Assign mir181 binding sites to a specific transcript

Melina Klostermann

18 March, 2024

Contents

1 Libraries and settings

```
# ----------------------------------------
# libraries
# ----------------------------------------
library(tidyverse)
library(GenomicRanges)
library(GenomicFeatures)
library(Biostrings)


# ----------------------------------------
# settings
# ----------------------------------------

here <- here::here()

out <- paste0(here,"/Figure4/03_assign_transcripts/")
```

2 What was done?

- Each binding site is assigned to a specific transcript isoform
- The following criteria are used to decide on the isoform:

1) Previously assigned region (see binding site definitions), if more then one transcript is possible:

2) if possible protein coding transcripts, if still more then one transcript is possible:

3) best transcript support level, if still more then one transcript is possible:

4) longest transcript

- Then the mir181 binding sites are mapped to their respective transcript coordinates.
- The transcript annotations are later used for motiv discovery and structure predictions.

```
#-------------------------------------
# Files
#-------------------------------------
anno <- readRDS(paste0(here,"/Supporting_scripts/annotation_preprocessing/annotation.rds"))
anno$gene_id <- sub("\\..*", "", anno$gene_id)
anno$transcript_id <- sub("\\..*", "", anno$transcript_id)

mir181_bs <- readRDS(paste0(here,"/Figure4/01_MRE_bound_gene_and_bound_region/mir181_bs_afterFigure4B.r
```

3 Assign each binding site to a specific transcript

```
# get appris transcripts (when there are multiple take the longest)
transcripts <- anno[anno$type=="transcript"] %>% as.data.frame(.)
#transcripts$appris <- grepl(transcripts$tag, pattern= "appris_principal_1")


# get regional annotations
three_utr <- anno[anno$type=="three_prime_UTR"] %>% as.data.frame(.) %>%
  dplyr::select(seqnames, start, end, width, strand, transcript_id, gene_id) %>%
  left_join(transcripts %>% dplyr::select(., width, transcript_id, transcript_type, transcript_support_l
  makeGRangesFromDataFrame(., keep.extra.columns = T)

cds <- anno[anno$type=="CDS"] %>% as.data.frame(.) %>%
  dplyr::select(seqnames, start, end, width, strand, transcript_id, gene_id) %>%
  left_join(transcripts %>% dplyr::select(., width, transcript_id, transcript_type, transcript_support_l
  makeGRangesFromDataFrame(., keep.extra.columns = T)

five_utr <- anno[anno$type=="five_prime_UTR"] %>% as.data.frame(.) %>%
  dplyr::select(seqnames, start, end, width, strand, transcript_id, gene_id) %>%
  left_join(transcripts %>% dplyr::select(., width, transcript_id, transcript_type, transcript_support_l
  makeGRangesFromDataFrame(., keep.extra.columns = T)


# select best transcript per bs
# 3'utr
bs_utr3 <- mir181_bs %>% subset(., region == "utr3") %>%
  makeGRangesFromDataFrame(., keep.extra.columns = T)
NROW(bs_utr3)
```

```
## [1] 5607
```

```
i_utr3 <- findOverlaps(bs_utr3, three_utr, type = "any")
t <- as.data.frame(three_utr[subjectHits(i_utr3)]) %>% dplyr::select(width_tx, transcript_type, transcri
t$transcript_type <- factor(t$transcript_type, levels = c("protein_coding"))

bs_utr3 <-  bs_utr3[queryHits(i_utr3)] %>%
```

```
  as.tibble()


bs_utr3_possibilities <- cbind(bs_utr3, t) %>%
  group_by(mir181BS_ID, transcript_type, transcript_support_level, .by_group = T) %>%
  summarize(pos = length(start)) %>%
  group_by(mir181BS_ID) %>%
  dplyr::slice(1)


bs_utr3 <- cbind(bs_utr3, t) %>%
  group_by(mir181BS_ID) %>%
  arrange(transcript_type, transcript_support_level, .by_group = T) %>%
  dplyr::slice(1)
nrow(bs_utr3)
```

```
## [1] 5607
```

```
# cds
bs_cds <- mir181_bs %>% subset(., region == "cds") %>%
  makeGRangesFromDataFrame(., keep.extra.columns = T)
NROW(bs_cds)
```

```
## [1] 4284
```

```
i_cds <- findOverlaps(bs_cds, cds, type = "any")
t <- as.data.frame(cds[subjectHits(i_cds)]) %>% dplyr::select(width_tx, transcript_type, transcript_sup

bs_cds <-  bs_cds[queryHits(i_cds)] %>%
  as.tibble()

bs_cds_possibilities <- cbind(bs_cds, t) %>%
  group_by(mir181BS_ID, transcript_type, transcript_support_level, .by_group = T) %>%
  summarize(pos = length(start)) %>%
  group_by(mir181BS_ID) %>%
  dplyr::slice(1)

bs_cds <- cbind(bs_cds, t) %>%
  group_by(mir181BS_ID) %>%
  arrange(transcript_type, transcript_support_level, .by_group = T) %>%
  dplyr::slice(1)
nrow(bs_cds)
```

```
## [1] 4284
```

```
# 5'utr
bs_utr5 <- mir181_bs %>% subset(., region == "utr5") %>%
  makeGRangesFromDataFrame(., keep.extra.columns = T)
NROW(bs_utr5)
```

```
## [1] 582
```

```
i_utr5 <- findOverlaps( bs_utr5, five_utr, type = "any")
t <- as.data.frame(five_utr[subjectHits(i_utr5)]) %>% dplyr::select(width_tx, transcript_type, transcri

bs_utr5 <-  bs_utr5[queryHits(i_utr5)] %>%
  as.tibble()
```

```
bs_utr5_possibilities <- cbind(bs_utr5, t) %>%
  group_by(mir181BS_ID, transcript_type, transcript_support_level, .by_group = T) %>%
  summarize(pos = length(start)) %>%
  group_by(mir181BS_ID) %>%
  dplyr::slice(1)

bs_utr5 <- cbind(bs_utr5, t) %>%
  group_by(mir181BS_ID) %>%
  arrange(transcript_type, transcript_support_level, .by_group = T) %>%
  dplyr::slice(1)
nrow(bs_utr5)
```

```
## [1] 582
```

```
mir181_bs <- rbind(bs_utr3, bs_cds, bs_utr5)
```

4 How many can not be unambigously assigned?

```
possibilities <- rbind(bs_utr3_possibilities,
                       bs_cds_possibilities,
                       bs_utr5_possibilities)

(possibilities %>% subset(pos > 1) %>% nrow()) / nrow(possibilities)
```

```
## [1] 0.3286546
```

5 Get transcript coordinates

```
############################
# BS sequence considering mature transcripts
############################
# prepare a txdb of expressed transcripts
anno_transcripts_exons <- anno[anno$type != "gene"]
anno_transcripts_exons$transcript_id <- sub("\\..*", "", anno_transcripts_exons$transcript_id)
anno_transcripts_GR_list <- anno_transcripts_exons %>%
  splitAsList(., f = .$transcript_id) %>%
  GRangesList(.)

txdb <- makeTxDbFromGRanges(unlist(anno_transcripts_GR_list))

# prepare a transcript mapper (contains transcript ids and names together with genomic positions of tra
transcripts_txdb_mapper <- transcripts(txdb)

# get transcript-relative coordinates of BS
mir181_bs <- makeGRangesFromDataFrame(mir181_bs, keep.extra.columns = T)
mir181_bs_tx <- mapToTranscripts(mir181_bs, txdb, extractor.fun = GenomicFeatures::exonsBy, ignore.stra
# Mapped position is computed by counting from the transcription start site (TSS) and is not affected b

# readd metadata
gen_pos <- mir181_bs[mir181_bs_tx$xHits] %>%
  as.data.frame() %>%
  dplyr::select(start, end, strand, seqnames) %>%
```

```
  rename( start = "genomic_start" , end = "genomic_end", strand = "genomic_strand", seqnames = "genomic_

elementMetadata(mir181_bs_tx) <- c(elementMetadata(mir181_bs_tx), elementMetadata(mir181_bs[mir181_bs_t

# change the seqnames to the transcript names
names(mir181_bs_tx) <- 1: NROW(mir181_bs_tx)
mir181_bs_tx <- as.data.frame(mir181_bs_tx)
mir181_bs_tx$seqnames <- transcripts_txdb_mapper$tx_name[mir181_bs_tx$transcriptsHits]


mir181_bs_tx <- mir181_bs_tx %>% subset(seqnames == transcript_id)
```

- Number of mirBS: 10473
- Number of mirBS on transcripts (without intron): 10301
- Number of enriched mirBS: 4658
- Number of enriched mirBS on transcripts: 4552

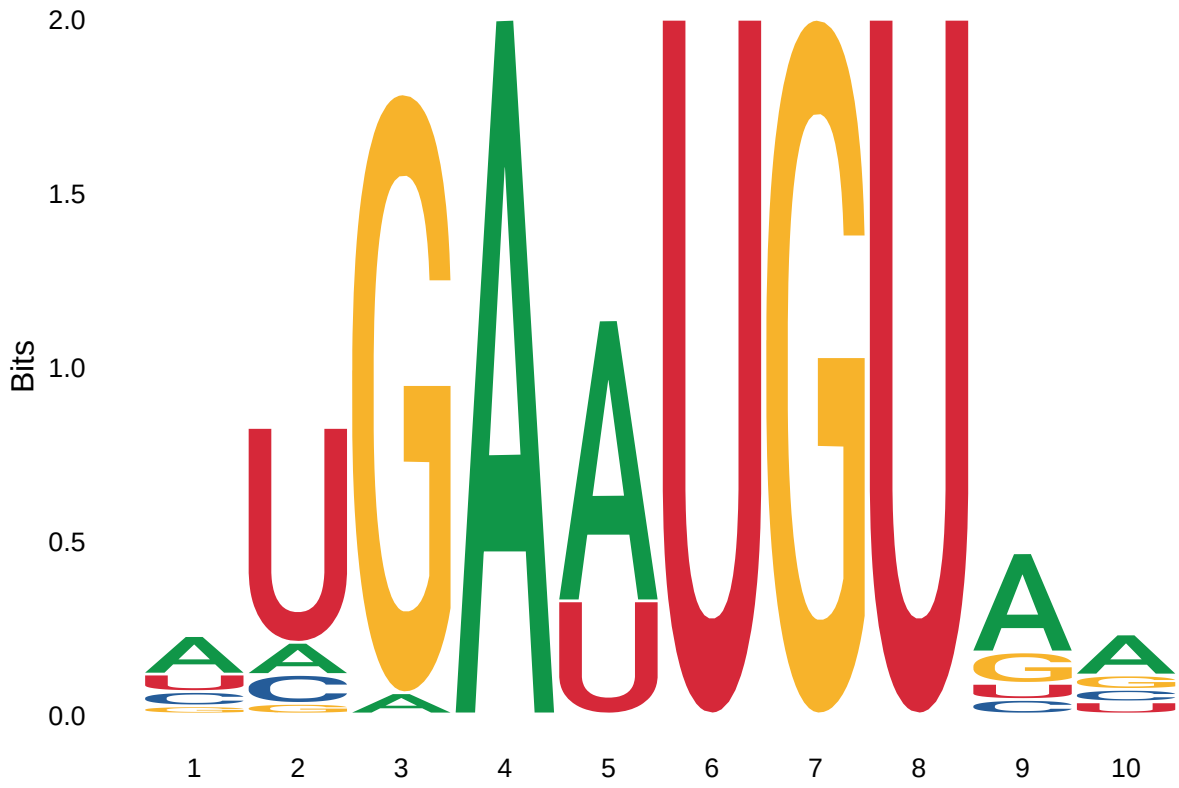
Comment: we use findOverlaps with the center position of the binding site to assign the regions in the binding site definition scripts. The mapToTranscripts will only keep binding sites, that are completely inside the transcript. This is why we loose a few binding sites, when mapping to transcripts here.

6 Do XSTREME motif search with other transcript assignment

```
mir181_bs <- makeGRangesFromDataFrame(mir181_bs_tx, keep.extra.columns = T) %>%
  shift(., 200) %>%
  as.data.frame(.)

mir181_enriched_set <- mir181_bs %>%
  subset(set %in% c("ago_bs_mir181_chi&mir181_enriched", "mir181_enriched"))

#----------------------------
# get sequence 200nt around binding sites
#----------------------------

transcript_fasta <- readDNAStringSet("/Users/melinaklostermann/Documents/projects/anno/gencodevM23/genco

transcript_anno_meta <- names(transcript_fasta)
transcript_anno_meta <- data.frame(all = transcript_anno_meta) %>%
  tidyr::separate(., col = all,
                  into = c("transcript_id", "gene_id", "a", "b", "isoform_name", "gene_name", "entrez_g

names_transcript_fasta <- sub("\\..*", "", transcript_anno_meta$transcript_id)

# add N in beginning in end to not run out of transcripts when search motif
n200 <- c(rep("N",200)) %>%
  paste(., collapse = "") %>%
  RNAStringSet()

transcript_fasta <- xscat(n200, transcript_fasta, n200)
names(transcript_fasta) <- names_transcript_fasta
```

```r
transcript_fasta_df <- data.frame(tx_name = names(transcript_fasta), width = width(transcript_fasta))

# sequences
mir181_bs_200_both_sides <- as.data.frame(mir181_enriched_set) %>%
  left_join(transcript_fasta_df, by= c(seqnames = "tx_name"), suffix = c(".bs", ".tx")) %>%
  mutate(end = end + 197, start = start -197) %>%
  dplyr::filter((end <  width.tx)  & (start > 0)) %>%
  makeGRangesFromDataFrame(., keep.extra.columns = T)

mir181_bs_200_both_sides_seq <- BSgenome::getSeq(x = transcript_fasta, mir181_bs_200_both_sides) %>%
  RNAStringSet()

names(mir181_bs_200_both_sides_seq) <- 1:NROW(mir181_bs_200_both_sides_seq)

# write fasta file for XSTREME
writeXStringSet(mir181_bs_200_both_sides_seq, filepath = paste0(out, "mirBS_200_both_sides_transcripts_
```

```r
# plot logo
motif <- read.table("/Users/melinaklostermann/Documents/projects/AgoCLIP_miR181/R_github/miR181_paper/F

colnames(motif) <- c("A", "C", "G", "U")
motif <- t(motif)

logo <- ggseqlogo::ggseqlogo(motif)

logo
```



```
ggsave(logo, filename = paste0(out, "ReviewerFigureQ8_xstreme_logo.pdf"), height = 6, width = 8, units =
```

7 Session info

```
sessionInfo()
```

```
## R version 4.3.2 (2023-10-31)
## Platform: x86_64-apple-darwin20 (64-bit)
## Running under: macOS Ventura 13.1
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.3-x86_64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.3-x86_64/Resources/lib/libRlapack.dylib;  LAPACK v
##
## 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/Berlin
## tzcode source: internal
##
## attached base packages:
## [1] stats4    stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] Biostrings_2.70.2      XVector_0.42.0         GenomicFeatures_1.54.3
##  [4] AnnotationDbi_1.64.1   Biobase_2.62.0         GenomicRanges_1.54.1
##  [7] GenomeInfoDb_1.38.6    IRanges_2.36.0         S4Vectors_0.40.2
## [10] BiocGenerics_0.48.1    lubridate_1.9.3        forcats_1.0.0
## [13] stringr_1.5.1          dplyr_1.1.4            purrr_1.0.2
## [16] readr_2.1.5            tidyr_1.3.1            tibble_3.2.1
## [19] ggplot2_3.4.4          tidyverse_2.0.0        knitr_1.45
##
## loaded via a namespace (and not attached):
##  [1] DBI_1.2.1                   bitops_1.0-7
##  [3] biomaRt_2.58.2              rlang_1.1.3
##  [5] magrittr_2.0.3              matrixStats_1.2.0
##  [7] compiler_4.3.2              RSQLite_2.3.5
##  [9] png_0.1-8                   systemfonts_1.0.5
## [11] vctrs_0.6.5                 pkgconfig_2.0.3
## [13] crayon_1.5.2                fastmap_1.1.1
## [15] dbplyr_2.4.0                labeling_0.4.3
## [17] utf8_1.2.4                  Rsamtools_2.18.0
## [19] rmarkdown_2.25              tzdb_0.4.0
## [21] ragg_1.2.7                  bit_4.0.5
## [23] xfun_0.42                   zlibbioc_1.48.0
## [25] ggseqlogo_0.2               cachem_1.0.8
## [27] progress_1.2.3              blob_1.2.4
## [29] highr_0.10                  DelayedArray_0.28.0
## [31] BiocParallel_1.36.0         parallel_4.3.2
## [33] prettyunits_1.2.0           R6_2.5.1
## [35] stringi_1.8.3               rtracklayer_1.62.0
```

```
## [37] SummarizedExperiment_1.32.0 Matrix_1.6-5
## [39] timechange_0.3.0            tidyselect_1.2.0
## [41] rstudioapi_0.15.0           abind_1.4-5
## [43] yaml_2.3.8                  codetools_0.2-19
## [45] curl_5.2.0                  lattice_0.22-5
## [47] withr_3.0.0                 KEGGREST_1.42.0
## [49] evaluate_0.23               BiocFileCache_2.10.1
## [51] xml2_1.3.6                  pillar_1.9.0
## [53] filelock_1.0.3              MatrixGenerics_1.14.0
## [55] generics_0.1.3              rprojroot_2.0.4
## [57] RCurl_1.98-1.14             hms_1.1.3
## [59] munsell_0.5.0               scales_1.3.0
## [61] glue_1.7.0                  tools_4.3.2
## [63] BiocIO_1.12.0               BSgenome_1.70.2
## [65] GenomicAlignments_1.38.2    XML_3.99-0.16.1
## [67] grid_4.3.2                  colorspace_2.1-0
## [69] GenomeInfoDbData_1.2.11     restfulr_0.0.15
## [71] cli_3.6.2                   rappdirs_0.3.3
## [73] textshaping_0.3.7           fansi_1.0.6
## [75] S4Arrays_1.2.0              gtable_0.3.4
## [77] digest_0.6.34               SparseArray_1.2.4
## [79] rjson_0.2.21                farver_2.1.1
## [81] memoise_2.0.1               htmltools_0.5.7
## [83] lifecycle_1.0.4             httr_1.4.7
## [85] here_1.0.1                  bit64_4.0.5
```