# Promoters_Galagan

## Contents


**DISTANCE FROM ALL BS'S TO TSS** **1**
PLOT FUNCTION . . . . . 1
EJEMPLO araC . . . . . 3
PLOT FOUND . . . . . 3
PLOT NOT FOUND . . . . . 4
FUNCION PARA REALIZAR LAS GRAFICAS AUTOMATICAMENTE . . . . . 4


```
## SET WORKING DIRECTORY
setwd("/Users/laura/Documents/PGC/Overlap-BS/")
```

# DISTANCE FROM ALL BS'S TO TSS

*ALL BINDING SITES ASSOCIATED WITH A TU AND AN EFFECT. ALL BS FROM REGULATORY INTERACTIONS*

FILE: Data/BindingSiteSet.txt

```
bs <- read.table("Data/BindingSiteSet.txt", header=F, sep="\t", stringsAsFactors = F)
names(bs) <- c("TF.ID", "TF.NAME", "TFBS.ID", "TFBS.LEFT", "TFBS.RIGTH", "TFBS.STRAND", "TF.GENE.ID", "
```

## PLOT FUNCTION

CONVENCIONES PARA LA GRAFICA

TRANSPARENCY LEVEL = EVIDENCE:

- NO TRANSPARENT = STRONG
- TRANSPARENT = WEAK AND NO EVIDENCE

TYPE OF LINE = EVIDENCE

- CONTINUOUS LINE = ACTIVATOR
- DASHED LINE = REPRESSOR
- DOTTED LINE = DUAL

```
plot_bs_rectangle <- function(bs.test, text.size, palette, main){
  names.prom <- bs.test$PROMOTER
  n.prom <- length(unique(names.prom))
  min.dist <- bs.test$TSS.DIST - ((bs.test$TFBS.RIGTH - bs.test$TFBS.LEFT)/2)
  max.dist <- bs.test$TSS.DIST + ((bs.test$TFBS.RIGTH - bs.test$TFBS.LEFT)/2)
  min.all <- min(min.dist, na.rm = T)
  max.all <- max(max.dist, na.rm = T)

  evidence <- bs.test$EVIDENCE
  evidence[evidence == "Strong"] <- 1
  evidence[evidence == "Weak"] <- 0.3
```

```
  evidence[evidence == "" ] <- 0
  evidence <- as.numeric(evidence)

  effect <- bs.test$EFFECT
  effect[effect == "-"] <- 5
  effect[effect == "+"] <- 1
  effect[effect == "+-" | effect == "?"] <- 3
  effect <- as.numeric(effect)

  lwd <- bs.test$found
  lwd[lwd == 1] <- 2
  lwd[lwd == 0] <- 0.5

  prom.index <- sapply(bs.test$PROMOTER, function(x,unique){ match(x, unique)}, unique = unique(names.pr

  TF <- unique(bs.test$TF.NAME)
  if(length(TF) == 1){
    col <- "royalblue"
  }else if(length(TF) == 2){
    col <-c("royalblue", "forestgreen")
  }else{
    if (palette == "rainbow" ){
      col = rainbow(length(TF))
#      col = c(rainbow(floor(length(TF)/2)), terrain.colors(ceiling(length(TF)/2)))
    }else{
      col <- brewer.pal(n = length(TF), name = palette)
    }
  }
  col.index <- sapply(bs.test$TF.NAME, function(x, TF, col){
    index <- match(x, TF)
    col[index]}, TF = TF, col = col, simplify =T)

  y1 <- prom.index - 0.3
  y2 <- prom.index + 0.3

  data <- data.frame(x1 = min.dist, x2 = max.dist, y1 = y1, y2 = y2, tf= bs.test$TF.NAME, tf.col=col.inc

  names <- unique(names.prom[order(y1)])
  {
   print(ggplot() +
    scale_x_continuous(name="TSS.DIST") +
    scale_y_continuous(name="PRONOTER", labels = c("", names), breaks = seq(0, max(y2))) +
    geom_rect(data=data, mapping=aes(xmin=x1, xmax=x2, ymin=y1, ymax=y2, fill=tf.col ), lty = effect, co
    geom_text(data=data, aes(x=x1+(x2-x1)/2, y=y1+(y2-y1)/2, label=tf), position = "identity", size=text
    geom_hline(yintercept = seq(1, n.prom), lty = 2, alpha = 0.3) +
    theme(legend.position="none") +
    ggtitle(main))
  }
}
```

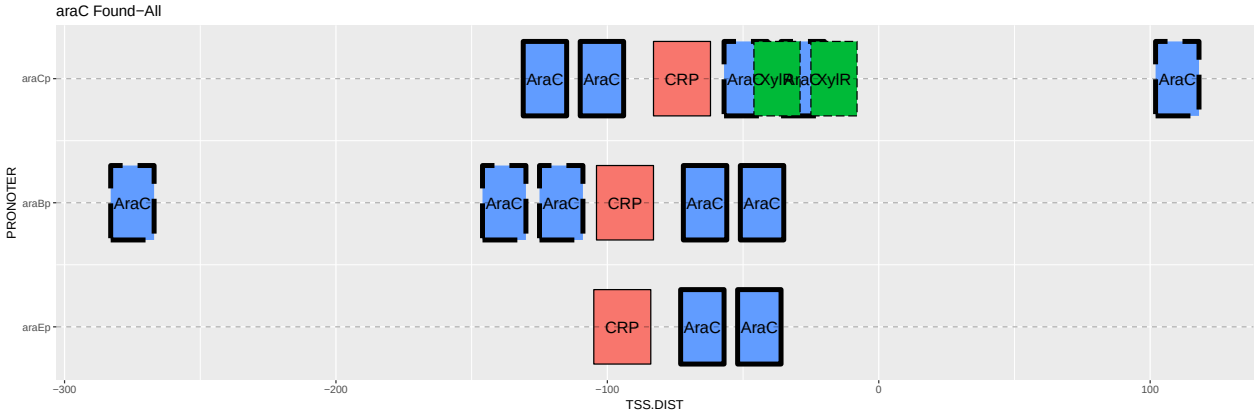
## EJEMPLO araC

```
tf <- read.table("Data/HT-BSs/araC.csv", header=F, sep=",", stringsAsFactors = F)
names(tf) <- c("found", "ID1", "TF.NAME", "ID2", "TFBS.CENTER", "TFBS.LEFT", "TFBS.RIGTH", "TFBS.STRAND
```

### PLOT FOUND

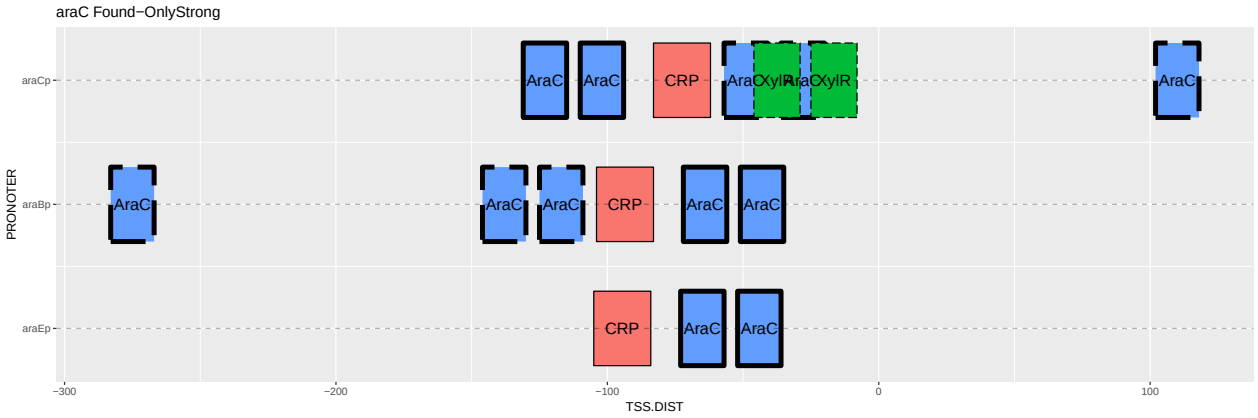
### ALL BS'S

```
tf.found <- subset(tf, found ==1)
bs.found <- subset(bs, PROMOTER %in% tf.found$PROMOTER)
bs.found <- bs.found[!(duplicated(paste(bs.found$TF.NAME, paste(bs.found$PROMOTER, bs.found$TSS.DIST)))

bs.found.lwd <- as.numeric(paste(paste(bs.found$PROMOTER, bs.found$TF.NAME), bs.found$TSS.DIST) %in% pas
bs.found$found = bs.found.lwd

plot_bs_rectangle(bs.found, 5, "Set1", main = paste("araC", "Found-All"))
```




### ONLY STRONG BS'S

```
bs.found.strong <- subset(bs.found, EVIDENCE == "Strong")
plot_bs_rectangle(bs.found.strong, 5, "Set1", main = paste("araC", "Found-OnlyStrong"))
```



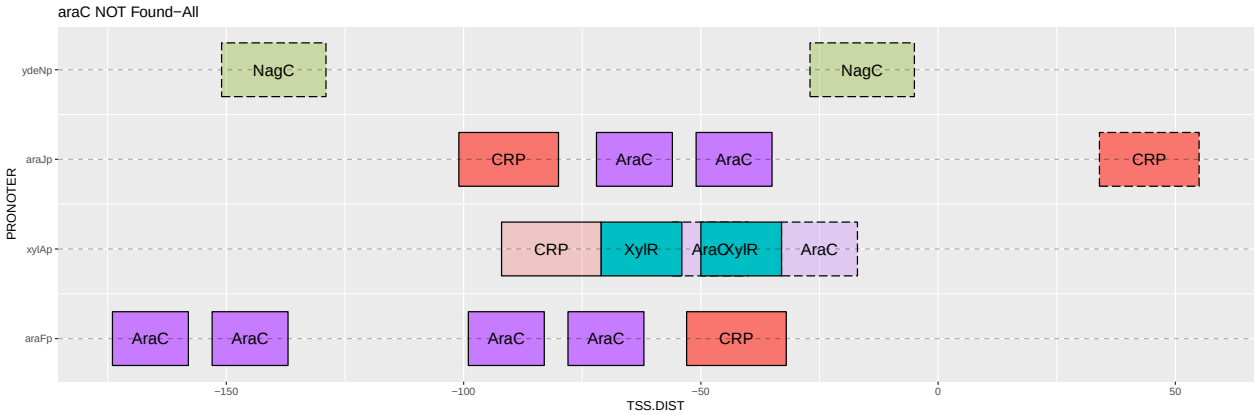
## PLOT NOT FOUND

### ALL BS'S

```
tf.NOT.found <- subset(tf, found ==0)
bs.NOT.found <- subset(bs, PROMOTER %in% tf.NOT.found$PROMOTER)
  bs.NOT.found <- bs.NOT.found[!(duplicated(paste(bs.NOT.found$TF.NAME, paste(bs.NOT.found$PROMOTER, bs

bs.NOT.found.lwd <- as.numeric(!(paste(paste(bs.NOT.found$PROMOTER, bs.NOT.found$TF.NAME), bs.NOT.found
bs.NOT.found.lwd[bs.NOT.found$TF.NAME != unique(tf$TF.NAME) ] <- 0
bs.NOT.found$found = bs.NOT.found.lwd

plot_bs_rectangle(bs.NOT.found, 5, "Set1", main = paste("araC", "NOT Found-All"))
```

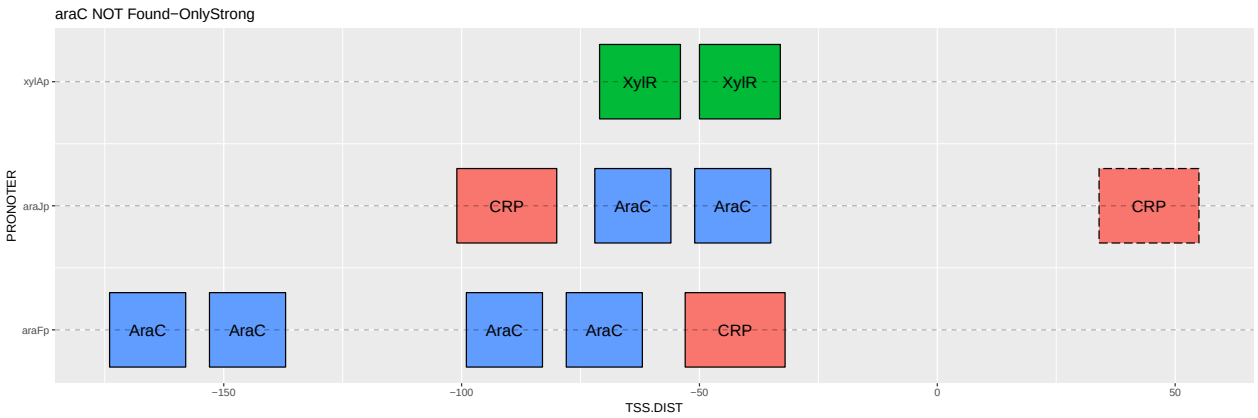
```
## Warning: Removed 1 rows containing missing values (geom_rect).
```

```
## Warning: Removed 1 rows containing missing values (geom_text).
```




### ONLY STRONG BS'S

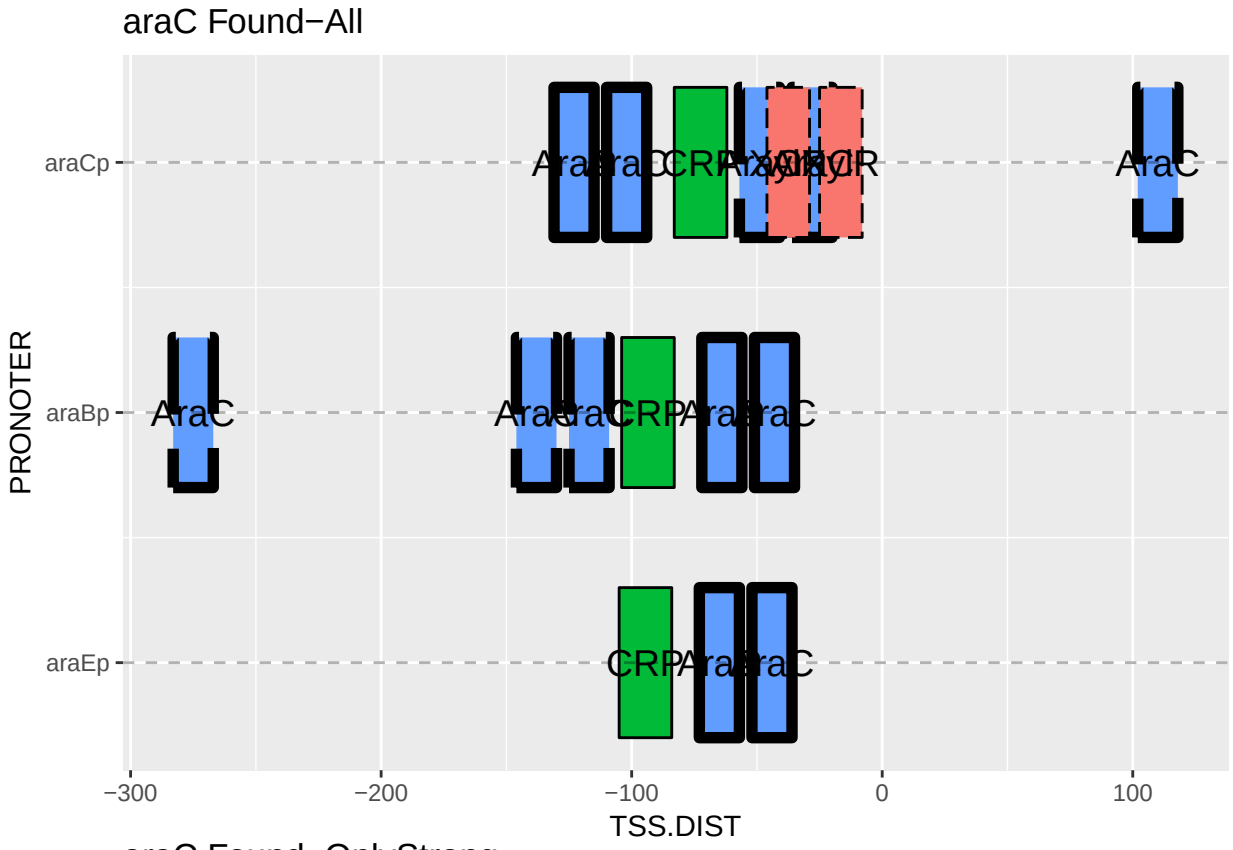
```
bs.NOT.found.strong <- subset(bs.NOT.found, EVIDENCE == "Strong")
plot_bs_rectangle(bs.NOT.found.strong, 5, "Set1", main = paste("araC", "NOT Found-OnlyStrong"))
```

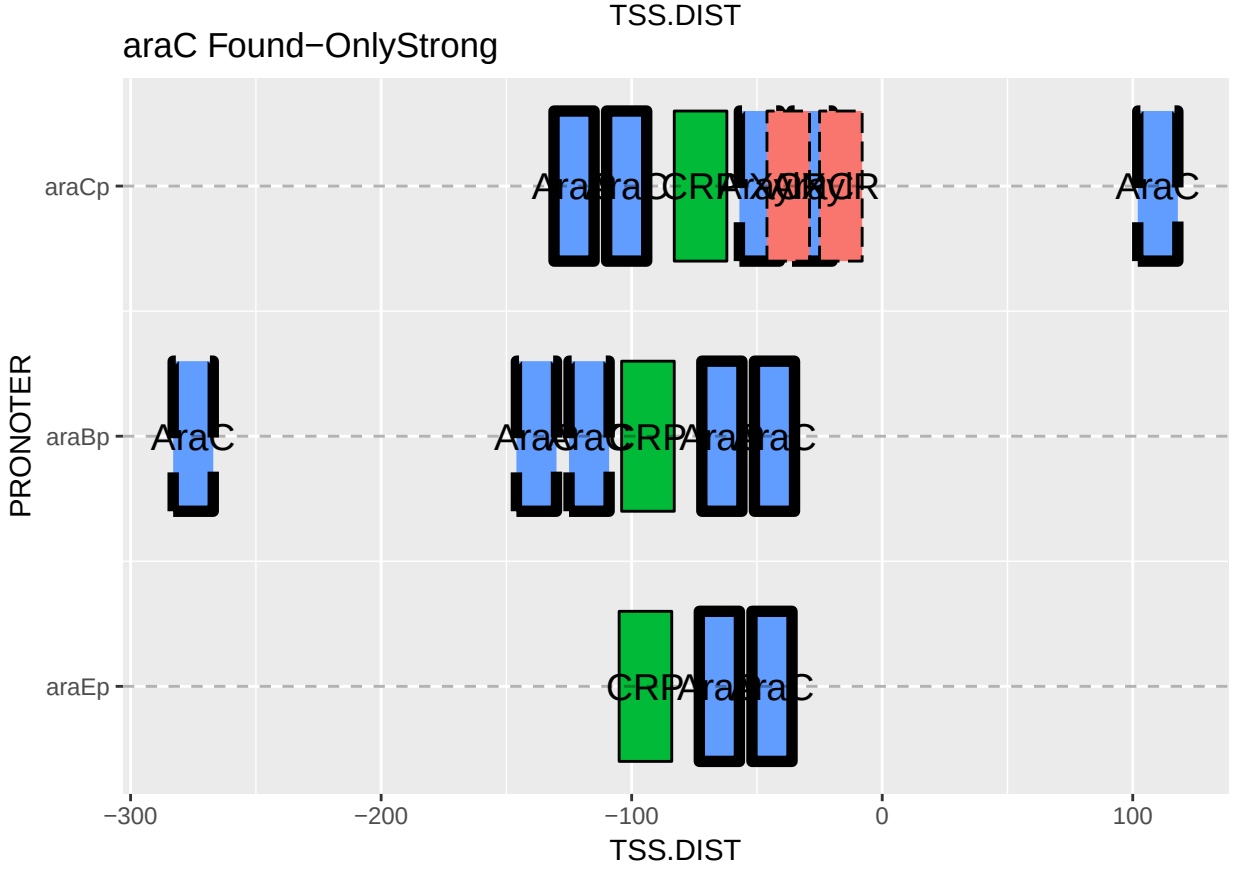



## FUNCION PARA REALIZAR LAS GRAFICAS AUTOMATICAMENTE

```
plot_perTF <- function(tf.name){
  tf <- read.table(paste(paste("Data/HT-BSs/", tf.name, sep =""), "csv", sep = "."),
                   header=F, sep=",", stringsAsFactors = F)
  names(tf) <- c("found", "ID1", "TF.NAME", "ID2", "TFBS.CENTER", "TFBS.LEFT", "TFBS.RIGTH", "TFBS.STRA

  ### PLOT FOUND

  #### ALL BS'S

  tf.found <- subset(tf, found ==1)
  bs.found <- subset(bs, PROMOTER %in% tf.found$PROMOTER)
  bs.found <- bs.found[!(duplicated(paste(bs.found$TF.NAME, paste(bs.found$PROMOTER, bs.found$TSS.DIST)

  bs.found.lwd <- as.numeric(paste(paste(bs.found$PROMOTER, bs.found$TF.NAME), bs.found$TSS.DIST)
                             %in% paste(paste(tf.found$PROMOTER, tf.found$TF.NAME), tf.found$TSS.DIST))
  bs.found$found = bs.found.lwd

  plot_bs_rectangle(bs.found, 5, "rainbow", main = paste(tf.name, "Found-All"))

  #### ONLY STRONG BS'S
  bs.found.strong <- subset(bs.found, EVIDENCE == "Strong")
  plot_bs_rectangle(bs.found.strong, 5, "rainbow", main = paste(tf.name, "Found-OnlyStrong"))

  ### PLOT NOT FOUND

  #### ALL BS'S

  tf.NOT.found <- subset(tf, found ==0)
  bs.NOT.found <- subset(bs, PROMOTER %in% tf.NOT.found$PROMOTER)
  bs.NOT.found <- bs.NOT.found[!(duplicated(paste(bs.NOT.found$TF.NAME, paste(bs.NOT.found$PROMOTER, bs

  bs.NOT.found.lwd <- as.numeric(!(paste(paste(bs.NOT.found$PROMOTER, bs.NOT.found$TF.NAME), bs.NOT.fou
                                   %in% paste(paste(tf.NOT.found$PROMOTER, tf.NOT.found$TF.NAME), tf.NO
  bs.NOT.found.lwd[bs.NOT.found$TF.NAME != unique(tf$TF.NAME) ] <- 0
  bs.NOT.found$found = bs.NOT.found.lwd

  plot_bs_rectangle(bs.NOT.found, 5, "rainbow", main = paste(tf.name, "NOT Found-All"))

  #### ONLY STRONG BS'S

  bs.NOT.found.strong <- subset(bs.NOT.found, EVIDENCE == "Strong")
  plot_bs_rectangle(bs.NOT.found.strong, 5, "rainbow", main = paste(tf.name, "NOT Found-OnlyStrong"))
}
```

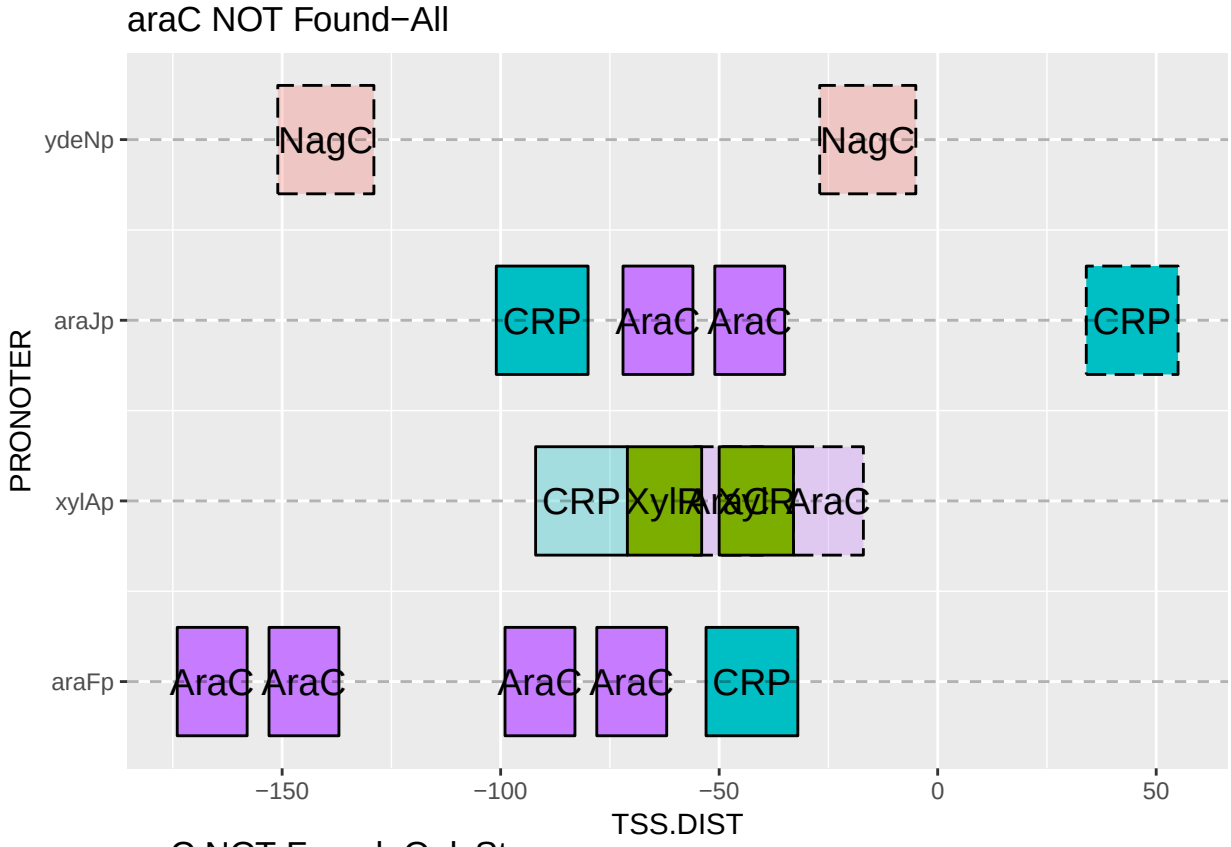
```
tf.list <- c("araC", "narP", "nsrR", "uxuR")

lapply(tf.list, plot_perTF)
```

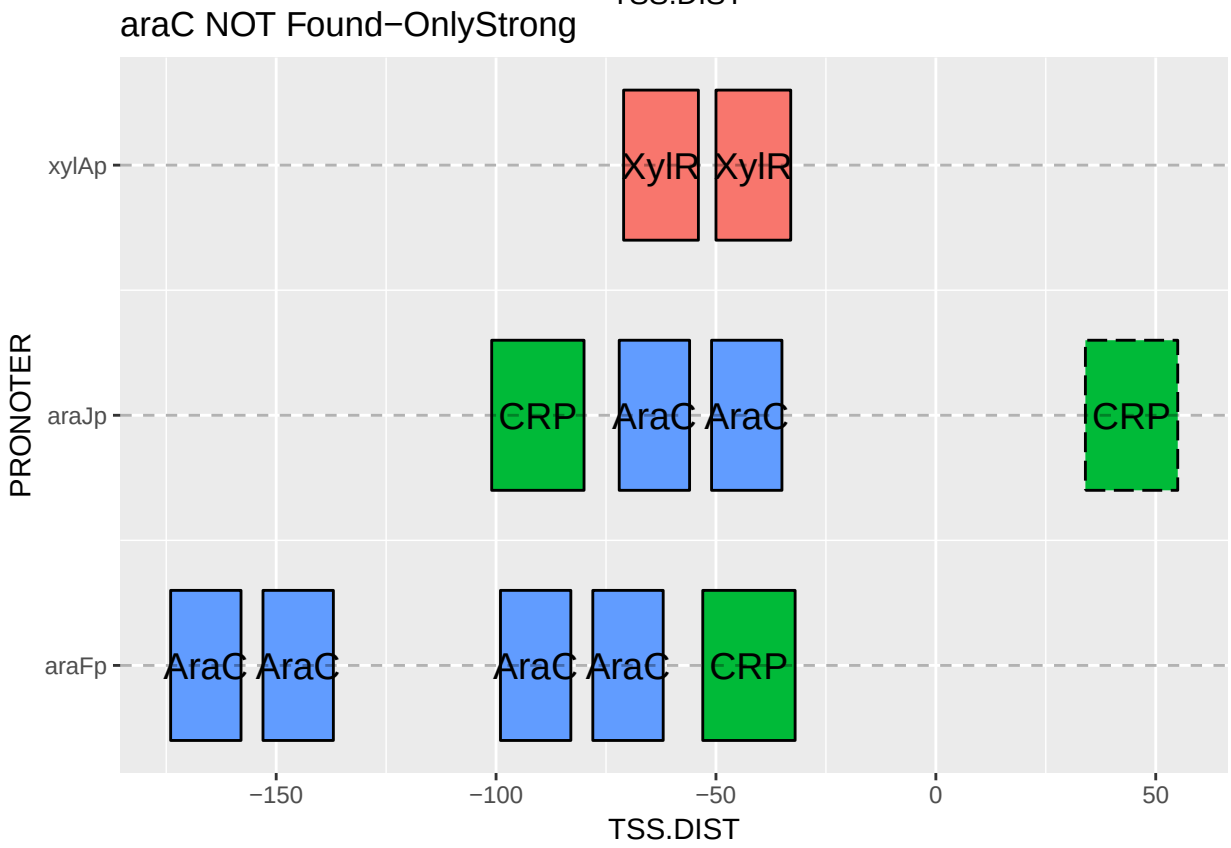


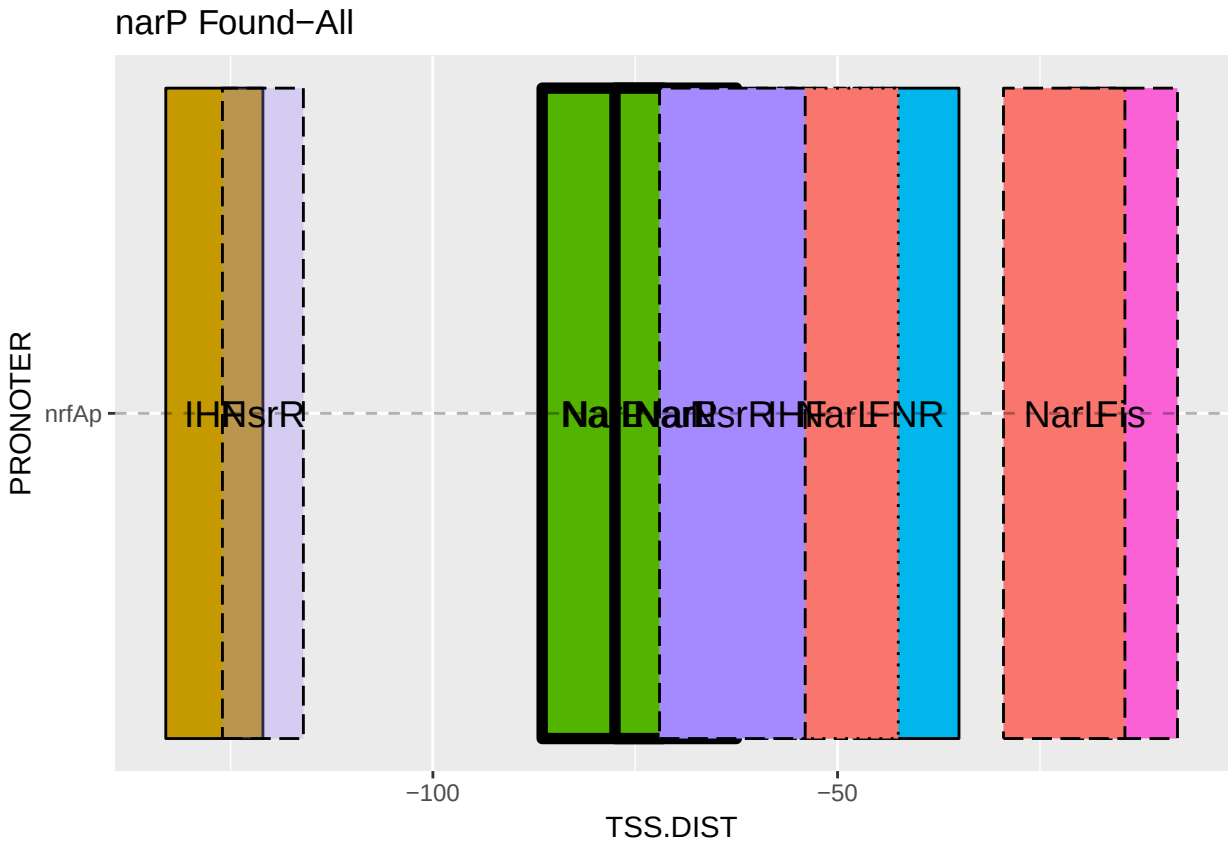

```
## Warning: Removed 1 rows containing missing values (geom_rect).
```

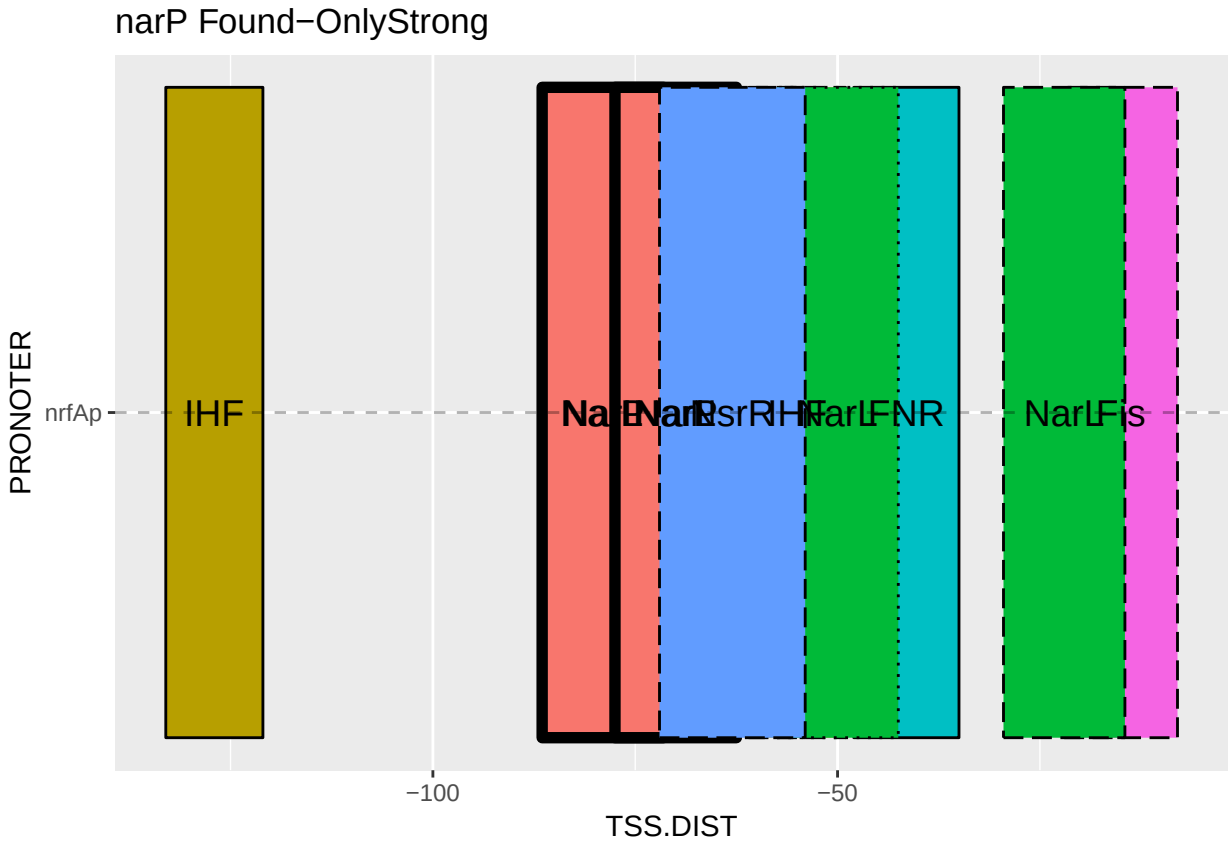
```
## Warning: Removed 1 rows containing missing values (geom_text).
```



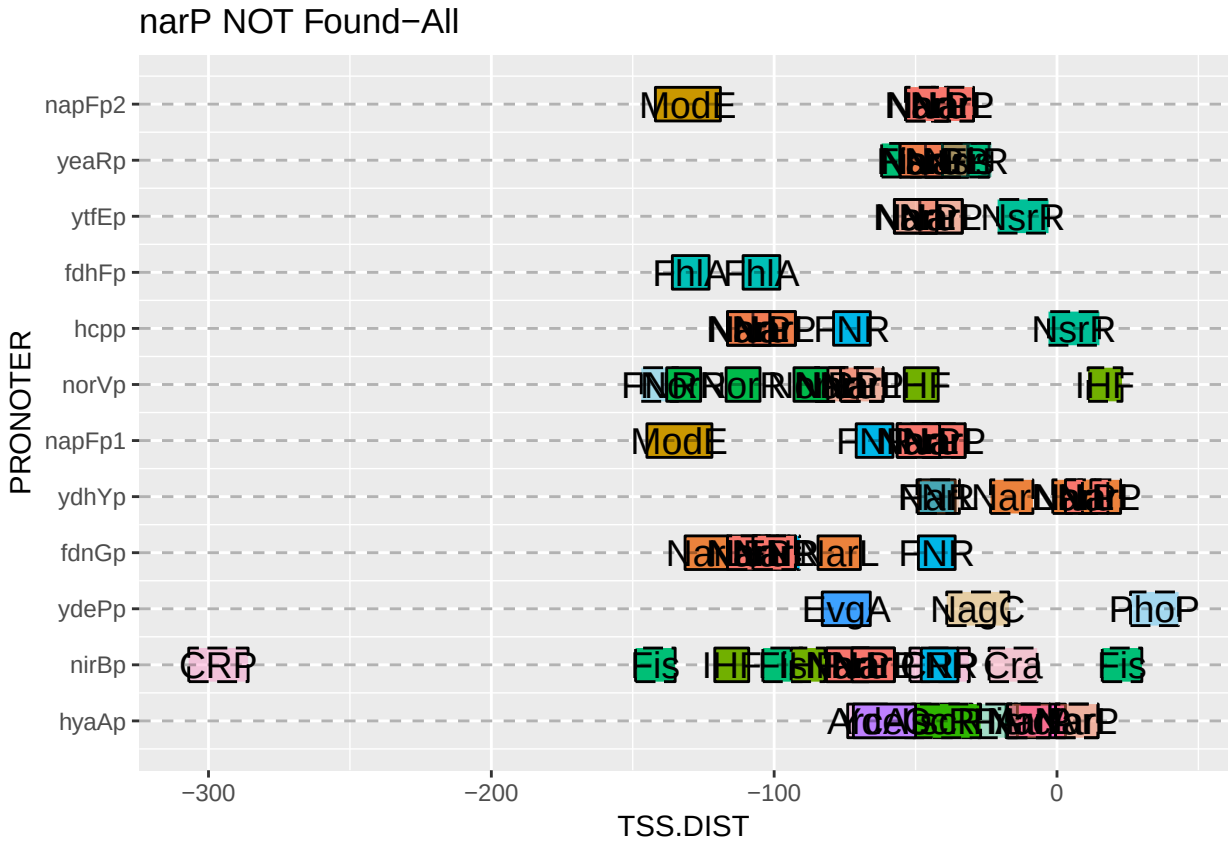



```
## Warning: Removed 1 rows containing missing values (geom_rect).

## Warning: Removed 1 rows containing missing values (geom_text).
```



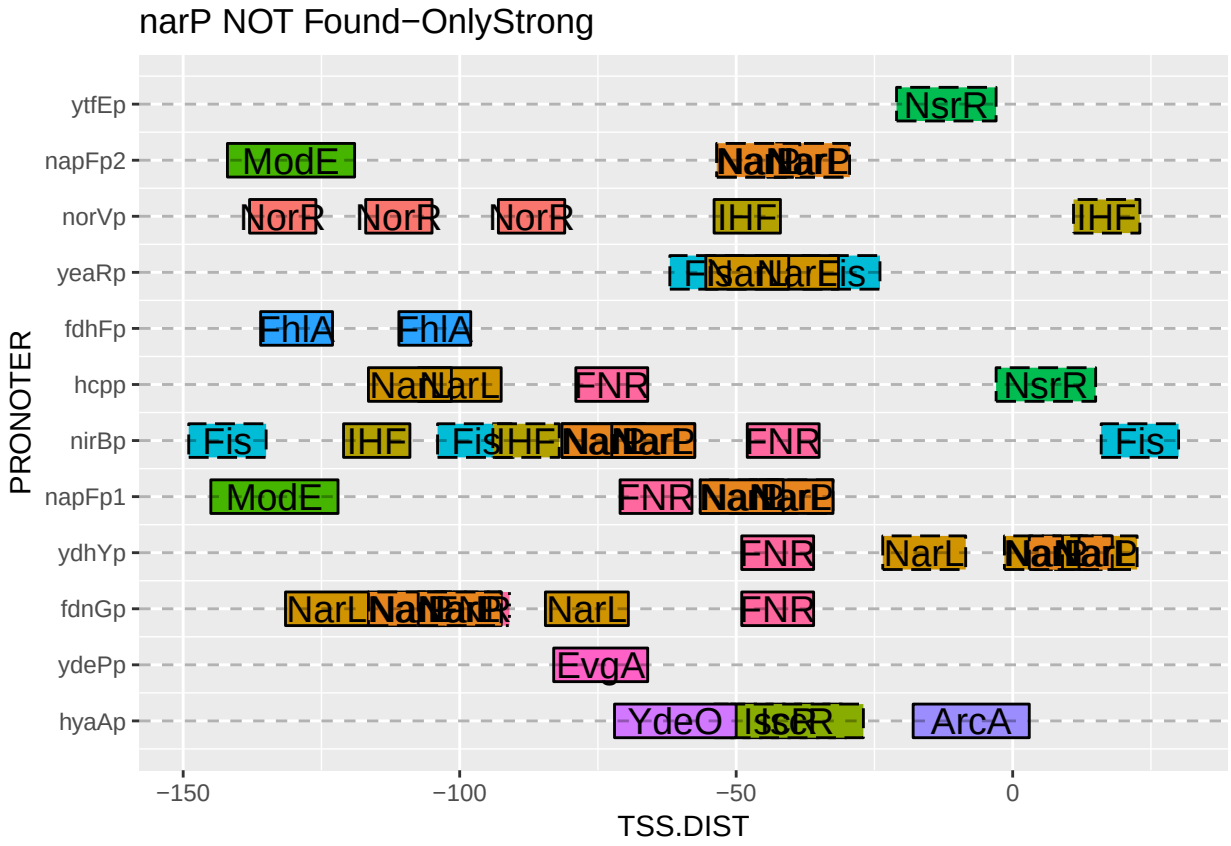



```
## Warning: Removed 14 rows containing missing values (geom_rect).
## Warning: Removed 14 rows containing missing values (geom_text).
```



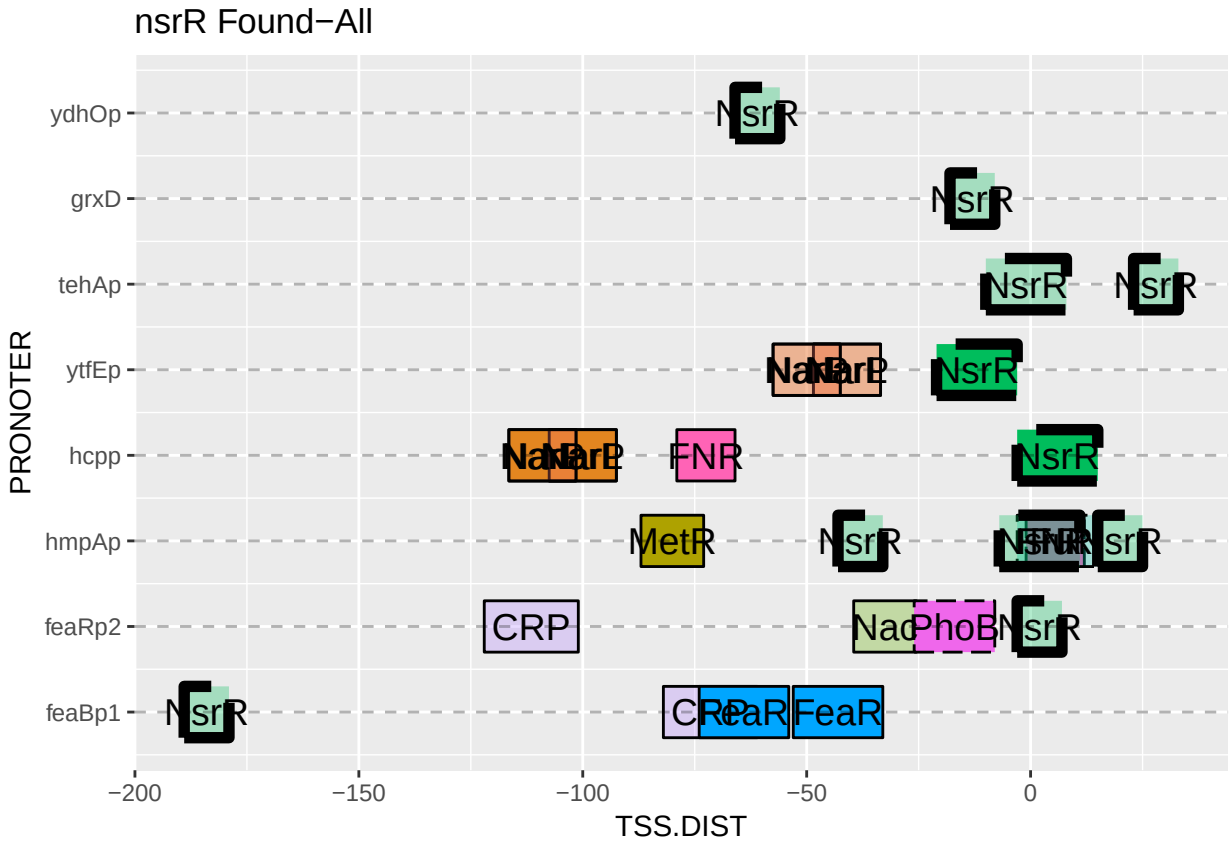

```
## Warning: Removed 8 rows containing missing values (geom_rect).
```

```
## Warning: Removed 8 rows containing missing values (geom_text).
```



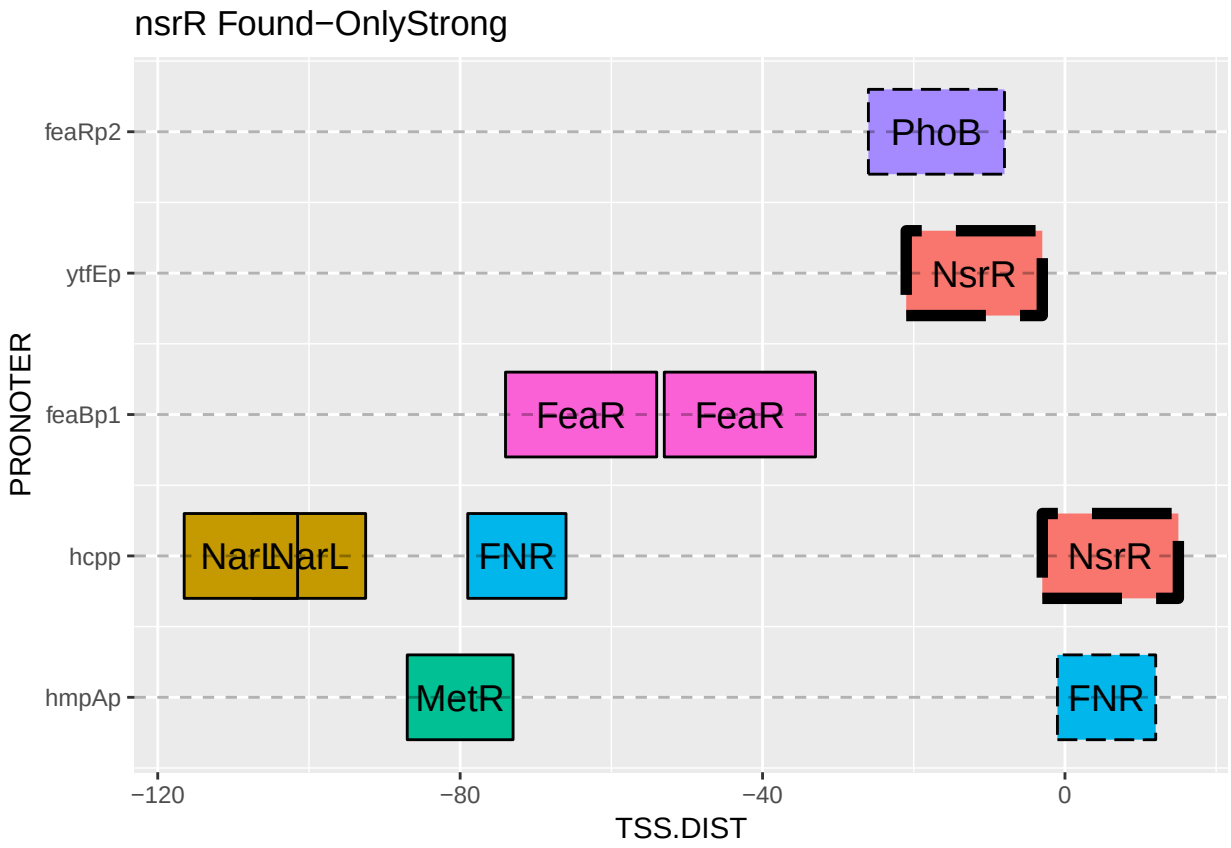

```
## Warning: Removed 2 rows containing missing values (geom_rect).
```

```
## Warning: Removed 2 rows containing missing values (geom_text).
```

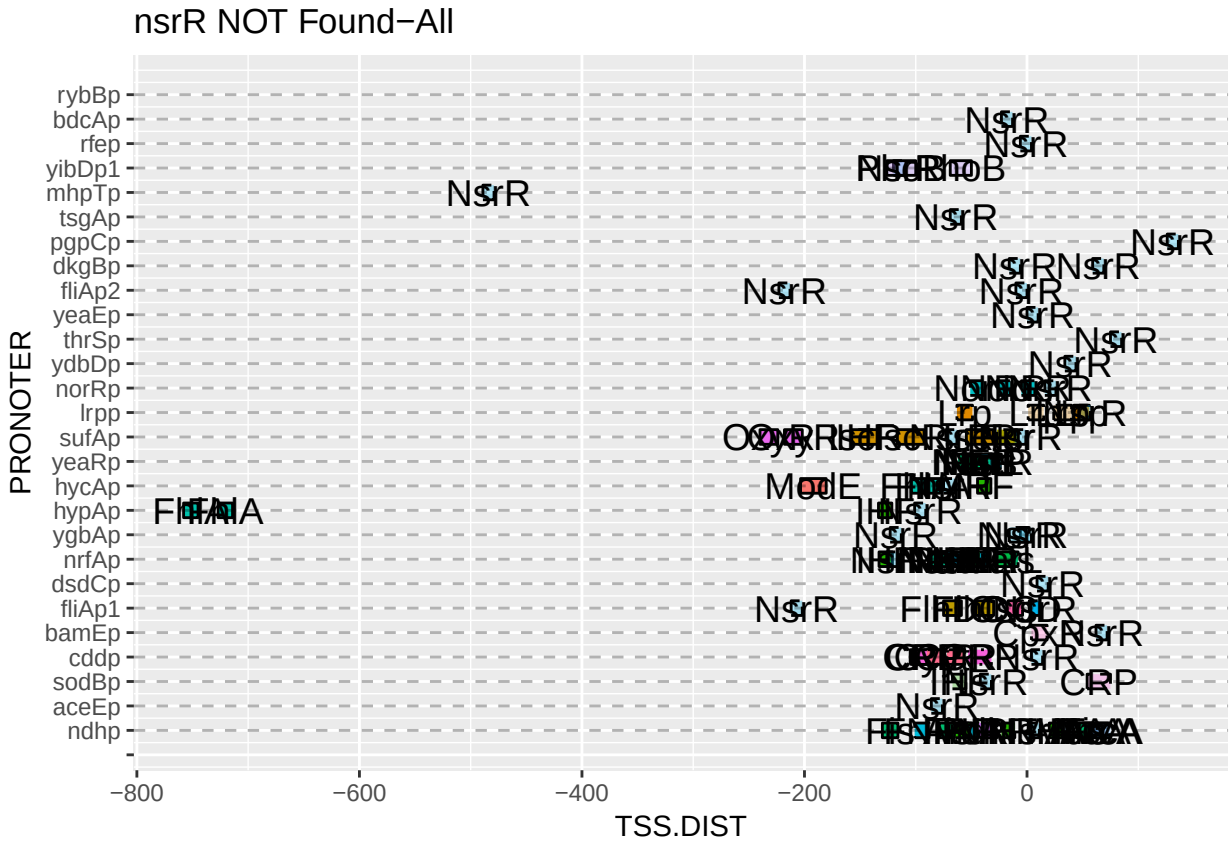



```
## Warning: Removed 1 rows containing missing values (geom_rect).
```

```
## Warning: Removed 1 rows containing missing values (geom_text).
```

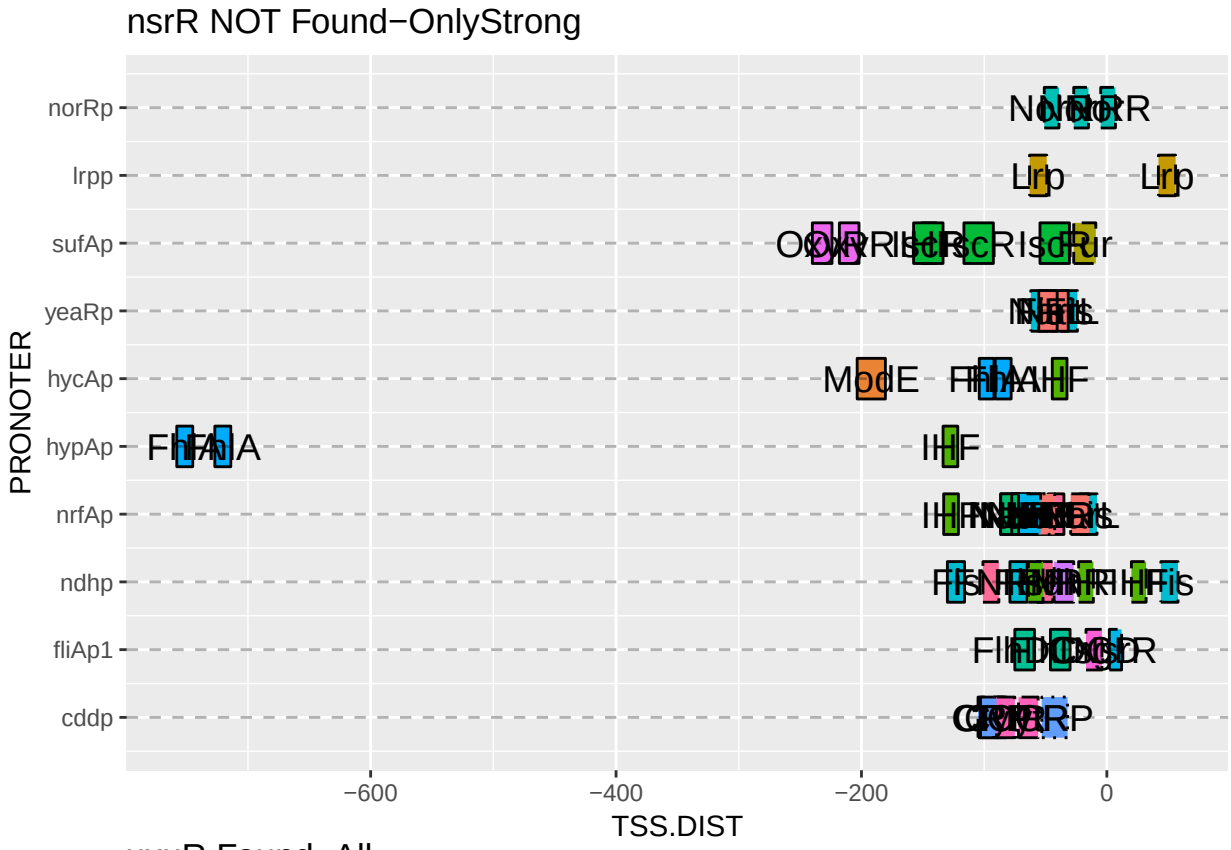


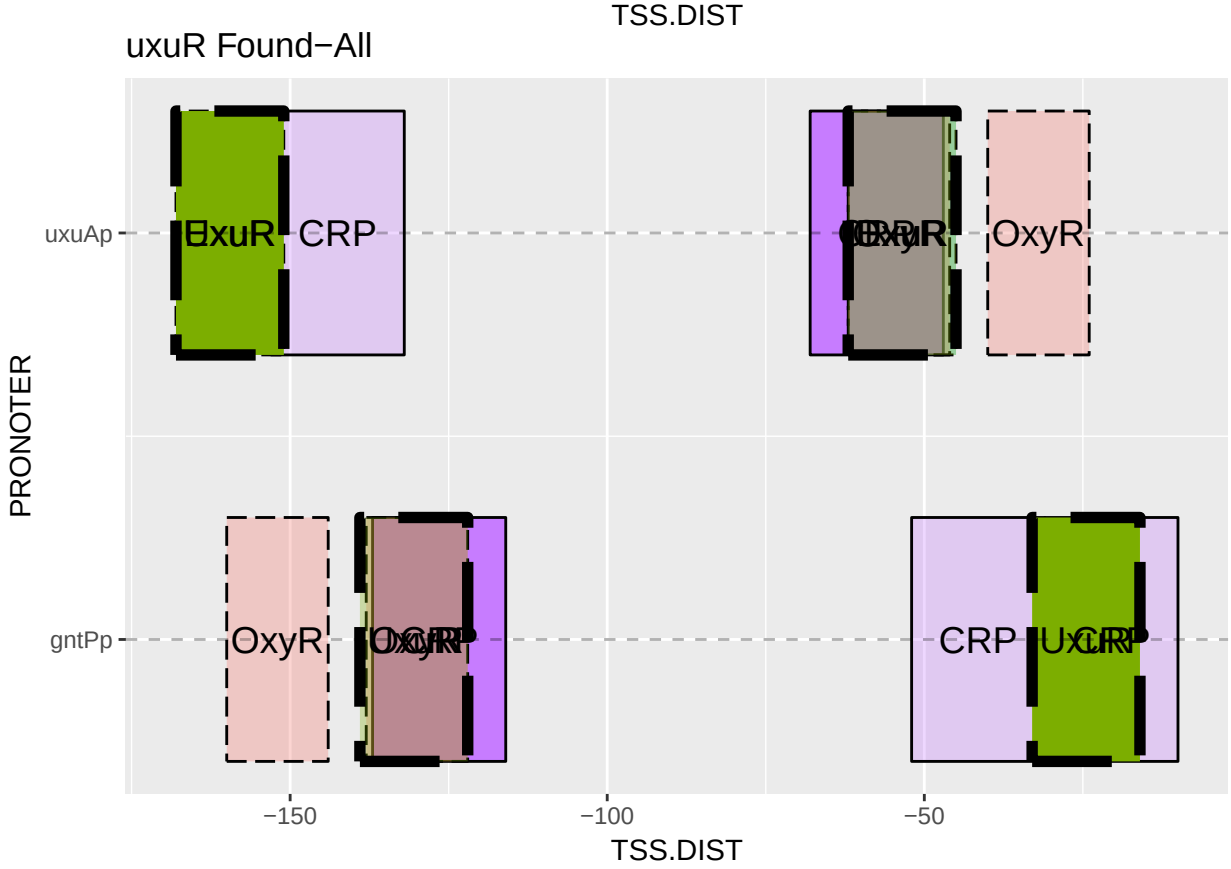

```
## Warning: Removed 12 rows containing missing values (geom_rect).
## Warning: Removed 12 rows containing missing values (geom_text).
```

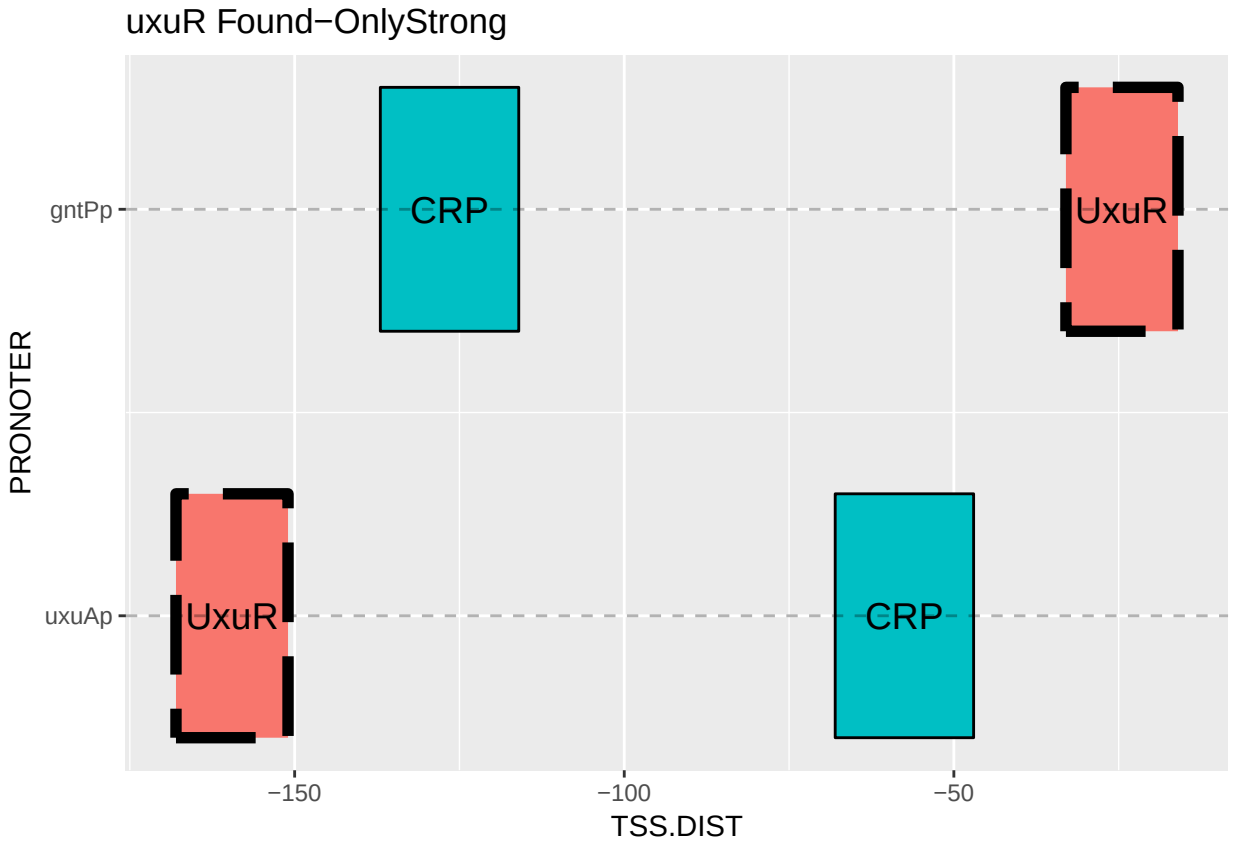



```
## Warning: Removed 3 rows containing missing values (geom_rect).
```

```
## Warning: Removed 3 rows containing missing values (geom_text).
```

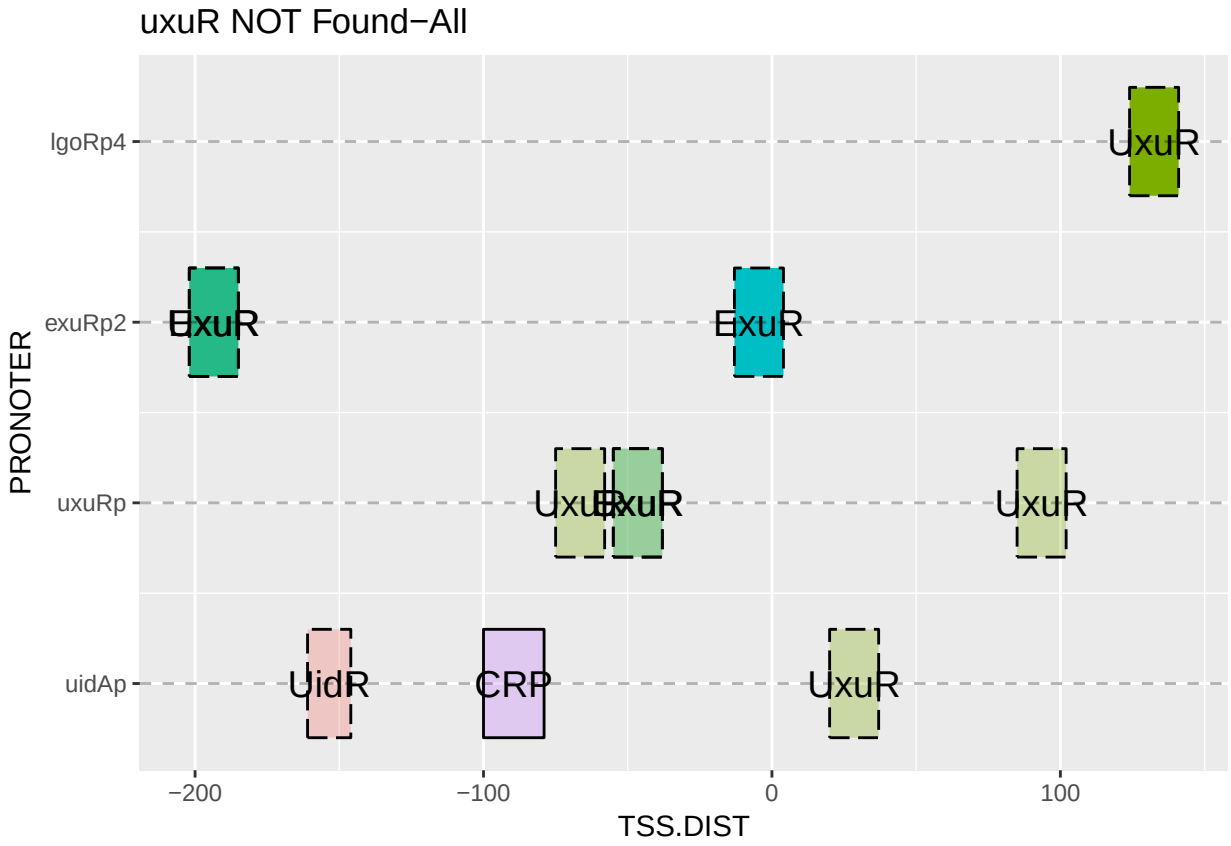

nsrR NOT Found−OnlyStrong
norRp
lrpp
sufAp
yeaRp
hycAp
hypAp
nrfAp
ndhp
fliAp1
cddp
PRONOTER
−600
−400
−200
0
TSS.DIST


uxuR Found−All
uxuAp
gntPp
PRONOTER
UxuR
CRP
OxyR
−150
−100
−50
TSS.DIST



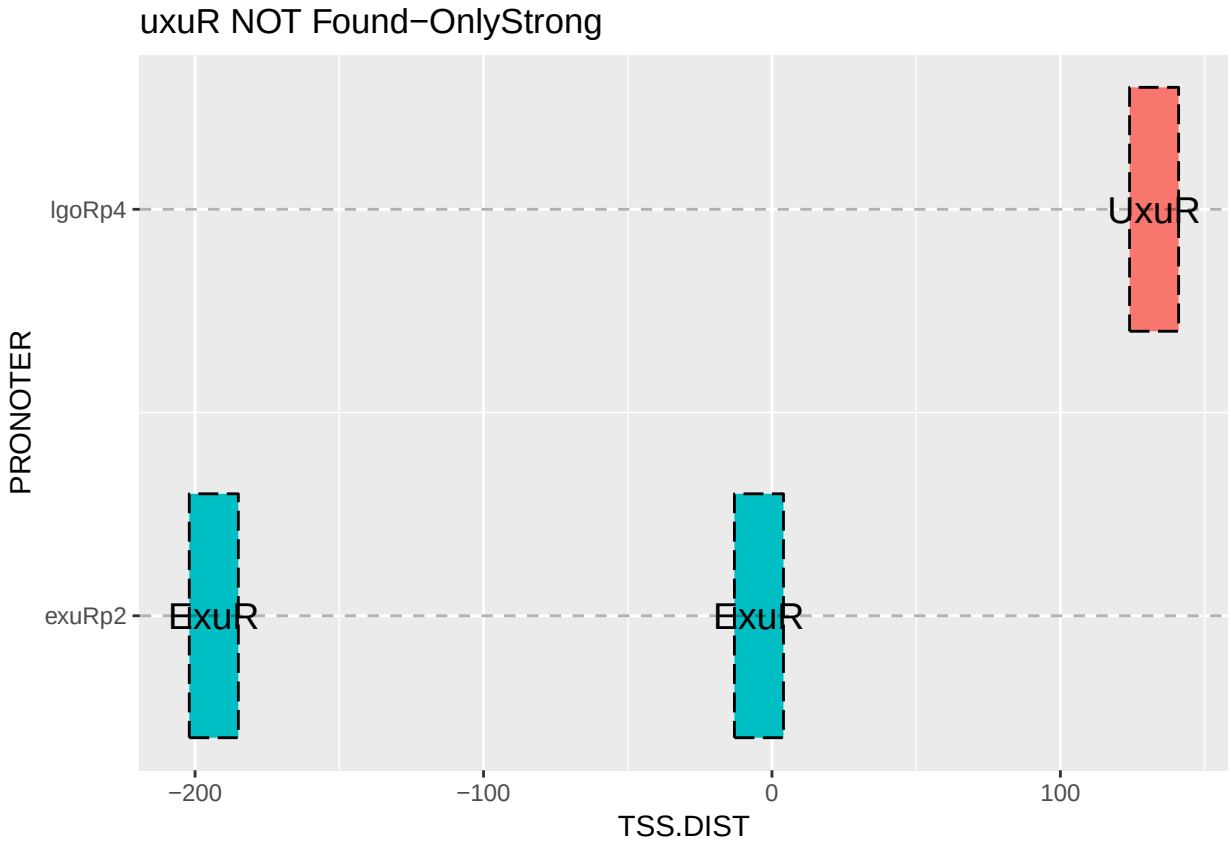

```
## Warning: Removed 2 rows containing missing values (geom_rect).
```

```
## Warning: Removed 2 rows containing missing values (geom_text).
```

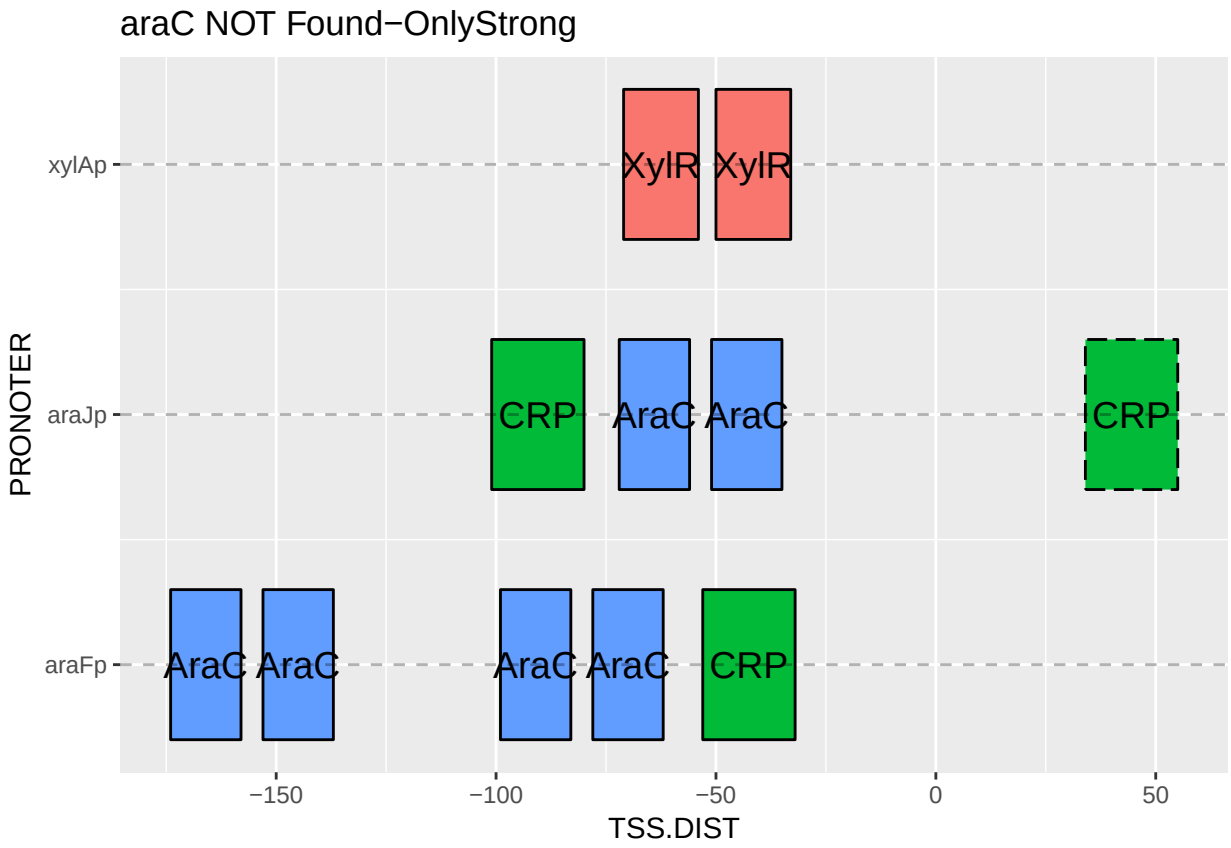



```
## Warning: Removed 1 rows containing missing values (geom_rect).
```

```
## Warning: Removed 1 rows containing missing values (geom_text).
```


uxuR NOT Found−OnlyStrong
lgoRp4
exuRp2
PRONOTER
UxuR
ExuR
ExuR
−200
−100
0
100
TSS.DIST

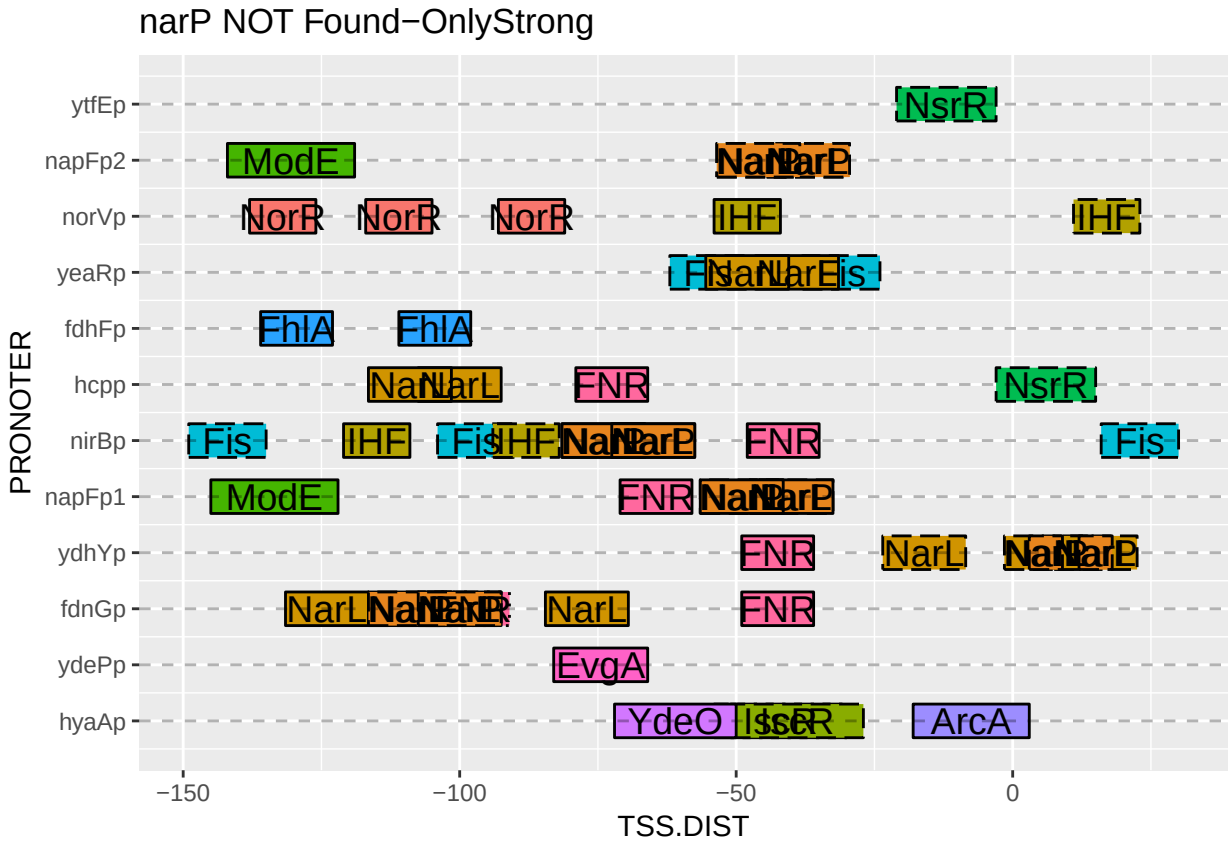
```
## [[1]]
```




```
##
## [[2]]
```

```
## Warning: Removed 8 rows containing missing values (geom_rect).
```

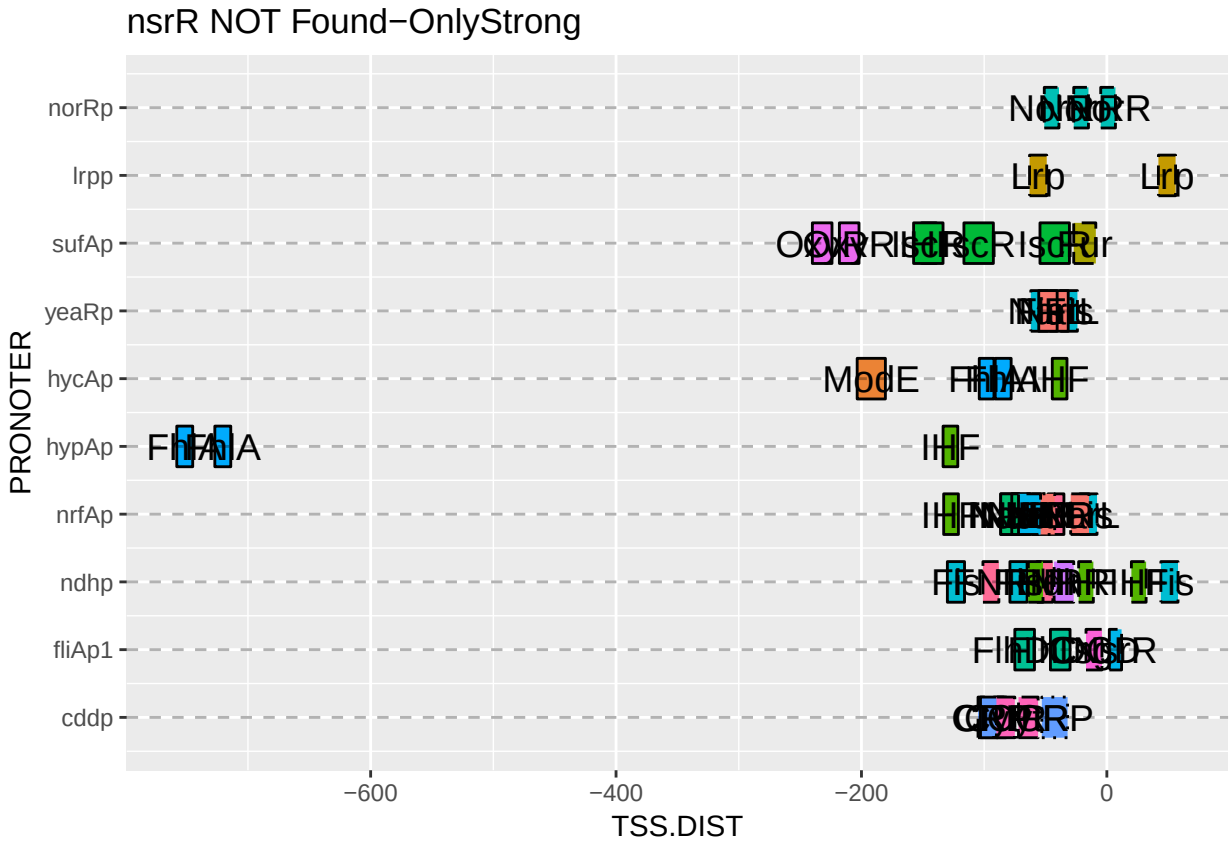
```
## Warning: Removed 8 rows containing missing values (geom_text).
```




```
##
## [[3]]
```

```
## Warning: Removed 3 rows containing missing values (geom_rect).
```

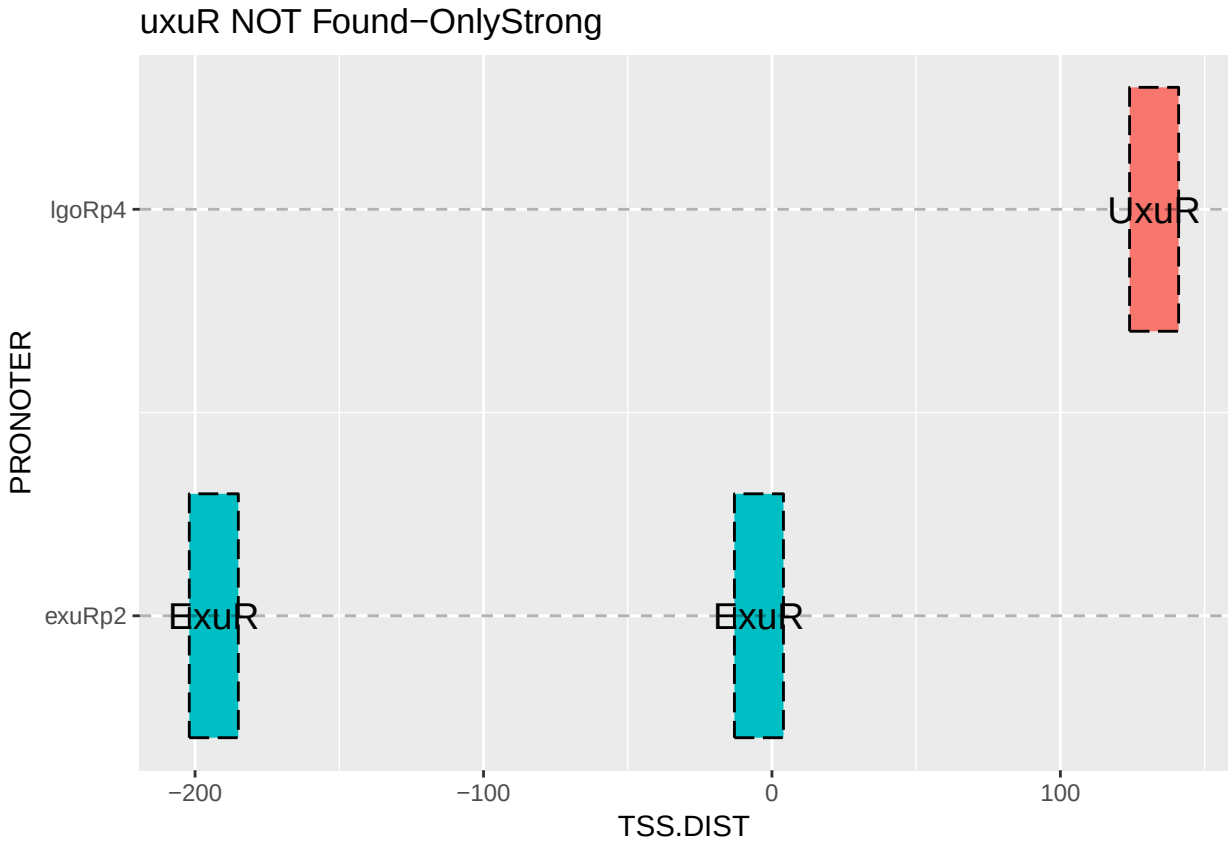
```
## Warning: Removed 3 rows containing missing values (geom_text).
```




```
##
## [[4]]
```

```
## Warning: Removed 1 rows containing missing values (geom_rect).
```

```
## Warning: Removed 1 rows containing missing values (geom_text).
```


uxuR NOT Found−OnlyStrong
lgoRp4
exuRp2
PRONOTER
UxuR
ExuR
ExuR
−200
−100
0
100
TSS.DIST