Introduction to Bulk RNAseq data analysis

Gene Set Testing for RNA-seq

Last modified: 19 Mar 2024

The list of differentially expressed genes is sometimes so long that its interpretation becomes cumbersome and time consuming. It may also be very short while some genes have low p-value yet higher than the given threshold.

A common downstream procedure to combine information across genes is gene set testing. It aims at finding pathways or gene networks the differentially expressed genes play a role in.

Various ways exist to test for enrichment of biological pathways. We will look into over representation and gene set enrichment analyses.

A gene set comprises genes that share a biological function, chromosomal location, or any other relevant criterion.

To save time and effort there are a number of packages that make applying these tests to a large number of gene sets simpler, and which will import gene lists for testing from various sources.

Today we will use `clusterProfiler` (https://yulab-smu.github.io/clusterProfiler-book/index.html).

Over-representation

Method

This method tests whether genes in a pathway are present in a subset of our data in a higher number than expected by chance (explanations derived from the clusterProfiler manual (https://yulab-smu.github.io/clusterProfiler-book/index.html)).

Genes in the experiment are split in two ways:

- annotated to the pathway or not
- differentially expressed or not

We can then create a contingency table with:

- rows: genes in pathway or not
- columns: genes differentially expressed or not

And test for independence of the two variables with the Fisher exact test.

`clusterProfiler`

`clusterprofiler` (Yu et al. 2012) supports direct online access of the current KEGG database (KEGG: Kyoto Encyclopedia of Genes and Genomes), rather than relying on R annotation packages. It also provides some nice visualisation options.

We first search the resource for mouse data:

```
library(tidyverse)
library(clusterProfiler)

search_kegg_organism('mouse', by='common_name')
```

```
##      kegg_code                          scientific_name
## 26        mmur                       Microcebus murinus
## 30         mmu                            Mus musculus
## 31        mcal                              Mus caroli
## 32        mpah                              Mus pahari
## 34        mcoc                          Mastomys coucha
## 40        pleu                       Peromyscus leucopus
## 50        plop        Perognathus longimembris pacificus
## 113       mmyo                             Myotis myotis
## 188       csti                          Colius striatus
## 5722       asf Candidatus Arthromitus sp. SFB-mouse-Japan
## 5723       asm   Candidatus Arthromitus sp. SFB-mouse-Yit
## 5724       aso    Candidatus Arthromitus sp. SFB-mouse-NL
##                                  common_name
## 26                            gray mouse lemur
## 30                                 house mouse
## 31                                Ryukyu mouse
## 32                                 shrew mouse
## 34                southern multimammate mouse
## 40                        white-footed mouse
## 50                      Pacific pocket mouse
## 113                   greater mouse-eared bat
## 188                        speckled mousebird
## 5722 Candidatus Arthromitus sp. SFB-mouse-Japan
## 5723   Candidatus Arthromitus sp. SFB-mouse-Yit
## 5724    Candidatus Arthromitus sp. SFB-mouse-NL
```

We will use the 'mmu' 'kegg_code'.

KEGG enrichment analysis

The input for the KEGG enrichment analysis is the list of gene IDs of significant genes.

We now load the R object keeping the outcome of the differential expression analysis for the d11 contrast.

```
shrink.d11 <- readRDS("RObjects/Shrunk_Results.d11.rds")
```

We will only use genes that have:

- an adjusted p-value (FDR, False Discovery Rate) of less than 0.05
- and an absolute fold change greater than 2.

We need to remember to eliminate genes with missing values in the FDR as a result of the independent filtering by DESeq2.

For this tool we need to use Entrez IDs, so we will also need to eliminate genes with a missing Entrez ID (NA values in the 'Entrez' column).

```
sigGenes <- shrink.d11 %>%
    drop_na(Entrez, padj) %>%
    filter(padj < 0.05 & abs(log2FoldChange) > 1) %>%
    pull(Entrez)

keggRes <- enrichKEGG(gene = sigGenes, organism = 'mmu')
```

```
## Reading KEGG annotation online: "https://rest.kegg.jp/link/mmu/pathway"...
```

```
## Reading KEGG annotation online: "https://rest.kegg.jp/list/pathway/mmu"...
```

```
as_tibble(keggRes)
```

```
## # A tibble: 74 × 11
##    category    subcategory ID    Description GeneRatio BgRatio   pvalue p.adjust
##    <chr>       <chr>       <chr> <chr>       <chr>     <chr>      <dbl>    <dbl>
##  1 Organismal… Immune sys… mmu0… Antigen pr… 39/344    87/9467 1.47e-33 3.46e-31
##  2 Human Dise… Infectious… mmu0… Epstein-Ba… 55/344    228/94… 1.21e-30 1.43e-28
##  3 Human Dise… Immune dis… mmu0… Graft-vers… 31/344    60/9467 2.71e-29 2.14e-27
##  4 Human Dise… Endocrine … mmu0… Type I dia… 32/344    67/9467 8.69e-29 5.13e-27
##  5 Cellular P… Transport … mmu0… Phagosome … 47/344    179/94… 5.41e-28 2.55e-26
##  6 Human Dise… Immune dis… mmu0… Allograft … 30/344    60/9467 8.14e-28 3.20e-26
##  7 Human Dise… Infectious… mmu0… Influenza … 45/344    173/94… 1.24e-26 4.18e-25
##  8 Environmen… Signaling … mmu0… Cell adhes… 44/344    179/94… 6.01e-25 1.77e-23
##  9 Human Dise… Infectious… mmu0… Leishmania… 28/344    70/9467 1.16e-22 3.03e-21
## 10 Human Dise… Cardiovasc… mmu0… Viral myoc… 31/344    92/9467 2.48e-22 5.85e-21
## # ℹ 64 more rows
## # ℹ 3 more variables: qvalue <dbl>, geneID <chr>, Count <int>
```

Visualise a pathway in a browser

`clusterProfiler` has a function `browseKegg` to view the KEGG pathway in a browser, highlighting the genes we selected as differentially expressed.

We will show one of the top hits: pathway 'mmu04612' for 'Antigen processing and presentation'.

```
browseKEGG(keggRes, 'mmu04612')
```

Visualise a pathway as a file

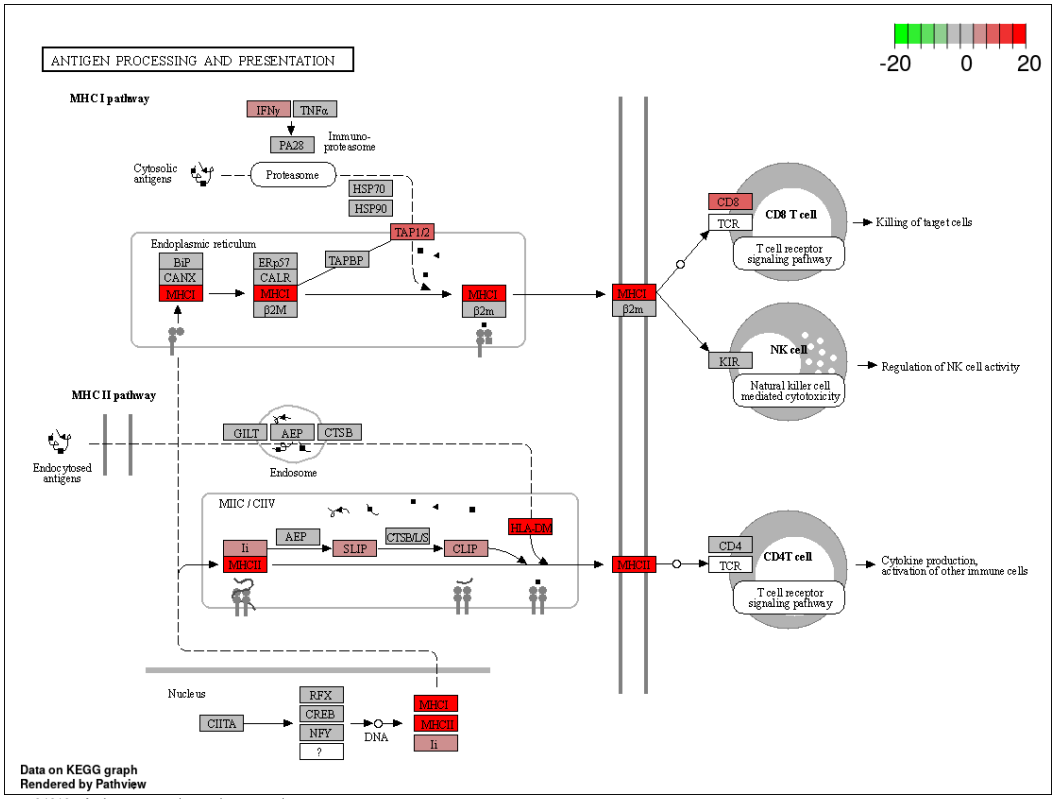
The package `pathview` (Luo et al. 2013) can be used to generate figures of KEGG pathways.

One advantage over the `clusterProfiler` browser method `browseKEGG` is that genes can be coloured according to fold change levels in our data. To do this we need to pass `pathview` a named vector of fold change values (one could in fact colour by any numeric vector, e.g. p-value).

The package plots the KEGG pathway to a `png` file in the working directory.

```
library(pathview)
logFC <- shrink.d11$log2FoldChange
names(logFC) <- shrink.d11$Entrez
pathview(gene.data = logFC,
         pathway.id = "mmu04612",
         species = "mmu",
         limit = list(gene=20, cpd=1))
```

mmu04612.pathview.png:



mmu04612 - Antigen processing and presentation

Exercise 1

1. Use `pathview` to export a figure for "mmu04659" or "mmu04658", but this time only use genes that are statistically significant at padj < 0.01

GO term enrichment analysis

`clusterProfiler` can also perform over-representation analysis on GO terms using the command `enrichGO` . For this analysis we will use Ensembl gene IDs instead of Entrez IDs and in order to do this we need to load another package which contains the mouse database called `org.Mm.eg.db` .

To run the GO enrichment analysis, this time we also need a couple of extra things. Firstly, we should provide a list of the 'universe' of all the genes in our DE analysis not just the ones we have selected as significant.

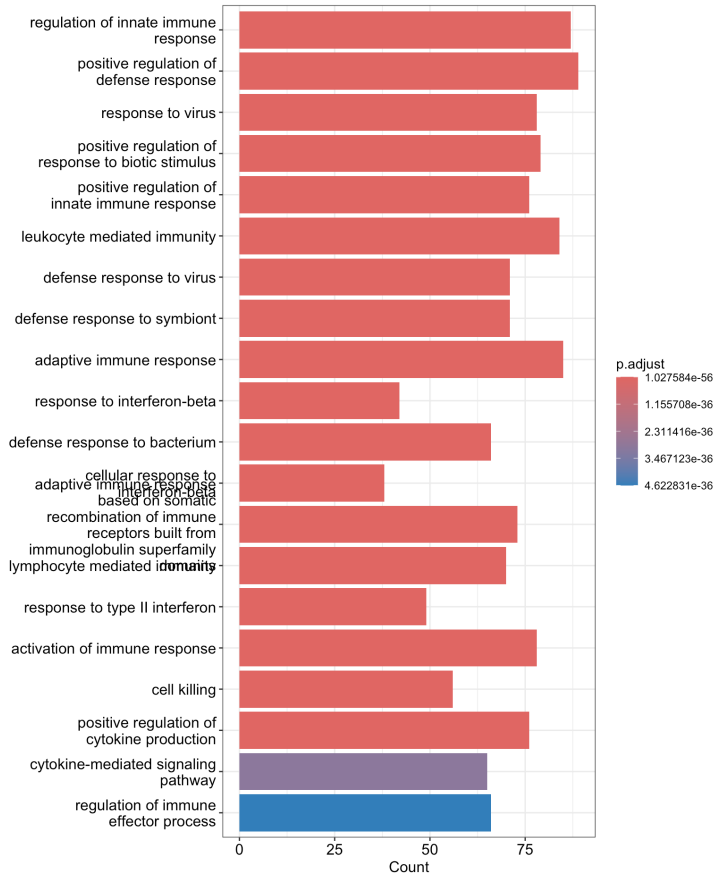
Gene Ontology terms are divided into 3 categories. - Metabolic Functions - Biological Processes - Cellular Components

For this analysis we will narrow our search terms in the 'Biological Processes' Ontology so we can add the parameter "BP" with the 'ont' argument (the default is Molecular Functions).

```
library(org.Mm.eg.db)

sigGenes_GO <-  shrink.d11 %>%
    drop_na(padj) %>%
    filter(padj < 0.01 & abs(log2FoldChange) > 2) %>%
    pull(GeneID)

universe <- shrink.d11$GeneID

ego <- enrichGO(gene          = sigGenes_GO,
                universe      = universe,
                OrgDb         = org.Mm.eg.db,
                keyType       = "ENSEMBL",
                ont           = "BP",
                pvalueCutoff  = 0.01,
                readable      = TRUE)
```

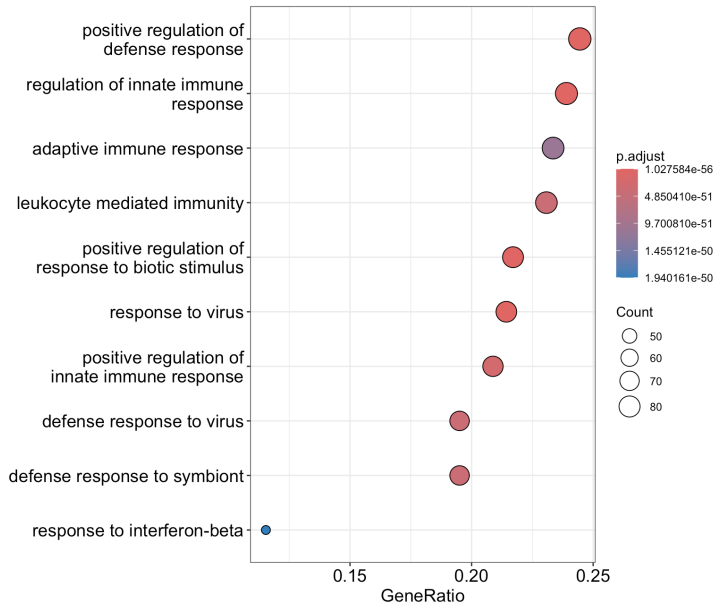
We can use the `barplot` function to visualise the results. Count is the number of differentially expressed in each gene ontology term.

```
barplot(ego, showCategory=20)
```



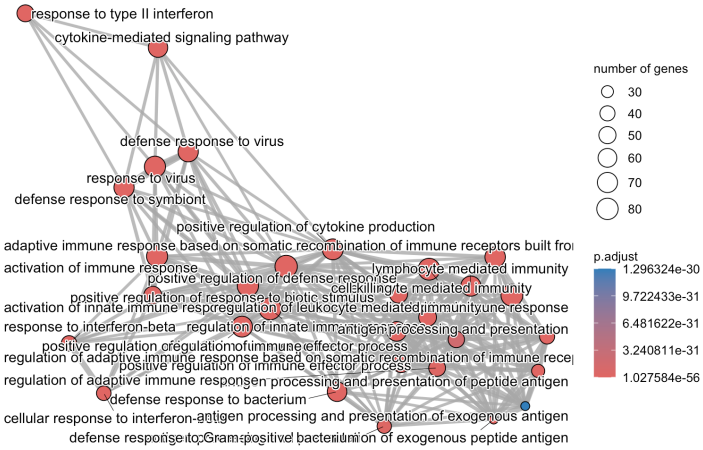
or perhaps the `dotplot` version is more informative. Gene ratio is Count divided by the number of genes in that GO term.

```
dotplot(ego, font.size = 14)
```



Another visualisation that can be nice to try is the `emapplot` which shows the overlap between genes in the different GO terms.

```
library(enrichplot)
ego_pt <- pairwise_termsim(ego)
emapplot(ego_pt, cex.params = list(category_label = 0.8))
```



GSEA analysis

Gene Set Enrichment Analysis (GSEA) identifies gene sets that are enriched in the dataset between samples (Subramanian et al. 2005).

The software is distributed by the Broad Institute (http://software.broadinstitute.org/gsea/index.jsp) and is freely available for use by academic and non-profit organisations. The Broad also provide a number of very well curated gene sets for testing against your data - the Molecular Signatures Database (MSigDB) (http://software.broadinstitute.org/gsea/msigdb/index.jsp).

These gene lists are made availalble for R in the Bioconductor package `msigdb` and the available dataset can be explored via `ExperimentHub`.

First, we need to locate the correct database and download the data.

```
library(msigdb)
library(ExperimentHub)
```

```
## Loading required package: AnnotationHub
```

```
## Loading required package: BiocFileCache
```

```
## Loading required package: dbplyr
```

```
##
## Attaching package: 'dbplyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     ident, sql
```

```
##
## Attaching package: 'AnnotationHub'
```

```
## The following object is masked from 'package:Biobase':
##
##     cache
```

```
eh = ExperimentHub()
query(eh , c('msigdb', 'mm', '2023'))
```

```
## ExperimentHub with 10 records
## # snapshotDate(): 2023-10-24
## # $dataprovider: Broad Institute, EBI
## # $species: Mus musculus, Homo sapiens
## # $rdataclass: GSEABase::GeneSetCollection, data.frame
## # additional mcols(): taxonomyid, genome, description,
## #   coordinate_1_based, maintainer, rdatadateadded, preparerclass, tags,
## #   rdatapath, sourceurl, sourcetype
## # retrieve records with, e.g., 'object[["EH8285"]]'
##
##            title
##   EH8285 | msigdb.v2022.1.mm.EZID
##   EH8286 | msigdb.v2022.1.mm.idf
##   EH8287 | msigdb.v2022.1.mm.SYM
##   EH8291 | msigdb.v2023.1.mm.EZID
##   EH8292 | msigdb.v2023.1.mm.idf
##   EH8293 | msigdb.v2023.1.mm.SYM
##   EH8297 | msigdb.v7.5.1.mm.EZID
##   EH8298 | msigdb.v7.5.1.mm.idf
##   EH8299 | msigdb.v7.5.1.mm.SYM
##   EH8300 | imex_hsmm_0722
```

The most recent available release of MSigDb is “msigdb.v2023.1”, so we’ll download this one. We have the option to use Entrez IDs or gene symbols. As we already have gene symbols in our annotation, we’ll use these. We could, on the other hand, choose to map our Ensembl IDs to Entrez IDs and use those instead.

```
msigdb.mm <- getMsigdb(org = 'mm', id = 'SYM', version = '2023.1')
```

```
## see ?msigdb and browseVignettes('msigdb') for documentation
```

```
## loading from cache
```

```
## require("GSEABase")
```

```
msigdb.mm
```

```
## GeneSetCollection
##   names: 10qA1, 10qA2, ..., ZZZ3_TARGET_GENES (45953 total)
##   unique identifiers: Epm2a, Esr1, ..., Gm52481 (56208 total)
##   types in collection:
##     geneIdType: SymbolIdentifier (1 total)
##     collectionType: BroadCollection (1 total)
```

```
listCollections(msigdb.mm)
```

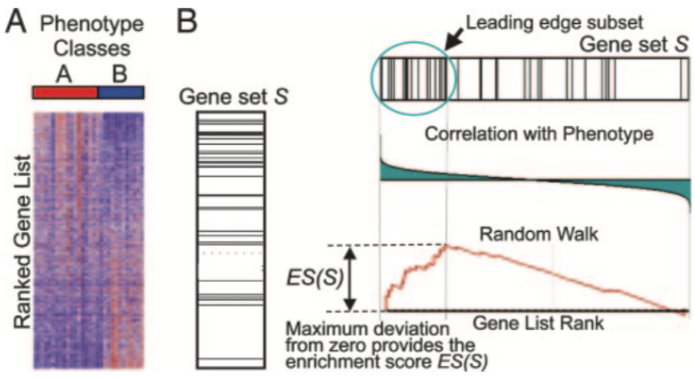
```
## [1] "c1" "c3" "c2" "c8" "c6" "c7" "c4" "c5" "h"
```

Method

The analysis is performed by:

1. ranking all genes in the data set
2. identifying in the ranked data set the rank positions of all members of the gene set
3. calculating an enrichment score (ES) that represents the difference between the observed rankings and that which would be expected assuming a random rank distribution.

The article describing the original software is available here (http://www.pnas.org/content/102/43/15545.long), while this commentary on GSEA (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1266131/) provides a shorter description.



We will use `clusterProfiler`'s `GSEA` (http://yulab-smu.top/clusterProfiler-book/chapter2.html#gene-set-enrichment-analysis) package (Yu et al. 2012) that implements the same algorithm in R.

Rank genes

We need to provide `GSEA` with a vector containing values for a given gene mtric, e.g. log(fold change), sorted in decreasing order.

To start with we will simply use a rank the genes based on their fold change.

We must exclude genes with no Ensembl ID.

Also, we should use the shrunk LFC values.

```
rankedGenes <- shrink.d11 %>%
  drop_na(GeneID, padj, log2FoldChange) %>%
  mutate(rank = log2FoldChange) %>%
  arrange(desc(rank)) %>%
  pull(rank, Symbol)
head(rankedGenes)
```

```
##   Cxcl10      Ubd    Gbp10     Saa3    Cxcl9   Cxcl11
## 8.439020 8.307955 7.818902 7.787435 7.783766 7.669303
```

Load pathways

For `clusterProfiler` we need the genes and genesets to be in the form of is a `tibble` with information on each {gene set; gene} pair in the rows.

```
hallmarks = subsetCollection(msigdb.mm, 'h')
msigdb_ids = geneIds(hallmarks)

term2gene <- enframe(msigdb_ids, name = "gs_name", value = "symbol") %>%
  unnest(symbol)

head(term2gene)
```

```
## # A tibble: 6 × 2
##   gs_name               symbol
##   <chr>                 <chr>
## 1 HALLMARK_ADIPOGENESIS Abca1
## 2 HALLMARK_ADIPOGENESIS Abca4
## 3 HALLMARK_ADIPOGENESIS Abca7
## 4 HALLMARK_ADIPOGENESIS Abca13
## 5 HALLMARK_ADIPOGENESIS Abca12
## 6 HALLMARK_ADIPOGENESIS Abca17
```

Conduct analysis

Arguments passed to `GSEA` include:

- ranked genes
- pathways
- gene set minimum size
- gene set maximum size

```
gseaRes <- GSEA(rankedGenes,
                TERM2GENE = term2gene,
                pvalueCutoff = 1.00,
                minGSSize = 15,
                maxGSSize = 500)
```

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

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

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

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

Let's look at the top 10 results.

```
as_tibble(gseaRes) %>%
  arrange(desc(abs(NES))) %>%
  top_n(10, wt=-p.adjust) %>%
  dplyr::select(-core_enrichment) %>%
  mutate(across(c("enrichmentScore", "NES"), ~round(.x, digits=3))) %>%
  mutate(across(c("pvalue", "p.adjust", "qvalue"), scales::scientific))
```

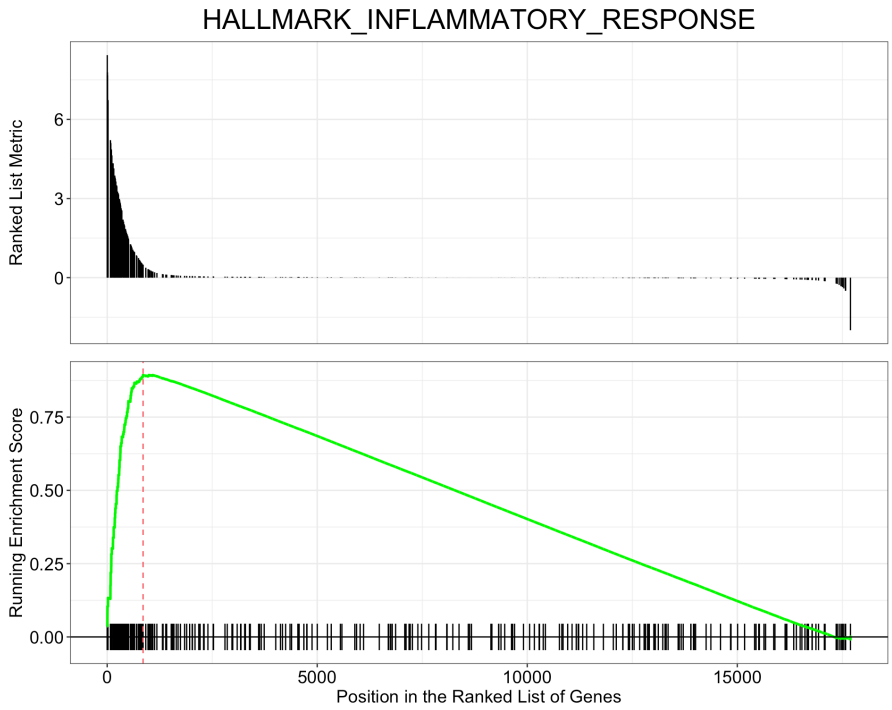
	ID	Description	setSize	enrichmentScore	NES	pvalue	p.adjust	qvalue	rank
1	HALLMARK_INTERFERON_ALPHA_RESPONSE	HALLMARK_INTERFERON_ALPHA_RESPONSE	154	0.954	1.395	1.00e-10	1.67e-09	1.30e-09	766
2	HALLMARK_INTERFERON_GAMMA_RESPONSE	HALLMARK_INTERFERON_GAMMA_RESPONSE	283	0.947	1.387	1.00e-10	1.67e-09	1.30e-09	893
3	HALLMARK_ALLOGRAFT_REJECTION	HALLMARK_ALLOGRAFT_REJECTION	285	0.929	1.361	1.00e-10	1.67e-09	1.30e-09	859
4	HALLMARK_IL6_JAK_STAT3_SIGNALING	HALLMARK_IL6_JAK_STAT3_SIGNALING	108	0.925	1.357	3.18e-07	2.27e-06	1.77e-06	853
5	HALLMARK_INFLAMMATORY_RESPONSE	HALLMARK_INFLAMMATORY_RESPONSE	289	0.895	1.311	2.60e-10	3.25e-09	2.53e-09	854
6	HALLMARK_IL2_STAT5_SIGNALING	HALLMARK_IL2_STAT5_SIGNALING	293	0.888	1.302	3.15e-09	3.15e-08	2.45e-08	833
7	HALLMARK_TNFA_SIGNALING_VIA_NFKB	HALLMARK_TNFA_SIGNALING_VIA_NFKB	263	0.887	1.298	3.46e-08	2.88e-07	2.24e-07	1110
8	HALLMARK_APOPTOSIS	HALLMARK_APOPTOSIS	220	0.873	1.275	8.99e-06	5.00e-05	3.89e-05	1110
9	HALLMARK_COMPLEMENT	HALLMARK_COMPLEMENT	287	0.868	1.271	2.52e-06	1.57e-05	1.23e-05	1123
10	HALLMARK_COAGULATION	HALLMARK_COAGULATION	186	0.864	1.26	3.33e-04	1.66e-03	1.30e-03	1095

Enrichment score plot

The enrichment score plot displays along the x-axis that represents the decreasing gene rank:

- genes involved in the pathway under scrutiny: one black tick per gene in the pathway (no tick for genes not in the pathway)
- the enrichment score: the green curve shows the difference between the observed rankings and that which would be expected assuming a random rank distribution.

```
gseaplot(gseaRes,
         geneSetID = "HALLMARK_INFLAMMATORY_RESPONSE",
         title = "HALLMARK_INFLAMMATORY_RESPONSE")
```



Remember to check the GSEA article (http://www.pnas.org/content/102/43/15545.full) for the complete explanation.

Exercise 2

Another common way to rank the genes is to order by pvalue while sorting so that upregulated genes are at the start and downregulated at the end. You can do this combining the sign of the fold change and the pvalue.

1. Rank the genes by statistical significance - you will need to create a new ranking value using `-log10(pvalue) * sign(log2FoldChange)`.
2. Run GSEA using the new ranked genes and the H pathways.
3. Conduct the same analysis for the day 33 Infected vs Uninfected contrast.

References

Luo, Weijun, Brouwer, and Cory. 2013. "Pathview: An r/Bioconductor Package for Pathway-Based Data Integration and Visualization." *Bioinformatics* 29 (14): 1830–31. https://doi.org/10.1093/bioinformatics/btt285 (https://doi.org/10.1093/bioinformatics/btt285).

Subramanian, Aravind, Pablo Tamayo, Vamsi K. Mootha, Sayan Mukherjee, Benjamin L. Ebert, Michael A. Gillette, Amanda Paulovich, et al. 2005. "Gene Set Enrichment Analysis: A Knowledge-Based Approach for Interpreting Genome-Wide Expression Profiles." *Proceedings of the National Academy of Sciences* 102 (43): 15545–50. https://doi.org/10.1073/pnas.0506580102 (https://doi.org/10.1073/pnas.0506580102).

Yu, Guangchuang, Li-Gen Wang, Yanyan Han, and Qing-Yu He. 2012. "clusterProfiler: An r Package for Comparing Biological Themes Among Gene Clusters." *OMICS: A Journal of Integrative Biology* 16 (5): 284–87. https://doi.org/10.1089/omi.2011.0118 (https://doi.org/10.1089/omi.2011.0118).