GONetwork

Sara B. Linker, Sarah Voter, Fred H Gage

May 2, 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 11 ancestors.

```
> D <- GoTheDist(M,method="cosine",minparents=11)
```

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)
```

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 = 20, 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").

```
> 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     4      100      28 73.607875 54.392125  21.433103 3.663906e-06
2     5        9      25 19.552092 14.447908  12.161681 4.878139e-04
3    10       15       0  8.625923  6.374077   9.413425 2.154025e-03
4    12        3      11  8.050861  5.949139   6.053671 1.387742e-02
5    13       18       3 12.076292  8.923708   5.732353 1.665515e-02
          padj
1 7.327813e-05
2 9.268463e-03
3 3.877245e-02
4 2.359161e-01
5 2.664824e-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[tab$group == 10, "destination"]))
> findTerms(M, term_genes,proportion.shared = 0.5)
```

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