1 Introduction

In this lab we will use the graph, Rgraphviz, and RBGL Bioconductor packages to assess the relationship between sets of genes that cluster together based on expression value from microarray experiments (the transcriptome) and sets of genes whose proteins are known to physically interact (the interactome). The lab follows the discussion in section 22.2 of Gentleman et al. (2005), which in turn is based on Balasubramanian et al. (2004).

The approach we will take is to create two graphs, one where edges represent the fact that two genes are in the same cluster and the other where an edge represents the fact that the two genes physically interact. If there is an association between clustering (which is based on gene expression) and physically interacting then we anticipate that when there is an edge between two proteins in one graph there will also be an edge between them in the other graph.

We test this hypothesis by creating the two graphs and counting how many edges they have in common. Then, we generate a reference distribution by permuting the labels on one of the two graphs (either one is fine) and for each permutation count edges in common. Finally, we compare the observed number of edges with the permutation distribution and conclude that there is a relationship if the observed number is large, as determined by the reference distribution.

2 Required Packages

Exercise 1
Use the library function to load the following packages: Biobase, graph, Rgraphviz, RBGL, RColorBrewer, RbcBook1, and yeastExpData.
3 The Data

The curated data in \texttt{yeastExpData} contains both gene expression data from a yeast cell-cycle experiment, including cluster membership (\texttt{ccyclered}), and protein-protein interaction (PPI) data extracted from published papers (\texttt{litG}).

\begin{verbatim}
> data(ccyclered)
> data(litG)
\end{verbatim}

Exercise 2
What type of object is \texttt{litG}? How do you find out more about this class? Use the \texttt{nodes} method to extract the first five nodes of \texttt{litG}.

Exercise 3
Explore the \texttt{ccyclered} data to determine what type of R object it is and what kind of data it contains.

4 Graph Basics

The \texttt{RBGL} package has a number of different graph algorithms implemented. In these next few exercises we will see how to use a few of them. We will use the \texttt{litG} graph for our examples.

A graph can consist of one or more connected components. You can find them using the \texttt{connectedComp} function.

\begin{verbatim}
> cc1 = connectedComp(litG)
> length(cc1)

[1] 2642

> cc1lens = sapply(cc1, length)
> table(cc1lens)

cc1lens
     1  2  3  4  5  6  7  8 12 13 36 88
2587 29 10  7  1  1  2  1  1  1  1  1
\end{verbatim}

Exercise 4
How many connected components are there? What is the size of the largest connected component? How many singletons are there? What are the elements (or values) stored in \texttt{cc1}? Create a subgraph of \texttt{litG} which has only the connected components of size 4 or more.
Let's plot the component of size 12 using the Rgraphviz package. We first compute the subgraph and then lay it out. There are many options for node color, line color and type, node shape etc. If you are interested in more complex pictures you should read the vignette from the Rgraphviz package.

```r
> sg12 <- subGraph(cc1[cc1lens == 12][[1]], litG)
> l12 <- agopen(sg12, layoutType = "dot", nodeAttrs = makeNodeAttrs(sg12, + fillcolor = "steelblue2"), name = "")
> plot(l12)
```

![Graph of size 12 component](image)

Figure 1: The connected component of size 12.

**Exercise 5**

*Layout the graph using the two other layout types from Rgraphviz, namely neato and twopi.*

Next let’s extract the largest component and use it to compute some other quantities. Here we compute the shortest path between a rather arbitrarily selected pair of nodes.
Exercise 6
What sort of object is \( \text{sps} \)? What does the manual page say about it? Can you plot the graph and identify that this indeed is the shortest path? (You could color these nodes differently from the rest).

If you want to find the diameter of the graph it is defined as the longest shortest path between any two nodes. To compute this we use the function \texttt{johnson.all.pairs.sp}.

\begin{verbatim}
> allp = johnson.all.pairs.sp(sg88)
\end{verbatim}

Exercise 7
What type of object is \( \text{allp} \)? What data does it contain? What is the diameter of \( \text{allp} \)?

5 The Analysis

We now return to the analysis we discussed at the beginning of this lab. As suggested above we want to create two graphs, one that reflects protein interactions and one that reflects clustered gene expression. The first one of these graphs is \texttt{litG} and we will need to use the \texttt{ccyclered} to create the second one.

\[ \text{Cho et al.} \ (1998) \] discuss the k means clustering of 2885 Saccharomyces genes into 30 clusters with measurements taken over two synchronized cell cycles. These data are stored as \texttt{ccyclered} and we next explain how to extract what is needed.

5.1 Cluster graph

The first step is to create a cluster graph from the \texttt{ccyclered} data in which edges are between all genes that are in the same cluster. The clusters are given in the first column (named \texttt{Cluster}) of the data.frame. There is a specialized graph class, \texttt{clusterGraph} that can be used to represent clusters.

We need to compute the set of genes in each cluster and you will do that by building a list, where each entry contains the names of the genes in that cluster.

Exercise 8
Use the \texttt{split} function and the \texttt{Y.name} and \texttt{Cluster} columns of the \texttt{ccyclered} dataframe to create a list that maps gene name to cluster name. Store the list in a variable named \texttt{clusts}.

Next we use the \texttt{clusts} list from the previous exercise to create a \texttt{clusterGraph} instance using \texttt{new}:
Exercise 9
How many connected components does the cluster graph \texttt{cg1} have? Hint: \texttt{apropos("connect")}.

5.2 PPI graph

We next turn our attention to a brief exploration of the literature-based PPI data stored as a \texttt{graphNEL} object.

Exercise 10
Store the connected components of \texttt{litG} in a variable called \texttt{ccLit}.

Exercise 11
\begin{itemize}
  \item[a)] Use \texttt{listLen} to compute the size of each connected component. Store the result in \texttt{cclens}.
  \item[b)] Use \texttt{table} to summarize the sizes of the connected components stored in \texttt{cclens}.
  \item[c)] How many singleton components are there?
\end{itemize}

Creating an index vector that orders the connected components by size will allow us to easily access the smallest and largest components. In the example below, we list the genes in the eighth largest connected component.

> \texttt{ord <- order(cclens, decreasing = TRUE)}
> \texttt{ccLit[ord[8]]}

\begin{verbatim}
[1] "YMR080C" "YLL026W" "YJR132W" "YDR172W" "YNL112W"
[6] "YBR143C"
\end{verbatim}

Exercise 12
Use the \texttt{subGraph} method to create two new graphs \texttt{sG1} and \texttt{sG2}, the first and second largest connected components of of the \texttt{litG} graph.

Now we plot \texttt{sG1} and \texttt{sG2} using \texttt{Rgraphviz}:

> \texttt{sG1 <- agopen(sG1, layoutType = "neato", nodeAttrs = makeNodeAttrs(sG1),}
> \texttt{+ name = "")}
> \texttt{sG2 <- agopen(sG2, layoutType = "neato", nodeAttrs = makeNodeAttrs(sG2,}
> \texttt{+ fillcolor = "#a6cee3"), name = "")}

> \texttt{plot(sG1)}

> \texttt{plot(sG2)}
5.3 Testing associations

It is now easy to determine how many pairs of genes have both a protein-protein interaction and are found in the same expression cluster. To compute this, find the intersection of the cluster-graph and the literature graph using \texttt{intersection}:

\begin{verbatim}
> commonG <- intersection(cg1, litG)
\end{verbatim}

Exercise 13

How many edges are common to the two graphs (\texttt{cg1} and \texttt{litG})?

Now we will try to determine whether the number of common edges is statistically interesting or not. We will do this by generating a null distribution via permutation of node labels on the observed graph.

Here is a function that can be used to generate values from the desired null distribution. Unfortunately, running this function with the current implementation is very slow.

\begin{verbatim}
> nodePerm <- function(g1, g2, B = 1000) {
+   n1 <- nodes(g1)
+   sapply(1:B, function(i) {
+     nodes(g1) <- sample(n1)
+     numEdges(intersection(g1, g2))
+   })
+ }
\end{verbatim}

Exercise 14

Describe what the \texttt{nodePerm} function is doing to make sure you understand how it works.

Since the \texttt{nodePerm} function is slow, we’ve computed 500 iterations ahead of time. Load the precomputed result as follows:
> data(nPdist)
> summary(nPdist)

**Exercise 15**
*Plot the nPdist data and decide if the number of edges in common between litG and cg1 is statistically interesting.*

6 **Some harder problems**

In this section we present some problems that are more open ended. They are not formally part of this Lab, but are here for those who finish early, or who are particularly interested in these sorts of applications.

To answer the last two questions you will need to obtain a newer Bioconductor package called ScISI.

- Which of the expression clusters have intersections and with which of the literature clusters?
- Are there expression clusters that have a number of literature cluster edges going between them (and hence suggesting that the expression clustering was too fine or that the genes involved in the literature cluster are not cell-cycle regulated).
- Are there known cell-cycle regulated protein complexes, and do the genes involved tend to cluster together in both graphs?
- Is the expression behavior of genes that are involved in multiple protein complexes different from that of genes that are involved in only one complex?

The version number of R and packages loaded for generating this document are:

Version 2.3.1 Patched (2006-06-08 r38315)
powerpc-apple-darwin8.6.0

attached base packages:
[1] "tools" "methods" "stats" "graphics"
[5] "grDevices" "utils" "datasets" "base"

other attached packages:
yeastExpData RbcBook1 RBGL Rgraphviz
   "0.6.0" "1.0.2" "1.8.1" "1.10.0"
graph Ruuid Biobase
   "1.10.6" "1.10.0" "1.10.0"
References

