Using DESeq2 for differential gene expression analysis Part 1

2024-03-13

ACESO Genomics and TIDREC collaboration on the SCOPE project

Goal: - use the DESeq2 analysis to identify specific enriched or depleted genes - perform gene set enrichment analysis to find relevant pathways

1. Setup your environment

```
## Clean
rm(list = ls())
gc()
```

```
##          used (Mb) gc trigger (Mb) limit (Mb) max used (Mb)
## Ncells 478009 25.6    1031045 55.1         NA   669388 35.8
## Vcells 898957  6.9    8388608 64.0     256000  1851808 14.2
```

```
##
## PACKAGES
##

## load basic packages
suppressPackageStartupMessages(suppressWarnings({
  library(data.table);library(parallel);library(tidyr);library(tidyverse)}))


## For plotting
suppressPackageStartupMessages(suppressWarnings({
  library(ggpubr);library(ggbeeswarm);library(RColorBrewer);library(ggdendro);
  library(ggridges);library(ggrepel)}))


## For clustering
suppressPackageStartupMessages(suppressWarnings({library(pheatmap)}))

## For DESeq analysis
suppressPackageStartupMessages(suppressWarnings({library(DESeq2)}))
#library(sva) <- could be used for batch normalization

## For GSEA
suppressPackageStartupMessages(suppressWarnings({
  library(clusterProfiler);library(msigdb);library(msigdbr);
  library(enrichplot);library(ggupset)}))

##
## DIRECTORIES
```

```
##

TAB.DIR <- "/Users/a_PGenzor/Documents/GITHUB/ahjf_scope/results/tables/"
FIG.DIR <- "/Users/a_PGenzor/Documents/GITHUB/ahjf_scope/results/figures/"
#SES.DIR <- "/Users/a_PGenzor/Documents/GITHUB/ahjf_scope/sessions/"

##
## VERSION AND CONTROLS
##

aSeed="1003"
set.seed(aSeed)
version.date = "10MAR24"
```

2. Load the data

- Loading count data should be relatively simple since it should all be contained in the single matrix.
- Make sure that the column names in count matrix match the names in your metadata tables or that there is a way to calcualte them

```
##
## gene count table
count_table_path <- "/Users/a_PGenzor/Documents/GITHUB/ahjf_scope/data/from_kimkee/10MAR24/RNASeq_COVID_
cnt.dt <- fread(count_table_path)
cnt.dt[1:5,1:3]
```

```
##                            gene_id
## 1:       ENSG00000223764.2|LINC02593
## 2: ENSG00000272438.1|ENSG00000272438
## 3: ENSG00000230699.2|ENSG00000230699
## 4: ENSG00000241180.1|ENSG00000241180
## 5: ENSG00000288531.1|ENSG00000288531
##    02_D0_DKDL230010052-1A_HNG3NDSX7_L1.nonovel.gtf
## 1:                                              10
## 2:                                               0
## 3:                                               0
## 4:                                               0
## 5:                                               9
##    06_D0_DKDL230011035-1A_HC2HKDSX7_L3.nonovel.gtf
## 1:                                               7
## 2:                                               0
## 3:                                               9
## 4:                                               0
## 5:                                               0
```

```
## splitting complex names into pieces
colnames(cnt.dt)[1:5]
```

```
## [1] "gene_id"
## [2] "02_D0_DKDL230010052-1A_HNG3NDSX7_L1.nonovel.gtf"
## [3] "06_D0_DKDL230011035-1A_HC2HKDSX7_L3.nonovel.gtf"
## [4] "100_D0_DKDL230011026-1A_HC2HKDSX7_L3.nonovel.gtf"
## [5] "11_D0_DKDL230011071-1A_HC2HKDSX7_L4.nonovel.gtf"
```

```
unlist(tstrsplit(colnames(cnt.dt),split="_",keep = 1))
```

```
##   [1] "gene" "02"   "06"   "100"  "11"   "14"   "21"   "26"   "31"   "32"
##  [11] "34"   "35"   "36"   "37"   "39"   "03"   "43"   "45"   "46"   "47"
##  [21] "49"   "51"   "53"   "54"   "56"   "57"   "58"   "60"   "61"   "62"
##  [31] "67"   "68"   "69"   "70"   "72"   "74"   "75"   "77"   "78"   "79"
##  [41] "80"   "81"   "82"   "85"   "86"   "87"   "88"   "89"   "90"   "91"
##  [51] "92"   "93"   "94"   "95"   "98"   "02"   "03"   "06"   "11"   "14"
##  [61] "21"   "32"   "36"   "37"   "39"   "43"   "45"   "47"   "49"   "51"
##  [71] "54"   "56"   "57"   "58"   "60"   "62"   "67"   "75"   "77"   "78"
##  [81] "79"   "80"   "81"   "82"   "85"   "86"   "87"   "88"   "89"   "90"
##  [91] "91"   "92"   "93"   "94"   "98"   "100"  "32"   "49"   "62"   "92"
## [101] "93"   "100"  "32"   "49"   "62"   "92"   "93"
```

```
unlist(tstrsplit(colnames(cnt.dt),split="_",keep = 2))
```

```
##   [1] "id"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"
##  [13] "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"
##  [25] "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"
##  [37] "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"
##  [49] "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D0"  "D28" "D28" "D28" "D28" "D28"
##  [61] "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D28"
##  [73] "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D28"
##  [85] "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D28" "D3"
##  [97] "D3"  "D3"  "D3"  "D3"  "D3"  "D7"  "D7"  "D7"  "D7"  "D7"  "D7"
```

```
## make new colnames
cnt_new_col_names <- paste(unlist(tstrsplit(colnames(cnt.dt),split="_",keep = 2)),
      unlist(tstrsplit(colnames(cnt.dt),split="_",keep = 1)),
      sep = "_")
cnt_new_col_names <-gsub("id_gene","gene_id",cnt_new_col_names)
colnames(cnt.dt) <- cnt_new_col_names
cnt.dt[1:5,1:5] ## <- ready to use
```

```
##                              gene_id D0_02 D0_06 D0_100 D0_11
## 1:       ENSG00000223764.2|LINC02593    10     7      7    14
## 2: ENSG00000272438.1|ENSG00000272438     0     0      0     0
## 3: ENSG00000230699.2|ENSG00000230699     0     9     17     7
## 4: ENSG00000241180.1|ENSG00000241180     0     0      0     0
## 5: ENSG00000288531.1|ENSG00000288531     9     0     21    30
```

```
##
## metadata
metadata_path <- "/Users/a_PGenzor/Documents/GITHUB/ahjf_scope/data/from_kimkee/10MAR24/RNASeq_COVID/Me
meta.dt <- fread(metadata_path)

meta_cols_to_use <- c("IGU_Code","sex","pathogen","disease","time")
meta.clean.dt <- meta.dt[,.SD,.SDcols = meta_cols_to_use]
meta.clean.dt[,"subject":=tstrsplit(IGU_Code,split="_",keep = 1)]
meta.clean.dt[,"seq_id":= paste(time,subject,sep = "_")]
meta.clean.dt
```

```
##      IGU_Code    sex   pathogen disease time subject seq_id
##   1:    02_D0 Female SARS-CoV-2 COVID19   D0      02  D0_02
##   2:   02_D28 Female SARS-CoV-2 COVID19  D28      02 D28_02
##   3:    03_D0 Female SARS-CoV-2 COVID19   D0      03  D0_03
##   4:   03_D28 Female SARS-CoV-2 COVID19  D28      03 D28_03
##   5:    06_D0 Female SARS-CoV-2 COVID19   D0      06  D0_06
```

```
##  ---
## 102:    98_D0   Male SARS-CoV-2 COVID19   D0      98  D0_98
## 103:   98_D28   Male SARS-CoV-2 COVID19  D28      98 D28_98
## 104:   100_D0   Male SARS-CoV-2 COVID19   D0     100 D0_100
## 105:   100_D3   Male SARS-CoV-2 COVID19   D3     100 D3_100
## 106:   100_D7   Male SARS-CoV-2 COVID19   D7     100 D7_100
```

3. Format the data

Once the data is loaded in a clean way, make sure that you format the data types to ones that can be used by DESeq - eg. matrix instead of table and so on - This is a good place to filter your data to remove uninformative genes - Here you will also be combining the sample information with the metadata so that they correspond to each other during analysis - NOTE: metadata and data alignment is a key for analysis.

```
##
## Filter raw counts
##

cnt.dt[1:5,1:5]
```

```
##                              gene_id D0_02 D0_06 D0_100 D0_11
## 1:       ENSG00000223764.2|LINC02593    10     7      7    14
## 2: ENSG00000272438.1|ENSG00000272438     0     0      0     0
## 3: ENSG00000230699.2|ENSG00000230699     0     9     17     7
## 4: ENSG00000241180.1|ENSG00000241180     0     0      0     0
## 5: ENSG00000288531.1|ENSG00000288531     9     0     21    30
```

```
## summarize raw counts
cnt.dt.sumarized <- cnt.dt[,list(max=max(.SD),
                                 min=min(.SD),
                                 mean=mean(unlist(.SD))), by=gene_id]
cnt.dt.sumarized
```

```
##                                  gene_id max min        mean
##     1:       ENSG00000223764.2|LINC02593  38   0  6.40566038
##     2: ENSG00000272438.1|ENSG00000272438   2   0  0.01886792
##     3: ENSG00000230699.2|ENSG00000230699  49   0  7.40566038
##     4: ENSG00000241180.1|ENSG00000241180   0   0  0.00000000
##     5: ENSG00000288531.1|ENSG00000288531  95   0 12.57547170
##    ---
## 61902: ENSG00000275249.1|ENSG00000275249  16   0  2.42452830
## 61903: ENSG00000274792.1|ENSG00000274792  14   0  1.30188679
## 61904: ENSG00000278510.1|ENSG00000278510   6   0  0.35849057
## 61905: ENSG00000277196.4|ENSG00000277196  19   0  1.67924528
## 61906:              ENSG00000277374.1|U1   4   0  0.54716981
```

```
## get gene names that have sufficient expression
## NOTE: this parameter is subjective and you can/should play with your cutoff value
## NOTE: sometimes, it makes more sense to not use min if you think some genes are on/off in subjects
## NOTE: counts are not like TPM, 10 counts per gene may still mean gene is off

cnt.dt.sumarized[mean > 0]
```

```
##                                  gene_id max min        mean
##     1:       ENSG00000223764.2|LINC02593  38   0  6.40566038
##     2: ENSG00000272438.1|ENSG00000272438   2   0  0.01886792
##     3: ENSG00000230699.2|ENSG00000230699  49   0  7.40566038
```

```
##     4: ENSG00000288531.1|ENSG00000288531  95   0 12.57547170
##     5:          ENSG00000230368.2|FAM41C  73   0 14.16981132
##    ---
## 61696: ENSG00000275249.1|ENSG00000275249  16   0  2.42452830
## 61697: ENSG00000274792.1|ENSG00000274792  14   0  1.30188679
## 61698: ENSG00000278510.1|ENSG00000278510   6   0  0.35849057
## 61699: ENSG00000277196.4|ENSG00000277196  19   0  1.67924528
## 61700:              ENSG00000277374.1|U1   4   0  0.54716981
```

```
cnt.dt.sumarized[mean > 10]
```

```
##                                 gene_id  max min       mean
##     1: ENSG00000288531.1|ENSG00000288531   95   0   12.57547
##     2:          ENSG00000230368.2|FAM41C   73   0   14.16981
##     3:         ENSG00000187961.15|KLHL17  547  28  200.42453
##     4:        ENSG00000187583.11|PLEKHN1   80   0   33.10377
##     5:          ENSG00000188976.11|NOC2L  836 102  411.50000
##    ---
## 24737: ENSG00000267793.1|ENSG00000267793   75   0   14.43396
## 24738: ENSG00000260197.1|ENSG00000260197  412   0   93.06604
## 24739:          ENSG00000012817.16|KDM5D 7361   0 1614.75472
## 24740: ENSG00000288049.1|ENSG00000288049  170   0   35.42453
## 24741:         ENSG00000198692.10|EIF1AY 1259   0  298.06604
```

```
cnt.dt.sumarized[mean > 50]
```

```
##                                 gene_id   max min       mean
##     1:         ENSG00000187961.15|KLHL17   547  28  200.42453
##     2:          ENSG00000188976.11|NOC2L   836 102  411.50000
##     3: ENSG00000272512.1|ENSG00000272512  1622   0   73.54717
##     4:           ENSG00000188290.11|HES4  2402   0  190.87736
##     5:          ENSG00000187608.10|ISG15 34685  51 1960.95283
##    ---
## 16065:         ENSG00000215580.12|BCORP1   913   0  182.10377
## 16066:         ENSG00000131002.14|TXLNGY  6766   0 1647.10377
## 16067: ENSG00000260197.1|ENSG00000260197   412   0   93.06604
## 16068:          ENSG00000012817.16|KDM5D  7361   0 1614.75472
## 16069:         ENSG00000198692.10|EIF1AY  1259   0  298.06604
```

```
gene_ids_to_include <- cnt.dt.sumarized[mean > 50][["gene_id"]]

## filter data
cnt.filtered.dt <- cnt.dt[gene_id %in% gene_ids_to_include]
cnt.filtered.dt[1:5,1:5]
```

```
##                             gene_id D0_02 D0_06 D0_100 D0_11
## 1:         ENSG00000187961.15|KLHL17   152   130     49    39
## 2:          ENSG00000188976.11|NOC2L   427   380    111   116
## 3: ENSG00000272512.1|ENSG00000272512   362    46      6    12
## 4:           ENSG00000188290.11|HES4  1337   404      3    20
## 5:          ENSG00000187608.10|ISG15 11707  4699    105   150
```

```
##
## Make sample information table
##

## create a sample information table from cnt table
```

```
## NOTE: this will make sure you will always have the right samples present
si.dt <- data.table("seq_id"=colnames(cnt.dt)[-1])
si.dt[,"subject":=tstrsplit(seq_id,split="_",keep = 2)]
si.dt[,"time":=tstrsplit(seq_id,split="_",keep = 1)]
si.dt
```

```
##      seq_id subject time
##   1:  D0_02      02   D0
##   2:  D0_06      06   D0
##   3: D0_100     100   D0
##   4:  D0_11      11   D0
##   5:  D0_14      14   D0
##  ---
## 102:  D7_32      32   D7
## 103:  D7_49      49   D7
## 104:  D7_62      62   D7
## 105:  D7_92      92   D7
## 106:  D7_93      93   D7
```

```
##
## Load metadata and add to the sample information
##

# peak at ready metadata
meta.clean.dt
```

```
##      IGU_Code    sex   pathogen disease time subject seq_id
##   1:    02_D0 Female SARS-CoV-2 COVID19   D0      02  D0_02
##   2:   02_D28 Female SARS-CoV-2 COVID19  D28      02 D28_02
##   3:    03_D0 Female SARS-CoV-2 COVID19   D0      03  D0_03
##   4:   03_D28 Female SARS-CoV-2 COVID19  D28      03 D28_03
##   5:    06_D0 Female SARS-CoV-2 COVID19   D0      06  D0_06
##  ---
## 102:    98_D0   Male SARS-CoV-2 COVID19   D0      98  D0_98
## 103:   98_D28   Male SARS-CoV-2 COVID19  D28      98 D28_98
## 104:   100_D0   Male SARS-CoV-2 COVID19   D0     100 D0_100
## 105:   100_D3   Male SARS-CoV-2 COVID19   D3     100 D3_100
## 106:   100_D7   Male SARS-CoV-2 COVID19   D7     100 D7_100
```

```
# add to si.dt to make a master table (mt)
si.dt
```

```
##      seq_id subject time
##   1:  D0_02      02   D0
##   2:  D0_06      06   D0
##   3: D0_100     100   D0
##   4:  D0_11      11   D0
##   5:  D0_14      14   D0
##  ---
## 102:  D7_32      32   D7
## 103:  D7_49      49   D7
## 104:  D7_62      62   D7
## 105:  D7_92      92   D7
## 106:  D7_93      93   D7
```

```
si.mt.dt <- meta.clean.dt[si.dt,on=.(seq_id=seq_id,time=time,subject=subject)]
si.mt.dt[1:5,]
```

```
##    IGU_Code    sex   pathogen disease time subject seq_id
## 1:    02_D0 Female SARS-CoV-2 COVID19   D0      02  D0_02
## 2:    06_D0 Female SARS-CoV-2 COVID19   D0      06  D0_06
## 3:   100_D0   Male SARS-CoV-2 COVID19   D0     100 D0_100
## 4:    11_D0 Female SARS-CoV-2 COVID19   D0      11  D0_11
## 5:    14_D0   Male SARS-CoV-2 COVID19   D0      14  D0_14
```

```
## filter table to keep only comparison samples
si.mt.comp.dt <- si.mt.dt[time %in% c("D0","D28")]
si.mt.comp.dt[,.N,by=list(disease, time)]
```

```
##        disease time  N
## 1:     COVID19   D0 30
## 2: Non-COVID19   D0 24
## 3:     COVID19  D28 28
## 4: Non-COVID19  D28 12
```

```
si.mt.comp.dt[,.N,by=time]
```

```
##    time  N
## 1:   D0 54
## 2:  D28 40
```

```
##
## Filter count table to keep comparison columns
##

# present columns and their format
colnames(cnt.filtered.dt)
```

```
##   [1] "gene_id" "D0_02"   "D0_06"   "D0_100"  "D0_11"   "D0_14"   "D0_21"
##   [8] "D0_26"   "D0_31"   "D0_32"   "D0_34"   "D0_35"   "D0_36"   "D0_37"
##  [15] "D0_39"   "D0_03"   "D0_43"   "D0_45"   "D0_46"   "D0_47"   "D0_49"
##  [22] "D0_51"   "D0_53"   "D0_54"   "D0_56"   "D0_57"   "D0_58"   "D0_60"
##  [29] "D0_61"   "D0_62"   "D0_67"   "D0_68"   "D0_69"   "D0_70"   "D0_72"
##  [36] "D0_74"   "D0_75"   "D0_77"   "D0_78"   "D0_79"   "D0_80"   "D0_81"
##  [43] "D0_82"   "D0_85"   "D0_86"   "D0_87"   "D0_88"   "D0_89"   "D0_90"
##  [50] "D0_91"   "D0_92"   "D0_93"   "D0_94"   "D0_95"   "D0_98"   "D28_02"
##  [57] "D28_03"  "D28_06"  "D28_11"  "D28_14"  "D28_21"  "D28_32"  "D28_36"
##  [64] "D28_37"  "D28_39"  "D28_43"  "D28_45"  "D28_47"  "D28_49"  "D28_51"
##  [71] "D28_54"  "D28_56"  "D28_57"  "D28_58"  "D28_60"  "D28_62"  "D28_67"
##  [78] "D28_75"  "D28_77"  "D28_78"  "D28_79"  "D28_80"  "D28_81"  "D28_82"
##  [85] "D28_85"  "D28_86"  "D28_87"  "D28_88"  "D28_89"  "D28_90"  "D28_91"
##  [92] "D28_92"  "D28_93"  "D28_94"  "D28_98"  "D3_100"  "D3_32"   "D3_49"
##  [99] "D3_62"   "D3_92"   "D3_93"   "D7_100"  "D7_32"   "D7_49"   "D7_62"
## [106] "D7_92"   "D7_93"
```

```
# wanted columns and their format matching
si.mt.comp.dt[1:5,]
```

```
##    IGU_Code    sex   pathogen disease time subject seq_id
## 1:    02_D0 Female SARS-CoV-2 COVID19   D0      02  D0_02
## 2:    06_D0 Female SARS-CoV-2 COVID19   D0      06  D0_06
## 3:   100_D0   Male SARS-CoV-2 COVID19   D0     100 D0_100
```

```
## 4:    11_D0 Female SARS-CoV-2 COVID19   D0      11  D0_11
## 5:    14_D0   Male SARS-CoV-2 COVID19   D0      14  D0_14
```

```
wanted_comp_columns <- si.mt.comp.dt[["seq_id"]]
wanted_comp_columns
```

```
##  [1] "D0_02"  "D0_06"  "D0_100" "D0_11"  "D0_14"  "D0_21"  "D0_26"  "D0_31"
##  [9] "D0_32"  "D0_34"  "D0_35"  "D0_36"  "D0_37"  "D0_39"  "D0_03"  "D0_43"
## [17] "D0_45"  "D0_46"  "D0_47"  "D0_49"  "D0_51"  "D0_53"  "D0_54"  "D0_56"
## [25] "D0_57"  "D0_58"  "D0_60"  "D0_61"  "D0_62"  "D0_67"  "D0_68"  "D0_69"
## [33] "D0_70"  "D0_72"  "D0_74"  "D0_75"  "D0_77"  "D0_78"  "D0_79"  "D0_80"
## [41] "D0_81"  "D0_82"  "D0_85"  "D0_86"  "D0_87"  "D0_88"  "D0_89"  "D0_90"
## [49] "D0_91"  "D0_92"  "D0_93"  "D0_94"  "D0_95"  "D0_98"  "D28_02" "D28_03"
## [57] "D28_06" "D28_11" "D28_14" "D28_21" "D28_32" "D28_36" "D28_37" "D28_39"
## [65] "D28_43" "D28_45" "D28_47" "D28_49" "D28_51" "D28_54" "D28_56" "D28_57"
## [73] "D28_58" "D28_60" "D28_62" "D28_67" "D28_75" "D28_77" "D28_78" "D28_79"
## [81] "D28_80" "D28_81" "D28_82" "D28_85" "D28_86" "D28_87" "D28_88" "D28_89"
## [89] "D28_90" "D28_91" "D28_92" "D28_93" "D28_94" "D28_98"
```

```
## filter raw counts to keep the same samples
cnt.filtered.comp.dt <- cnt.filtered.dt[,.SD,.SDcols = c("gene_id",wanted_comp_columns)]
cnt.filtered.comp.dt[1:5,1:5]
```

```
##                              gene_id D0_02 D0_06 D0_100 D0_11
## 1:         ENSG00000187961.15|KLHL17   152   130     49    39
## 2:          ENSG00000188976.11|NOC2L   427   380    111   116
## 3: ENSG00000272512.1|ENSG00000272512   362    46      6    12
## 4:           ENSG00000188290.11|HES4  1337   404      3    20
## 5:          ENSG00000187608.10|ISG15 11707  4699    105   150
```

```
##
## Format into right types
##date

## counts need to be a matrix where rownames are gene_id
cnt.comp.mat <- as.matrix(x = cnt.filtered.comp.dt, rownames = "gene_id")

## sample information can remain a data table
si.mt.comp.dt[1:5,]
```

```
##    IGU_Code    sex   pathogen disease time subject seq_id
## 1:    02_D0 Female SARS-CoV-2 COVID19   D0      02  D0_02
## 2:    06_D0 Female SARS-CoV-2 COVID19   D0      06  D0_06
## 3:   100_D0   Male SARS-CoV-2 COVID19   D0     100 D0_100
## 4:    11_D0 Female SARS-CoV-2 COVID19   D0      11  D0_11
## 5:    14_D0   Male SARS-CoV-2 COVID19   D0      14  D0_14
```

4. Run DESeq2 analysis

- Once the data has been prepared, the DESeq package can be employed and comparative analysis performed. The analysis consists of three simple steps:
  1. Create a DESeq object using the raw counts and metadata from previous section. And specifying the comparison MODEL.
  2. Running the DESeq command.
  3. Retrieval of the result tables for plotting and analysis.

```
##
## Create a DESeq object
##

## Data
#cnt.comp.mat[1:5,1:5]
#si.mt.comp.dt[1:5]

## load data into deseq object
dds <- DESeqDataSetFromMatrix(countData = cnt.comp.mat,
                              colData = si.mt.comp.dt,
                              design = ~time)
```

```
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

```
## add condition to the modeling
dds.sex <- DESeqDataSetFromMatrix(countData = cnt.comp.mat,
                              colData = si.mt.comp.dt,
                              design = ~time+sex)
```

```
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

```
##
## Run DESeq Analysis
##

## two modes - with and without sex consideration
dds <- DESeq(dds)
```

```
## estimating size factors
```

```
## estimating dispersions
```

```
## gene-wise dispersion estimates
```

```
## mean-dispersion relationship
```

```
## final dispersion estimates
```

```
## fitting model and testing
```

```
## -- replacing outliers and refitting for 403 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)
```

```
## estimating dispersions
```

```
## fitting model and testing
```

```
dds.sex <- DESeq(dds.sex)
```

```
## estimating size factors
```

```
## estimating dispersions
```

```
## gene-wise dispersion estimates
```

```
## mean-dispersion relationship
```

```
## final dispersion estimates
```

```
## fitting model and testing
```

```
## -- replacing outliers and refitting for 273 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)
```

```
## estimating dispersions
```

```
## fitting model and testing
```

```
##
## View and retrieve the results
##

## Look at results without sex consideration
resultsNames(dds)
```

```
## [1] "Intercept"      "time_D28_vs_D0"
```

```
res <- results(object = dds, name = "time_D28_vs_D0", alpha = 0.05)
summary(res)
```

```
##
## out of 16068 with nonzero total read count
## adjusted p-value < 0.05
## LFC > 0 (up)       : 931, 5.8%
## LFC < 0 (down)     : 2038, 13%
## outliers [1]       : 0, 0%
## low counts [2]     : 1, 0.0062%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
## Look at the results with sex consideration
resultsNames(dds.sex)
```

```
## [1] "Intercept"          "time_D28_vs_D0"     "sex_Male_vs_Female"
```

```
res.sex <- results(object = dds.sex, name = "time_D28_vs_D0", alpha = 0.05)
summary(res.sex)
```

```
##
## out of 16069 with nonzero total read count
## adjusted p-value < 0.05
## LFC > 0 (up)       : 934, 5.8%
## LFC < 0 (down)     : 1926, 12%
## outliers [1]       : 0, 0%
## low counts [2]     : 0, 0%
## (mean count < 2)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
## export a table of results for each
res.dt <- as.data.table(results(object = dds, name = "time_D28_vs_D0", alpha = 0.05), keep.rownames=TRU
```

```
## Warning in .local(x, row.names, optional, ...): Arguments in '...' ignored
```

```
colnames(res.dt) <- gsub("rn","gene_id",colnames(res.dt))

res.sex.dt <- as.data.table(results(object = dds.sex, name = "time_D28_vs_D0", alpha = 0.05), keep.rowna
```

```
## Warning in .local(x, row.names, optional, ...): Arguments in '...' ignored
```

```
colnames(res.sex.dt) <- gsub("rn","gene_id",colnames(res.sex.dt))

## RESULT TABLES
res.dt <- res.dt[order(padj,log2FoldChange)]
res.dt
```

```
##                                  gene_id  baseMean log2FoldChange      lfcSE
##     1:        ENSG00000108387.16|SEPTIN4  312.7207  -2.802464e+00 0.27604306
##     2:          ENSG00000165949.13|IFI27 1174.0043  -4.364281e+00 0.44337569
##     3:          ENSG00000184979.11|USP18  830.0480  -3.040946e+00 0.32843781
##     4:        ENSG00000196141.14|SPATS2L  995.0307  -2.607314e+00 0.28203196
##     5:          ENSG00000187608.10|ISG15 2014.3514  -3.131625e+00 0.35052613
##    ---
## 16065:          ENSG00000155903.14|RASA2 3779.8496  -2.503228e-05 0.04456558
## 16066:         ENSG00000257246.2|PHB1P19  104.8990  -4.350574e-05 0.11009555
## 16067:       ENSG00000286219.2|NOTCH2NLC 4280.7756  -9.717332e-06 0.08605497
## 16068:           ENSG00000165195.16|PIGA  497.5808   3.578511e-06 0.05084592
## 16069: ENSG00000269693.1|ENSG00000269693    0.0000   0.000000e+00 0.00000000
##                  stat       pvalue         padj
##     1: -1.015227e+01 3.237387e-24 5.201833e-20
##     2: -9.843303e+00 7.326482e-23 5.886095e-19
##     3: -9.258817e+00 2.067101e-20 9.472012e-17
##     4: -9.244747e+00 2.357982e-20 9.472012e-17
##     5: -8.934070e+00 4.106169e-19 1.319558e-15
##    ---
## 16065: -5.616955e-04 9.995518e-01 9.997385e-01
## 16066: -3.951635e-04 9.996847e-01 9.998092e-01
## 16067: -1.129201e-04 9.999099e-01 9.999438e-01
## 16068:  7.037950e-05 9.999438e-01 9.999438e-01
## 16069:  0.000000e+00 1.000000e+00           NA
```

```
res.sex.dt <- res.sex.dt[order(padj,log2FoldChange)]
res.sex.dt
```

```
##                                  gene_id    baseMean log2FoldChange      lfcSE
##     1:          ENSG00000165949.13|IFI27  1174.00428  -4.321991e+00 0.44438975
##     2:          ENSG00000184979.11|USP18   830.04803  -2.860923e+00 0.31575707
##     3:          ENSG00000187608.10|ISG15  2014.35142  -3.084138e+00 0.34314627
##     4:        ENSG00000142089.17|IFITM3 13186.71862  -2.023653e+00 0.23732131
##     5:          ENSG00000161133.18|USP41    72.44594  -2.609214e+00 0.30872184
##    ---
## 16065:            ENSG00000172336.5|POP7    85.59053  -4.234665e-05 0.09927925
## 16066:          ENSG00000063601.17|MTMR1  1324.63352  -9.389151e-06 0.04372301
## 16067: ENSG00000276136.1|ENSG00000276136   452.38103   2.133293e-05 0.10818643
## 16068:         ENSG00000196417.13|ZNF765   513.27454   2.447131e-05 0.08010933
## 16069:            ENSG00000134548.11|SPX    60.77116   1.316112e-04 0.19918038
##                  stat       pvalue         padj
##     1: -9.7256768058 2.343417e-22 3.765638e-18
##     2: -9.0605202753 1.298298e-19 1.043118e-15
##     3: -8.9878243218 2.521727e-19 1.350721e-15
##     4: -8.5270582503 1.501163e-17 6.030547e-14
##     5: -8.4516667939 2.871732e-17 8.620821e-14
##    ---
## 16065: -0.0004265408 9.996597e-01 9.998427e-01
```

```
## 16066: -0.0002147417 9.998287e-01 9.998427e-01
## 16067:  0.0001971868 9.998427e-01 9.998427e-01
## 16068:  0.0003054739 9.997563e-01 9.998427e-01
## 16069:  0.0006607637 9.994728e-01 9.998427e-01
```

5. DESeq result plots
6. GSEA analysis
7. GSEA plots