GONetwork

Sara B. Linker, Sarah Voter, Fred H Gage

May 3, 2017

Introduction

GONetwork is an R toolkit that detects statistically significant functional annotation differences between gene lists and has direct application to RNA-seq, ChIP-seq, and GWAS experiments. Through a short workflow GONetwork: (1) calculates a gene network based on functional overlap, (2) clusters genes within that defined functional space, and (3) statistically tests differential enrichment within each cluster to determine functional annotation that is unique to a given condition.

Background

Currently, standard functional analyses are restricted by the requirement of exact overlap of a gene list with a dictionary of predefined terms. This strategy is limited by the lack of complete annotation. Furthermore, it is limited by reduced power in small genes sets which often results in broad, biologically ambiguous terms. Contrary to these current methods that require functional equivalence, GONetwork compares genes via functional similarity. Functional similarity initially broadens the space of terms identified and then narrows in on the highly specific terms within a gene set. This alternative strategy returns a highly robust and informative set of enriched functional terms. Within the terms that are enriched within a complete dataset, GONetwork then tests differential enrichment of each annotation cluster based on the condition from which the gene was identified. GONetwork returns to the user a highly specific set of functional terms that are relevant to the study as well as those that are specific to a given condition. This enables researchers to narrow in on the informative functional groupings present within their dataset.

Run

GONetwork is designed to take a list of genes and calculate a similarity matrix based on functional annotation. The matrix (M) is created with with knowledge of the hierarchical representation of GO terms. Since there are many GO terms available and few GO terms relevant for each gene, M is a highly sparse matrix. A cosine function is applied to successfully work with this sparse data, however the user can input other common distance measures. We've provided a test dataset to work with generated from real data examining neural progenitor cells throughout differentiation into neurons. Gene expression was compared between neural progenitor cells and neurons. *differentiation* contains the significant genes from this analysis.

```
> data(differentiation)
> head(differentiation)
```

```
[1] "GTF3C2-AS1" "PRMT5-AS1"  "FANCD2OS"   "MED4-AS1"   "LRRC37A16P"
[6] "PKD2L2"
```

Generate GOterm Matrix

Use **getGo()** to create the matrix of GO terms based on the gene list.

```
> M <- getGo(differentiation,species = "human")
```

Once the GO term matrix is created, use **GoTheDist()** to calculate the distances between genes. Here, we're using the cosine distance to calculate similarity. It is useful to restrict the analysis to highly informative GO terms. Here we've required that all of the GO terms have at least 15 ancestors.

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

Plot the full network

cyto() will convert the matrix to Cytoscape format. The overall connectivity of the network can be adjusted by changing the cutoff values. Here, we exclude genes with a similarity score of less that 0.2.

```
> tab<-cyto(D,cutoff = 0.3)
```

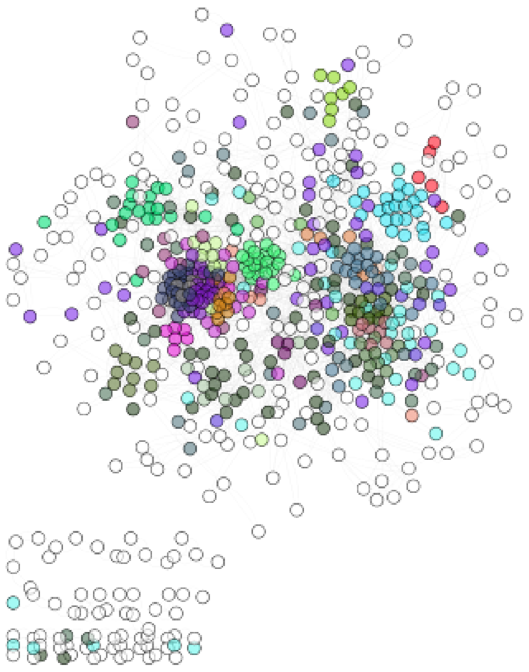

Figure 1: Cytoscape plot of all genes labelled by gene cluster

Identify gene clusters

AssignCluster() will determine gene clusters given a user defined k. Here we've turned on the additional option to remove stragglers. This recalculates gene clusters after removing genes that are not clearly defined with GO annotation.

```
> tab <-AssignCluster(tab = tab, D = D, K = 30, rmv_stragglers = TRUE)
```

Differential functional enrichment

You can then calculate differential functional enrichment between conditions using **testClusters()**. Test clusters requires an additional variable **group** which identifies which condition each gene is related to. **group** is a data.frame where the first column contains all of the gene names from **differentiation** and the second column contains the group designation to be tested ("up-regulated" or "down-regulated").

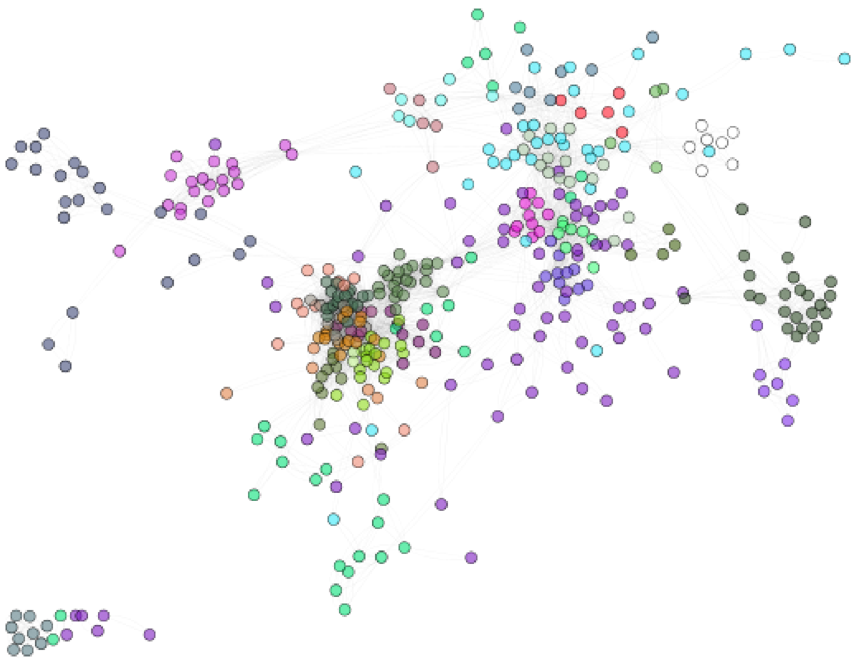

Figure 2: Cytoscape plot after refinement

```
> data(group)
> k.test <- testClusters(tab,group)
> k.test[k.test$prop_test_p < 0.05,]

  group obs.high obs.low  exp.high   exp.low odds_ratio  prop_test_p
1    21        1      21 12.651354  9.348646  23.130878 1.513407e-06
2     4       53      11 36.803938 27.196062  15.752921 7.217642e-05
3    22        1       8  5.175554  3.824446   6.142748 1.319521e-02
4    14        0       6  3.450369  2.549631   5.936919 1.482692e-02
5    12       10       0  5.750615  4.249385   5.752805 1.646235e-02
6    18       10       0  5.750615  4.249385   5.752805 1.646235e-02
7    26        8       0  4.600492  3.399508   4.300493 3.810133e-02
8    11       24       7 17.826907 13.173093   4.248520 3.928454e-02
          padj
1 4.540221e-05
2 2.093116e-03
3 3.694659e-01
4 4.003268e-01
5 4.280211e-01
6 4.280211e-01
7 9.144320e-01
8 9.144320e-01
```

Examine annotation associated with each group

findTerms returns a list of GO terms shared within each gene group. Here, we've required that at least 40 percent of the genes share the annotation term.

```
> term_genes <- unique(as.character(tab[as.character(tab$group) == 21, "origin"]))
> head(term_genes)
```

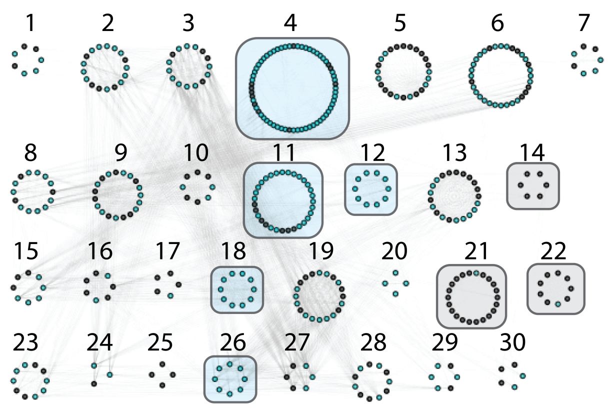

Figure 3: Genes grouped by clustering and colored by differential expression estimate. All clusters represent genes that are enriched for specific functional annotation. Note the interconnectedness wihtin a group as well as between groups. The between groups represents gene clusters with similar annotation. Highlighted clusters contain more genes from a given condition. For example, cluster 22 primarily contains genes that are more highly expressed in neurons. Below we will examine what genes and terms are present within these clusters.

```
[1] "GRIA2"  "GRIN2A" "GRIK4"  "GRIK1"  "GRIA3"  "SSPN"
```

```
> findTerms(M, term_genes,proportion.shared = 0.3)
```

```
 [1] "protein binding"
 [2] "plasma membrane"
 [3] "integral component of plasma membrane"
 [4] "transport"
 [5] "neuropeptide signaling pathway"
 [6] "synaptic transmission"
 [7] "cell junction"
 [8] "ion transmembrane transport"
 [9] "postsynaptic membrane"
[10] "excitatory postsynaptic potential"
[11] "chloride transmembrane transport"
```

```
> findTerms(M, term_genes,proportion.shared = 0.6)
```

```
[1] "plasma membrane"       "cell junction"         "postsynaptic membrane"
```

Examine genes associated with each annotation

findGenes returns genes in your dataset exactly associated with a given functional term.

```
> findGenes(M,tab, group = 21, term = "postsynaptic membrane")
```

```
 [1] "GRIN2D" "GRIA3"  "GRIK4"  "GLRB"   "GABRG2" "GRIK1"  "SSPN"   "GLRA2"
 [9] "GRIA2"  "GABRB1" "GABRA3" "GABRG3" "GLRA3"  "GABRA2"
```

Alternatively, you can use **findGenes** to return genes in any GO term. Here we are looking for which genes, and associated clusters, contain genes involved in the cell cycle.

```
> fg <- findGenes(M,tab, term = "cell cycle")
> head(fg)
```

```
    gene group
1  SPDYA     4
2   GPS2     2
3 TRIM71    11
4   DBF4    12
5 KIF20B     6
6  PRR11     0
```

We can then trace back which are the top clusters that have this term. Since cell cycle is a common term with only 5 ancestors it appears across many genes including the group 0, which are unclustered genes, as well as groups 4, and 11.

```
> fg.table <- data.frame(table(fg$group))
> fg.table <- fg.table[order(fg.table$Freq,decreasing = TRUE),]
```

Let's take a look at what the top functional categories are in groups 4 and 11, which contain genes involved in the cell cycle

```
> #Group 4
> term_genes <- unique(as.character(tab[as.character(tab$group) == 4, "origin"]))
> head(term_genes)

[1] "CDC6"   "POLQ"   "TICRR"  "POLE2"  "POLE"   "CDC25A"

> findTerms(M, term_genes,proportion.shared = 0.3)

[1] "protein binding" "nucleus"         "nucleoplasm"     "cytoplasm"
[5] "cytosol"

> findTerms(M, term_genes,proportion.shared = 0.6)

[1] "protein binding" "nucleoplasm"

> #Group 11
> term_genes <- unique(as.character(tab[as.character(tab$group) == 11, "origin"]))
> head(term_genes)

[1] "IGF2BP1" "IGF2BP3" "CDCA3"   "SPC24"   "KLHL1"   "CDKL2"

> findTerms(M, term_genes,proportion.shared = 0.3)

[1] "protein binding" "nucleus"         "nucleoplasm"     "cytoplasm"
[5] "cytosol"         "cell division"

> findTerms(M, term_genes,proportion.shared = 0.6)

[1] "protein binding" "cytosol"
```