

Figure 7 YTHDF1

Riccardo Mosca

2025-05-13

This markdown show how to generate Figure 7

Loading packages and required data: Input in TPM related to RAPseq and T7RAPseq, YTHDF1 metafile with binding sites of RAPseq Input and T7 input, generated in the script "YTHDF1_metafile_annotation"

```
library(eulerr)
library(ggplot2)
library(reshape2)
library(gplots)
library(tidyverse)
library(ggpubr)
library(dendextend)
library(dplyr)
library(gtools)
library(corrplot)
library(clusterProfiler)
library(org.Hs.eg.db)
library(rrvgo)
library(LSD)
library(stringr)

Inputs <- read.table(file =
"/Users/riccardomosca/Desktop/RAPseq_PAPER/Inputs_TPM.txt",
  stringsAsFactors = F, header = T)

YTHDF1 <- read.table(file =
"/Users/riccardomosca/Desktop/RAPseq_PAPER/PEAKs/ANNOTATED/T7_Fig5/YTHDF1/T7Y
THDF1_scored_annotated.txt",
  header = T, stringsAsFactors = F)
head(Inputs)

##           Geneid      tpm      tpmT7
## 1 ENSG00000000003 11.479072 12.808744
## 2 ENSG000000000419 38.450198 36.641031
## 3 ENSG000000000460  2.275249  2.251449
## 4 ENSG00000001036 37.303358 35.645586
## 5 ENSG00000001084  6.370261  6.188329
## 6 ENSG00000001167  9.356073  8.646681

head(YTHDF1)
```

```
## chr start end peak_ID RBP strand
## 1 chr1 1387212 1387220 chr1_1387202_1387251 - YTHDF1_final -
## 2 chr1 1440085 1440093 chr1_1440065_1440110 + YTHDF1_final +
## 3 chr1 2308290 2308298 chr1_2308269_2308373 + YTHDF1_final +
## 4 chr1 6224723 6224731 chr1_6224711_6224765 - YTHDF1_final -
## 5 chr1 6342574 6342582 chr1_6342564_6342591 - YTHDF1_final -
## 6 chr1 6550420 6550428 chr1_6550398_6550459 - YTHDF1_final -
## YTHDF1_rep1_signal YTHDF1_rep2_signal halo_signal input_signal
## 1 38 37 8 4
## 2 42 42 11 11
## 3 42 45 2 7
## 4 91 101 2 6
## 5 73 45 10 6
## 6 30 36 7 6
## T7YTHDF1_rep1_signal T7YTHDF1_rep2_signal T7halo_signal T7input_signal
## 1 6 5 9 2
## 2 15 14 10 10
## 3 3 3 6 10
## 4 5 4 3 6
## 5 5 6 4 1
## 6 12 9 11 3
##
pos_fa
## 1
CACTTGCCTGAGAGCACACCCGGGGGTCAAAGGGCAGCCACCGGGGGTCAAAGGGCAGCCATCAGGTACTCCCCAG
GGAAGGGCTTGC GGCCACCACTGCAACCCCGCCTCACCTCCGATGCCTGCTGTGCCCAGGGTGGTCCCGCTCA
TAGCGACGGCCTGTGCGTTCATACGACCTCGAGCGCTCTCGTCGCT
## 2
GCGGGACCCCGCAGCAGCCCCGGCCCCATCCCCGCCAGAGCCGGGCGTCGTGTGGGTCCGTGGGTGATAATTGAGA
GCGTCAGACCCAGGACTGTTCAAGGGAGGAGCCCCGGTCAGACTCCCACGTGTGAAGACCGGGCCCCAAGTGGCAAGG
GCTGGCCTGGGGCGGGCAGCTTGGGTCTGGACGTTGATAGGAAGC
## 3
TTCTAACACCCGCGGCAGCTGCACGCGCTCACAGAAGGTGGAGGTTACTTGCCCAGGTACAGACGACCTCGGGGCAG
TGACGAGCAAAGACCAGAGACTGCTGAGCCCTCGCATCTGGGTGGCGGAATTGCCTGCGGGGTTTTGCCCTTGTTT
ACTGAGGGGGGTCTTGTTGCTGCTGAAGCCCCCACCCTTCTAA
## 4
TTCGGTTAAATGAAAGGTTGTAAATGCATTTTTACCGCTGATAAGAAGGGGTACCTGCTACCCCTTTCTGCTGTGGA
GTGTTGCTGAGGACAGAGTCCATCTCTCCAGCGAGTCCTGGTTATCCTTTACTTACACTCTGGCCTCGGGGGCAGT
GGCGTGTGGCCTCGGTCTCCCAAGTAACTCTGGAGGGCGCTGTG
## 5
GGACCTTGCTGTGTTGCCCAGGCAGGAATGCAATGGCTATTACAGGCACAATCATGCTCACTGCAGCCTCAAACCTC
CTCCTGGGCTCAAGCGATCCTCCAGCCTCAGCCTCCTGAGTAGCTGCGACTACAGGCACATGCAACCACACCTAACT
TTTTTTTAGAAAAATTACTTTCAATTGCTAGCCAGGTGGAATTGAAC
## 6
CTGGCCCTAAAAATTATGAATTAGCCTTATTACAGTAGGCCCTCAGTAAATTACCCAGTTCTTTTCATTACCAAGGT
TAAGATGAGATTTGAGCAAATTCGGGCTTCCCTTTTCAGTTCCTTCTTGCTTTTCTCAGGCTGTGAGTAGTGAAGT
GCATGGATACTCAAGCAAGAGTGGGTATACACAGAGAAGATGTCTT
##
neg_fa
## 1
```

AGAAACAAGTTCGTAGAGCGAGATAAGGCCCTCTACAGGTCAAACAGCGACGCAAGCAGACTGTAACCTCCGGCTC
CCAGCTGGGACCCAGAATTACACAGACATCACAGAAATGCAGATGGGGCTGGACGCGGGGGCTCACGCCTGTGAGC
CCAGCACTTCGGGAGGCCGAGGAGGGCGGATCACAAAGTCAATACA

2

AGGGGGCTGGGGCTCTGCAGGGCCATTCCCACCGCAGTCTCTGCCACGGCCCGGAGCCCTGGGCCCAGGGCCACGC
GGGCTCCCTCTGCTGGCAAAGCCACACCCTGCACCGCATGGGGTCCACTGCCCTTTCCCCACGTGGACTCTGCGTG
ACCCAGGAAGTGCAGCATTGAGGTGGTCTCAGTCCCTCCCTCAGG

3

ACAGGTGCGAGTCCGTTTCTTTTCAGCAGAAGGGGGAAGAGGTGTCGCTGTGTGGGCTGCTGACTCCTCTGTGTGT
GAGGCCCTTATCTAAGTGATTGTGTATTAGTTTAATTCTCATTATATTTCTATACTGAAAGAAGATTTTAAACGA
AGGGAAAAACAACAGCAATAACATTCATATCTCTGGAGCAGCTAAC

4

TTACACCACAGCTGCCTGGGCCTCTGCCTCCTAAACCCACAGCCTCTGCTTCAGCACGAGATACAGTGCCACAAAAC
TCAGCTTCCGAAATGCCCCCTTCAGCAGATTACCTCCACGCTCAGAGAGCTCCAGAGACTGAACCCACCCCCTGAG
TCTGGGTTTCCAACCCAGAAGAGCAGCACTTCGCCCACTTCCCCA

5

CGATGCTTCCCCATTCTGTTTTGCACGGGCTTGTCCTCCAGCAAACCTGCCCTACTCACAACAGGTCACAGTAAAG
CGATGCCTGGAAGTGTCTCTTTTACCCCCAAGTCAATGTATGTGGCTATGGCGCCACCCATTGGCCAAAGGCTGTTA
GGGCAGGGCTGCACAGAGCCAGGCAAGAATTAGGCACCGCCTCTGG

6

ATCCCAGAACTCTGGGAGAAGGAGGTGGGGGGATCACTTGAGTGAGGCCAGGAGTTTGAGACCAGCCTGGATAACAT
GGCAAACCCCCCTGCACCCACCCCCGCCACCACCATCTCTACTAAATTACAAAAATTAGCTGGGCATGTGCCTGT
AATCCCAGCTACTCCAGAGGCTGAGGCATGAGAATCGCTTGAATCC

##	FCH_rep1	FCH_rep2	FCI_rep1	FCI_rep2	FCH_T7rep1	FCH_T7rep2	FCI_T7rep1
## 1	4.750000	4.625000	9.500000	9.250000	0.666667	0.555556	3.000000
## 2	3.818182	3.818182	3.818182	3.818182	1.500000	1.400000	1.500000
## 3	21.000000	22.500000	6.000000	6.428571	0.500000	0.500000	0.300000
## 4	45.500000	50.500000	15.166667	16.833333	1.666667	1.333333	0.833333
## 5	7.300000	4.500000	12.166667	7.500000	1.250000	1.500000	5.000000
## 6	4.285714	5.142857	5.000000	6.000000	1.090909	0.818181	4.000000

##	FCI_T7rep2	Mean_FCs_rep1	Mean_FCs_rep2	Mean_FCs_T7rep1	Mean_FCs_T7rep2
## 1	2.500000	7.125000	6.937500	1.833333	1.527778
## 2	1.400000	3.818182	3.818182	1.500000	1.400000
## 3	0.300000	13.500000	14.464286	0.400000	0.400000
## 4	0.666667	30.333333	33.666667	1.250000	1.000000
## 5	6.000000	9.733333	6.000000	3.125000	3.750000
## 6	3.000000	4.642857	5.571429	2.545455	1.909091

##	BS_rep1	BS_rep2	BS_T7rep1	BS_T7rep2	BS	BS_T7	Mean_FCH
## 1	30.79374	29.72998	3.666667	2.7612367	30.26186	3.2139517	4.687500
## 2	17.02692	17.02692	4.6311943	4.2000000	17.02692	4.4155971	3.818182
## 3	60.20233	65.87887	0.5287712	0.5287712	63.04060	0.5287712	21.750000
## 4	168.02115	191.44552	2.2591937	1.5849625	179.73333	1.9220781	48.000000
## 5	50.89384	27.32753	5.6479841	7.5000000	39.11068	6.5739921	5.900000
## 6	18.57143	23.66702	7.1459943	4.6952785	21.11923	5.9206364	4.714286

##	Mean_FCI	Mean_FCH_T7	Mean_FCI_T7	MeanFC	MeanFC_T7	gene_ID
## 1	9.375000	0.6111111	2.75	7.031250	1.680556	ENSG00000221978.12
## 2	3.818182	1.4500000	1.45	3.818182	1.450000	ENSG00000179403.12
## 3	6.214286	0.5000000	0.30	13.982143	0.400000	ENSG00000157933.10
## 4	16.000000	1.5000000	0.75	32.000000	1.125000	ENSG00000116237.16
## 5	9.833333	1.3750000	5.50	7.866667	3.437500	ENSG00000097021.20

```
## 6  5.500000  0.9545455      3.50  5.107143  2.227273  ENSG00000162408.11
##   feature gene_name      gene_type
## 1   3UTR      CCNL2 protein_coding
## 2   3UTR      VWA1  protein_coding
## 3   3UTR      SKI   protein_coding
## 4   3UTR      ICMT  protein_coding
## 5  intron      ACOT7 protein_coding
## 6   CDS       NOL9  protein_coding
```

Figure 7B

```
fits <- summary(lm(Inputs$tpm~Inputs$tpmT7))
R2 <- as.character(round(fits$adj.r.squared,2))
R2 <- paste("R2 = ", R2, sep="")
spearman <- as.character(round(cor(Inputs$tpmT7,Inputs$tpm, method =
"spearman"),2))
spearman <- paste("Spearman = ", spearman, sep="")
pearson <- as.character(round(cor(Inputs$tpmT7,Inputs$tpm, method =
"pearson"),2))
pearson <- paste("Pearson = ", pearson, sep="")
N <- paste("n = ", nrow(Inputs), sep="")

TPM_RAP <- Inputs$tpm
TPM_T7RAP <- Inputs$tpmT7

layout.matrix <- matrix(c(2, 1, 0, 3), nrow = 2, ncol = 2)
layout(mat = layout.matrix, heights = c(0.5, 2), widths = c(2, 0.5))
# scatterplot
par(mar = c(5, 5, 0, 0))
heatscatter(log2(TPM_T7RAP),log2(TPM_RAP), colpal=c("black","#9F9F9F"), alpha
= 80, cex=0.8, bty="l", las=1, xlim=c(0,18), ylim=c(0,18), main="", pch=16)
text(x=4.4,y=16,labels=spearman)
text(x=3.9,y=15,labels=pearson)
text(x=2.5,y=14,labels=R2)
text(x=2.6,y=13,labels=N)
# density plot T7RAPseq Input
par(mar = c(0.5, 5, 0.5, 0))
d1 <- density(log2(TPM_T7RAP), bw=0.2)
plot(d1$x,d1$y, main=NA, bty="n", xlab=NA, type="l", ylab="density",
ylim=c(0,0.3), bty="n", xlim=c(0,18), las=1)
abline(v=median(log2(TPM_T7RAP)))
text(x=5,y=0.29,labels=round(median(log2(TPM_T7RAP)),1))
# density plot RAPseq Input
par(mar = c(5, 0.5, 0, 0.5))
d2 <- density(log2(TPM_RAP), bw=0.2)
plot(d2$y,d2$x, main=NA, bty="n", ylab=NA, type="l", xlab="density",
xlim=c(0,0.3), bty="n", ylim=c(0,18), las=2)
```

```
abline(h=median(log2(TPM_RAP)))
text(x=0.275,y=5,labels=round(median(log2(TPM_RAP)),1))
```

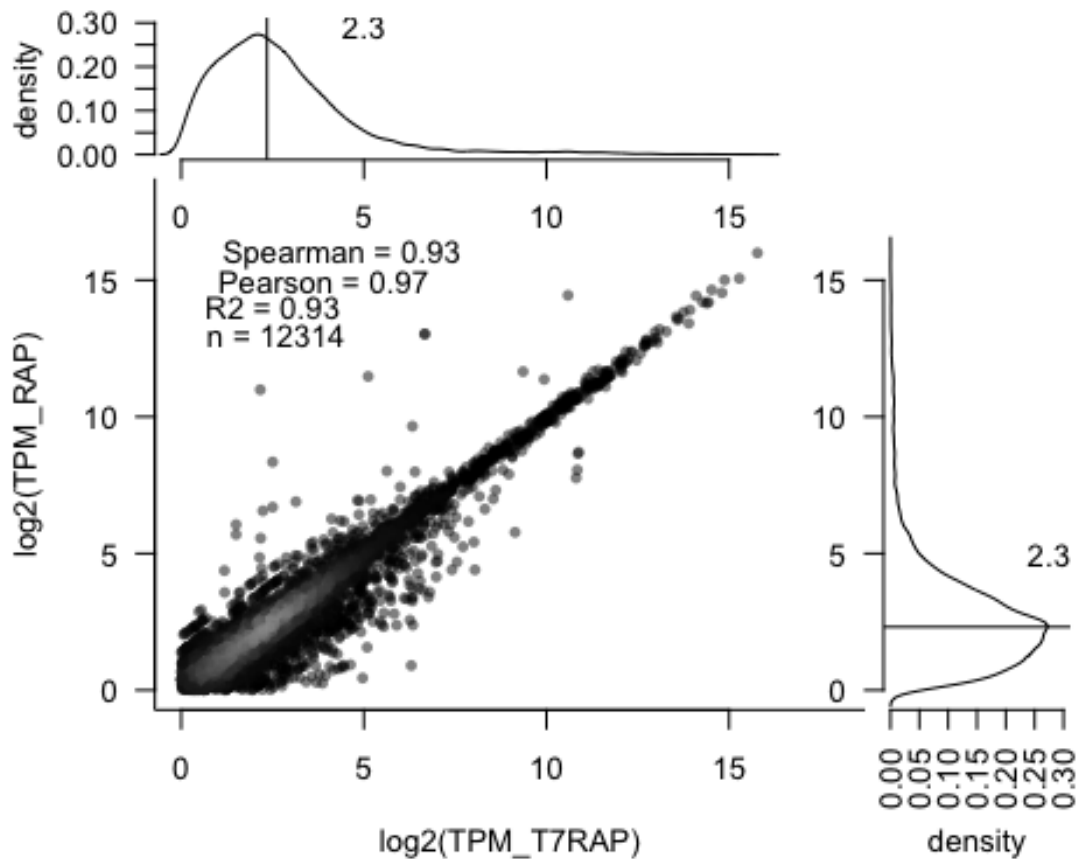


Figure 7C

```
# Only the RAPseq YTHDF1 binding sites are considered and compared with the
# signal on the T7RAPseq
YTHDF1$T7input_signal[YTHDF1$T7Input_signal == 0] <-
min(YTHDF1[YTHDF1$T7Input_signal !=
    0, "T7Input_signal"])
YTHDF1$T7halo_signal[YTHDF1$T7halo_signal == 0] <-
min(YTHDF1[YTHDF1$T7halo_signal !=
    0, "T7halo_signal"])

RAP_enrichments <- YTHDF1$peak_ID
T7RAP_enrichments <- YTHDF1[YTHDF1$T7YTHDF1_rep1_signal/YTHDF1$T7input_signal
> 1 &
    YTHDF1$T7YTHDF1_rep2_signal/YTHDF1$T7input_signal > 1 &
```

```

YTHDF1$T7YTHDF1_rep1_signal/YTHDF1$T7halo_signal >
  1 & YTHDF1$T7YTHDF1_rep2_signal/YTHDF1$T7halo_signal > 1, "peak_ID"]

AB <- list(RAP_enrichments, T7RAP_enrichments)
names(AB) <- c("with modif", "without modif")
v <- euler(AB, shape = "circle")

plot(v, fills = c("#93ADD0", "#8CBDA0"), quantities = TRUE, edges = F)

```

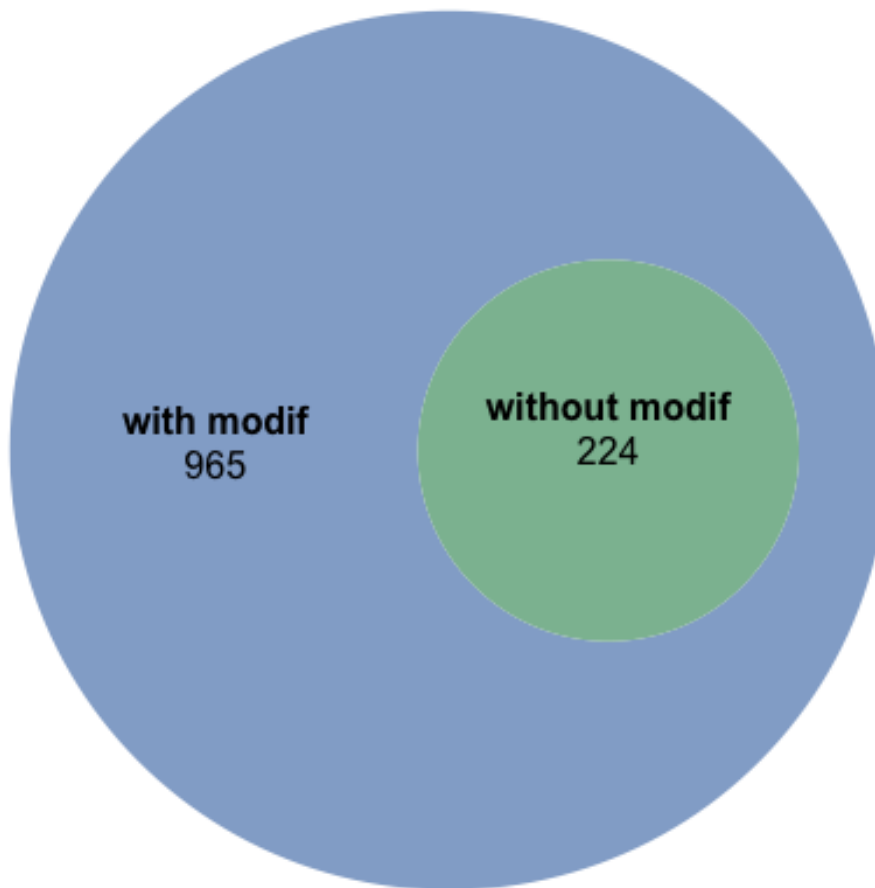


Figure 7D

```

FCs <- YTHDF1[, c("BS_rep1", "BS_rep2", "BS_T7rep1", "BS_T7rep2")]
FCs <- melt(FCs, value.name = "Binding_Score", variable.name = "Assay")
RAP <- rep("RAP", nrow(FCs)/2)
T7RAP <- rep("T7RAP", nrow(FCs)/2)
FCs$Assay_Type <- c(RAP, T7RAP)

plot <- ggplot(data = FCs, aes(x = Assay, y = log2(Binding_Score + 1))) +
  geom_jitter(aes(color = Assay_Type),

```

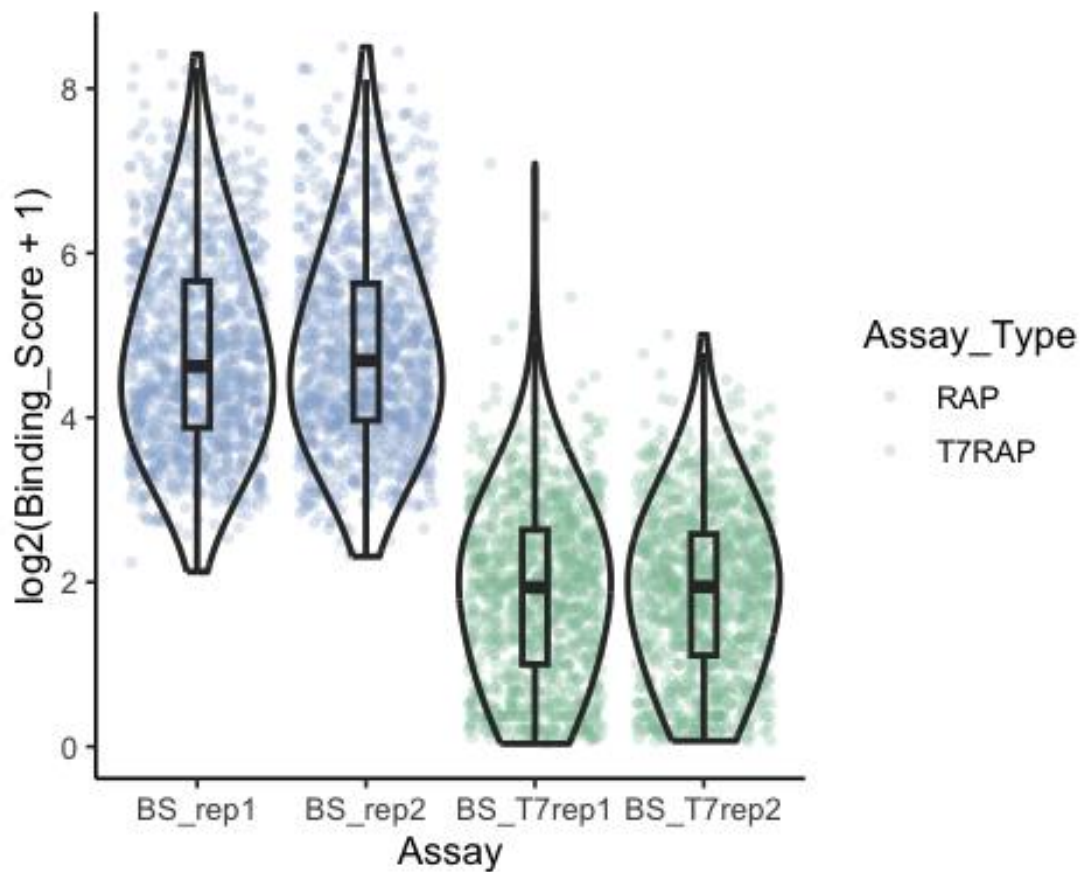
```

    pch = 16, alpha = 0.25) + scale_color_manual(values = c("#93ADD0",
"#8ABEA0")) +
    theme_classic(base_size = 12.5)

Fig7D <- plot + geom_violin(trim = T, bw = 0.75, scale = "width", lwd = 1,
fill = NA) +
    geom_boxplot(outlier.shape = NA, width = 0.15, lwd = 1, fill = NA)

```

Fig7D



```

list <- list(BS_T71 = YTHDF1$BS_T7rep1, BS_T72 = YTHDF1$BS_T7rep2)

# run wilcox.test() for each, comparing to the overlap vector
wt <- lapply(names(list), function(nm) {
  y <- list[[nm]]
  test <- wilcox.test(YTHDF1$BS_rep1, y)
  data.frame(comparison = paste0("BS1_vs_", nm), W =
as.numeric(test$statistic),
    p.value = test$p.value)
})

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

```

Figure 7E

```
# Considering all the different combinations of the DRACH motif of YTHDF1 (X.
# Wang et al., Cell, 2015)
DRACH <- c("GGACT", "GAACT", "GGACA", "GAACA", "GGACC", "GAACC", "AAACT",
"AGACT",
"AAACA", "AGACA", "AAACC", "AGACC", "TGACT", "TAACT")

COUNTS <- c()
for (i in DRACH) {
  COUNTS <- c(COUNTS, length(grep(i, str_sub(YTHDF1$pos_fa, 85, 115)))) #
  30 nucleotide window
}

names(COUNTS) <- DRACH
COUNTS <- COUNTS[order(COUNTS)]

NEG_COUNTS <- c()
for (i in names(COUNTS)) {
  NEG_COUNTS <- c(NEG_COUNTS, length(grep(i, str_sub(YTHDF1$neg_fa, 85,
115))))
}
names(NEG_COUNTS) <- names(COUNTS)

FCs <- COUNTS/NEG_COUNTS
FCs <- FCs[order(FCs)]
names(FCs) <- gsub("T", "U", names(FCs))

barplot(height = FCs, horiz = T, las = 1, xlab = "Counts: Bound sites /
Control sites",
col = "#93ADD0", xlim = c(0, 10), space = 0.1)
abline(v = c(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11), lty = 2, lwd = 1)
```

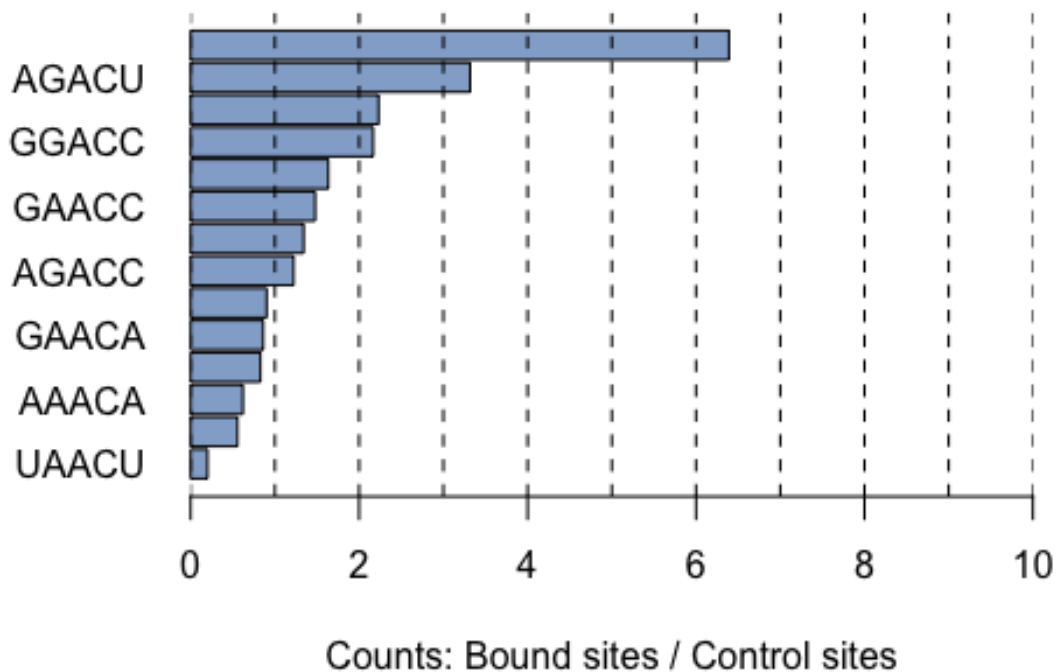



Figure 7F

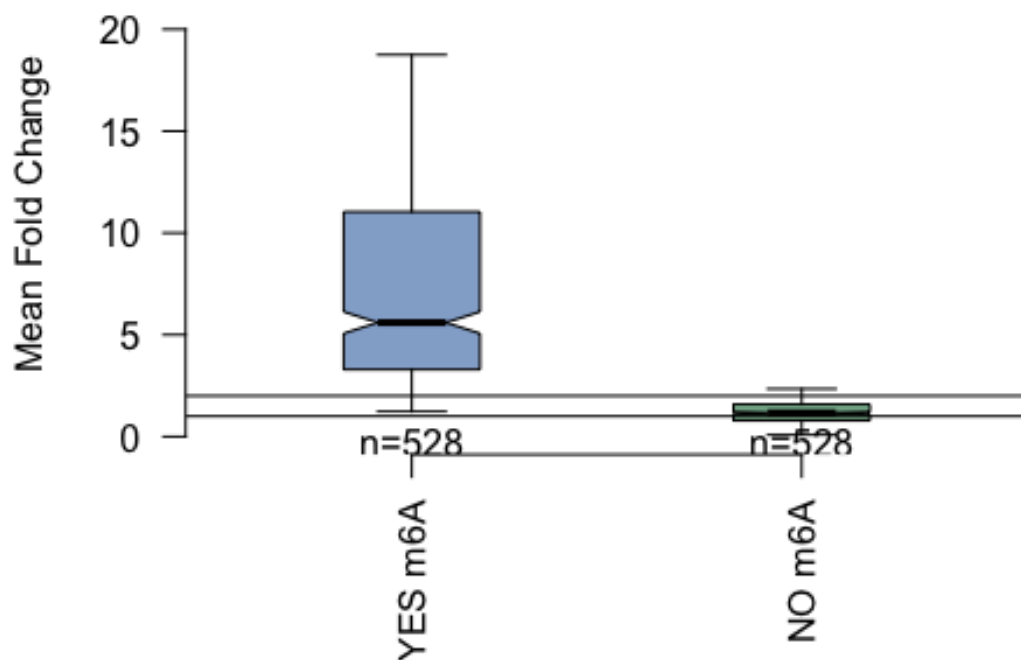
```
# Most rapresented motif
Top_motif <- YTHDF1[grep("GGACT|GGAC|GACT",str_sub(YTHDF1$pos_fa,70,130)),]

BS_Top_motif <- list(log2( Top_motif$MeanFC+1 ),log2( Top_motif$MeanFC_T7+1
))
BS_Top_motif<- list(Top_motif$MeanFC,Top_motif$MeanFC_T7)

names(BS_Top_motif) <- c("YES m6A","NO m6A")

par(bty="n")
boxplot2(BS_Top_motif, outline=F, range=1, boxwex=0.35, notch=T, las=2,
ylab="Mean Fold Change", ylim=c(0,22), lty=1, main="m6A dependency in
GGACU|GGAC|GACU peaks", col=c("93ADD0","#84B297"))
abline(h=c(1,2))
```

m6A dependency in GGACU|GGAC|GACU peaks



```
wt <- wilcox.test(BS_Top_motif$`YES m6A`, BS_Top_motif$`NO m6A`)
wt <- wt$p.value
```

Figure 7G

```
YTHDF1$positive_fa_check <- str_sub(YTHDF1$pos_fa, 85, 115)
YTHDF1$negative_fa_check <- str_sub(YTHDF1$neg_fa)
STRING_posT7 <- as.character(paste(YTHDF1$positive_fa_check, collapse =
"NN"))
STRING_negT7 <- as.character(paste(YTHDF1$negative_fa_check, collapse =
"NN"))

# Computing the counts of the most represented motif in k=5
k = 5
bases <- c("A", "C", "T", "G")
kmers <- unite(as.data.frame(permutations(n = 4, r = k, v = bases,
repeats.allowed = T)),
col = kmers, sep = "")

kmers <- cbind(kmers, str_count(STRING_posT7, kmers[, 1]))
kmers <- cbind(kmers, str_count(STRING_negT7, kmers[, 1]))
colnames(kmers) <- c("K", "pos", "neg")
```

```

kmers$pos_fraction <- kmers$pos/sum(kmers$pos)
kmers$neg_fraction <- kmers$neg/sum(kmers$neg)
kmers <- kmers[kmers$pos > 10, ]
kmers$Enr <- kmers$pos_fraction/kmers$neg_fraction
kmers$RBP <- rep("YTHDF1", nrow(kmers))
kmers$K_length <- rep("5", nrow(kmers))
kmers <- kmers[(kmers$pos_fraction - kmers$neg_fraction) > 0, ]
kmers$Norm_enr <- kmers$pos_fraction * kmers$Enr
kmers5 <- kmers

noGGACT <- kmers5[grepl("GGACT", kmers5$K, invert = T), ]
GGACT <- kmers5[grepl("GGACT", kmers5$K), ]
GGACT$motif <- "GGACT"
GGAC <- noGGACT[grepl("GGAC", noGGACT$K), ]
GGAC$motif <- rep("GGAC", nrow(GGAC))
GACT <- noGGACT[grepl("GACT", noGGACT$K), ]
GACT$motif <- rep("GACT", nrow(GACT))
noMotif <- noGGACT[grepl("GACT|GGAC|GGACT", noGGACT$K, invert = T), ]
noMotif$motif <- rep("None", nrow(noMotif))
kmers5 <- rbind(GGACT, GGAC, GACT, noMotif)

# Computing the counts of the most represented motif in k=6
k = 6
bases <- c("A", "C", "T", "G")
kmers <- unite(as.data.frame(permutations(n = 4, r = k, v = bases,
repeats.allowed = T)),
  col = kmers, sep = "")

kmers <- cbind(kmers, str_count(STRING_posT7, kmers[, 1]))
kmers <- cbind(kmers, str_count(STRING_negT7, kmers[, 1]))
colnames(kmers) <- c("K", "pos", "neg")

kmers$pos_fraction <- kmers$pos/sum(kmers$pos)
kmers$neg_fraction <- kmers$neg/sum(kmers$neg)
kmers <- kmers[kmers$pos > 10, ]
kmers$Enr <- kmers$pos_fraction/kmers$neg_fraction
kmers$RBP <- rep("YTHDF1", nrow(kmers))
kmers$K_length <- rep("6", nrow(kmers))
kmers <- kmers[(kmers$pos_fraction - kmers$neg_fraction) > 0, ]
kmers$Norm_enr <- kmers$pos_fraction * kmers$Enr
kmers6 <- kmers

noGGACT <- kmers6[grepl("GGACT", kmers6$K, invert = T), ]
GGACT <- kmers6[grepl("GGACT", kmers6$K), ]
GGACT$motif <- rep("GGACT", nrow(GGACT))
GGAC <- noGGACT[grepl("GGAC", noGGACT$K), ]
GGAC$motif <- rep("GGAC", nrow(GGAC))

```

```

GACT <- noGGACT[grep("GACT", noGGACT$K), ]
GACT$motif <- rep("GACT", nrow(GACT))
noMotif <- noGGACT[grep("GACT|GGAC|GGACT", noGGACT$K, invert = T), ]
noMotif$motif <- rep("None", nrow(noMotif))
kmers6 <- rbind(GGACT, GGAC, GACT, noMotif)

# Computing the counts of the most represented motif in k=6
k = 7
bases <- c("A", "C", "T", "G")
kmers <- unite(as.data.frame(permutations(n = 4, r = k, v = bases,
repeats.allowed = T)),
  col = kmers, sep = "")

kmers <- cbind(kmers, str_count(STRING_posT7, kmers[, 1]))
kmers <- cbind(kmers, str_count(STRING_negT7, kmers[, 1]))
colnames(kmers) <- c("K", "pos", "neg")

kmers$pos_fraction <- kmers$pos/sum(kmers$pos)
kmers$neg_fraction <- kmers$neg/sum(kmers$neg)
kmers <- kmers[kmers$pos > 10, ]
kmers$Enr <- kmers$pos_fraction/kmers$neg_fraction
kmers$RBP <- rep("YTHDF1", nrow(kmers))
kmers$K_length <- rep("7", nrow(kmers))
kmers <- kmers[(kmers$pos_fraction - kmers$neg_fraction) > 0, ]
kmers$Norm_enr <- kmers$pos_fraction * kmers$Enr
kmers7 <- kmers

noGGACT <- kmers7[grep("GGACT", kmers7$K, invert = T), ]
GGACT <- kmers7[grep("GGACT", kmers7$K), ]
GGACT$motif <- rep("GGACT", nrow(GGACT))
GGAC <- noGGACT[grep("GGAC", noGGACT$K), ]
GGAC$motif <- rep("GGAC", nrow(GGAC))
GACT <- noGGACT[grep("GACT", noGGACT$K), ]
GACT$motif <- rep("GACT", nrow(GACT))
noMotif <- noGGACT[grep("GACT|GGAC|GGACT", noGGACT$K, invert = T), ]
noMotif$motif <- rep("None", nrow(noMotif))
kmers7 <- rbind(GGACT, GGAC, GACT, noMotif)

kmers <- rbind(kmers5, kmers6, kmers7)
kmers$K_length <- factor(kmers$K_length, levels = c("5", "6", "7"))

plot <- ggplot() + geom_jitter(data = kmers[kmers$motif != "None", ], aes(x =

```

```

K_length,
  y = Norm_enr, size = log2(Norm_enr + 2), color = motif), width = 0.15,
alpha = 0.75) +
  scale_color_manual(values = c("#9ECAE1", "#DEEBF7", "#08306B")) + ylim(0,
0.06)

Fig7G <- plot + geom_jitter(data = kmers[kmers$motif == "None", ], aes(x =
K_length,
  y = Norm_enr, size = log2(Norm_enr + 2)), width = 0.25, alpha = 0.25,
color = "grey75") +
  theme(panel.background = element_blank(), panel.grid.major =
element_line(size = 0.25,
  linetype = 2, color = alpha("grey40", 0.5)), panel.grid.minor =
element_line(size = 0.1,
  linetype = 2, color = alpha("grey40", 0.5)), axis.text =
element_text(size = 20,
  color = "black"), panel.border = element_rect(fill = NA, color =
"black",
  size = 1), axis.ticks.length = unit(3, "mm"), axis.ticks =
element_line(size = 0.75,
  color = "black"))

```

Fig7G

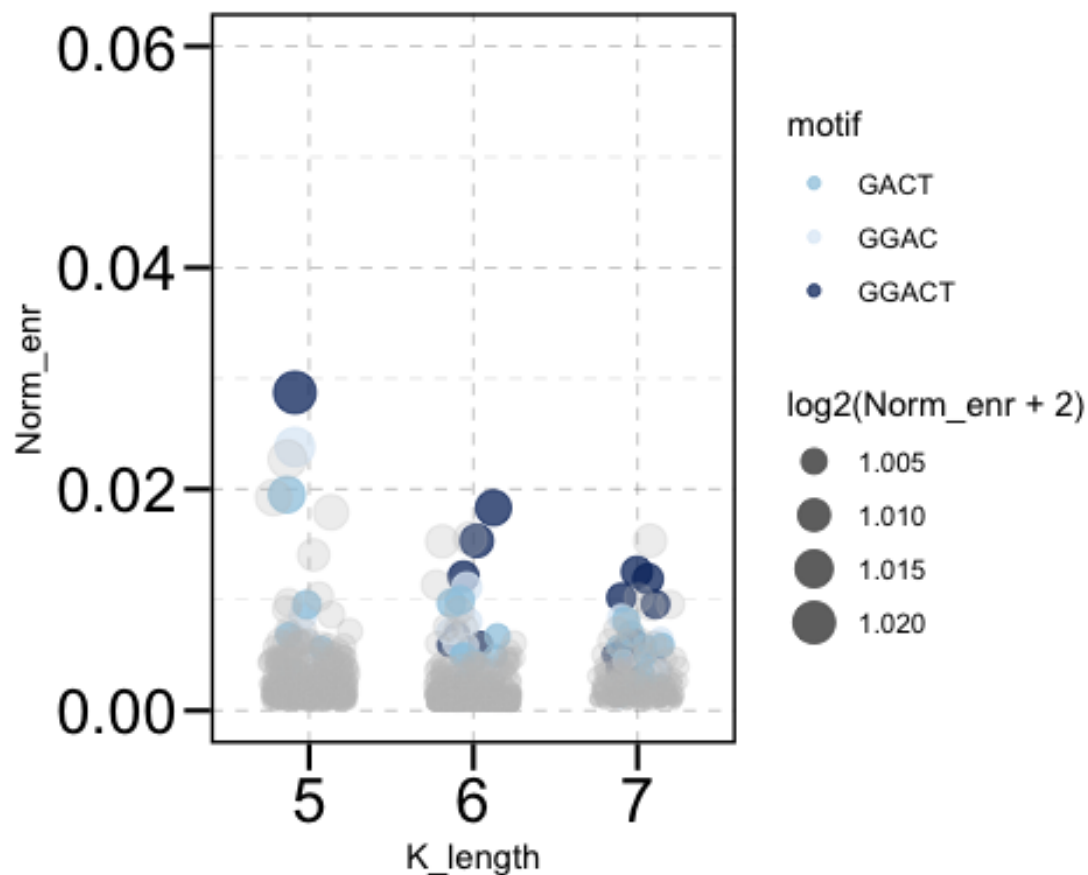


Figure 7H

```
YTHDF1$BS_T7[YTHDF1$BS_T7 == 0] <- min(YTHDF1$BS_T7[YTHDF1$BS_T7 != 0])
YTHDF1$m6A_dependency <- YTHDF1$BS/YTHDF1$BS_T7
YTHDF1$positive_fa_check <- str_sub(YTHDF1$pos_fa,70,130)

GGACT <-
YTHDF1[grep("GGACT",YTHDF1$positive_fa_check),c("peak_ID","positive_fa_check",
"m6A_dependency","MeanFC","BS","MeanFC_T7","BS_T7")]
GGACT$motif <- rep("GGACT",nrow(GGACT))

noGGACT <- YTHDF1[grep("GGACT",YTHDF1$positive_fa_check, invert =
T),c("peak_ID","positive_fa_check","m6A_dependency","MeanFC","BS",
"MeanFC_T7","BS_T7")]
noGGACT$motif <- rep("noGGACT",nrow(noGGACT))

GACT <- noGGACT[grep("GACT",noGGACT$positive_fa_check),]
GACT <- GACT[grep("GGAC",GACT$positive_fa,invert = T),]
GACT$motif <- rep("GACT",nrow(GACT))
```

```

GGAC <- noGGACT[grep("GGAC",noGGACT$positive_fa_check),]
GGAC <- GGAC[grep("GACT",GGAC$positive_fa,invert = T),]
GGAC$motif <- rep("GGAC",nrow(GGAC))

all_4mers <- noGGACT[grep("GACT|GGAC",noGGACT$positive_fa_check),]
unique_4mers <- rbind(GACT,GGAC)

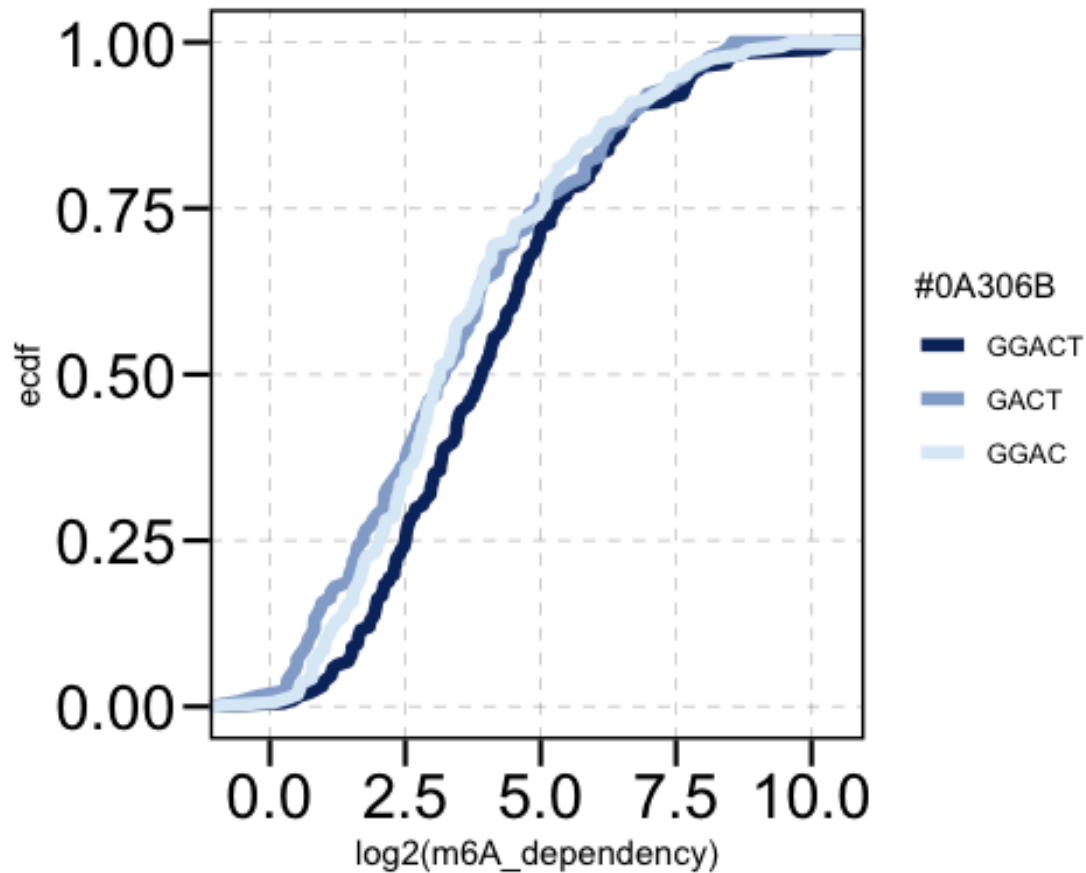
two_or_more_4mers <-
all_4mers[grep(paste(setdiff(all_4mers$Peak_ID,unique_4mers$Peak_ID),collapse
= "|"),all_4mers$Peak_ID),]
two_or_more_4mers$motif <- rep("2orMORE", nrow(two_or_more_4mers))

YTH_motifs <- rbind(GGACT,unique_4mers,two_or_more_4mers)
YTH_motifs$motif <- factor(YTH_motifs$motif, levels =
c("GGACT","GACT","GGAC"))

Fig7H <- ggplot(data=YTH_motifs) +
  stat_ecdf(aes(log2(m6A_dependency), color=motif), geom = "line", lwd=2) +
  theme(panel.background = element_blank(),
    panel.grid.major = element_line(size = 0.25, linetype = 2,
color=alpha("grey50",0.5)),
    panel.grid.minor = element_blank(),
    axis.text = element_text(size=20,color = "black"),
    panel.border = element_rect(fill=NA,color="black", size = 1),
    axis.ticks.length = unit(3,"mm"),
    axis.ticks = element_line(size = 0.75, color = "black")) +
  scale_color_manual(values = c("#0A306B", "#93ACD1",
"#DDEBF7"),c("#0A306B", "#93ACD1", "#DDEBF7"))

```

Fig7H



```
list <- list(
  GACU = YTH_motifs[YTH_motifs$motif == "GACT", "m6A_dependency"],
  GGAC = YTH_motifs[YTH_motifs$motif == "GGAC", "m6A_dependency"]
)

# run wilcox.test() for each, comparing to the overlap vector
wt <- lapply(names(list), function(nm) {
  y <- list[[nm]]
  test <- wilcox.test(YTH_motifs[YTH_motifs$motif ==
    "GGACT", "m6A_dependency"], y)
  data.frame(
    comparison = paste0("GGACU_vs_", nm),
    W          = as.numeric(test$statistic),
    p.value    = test$p.value
  )
})

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

Figure 7I


```

# Extract unique gene IDs from the 'gene_ID' column
gene_IDs <- YTHDF1 %>%
  mutate(gene_ID = strsplit(gene_ID, "\\.") %>%
    lapply(., function(x) x[1]) %>%
    unlist())
gene_IDs <- unique(gene_IDs)

expressed_genes <- Inputs %>%
  mutate(Geneid = strsplit(Geneid, "\\.") %>%
    lapply(., function(x) x[1]) %>%
    unlist())

expressed_genes <- unique(c(expressed_genes$Geneid, gene_IDs$gene_ID))

ALL_GOs <- enrichGO(gene = gene_IDs$gene_ID, universe = expressed_genes,
  keyType = "ENSEMBL",
  OrgDb = org.Hs.eg.db, ont = "ALL", pAdjustMethod = "BH", pvalueCutoff =
  0.05,
  qvalueCutoff = 0.05, minGSSize = 15, readable = TRUE)
ALL_GO <- as.data.frame(ALL_GOs)

# Molecular function
MF <- ALL_GO[ALL_GO$ONTOLOGY == "MF", ]
# Collapse child GOs into parent GOs
simMatrix <- calculateSimMatrix(MF$ID, orgdb = "org.Hs.eg.db", ont = "MF",
  method = "Rel")
scores <- setNames(-log10(MF$qvalue), MF$ID)
reducedTerms_MF <- reduceSimMatrix(simMatrix, scores, threshold = 0.7, orgdb
  = "org.Hs.eg.db")
MFs <- reducedTerms_MF[, c("go", "parentTerm")]
MFs$Ont <- rep("GO:MF", nrow(MFs))
colnames(MFs) <- c("ID", "parentTerm", "Ont")

# Cellular component
CC <- ALL_GO[ALL_GO$ONTOLOGY == "CC", ]
# Collapse child GOs into parent GOs
simMatrix <- calculateSimMatrix(CC$ID, orgdb = "org.Hs.eg.db", ont = "CC",
  method = "Rel")
scores <- setNames(-log10(CC$qvalue), CC$ID)
reducedTerms_CC <- reduceSimMatrix(simMatrix, scores, threshold = 0.7, orgdb
  = "org.Hs.eg.db")
CCs <- reducedTerms_CC[, c("go", "parentTerm")]
CCs$Ont <- rep("GO:CC", nrow(CCs))
colnames(CCs) <- c("ID", "parentTerm", "Ont")

```

```

# Biological process
BP <- ALL_GO[ALL_GO$ONTOLOGY == "BP", ]
# Collapse child GOs into parent GOs
simMatrix <- calculateSimMatrix(BP$ID, orgdb = "org.Hs.eg.db", ont = "BP",
method = "Rel")
scores <- setNames(-log10(BP$qvalue), BP$ID)
reducedTerms_BP <- reduceSimMatrix(simMatrix, scores, threshold = 0.7, orgdb
= "org.Hs.eg.db")
BPs <- reducedTerms_BP[, c("go", "parentTerm")]
BPs$Ont <- rep("GO:BP", nrow(BPs))
colnames(BPs) <- c("ID", "parentTerm", "Ont")

Parent_GOs <- rbind(BPs, CCs, MFs)
Child_GOs <- as.data.frame(ALL_GOs)
Child_GOs <- Child_GOs[, c("ID", "p.adjust", "geneID")]

merged_GO <- merge(Parent_GOs, Child_GOs, by = "ID")

# Initial dummy setup to allow rbind to work
ontology_vector <- c("O", "O")
parent_term_vector <- c("pT", "pT")
fdr_vector <- c("pp", "pp")
gene_vector <- c("gg", "gg")

# Initialize the result data frame
go_gene_mapping <- data.frame(ontology_vector, parent_term_vector,
fdr_vector, gene_vector)
colnames(go_gene_mapping) <- c("Ontology", "parentTerm", "FDR", "gene_name")

# Loop over each parent term
for (term in unique(merged_GO$parentTerm)) {

  gene_vector <- paste(merged_GO[grepl(term, merged_GO$parentTerm),
"geneID"], collapse = "/")
  gene_vector <- unique(unlist(str_split(gene_vector, "\\|")))

  fdr_vector <- rep(median(merged_GO[grepl(term, merged_GO$parentTerm),
"p.adjust"]),
length(gene_vector))
  parent_term_vector <- rep(term, length(gene_vector))

  ontology_vector <- unique(merged_GO[grepl(term, merged_GO$parentTerm),
"Ont"])

```

```

ontology_vector <- rep(ontology_vector, length(gene_vector))

current_block <- data.frame(ontology_vector, parent_term_vector,
fdr_vector,
gene_vector)
colnames(current_block) <- c("Ontology", "parentTerm", "FDR",
"gene_name")

go_gene_mapping <- rbind(go_gene_mapping, current_block)
}

# Remove the initial dummy rows
go_gene_mapping <- go_gene_mapping[-c(1, 2), ]

# Extract BS values and merge with GO mapping
BS_values <- YTHDF1[, c("gene_name", "feature", "BS", "BS_T7")]
GO_BS_value <- merge(BS_values, go_gene_mapping, by = "gene_name")
GO_BS_value$BS_T7[GO_BS_value$BS_T7 == 0] <-
min(GO_BS_value$BS_T7[GO_BS_value$BS_T7 !=
0])

# Summarize total BS (binding sites) per parent GO term
bs_per_pathway <- GO_BS_value[, c("parentTerm", "BS")]
bs_per_pathway <- bs_per_pathway %>%
group_by(parentTerm) %>%
summarize(Pathway_BS = sum(BS))
bs_per_pathway <- as.data.frame(bs_per_pathway)

# Summarize total BS_T7 per parent GO term
bs_t7_per_pathway <- GO_BS_value[, c("parentTerm", "BS_T7")]
bs_t7_per_pathway <- bs_t7_per_pathway %>%
group_by(parentTerm) %>%
summarize(Pathway_BST7 = sum(BS_T7))
bs_t7_per_pathway <- as.data.frame(bs_t7_per_pathway)

# Merge BS and BS_T7 summaries
merged_bs_data <- merge(bs_per_pathway, bs_t7_per_pathway, by = "parentTerm")

# Add Ontology info (only unique pairings)
binding_site_counts <- unique(GO_BS_value[, c("parentTerm", "Ontology")])
merged_bs_data <- merge(merged_bs_data, binding_site_counts, by =
"parentTerm")

# Calculate m6A dependency score
merged_bs_data$m6A_dependency <-
merged_bs_data$Pathway_BS/merged_bs_data$Pathway_BST7

```

```

# Count how many genes (binding sites) per parentTerm
binding_site_counts <-
as.data.frame(table(as.character(GO_BS_value$parentTerm)))
colnames(binding_site_counts) <- c("parentTerm", "Binding_Sites")
merged_bs_data <- merge(merged_bs_data, binding_site_counts, by =
"parentTerm")

# Merge FDR values
fdr_info <- unique(GO_BS_value[, c("parentTerm", "FDR")])
merged_bs_data <- merge(merged_bs_data, fdr_info, by = "parentTerm")

# Filter top 10 pathways by m6A dependency
merged_bs_data$FDR <- as.numeric(merged_bs_data$FDR)
top_pathways <- merged_bs_data %>%
  top_n(10, m6A_dependency)

# Remove duplicates and order factor levels by dependency
top_pathways <- top_pathways[!duplicated(top_pathways$parentTerm), ]
top_pathways$parentTerm <- factor(top_pathways$parentTerm, levels =
top_pathways[order(top_pathways$m6A_dependency),
"parentTerm"])

term_selected <- unique(GO_BS_value[grepl("stem cell population
maintenance|hormone transport",
GO_BS_value$parentTerm), c("gene_name", "parentTerm")])
term_selected <- merge(term_selected, YTHDF1, by = "gene_name")
term_selected <- unique(term_selected[, c("gene_name", "parentTerm",
"peak_ID", "BS",
"BS_T7", "MeanFC", "MeanFC_T7", "feature", "YTHDF1_rep1_signal",
"YTHDF1_rep2_signal")])
term_selected$Signal <- (term_selected$YTHDF1_rep1_signal +
term_selected$YTHDF1_rep2_signal)/2
term_selected$BS_T7[term_selected$BS_T7 < 1] <- 1
term_selected$m6A_dependency_sites <- term_selected$BS/term_selected$BS_T7
term_selected <- term_selected[term_selected$m6A_dependency_sites >= 10, ]
term_selected$Rank <- rank(-term_selected$m6A_dependency_sites)
term_selected <- term_selected[order(term_selected$Rank), ]
term_selected$COLOR <- gsub("hormone transport", "#DFC27D",
term_selected$parentTerm)
term_selected$COLOR <- gsub("stem cell population maintenance", "#E31A1C",
term_selected$COLOR)

```

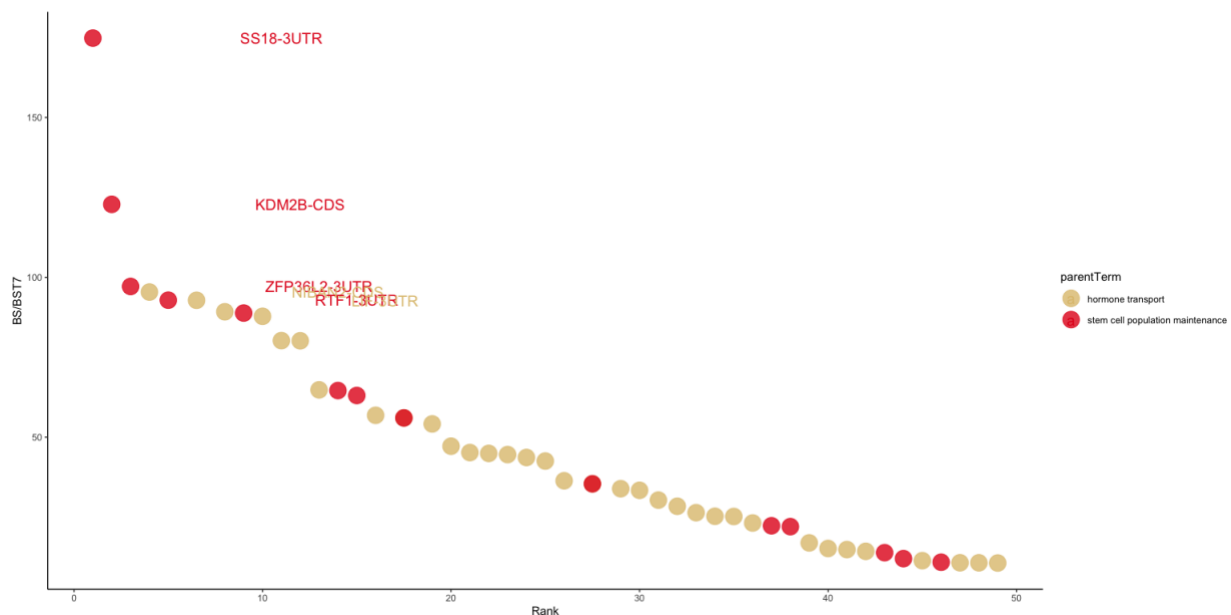
```
# The best two for each term are showed in the figure, LIF belong to both, so
# the stem cell population maintenance one is deleted
```

```
term_selected <- term_selected %>%
  filter(!(gene_name == "LIF" & parentTerm == "stem cell population
maintenance"))
top6 <- head(term_selected, 6)
```

```
plot <- ggplot(data = term_selected, aes(x = Rank, y =
(m6A_dependency_sites), color = parentTerm)) +
  geom_point(size = 7, alpha = 0.8) + theme_classic() + ylab("BS/BST7")
```

```
Fig7I <- plot + geom_text(data = top6, aes(x = Rank + 10, y =
(m6A_dependency_sites),
  color = parentTerm, label = paste(gene_name, feature, sep = "-")), size =
5) +
  scale_color_manual(values = c("#DFC27D", "#E31A1C"))
```

Fig7I



```
# In the manuscript only the top 2 for each class are showed.
```

```
sessionInfo()
```

```
## R version 4.3.2 (2023-10-31)
## Platform: aarch64-apple-darwin20 (64-bit)
## Running under: macOS Sonoma 14.5
##
## Matrix products: default
```

```

## BLAS:   /Library/Frameworks/R.framework/Versions/4.3-
arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.3-
arm64/Resources/lib/libRlapack.dylib; LAPACK version 3.11.0
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: Europe/Stockholm
## tzcode source: internal
##
## attached base packages:
## [1] stats4      stats      graphics  grDevices  utils      datasets  methods
## [8] base
##
## other attached packages:
## [1] LSD_4.1-0          rrvgo_1.14.2          org.Hs.eg.db_3.18.0
## [4] AnnotationDbi_1.64.1 IRanges_2.36.0        S4Vectors_0.40.2
## [7] Biobase_2.62.0     BiocGenerics_0.48.1   clusterProfiler_4.10.1
## [10] corrplot_0.95      gtools_3.9.5          dendextend_1.19.0
## [13] ggpubr_0.6.0       lubridate_1.9.4       forcats_1.0.0
## [16] stringr_1.5.1      dplyr_1.1.4           purrr_1.0.4
## [19] readr_2.1.5        tidyr_1.3.1           tibble_3.2.1
## [22] tidyverse_2.0.0    gplots_3.2.0          reshape2_1.4.4
## [25] ggplot2_3.5.2      eulerr_7.0.2
##
## loaded via a namespace (and not attached):
## [1] RColorBrewer_1.1-3      rstudioapi_0.17.1      jsonlite_2.0.0
## [4] umap_0.2.10.0          magrittr_2.0.3         farver_2.1.2
## [7] rmarkdown_2.29         fs_1.6.6               zlibbioc_1.48.2
## [10] vctrs_0.6.5            memoise_2.0.1          RCurl_1.98-1.17
## [13] askpass_1.2.1          ggtree_3.10.1          rstatix_0.7.2
## [16] htmltools_0.5.8.1      broom_1.0.8            Formula_1.2-5
## [19] gridGraphics_0.5-1     KernSmooth_2.23-26     plyr_1.8.9
## [22] cachem_1.1.0           igraph_2.1.4           mime_0.13
## [25] lifecycle_1.0.4        pkgconfig_2.0.3        gson_0.1.0
## [28] Matrix_1.6-5           R6_2.6.1               fastmap_1.2.0
## [31] shiny_1.10.0           GenomeInfoDbData_1.2.11 digest_0.6.37
## [34] aplot_0.2.5            enrichplot_1.22.0      colorspace_2.1-1
## [37] patchwork_1.3.0        RSpectra_0.16-2        RSQLite_2.3.11
## [40] labeling_0.4.3         timechange_0.3.0       httr_1.4.7
## [43] polyclip_1.10-7        abind_1.4-8            compiler_4.3.2
## [46] bit64_4.6.0-1          withr_3.0.2            backports_1.5.0
## [49] BiocParallel_1.36.0    carData_3.0-5          viridis_0.6.5
## [52] DBI_1.2.3              ggforce_0.4.2          ggsignif_0.6.4
## [55] MASS_7.3-60.0.1        openssl_2.3.2          HDO.db_0.99.1
## [58] caTools_1.18.3         tools_4.3.2            scatterpie_0.2.4
## [61] ape_5.8-1              httpuv_1.6.16          glue_1.8.0
## [64] promises_1.3.2         nlme_3.1-168           GOSemSim_2.28.1
## [67] polylabelr_0.3.0       shadowtext_0.1.4       grid_4.3.2

```

## [70]	gridBase_0.4-7	fgsea_1.28.0	generics_0.1.3
## [73]	gtable_0.3.6	tzdb_0.5.0	data.table_1.17.0
## [76]	hms_1.1.3	xml2_1.3.8	tidygraph_1.3.1
## [79]	car_3.1-3	XVector_0.42.0	ggrepel_0.9.6
## [82]	pillar_1.10.2	yulab.utils_0.2.0	later_1.4.2
## [85]	splines_4.3.2	tweenr_2.0.3	treeio_1.26.0
## [88]	lattice_0.22-7	bit_4.6.0	tidyselect_1.2.1
## [91]	GO.db_3.18.0	tm_0.7-16	Biostrings_2.70.3
## [94]	knitr_1.50	gridExtra_2.3	NLP_0.3-2
## [97]	xfun_0.52	graphlayouts_1.2.2	pheatmap_1.0.12
## [100]	stringi_1.8.7	lazyeval_0.2.2	ggfun_0.1.8
## [103]	yaml_2.3.10	evaluate_1.0.3	codetools_0.2-20
## [106]	wordcloud_2.6	ggraph_2.2.1	qvalue_2.34.0
## [109]	ggplotify_0.1.2	cli_3.6.5	reticulate_1.42.0
## [112]	xtable_1.8-4	treemap_2.4-4	dichromat_2.0-0.1
## [115]	Rcpp_1.0.14	GenomeInfoDb_1.38.8	png_0.1-8
## [118]	parallel_4.3.2	blob_1.2.4	DOSE_3.28.2
## [121]	bitops_1.0-9	slam_0.1-55	viridisLite_0.4.2
## [124]	tidytrees_0.4.6	scales_1.4.0	crayon_1.5.3
## [127]	rlang_1.1.6	cowplot_1.1.3	fastmatch_1.1-6
## [130]	KEGGREST_1.42.0	formatR_1.14	