FGT T5: Functional Enrichment Analysis

Getting Started

See
Goeman JJ, Bühlmann P. Analyzing gene expression data in terms of gene sets:methodological issues. Bioinformatics. 2007 Apr 15;23(8):980-7. PubMed PMID: 17303618
See also https://support.bioconductor.org/p/34405/

When we complete a differential gene expression analysis experiment we have often generated a long list of genes ranked by their log odds score or their multiple testing adjusted p-value. What we would like to know is -are the genes that are differentially expressed enriched (or depleted) in some functional properties? For example, a set of genes might be involved in the same biological process eg in cell division, might change in every experiment where cell division rates change between the treated and control. If we can identify these genes, they tell us about the processes that differ between our treated and control samples. To identify these so-called functional signatures in the data, we compare our differentially expressed gene list to reference lists of genes that have previously been identified as changing during a particular process. If we identify a statistically significant overlap between our list and the reference list, then this suggests that these lists share a biological feature in common.

The most common (naïve) approach to functional enrichment is to use a simple statistical test such as the hypergeometric distribution to model the probabilities of encountering a particular percentage overlap between a query gene list and a reference list. So, for example, if 10 % of all genes are transcription factors, our differential gene list has 40% transcription factors, is this an enrichment over the number of transcription factors we expect to draw if we select genes randomly. In practice we compare our list of differentially expressed genes to a database of functional signatures- so we need also to account for multiple testing.

Simple enrichment approaches make a number of assumptions, such as that genes change between treatment and control independently of each other. Consider what might happen if a set of genes are known to be part of one complex- compared to another complex which might contain only a single gene product. When the large complex changes number of genes from the complex might be co-regulated and this leads to some lists appearing over-important. Most tests also does not account for genes that are on many lists–in this case we don't really know which of the processes that these genes are involved in are changing in the experiment. Ideally, we also want to use a test that uses a rank or against the statistical model of differential expression rather than a list generated by applying an artificial threshold to our differential gene list. Unfortunately, when enrichment analysis is performed poorly it often under-estimates p-values and over-estimates statistical significance of biological relationships. This results in lots of spurious gene enrichments that then cause confusion in the scientific literature.

Common Limma tools for R enrichment analysis include

- Mroast – a “self-contained test”
- Camera- a competitive test
- Romer- a GSEA type test that is a commonly used competitive test

Molecular Signatures Database

This is a collection of gene list signatures split into different groups- see http://software.broadinstitute.org/gsea/msigdb/index.jsp for more details.

These signatures are also available for mouse and human in a form that can be loaded into R easily and we will use these in this tutorial (see http://bioinf.wehi.edu.au/MSigDB/index.html).

In this practical we will use Hallmark Gene Signatures which are “coherently expressed signatures derived by aggregating many MSigDB gene sets to represent well-defined biological states or processes.” Note that these signatures are just collections of lists of IDs- Entrez gene IDs in this case. So we need to match these identifiers to Affymetrix gene array Probesets in order to compare lists. We use annotation maps to store Entrez IDs within the expression set we generated last week. This is then used to map to the expression signatures.

Load the Tutorial Files

- Login to your account
- Create a directory for this week’s files e.g. `mkdir tutorial5`
- Change working directory to this folder e.g. `cd tutorial5`
- Copy the tutorial data into this folder e.g.
  ```
  cp /home/shared_files/enrichment.Rdata .
  cp /home/shared_files/MSigDB/Mm.h.all.v7.1.entrez.rds .
  ```
- Start R

```
#load the required libraries for the tutorials...
library(affy)
library(limma)
library(mouse4302.db)
library(annotate)
```

Now load the R data objects from the saved file

```
#check the tutorial data has loaded into the current
#working directory.
dir()
#You now need to load the .Rdata file into
#your workspace to recover the saved files

load("enrichment.Rdata")

#Note: In theory you could use
#load("Mm.h.all.v7.1.entrez.rds")
#But load of this type of file is not supported by all
#versions of R- so we use readRDS instead

Mm.H <- readRDS("Mm.h.all.v7.1.entrez.rds")

#Check that you have the required objects
ls()
```

This creates some example results files from the previous statistical analysis class and also adds the Mm.H object which contains part of the molecular signatures database.

We will use the same data as last week- but here we just need to use the pre-existing design and contrast matrices

Name	Filename	Description
ESC.1	GSM272753.CEL	Embryonic Stem cells sample 1
ESC.2	GSM272836.CEL	Embryonic Stem cells sample 2
ESC.3	GSM272837.CEL	Embryonic Stem cells sample 3
iPSC2.2	GSM272839.CEL	Induced pluripotent stem (iPS) cells (Oct4, Klf4) sample 2
iPSC2.3	GSM272846.CEL	Induced pluripotent stem (iPS) cells (Oct4, Klf4) sample 3
NSC.1	GSM272847.CEL	Neural stem cells (NSC) sample 2
NSC.2	GSM272848.CEL	Neural Stem cells (NSC) sample 3
iPSC2.1	GSM272890.CEL	Induced pluripotent stem (iPS) cells (Oct4, Klf4) sample 1
iPSC4.1	GSM279200.CEL	Induced pluripotent stem (iPS) cells (Oct4, Sox2, c-Myc, Klf4) sample 1
iPSC4.2	GSM279201.CEL	Induced pluripotent stem (iPS) cells (Oct4, Sox2, c-Myc, Klf4) sample 2
iPSC4.3	GSM279202.CEL	Induced pluripotent stem (iPS) cells (Oct4, Sox2, c-Myc, Klf4) sample 3

Annotating the Expression Data with Entrez (and other useful) IDs

Here we add extra annotation data to the expression table so we can use this to map to the gene signatures.

```
#Show the full contents of the annotation package
ls("package:mouse4302.db")

#Show the annotation keys in this database
keytypes(mouse4302.db)
```

```
#Mapping from
#
#http://genomicsclass.github.io/book/pages/mapping_features.ht
#ml
sampleNames(eset)
```

Annotate the Expression Data

```
#Here we select from the annotation a number of keys with the
primary key being PROBEID

res <- select(mouse4302.db, keys = rownames(eset), columns =
c("ENTREZID", "ENSEMBL","SYMBOL"), keytype="PROBEID")

#View the top of the table

head(res)

#find the index of each row of the expression set in the
#annotation object res

idx <- match(rownames(eset), res$PROBEID)

#Use the index to set the phenotypic data in the ExpressionSet

fData(eset) <- res[idx, ]

head(fData(eset), 10)

#Find all rows that don’t have an EntrezID and remove then

eset_t<-eset[is.na(fData(eset)$ENTREZID)==0,]
```

Statistical Enrichment Analysis Using Limma Design & Contrast Matrices

To run the Limma differential analysis we generated a design matrix and a contrast matrix. We give the enrichment analysis the design and one contrast.

```
Design

     ESC  iPS2  NSC  iPS4
1     1    0     0    0
2     1    0     0    0
3     1    0     0    0
4     0    1     0    0
5     0    1     0    0
6     0    0     1    0
7     0    0     1    0
8     0    1     0    0
9     0    0     0    1
10    0    0     0    1
11    0    0     0    1
```

The contrastmatrix object specifies the comparisons that will be made.

```
Levels ESC - iPS2 ESC - NSC ESC - iPS4
  ESC           1          1           1
  iPS2         -1          0           0
  NSC           0         -1           0
  iPS4          0          0          -1
```

Run the Enrichment Analysis Using MRoast or Camera or Romer

We first need to convert the lists of IDs in the signatures database into an index into the expression table- this allows efficient compute of the results. Then we run the analysis and save the results. Common packages for R enrichment analysis include

- Mroast – a "self-contained test"
- Camera- a competitive test
- Romer- a GSEA type test that is a commonly used competitive test

To run these commands we run mroast or camera or romer.

```
#convert to indexes

H.indices <- ids2indices(Mm.H,fData(eset_t)$ENTREZID)
```

```
#Pick the most suitable enrichment analysis tool to find
#enrichment signatures in the data and run this tool So:-if
#you want to run mroast

results <-mroast(eset_t,

                 index=H.indices,

                 design=design,

                 contrast=contrastmatrix[,1],

                 adjust.method = "BH")

#if you want to run camera

results <-camera(eset_t,

                index=H.indices,

                design=design,

                contrast=contrastmatrix[,1])

#if you want to run romer (takes 5mins or so)

results <-romer(eset_t,

                index=H.indices,

                design=design,

                contrast=contrastmatrix[,1] )

#View the results

results

#Use help for other parameters. Note we might decide to use
#exactly the same model as our differential gene analysis for
#the enrichment analysis- in this case we can extract it from
#the fit

#sv <- squeezeVar(fit$sigma^2,df=fit$df.residual)
```

Writing a Summary Table

```
write.table(results,"enrichment.txt",sep="\t")

#You can then examine the results in "enrichment.txt".  It is
a text file.  It can be downloaded to view in a spreadsheet
such as Excel.
```

Questions

- How do you interpret the output from this programme
- If you have time repeat the analysis with other signatures from the molecular signatures database. These are all in `/home/shared_files/MSigDB/`
- Why did you pick the enrichment analysis tool you used for this data?
- For the R code where we run one of the functions with the labels "if you want to run…", there are three code blocks (one for each enrichment analysis) that are almost identical. This code duplication introduces the risk of errors in the code. How could you change the R code to reduce the amount of code duplication?