Using GoNetwork

Sara Linker

March 29, 2017

Introduction

The purpose of GoNetwork is to create a functionally enriched gene network using GO terminology. Networks created using this program are easily plotted with molecular visualization software platforms such as Cytoscape. The sample data used in this markdown is a list of homo sapien genes downloaded from the AmiGO 2 website associated with the following go terms: intracellular transport (GO:0046907), intracellular vesicle (GO:0097708), ATP binding (GO:0005524), postsynaptic density (GO:0014069), and negative regulation of transcription from polymerase II promoter (GO:0000122). *Is it overkill to type out all of the GO terms and numbers?*

Run

Use **getGo()** to create a matrix of GO terms based on your gene list. GoNetwork improves upon previous functional gene enrichment tools by applying weights to the binary GO term matrix based on the number of parents in the GO hierarchy.

```
> #load data:
> data("differentiation")
> #M <- getGo(genes,species = "human")
```

Once the GO term matrix is created, use **GoTheDist()** to calculate the distances between genes. **GoTheDist()** allows users to restrict the scope of GO terms based on minimum number of genes that fall under a GO term and minimum number of parents. Users may also choose whether the distances are calculated using the cosine function, manhattan distance or euclidean method: *"cosine", "manhattan" or "euclidean"*. In this case, the cosine function is used.

```
> #D <- GoTheDist(M,method="cosine",Min=6,minparents=15)
> #quantile(D)
```

After distances have been calculated, convert the matrix to Cytoscape format using **cyto()**. Stringency can be adjusted by changing the lower and upper cutoff values. *low.cutoff* is the minimum distance between two genes for the connection to be sent to cytoscape view, whereas *high.cutoff* is the maximum distance. *Note: this may be confusing to users because low.cutoff is used for cosine and high.cutoff must be used for euclidean and manhattan.*

```
> #tab<-cyto(D,method="cosine",low.cutoff=.5,high.cutoff=Inf)
> #head(tab)
>
```

Users can specify the cutoff values manually, or use **cutoff()** to calculate the cutoff values automatically. This function locates the modes in the distribution of distance values and calculates the cutoff value that will provide optimal clustering. *Sentence about stringency after figuring out with Sara*

```
> #cut<-cutoff(M,method = "cosine",stringency=2.5,Min = 6,minparents = 15)
> #cut
```

Then write the resulting matrix to a .txt file.

```
> #forcyto<-cyto(D,method="cosine",low.cutoff=cut,high.cutoff=Inf)
> #write.table(forcyto,row.names = F,file = "forcyto.txt",col.names = T,sep=",")
```

Other Functions Included

Use **symbolConvert()** to convert alternative gene notations to the desired format. In this case, a list of entrezgene IDs are converted to their respective HGNC gene symbols.

```
> #head(gene.numbers)
> #gene.symbols<-symbolConvert(gene.numbers,"entrezgene","hgnc_symbol")
> #head(gene.symbols)
```

Use **exactTerms()** to identify GO terms containing a specified character string

```
> #e<-exactTerms(colnames(M), term = "vesicle")
> #head(e)
> #Note: ask Sara if this is the correct use of the function
```

Use **findGenes()** to return a list of genes with GO terms containing a specified character string. We can use one of the terms found using the exactTerms() function above:

```
>
> #head(findGenes(M, term = "acrosomal.vesicle"))
>
```

If you would like to identify the GO terms that a group of genes have in common, for instance, a list of genes taken from a cluster seen after plotting in cytoscape, use **findTerms()**:

```
>
> #findTerms(M, geneA = "APOE" ,geneB = "FNBP1L",geneC="TBC1D10A")
```