Characterization of RI events regulated by both UPF1 and HNRNPC

Luca Ducoli

01 July, 2024

This is the pipeline used to analyze the RI events which are regulated by HNRNPC and UPF1 during EGF stimulation.

1. Characterization of HNRNPC-UPF1 RI event

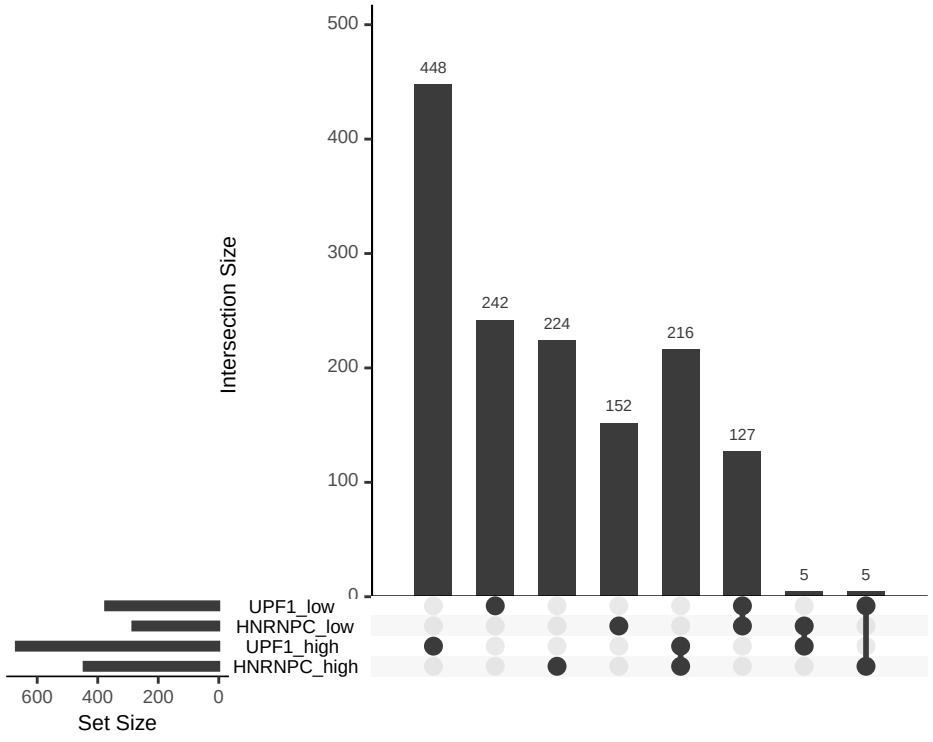
We first characterized the RI region that are co-regulated by UPF1 and HNRNPC.

```
#Needed libraries
library(ggplot2)
library(viridis)
library(hrbrthemes)
library(tidyverse)
library(tidyr)
library(stringr)
library(pheatmap)
library(UpSetR)
library(gridExtra)
library(grid)
library(RColorBrewer)
library(reshape2)
library(psych)
library(factoextra)
library(ggpubr)
library(ggrepel)
library(gprofiler2)
library(dplyr)
library(tximportData)
library(GenomicFeatures)
library(tximport)
library(SummarizedExperiment)
library(tximeta)
library(fishpond)
library(org.Hs.eg.db)
library(rtracklayer)
library(bedtoolsr)
library(DESeq2)
library(fishpond)
library(org.Hs.eg.db)
library(rstatix)
library(broom)
library(gap)
library(ggExtra)
library(ggpmisc)
library(MASS)
options(bedtools.path = "~/opt/miniconda3/bin/")
```

Upset plot AS event between UPF1 and HNRNPC

```
All_AS_overlap <- read.delim("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/2_EGF/
    1_RNA-Seq_splicing/2_Splicing_analysis/2_Correlation/Sign_AS_events_all.txt", header = TRUE)
All_AS_overlap <- subset(All_AS_overlap, AS == "RI")

list_region <- list( HNRNPC_high = unique(c(unique(All_AS_overlap$region[All_AS_overlap$hnrnpc ==
     1 & All_AS_overlap$hnrnpc_dPSI > 0]))),
              UPF1_high = unique(c(unique(All_AS_overlap$region[All_AS_overlap$upf1 == 1 & All_AS
    _overlap$upf1_dPSI > 0]))),
              HNRNPC_low = unique(c(unique(All_AS_overlap$region[All_AS_overlap$hnrnpc == 1 & All
    _AS_overlap$hnrnpc_dPSI < 0]))),
              UPF1_low = unique(c(unique(All_AS_overlap$region[All_AS_overlap$upf1 == 1 & All_AS_
    overlap$upf1_dPSI < 0]))))

upset <- upset(fromList(list_region),
  sets = c(names(list_region)[1],names(list_region)[2], names(list_region)[3], names(list_region)
    [4]),
  mb.ratio = c(0.8, 0.2),
  number.angles = 0,
  text.scale = 1,
  point.size = 3,
  line.size = 1,
  keep.order = TRUE
)
```

```
upset
```




```
#Save the plot as pdf
pdf("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/2_EGF/8_DTE_analysis/0_UPF1_
    HNRNPC/0_Overlap/Upsetplot_UPF1_HNRNPC.pdf", height = 5, width = 4)
upset
```

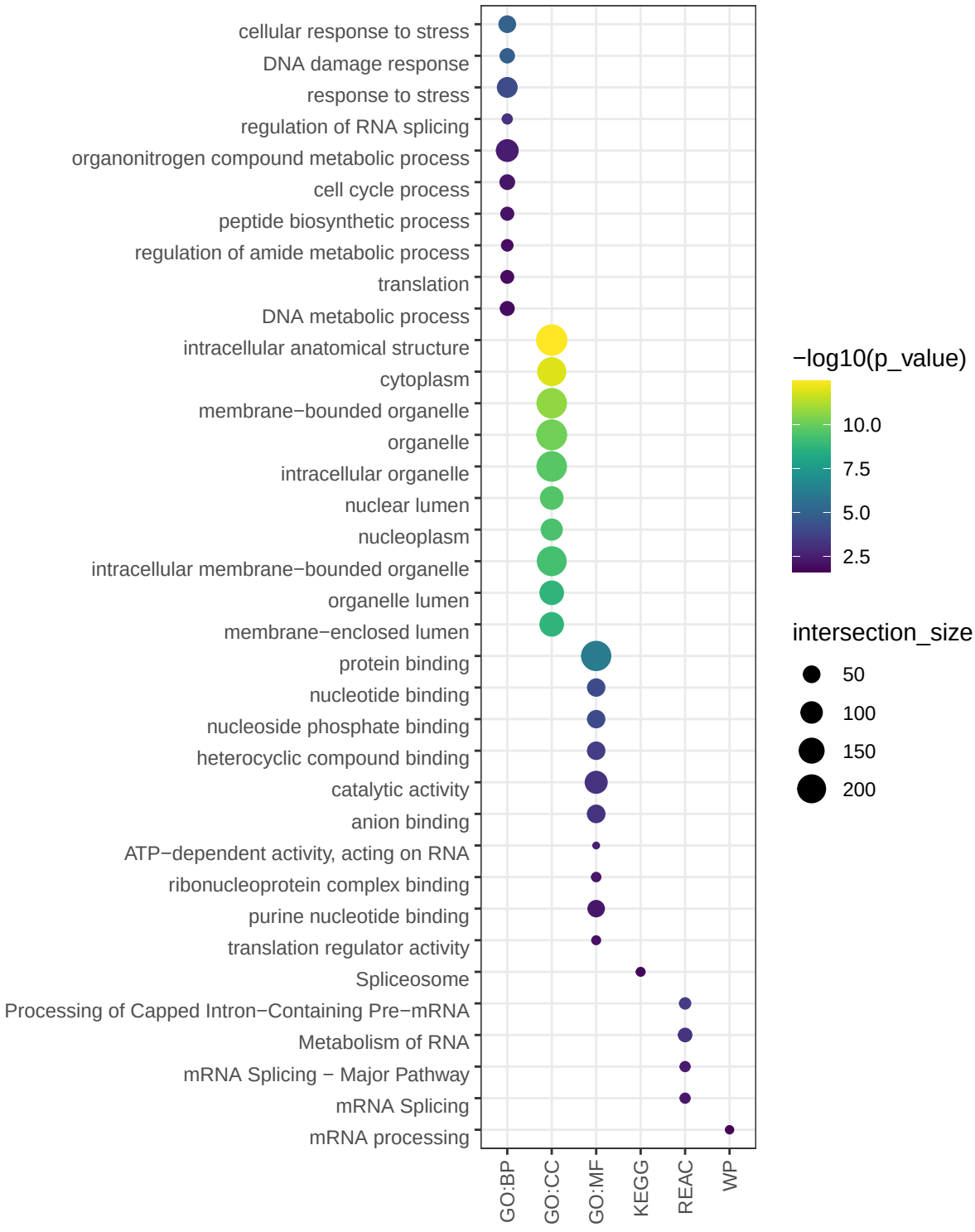
```
dev.off()
```

```
overlap <- function (input) {
  elements <- unique(unlist(input))
  data <- unlist(lapply(input, function(x) {
    x <- as.vector(match(elements, x))
  }))
  data[is.na(data)] <- as.integer(0)
  data[data != 0] <- as.integer(1)
  data <- data.frame(matrix(data, ncol = length(input), byrow = F))
  data <- data[which(rowSums(data) != 0), ]
  names(data) <- names(input)
  row.names(data) <- elements
  return(data)
}

#Binary table with colnames:
sign.regions <- overlap(list_region)

sign.regions$region <- rownames(sign.regions)
rownames(sign.regions) <- NULL

sign.regions <- merge(sign.regions, All_AS_overlap %>% dplyr::select(region, hnrnpc_dPSI, upf1_
    dPSI, AS), by = "region")
sign.regions <- sign.regions[(sign.regions$HNRNPC_high == 1 & sign.regions$UPF1_high == 1) | (
    sign.regions$HNRNPC_low == 1 & sign.regions$UPF1_low == 1),]

write.table(sign.regions, file = "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/2_
    EGF/8_DTE_analysis/0_UPF1_HNRNPC/0_Overlap/HNRNPC_UPF1_RIevents.txt", quote = F, row.names =
    F, sep = "\t")
```

Gene ontology analysis

```
#Get the genes
AS_event <- sign.regions %>% mutate("gene_id" = sapply(strsplit(sign.regions$region, "_"),
    function(x) x[1]))
AS_event <- AS_event %>% mutate("geneID" = sapply(strsplit(AS_event$gene_id, "\\."), function(x)
    x[1]))

set.seed(3)
#Run GO analysis on UVC data
RI.GO  = gost(AS_event$geneID[AS_event$AS == "RI"], organism = "hsapiens")
RI.GO$result$AS <- "RI"

gp.res <- RI.GO$result

#Take top 20 terms for each source
gp.res <- gp.res %>% group_by(source) %>% dplyr::slice(1:10)
gp.bp <- gp.res[gp.res$source %in% c("GO:BP", "GO:CC", "GO:MF", "KEGG", "REAC", "WP"),]
gp.bp$term_name <- factor(gp.bp$term_name, levels = unique(gp.bp$term_name))
gp.bp$source <- factor(gp.bp$source, levels = unique(gp.bp$source))

#Prepare the bubble plot
ggplot(data = gp.bp, aes(x = source, y = term_name, color = -log10(p_value), size = intersection_
    size)) +
  geom_point() +
  scale_color_viridis(option = "D") +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  ylab("") +
  xlab("") +
```

```
theme(axis.text.y = element_text(vjust = 1, hjust=1))+
scale_y_discrete(limits=rev)
```




```
# Save the bubble plot as pdf
```

```
pdf("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/2_EGF/8_DTE_analysis/0_UPF1_
    HNRNPC/0_Overlap/GO_AS.pdf", height = 8, width = 6.5)
ggplot(data = gp.bp, aes(x = source, y = term_name, color = -log10(p_value), size = intersection_
    size)) +
  geom_point() +
  scale_color_viridis(option = "D") +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  ylab("") +
  xlab("") +
  theme(axis.text.y = element_text(vjust = 1, hjust=1))+
  scale_y_discrete(limits=rev)
dev.off()
```

2. Differential transcript expression analysis

After Salmon quantification (PMID:28263959), we performed differential transcript analysis using the Swish R package (PMID:31372651). We then compared the logFC trend of significant deregulated RI transcripts vs the other signficant DE transcripts.

```
#Load salmon quant data
dir <- "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/2_EGF/8_DTE_analysis/0_UPF1_
    HNRNPC/1_DTE/Salmon_20/counts"
sample <- c("ctrl_1h_1_quant","ctrl_1h_2_quant","hnrnpc_1h_1_quant","hnrnpc_1h_2_quant","upf1_1h_
    1_quant","upf1_1h_2_quant")

coldata <- data.frame(condition = factor(rep(c("control", "KD_hnrnpc", "KD_upf1"), each = 2)),
    repl = factor(rep(c("1","2"),3)))
coldata$names <- sample
coldata$files <- file.path(dir, coldata$names, "quant.sf")

se <- tximeta(coldata, skipMeta = TRUE, skipSeqinfo = TRUE)
se <- scaleInfReps(se)
se <- labelKeep(se)
se <- se[mcols(se)$keep,]
write.table(assay(se), file = "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/2_EGF
    /8_DTE_analysis/0_UPF1_HNRNPC/1_DTE/DTE_norm_counts.txt", quote = FALSE, sep = "\t", row.
    names = T)
```

```
set.seed(1)
#Prepare annotation - just use the following code only once and saveDb will store the annotation
    as a database file that can be accessed later with loadDb
# txdb.filename <- "gencode.v39.annotation.sqlite"
# gtf <- file.path(dir, "gencode.v39.annotation.gtf")
# txdb <- makeTxDbFromGFF(gtf)
# saveDb(txdb, txdb.filename)

#Load annotation
txdb <- loadDb("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/2_EGF/8_DTE_analysis
    /0_UPF1_HNRNPC/1_DTE/Genome/gencode.v39.annotation.sqlite")
k <- keys(txdb, keytype = "TXNAME")
tx2gene <- AnnotationDbi::select(txdb, k, "GENEID", "TXNAME")

#HNRNPC
hnrnpc.se <- se[,se$condition %in% c("control","KD_hnrnpc")]
hnrnpc.se$condition <- factor(hnrnpc.se$condition, levels=c("control","KD_hnrnpc"))

hnrnpc.se <- swish(hnrnpc.se, x="condition")

hnrnpc.dte.res <- as.data.frame(mcols(hnrnpc.se))
```

```
hnrnpc.dte.res$TXNAME <- rownames(hnrnpc.dte.res)
rownames(hnrnpc.dte.res) <- NULL
hnrnpc.dte.res <- merge(hnrnpc.dte.res, tx2gene, by = "TXNAME")
write.table(hnrnpc.dte.res, file = "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/
    2_EGF/8_DTE_analysis/0_UPF1_HNRNPC/1_DTE/DTE_hnrnpc_results.txt", quote = FALSE, sep = "\t",
    row.names = FALSE)

#UPF1
UPF1.se <- se[,se$condition %in% c("control","KD_upf1")]
UPF1.se$condition <- factor(UPF1.se$condition, levels=c("control","KD_upf1"))

UPF1.se <- swish(UPF1.se, x="condition")

UPF1.dte.res <- as.data.frame(mcols(UPF1.se))
UPF1.dte.res$TXNAME <- rownames(UPF1.dte.res)
rownames(UPF1.dte.res) <- NULL
UPF1.dte.res <- merge(UPF1.dte.res, tx2gene, by = "TXNAME")
write.table(UPF1.dte.res, file = "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/2_
    EGF/8_DTE_analysis/0_UPF1_HNRNPC/1_DTE/DTE_upf1_results.txt", quote = FALSE, sep = "\t", row.
    names = FALSE)
```

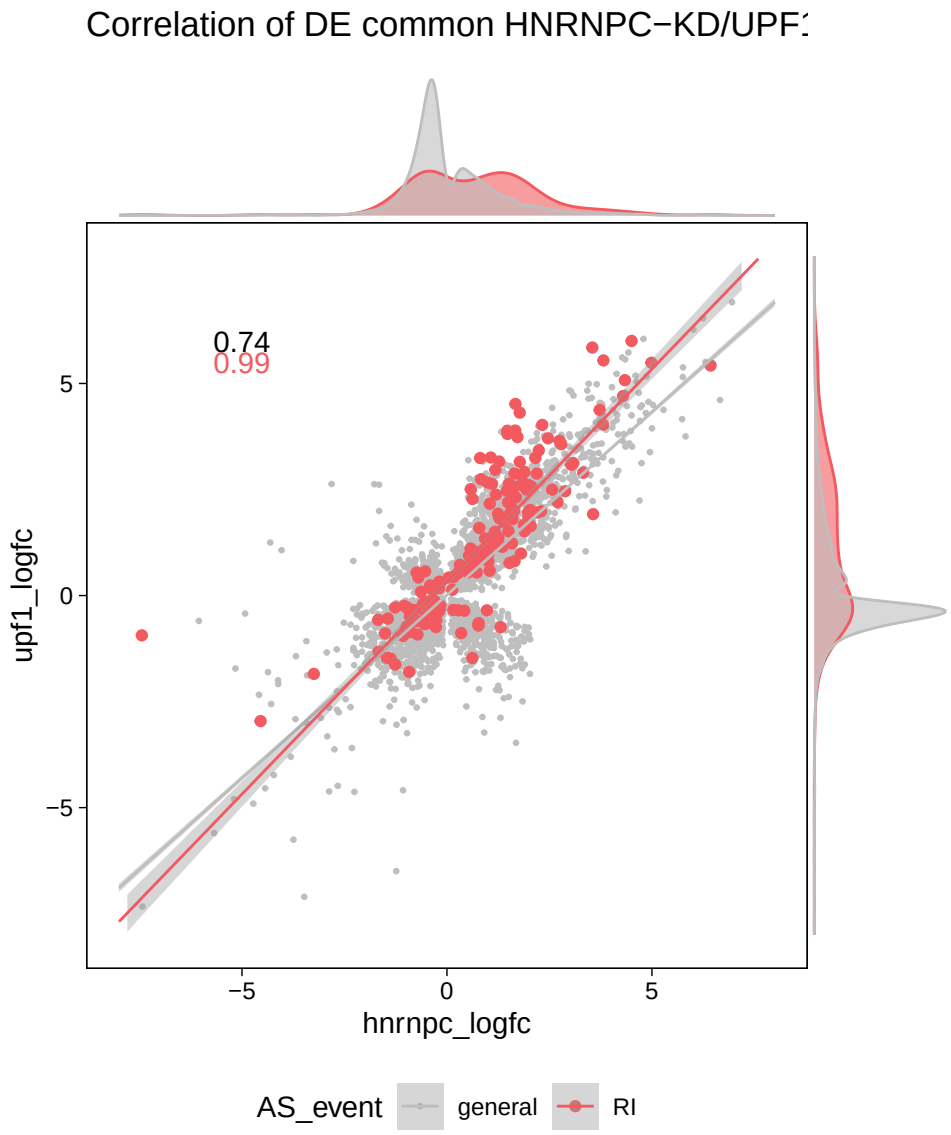
Regression analysis of DE RI transcripts

```
#Get transcript names from rMATs output
gtf <- rtracklayer::import('~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/2_EGF/8_
    DTE_analysis/0_UPF1_HNRNPC/1_DTE/Genome/gencode.v39.annotation.gtf')
gtf_df=as.data.frame(gtf)

#Splicing results
AS_event <- sign.regions
AS_event <- AS_event %>% mutate("gene_id" = sapply(strsplit(AS_event$region, "_"), function(x) x
    [1]),
                                "gene_name"= sapply(strsplit(AS_event$region, "_"), function(x) x
    [2]),
                                "chr"= sapply(strsplit(AS_event$region, "_"), function(x) x[3]),
                                "strand"= sapply(strsplit(AS_event$region, "_"), function(x) x
    [4]),
                                "start" = sapply(strsplit(AS_event$region, "_"), function(x) x
    [5]),
                                "end"= sapply(strsplit(AS_event$region, "_"), function(x) x[6]))
    %>% dplyr::select(c(region,chr,start,end, strand, gene_id, gene_name, AS))

AS_event_tx <- merge(AS_event %>% dplyr::select(region,gene_id,AS), gtf_df[gtf_df$type == "
    transcript",], by = "gene_id") %>% dplyr::select(c(seqnames,start,end,transcript_id, strand,
    gene_id, gene_name, AS))
AS_event_bed <- AS_event %>% dplyr::select(chr, start, end, strand, region, gene_id, AS)
AS_event_tx_bed <- AS_event_tx %>% dplyr::select(seqnames, start, end, strand, transcript_id,
    gene_id)

#bedtools to get regions inside transcripts
AS_event_tx2 <- bedtoolsr::bt.intersect(AS_event_bed, AS_event_tx_bed, loj = TRUE)
AS_event_tx2 <- unique(AS_event_tx2)

colnames(AS_event_tx2) <- c("AS_chr", "AS_start", "AS_end", "AS_strand", "AS_region", "AS_geneID"
    , "AS_event", "TX_chr", "TX_start", "TX_end", "TX_strand", "TX_id", "TX_geneID")
AS_event_tx2 <- AS_event_tx2 %>% mutate(geneIDmatch = case_when(AS_geneID != TX_geneID ~ "0",AS_
    geneID == TX_geneID ~ "1"))
AS_event_tx2 <- subset(AS_event_tx2, geneIDmatch > 0)

write.table(AS_event_tx2, file = "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/2_
    EGF/8_DTE_analysis/0_UPF1_HNRNPC/1_DTE/AS_HNRNPC_UPF1_transcripts.txt", quote = FALSE, row.
    names = FALSE, sep = "\t")
```

```
#Prepare DTE results
logfc.hnrnpc <- hnrnpc.dte.res %>% dplyr::select(TXNAME, log2FC, qvalue)
logfc.upf1 <- UPF1.dte.res %>% dplyr::select(TXNAME, log2FC, qvalue)

colnames(logfc.hnrnpc) <- c("TX_id", "hnrnpc_logfc", "hnrnpc_qvalue")
colnames(logfc.upf1) <- c("TX_id", "upf1_logfc", "upf1_qvalue")

logfc.table.vsupf1 <- merge(logfc.hnrnpc, logfc.upf1, by = "TX_id")
logfc.table.vsupf1.sign <- subset(logfc.table.vsupf1, hnrnpc_qvalue < 0.05 & upf1_qvalue < 0.05)

#Merge logfc table with rMATs transcript
as.tr.fc <- merge(logfc.table.vsupf1.sign, unique(AS_event_tx2 %>% dplyr::select(TX_id, TX_geneID
    , AS_event)), by = "TX_id")
write.table(as.tr.fc, file = "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/2_EGF/
    8_DTE_analysis/0_UPF1_HNRNPC/1_DTE/AS_HNRNPC_UPF1_logFC_high_sign.txt", quote = FALSE, row.
    names = FALSE, sep = "\t")

#Regression analysis
all <- setdiff(logfc.table.vsupf1.sign$TX_id, as.tr.fc$TX_id)
logfc.table.vsupf1.sign2 <- subset(logfc.table.vsupf1.sign, TX_id %in% all)
logfc.table.vsupf1.sign2$AS_event <- "general"
logfc.table.vsupf1.sign2$type <- "general"

as.tr.fc.plot <- as.tr.fc
as.tr.fc.plot$type <- "AS_event"

model <- rlm(upf1_logfc ~ hnrnpc_logfc, data = logfc.table.vsupf1.sign2)
model2 <- rlm(upf1_logfc ~ hnrnpc_logfc, data = as.tr.fc.plot)

rlm_rsquare <- function(model) {
#Extract the weights and residuals
weights <- model$w
residuals <- model$resid
fitted_values <- model$fitted.values
response <- logfc.table.vsupf1.sign2$upf1_logfc

#Calculate the weighted R-squared
weighted_ss_residuals <- sum(weights * residuals^2)
weighted_ss_total <- sum(weights * (response - mean(response))^2)
weighted_r_squared <- 1 - (weighted_ss_residuals / weighted_ss_total)

return(weighted_r_squared)
}
model.rsquare <- rlm_rsquare(model)
model2.rsquare <- rlm_rsquare(model2)

chow.test <- gap::chow.test(y1=logfc.table.vsupf1.sign2$upf1_logfc,x1=logfc.table.vsupf1.sign2$
    hnrnpc_logfc,y2=as.tr.fc.plot$upf1_logfc,x2=as.tr.fc.plot$hnrnpc_logfc)
model.res <- data.frame(general = model.rsquare, RI_event = model2.rsquare, chow.pvalue = chow.
    test[[4]])
write.table(model.res, "~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/2_EGF/8_DTE_
    analysis/0_UPF1_HNRNPC/1_DTE/Chowtest_allASevents.txt", quote = F, sep = "\t")

as.tr.fc.plot <- rbind(logfc.table.vsupf1.sign2, as.tr.fc.plot %>% dplyr::select(-TX_geneID))
as.tr.fc.plot$AS_event <- factor(as.tr.fc.plot$AS_event)

#Generate the scatter plot
ggplot3 <- ggplot(data=as.tr.fc.plot, aes(x=hnrnpc_logfc, y=upf1_logfc, group=type)) +
  geom_point(aes(color = AS_event, size = AS_event)) +
  scale_size_manual(values=c(0.5,1.5))+
  geom_smooth(aes(group=AS_event, color=AS_event), method='rlm', formula= y~x, size=0.5,
    fullrange=TRUE) +
  scale_color_manual(values=c(general = "grey", RI = "#f25a5f"))+
  annotate(geom="text", x=-5, y=6, label=round(model.rsquare,2), color="black") +
  annotate(geom="text", x=-5, y=5.5, label=round(model2.rsquare,2), color="#f25a5f") +
  xlim(-8, 8) +
  ylim(-8, 8) +
  theme_linedraw() + theme(panel.grid.major = element_blank(), legend.position = "bottom",
```

```
        panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        axis.line = element_blank()) + ggtitle("Correlation of DE common HNRNPC-KD/UPF1-KD AS
    transcripts")
ggplot3 <- ggMarginal(ggplot3, groupColour = TRUE, groupFill = TRUE, type="density", alpha=0.6,
    size = 5)
```

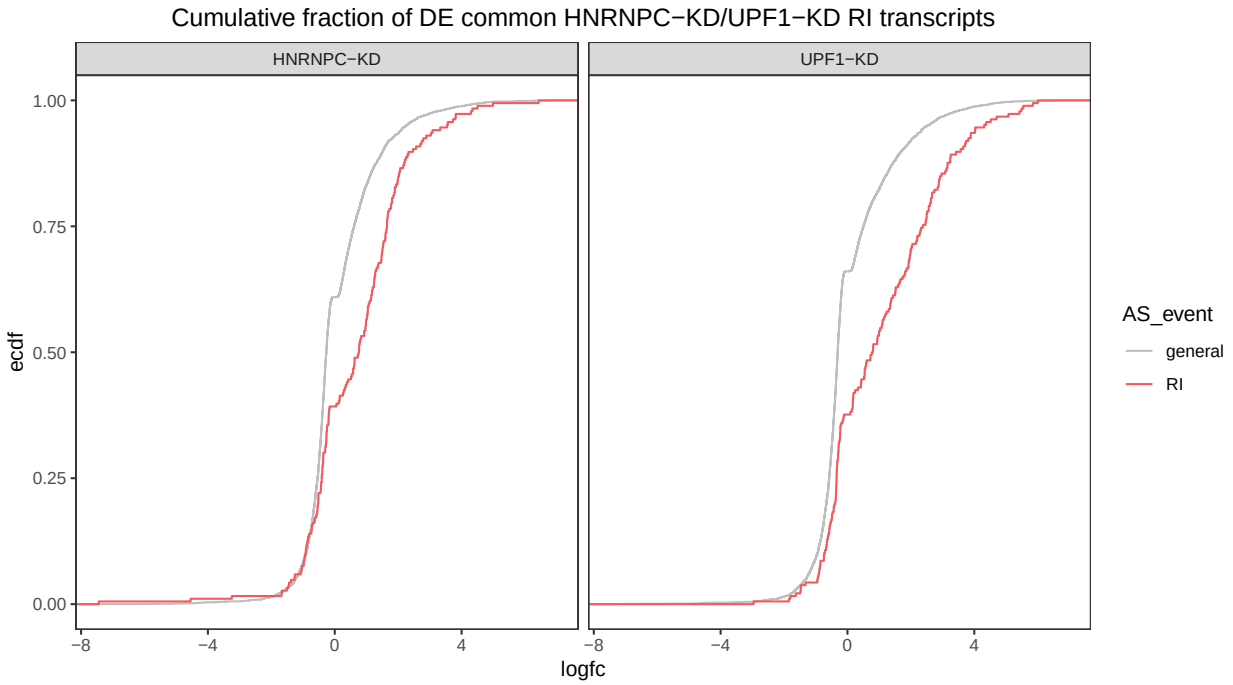
```
ggplot3
```




```
#Save the plot as pdf
pdf("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/2_EGF/8_DTE_analysis/0_UPF1_
    HNRNPC/1_DTE/Regression_DTE_HNRNPC_UPF1.pdf", height = 6, width = 5)
ggplot3
dev.off()
```

Cumulative fraction of DE RI transcripts

```
#Cumulative fraction of UPF1 RI
as.tr.fc.cdf <- as.tr.fc.plot %>% dplyr::select(TX_id, hnrnpc_logfc, AS_event) %>% dplyr::rename(
    logfc = hnrnpc_logfc)
as.tr.fc.cdf$rbp <- "HNRNPC-KD"

as.tr.fc.cdf2 <- as.tr.fc.plot %>% dplyr::select(TX_id, upf1_logfc, AS_event) %>% dplyr::rename(
    logfc = upf1_logfc)
as.tr.fc.cdf2$rbp <- "UPF1-KD"

as.tr.fc.cdf <- rbind(as.tr.fc.cdf, as.tr.fc.cdf2)

cdf <- ggplot(data=as.tr.fc.cdf, aes(x=logfc, group=AS_event, colour=AS_event)) +
  stat_ecdf() +
  ggtitle("Cumulative fraction of DE common HNRNPC-KD/UPF1-KD RI transcripts") +
  scale_color_manual(values = c(A3SS = "#7959b7", A5SS = "#00b7e6", general = "grey", MXE = "#
    ff955e", RI = "#f25a5f", SE = "#4487ab")) +
  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_grid(cols=vars(rbp))

cdf
```




```
#Save the plot as pdf
pdf("~/Documents/Postdoc/PD_Projects/3_irCLIP-RNP/MS/siRNA_EGF_SG/2_EGF/8_DTE_analysis/0_UPF1_
    HNRNPC/1_DTE/ECDF_DTE_HNRNPC_UPF1.pdf", height = 5, width = 9)
cdf
dev.off()
```

```
#KS test
```

```
hnrncp.kstest <- data.frame( ks.pvalue =  c(ks.test(as.tr.fc.cdf$logfc[as.tr.fc.cdf$AS_event == "
    general" & as.tr.fc.cdf$rbp == "HNRNPC-KD"], as.tr.fc.cdf$logfc[as.tr.fc.cdf$AS_event == "RI"
     & as.tr.fc.cdf$rbp == "HNRNPC-KD"] , alternative = 'greater')[["p.value"]]))

UPF1.kstest <- data.frame( ks.pvalue =  c(ks.test(as.tr.fc.cdf$logfc[as.tr.fc.cdf$AS_event == "
    general" & as.tr.fc.cdf$rbp == "UPF1-KD"], as.tr.fc.cdf$logfc[as.tr.fc.cdf$AS_event == "RI" &
     as.tr.fc.cdf$rbp == "UPF1-KD"] , alternative = 'greater')[["p.value"]]))

hnrncp.kstest
```

```
##       ks.pvalue
## 1 1.489919e-13
```

```
UPF1.kstest
```

```
##       ks.pvalue
## 1 2.220446e-15
```

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] stats4    grid      stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] MASS_7.3-60.0.1             ggpmisc_0.5.5
##  [3] ggpp_0.5.6                  ggExtra_0.10.1
##  [5] gap_1.5-3                   gap.datasets_0.0.6
##  [7] broom_1.0.5                 rstatix_0.7.2
##  [9] DESeq2_1.38.3               bedtoolsr_2.30.0-4
## [11] rtracklayer_1.56.1          org.Hs.eg.db_3.15.0
## [13] fishpond_2.2.0              tximeta_1.17.2
## [15] SummarizedExperiment_1.28.0 MatrixGenerics_1.10.0
## [17] matrixStats_1.2.0           tximport_1.24.0
## [19] GenomicFeatures_1.48.4      AnnotationDbi_1.60.2
## [21] Biobase_2.58.0              GenomicRanges_1.50.2
## [23] GenomeInfoDb_1.34.9         IRanges_2.32.0
## [25] S4Vectors_0.36.2            BiocGenerics_0.44.0
## [27] tximportData_1.24.0         gprofiler2_0.2.3
## [29] ggrepel_0.9.5               ggpubr_0.6.0
## [31] factoextra_1.0.7            psych_2.4.3
## [33] reshape2_1.4.4              RColorBrewer_1.1-3
```

```
## [35] gridExtra_2.3                UpSetR_1.4.0
## [37] pheatmap_1.0.12              lubridate_1.9.3
## [39] forcats_1.0.0                stringr_1.5.1
## [41] dplyr_1.1.4                  purrr_1.0.2
## [43] readr_2.1.5                  tidyr_1.3.1
## [45] tibble_3.2.1                 tidyverse_2.0.0
## [47] hrbrthemes_0.8.7             viridis_0.6.5
## [49] viridisLite_0.4.2            ggplot2_3.5.0
##
## loaded via a namespace (and not attached):
##   [1] utf8_1.2.4                    tidyselect_1.2.1
##   [3] RSQLite_2.3.5                 htmlwidgets_1.6.4
##   [5] BiocParallel_1.32.6           munsell_0.5.0
##   [7] codetools_0.2-19              miniUI_0.1.1.1
##   [9] withr_3.0.0                   colorspace_2.1-0
##  [11] filelock_1.0.3                highr_0.10
##  [13] knitr_1.45                    rstudioapi_0.15.0
##  [15] SingleCellExperiment_1.20.1   ggsignif_0.6.4
##  [17] Rttf2pt1_1.3.12               fontLiberation_0.1.0
##  [19] labeling_0.4.3                Rdpack_2.6
##  [21] GenomeInfoDbData_1.2.9        mnormt_2.1.1
##  [23] farver_2.1.1                  bit64_4.0.5
##  [25] vctrs_0.6.5                   generics_0.1.3
##  [27] xfun_0.42                     timechange_0.3.0
##  [29] BiocFileCache_2.4.0           fontquiver_0.2.1
##  [31] R6_2.5.1                      locfit_1.5-9.9
##  [33] AnnotationFilter_1.22.0       bitops_1.0-7
##  [35] cachem_1.0.8                  DelayedArray_0.24.0
##  [37] vroom_1.6.5                   promises_1.2.1
##  [39] BiocIO_1.6.0                  scales_1.3.0
##  [41] gtable_0.3.4                  ensembldb_2.20.2
##  [43] MatrixModels_0.5-3            rlang_1.1.3
##  [45] systemfonts_1.0.5             splines_4.2.1
##  [47] extrafontdb_1.0               lazyeval_0.2.2
##  [49] BiocManager_1.30.22           yaml_2.3.8
##  [51] abind_1.4-5                   backports_1.4.1
##  [53] httpuv_1.6.14                 qvalue_2.28.0
##  [55] extrafont_0.19                tools_4.2.1
##  [57] ellipsis_0.3.2                polynom_1.4-1
##  [59] Rcpp_1.0.12                   plyr_1.8.9
##  [61] progress_1.2.3                zlibbioc_1.44.0
##  [63] RCurl_1.98-1.14               prettyunits_1.2.0
##  [65] svMisc_1.2.3                  crul_1.4.0
##  [67] magrittr_2.0.3                data.table_1.15.2
##  [69] SparseM_1.81                  ProtGenerics_1.30.0
##  [71] hms_1.1.3                     mime_0.12
##  [73] evaluate_0.23                 xtable_1.8-4
##  [75] XML_3.99-0.16.1               compiler_4.2.1
##  [77] biomaRt_2.52.0                fontBitstreamVera_0.1.1
##  [79] crayon_1.5.2                  htmltools_0.5.7
##  [81] mgcv_1.9-1                    later_1.3.2
##  [83] tzdb_0.4.0                    geneplotter_1.76.0
##  [85] DBI_1.2.2                     dbplyr_2.4.0
##  [87] rappdirs_0.3.3                Matrix_1.6-5
##  [89] car_3.1-2                     cli_3.6.2
##  [91] rbibutils_2.2.16              parallel_4.2.1
##  [93] pkgconfig_2.0.3               GenomicAlignments_1.34.1
##  [95] plotly_4.10.4                 xml2_1.3.6
##  [97] annotate_1.76.0               XVector_0.38.0
##  [99] digest_0.6.35                 httpcode_0.3.0
## [101] Biostrings_2.66.0             rmarkdown_2.26
## [103] gdtools_0.3.5                 restfulr_0.0.15
## [105] curl_5.2.1                    quantreg_5.97
## [107] shiny_1.8.0                   Rsamtools_2.14.0
## [109] gtools_3.9.5                  rjson_0.2.21
## [111] lifecycle_1.0.4               nlme_3.1-164
## [113] jsonlite_1.8.8                carData_3.0-5
## [115] fansi_1.0.6                   pillar_1.9.0
```

```
## [117] lattice_0.22-5                survival_3.5-8
## [119] KEGGREST_1.38.0               fastmap_1.1.1
## [121] httr_1.4.7                    interactiveDisplayBase_1.34.0
## [123] glue_1.7.0                    png_0.1-8
## [125] BiocVersion_3.15.2            bit_4.0.5
## [127] stringi_1.8.3                 blob_1.2.4
## [129] gfonts_0.2.0                  AnnotationHub_3.4.0
## [131] memoise_2.0.1
```