Alveolar Macrophages after Murine Lung Transplant

180166459

2/8/2021

## R Markdown

# **BMS353**

# **Bioinformatics**

**Alveolar Macrophages after Murine Lung Transplant - (mouse)**

FOR NOT INCLUDING R CODE IN REPORT: {r echo = FALSE} FOR NOT INCLUDING OUTPUT/CONSOLE PRINT in REPORT: {r eval = FALSE}

This is an R Markdown document. Markdown is a simple formatting syntax.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

**INTRODUCTION**

! [Alt text] (Users353-Alveolar-Macrophages—RNA-seq-dataset1macrophages.jpg)

In this report we will analyze data for a research study that covers the following topic: Alveolar Macrophages after Murine Lung Transplant. For this we have the following resources provided:

[GEO](https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE116583)

[Google drive](https://drive.google.com/drive/folders/1AqYi0Ps5t5xo6XYWXjOtO6eDOMpT1ADE) - Google drive folder with the following content:

* meta\_data
* salmon-quant.sf files
* tx2gene.csv

The project name BMS353-Alveolar-Macrophages—RNA-seq-dataset with all files and the code are available on github in a repository:

[Github](https://github.com/IoanaAndra/BMS353-Alveolar-Macrophages---RNA-seq-dataset)

In order to use the necessary functions and analyze the data, we use some packages available in R:

* readr
* ggplot2
* dplyr

Every chunk of code can be name like this:

Installing packages in R

#SECTION FOR PACKAGES  
#installing all necessary packages  
# IF using same R version (4.0.3) and have all packages installed already, all lines of this section can remain commented   
  
   
# install.packages("readr")  
# install.packages("ggplot2")  
# install.packages("dplyr")  
# install.packages("learnr")  
# install.packages("stringi")  
# install.packages("tidyverse")  
  
if (!requireNamespace("BiocManager", quietly = TRUE)) # installing tximport  
 install.packages("BiocManager")  
#   
# BiocManager::install("tximport")  
  
install.packages("jsonlite")  
# BiocManager::install("GenomicFeatures")  
  
  
  
install.packages() #check isntalled packages. This can be seen also in Packages tab  
  
old.packages()   
  
# update.packages(ask = FALSE) #update installed packages without asking permission from user

Reading tx2gene file using read.csv()

With head, we display first rows of data

library(readr)

## Warning: package 'readr' was built under R version 4.0.5

tx2GeneFile <- read.csv(file = 'tx2gene.csv')  
head(tx2GeneFile)

## TXNAME GENEID  
## 1 ENSMUST00000193812 ENSMUSG00000102693  
## 2 ENSMUST00000082908 ENSMUSG00000064842  
## 3 ENSMUST00000192857 ENSMUSG00000102851  
## 4 ENSMUST00000161581 ENSMUSG00000089699  
## 5 ENSMUST00000192183 ENSMUSG00000103147  
## 6 ENSMUST00000193244 ENSMUSG00000102348

#view(tx2GeneFile)

We are going to display tx2gene.csv

library(readr) #package used to import spreadsheets in R  
  
dataPath <- "tx2gene.csv" #asigning value which represent path of file  
  
file.exists(dataPath) #file.exists() return TRUE id file can be found or FALSE if it is not found

## [1] TRUE

dataGenes <- read.csv(dataPath) #storing content of csv file into variable data\_genes  
  
  
  
#check data frame that we created  
  
 # testData <- read\_table(dataGenes) ##\*\*\*\*\*\*\*Error: `file` must be a string, raw vector or a connection.  
#   
# View(test) #view() - invoking a spreadsheet style data viewer nb  
#   
#   
# head(dataGenes)

**ANALYSIS**

library(tximport)  
library(readr)  
library(dplyr)

## Warning: package 'dplyr' was built under R version 4.0.5

##   
## Attaching package: 'dplyr'

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

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

# Read the sample information into R  
sampleinfo <- read.delim("meta\_data/sampleInfo.txt")  
View(sampleinfo) # view sampleinfo.txt in a new tab  
#glimpse(sampleinfo)  
sampleinfo

## geo\_accession name condition time run  
## 1 GSM3243460 N01\_AM\_Naive naive 0hr SRR7457557  
## 2 GSM3243461 N02\_AM\_Naive naive 0hr SRR7457558  
## 3 GSM3243462 N03\_AM\_Naive naive 0hr SRR7457559  
## 4 GSM3243463 N04\_AM\_Naive naive 0hr SRR7457560  
## 5 GSM3243464 R01\_AM\_Allo2h post\_reperfusion 2hr SRR7457553  
## 6 GSM3243465 R02\_AM\_Allo2h post\_reperfusion 2hr SRR7457554  
## 7 GSM3243466 R03\_AM\_Allo2h post\_reperfusion 2hr SRR7457555  
## 8 GSM3243467 R04\_AM\_Allo2h post\_reperfusion 2hr SRR7457556  
## 9 GSM3243468 R05\_AM\_Allo24h post\_reperfusion 24hr SRR7457551  
## 10 GSM3243469 R06\_AM\_Allo24h post\_reperfusion 24hr SRR7457552  
## 11 GSM3243470 R07\_AM\_Allo24h post\_reperfusion 24hr SRR7457561  
## 12 GSM3243471 R08\_AM\_Allo24h post\_reperfusion 24hr SRR7457562

rownames(sampleinfo) <- sampleinfo$run   
  
dirs <- list.files("salmon\_quant/")  
quant\_files <- list.files("salmon\_quant/",pattern="quant.sf",recursive = TRUE,full.names = TRUE)  
#names(quant\_files) <- dirs  
quant\_files

## [1] "salmon\_quant//SRR7457551/quant.sf" "salmon\_quant//SRR7457552/quant.sf"  
## [3] "salmon\_quant//SRR7457553/quant.sf" "salmon\_quant//SRR7457554/quant.sf"  
## [5] "salmon\_quant//SRR7457555/quant.sf" "salmon\_quant//SRR7457556/quant.sf"  
## [7] "salmon\_quant//SRR7457557/quant.sf" "salmon\_quant//SRR7457558/quant.sf"  
## [9] "salmon\_quant//SRR7457559/quant.sf" "salmon\_quant//SRR7457560/quant.sf"  
## [11] "salmon\_quant//SRR7457561/quant.sf" "salmon\_quant//SRR7457562/quant.sf"

# tx2gene links transcript IDs to gene IDs for summarization  
tx2geneVariable <- read.csv("tx2gene.csv")  
  
txi <- tximport(files=quant\_files, type="salmon", tx2gene=tx2geneVariable)

## reading in files with read\_tsv

## 1 2 3 4 5 6 7 8 9 10 11 12   
## transcripts missing from tx2gene: 76  
## summarizing abundance  
## summarizing counts  
## summarizing length

View(txi) #view txi in a new tab  
  
#inspect salmon output (quant.sf files)  
quants <- read\_tsv(quant\_files[1])

## Rows: 132374 Columns: 5  
## -- Column specification --------------------------------------------------------  
## Delimiter: "\t"  
## chr (1): Name  
## dbl (4): Length, EffectiveLength, TPM, NumReads  
##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

head(quants)

## # A tibble: 6 x 5  
## Name Length EffectiveLength TPM NumReads  
## <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 ENSMUST00000193812 1070 756. 0 0  
## 2 ENSMUST00000082908 110 4 0 0  
## 3 ENSMUST00000162897 4153 3719. 0 0  
## 4 ENSMUST00000159265 2989 2604. 0.0174 2  
## 5 ENSMUST00000070533 3634 3376. 0 0  
## 6 ENSMUST00000192857 480 230 0 0

# spec(quants)  
   
 filter(quants, quants$TPM ==0) #filter data where TPM = 0

## # A tibble: 60,226 x 5  
## Name Length EffectiveLength TPM NumReads  
## <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 ENSMUST00000193812 1070 756. 0 0  
## 2 ENSMUST00000082908 110 4 0 0  
## 3 ENSMUST00000162897 4153 3719. 0 0  
## 4 ENSMUST00000070533 3634 3376. 0 0  
## 5 ENSMUST00000192857 480 230 0 0  
## 6 ENSMUST00000195335 2819 250 0 0  
## 7 ENSMUST00000192336 2233 250 0 0  
## 8 ENSMUST00000194099 2309 250 0 0  
## 9 ENSMUST00000161581 250 20 0 0  
## 10 ENSMUST00000192973 2057 250 0 0  
## # ... with 60,216 more rows

#print.data.frame(quants) #print quants, maximum output to print in console 200 lines  
   
tail(quants, n = 3)

## # A tibble: 3 x 5  
## Name Length EffectiveLength TPM NumReads  
## <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 ENSMUST00000179077 887 762. 17.0 574.   
## 2 ENSMUST00000189352 548 295. 1.45 18.9  
## 3 ENSMUST00000178569 1083 250 0 0

max(quants$TPM)

## [1] 63095.94

After asigning to quants variable the files

rpk <- quants$NumReads / quants$EffectiveLength  
scale\_factor <- sum(rpk) / 1e6  
tpm <- rpk / scale\_factor  
  
#define transcript mapping  
  
gtf\_file <- "Mus\_musculus.GRCm38.91.chr.gtf.gz"  
file.exists(gtf\_file)

## [1] TRUE

download.file("ftp://ftp.ensembl.org/pub/release-91/gtf/mus\_musculus/Mus\_musculus.GRCm38.91.chr.gtf.gz",destfile = gtf\_file) #gtf based on organism of interest  
  
# create a database of transcripts  
# Could take a few minutes to run the makeTxDbFromGFF command  
library(GenomicFeatures)

## Warning: package 'GenomicFeatures' was built under R version 4.0.4

## Loading required package: BiocGenerics

## Warning: package 'BiocGenerics' was built under R version 4.0.5

## Loading required package: parallel

##   
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':  
##   
## clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,  
## clusterExport, clusterMap, parApply, parCapply, parLapply,  
## parLapplyLB, parRapply, parSapply, parSapplyLB

## The following objects are masked from 'package:dplyr':  
##   
## combine, intersect, setdiff, union

## The following objects are masked from 'package:stats':  
##   
## IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':  
##   
## anyDuplicated, append, as.data.frame, basename, cbind, colnames,  
## dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,  
## grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,  
## order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,  
## rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,  
## union, unique, unsplit, which.max, which.min

## Loading required package: S4Vectors

## Loading required package: stats4

##   
## Attaching package: 'S4Vectors'

## The following objects are masked from 'package:dplyr':  
##   
## first, rename

## The following object is masked from 'package:base':  
##   
## expand.grid

## Loading required package: IRanges

##   
## Attaching package: 'IRanges'

## The following objects are masked from 'package:dplyr':  
##   
## collapse, desc, slice

## The following object is masked from 'package:grDevices':  
##   
## windows

## Loading required package: GenomeInfoDb

## Warning: package 'GenomeInfoDb' was built under R version 4.0.5

## Loading required package: GenomicRanges

## Loading required package: AnnotationDbi

## Loading required package: Biobase

## Welcome to Bioconductor  
##   
## Vignettes contain introductory material; view with  
## 'browseVignettes()'. To cite Bioconductor, see  
## 'citation("Biobase")', and for packages 'citation("pkgname")'.

##   
## Attaching package: 'AnnotationDbi'

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

txdb <- makeTxDbFromGFF(gtf\_file)

## Import genomic features from the file as a GRanges object ...

## OK

## Prepare the 'metadata' data frame ... OK  
## Make the TxDb object ...

## Warning in .get\_cds\_IDX(mcols0$type, mcols0$phase): The "phase" metadata column contains non-NA values for features of type  
## stop\_codon. This information was ignored.

## OK

#specify number of keys and columns  
  
keytypes(txdb)

## [1] "CDSID" "CDSNAME" "EXONID" "EXONNAME" "GENEID" "TXID" "TXNAME"

columns(txdb)

## [1] "CDSCHROM" "CDSEND" "CDSID" "CDSNAME" "CDSPHASE"   
## [6] "CDSSTART" "CDSSTRAND" "EXONCHROM" "EXONEND" "EXONID"   
## [11] "EXONNAME" "EXONRANK" "EXONSTART" "EXONSTRAND" "GENEID"   
## [16] "TXCHROM" "TXEND" "TXID" "TXNAME" "TXSTART"   
## [21] "TXSTRAND" "TXTYPE"

#get names for all transcripts - using keys function  
#compose query - using select function - this will return data frame  
  
k <- keys(txdb, keytype="TXNAME")  
tx\_map <- select(txdb, keys = k, columns="GENEID", keytype = "TXNAME")

## 'select()' returned 1:1 mapping between keys and columns

#visualise first rows of the transcript map  
head(tx\_map)

## TXNAME GENEID  
## 1 ENSMUST00000193812 ENSMUSG00000102693  
## 2 ENSMUST00000082908 ENSMUSG00000064842  
## 3 ENSMUST00000192857 ENSMUSG00000102851  
## 4 ENSMUST00000161581 ENSMUSG00000089699  
## 5 ENSMUST00000192183 ENSMUSG00000103147  
## 6 ENSMUST00000193244 ENSMUSG00000102348

#use tximport package  
  
library(tximport)  
tx2gene <- tx\_map  
# write.csv(tx2gene,file="tx2gene.csv",row.names = FALSE,quote=FALSE)  
txi <- tximport(quant\_files,type="salmon",tx2gene = tx2gene)  
  
table(tx\_map$TXNAME %in% quants$Name)  
  
tx2gene <- tx\_map  
txi <- tximport(quant\_files,type="salmon",tx2gene = tx2gene,ignoreTxVersion = TRUE)  
  
names(txi)  
  
head(txi$counts) #displays   
  
all(rownames(sampleinfo) == colnames(txi$counts))  
  
library(tidyr)  
  
quants <- separate(quants, Name, c("TXNAME","Number"),remove = FALSE)  
head(quants)  
#ERRORTEXT: Expected 2 pieces. Missing pieces filled with `NA` in 132374 rows [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].Error in left\_join(quants, tx\_map, by = "TXNAME") :   
# could not find function "left\_join"  
  
  
#commented code below  
# quants <- left\_join(quants, tx\_map, by="TXNAME")  
# head(quants)  
#   
# tx2gene <- dplyr:::select(quants, Name, GENEID)  
# head(tx2gene)  
#   
# any(is.na(tx2gene$GENEID))  
#   
# tx2gene <- filter(tx2gene, !is.na(GENEID))  
#   
# library(tximport)  
# txi <- tximport(quant\_files,type="salmon",tx2gene = tx2gene)

**CONCLUSIONS**

*References*