BioMartConversion

Fatma Betul Dincaslan

2022-10-04

BioMart Conversion to Explore the Distribution of Gene/Transcripts Types of Count Data

If you are a sequencing assay developer, and curios about how many/what type of gene/transcripts that you are able to capture after a sequencing run, this annotation script might be helpful. The visualization will be a big plus to interpret the results easily. Hope you enjoy, too!

Packages

```
# un/comment block of code use Command+Shift+C
# packages required
# if (!require("BiocManager", quietly = TRUE))
#  install.packages("BiocManager")
# BiocManager::install("biomaRt")     # Install bioMart package
library(biomaRt)                     # Load bioMart
# install.packages("dplyr")           # Install dplyr package
library("dplyr")                     # Load dplyr
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:biomaRt':
##
##     select

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# install.packages("devtools")
# devtools::install_github("tidyverse/ggplot2")
# install.packages("ggrepel")
# install.packages("tidyverse")
library(ggplot2)
library(ggrepel)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --

## v tibble  3.1.7     v purrr   0.3.4
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x dplyr::select() masks biomaRt::select()
```

```
#library(ggpubr)
#theme_set(theme_pubr())
# install.packages("lessR")
library(lessR)
```

```
##
## lessR 4.2.2                         feedback: gerbing@pdx.edu
## --------------------------------------------------------------
## > d <- Read("")   Read text, Excel, SPSS, SAS, or R data file
##   d is default data frame, data= in analysis routines optional
##
## Learn about reading, writing, and manipulating data, graphics,
## testing means and proportions, regression, factor analysis,
## customization, and descriptive statistics from pivot tables.
##   Enter:  browseVignettes("lessR")
##
## View changes in this or recent versions of lessR.
##   Enter: help(package=lessR)  Click: Package NEWS
##   Enter: interact()  for access to interactive graphics
##   New function: reshape_long() to move data from wide to long
##
##
## Attaching package: 'lessR'
##
## The following objects are masked from 'package:dplyr':
##
##     recode, rename
```

```
# install.packages("UpSetR")
library(UpSetR)
# install.packages("gcookbook")
library(gcookbook)
```

Ensembl Reference Data and Atribute Selection

```
# useful tutorials: https://bioconductor.org/packages/release/bioc/vignettes/biomaRt/inst/doc/accessing
# https://bioconductor.org/packages/release/bioc/vignettes/biomaRt/inst/doc/accessing_other_marts.html
# might be useful https://stackoverflow.com/questions/28543517/how-can-i-convert-ensembl-id-to-gene-sym

# because my seq data is from human cell line, I will chose homo sapiens
ensembl <- useEnsembl(biomart = "genes", dataset = "hsapiens_gene_ensembl")
```

```
## Ensembl site unresponsive, trying useast mirror
```

```
mart <- useDataset("hsapiens_gene_ensembl", useMart("ensembl"))
# these are the attributes that you can select
attributes = listAttributes(ensembl)
# we will mostly be interested in these
# description Gene description feature_page
# gene_biotype Gene type feature_page
# hgnc_symbol HGNC symbol feature_page
# external_synonym Gene Synonym feature_page
# ensembl_gene_id Gene stable ID feature_page
```

Setting the directory and Retrieval of/Modification on Count Data

```
# dir <- "/Users/Your/Path" # example /Users/lion/202209Fastq"
file <- "Sample_STAR_Cut_counts.txt" # retrieve the file

# set the directory first
setwd(dir)
# read rhe file
# genlist <- read.csv(file = file, header = FALSE) #lets use htseq counts only here for this explorator
genlist <- read.csv(file = file, header = FALSE, sep = "")
head(genlist)
```

```
##                    V1 V2
## 1 ENSG00000000003.15  0
## 2  ENSG00000000005.6  0
## 3 ENSG00000000419.14  2
## 4 ENSG00000000457.14  0
## 5 ENSG00000000460.17  3
## 6 ENSG00000000938.13  0
```

```
# View(genlist)

# remove last 5 line which corresponds to alignment statistics overall
rown_genelist <- dim(genlist)[1]
genlist<- genlist[-c((rown_genelist-4):rown_genelist) , ]

# add column names, id and counts
names(genlist)[1] <- "id"
names(genlist)[2] <- "counts"
head(genlist)
```

```
##                    id counts
## 1 ENSG00000000003.15      0
## 2  ENSG00000000005.6      0
## 3 ENSG00000000419.14      2
## 4 ENSG00000000457.14      0
## 5 ENSG00000000460.17      3
## 6 ENSG00000000938.13      0
```

Building a BioMart Query

getBM() will help to retrieve relevant attributes of the matching Ensembl IDs from Ensembl BioMart Server.

```
# # if you retrieve data without versions such as ENSGXXXXXXX instead of ENSGXXXXXXX.y
# BMconvert <- getBM(values = genlist$id, mart = mart, attributes = c("ensembl_gene_id", "external_syno
BMconvert <- getBM(values = genlist$id, mart = mart, attributes = c("ensembl_gene_id_version", "externa
head(BMconvert)
```

```
##   ensembl_gene_id_version external_synonym hgnc_symbol gene_biotype
## 1       ENSG00000210049.1             MTTF       MT-TF      Mt_tRNA
## 2       ENSG00000210049.1             trnF       MT-TF      Mt_tRNA
## 3       ENSG00000211459.2              12S     MT-RNR1      Mt_rRNA
## 4       ENSG00000211459.2           MOTS-c     MT-RNR1      Mt_rRNA
## 5       ENSG00000211459.2           MTRNR1     MT-RNR1      Mt_rRNA
## 6       ENSG00000210077.1             MTTV       MT-TV      Mt_tRNA
##                                                                description
## 1 mitochondrially encoded tRNA-Phe (UUU/C) [Source:HGNC Symbol;Acc:HGNC:7481]
## 2 mitochondrially encoded tRNA-Phe (UUU/C) [Source:HGNC Symbol;Acc:HGNC:7481]
## 3         mitochondrially encoded 12S rRNA [Source:HGNC Symbol;Acc:HGNC:7470]
## 4         mitochondrially encoded 12S rRNA [Source:HGNC Symbol;Acc:HGNC:7470]
## 5         mitochondrially encoded 12S rRNA [Source:HGNC Symbol;Acc:HGNC:7470]
## 6   mitochondrially encoded tRNA-Val (GUN) [Source:HGNC Symbol;Acc:HGNC:7500]
```

```
# merging the data
merged_list <- merge(genlist, BMconvert, by = 1, all =TRUE)
head(merged_list)
```

```
##                   id counts external_synonym hgnc_symbol   gene_biotype
## 1 ENSG00000000003.15      0          TSPAN-6      TSPAN6 protein_coding
## 2 ENSG00000000003.15      0             T245      TSPAN6 protein_coding
## 3 ENSG00000000003.15      0           TM4SF6      TSPAN6 protein_coding
## 4  ENSG00000000005.6      0           BRICD4        TNMD protein_coding
## 5  ENSG00000000005.6      0           tendin        TNMD protein_coding
## 6  ENSG00000000005.6      0              TEM        TNMD protein_coding
##                                          description
## 1 tetraspanin 6 [Source:HGNC Symbol;Acc:HGNC:11858]
## 2 tetraspanin 6 [Source:HGNC Symbol;Acc:HGNC:11858]
## 3 tetraspanin 6 [Source:HGNC Symbol;Acc:HGNC:11858]
## 4   tenomodulin [Source:HGNC Symbol;Acc:HGNC:17757]
## 5   tenomodulin [Source:HGNC Symbol;Acc:HGNC:17757]
## 6   tenomodulin [Source:HGNC Symbol;Acc:HGNC:17757]
```

```
# select only with assigned read counts which is counts >0
genlist_c <- merged_list[merged_list$counts>0, ]
# more data cleaning to filter out unique gene/transcript IDs only
# remove non-unique "ensembl id" and the counts (if possible combine counts of same synonym in one gene
genl_unique <- distinct(genlist_c, id, .keep_all = TRUE)
```

Getting the statistics of the frequency of gene biotypes

```r
# factorize the gene_biotype
factor(genl_unique$gene_biotype)
```

```r
# determine the frequency of these in the given data set
genty_freq <- table(genl_unique$gene_biotype)
head(genty_freq)
```

```
##
## artifact   lncRNA    miRNA misc_RNA  Mt_rRNA  Mt_tRNA
##        2      785       18      198        2       11
```

```r
# percent-wise representation of the unique occurrence of genes with assigned gene_types
genty_freq <- as.data.frame(genty_freq)
names(genty_freq)[1] <- "types"
names(genty_freq)[2] <- "freq"
# head(genty_freq)
sumf <- sum(genty_freq$freq)
genty_freq <- genty_freq%>% mutate(perc = (freq/sumf)*100)
# lets see them ona proper table format
knitr::kable(genty_freq)
```

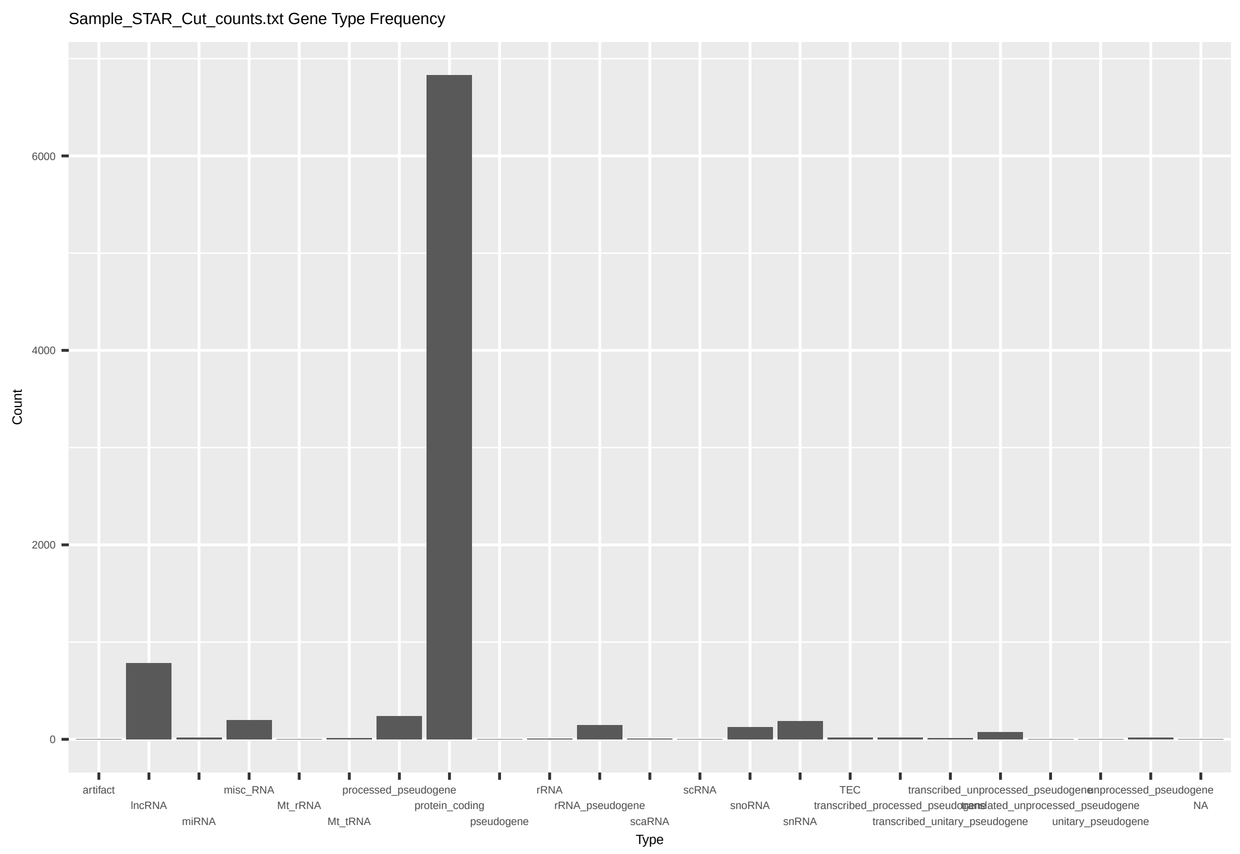
types	freq	perc
artifact	2	0.0229595
lncRNA	785	9.0115945
miRNA	18	0.2066353
misc_RNA	198	2.2729882
Mt_rRNA	2	0.0229595
Mt_tRNA	11	0.1262771
processed_pseudogene	241	2.7666169
protein_coding	6830	78.4066123
pseudogene	1	0.0114797
rRNA	7	0.0803582
rRNA_pseudogene	148	1.6990013
scaRNA	10	0.1147974
scRNA	1	0.0114797
snoRNA	125	1.4349673
snRNA	187	2.1467111
TEC	20	0.2295948
transcribed_processed_pseudogene	17	0.1951556
transcribed_unitary_pseudogene	15	0.1721961
transcribed_unprocessed_pseudogene	73	0.8380209
translated_unprocessed_pseudogene	1	0.0114797
unitary_pseudogene	1	0.0114797
unprocessed_pseudogene	18	0.2066353

Visualization of Frequency Graphs

I will share three different ways of visualization. You can pick up the one suits you more. If you are comparing two data sets, you might also want to use Upset graphs.

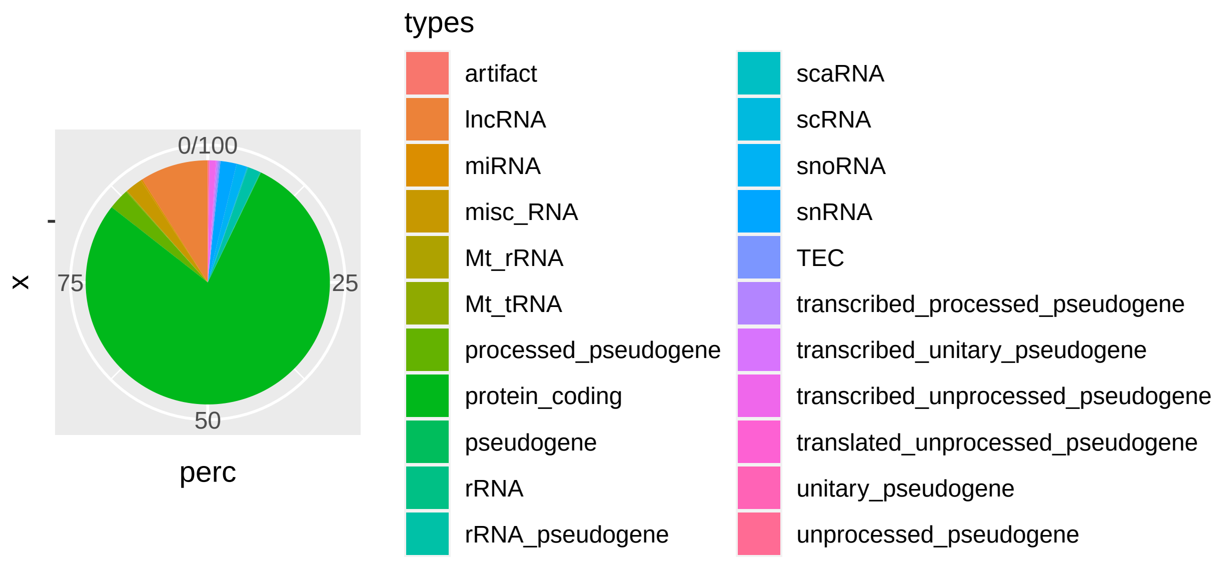
```r
# plot the percentage of different RNA types with a bar graph
# might be helpful: https://www.r-bloggers.com/2019/05/detailed-guide-to-the-bar-chart-in-r-with-ggplot
f_plot <- ggplot(data.frame(genl_unique$gene_biotype), aes(x=genl_unique$gene_biotype)) +
  geom_bar()+
  theme(text = element_text(size = 5))+
  scale_x_discrete(guide = guide_axis(n.dodge = 3))+
  labs(title=paste(file,"Gene Type Frequency"), x ="Type", y = "Count")

f_plot
```



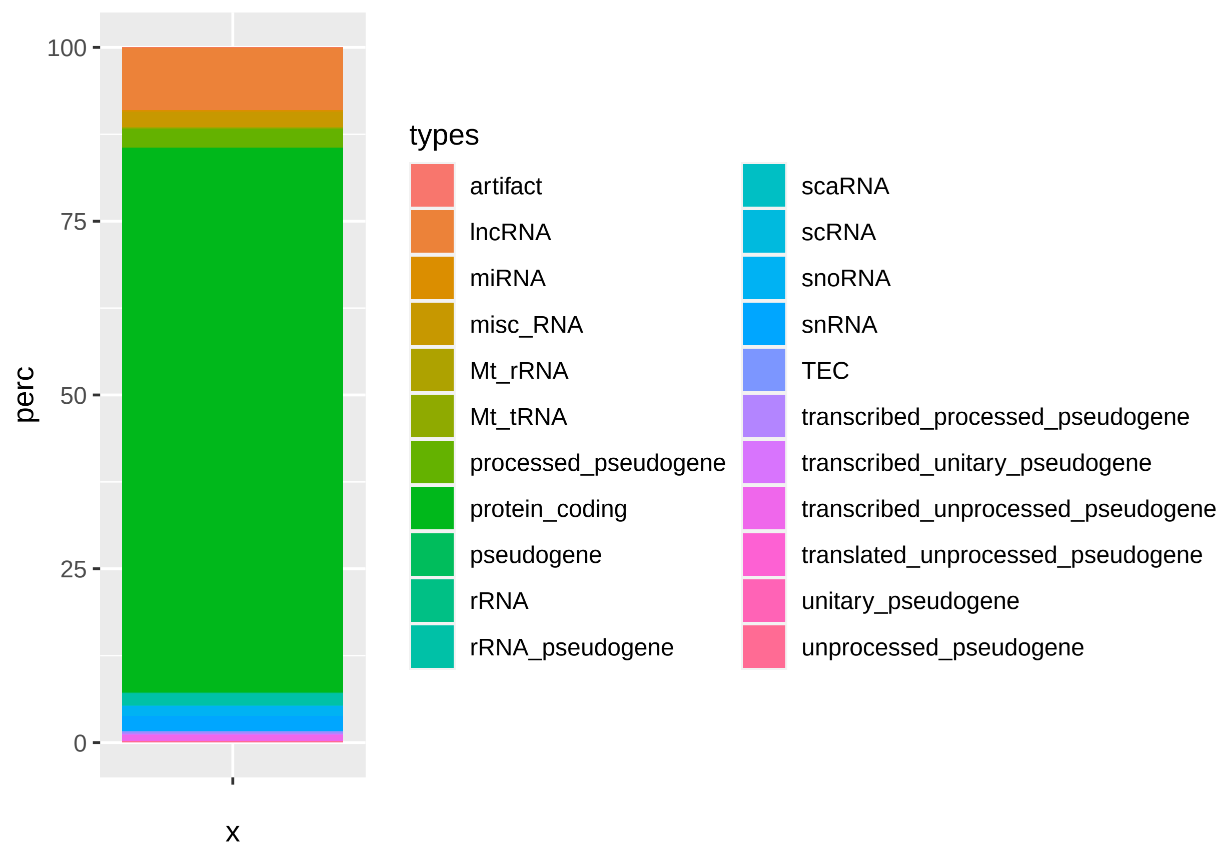
```r
# alternative pie chart
# might be helpful: https://ggplot2.tidyverse.org/reference/coord_polar.html
pc <- ggplot(genty_freq, aes(x="", y=perc, fill=types)) +
  geom_bar(stat="identity", width=1) +
  coord_polar("y", start=0)

pc
```



```
# or stacked bar chart representation
# might be helpful: https://r-graph-gallery.com/48-grouped-barplot-with-ggplot2.html
sbp<- ggplot(genty_freq, aes(x="", y=perc, fill=types)) +
  geom_bar(width = 1, stat = "identity")

sbp
```



```
# # find the unique values for each and intersections, then plot upsetR, an example code given below
# # might be helpful: https://cran.r-project.org/web/packages/UpSetR/vignettes/basic.usage.html
# Out_list <- list(S1 = rownames(genl_unique1), S2 = rownames(genl_unique2))
# Out_gns <- UpSetR::fromList(Out_list)
# UpSetR::upset(Out_gns, order.by = "freq")
```

Session Info

```
sessionInfo()
```

```
## R version 4.2.0 (2022-04-22)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur/Monterey 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] stats     graphics  grDevices utils     datasets  methods   base
```

```
##
## other attached packages:
##  [1] gcookbook_2.0      UpSetR_1.4.0       lessR_4.2.2        forcats_0.5.1
##  [5] stringr_1.4.0      purrr_0.3.4        readr_2.1.2        tidyr_1.2.0
##  [9] tibble_3.1.7       tidyverse_1.3.2    ggrepel_0.9.1      ggplot2_3.3.6.9000
## [13] dplyr_1.0.9        biomaRt_2.52.0
##
## loaded via a namespace (and not attached):
##   [1] bitops_1.0-7           fs_1.5.2               lubridate_1.8.0
##   [4] bit64_4.0.5            RColorBrewer_1.1-3     filelock_1.0.2
##   [7] progress_1.2.2         httr_1.4.3             GenomeInfoDb_1.32.2
##  [10] tools_4.2.0            backports_1.4.1        utf8_1.2.2
##  [13] R6_2.5.1               DBI_1.1.3              BiocGenerics_0.42.0
##  [16] colorspace_2.0-3       withr_2.5.0            gridExtra_2.3
##  [19] tidyselect_1.1.2       prettyunits_1.1.1      bit_4.0.4
##  [22] curl_4.3.2             compiler_4.2.0         cli_3.3.0
##  [25] rvest_1.0.2            Biobase_2.56.0         xml2_1.3.3
##  [28] labeling_0.4.2         scales_1.2.0           DEoptimR_1.0-11
##  [31] robustbase_0.95-0      rappdirs_0.3.3         digest_0.6.29
##  [34] rmarkdown_2.14         XVector_0.36.0         jpeg_0.1-9
##  [37] pkgconfig_2.0.3        htmltools_0.5.3        highr_0.9
##  [40] dbplyr_2.2.1           fastmap_1.1.0          rlang_1.0.4
##  [43] readxl_1.4.0           rstudioapi_0.13        RSQLite_2.2.15
##  [46] farver_2.1.1           generics_0.1.3         jsonlite_1.8.0
##  [49] zip_2.2.0              googlesheets4_1.0.0    RCurl_1.98-1.7
##  [52] magrittr_2.0.3         GenomeInfoDbData_1.2.8 leaps_3.1
##  [55] interp_1.1-3           Rcpp_1.0.9             munsell_0.5.0
##  [58] S4Vectors_0.34.0       fansi_1.0.3            lifecycle_1.0.1
##  [61] stringi_1.7.8          yaml_2.3.5             zlibbioc_1.42.0
##  [64] plyr_1.8.7             BiocFileCache_2.4.0    grid_4.2.0
##  [67] blob_1.2.3             crayon_1.5.1           deldir_1.0-6
##  [70] lattice_0.20-45        Biostrings_2.64.0      haven_2.5.0
##  [73] hms_1.1.1              KEGGREST_1.36.3        knitr_1.39
##  [76] pillar_1.8.0           stats4_4.2.0           reprex_2.0.1
##  [79] XML_3.99-0.10          glue_1.6.2             evaluate_0.15
##  [82] latticeExtra_0.6-30    modelr_0.1.8           png_0.1-7
##  [85] vctrs_0.4.1            tzdb_0.3.0             cellranger_1.1.0
##  [88] gtable_0.3.0           assertthat_0.2.1       cachem_1.0.6
##  [91] openxlsx_4.2.5         xfun_0.31              broom_1.0.0
##  [94] viridisLite_0.4.0      googledrive_2.0.0      gargle_1.2.0
##  [97] AnnotationDbi_1.58.0   memoise_2.0.1          IRanges_2.30.0
## [100] ellipse_0.4.3          ellipsis_0.3.2
```