.3
##  [15] MatrixGenerics_1.10.0       GenomeInfoDbData_1.2.9
##  [17] polyclip_1.10-6             farver_2.1.1
##  [19] bit64_4.0.5                 vctrs_0.6.5
##  [21] generics_0.1.3              lambda.r_1.2.4
##  [23] xfun_0.42                   timechange_0.3.0
##  [25] BiocFileCache_2.4.0         ggseqlogo_0.2
##  [27] regioneR_1.28.0             R6_2.5.1
##  [29] doParallel_1.0.17           clue_0.3-65
##  [31] locfit_1.5-9.9              rsvg_2.6.0
```

```
##  [33] AnnotationFilter_1.22.0      grImport2_0.3-1
##  [35] bitops_1.0-7                 cachem_1.0.8
##  [37] DelayedArray_0.24.0          Ringo_1.60.0
##  [39] BiocIO_1.6.0                 gtable_0.3.4
##  [41] Cairo_1.6-2                  affy_1.76.0
##  [43] sandwich_3.1-0               ensembldb_2.20.2
##  [45] rlang_1.1.3                  genefilter_1.78.0
##  [47] GlobalOptions_0.1.2          splines_4.2.1
##  [49] lazyeval_0.2.2               prismatic_1.1.1
##  [51] BiocManager_1.30.22          yaml_2.3.8
##  [53] RBGL_1.72.0                  tools_4.2.1
##  [55] affyio_1.68.0                gplots_3.1.3.1
##  [57] Rsolnp_1.16                  DNAcopy_1.70.0
##  [59] Rcpp_1.0.12                  base64enc_0.1-3
##  [61] progress_1.2.3               zlibbioc_1.44.0
##  [63] RCurl_1.98-1.14              prettyunits_1.2.0
##  [65] GetoptLong_1.0.5             zoo_1.8-12
##  [67] SummarizedExperiment_1.28.0  cluster_2.1.6
##  [69] magick_2.8.2                 futile.options_1.0.1
##  [71] truncnorm_1.0-9              ProtGenerics_1.30.0
##  [73] matrixStats_1.2.0            hms_1.1.3
##  [75] evaluate_0.23                xtable_1.8-4
##  [77] XML_3.99-0.16.1              VennDiagram_1.7.3
##  [79] jpeg_0.1-10                  shape_1.4.6.1
##  [81] compiler_4.2.1               biomaRt_2.52.0
##  [83] KernSmooth_2.23-22           crayon_1.5.2
##  [85] htmltools_0.5.7              tzdb_0.4.0
##  [87] geneplotter_1.76.0           DBI_1.2.2
##  [89] formatR_1.14                 dbplyr_2.4.0
##  [91] rappdirs_0.3.3               Matrix_1.6-5
##  [93] cli_3.6.2                    bedtoolsr_2.30.0-4
##  [95] vsn_3.66.0                   parallel_4.2.1
##  [97] pkgconfig_2.0.3              GenomicAlignments_1.34.1
##  [99] xml2_1.3.6                   InteractionSet_1.24.0
## [101] foreach_1.5.2                annotate_1.76.0
## [103] multtest_2.52.0              digest_0.6.35
## [105] graph_1.74.0                 polylabelr_0.2.0
## [107] rmarkdown_2.26               edgeR_3.40.2
## [109] restfulr_0.0.15              curl_5.2.1
## [111] Rsamtools_2.14.0             gtools_3.9.5
## [113] rjson_0.2.21                 lifecycle_1.0.4
## [115] futile.logger_1.4.3          limma_3.54.2
## [117] fansi_1.0.6                  pillar_1.9.0
## [119] lattice_0.22-5               KEGGREST_1.38.0
## [121] fastmap_1.1.1                httr_1.4.7
## [123] glue_1.7.0                   png_0.1-8
## [125] iterators_1.0.14             bit_4.0.5
## [127] stringi_1.8.3                rematch2_2.1.2
## [129] blob_1.2.4                   DESeq2_1.38.3
## [131] memoise_2.0.1                gsmoothr_0.1.7
```