Deconvolution of CommonMind data

Chong Jin

2019-04-01

Contents

We construct the bulk TPM and use ICeDT to perform deconvolution.

Introduction

We will use CIBERSORT and ICeDT to do cell type deconvolution. Before that we need to compile a list of signature genes and their cell type-specific expression from single cell data. This file will first deal with MTG single cell data Allen Brain Atlas MTG (middle temporal gyrus) dataset. We will also incorporate psychENCODE single cell data later.

The procedure is:

1. Process the single cell RNA-seq data to generate cell type label based on k-means clustering. Since the data set already has its own cell type labels, we take the intersect of cell type labels in agreement from the authors and from our k-means clustering.
2. Use the `MAST` package to compile a list of signature genes (with FDR and log fold change) for every cell type (six in all).
3. We select the top 100 in log fold change as signature genes for every cell type. Then we calculate TPM for each cell type by pooling the UMI from every cell type.
4. Deconvolute using CIBERSORT and ICeDT.

```
library(AnnotationDbi)
library(GenomicFeatures)
```

```
# obtain gene length from gtf (used in bulk expression quantification):
MTG_dir = "~/scRNAseq_pipelines/MTG"
psychENCODE_dir = "~/psychENCODE_data"
gencode_fn = file.path(MTG_dir,"Homo_sapiens.GRCh37.70.processed.gtf")
genelength_fn = file.path(MTG_dir,"ExonicGeneLengths_GRCh37.RData")
if( !file.exists(genelength_fn) ){
  exdb = GenomicFeatures::makeTxDbFromGFF(file = gencode_fn,
                                          format="gtf",dataSource = gtf_link)
  exons.list.per.gene = GenomicFeatures::exonsBy(exdb,by="gene")
  exonic.gene.sizes = lapply(exons.list.per.gene,function(x){sum(width(reduce(x)))})
  save(file = genelength_fn, exonic.gene.sizes)
}
load(genelength_fn)
names(exonic.gene.sizes) = sub("\\.[^.]*", "", names(exonic.gene.sizes))
```

```
# obtain bulk TPM by parsing from counts
bulk_count = readRDS("CMC_MSSM-Penn-Pitt_Paul_geneExpressionRaw.rds")$so1
dim(bulk_count)
```

```
## [1] 56632   613
```

```
bulk_count[1:5,1:5]
```

```
##                 MSSM_RNA_BP_PFC_10 MSSM_RNA_BP_PFC_11 MSSM_RNA_BP_PFC_12
## ENSG00000000003                103                258                148
## ENSG00000000005                  2                  5                  4
## ENSG00000000419                372                531                597
## ENSG00000000457                211                326                324
## ENSG00000000460                141                239                167
##                 MSSM_RNA_BP_PFC_13 MSSM_RNA_BP_PFC_14
## ENSG00000000003                170                216
## ENSG00000000005                  4                  7
## ENSG00000000419                522                890
## ENSG00000000457                225                327
## ENSG00000000460                183                271
```

```
length(names(exonic.gene.sizes))
```

```
## [1] 56632
```

```
length(rownames(bulk_count))
```

```
## [1] 56632
```

```
all(names(exonic.gene.sizes) %in% rownames(bulk_count))
```

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

```
all(names(exonic.gene.sizes) == rownames(bulk_count))
```

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

```
TPM = bulk_count/unlist(exonic.gene.sizes)
TPM = (1e6)*t(t(TPM/colSums(TPM)))
TPM[1:5,1:5]
```

```
##                 MSSM_RNA_BP_PFC_10 MSSM_RNA_BP_PFC_11 MSSM_RNA_BP_PFC_12
## ENSG00000000003         4.27125024         26.7868573          4.9813756
## ENSG00000000005         0.06268007          0.4737888          0.1432255
## ENSG00000000419        21.11384083         25.4641620         30.5903142
## ENSG00000000457         4.22701944         20.2268505          5.3671536
## ENSG00000000460         2.57058819          9.4496441          1.6088558
##                 MSSM_RNA_BP_PFC_13 MSSM_RNA_BP_PFC_14
## ENSG00000000003          7.8964969         11.9143070
## ENSG00000000005          0.3341581          0.5851302
## ENSG00000000419         43.3972374         49.6350949
## ENSG00000000457         14.4801590          5.8619844
## ENSG00000000460          4.1435438          3.1313224
```

```
# read reference TPM by parsing single cell data (note that this is probably in hg38:)
tpm_signature_genes = readRDS(file.path(psychENCODE_dir,"tpm_signature_genes.rds"))
table(rownames(tpm_signature_genes) %in% rownames(TPM))
```

```
##
```

```
## FALSE  TRUE
##     5   595
```

```
tpm_signature_genes = tpm_signature_genes[rownames(tpm_signature_genes) %in% rownames(TPM),]
```

```
bulk_subset = TPM[rownames(tpm_signature_genes), ]
all(rownames(bulk_subset) == rownames(tpm_signature_genes))
```

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

Deconvolution

```
par(mar=c(5,4,1,1), bty="n", mfrow=c(1,2), cex=0.8)

# run CIBERSORT
# change the input format according to the CIBERSORT website:
bulk_subset[1:5,1:5]
```

```
##                 MSSM_RNA_BP_PFC_10 MSSM_RNA_BP_PFC_11 MSSM_RNA_BP_PFC_12
## ENSG00000179399          10.838144          38.521471          22.964193
## ENSG00000234377           1.733163           3.950815           3.442654
## ENSG00000080573           7.530356           7.111844           9.882323
## ENSG00000164199          13.849694          10.490231          34.667779
## ENSG00000130203         107.715609         728.077275         188.022663
##                 MSSM_RNA_BP_PFC_13 MSSM_RNA_BP_PFC_14
## ENSG00000179399          8.7964158          13.035092
## ENSG00000234377          0.6444057           2.183647
## ENSG00000080573          8.7106225          11.821318
## ENSG00000164199          9.7931692          59.199854
## ENSG00000130203        481.9477049         160.689464
```

```
tpm_signature_genes[1:5,1:5]
```

```
##                      Astro       Exc        Inh     Micro     Oligo
## ENSG00000179399 6668.1804 224.02348 341.096681 305.63340 166.33537
## ENSG00000234377  908.2096  13.18046   9.771217  37.18847  14.31031
## ENSG00000080573 1193.2748  29.04559  19.193073  12.10691  19.61601
## ENSG00000164199  576.3236  15.39329   8.616831  24.34366  11.64489
## ENSG00000130203 2463.4377  44.64310  48.179781 298.39039  58.41827
```

```
write.table(cbind(rowname=rownames(tpm_signature_genes), tpm_signature_genes),
            "signature_genes_brain_from_psychENCODE.txt",
            sep="\t", quote=FALSE, row.names = FALSE)
write.table(cbind(rowname=rownames(bulk_subset), bulk_subset),
            "mixture_brain_from_CMC.txt",
            sep="\t", quote=FALSE, row.names = FALSE)
# read the output generated from CIBERSORT website (no Quantile normalization, 1000 permutations)
cibersort_results = read.csv(file.path(psychENCODE_dir,
            "CIBERSORT.Output_CMC_using_sig_genes_from_psychENCODE600_no_QN.csv"))
prop_cibersort = cibersort_results[,2:7]
prop_cibersort[1:5,1:5]
```

```
##        Astro       Exc        Inh Micro     Oligo
## 1 0.07430303 0.4154224 0.07759200     0 0.4326826
## 2 0.13247500 0.2777463 0.14694312     0 0.4428356
```

```
## 3 0.03183193 0.6138444 0.19217242     0 0.1621512
## 4 0.00000000 0.5771741 0.07448533     0 0.3483406
## 5 0.05845142 0.4612981 0.18363447     0 0.2966160
```

```
dim(prop_cibersort)
```

```
## [1] 613   6
```

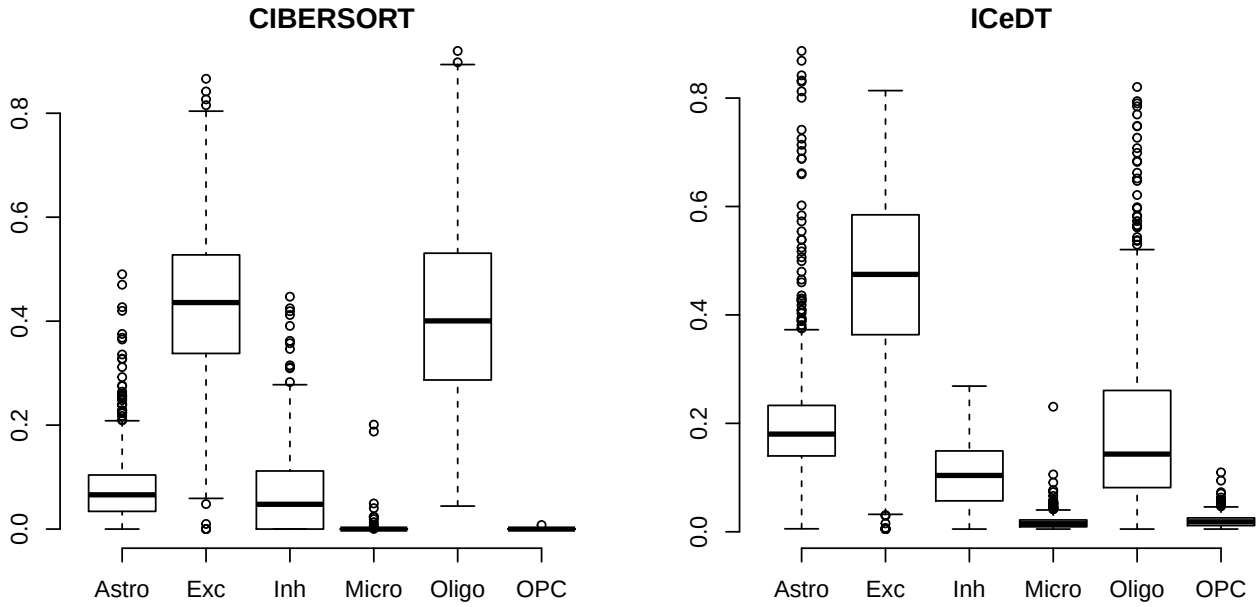
```
boxplot(prop_cibersort, main="CIBERSORT")


# run ICeDT
if (!file.exists(file.path(psychENCODE_dir,"CMC_ICeDT_fitw0_genes_from_psychENCODE.rds"))) {
  # takes 15min on my laptop.
  date()
  fitw0 = ICeDT::ICeDT(Y=bulk_subset, Z=tpm_signature_genes,
                       tumorPurity=rep(0, ncol(bulk_subset)), refVar=NULL,
                       rhoInit=NULL, maxIter_prop = 500, maxIter_PP = 250,
                       rhoConverge = 1e-2)
  date()
  saveRDS(fitw0, file = file.path(psychENCODE_dir,"CMC_ICeDT_fitw0_genes_from_psychENCODE.rds"))
}
fitw0 = readRDS(file.path(psychENCODE_dir,"CMC_ICeDT_fitw0_genes_from_psychENCODE.rds"))

# plot ICeDT
prop_icedt = t(fitw0$rho)[,-1]
prop_icedt[1:5,1:5]
```

```
##                         Astro       Exc       Inh       Micro      Oligo
## MSSM_RNA_BP_PFC_10 0.1166854 0.6394412 0.1170381 0.005000000 0.09567739
## MSSM_RNA_BP_PFC_11 0.2666046 0.3249298 0.1407645 0.007896343 0.21438045
## MSSM_RNA_BP_PFC_12 0.1062061 0.5532598 0.2555638 0.009652946 0.05078336
## MSSM_RNA_BP_PFC_13 0.1346518 0.6114900 0.1086516 0.018072502 0.10876149
## MSSM_RNA_BP_PFC_14 0.1798983 0.5465042 0.1571795 0.019389423 0.06437430
```

```
dim(prop_icedt)
```

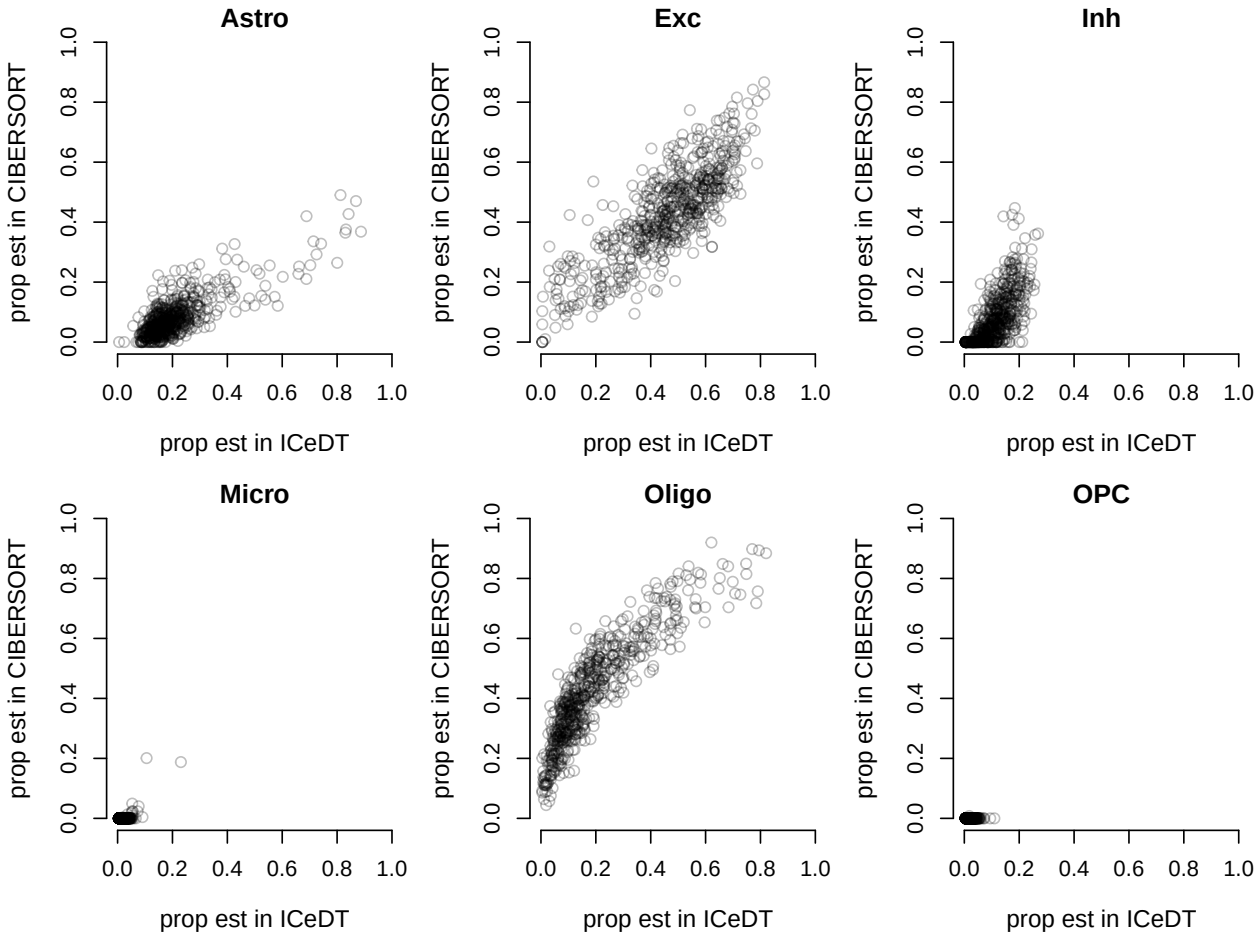
```
## [1] 613   6
```

```
boxplot(prop_icedt, main="ICeDT")
```




```
#> table(colData(sce)$cell_type)
#
#Astro   Exc   Inh Micro Oligo   OPC
# 2647 13353  4064   115  2425  1046
```

```
par(mar=c(5,4,1,1), bty="n", mfrow=c(2,3), cex=0.8)
for (i in colnames(prop_cibersort)) {
 plot(prop_icedt[,i], prop_cibersort[,i],
                xlab="prop est in ICeDT",
                ylab="prop est in CIBERSORT",
                # xlab=xylab_formatter(deparse(substitute(data1))),
                # ylab=xylab_formatter(deparse(substitute(data2))),
                pch = 1, cex = 1.1, cex.lab=1.1, col=rgb(0,0,0,0.25),
                xlim=c(0,1), ylim=c(0,1),
                # nrpoints=0,
                main=i
        )
}
```



More analysis based on ICeDT results

```
par(mar=c(5,4,1,1), bty="n", mfrow=c(1,2), cex=0.8)
Geneset = "psychENCODE600"
# dir.create("./figures")

p1 = fitw0$cProb
dim(p1)
```

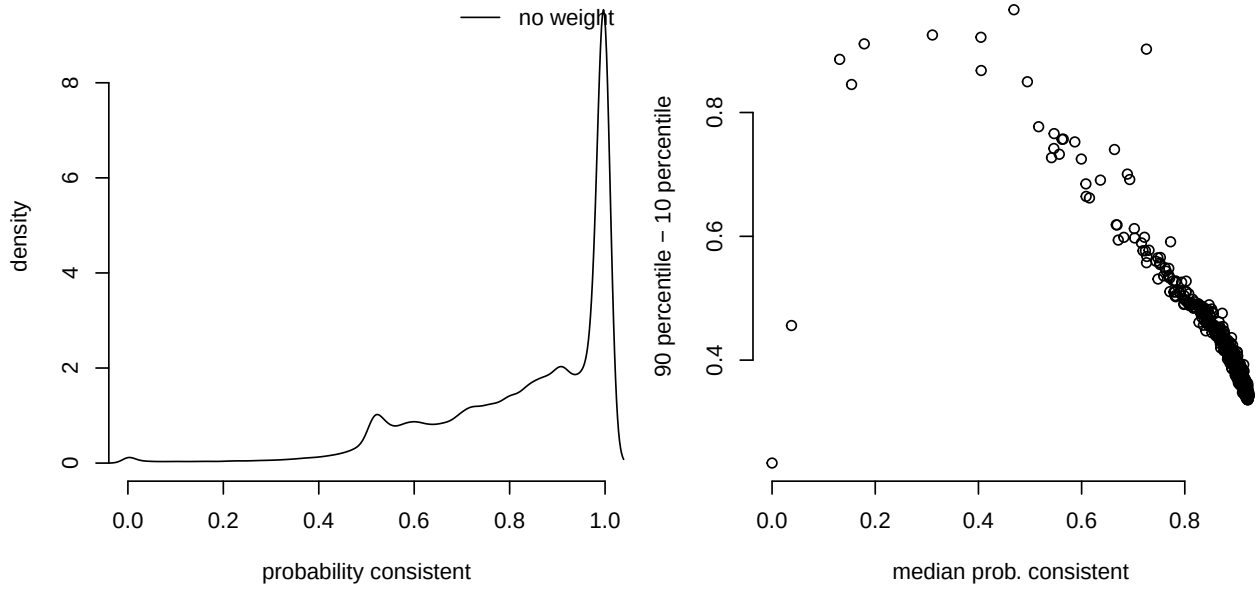
```
## [1] 595 613
```

```
p1[1:2,1:5]
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] 0.4522327 0.5036935 0.9978593 0.8642138 0.2367322
## [2,] 0.4911941 0.4549427 0.9985036 0.7472776 0.2766379
```

```
p1 = data.matrix(p1)

q90 <- function(v){
  qs = quantile(v, probs=c(0.10, 0.90))
  qs[2] - qs[1]
}
```

```
# pdf(sprintf("./figures/probConsistent_GeneSet%s.pdf",Geneset),
#     width=9, height=3)
plot(density(c(p1))$y, main="", xlim=c(0,1),
     xlab="probability consistent", ylab="density", type="n")
lines(density(c(p1)), lty=1, col="black")
legend("topright", c("no weight"), lty=c(1,2),
       col=c("black"), bty="n")

plot(apply(p1, 1, median), apply(p1, 1, q90),
     xlab="median prob. consistent", ylab="90 percentile - 10 percentile")
```




```
# dev.off()

# Scatterplot of predicted vs. observed gene expression
dim(fitw0$rho[-1,])
```

```
## [1]   6 613
```

```
dim(tpm_signature_genes)
```

```
## [1] 595   6
```

```
dim(bulk_subset)
```

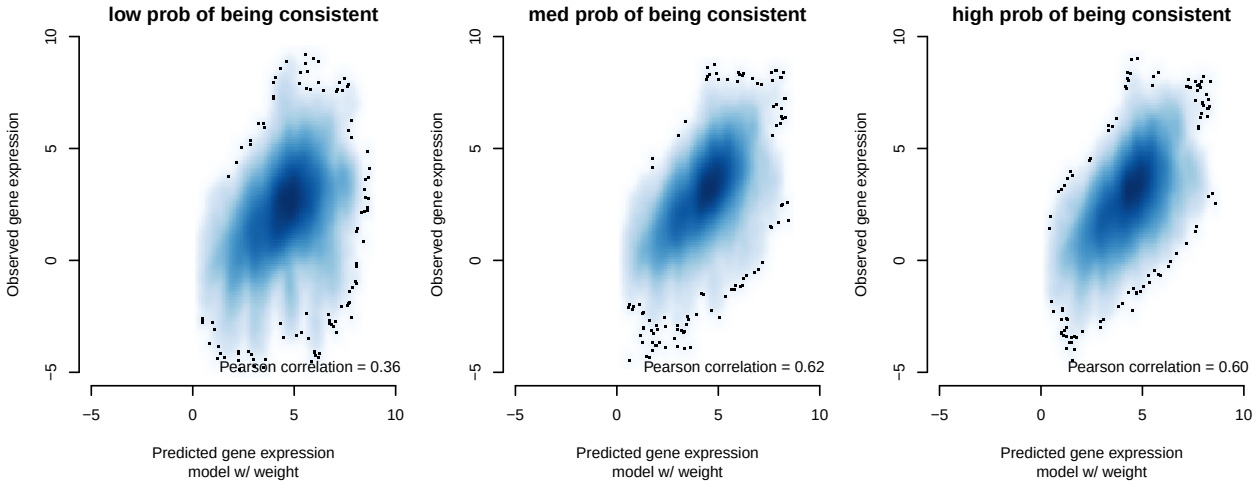
```
## [1] 595 613
```

```
predicted_bulk_w0 = tpm_signature_genes %*% fitw0$rho[-1,]
p1_cutoffs = quantile(p1, c(0.333,0.666))

cat(sprintf("Consistent probability cutoffs for model w/ weight: %.3f, %.3f \n",
            p1_cutoffs[1], p1_cutoffs[2]))
```

```
## Consistent probability cutoffs for model w/ weight: 0.791, 0.976
```

```
# pdf(sprintf("./figures/ExpectedVsObservedExpr_GeneSet%s.pdf",
#             Geneset), width=9, height=6)
par(mar=c(5,4,1,1), bty="n", mfrow=c(1,3), cex=0.6)
```

```r
# plot observed vs. expected expression, stratified by 3-quantiles
plot_log1p = function(x, y, ...) {
  smoothScatter(log(x+1e-5), log(y+1e-5), xlim=c(-5, 10), ylim=c(-5, 10), ...)
  legend("bottomright", bty="n",
         legend=sprintf("Pearson correlation = %.2f", cor(log(x+1e-5), log(y+1e-5))))
}

plot_log1p(c(predicted_bulk_w0)[p1 < p1_cutoffs[1]], c(bulk_subset)[p1 < p1_cutoffs[1]],
     xlab="Predicted gene expression", ylab="Observed gene expression",
     sub="model w/ weight", main="low prob of being consistent")
plot_log1p(c(predicted_bulk_w0)[p1 >= p1_cutoffs[1] & p1 <= p1_cutoffs[2]],
           c(bulk_subset)[p1 >= p1_cutoffs[1] & p1 <= p1_cutoffs[2]],
     xlab="Predicted gene expression", ylab="Observed gene expression",
     sub="model w/ weight", main="med prob of being consistent")
plot_log1p(c(predicted_bulk_w0)[p1 > p1_cutoffs[2]], c(bulk_subset)[p1 > p1_cutoffs[2]],
     xlab="Predicted gene expression", ylab="Observed gene expression",
     sub="model w/ weight", main="high prob of being consistent")
```




```r
# dev.off()
```

```r
sessionInfo()
```

```
## R version 3.5.3 (2019-03-11)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 18.04.2 LTS
##
## Matrix products: default
## BLAS: /usr/lib/x86_64-linux-gnu/openblas/libblas.so.3
## LAPACK: /usr/lib/x86_64-linux-gnu/libopenblasp-r0.2.20.so
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
```

```
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
## [1] GenomicFeatures_1.34.3 GenomicRanges_1.34.0   GenomeInfoDb_1.18.2
## [4] AnnotationDbi_1.44.0   IRanges_2.16.0         S4Vectors_0.20.1
## [7] Biobase_2.42.0         BiocGenerics_0.28.0
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.0                  compiler_3.5.3
##  [3] XVector_0.22.0              prettyunits_1.0.2
##  [5] bitops_1.0-6                tools_3.5.3
##  [7] zlibbioc_1.28.0             progress_1.2.0
##  [9] biomaRt_2.38.0              digest_0.6.18
## [11] bit_1.1-14                  lattice_0.20-38
## [13] RSQLite_2.1.1               evaluate_0.13
## [15] memoise_1.1.0               pkgconfig_2.0.2
## [17] rlang_0.3.1                 Matrix_1.2-15
## [19] DelayedArray_0.8.0          DBI_1.0.0
## [21] yaml_2.2.0                  xfun_0.5
## [23] GenomeInfoDbData_1.2.0      rtracklayer_1.42.1
## [25] httr_1.4.0                  stringr_1.4.0
## [27] knitr_1.21                  Biostrings_2.50.2
## [29] hms_0.4.2                   grid_3.5.3
## [31] bit64_0.9-7                 R6_2.4.0
## [33] BiocParallel_1.16.6         XML_3.98-1.17
## [35] rmarkdown_1.11              blob_1.1.1
## [37] magrittr_1.5                matrixStats_0.54.0
## [39] GenomicAlignments_1.18.1    Rsamtools_1.34.1
## [41] htmltools_0.3.6             SummarizedExperiment_1.12.0
## [43] assertthat_0.2.0            KernSmooth_2.23-15
## [45] stringi_1.3.1               RCurl_1.95-4.11
## [47] crayon_1.3.4
```