Homework Assignment # 3
Due Date: April 25, 2015

Instructor: Daniel P. Gaile, PhD
STA 525: Statistics For Bioinformatics

April 9, 2016

The Biomarker Challenge

1. Provide a comprehensive write-up of the in-class portion of the Biomaerker Challenge

2. For your Biomarker Challenge Array Data: cluster (using hclust) and conduct a Fisher's Exact Test for association between cluster and sample population.

   (a) Try two different distance metrics

   (b) Try a two different clustering methods

   (c) Try three different values for K, the number of clusters.

   (d) Report and compare your results. Additionally, relate your results to the collection of univariate t-tests that you performed when we conducted the Biomarker Challenge in class. If your group had a smaller budget and you were unable to find any statistically significant t-test results, did any of the cluster association studies provide a significant p-value?

3. Cluster the features of your array data.

   (a) Load the file GeneMap.RData which provides a variable GeneMap that maps each feature (e.g., gene) to a known set.

   (b) Perform hierarchical clustering of features. Do any of the clusters appear to be "enriched" for certain gene sets?

   (c) Examine the features that you identified for validation. Does that collection appear to be "enriched" for any of the GeneMap sets?

4. Using the GeneMap.RData, perform a GSA analysis of the type we discussed in lecture, i.e., select a p-value cut-off and perform Fishers Exact Tests.

5. Using the GeneMap.RData, perform a GSA analysis using *piano*.

6. Using the GeneMap.RData, perform a GSEA analysis of the type that we discussed in lecture.

GSEA problem

Consider the Berry, 2010 example dataset that we have looked at in various lectures, videos and in HW#1. Let's consider the training dataset, which has GEO identifier gse19439 .

Preparation

Start by doing the following:

1. Install the biomaRt Bioconductor package and read the vignette.
2. Install the piano Bioconductor package and read the vignette.

An example analysis

We would like to do an Gene Set Analysis. There are many variants out there, we will use the piano package to perform one type of analysis.

1. Load the gse19439 data and perform the Kruskal Wallis scan mentioned in HW#1:

```
LoadScanFlag=F  # flip this flag to T to load the data and perform the KW scan

 if(LoadScanFlag){

   require(GEOquery)

   gse19439=getGEO("GSE19439",GSEMatrix=T)
   gse19439=gse19439[[1]]

   tmp=as.character(pData(phenoData(gse19439))[,1])
   J=length(tmp) # J=number of samples
   TBgroup=rep("",J)
   for(j in 1:J) TBgroup[j]=substring(tmp[j],1,3)
   # make a factor for TBgroup
   FTB=factor(TBgroup,levels=c("CON","LTB","PTB"))

   # get our expression set
   X=exprs(gse19439)

   #
   # Let's do a simple Kruskal-Wallis Scan across all features
   #

   # do a quick kruskal-wallis scan
   myKrusk=function(i){
     cat(i,"...",fill=F)
     kruskal.test(x=X[i,],g=FTB)$p.value
   }

   myPvals=mapply(myKrusk,1:(dim(X)[1])) ;save(file="myPvals.RData",myPvals)
   save(gse19439,myPvals,FTB,TBgroup,X,file="HW3loadscan.RData")
 }

if(!LoadScanFlag) load("HW3loadscan.RData")
```

2. Now, we have to look at the ”features” of our ExpressionSet. What EntrezIDs do they map to?

```
require(affy)

## Loading required package:  affy
## Loading required package:  BiocGenerics
## Loading required package:  parallel
##
## Attaching package:  'BiocGenerics'
## The following objects are masked from 'package:parallel':
##
##    clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##    clusterExport, clusterMap, parApply, parCapply, parLapply,
##    parLapplyLB, parRapply, parSapply, parSapplyLB
## The following objects are masked from 'package:stats':
##
##    IQR, mad, xtabs
## The following objects are masked from 'package:base':
##
##    anyDuplicated, append, as.data.frame, as.vector, cbind,
##    colnames, do.call, duplicated, eval, evalq, Filter, Find, get,
##    grep, grepl, intersect, is.unsorted, lapply, lengths, Map,
##    mapply, match, mget, order, paste, pmax, pmax.int, pmin,
##    pmin.int, Position, rank, rbind, Reduce, rownames, sapply,
##    setdiff, sort, table, tapply, union, unique, unlist, unsplit
## Loading required package:  Biobase
## Welcome to Bioconductor
##
##    Vignettes contain introductory material; view with
##    'browseVignettes()'.  To cite Bioconductor, see
##    'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
myfeat=featureData(gse19439)
# what annotation information do we have?
varLabels(myfeat)
```

```
##  [1] "ID"                     "nuID"
##  [3] "Species"                "Source"
##  [5] "Search_Key"             "Transcript"
##  [7] "ILMN_Gene"              "Source_Reference_ID"
##  [9] "RefSeq_ID"              "Unigene_ID"
## [11] "Entrez_Gene_ID"         "GI"
## [13] "Accession"              "Symbol"
## [15] "Protein_Product"        "Array_Address_Id"
## [17] "Probe_Type"             "Probe_Start"
## [19] "SEQUENCE"               "Chromosome"
## [21] "Probe_Chr_Orientation" "Probe_Coordinates"
## [23] "Cytoband"               "Definition"
## [25] "Ontology_Component"     "Ontology_Process"
## [27] "Ontology_Function"      "Synonyms"
## [29] "Obsolete_Probe_Id"      "GB_ACC"
```

```
# o.k., but what do those varLabels means?
myvarmdat=varMetadata(myfeat)

# Now we know what info we have, how can we extract it and have a look?
mypdat=pData(myfeat)

Entrezs=mypdat[,11]
```

3. Note that the EntrezIDs are not unique. Which is to say, that multiple features map to the same EntrezIDs. Our simple (but imperfect) solution: use the minimum p-value across the set of features that map to the same

EntrezID.

```
 UnqFlag=F # change this flag to T to run this code the first time..

 if(UnqFlag){
 # This is the EntrezIDs:

 unqEntrez=unique(Entrezs)
 mapEntrz=rep(NA,length(unqEntrez))
 for(i in 1:length(unqEntrez)){
   edx=which(Entrezs==unqEntrez[i])
   mapEntrz[i]=edx[order(myPvals[edx])[1]]
 }

 mapEntrz=mapEntrz[unqEntrez!=""]
 names(mapEntrz)=unqEntrez[unqEntrez!=""]
 myEntrezs=names(mapEntrz)
 myX=X[mapEntrz,]
 myP=myPvals[mapEntrz]

 save(mapEntrz,myEntrezs,myX,myP,file="UnqEntrezs.RData")
}

 if(!UnqFlag) load("UnqEntrezs.RData")
```

4. We want to run the runGSA command in piano. In order to do that, we will need to map each EntrezID to it's GO ID(s). Note that EntrezIDs may map to multiple GO IDs. We will use biomaRt to accomplish this..

```
 require("biomaRt")

## Loading required package:  biomaRt

 require("piano")

## Loading required package:  piano
## Warning:  replacing previous import 'BiocGenerics::union' by 'igraph::union' when loading
'piano'
## Warning:  replacing previous import 'BiocGenerics::normalize' by 'igraph::normalize' when
loading 'piano'

 # first, we need an ensembl.. Since our data is on human subjects..
 ensembl = useMart("ensembl",dataset="hsapiens_gene_ensembl")

 GOIDSflag=F # trip this flag to  run getBM - this one takes some time, however.

 if(GOIDSflag){
    goids = getBM(attributes=c("entrezgene","go_id"), filters="entrezgene",
                  values=myEntrezs, mart=ensembl)
    myGsc <- loadGSC(goids)
    save(goids,myGsc,file="goids.RData")
  }

 if(!GOIDSflag) load("goids.RData")

 head(goids)

##   entrezgene      go_id
## 1      91624 GO:0005515
## 2      91624 GO:0051493
## 3      91624 GO:0030334
## 4      91624 GO:0005924
## 5      91624 GO:0030018
## 6      91624 GO:0005856
```

5. Now, let's use the runGSA command and inspect the output.

```
runGSAflag=F # trip to True to run the analysis

if(runGSAflag){
  names(myP)=myEntrezs
  myP[is.na(myP)]=1.0 # a lazy work around to account for a missing P-value

  gsaRes <- runGSA(myP, gsc=myGsc)

  save(gsaRes,file="gsaRes.RData")

  GSAsummaryTable(gsaRes, save=TRUE, file="gsaResTab.xls")

  nw <- networkPlot(gsaRes,class="non")
}
```

Your Assignment

1. Run the example analysis provided above. Note that it may take some time to run. How many gene sets are tested? Do you feel that the results of the analysis are easy to navigate?
2. Consider running the analysis on a reduced number of gene sets. This can be accomplished by filtering the results of our biomaRt query. Rerun the analysis with a filter of your choosing and comment on the results. We will discuss setting up filters in lecture on Tuesday, April 12.