GO Enrichment

Run the `render()` function below and everything will be run with report at end.

```
library(rmarkdown)
render("skeleton_GO.Rmd", "pdf_document", output_file = paste(sample1,"_",sample2,"_","GO.pdf",sep=""))
```

Read in YAML guide

```
library(yaml)
yamls <- yaml.load_file("de.yml")
sample1 <- yamls$sample1
sample2 <- yamls$sample2

sample1
```

```
## [1] "tf2cmbr"
```

```
sample2
```

```
## [1] "wtcmbr"
```

```
library(goseq)
library(GO.db)
```

Setting up the DE table for GO analysis

File Input

Input the output from DE analysis. This is made for a list that includes only the significant genes.

```
sigOnly <- read.table(paste(sample1,"_",sample2,"_DE_sig.txt", sep=""), header = TRUE, fill = TRUE)
head(sigOnly)
```

```
##                  ITAG  logFC logCPM    PValue       FDR
## 1 Solyc00g010770.1.1 -3.263  4.004 1.603e-06 2.287e-04
## 2 Solyc00g019970.2.1 -2.662  6.541 2.650e-06 3.523e-04
## 3 Solyc00g023580.1.1 -2.761  5.336 2.741e-06 3.585e-04
## 4 Solyc00g072100.2.1 -3.102  4.185 8.098e-07 1.323e-04
## 5 Solyc00g085070.2.1 -4.442  7.872 9.889e-14 2.216e-10
## 6 Solyc00g121730.1.1 -2.339  4.693 8.527e-05 6.080e-03
##                                                                                                     S
## 1
## 2
## 3
## 4
## 5 Squalene monooxygenase (AHRD V1 ***- Q506K3_DATIN); contains Interpro domain(s)  IPR013698  Squalen
## 6
##          AGI symbol
## 1      <NA>   <NA>
## 2      <NA>   <NA>
## 3      <NA>   <NA>
## 4      <NA>   <NA>
## 5 AT4G37760   SQE3
## 6      <NA>   <NA>
##
## 1
## 2
## 3
## 4
## 5 squalene monooxygenase, putative / squalene epoxidase, putative; similar to XF1, oxidoreductase [A
## 6
##   X..identity alignment.length e.value bit.score percent.query.align
## 1          NA               NA      NA        NA                  NA
## 2          NA               NA      NA        NA                  NA
## 3          NA               NA      NA        NA                  NA
## 4          NA               NA      NA        NA                  NA
## 5       81.45              442       0       753               95.04
## 6          NA               NA      NA        NA                  NA
```

```
dim(sigOnly)
```

```
## [1] 401  14
```

```
colnames(sigOnly)
```

```
##  [1] "ITAG"                "logFC"               "logCPM"
##  [4] "PValue"              "FDR"                 "SGN_annotation"
##  [7] "AGI"                 "symbol"              "gene_name"
## [10] "X..identity"         "alignment.length"    "e.value"
## [13] "bit.score"           "percent.query.align"
```

```
colnames(sigOnly)[1] <- "itag"
```

Subset

First I need to subset the list to up or down regulated, then add a new colum that specififys 1. This column is need to for merging.

```
upITAG <- subset(sigOnly, logFC > 0, select = c(itag))
upITAG$up <- 1

downITAG <- subset(sigOnly, logFC < 0, select = c(itag))
downITAG$down <- 1

allITAG <- subset(sigOnly, select = c(itag))
allITAG$all <- 1
```

Merge I - with normalized ITAG length gene list

read in guide.

```
geneLength <- read.csv("../normalized_genes_length.csv")
head(geneLength)
```

```
##                  itag length
## 1 Solyc00g005040.2.1    357
## 2 Solyc00g005050.2.1    588
## 3 Solyc00g005060.1.1    273
## 4 Solyc00g005070.1.1     81
## 5 Solyc00g005080.1.1    297
## 6 Solyc00g005150.1.1   1143
```

```
#isolate just the gene list
genes <- subset(geneLength, select = c(itag))
```

First merge each table to geneLength

```
upITAGmerge <- merge(genes, upITAG, by = "itag", all= TRUE)
downITAGmerge <- merge(genes, downITAG, by = "itag", all= TRUE)
allITAGmerge <- merge(genes, allITAG, by = "itag", all= TRUE)
```

Merge II - Merge them all together.

```
matrixGOupdown <- merge(upITAGmerge, downITAGmerge, by = "itag", all = TRUE)
matrixGOupdownall <- merge(matrixGOupdown, allITAG, by = "itag", all = TRUE)
matrixGO <- merge(matrixGOupdownall, geneLength, by = "itag", all = TRUE)
```

Clean Up

```
matrixGO[is.na(matrixGO)] <- 0
head(matrixGO)
```

```
##                  itag up down all length
## 1 Solyc00g005040.2.1  0    0   0    357
## 2 Solyc00g005050.2.1  0    0   0    588
## 3 Solyc00g005060.1.1  0    0   0    273
## 4 Solyc00g005070.1.1  0    0   0     81
## 5 Solyc00g005080.1.1  0    0   0    297
## 6 Solyc00g005150.1.1  0    0   0   1143
```

This is if you want to write out the table of the GO matrix. #write.table(matrixGO, “mydata.txt”, sep=“”͡, quote= FALSE)

GO enrichment

The is the input of the GOslim categories. There are only two columns 1. itag and 2. go

```
pat <- matrixGO
head(pat)
```

```
##                  itag up down all length
## 1 Solyc00g005040.2.1  0    0   0    357
## 2 Solyc00g005050.2.1  0    0   0    588
## 3 Solyc00g005060.1.1  0    0   0    273
## 4 Solyc00g005070.1.1  0    0   0     81
## 5 Solyc00g005080.1.1  0    0   0    297
## 6 Solyc00g005150.1.1  0    0   0   1143
```

```
cate <- read.table("../melted.GOTable.txt",header=TRUE)
head(cate)
```

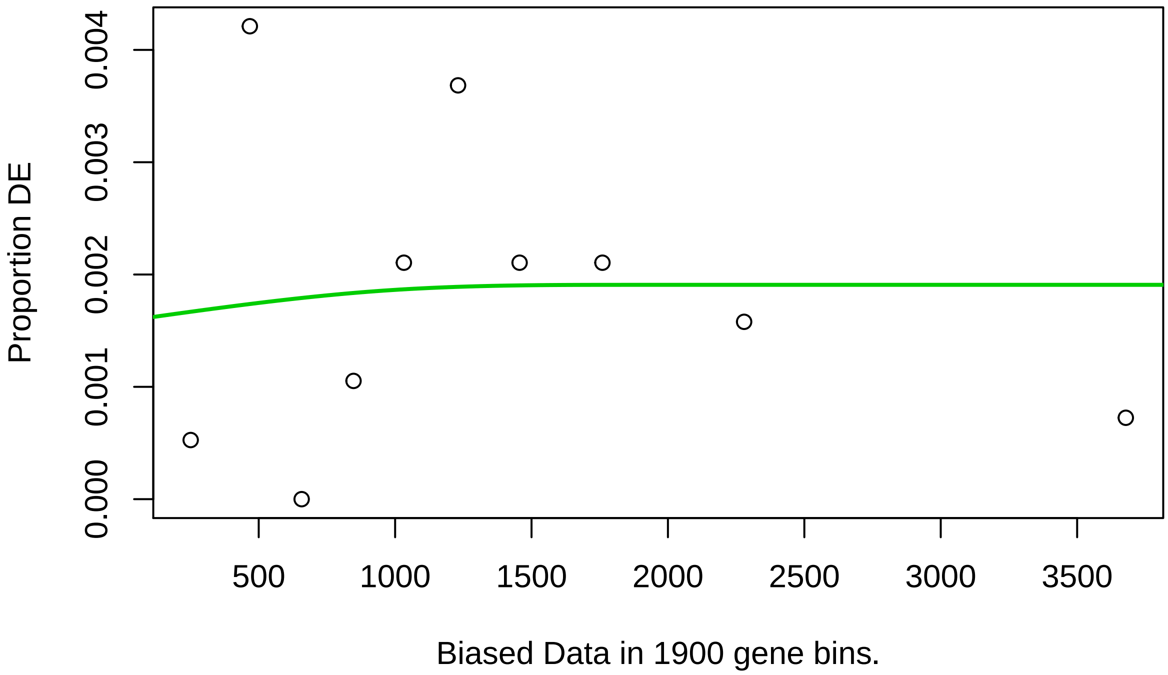
```
##                  itag         go
## 1 Solyc00g005000.2.1 GO:0006508
## 2 Solyc00g005040.2.1 GO:0005774
## 3 Solyc00g005050.2.1 GO:0005829
## 4 Solyc00g005080.1.1 GO:0005524
## 5 Solyc00g005130.1.1 GO:0006508
## 6 Solyc00g005150.1.1 GO:0003676
```

Subseting for GO analysis

Specify the column you are interested in `pat$all` refers to all the DE gene regardless if they are up or down regulated. If you want to specify down regulated, specify `pat$down`. I am going to put this into a loop, where each time the loop goes thought it will perform GO enrichment on all three types of lists of significant genes and them write them to a table.

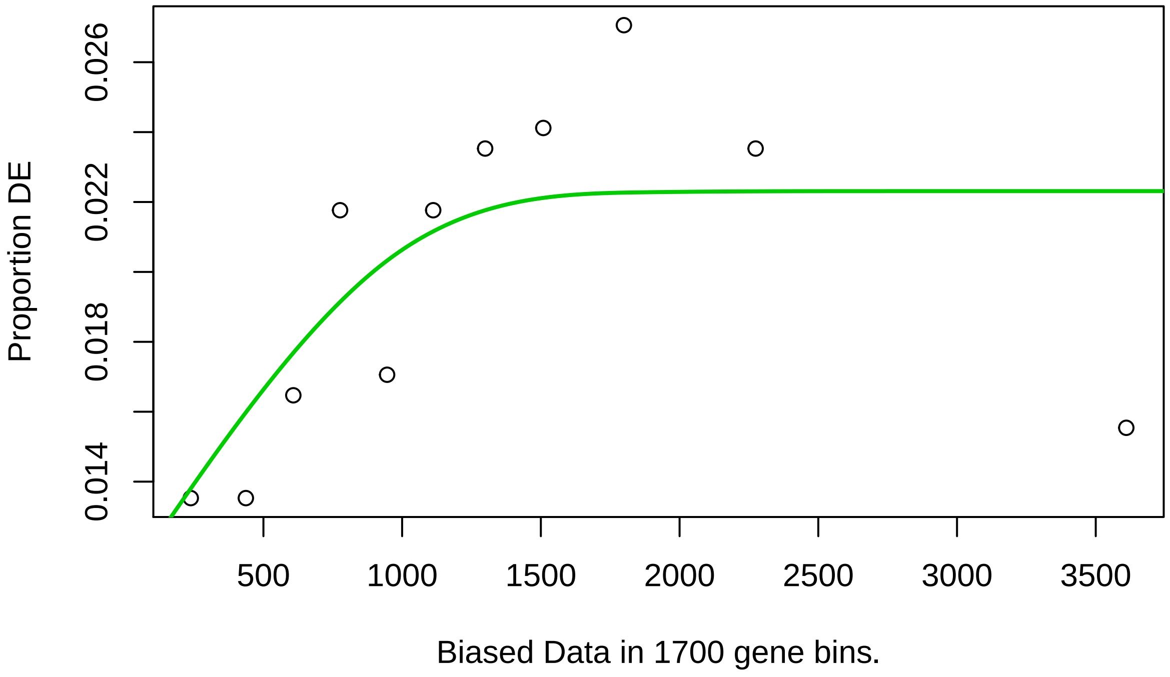
```
sigType <- c("up", "down", "all")

for(type in sigType) {

genes = as.integer(pat[,type])
names(genes) = pat$itag
table(genes)
length(genes)

pwf = nullp(genes,bias.data=pat$length)

GO.wall = goseq(pwf,gene2cat = cate)
head(GO.wall)

#This is going to correct for multiple testing.  You can specify the p-value cut-off of GO categories y

enriched.GO = GO.wall$category[p.adjust(GO.wall$over_represented_pvalue, method = "BH") < 0.05]

enriched.GO
my.GO <- as.character(enriched.GO)
my.GO.table <- Term(my.GO)
my.GO.table
t <- as.matrix(my.GO.table)

print(type) #this is for the knitr document
print(t) #this is for the knitr document

write.table(t, file=paste(sample1,"_",sample2,"DE1_sigonly_",type,"_GO.txt", sep=""))
}
```

```
## Using manually entered categories.
## For 2940 genes, we could not find any categories. These genes will be excluded.
## To force their use, please run with use_genes_without_cat=T (see documentation).
## This was the default behavior for version 1.15.1 and earlier.
## Calculating the p-values...
```



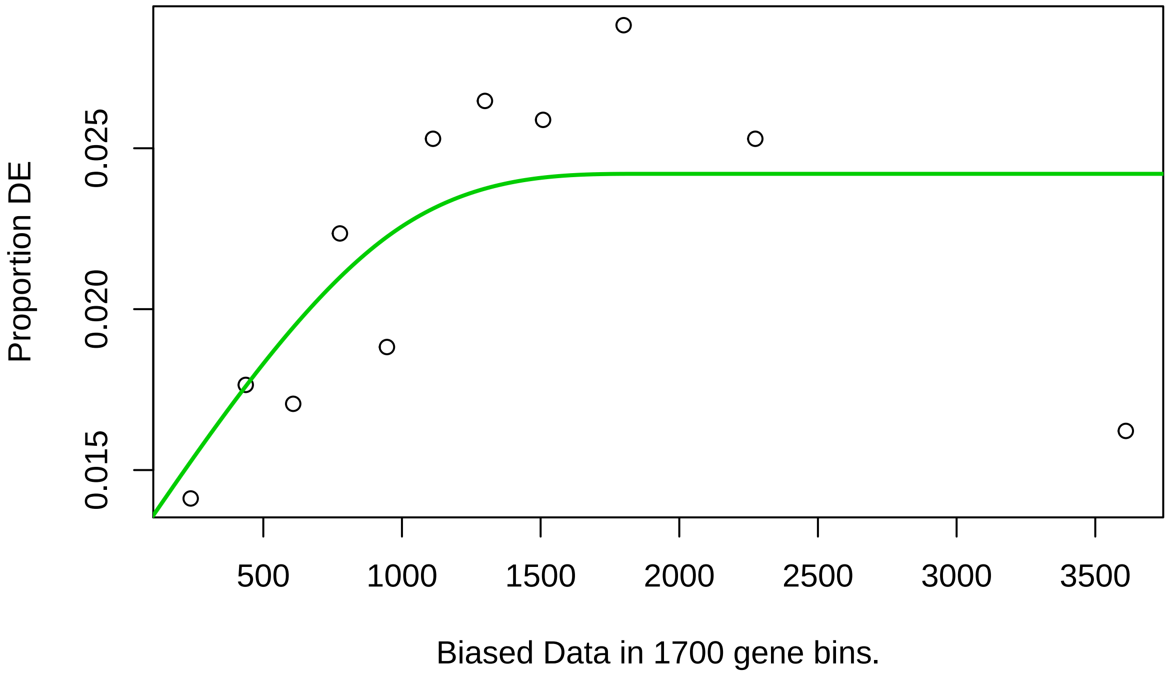
```
## [1] "up"
##      [,1]
```

```
## Using manually entered categories.
## For 2940 genes, we could not find any categories. These genes will be excluded.
## To force their use, please run with use_genes_without_cat=T (see documentation).
## This was the default behavior for version 1.15.1 and earlier.
## Calculating the p-values...
```



```
## [1] "down"
##            [,1]
## GO:0003964 "RNA-directed DNA polymerase activity"
## GO:0006278 "RNA-dependent DNA replication"
## GO:0015074 "DNA integration"
## GO:0043229 "intracellular organelle"
## GO:0006259 "DNA metabolic process"
```

```
## Using manually entered categories.
## For 2940 genes, we could not find any categories. These genes will be excluded.
## To force their use, please run with use_genes_without_cat=T (see documentation).
## This was the default behavior for version 1.15.1 and earlier.
## Calculating the p-values...
```



```
## [1] "all"
##            [,1]
## GO:0003964 "RNA-directed DNA polymerase activity"
## GO:0006278 "RNA-dependent DNA replication"
## GO:0015074 "DNA integration"
## GO:0043229 "intracellular organelle"
## GO:0006259 "DNA metabolic process"
```