Lecture 22

Gene Ontology Analysis – R code

MCB 416A/516A
Statistical Bioinformatics and Genomic Analysis

Prof. Lingling An
Univ of Arizona

Last time

- Introduction to Gene Ontology
- Gene set enrichment

Outline

- R packages for GO analysis
- R code in topGO

- Online softwares
  - DAVID (http://david.abcc.ncifcrf.gov/)
  - Gostat (www.***gostat****.wehi.edu.au/)*
  - …
- R packages
  - topGO
  - Gostat
  - …

Step1: Preparation

```
source("http://www.bioconductor.org/biocLite.R")
biocLite("topGO")
biocLite("ALL")

library(topGO)
library(ALL)

data(ALL)
```

When the topGO package is loaded three new environments GOBPTerm, GOMFTerm and GOMFTerm are created and binded to the package environment. These environments are build based on the GOTERM environment from package GO. They are used for fast recovering of the information specific to each ontology. In order to access all GO groups that belong to a specific ontology, e.g. Biological Process (BP), one can type:

```
BPterms <- ls(GOBPTerm)


MFterms <- ls(GOMFTerm)    ### for MF GO terms
CCterms <- ls(GOCCTerm)    ### for CC GO terms
```

load the annotation data

##The chip used for the experiment is HGU95aV2 Affymetrix

```
biocLite("hgu95av2.db")

library(hgu95av2.db)
```

Gene filtering

remove genes with low expression value and genes which might have very small variability across the samples using package "genefilter"

```
biocLite("genefilter")
library(genefilter)
f1 <- pOverA(0.25, log2(100))
f2 <- function(x) (IQR(x) > 0.5)
ff <- filterfun(f1, f2)
eset <- ALL[genefilter(ALL, ff), ]
```

The filter selects only 2400 probesets out of 12625 probesets available on the hgu95av2 array

Step 2: Creating a topGOdata object

This object will contain all information necessary for the GO analysis, namely the gene list, the list of interesting genes, the scores of genes (if available) and the part of the GO ontology (the GO graph) which needs to be used in the analysis.

need to define the set of genes that are to be annotated with GO terms. Usually, one starts with all genes present on the array. In our case we start with 2400 genes, genes that were not removed by the filter.

```
geneNames <- featureNames(eset)
length(geneNames)
```

2.1: Using score to determine a list of interesting gene

P-value for the t-test to discriminate between ALL cells delivered from either B-cell or T-cell precursors (95 B-cell ALL samples and 33 T-cell ALL samples).

```
>y = as.integer(sapply(eset$BT, function(x)
  return(substr(x, 1, 1) == "T")))
>table(y)
 0  1
95 33
>library(multtest)
>geneList = getPvalues(exprs(eset), classlabel = y,
  alternative = "greater", correction="BH")
> hist(geneList, br = 50)
```

2.1: Using score to determine a list of interesting gene -2

```
topDiffGenes <- function(allScore) {
  return(allScore < 0.0001)
 }
```

This function selects genes based on their scores (in our case the adjusted p-values) and returns a logical vector specifying which gene is selected and which not.

```
 x <- topDiffGenes(geneList)
sum(x)
```

2.2: build topGOdata object

```
>sampleGOdata <- new("topGOdata", ontology = "BP",
  allGenes = geneList, geneSel = topDiffGenes,
  description = "GO analysis of ALL data based on
  diff. expression.",  annot = annFUN.db, affyLib =
  "hgu95av2.db")


> sampleGOdata
```

Step 3: Running the desired tests

#Once we have an object of class topGOdata we can start with the enrichment analysis. Since for each gene we have a score and there is also a procedure to select interesting genes based on the scores we will use two types of test statistics:

#Fisher exact test and Kolmogorov-Smirnov (KS) test

#For KS, there are the classic and the elim methods.

```
>resultFisher <- runTest(sampleGOdata, algorithm =
  "classic", statistic = "fisher")
>resultKS <- runTest(sampleGOdata, algorithm =
  "classic", statistic = "ks")
>resultKS.elim <- runTest(sampleGOdata, algorithm =
  "elim", statistic = "ks")
```

To look at the results of significant GO terms, put all resulting p-values into a list. Then we can use the GenTable function to generate a table with the results.

```
>allRes <- GenTable(sampleGOdata, classicFisher =
  resultFisher, classicKS = resultKS, elimKS =
  resultKS.elim, orderBy = "elimKS", topNodes = 10)
> allRes
```

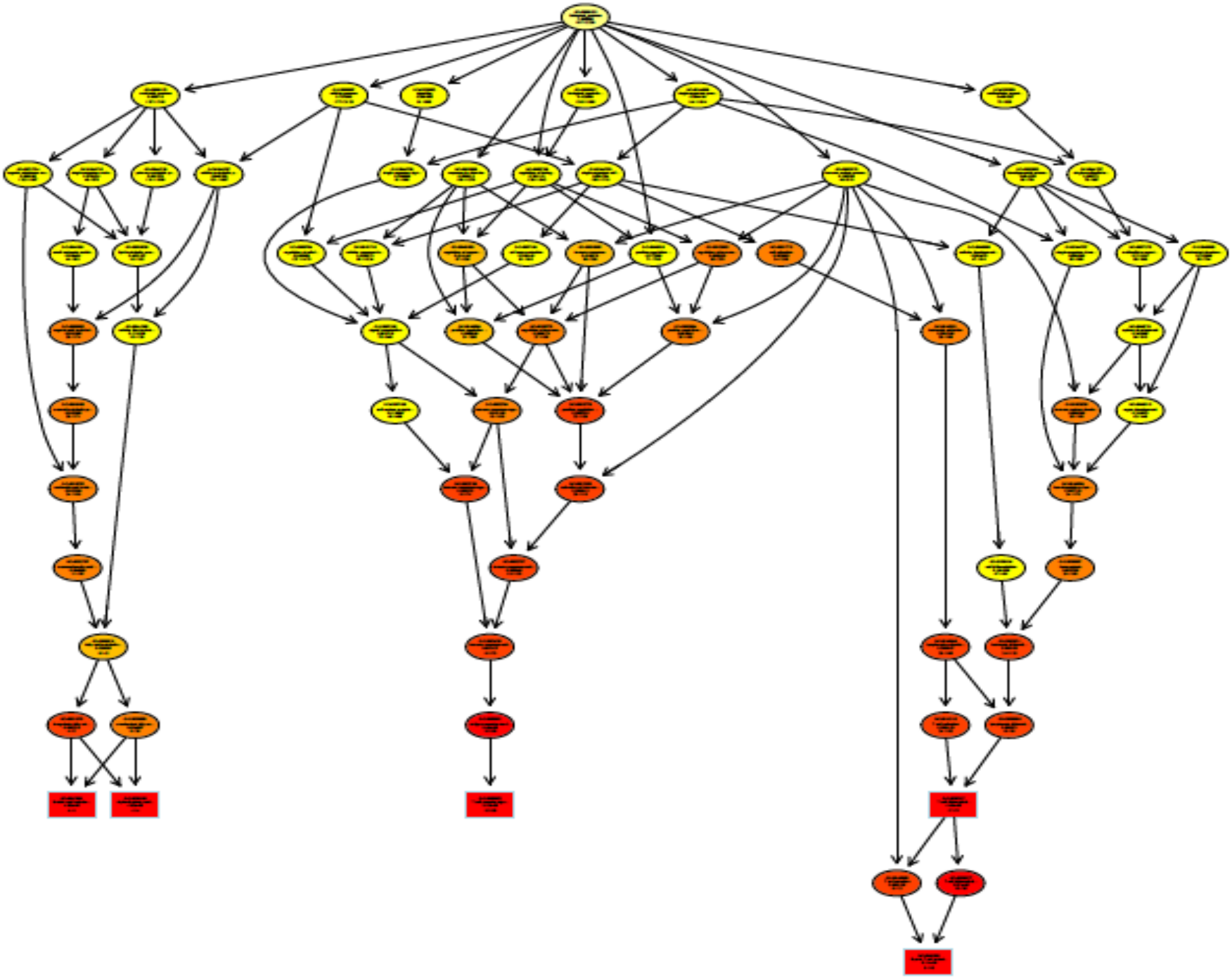
Optional part: graphs by Rgraphviz

Investigating how the significant GO terms are distributed over the GO graph

```
> biocLite("Rgraphviz")
```

```
>library(Rgraphviz)
```
install carefully (as it need us to change the path in the system settings)

```
>showSigOfNodes(sampleGOdata, score(resultFisher),
  firstSigNodes = 5, useInfo = "all")
```



In this plot,

- The subgraph induced by the top 5 GO terms identified by the classic algorithm for scoring GO terms for enrichment.
- Boxes indicate the 5 most significant terms. Box color represents the relative significance, ranging from dark red (most significant) to light yellow (least significant).
- Black arrows indicate is-a relationships and red arrows part-of relationships.

Check which genes in which top terms

For the selected GO terms (e.g., top 10), count the number of annotated genes and obtain their annotation.

```
sel.terms=allRes$GO.ID[1:10]
sel.terms
num.ann.genes <- countGenesInTerm(sampleGOdata,
  sel.terms)
num.ann.genes

ann.genes <- genesInTerm(sampleGOdata, sel.terms)
ann.genes
```