

# Differential Gene Expression and Pathway Analysis

2024-03-25

## BACKGROUND

ACESO Genomics and TIDREC collaboration on the SCOPE project. The example code below is meant to illustrate the process of standard gene expression and pathway analysis to the enable RNA-Seq analytical capabilities going forward.

## CONTENT:

- PART1 - Loading data, preparation and running of DESeq2 analysis
- PART2 - Plotting data and Gene Set Enrichment Analysis
- PART3 - Exploring results plots
- PART4 - Table1 and metadata summaries

## PART1

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 478458 25.6   1032328 55.2          NA   669388 35.8
## Vcells 903490  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(ggribes);library(ggplot2)}))
```

```

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

## 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 calculate 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_DO_DKDL230010052-1A_HNG3NDSX7_L1.nonovel.gtf
## 1:          10
## 2:          0
## 3:          0
## 4:          0
## 5:          9
##    06_DO_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" "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 DO_02 DO_06 DO_100 DO_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=TRUE)

## 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.rownames=TRUE)

## 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: ENSG000000276136.1|ENSG000000276136   452.38103  2.133293e-05 0.10818643
## 16068:          ENSG000000196417.13|ZNF765   513.27454  2.447131e-05 0.08010933
## 16069:          ENSG000000134548.11|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
```

## PART2

Goal: - Explore ways of plotting results from DESeq2 analysis - Use the results in Gene Set Enrichment Analysis

### 1. Plot PCA of the results

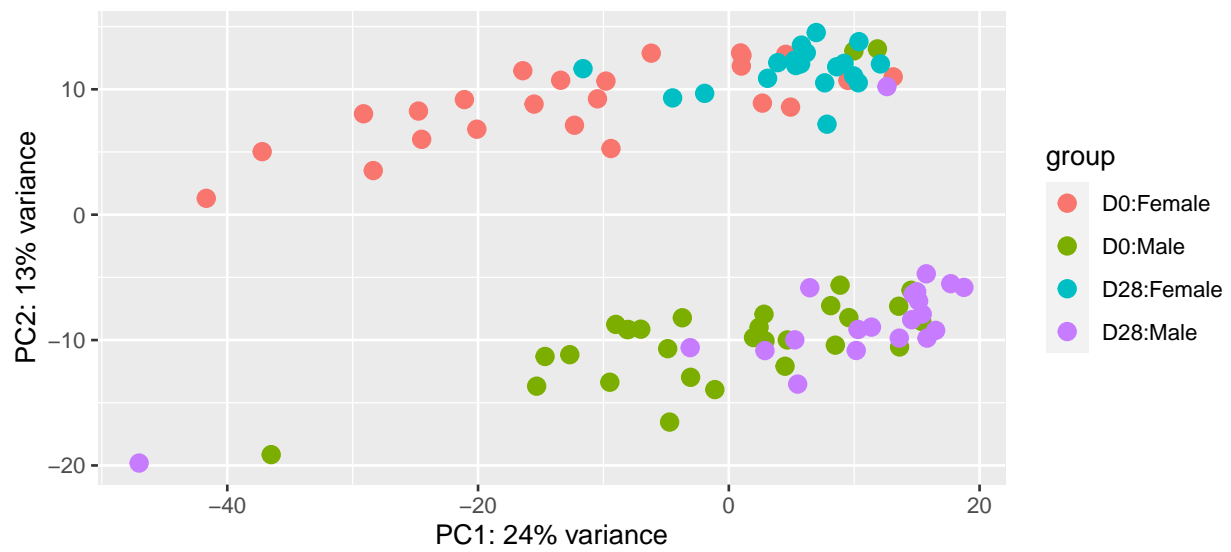
- PCA plot is one of the typical plots to evaluate whether there are any patterns in your data
- First, the data is normalized

```
# Input data
#dds.sex
#si.mt.comp.dt

## Stabilize the data using variance stabilizing transformation
vsd.sex <- vst(object = dds.sex)
vsd.sex

## class: DESeqTransform
## dim: 16069 94
## metadata(1): version
## assays(1): ''
## rownames(16069): ENSG000000187961.15|KLHL17 ENSG000000188976.11|NOC2L ...
## ENSG00000012817.16|KDM5D ENSG000000198692.10|EIF1AY
## rowData names(27): baseMean baseVar ... replace dispFit
## colnames(94): D0_02 D0_06 ... D28_94 D28_98
## colData names(9): IGU_Code sex ... sizeFactor replaceable

## Use native DESeq PCA plotting capabilities
#?plotPCA
plotPCA(object = vsd.sex, intgroup = c("time","sex"))
```



```
## Modify the plot by saving into object and adjusting the ggplot parameters within it  
## -> https://coolours.co/
```

```

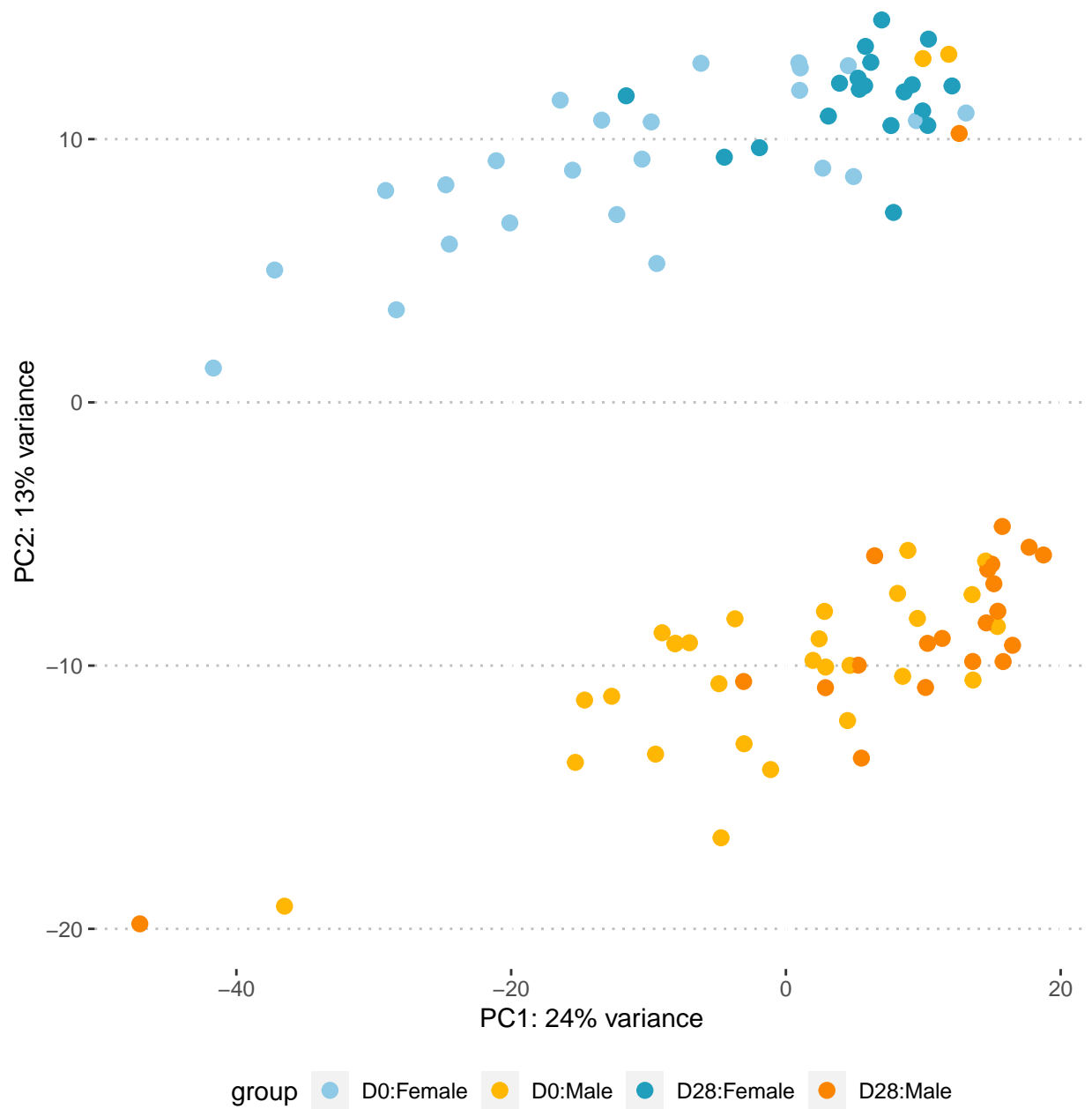
## -> above is a great website for color choosing

## number of colors should match the number of conditions
four_colors <- c("#8ecae6", "#ffb703", "#219ebc", "#fb8500")

## Use the ggplot capabilities to make nicer custom plot
pca.plot <- plotPCA(object = vsd.sex, intgroup = c("time", "sex"))
pca.plot + theme_pubclean() +
  geom_point(size = 2) +
  ggtitle(paste0("PCA using sex-adjusted DESeq2 results\n",
                 "NOTE: there is a sex difference")) +
  scale_colour_manual(values = four_colors) +
  theme(aspect.ratio = 1,
        legend.position = "bottom")

```

PCA using sex-adjusted DESeq2 results  
 NOTE: there is a sex difference



```
##
## PCA on dds (no sex adjustment)
## -> The PCA looks the same, however, the resulting genes are different due to different model [~time+sex]
```

```
## Stabilize the data
#vsd <- vst(object = dds)

## Plot similar plot using custom ggplot
#pca.plot <- plotPCA(object = vsd, intgroup = c("time","sex"))
#pca.plot + theme_pubclean() +
# geom_point(size = 2) +
# ggtitle(paste0("PCA using NOT-adjusted for sex DESeq2 results\n",
#               "NOTE: there is a sex difference")) +
# scale_colour_manual(values = four_colors) +
# theme(aspect.ratio = 1,
#       legend.position = "right")
```

## 2. Plot Boxplot of the normalized counts

- It can be an important control or piece of data to look at the distribution of expression of a particular gene in your data
- To do so we extract the normalized counts (or use TPM data) and use boxplot that summarized number of important statistics including median, quantiles and outliers

```
## Input
#dds.sex
#si.mt.comp.dt

## Extract normalized count data
# ?counts <- function that extracts normalized data from dds object
ncount.dt <- as.data.table(counts(dds.sex, normalized=TRUE), keep.rownames = TRUE)
colnames(ncount.dt) <- gsub("rn", "gene_id", colnames(ncount.dt))
ncount.dt[1:5,1:5]
```

```
##           gene_id      DO_02      DO_06      DO_100      DO_11
## 1: ENSG00000187961.15|KLHL17 148.5576 157.58938 122.268608 147.48129
## 2: ENSG00000188976.11|NOC2L 417.3295 460.64587 276.975826 438.66230
## 3: ENSG00000272512.1|ENSG00000272512 353.8016 55.76239 14.971666 45.37886
## 4: ENSG00000188290.11|HES4 1306.7203 489.73929 7.485833 75.63143
## 5: ENSG00000187608.10|ISG15 11441.8660 5696.24984 262.004160 567.23573
```

```
## Re-arrange the table and get gene names
ncount.dtm <- melt.data.table(data = ncount.dt, id.vars = "gene_id",
                             variable.name = "subject",
                             value.name = "ncount")
ncount.dtm[, "gene_name" := tstrsplit(gene_id, split="\\|", keep = 2)]
ncount.dtm
```

```
##           gene_id subject      ncount      gene_name
## 1: ENSG00000187961.15|KLHL17 DO_02 148.5576      KLHL17
## 2: ENSG00000188976.11|NOC2L DO_02 417.3295      NOC2L
## 3: ENSG00000272512.1|ENSG00000272512 DO_02 353.8016 ENSG00000272512
## 4: ENSG00000188290.11|HES4 DO_02 1306.7203      HES4
## 5: ENSG00000187608.10|ISG15 DO_02 11441.8660      ISG15
## ---
## 1510482: ENSG00000215580.12|BCORP1 D28_98 486.9312      BCORP1
```



```
## 1510483:      ENSG00000131002.14|TXLNGY  D28_98  3584.4561      TXLNGY
## 1510484: ENSG00000260197.1|ENSG00000260197  D28_98   181.7779 ENSG00000260197
## 1510485:      ENSG00000012817.16|KDM5D  D28_98  3306.3140      KDM5D
## 1510486:      ENSG00000198692.10|EIF1AY  D28_98   617.6069      EIF1AY
```

```
## Combine the normalized counts with metadata
```

```
ncount.dtm <- ncount.dtm[si.mt.comp.dt,on=.(subject=seq_id)]
ncount.dtm[1:5]
```

```
##           gene_id subject      ncount      gene_name
## 1:      ENSG00000187961.15|KLHL17  D0_02   148.5576      KLHL17
## 2:      ENSG00000188976.11|NOC2L   D0_02   417.3295      NOC2L
## 3: ENSG00000272512.1|ENSG00000272512  D0_02   353.8016 ENSG00000272512
## 4:      ENSG00000188290.11|HES4    D0_02  1306.7203      HES4
## 5:      ENSG00000187608.10|ISG15   D0_02 11441.8660      ISG15
##   IGU_Code  sex  pathogen disease time i.subject
## 1:    02_D0 Female SARS-CoV-2 COVID19  D0         02
## 2:    02_D0 Female SARS-CoV-2 COVID19  D0         02
## 3:    02_D0 Female SARS-CoV-2 COVID19  D0         02
## 4:    02_D0 Female SARS-CoV-2 COVID19  D0         02
## 5:    02_D0 Female SARS-CoV-2 COVID19  D0         02
```

```
## select genes of interest
```

```
goi <- c("IFI27","CCL2","CD177","XIST","CXCL10")
```

```
## subset the count table
```

```
ncount.goi.dtm <- ncount.dtm[gene_name %in% goi]
ncount.goi.dtm
```

```
##           gene_id subject      ncount gene_name IGU_Code  sex
## 1: ENSG00000169245.6|CXCL10  D0_02   4062.85445  CXCL10   02_D0 Female
## 2: ENSG00000165949.13|IFI27  D0_02    581.52475  IFI27   02_D0 Female
## 3: ENSG00000108691.10|CCL2   D0_02   1400.54617   CCL2   02_D0 Female
## 4: ENSG00000204936.10|CD177  D0_02    78.18820   CD177   02_D0 Female
## 5: ENSG00000229807.13|XIST   D0_02 142701.28745   XIST   02_D0 Female
## ---
## 466: ENSG00000169245.6|CXCL10 D28_98    73.00318  CXCL10   98_D28  Male
## 467: ENSG00000165949.13|IFI27 D28_98     8.03035  IFI27   98_D28  Male
## 468: ENSG00000108691.10|CCL2   D28_98   16.79073   CCL2   98_D28  Male
## 469: ENSG00000204936.10|CD177  D28_98   20.44089   CD177   98_D28  Male
## 470: ENSG00000229807.13|XIST   D28_98   44.53194   XIST   98_D28  Male
##           pathogen disease time i.subject
## 1: SARS-CoV-2 COVID19  D0         02
## 2: SARS-CoV-2 COVID19  D0         02
## 3: SARS-CoV-2 COVID19  D0         02
## 4: SARS-CoV-2 COVID19  D0         02
## 5: SARS-CoV-2 COVID19  D0         02
## ---
## 466: SARS-CoV-2 COVID19  D28         98
## 467: SARS-CoV-2 COVID19  D28         98
## 468: SARS-CoV-2 COVID19  D28         98
## 469: SARS-CoV-2 COVID19  D28         98
## 470: SARS-CoV-2 COVID19  D28         98
```

```
## Boxplot with all the points
```

```
ggplot() + theme_pubclean() +
```

```

# plots all the points
geom_quasirandom(data = ncount.goi.dtm,
                 aes(x = gene_name, y = ncount,
                    fill = sex, colour = time),
                 dodge.width = 0.8, size = 1) +
geom_boxplot(data = ncount.goi.dtm,
             aes(x = gene_name, y = ncount,
                fill = sex, colour = time),
             alpha = 0.5, outlier.shape = NA) +
ggtitle(paste0("Boxplot with ggbeeswarm plot showing distribution of the counts\n",
              "Data split by sex and timepoint; NOTE - there are few MALEs with high XIST expression"))
scale_colour_brewer(palette = "Dark2") +
scale_fill_brewer(palette = "Dark2") +
# use wrap to conveniently re-arrange results
facet_wrap(~sex) +
scale_y_log10() +
theme(aspect.ratio = 1.5,
      axis.text = element_text(colour = "black"),
      legend.position = "bottom")

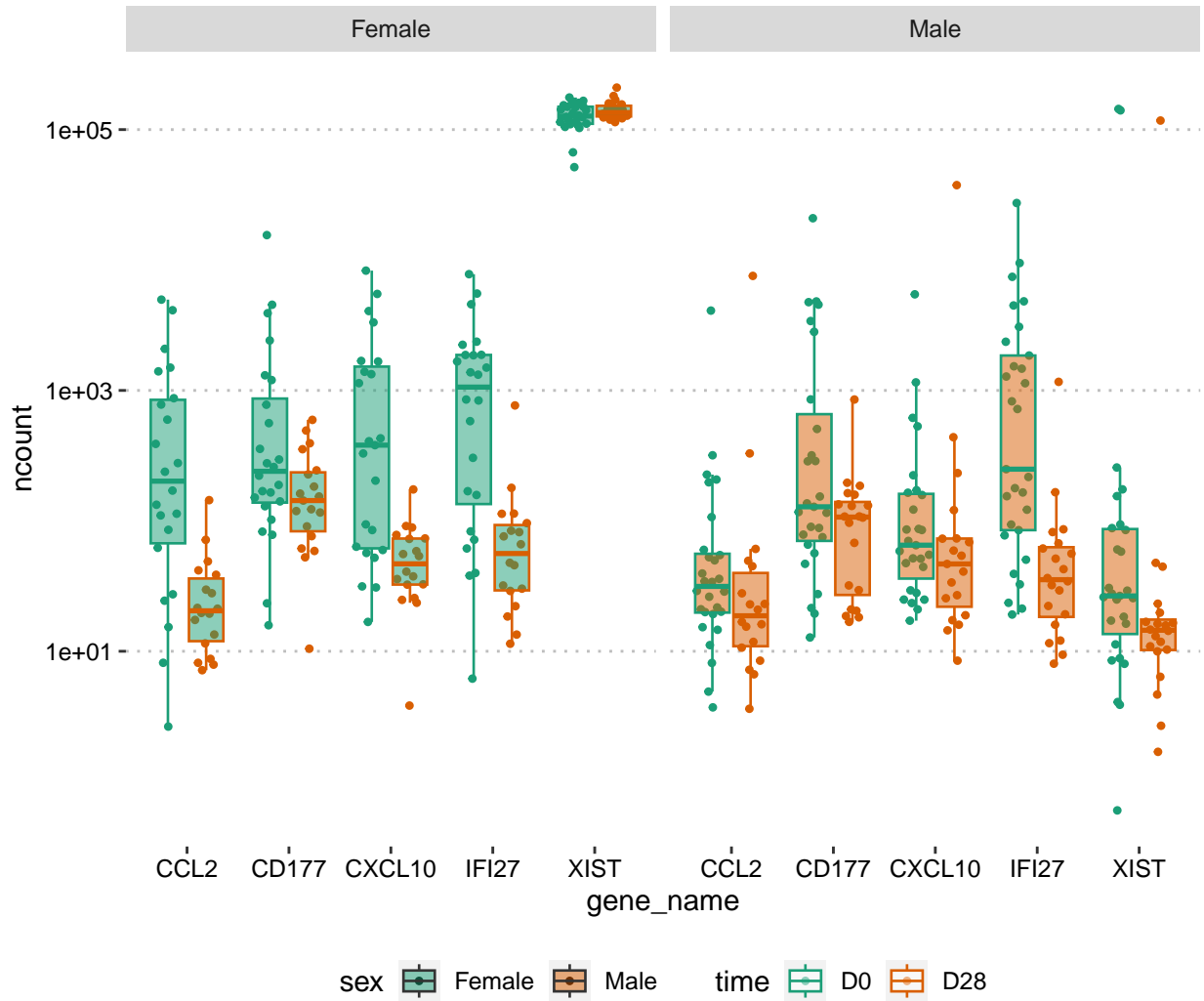
```

```

## Warning: Transformation introduced infinite values in continuous y-axis
## Transformation introduced infinite values in continuous y-axis
## Warning: Removed 26 rows containing non-finite values (`stat_boxplot()`).
## Warning: Removed 26 rows containing missing values (`geom_point()`).

```

Boxplot with ggbeeswarm plot showing distribution of the counts  
Data split by sex and timepoint; NOTE – there are few MALES with high  $\lambda$



## NOTE: PRACTICE - Try identifying and plotting sex-specific  
## genes that could help distinguish genetic sex

### 3. Volcano plot

- Volcanos are a common way to show overall change in gene expression in comparison of two conditions
- They combine statistical information with directional expression change information
- It is also nice to highlight few genes of interest on these plots

```
## Input data
#res.sex.dt

## Sig. up and down
## NOTE: ideally by padj value and can also be done by pvalue
res.up.dt <- res.sex.dt[padj <= 0.05][log2FoldChange > 0][order(-log2FoldChange)][1:10]
res.dn.dt <- res.sex.dt[padj <= 0.05][log2FoldChange < 0][order(log2FoldChange)][1:10]

## Volcano
ggplot() + theme_pubclean() +
  # plot non-significant points
  geom_point(data = res.sex.dt[pvalue > 0.05], ## non-significant genes - all
    aes(x = log2FoldChange, y = -log10(pvalue)),
    size=1, colour = "black") +

  # plot points by significant pvalue
  geom_point(data = res.sex.dt[pvalue <= 0.05],
    aes(x = log2FoldChange, y = -log10(pvalue)),
    size=1, colour = "grey50") +

  # plot only top significant - INCREASED
  geom_point(data = res.sex.dt[padj <= 0.05][log2FoldChange > 0],
    aes(x = log2FoldChange, y = -log10(pvalue)),
    size=1, colour = "firebrick1") +

  # add labels
  geom_text_repel(data = res.up.dt,
    aes(x = log2FoldChange, y = -log10(pvalue),
      label = unlist(tstrsplit(gene_id,split="\\|",keep = 2))),
    size=3, colour = "firebrick1", segment.linetype = "dotted",
    nudge_x = 1,
    direction = "y",
    force = 2,
    force_pull = NA,
    vjust=1,
    hjust=1,
    segment.size = 0.2) +

  # plot only top significant - DECREASED
  geom_point(data = res.sex.dt[padj <= 0.05][log2FoldChange < 0],
    aes(x = log2FoldChange, y = -log10(pvalue)),
    size=1, colour = "dodgerblue") +

  # add labels
  geom_text_repel(data = res.dn.dt,
    aes(x = log2FoldChange, y = -log10(pvalue),
```

```

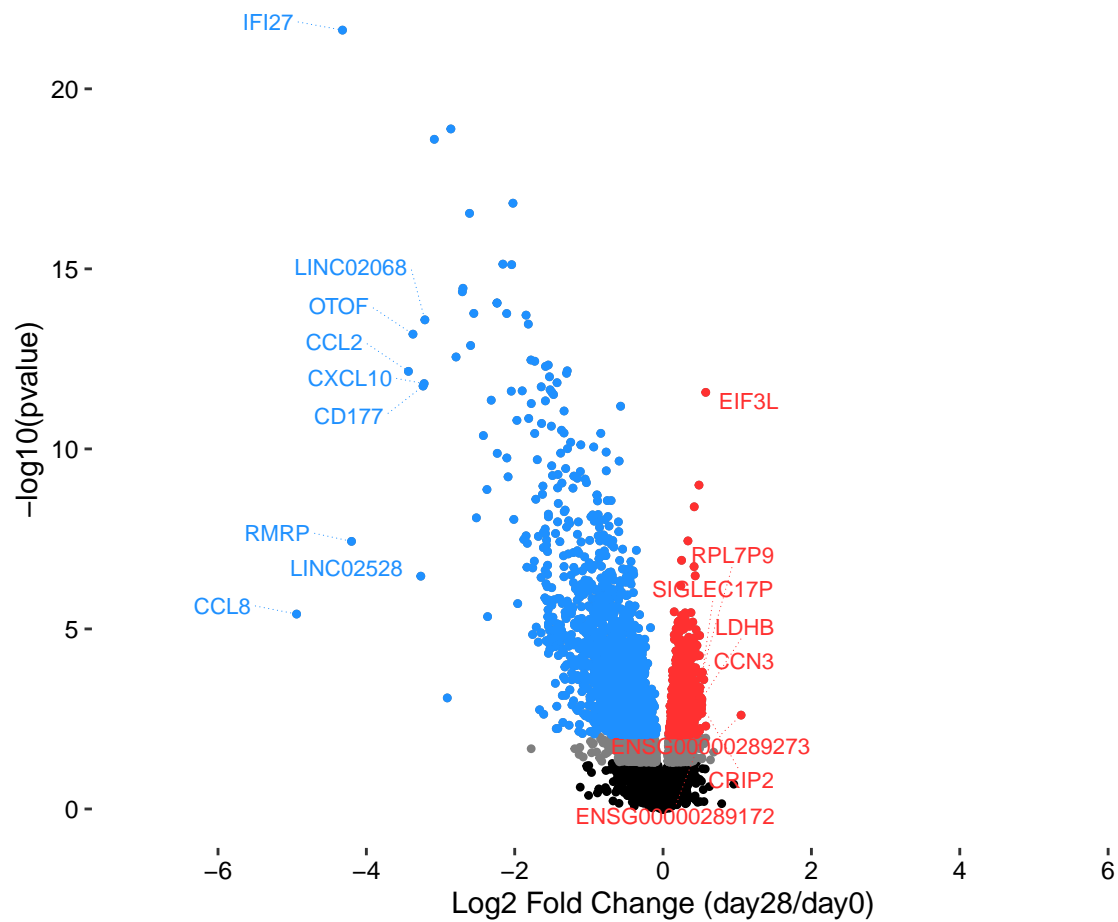
        label = unlist(tstrsplit(gene_id,split="\\|",keep = 2))),
        size=3, colour = "dodgerblue", segment.linetype = "dotted",
        nudge_x = -1,
        force = 3,
        force_pull = NA,
        vjust=0,
        direction = "y",
        segment.size = 0.2) +

# add scales and extras
scale_x_continuous(limits = c(-7,7), breaks = seq(-10,10,2)) +
ggtitle(paste0("Volcano showing results of DGE analysis comparing d28 versus d0\n",
               "red/blue indicate genes significantly changed in this comparison\n",
               "many inflammatory markers decrease by 28 days")) +
xlab("Log2 Fold Change (day28/day0)") +
theme(aspect.ratio = 0.75,
      panel.grid.major.y = element_blank(),
      axis.text = element_text(colour = "black"))

## Warning: Removed 2 rows containing missing values (`geom_point()`).
## Warning: Removed 1 rows containing missing values (`geom_point()`).
## Warning: Removed 1 rows containing missing values (`geom_text_repel()`).
## Warning: Removed 1 rows containing missing values (`geom_point()`).
## Warning: Removed 1 rows containing missing values (`geom_text_repel()`).
## Warning: ggrepel: 1 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps

```

Volcano showing results of DGE analysis comparing d28 versus d0  
 red/blue indicate genes significantly changed in this comparison  
 many inflammatory markers decrease by 28 days



#### 4. Gene Set Enrichment Analysis (GSEA)

- After identification of differentially changing genes, it is very informative to determine whether these changes amount to any systemic / pathway-specific changes
- It is much easier to interpret results of DGE in terms of pathway increase of single gene - such analysis has more significance
- To do this we extract significantly changing genes from DESeq results
  - When there are too few *padj* significant genes it is possible to do GSEA with genes that pass *pvalue* significance
  - If there are not significant genes even by *pvalue*, analysis really loses significance

```
## input data
#res.sex.dt

## First order the results by the log2FoldChange from increasing to decreasing
## NOTE: write into new object to not overwrite the original results
res.sex.gsea.dt <- res.sex.dt[order(-log2FoldChange)]
res.sex.gsea.dt
```

	gene_id	baseMean	log2FoldChange	lfcSE
1:	ENSG00000269693.1 ENSG00000269693	16.47864	24.8272814	2.9422018
2:	ENSG00000289273.1 ENSG00000289273	51.07667	1.0517758	0.3475690
3:	ENSG00000253755.1 IGHGP	52.29388	0.9532807	0.7521117
4:	ENSG00000261796.1 ISY1-RAB43	65.39494	0.7907526	2.1478173
5:	ENSG00000184702.20 SEPTIN5	96.50035	0.6822781	0.3072850
---				
16065:	ENSG00000108691.10 CCL2	266.02706	-3.4328214	0.4781748
16066:	ENSG00000269900.3 RMRP	19150.27418	-4.2001709	0.7630199
16067:	ENSG00000165949.13 IFI27	1174.00428	-4.3219911	0.4443898
16068:	ENSG00000108700.5 CCL8	69.89154	-4.9407004	1.0697586
16069:	ENSG00000215472.10 RPL17-C18orf32	15.85758	-10.6915338	2.6299579

```
##          stat      pvalue      padj
## 1: 8.4383340 3.218926e-17 8.620821e-14
## 2: 3.0260916 2.477372e-03 2.112998e-02
## 3: 1.2674723 2.049865e-01 3.949952e-01
## 4: 0.3681657 7.127497e-01 8.347187e-01
## 5: 2.2203429 2.639550e-02 1.031742e-01
## ---
## 16065: -7.1790102 7.021789e-13 4.339736e-10
## 16066: -5.5046674 3.698659e-08 4.571827e-06
## 16067: -9.7256768 2.343417e-22 3.765638e-18
## 16068: -4.6185189 3.864887e-06 1.996941e-04
## 16069: -4.0652870 4.797338e-05 1.265820e-03

## Make sure gene_name is available, if not extract it to new column
res.sex.gsea.dt[, "gene_name" := tstrsplit(gene_id, split="\\|", keep = 2)]
res.sex.gsea.dt
```

	gene_id	baseMean	log2FoldChange	lfcSE
1:	ENSG00000269693.1 ENSG00000269693	16.47864	24.8272814	2.9422018
2:	ENSG00000289273.1 ENSG00000289273	51.07667	1.0517758	0.3475690
3:	ENSG00000253755.1 IGHGP	52.29388	0.9532807	0.7521117
4:	ENSG00000261796.1 ISY1-RAB43	65.39494	0.7907526	2.1478173

```

##      5:      ENSG00000184702.20|SEPTIN5      96.50035      0.6822781 0.3072850
##      ---
## 16065:      ENSG00000108691.10|CCL2      266.02706      -3.4328214 0.4781748
## 16066:      ENSG00000269900.3|RMRP      19150.27418      -4.2001709 0.7630199
## 16067:      ENSG00000165949.13|IFI27      1174.00428      -4.3219911 0.4443898
## 16068:      ENSG00000108700.5|CCL8      69.89154      -4.9407004 1.0697586
## 16069: ENSG00000215472.10|RPL17-C18orf32      15.85758      -10.6915338 2.6299579
##      stat      pvalue      padj      gene_name
##      1: 8.4383340 3.218926e-17 8.620821e-14 ENSG00000269693
##      2: 3.0260916 2.477372e-03 2.112998e-02 ENSG00000289273
##      3: 1.2674723 2.049865e-01 3.949952e-01      IGHGP
##      4: 0.3681657 7.127497e-01 8.347187e-01      ISY1-RAB43
##      5: 2.2203429 2.639550e-02 1.031742e-01      SEPTIN5
##      ---
## 16065: -7.1790102 7.021789e-13 4.339736e-10      CCL2
## 16066: -5.5046674 3.698659e-08 4.571827e-06      RMRP
## 16067: -9.7256768 2.343417e-22 3.765638e-18      IFI27
## 16068: -4.6185189 3.864887e-06 1.996941e-04      CCL8
## 16069: -4.0652870 4.797338e-05 1.265820e-03 RPL17-C18orf32

## check if gene name is unique - duplicates cannot move further in analysis
summary(duplicated(res.sex.gsea.dt[["gene_name"]])) # <- there are 49 duplicates here

##      Mode      FALSE      TRUE
## logical 16020      49

## what do duplicates look like?
dup_gnames <- unique(res.sex.gsea.dt[duplicated(res.sex.gsea.dt[["gene_name"]])[["gene_name"]])
length(dup_gnames)

## [1] 28

res.sex.gsea.dt[gene_name %in% dup_gnames][order(gene_name)][1:6]

##      gene_id baseMean log2FoldChange      lfcSE
## 1: ENSG00000277739.1|5_8S_rRNA 104.18887      -0.73001360 0.18702955
## 2: ENSG00000275757.1|5_8S_rRNA 65.69818      -0.74176688 0.18618235
## 3: ENSG00000273730.1|5_8S_rRNA 63.50881      -0.77845299 0.19944816
## 4: ENSG00000197976.12_PAR_Y|AKAP17A 652.72880      0.05233269 0.05196628
## 5: ENSG00000197976.12|AKAP17A 638.01924      0.04448442 0.05398411
## 6: ENSG00000231259.5|ANAPC1P2 98.47333      0.18787592 0.45798714
##      stat      pvalue      padj gene_name
## 1: -3.9031992 9.492952e-05 0.002128958 5_8S_rRNA
## 2: -3.9840881 6.773975e-05 0.001644275 5_8S_rRNA
## 3: -3.9030342 9.499426e-05 0.002128958 5_8S_rRNA
## 4: 1.0070509 3.139103e-01 0.515453149 AKAP17A
## 5: 0.8240279 4.099237e-01 0.608111520 AKAP17A
## 6: 0.4102210 6.816439e-01 0.814919674 ANAPC1P2

##
## Select the genes to use for GSEA
##

## Sets cutoff of significance
res_cutoff <- 0.05

## Check: Since there are more than few hundred sig. genes by padj use those

```



```

nrow(res.sex.gsea.dt[padj<res_cutoff])

## [1] 2860

## Check: Are any of 2860 duplicated? <- YES some are
res.sex.gsea.dt[padj<res_cutoff][duplicated(res.sex.gsea.dt[padj<res_cutoff][["gene_name"]])]

##           gene_id baseMean log2FoldChange lfcSE
## 1: ENSG00000182162.11|P2RY8 1365.00269      0.2167833 0.04963399
## 2: ENSG00000169100.14|PAR_Y|SLC25A6 1191.44637      0.1981591 0.06998477
## 3: ENSG00000276596.1|U2 417.54855     -0.6324921 0.17393389
## 4: ENSG00000273709.1|U2 1095.03288     -0.6435656 0.21330581
## 5: ENSG00000277903.1|U2 413.31202     -0.6443804 0.20056111
## 6: ENSG00000278774.1|U2 480.69558     -0.6516084 0.17378676
## 7: ENSG00000274062.1|U2 417.40218     -0.6577563 0.17479295
## 8: ENSG00000275219.1|U2 419.36283     -0.6651420 0.17594991
## 9: ENSG00000274862.1|U2 442.21074     -0.6855793 0.17461317
## 10: ENSG00000278591.1|U2 439.43502     -0.6930374 0.17706424
## 11: ENSG00000274452.1|U2 404.24922     -0.7116059 0.18416220
## 12: ENSG00000275757.1|5_8S_rRNA 65.69818     -0.7417669 0.18618235
## 13: ENSG00000273730.1|5_8S_rRNA 63.50881     -0.7784530 0.19944816
##           stat      pvalue      padj gene_name
## 1: 4.367637 1.255978e-05 0.0004693561 P2RY8
## 2: 2.831460 4.633597e-03 0.0322325816 SLC25A6
## 3: -3.636393 2.764820e-04 0.0046039261 U2
## 4: -3.017103 2.552029e-03 0.0215834504 U2
## 5: -3.212888 1.314074e-03 0.0137742082 U2
## 6: -3.749471 1.772080e-04 0.0033755343 U2
## 7: -3.763060 1.678470e-04 0.0032487899 U2
## 8: -3.780292 1.566447e-04 0.0031114009 U2
## 9: -3.926275 8.627145e-05 0.0019946705 U2
## 10: -3.914045 9.076270e-05 0.0020687459 U2
## 11: -3.864017 1.115374e-04 0.0023984195 U2
## 12: -3.984088 6.773975e-05 0.0016442749 5_8S_rRNA
## 13: -3.903034 9.499426e-05 0.0021289578 5_8S_rRNA

res.sex.gsea.dt[padj<res_cutoff][gene_name %in% "P2RY8"]

##           gene_id baseMean log2FoldChange lfcSE      stat
## 1: ENSG00000182162.11|PAR_Y|P2RY8 1413.283      0.2170988 0.04945298 4.390005
## 2: ENSG00000182162.11|P2RY8 1365.003      0.2167833 0.04963399 4.367637
##           pvalue      padj gene_name
## 1: 1.133479e-05 0.0004275556 P2RY8
## 2: 1.255978e-05 0.0004693561 P2RY8

## Remove genes with _PAR_Y
res.sex.gsea.dt <- res.sex.gsea.dt[grep("_PAR_Y",gene_id, invert = TRUE)]

## Check: How do our duplicates look now?.... there are fewer
res.sex.gsea.dt[padj<res_cutoff][duplicated(res.sex.gsea.dt[padj<res_cutoff][["gene_name"]])]

##           gene_id baseMean log2FoldChange lfcSE      stat
## 1: ENSG00000276596.1|U2 417.54855     -0.6324921 0.1739339 -3.636393
## 2: ENSG00000273709.1|U2 1095.03288     -0.6435656 0.2133058 -3.017103
## 3: ENSG00000277903.1|U2 413.31202     -0.6443804 0.2005611 -3.212888
## 4: ENSG00000278774.1|U2 480.69558     -0.6516084 0.1737868 -3.749471

```

```
## 5:      ENSG00000274062.1|U2  417.40218      -0.6577563  0.1747929 -3.763060
## 6:      ENSG00000275219.1|U2  419.36283      -0.6651420  0.1759499 -3.780292
## 7:      ENSG00000274862.1|U2  442.21074      -0.6855793  0.1746132 -3.926275
## 8:      ENSG00000278591.1|U2  439.43502      -0.6930374  0.1770642 -3.914045
## 9:      ENSG00000274452.1|U2  404.24922      -0.7116059  0.1841622 -3.864017
## 10: ENSG00000275757.1|5_8S_rRNA  65.69818      -0.7417669  0.1861824 -3.984088
## 11: ENSG00000273730.1|5_8S_rRNA  63.50881      -0.7784530  0.1994482 -3.903034
##      pvalue      padj gene_name
## 1: 2.764820e-04 0.004603926      U2
## 2: 2.552029e-03 0.021583450      U2
## 3: 1.314074e-03 0.013774208      U2
## 4: 1.772080e-04 0.003375534      U2
## 5: 1.678470e-04 0.003248790      U2
## 6: 1.566447e-04 0.003111401      U2
## 7: 8.627145e-05 0.001994670      U2
## 8: 9.076270e-05 0.002068746      U2
## 9: 1.115374e-04 0.002398419      U2
## 10: 6.773975e-05 0.001644275 5_8S_rRNA
## 11: 9.499426e-05 0.002128958 5_8S_rRNA
```

```
## Keep only unique gene names
```

```
res.sex.gsea.dt <- res.sex.gsea.dt[!duplicated(gene_name)]
```

```
## Extract log2FoldChange for these genes into a vector
```

```
geneVec <- res.sex.gsea.dt[padj <= res_cutoff][["log2FoldChange"]]
```

```
## Add names for each gene to the vector
```

```
names(geneVec) <- res.sex.gsea.dt[padj <= res_cutoff][["gene_name"]]
```

```
## Check if vector names are duplicated
```

```
summary(duplicated(names(geneVec))) ## NO MORE duplications
```

```
##      Mode      FALSE
```

```
## logical      2844
```

```
##
```

```
## DO THIS IF YOU DID NOT FILTER BEFORE
```

```
## If above is yes: Find the duplicates and their names
```

```
#geneVec[duplicated(names(geneVec))]
```

```
#dup_names <- names(geneVec[duplicated(names(geneVec))])
```

```
#geneVec[names(geneVec) %in% dup_names]
```

```
## For now remove duplicates BUT - work to avoid having duplicate gene names
```

```
#geneVec <- geneVec[!duplicated(names(geneVec))]
```

```
##
```

```
##
```

```
## gene vector to proceed with GSEA analysis
```

```
geneVec[1:10]
```

```
## ENSG00000269693 ENSG00000289273 ENSG00000289172      EIF3L      KLRB1
##      24.8272814      1.0517758      0.5757175      0.5752445      0.5488337
##      SIGLEC17P      RPL7P9      LDHB      CRIP2      CCN3
```

```
##      0.5299390      0.5253123      0.5226034      0.5215145      0.5213166
```

```
##
```

```
## Use the msigdb package to load gene lists to compare with
```

```
##
```

```
##?msigdb
```

```
##msigdb(species = "Homo sapiens", category = "C7") # <- other gene sets
```

```
msig.H.dt <- as.data.table(msigdb(species = "Homo sapiens", category = "H"))
```

```
msig.H.dt[1:5,]
```

```
##      gs_cat gs_subcat      gs_name gene_symbol entrez_gene
## 1:      H      H      HALLMARK_ADIPOGENESIS      ABCA1      19
## 2:      H      H      HALLMARK_ADIPOGENESIS      ABCB8     11194
## 3:      H      H      HALLMARK_ADIPOGENESIS      ACAA2     10449
## 4:      H      H      HALLMARK_ADIPOGENESIS      ACADL      33
## 5:      H      H      HALLMARK_ADIPOGENESIS      ACADM      34
##      ensembl_gene human_gene_symbol human_entrez_gene human_ensembl_gene gs_id
## 1: ENSG00000165029      ABCA1      19      ENSG00000165029 M5905
## 2: ENSG00000197150      ABCB8     11194      ENSG00000197150 M5905
## 3: ENSG00000167315      ACAA2     10449      ENSG00000167315 M5905
## 4: ENSG00000115361      ACADL      33      ENSG00000115361 M5905
## 5: ENSG00000117054      ACADM      34      ENSG00000117054 M5905
##      gs_pmid gs_geoid gs_exact_source gs_url
## 1: 26771021
## 2: 26771021
## 3: 26771021
## 4: 26771021
## 5: 26771021
##
##      gs_description
## 1: Genes up-regulated during adipocyte differentiation (adipogenesis).
## 2: Genes up-regulated during adipocyte differentiation (adipogenesis).
## 3: Genes up-regulated during adipocyte differentiation (adipogenesis).
## 4: Genes up-regulated during adipocyte differentiation (adipogenesis).
## 5: Genes up-regulated during adipocyte differentiation (adipogenesis).
```

```
## View how many genes are involved in each category
```

```
msig.H.dt[,N,by = gs_name]
```

```
##      gs_name      N
## 1:      HALLMARK_ADIPOGENESIS 210
## 2:      HALLMARK_ALLOGRAFT_REJECTION 335
## 3:      HALLMARK_ANDROGEN_RESPONSE 102
## 4:      HALLMARK_ANGIOGENESIS 36
## 5:      HALLMARK_APICAL_JUNCTION 231
## 6:      HALLMARK_APICAL_SURFACE 46
## 7:      HALLMARK_APOPTOSIS 183
## 8:      HALLMARK_BILE_ACID_METABOLISM 114
## 9:      HALLMARK_CHOLESTEROL_HOMEOSTASIS 77
## 10:      HALLMARK_COAGULATION 162
## 11:      HALLMARK_COMPLEMENT 237
## 12:      HALLMARK_DNA_REPAIR 170
## 13:      HALLMARK_E2F_TARGETS 218
## 14: HALLMARK_EPITHELIAL_MESENCHYMAL_TRANSITION 204
## 15:      HALLMARK_ESTROGEN_RESPONSE_EARLY 216
```

```
## 16:          HALLMARK_ESTROGEN_RESPONSE_LATE 218
## 17:          HALLMARK_FATTY_ACID_METABOLISM 165
## 18:          HALLMARK_G2M_CHECKPOINT 204
## 19:          HALLMARK_GLYCOLYSIS 215
## 20:          HALLMARK_HEDGEHOG_SIGNALING 36
## 21:          HALLMARK_HEME_METABOLISM 214
## 22:          HALLMARK_HYPOXIA 215
## 23:          HALLMARK_IL2_STAT5_SIGNALING 216
## 24:          HALLMARK_IL6_JAK_STAT3_SIGNALING 103
## 25:          HALLMARK_INFLAMMATORY_RESPONSE 222
## 26:          HALLMARK_INTERFERON_ALPHA_RESPONSE 140
## 27:          HALLMARK_INTERFERON_GAMMA_RESPONSE 286
## 28:          HALLMARK_KRAS_SIGNALING_DN 220
## 29:          HALLMARK_KRAS_SIGNALING_UP 220
## 30:          HALLMARK_MITOTIC_SPINDLE 215
## 31:          HALLMARK_MTORC1_SIGNALING 211
## 32:          HALLMARK_MYC_TARGETS_V1 236
## 33:          HALLMARK_MYC_TARGETS_V2 60
## 34:          HALLMARK_MYOGENESIS 212
## 35:          HALLMARK_NOTCH_SIGNALING 34
## 36:          HALLMARK_OXIDATIVE_PHOSPHORYLATION 220
## 37:          HALLMARK_P53_PATHWAY 215
## 38:          HALLMARK_PANCREAS_BETA_CELLS 44
## 39:          HALLMARK_PEROXISOME 110
## 40:          HALLMARK_PI3K_AKT_MTOR_SIGNALING 118
## 41:          HALLMARK_PROTEIN_SECRETION 98
## 42:          HALLMARK_REACTIVE_OXYGEN_SPECIES_PATHWAY 58
## 43:          HALLMARK_SPERMATOGENESIS 144
## 44:          HALLMARK_TGF_BETA_SIGNALING 59
## 45:          HALLMARK_TNFA_SIGNALING_VIA_NFKB 228
## 46:          HALLMARK_UNFOLDED_PROTEIN_RESPONSE 115
## 47:          HALLMARK_UV_RESPONSE_DN 152
## 48:          HALLMARK_UV_RESPONSE_UP 191
## 49:          HALLMARK_WNT_BETA_CATENIN_SIGNALING 50
## 50:          HALLMARK_XENOBIOTIC_METABOLISM 224
##                                     gs_name N
```

```
## Extract only gene_name and pathway name columns
```

```
msig.H.t2g <- msig.H.dt[,.SD,.SDcols = c("gs_name","gene_symbol")]
msig.H.t2g
```

```
##                                     gs_name gene_symbol
## 1:          HALLMARK_ADIPOGENESIS          ABCA1
## 2:          HALLMARK_ADIPOGENESIS          ABCB8
## 3:          HALLMARK_ADIPOGENESIS          ACAA2
## 4:          HALLMARK_ADIPOGENESIS          ACADL
## 5:          HALLMARK_ADIPOGENESIS          ACADM
## ---
## 8205: HALLMARK_XENOBIOTIC_METABOLISM          UPB1
## 8206: HALLMARK_XENOBIOTIC_METABOLISM          UPP1
## 8207: HALLMARK_XENOBIOTIC_METABOLISM          VNN1
## 8208: HALLMARK_XENOBIOTIC_METABOLISM          VTN
## 8209: HALLMARK_XENOBIOTIC_METABOLISM          XDH
```

```
##
## Run GSEA
## => Much of the options can found online in the ClusterProfiler manual
##

?GSEA
agsea <- clusterProfiler::GSEA(geneList = geneVec,
                              TERM2GENE = msig.H.t2g,
                              minGSSize = 5, # minimum number of genes to match pathway
                              eps = 0,
                              pvalueCutoff = 1, # this way all pathways are returned
                              pAdjustMethod = "BH", # many other methods are out there
                              by = 'fgsea',
                              seed = TRUE)
```

```
## preparing geneSet collections...
```

```
## GSEA analysis...
```

```
## leading edge analysis...
```

```
## done...
```

```
agsea.dt <- as.data.table(x = agsea)
```

```
## Clean up and organize the results table
```

```
agsea.dt <- agsea.dt[order(p.adjust)]
agsea.dt[, "sig" := p.adjust<=0.05]
agsea.dt[, "updown" := ifelse(NES<0, "down", "up")]
agsea.dt[["ID"]] <- gsub("HALLMARK_", "", agsea.dt[["ID"]])
agsea.dt[["ID"]] <- tolower(gsub("_", " ", agsea.dt[["ID"]]))
agsea.dt[["ID"]] <- factor(agsea.dt[["ID"]], levels = agsea.dt[["ID"]])
agsea.dt[["Description"]] <- NULL
agsea.dt[1:4]
```

```
##
## ID setSize enrichmentScore NES pvalue
## 1: interferon alpha response 54 -0.6742649 -2.032744 4.775842e-09
## 2: interferon gamma response 88 -0.6115491 -1.917812 4.273297e-09
## 3: inflammatory response 64 -0.4748911 -1.451781 1.144414e-02
## 4: pancreas beta cells 7 -0.7125717 -1.550740 1.829138e-02
```

```
## p.adjust qvalue rank leading_edge
## 1: 1.146202e-07 1.080849e-07 619 tags=70%, list=22%, signal=56%
## 2: 1.146202e-07 1.080849e-07 619 tags=57%, list=22%, signal=46%
## 3: 1.831063e-01 1.726660e-01 850 tags=53%, list=30%, signal=38%
## 4: 2.194965e-01 2.069814e-01 376 tags=57%, list=13%, signal=50%
```

```
##
## 1: SAMD9L/UBE2L6/NUB1/HELZ2/IFI
## 2: SAMD9L/C1R/UBE2L6/VAMP5/MX2/HELZ2/IFI30/TOR1B/LAP3/CDKN1A/TRAFF1/SOCS3/EIF2AK2/MX1/IFIT3/PLSCR1/F
## 3:
## 4:
```

```
## sig updown
## 1: TRUE down
## 2: TRUE down
## 3: FALSE down
## 4: FALSE down
```

## ## FINAL RESULT

agsea.dt[1:4]

```
##              ID setSize enrichmentScore      NES      pvalue
## 1: interferon alpha response      54      -0.6742649 -2.032744 4.775842e-09
## 2: interferon gamma response      88      -0.6115491 -1.917812 4.273297e-09
## 3:      inflammatory response      64      -0.4748911 -1.451781 1.144414e-02
## 4:      pancreas beta cells        7      -0.7125717 -1.550740 1.829138e-02
##      p.adjust      qvalue rank      leading_edge
## 1: 1.146202e-07 1.080849e-07 619 tags=70%, list=22%, signal=56%
## 2: 1.146202e-07 1.080849e-07 619 tags=57%, list=22%, signal=46%
## 3: 1.831063e-01 1.726660e-01 850 tags=53%, list=30%, signal=38%
## 4: 2.194965e-01 2.069814e-01 376 tags=57%, list=13%, signal=50%
##
## 1:
## 2: SAMD9L/C1R/UBE2L6/VAMP5/MX2/HELZ2/IFI30/TOR1B/LAP3/CDKN1A/TRAFF1/SOCS3/EIF2AK2/MX1/IFIT3/PLSCR1/F
## 3:
## 4:
##      sig updown
## 1:  TRUE  down
## 2:  TRUE  down
## 3: FALSE  down
## 4: FALSE  down
```

## PART 3

Goal: - Explore ways of plotting results from GSEA and DESeq - Other ways of visualizing data

### 1. Barplot of up and down-regulated pathways

- This plot highlights the pathways that have increased or decreased based on the significance cut-offs
- It is sometimes helpful to plot the pathways that are not part of the significance

## ## Input data

agsea.dt

```
##              ID setSize enrichmentScore      NES
## 1:      interferon alpha response      54      -0.6742649 -2.0327443
## 2:      interferon gamma response      88      -0.6115491 -1.9178124
## 3:      inflammatory response      64      -0.4748911 -1.4517812
## 4:      pancreas beta cells        7      -0.7125717 -1.5507402
## 5:      unfolded protein response      14      -0.5841944 -1.4953965
## 6:      kras signaling dn          15      -0.5503737 -1.4315504
## 7:      coagulation              19      -0.5135701 -1.3914900
## 8:      kras signaling up         39      -0.4513407 -1.3225000
## 9:      oxidative phosphorylation 20       0.2943100  1.1838764
## 10:      myogenesis              26      -0.4502819 -1.2788929
## 11:      dna repair              10      -0.5222676 -1.2435179
## 12:      cholesterol homeostasis   9      -0.5312101 -1.2293405
## 13:      il2 stat5 signaling       40      -0.4184349 -1.2277787
## 14:      reactive oxygen species pathway 6       0.4305849  1.2091984
```

## 15:	mitotic spindle	43	-0.4022224	-1.1920642
## 16:	hypoxia	31	-0.4115007	-1.1916630
## 17:	tnfa signaling via nfkb	64	-0.3868463	-1.1826210
## 18:	uv response dn	19	-0.4388268	-1.1889771
## 19:	xenobiotic metabolism	28	-0.4034161	-1.1539519
## 20:	hedgehog signaling	6	-0.5412164	-1.1291672
## 21:	complement	52	-0.3679634	-1.1053509
## 22:	il6 jak stat3 signaling	22	-0.4011244	-1.1162869
## 23:	e2f targets	39	-0.3722452	-1.0907376
## 24:	uv response up	28	-0.3801067	-1.0872767
## 25:	apical junction	30	-0.3719308	-1.0724718
## 26:	g2m checkpoint	48	-0.3520941	-1.0514750
## 27:	estrogen response late	28	-0.3671310	-1.0501602
## 28:	apoptosis	36	-0.3469542	-1.0135272
## 29:	spermatogenesis	15	-0.3747844	-0.9748337
## 30:	epithelial mesenchymal transition	29	-0.3348670	-0.9617615
## 31:	glycolysis	28	-0.3219665	-0.9209694
## 32:	mtorc1 signaling	27	-0.3166688	-0.9054138
## 33:	myc targets v1	37	0.5154299	1.7970369
## 34:	p53 pathway	47	-0.2907260	-0.8674767
## 35:	tgf beta signaling	11	-0.3285450	-0.7989704
## 36:	estrogen response early	22	-0.2806557	-0.7810352
## 37:	notch signaling	7	-0.3582144	-0.7795670
## 38:	fatty acid metabolism	18	-0.2792705	-0.7500009
## 39:	bile acid metabolism	13	-0.2972065	-0.7452551
## 40:	angiogenesis	5	-0.3671841	-0.7321112
## 41:	adipogenesis	24	-0.2599291	-0.7274941
## 42:	pi3k akt mtor signaling	20	-0.2573072	-0.7011828
## 43:	androgen response	13	0.1666024	0.6597320
## 44:	allograft rejection	53	-0.2011937	-0.6054452
## 45:	heme metabolism	19	-0.2173558	-0.5889136
## 46:	protein secretion	14	0.1445934	0.5800196
## 47:	wnt beta catenin signaling	11	0.1565899	0.5759604
## 48:	peroxisome	17	-0.1979409	-0.5289916
##	ID	setSize	enrichmentScore	NES
##	pvalue	p.adjust	qvalue	rank
## 1:	4.775842e-09	1.146202e-07	1.080849e-07	619
## 2:	4.273297e-09	1.146202e-07	1.080849e-07	619
## 3:	1.144414e-02	1.831063e-01	1.726660e-01	850
## 4:	1.829138e-02	2.194965e-01	2.069814e-01	376
## 5:	4.638619e-02	4.453074e-01	4.199171e-01	19
## 6:	6.716418e-02	4.480432e-01	4.224969e-01	382
## 7:	7.588358e-02	4.480432e-01	4.224969e-01	668
## 8:	8.400810e-02	4.480432e-01	4.224969e-01	700
## 9:	8.379198e-02	4.480432e-01	4.224969e-01	1431
## 10:	1.453429e-01	6.015470e-01	5.672483e-01	750
## 11:	1.984127e-01	6.015470e-01	5.672483e-01	502
## 12:	2.043880e-01	6.015470e-01	5.672483e-01	1190
## 13:	1.548583e-01	6.015470e-01	5.672483e-01	656
## 14:	1.907216e-01	6.015470e-01	5.672483e-01	1622
## 15:	2.058527e-01	6.015470e-01	5.672483e-01	1319
## 16:	2.130479e-01	6.015470e-01	5.672483e-01	820
## 17:	1.783181e-01	6.015470e-01	5.672483e-01	1276
## 18:	2.494802e-01	6.652807e-01	6.273480e-01	748
				tags=70%, list=22%, signal=56%
				tags=57%, list=22%, signal=46%
				tags=53%, list=30%, signal=38%
				tags=57%, list=13%, signal=50%
				tags=14%, list=1%, signal=14%
				tags=53%, list=13%, signal=46%
				tags=53%, list=23%, signal=41%
				tags=41%, list=25%, signal=31%
				tags=95%, list=50%, signal=48%
				tags=50%, list=26%, signal=37%
				tags=30%, list=18%, signal=25%
				tags=89%, list=42%, signal=52%
				tags=35%, list=23%, signal=27%
				tags=100%, list=57%, signal=43%
				tags=63%, list=46%, signal=34%
				tags=39%, list=29%, signal=28%
				tags=62%, list=45%, signal=35%
				tags=53%, list=26%, signal=39%

```
## 19: 2.697436e-01 6.814575e-01 6.426025e-01 1312 tags=68%, list=46%, signal=37%
## 20: 3.168317e-01 7.241867e-01 6.828954e-01 995 tags=100%, list=35%, signal=65%
## 21: 3.047427e-01 7.241867e-01 6.828954e-01 1317 tags=62%, list=46%, signal=34%
## 22: 3.347065e-01 7.261538e-01 6.847503e-01 1619 tags=91%, list=57%, signal=39%
## 23: 3.491903e-01 7.261538e-01 6.847503e-01 1394 tags=67%, list=49%, signal=34%
## 24: 3.630769e-01 7.261538e-01 6.847503e-01 632 tags=25%, list=22%, signal=20%
## 25: 3.912155e-01 7.433051e-01 7.009237e-01 700 tags=33%, list=25%, signal=25%
## 26: 4.026236e-01 7.433051e-01 7.009237e-01 1394 tags=60%, list=49%, signal=31%
## 27: 4.215385e-01 7.494017e-01 7.066727e-01 724 tags=61%, list=25%, signal=46%
## 28: 4.730964e-01 8.110225e-01 7.647800e-01 1397 tags=64%, list=49%, signal=33%
## 29: 5.287846e-01 8.752298e-01 8.253263e-01 752 tags=33%, list=26%, signal=25%
## 30: 5.582822e-01 8.932515e-01 8.423205e-01 1622 tags=83%, list=57%, signal=36%
## 31: 6.102564e-01 9.376915e-01 8.842267e-01 1390 tags=57%, list=49%, signal=30%
## 32: 6.251277e-01 9.376915e-01 8.842267e-01 1778 tags=85%, list=63%, signal=32%
## 33: 9.639918e-01 9.733333e-01 9.178363e-01 470 tags=57%, list=17%, signal=48%
## 34: 7.285570e-01 9.733333e-01 9.178363e-01 1303 tags=49%, list=46%, signal=27%
## 35: 7.480315e-01 9.733333e-01 9.178363e-01 330 tags=64%, list=12%, signal=56%
## 36: 7.754892e-01 9.733333e-01 9.178363e-01 362 tags=55%, list=13%, signal=48%
## 37: 7.773723e-01 9.733333e-01 9.178363e-01 1372 tags=71%, list=48%, signal=37%
## 38: 8.056723e-01 9.733333e-01 9.178363e-01 614 tags=50%, list=22%, signal=39%
## 39: 8.057714e-01 9.733333e-01 9.178363e-01 1065 tags=46%, list=37%, signal=29%
## 40: 8.212878e-01 9.733333e-01 9.178363e-01 1604 tags=100%, list=56%, signal=44%
## 41: 8.345398e-01 9.733333e-01 9.178363e-01 1850 tags=83%, list=65%, signal=29%
## 42: 8.430353e-01 9.733333e-01 9.178363e-01 1235 tags=45%, list=43%, signal=26%
## 43: 9.207921e-01 9.733333e-01 9.178363e-01 267 tags=31%, list=9%, signal=28%
## 44: 9.676768e-01 9.733333e-01 9.178363e-01 1395 tags=49%, list=49%, signal=25%
## 45: 9.282744e-01 9.733333e-01 9.178363e-01 1738 tags=68%, list=61%, signal=27%
## 46: 9.733333e-01 9.733333e-01 9.178363e-01 1795 tags=93%, list=63%, signal=34%
## 47: 9.646018e-01 9.733333e-01 9.178363e-01 68 tags=18%, list=2%, signal=18%
## 48: 9.577614e-01 9.733333e-01 9.178363e-01 85 tags=59%, list=3%, signal=57%
##          pvalue      p.adjust      qvalue rank          leading_edge
##
## 1:
## 2: SAMD9L/C1R/UBE2L6/VAMP5/MX2/HELZ2/IFI30/TOR1B/LAP3/CDKN1A/TRAFD1/SOCS3/EIF2AK2/MX1/IFIT3/PLSCR1/
## 3:
## 4:
## 5:
## 6:
## 7:
## 8:
## 9:
## 10:
## 11:
## 12:
## 13:
## 14:
## 15:
## 16:
## 17: BCL3/JUNB/NAMPT/KYNU/MAFF/NFKBIE/
## 18:
## 19:
## 20:
## 21:
## 22:
```



```

## 23:
## 24:
## 25:
## 26:
## 27:
## 28:
## 29:
## 30:
## 31:
## 32:
## 33:
## 34:
## 35:
## 36:
## 37:
## 38:
## 39:
## 40:
## 41:
## 42:
## 43:
## 44:
## 45:
## 46:
## 47:
## 48:
##
##      sig updown
## 1:  TRUE  down
## 2:  TRUE  down
## 3: FALSE  down
## 4: FALSE  down
## 5: FALSE  down
## 6: FALSE  down
## 7: FALSE  down
## 8: FALSE  down
## 9: FALSE   up
##10: FALSE  down
##11: FALSE  down
##12: FALSE  down
##13: FALSE  down
##14: FALSE   up
##15: FALSE  down
##16: FALSE  down
##17: FALSE  down
##18: FALSE  down
##19: FALSE  down
##20: FALSE  down
##21: FALSE  down
##22: FALSE  down
##23: FALSE  down
##24: FALSE  down
##25: FALSE  down
##26: FALSE  down

```

```

## 27: FALSE    down
## 28: FALSE    down
## 29: FALSE    down
## 30: FALSE    down
## 31: FALSE    down
## 32: FALSE    down
## 33: FALSE     up
## 34: FALSE    down
## 35: FALSE    down
## 36: FALSE    down
## 37: FALSE    down
## 38: FALSE    down
## 39: FALSE    down
## 40: FALSE    down
## 41: FALSE    down
## 42: FALSE    down
## 43: FALSE     up
## 44: FALSE    down
## 45: FALSE    down
## 46: FALSE     up
## 47: FALSE     up
## 48: FALSE    down
##      sig updown

## Reorder to have NES reversed
agsea.dt <- agsea.dt[order(-NES)]

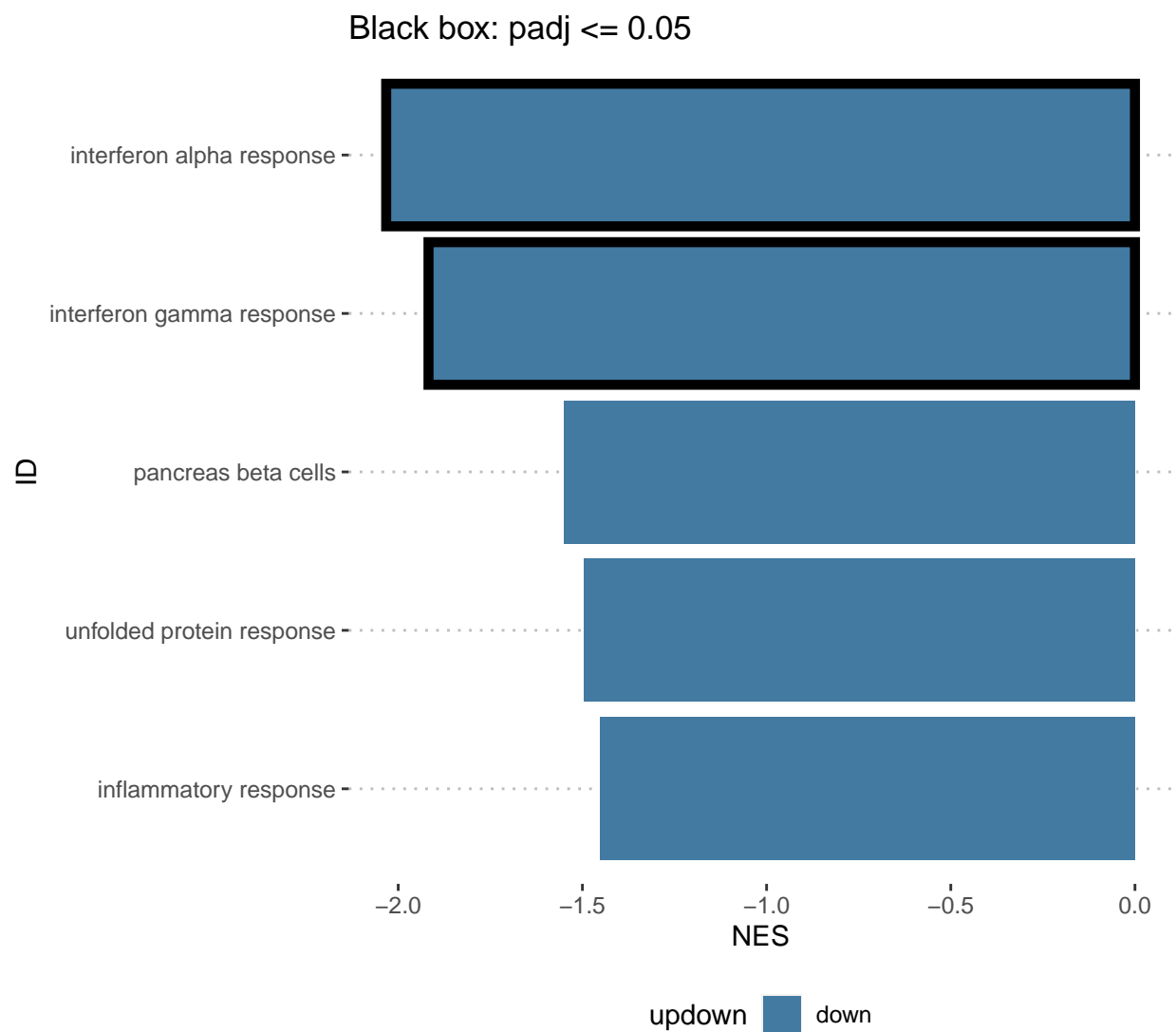
## Change order of labels by adjusting the factor levels
class(agsea.dt[["ID"]])

## [1] "factor"

agsea.dt[["ID"]] <- factor(agsea.dt[["ID"]], levels = agsea.dt[["ID"]])

## Plot - subset
ggp.d28.gset.bar <- ggplot() + theme_pubclean() +
  geom_bar(data = agsea.dt[pvalue <= 0.05],
    aes(x = NES, y = ID, fill = updown),
    stat = "identity") +
  geom_bar(data = agsea.dt[p.adjust <= 0.05], ## SHOW SIGNIFICANT
    aes(x = NES, y = ID),
    colour = "black", lwd = 2,
    fill=NA,
    stat = "identity") +
  scale_fill_manual(values = c("#427AA1", "#CE8D99")) +
  ggtitle("Black box: padj <= 0.05") +
  theme(aspect.ratio = 1,
    legend.position = "bottom",
    axis.text.y = element_text()); ggp.d28.gset.bar

```



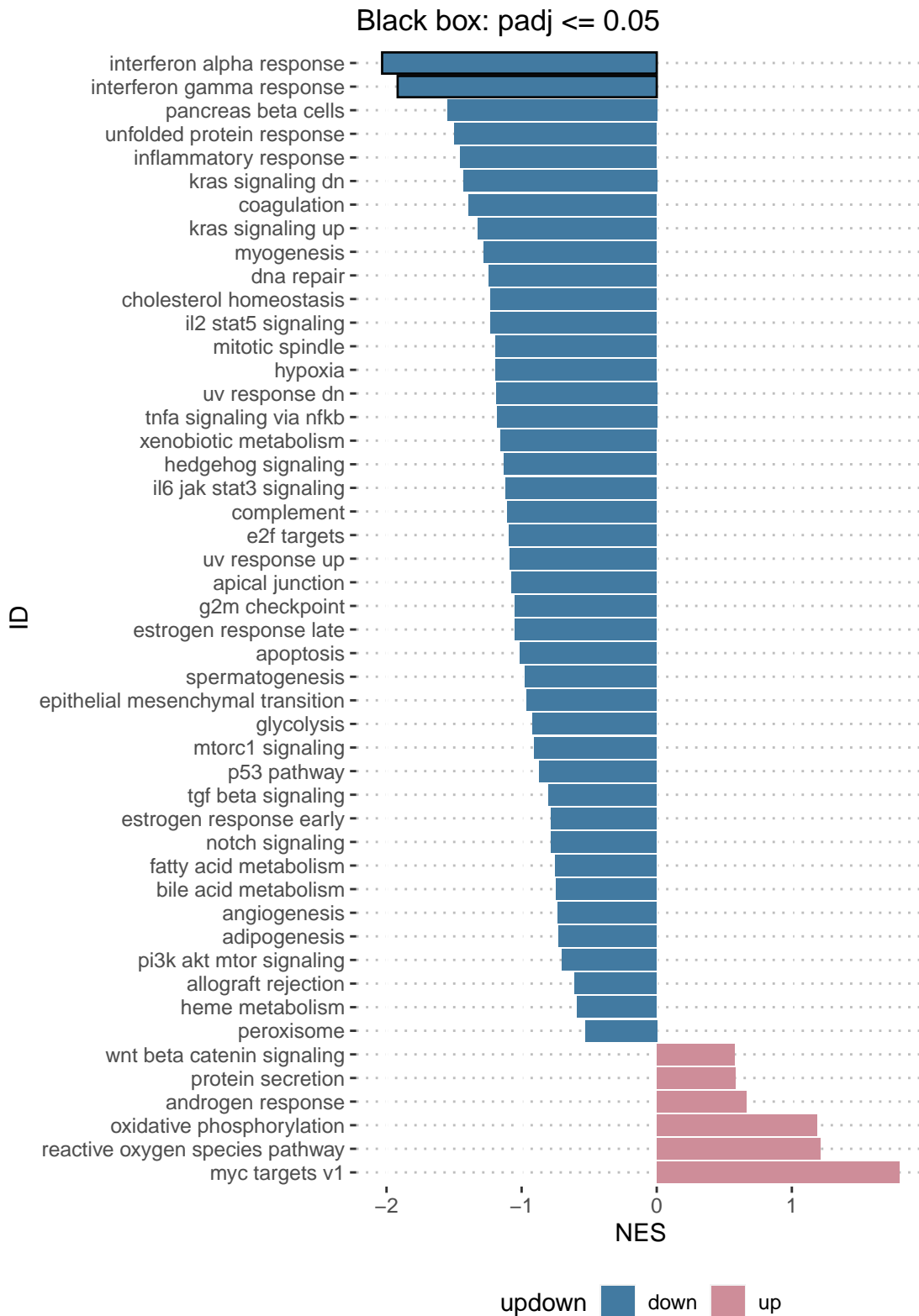
```
## Plot - all
ggp.d28.gset.bar <- ggplot() + theme_pubclean() +
  geom_bar(data = agsea.dt, ## ALL PATHWAYS
```

```

    aes(x = NES, y = ID, fill = updown),
    stat = "identity") +
  geom_bar(data = agsea.dt[p.adjust <= 0.05], ## SHOW SIGNIFICANT
    aes(x = NES, y = ID),
    colour = "black", lwd = 0.51,
    fill=NA,
    stat = "identity") +
  scale_fill_manual(values = c("#427AA1", "#CE8D99")) +
  ggtitle("Black box: padj <= 0.05") +

  theme(aspect.ratio = 2,
    legend.position = "bottom",
    axis.text.y = element_text()); ggp.d28.gset.bar

```



## 2. Leading edge plots - ClusterProfiler

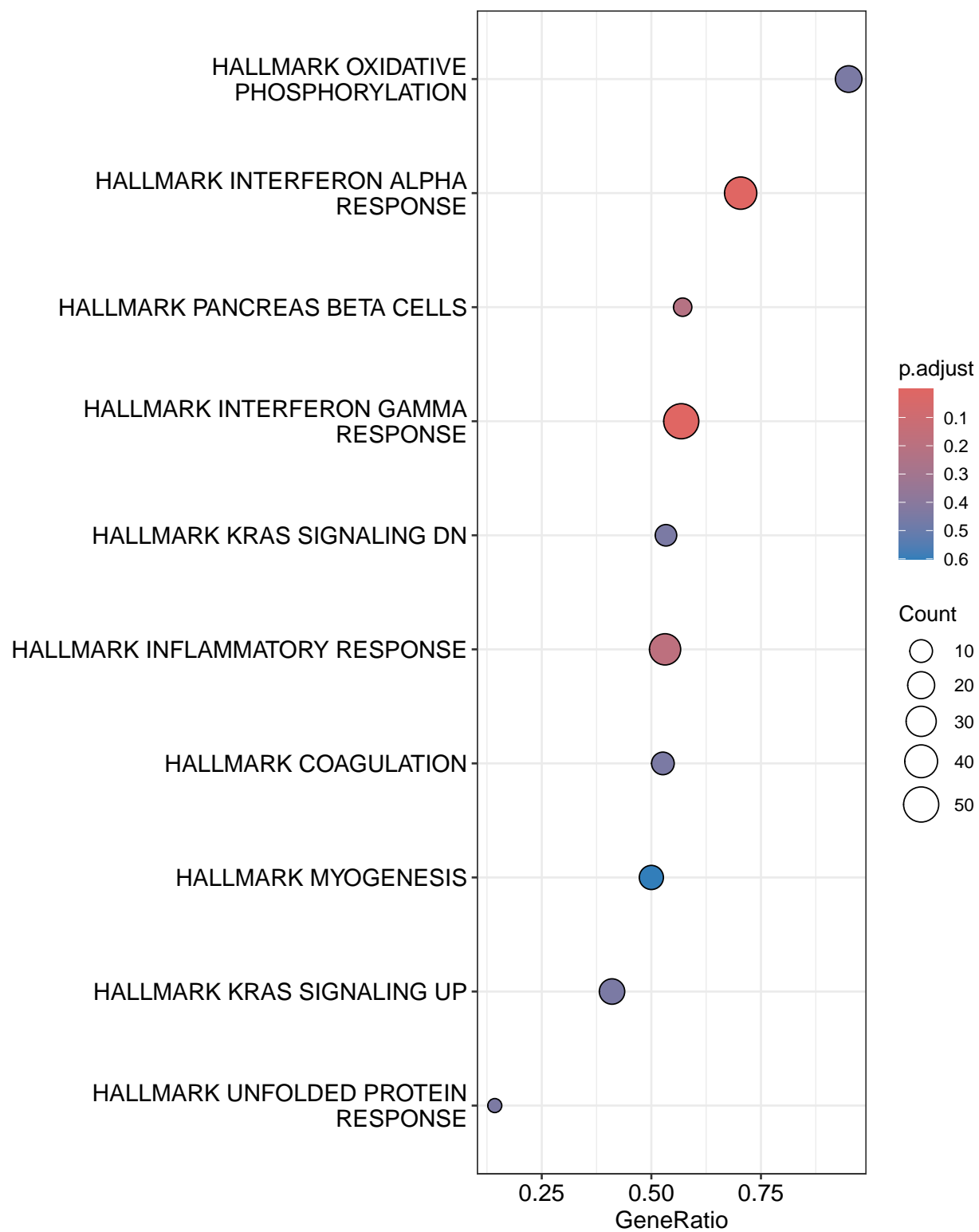
- These plots are helpful to demonstrate the distribution of the genes within the differential spectrum

- Secondly, it indirectly shows where are the genes responsible for the pathway direction located and how they contribute to enrichment
- Take your time to explore different plots in the book online and playig witht them to make them look clear in R (most of the are ggplot objects)
  - See the book: <https://yulab-smu.top/biomedical-knowledge-mining-book/enrichplot.html>

```
## Input data
#agsea

##
## Below plots are all parth of the ClusterProfiler package
##

## Dotplot
dotplot(object = agsea)
```



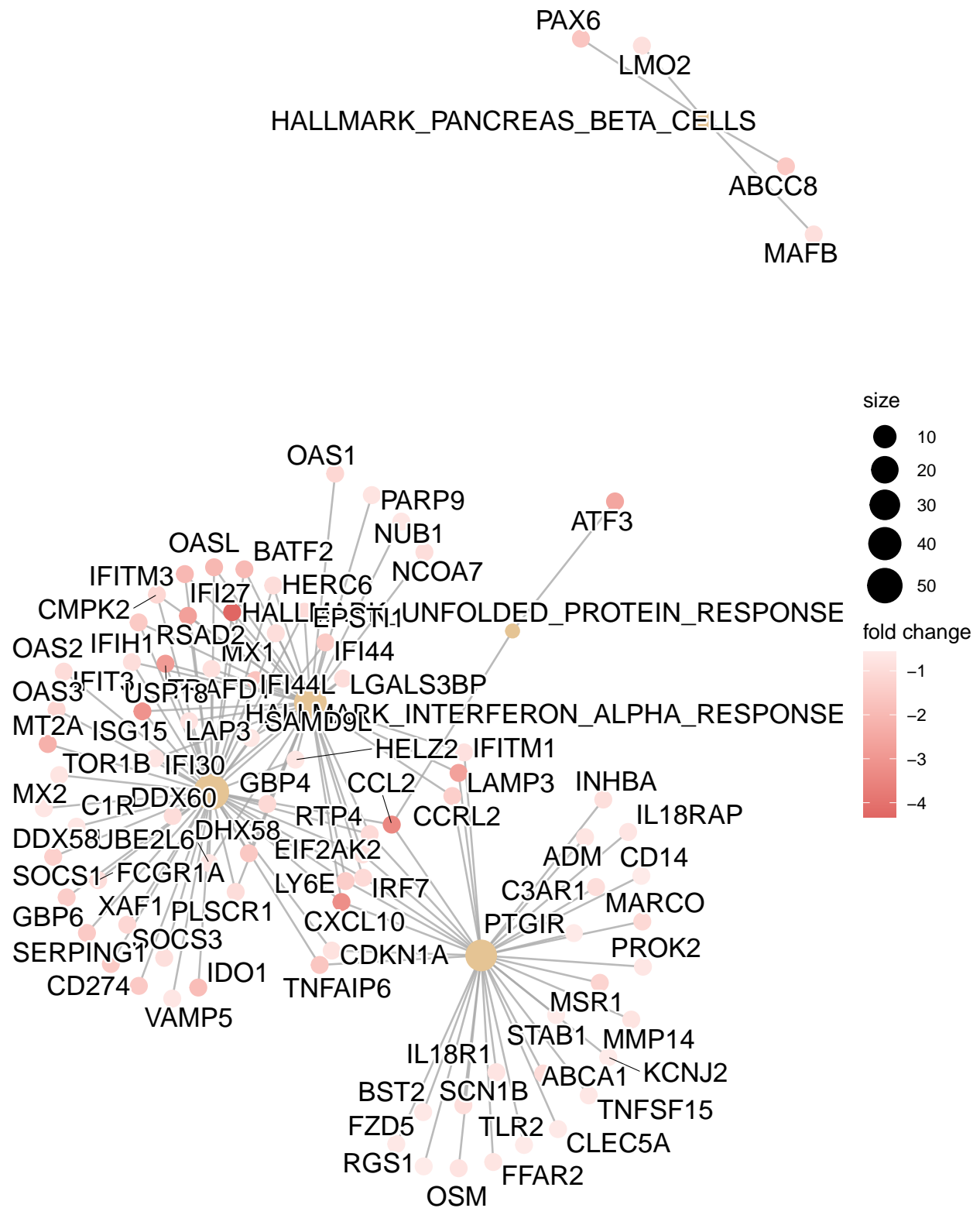
```
## Network plot
cnetplot(x = agsea, foldChange=geneVec, base.size =3 )
```

```
## Warning in cnetplot.enrichResult(x, ...): Use 'color.params = list(foldChange = your_value)' instead
## The foldChange parameter will be removed in the next version.

## Scale for size is already present.
## Adding another scale for size, which will replace the existing scale.

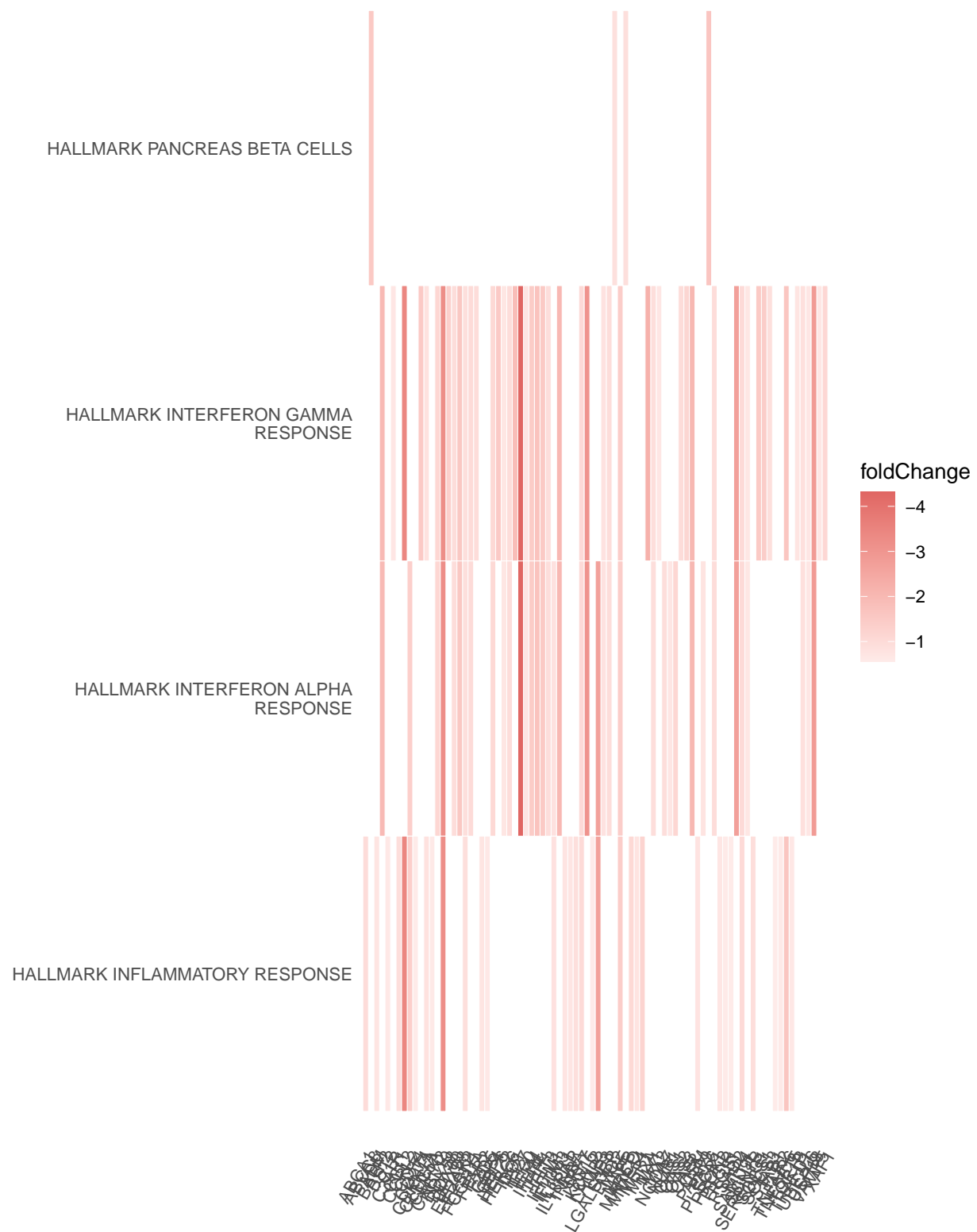
## Warning: ggrepel: 2 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```





```
##
## heatmap
## -> lets play with this one a little
```

```
##  
  
## basic  
heatplot(x = agsea, showCategory =4, foldChange = geneVec)
```

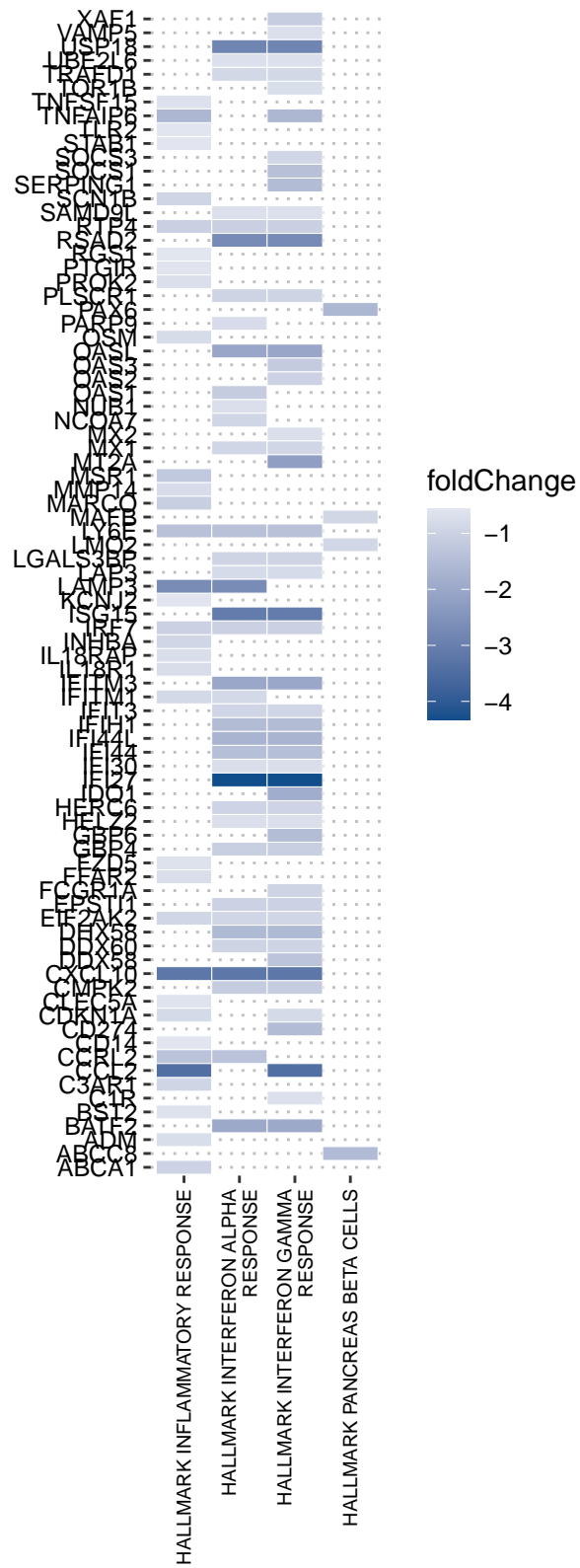


```
## Modified - This is a ggplot object
heatplot(x = agsea, showCategory =4, foldChange = geneVec) +
  theme_pubclean() +
```

```
scale_fill_gradient2(low = "dodgerblue4", mid = "white",high = "firebrick3") +  
  
coord_flip() +  
theme(aspect.ratio = 5,legend.position = "right",  
      axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1, size = 8, ),  
      axis.text = element_text(colour = "black"))
```

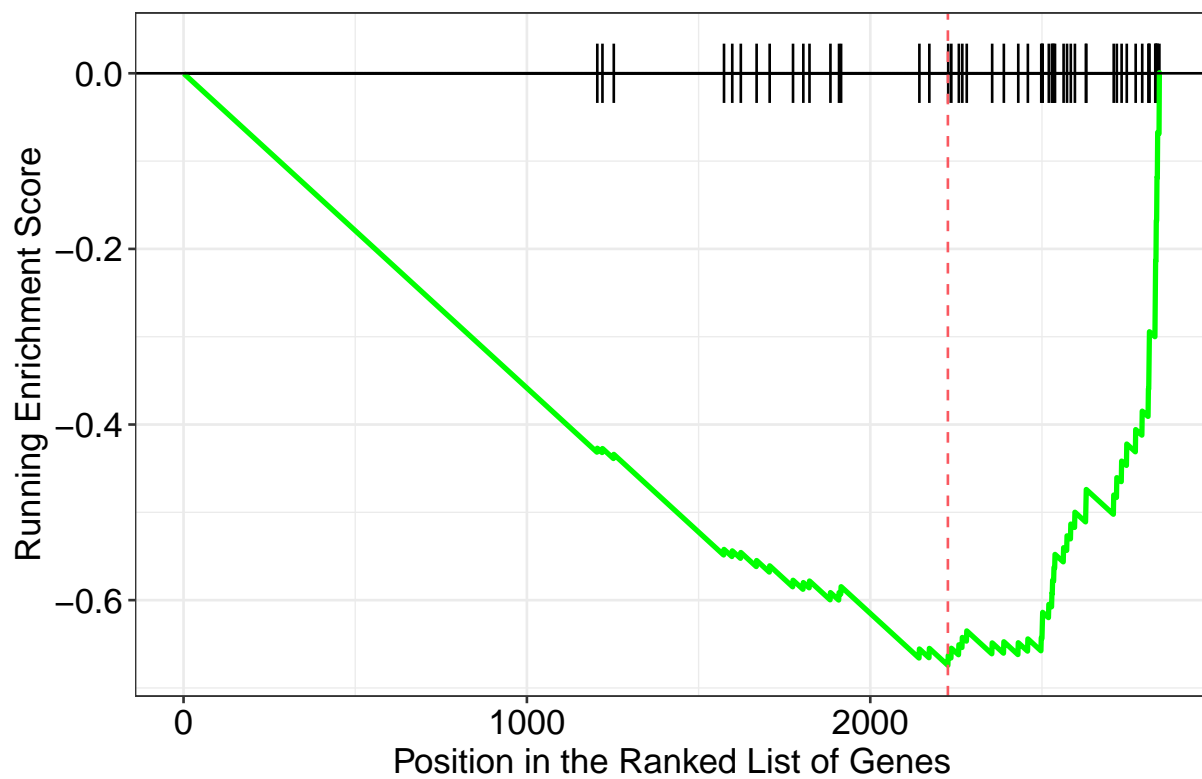
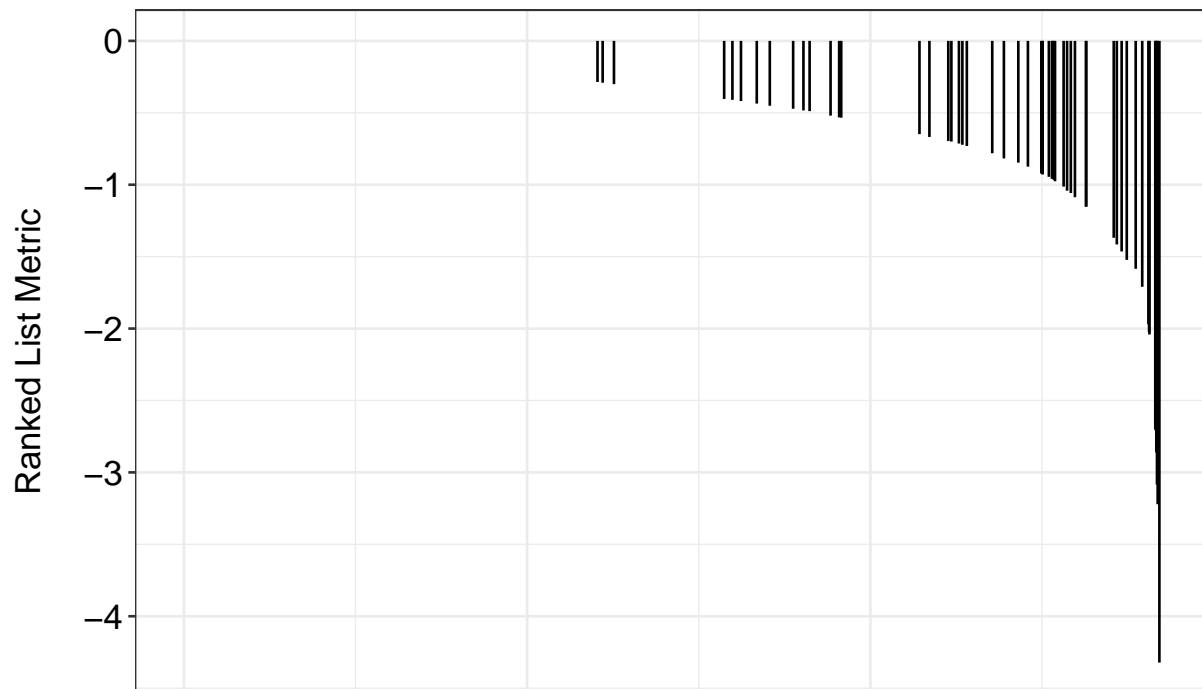
## Scale for fill is already present.

## Adding another scale for fill, which will replace the existing scale.

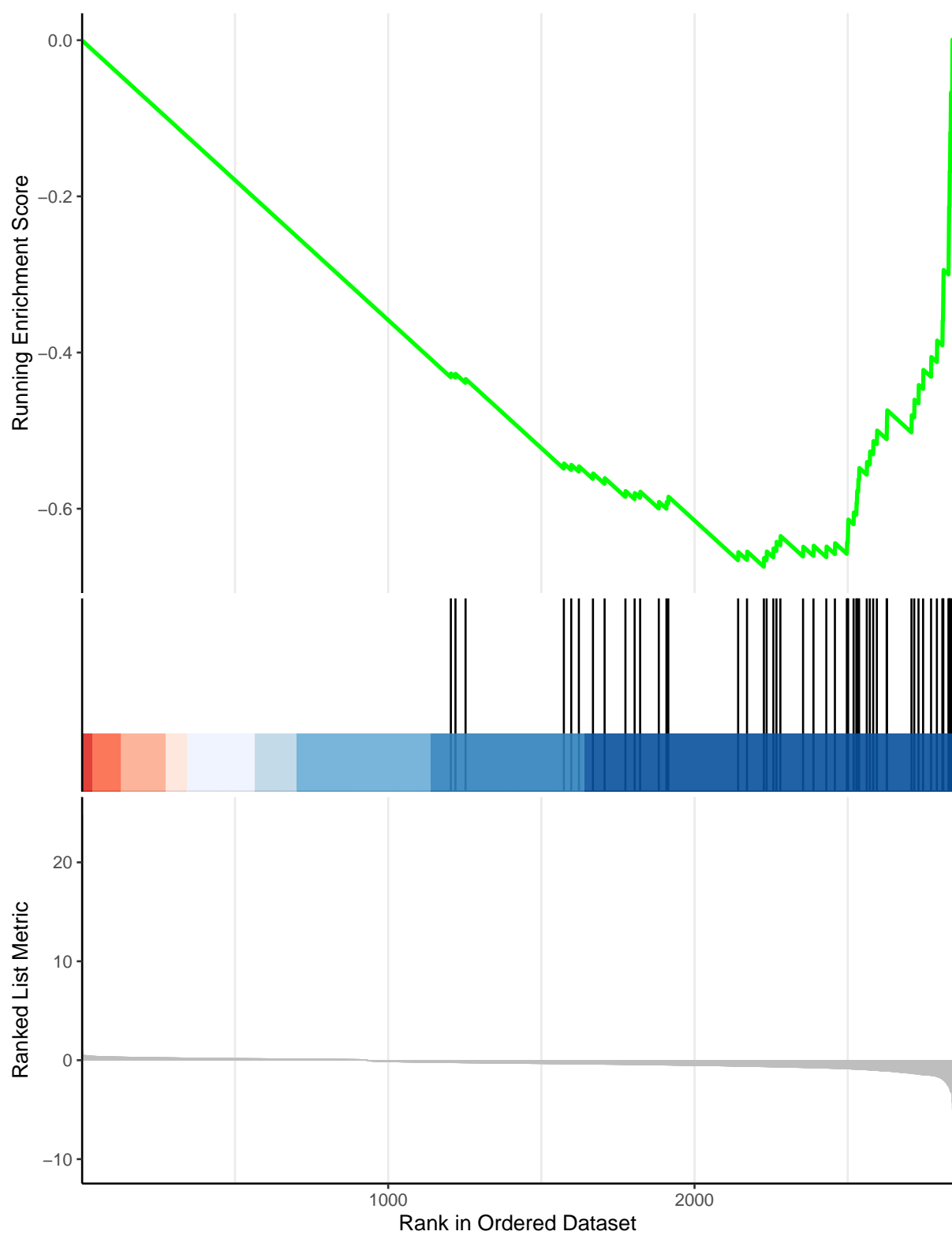


```
## Plot using pre-built function
?gseaplot2
```

```
## Leading edge - simple  
gseaplot(x = agsea, geneSetID = "HALLMARK_INTERFERON_ALPHA_RESPONSE")
```

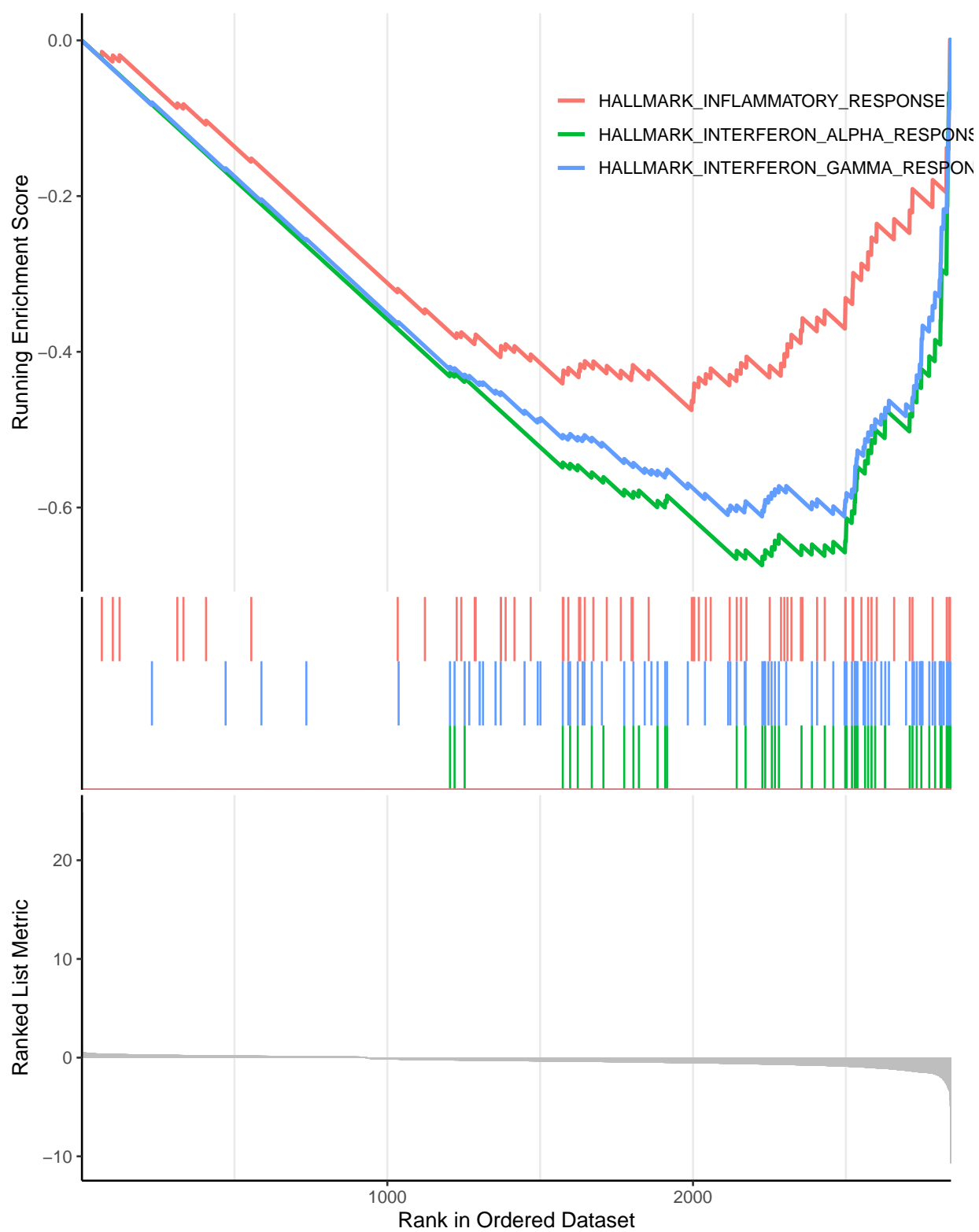


```
## Leading edge - more complicated
gseaplot2(x = agsea, geneSetID = "HALLMARK_INTERFERON_ALPHA_RESPONSE")
```



```
## Leading edge - multiple gene sets
gseaplot2(x = agsea, geneSetID = 1:3)
```





## NEXT

- Clustering heat maps
- Plotting your own heatmaps
- Basic Statistics
- Table1 of metadata