Analysis 7 - Getting a closer look at the clusters from the clusters from the superSOM,

Purpose

To get start to understand the differences in GO categories between the superSOM and SOM clusters.

Part 1 - Functions and required Data

Required Libraries

```
library(VennDiagram)
```

```
## Loading required package: grid
```

```
library(ggplot2)
library(reshape)
library(kohonen)
```

```
## Loading required package: class
##
## Attaching package: 'class'
##
## The following object is masked from 'package:reshape':
##
##     condense
##
## Loading required package: MASS
```

```
library(goseq)
```

```
## Loading required package: BiasedUrn
## Loading required package: geneLenDataBase
```

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

```
## Loading required package: AnnotationDbi
## Loading required package: BiocGenerics
## Loading required package: parallel
##
## Attaching package: 'BiocGenerics'
##
## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB
##
```

```
## The following object is masked from 'package:stats':
##
##     xtabs
##
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, as.vector, cbind,
##     colnames, do.call, duplicated, eval, evalq, Filter, Find, get,
##     intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rep.int, rownames, sapply, setdiff, sort,
##     table, tapply, union, unique, unlist
##
## Loading required package: Biobase
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
##
## Loading required package: GenomeInfoDb
##
## Attaching package: 'AnnotationDbi'
##
## The following object is masked from 'package:MASS':
##
##     select
##
## Loading required package: DBI
```

Read in data used for GO enrichment analysis

```
geneLength <- read.csv("../data/normalized_genes_length.csv")
cate <- read.table("../data/melted.GOTable.txt",header=TRUE)
```

Cluster Specific analysis

Now I want to take a look at what are is going on exactly in these clusters. The clusters start with the bottom left, which is cluster number 1.

This is a function that makes a boxplot showing the transformed values of expression in the clusters.

clusterVis

```
#displays transformed data in a box plot
clusterVis <- function(clustNum){

  sub_cluster <- subset(plot.data, ssom.unit.classif==clustNum)
  sub_data <- sub_cluster[,c(1, 9:14)] # just the sample types
  m.data <- melt(sub_data)
  p <- ggplot(m.data, aes(x=variable, y=value, color = genotype))
```

```
  p + geom_point(alpha=0.5,position="jitter", size=1) +
    geom_boxplot(alpha=0.75, outlier.size=0) +
    theme_bw()
}
```

clusterGO

This is a function that prints out any GO categories associated with the cluster.

```
clusterGO <- function(clustNum){
  ##GO Enrichment on the catergories
   dev.off()
  plot.new()

  #we need to first get the data in the right format.
  #First get the list of ITAG

  #sub_cluster
  sub_cluster <- subset(plot.data, ssom.unit.classif==clustNum)

  itag.sc <- as.data.frame(sub_cluster$gene)
  colnames(itag.sc)[1] <- "itag"
  itag.sc$sc <- 1

  #Since each orthologue between tf2 and wt are represented twice in this set, we have to keep only the

  itag.sc <- unique(itag.sc) #Check. Should cut the list in half. # dim(itag.sc) before and after

  #Merge all by itag
  matrixGO <- merge(itag.sc, geneLength, by = "itag", all = TRUE)
  matrixGO[is.na(matrixGO)] <- 0
  pat <- matrixGO

  #Now that we have the data in the right format we can proceed with GO enrichment.

    genes = as.integer(pat[,"sc"])
    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

    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(t) #this is for the knitr document
}
```

clustVis_line

This fucntion visualizes gene expression of all genes in clusters.

```
clusterVis_line <- function(clustNum) {
  sub_cluster <- subset(plot.data, ssom.unit.classif==clustNum)
  sub_data <- sub_cluster[,c(1, 2, 9:14)] # just the sample types
  sub_data <- melt(sub_data)
  sub_data <- within(sub_data, lineGroup <- paste(genotype, gene,sep='.'))
  ggplot(sub_data, aes(variable, value, group = lineGroup, color =  genotype )) +
    geom_line(alpha = .1, (aes(color = factor(genotype)))) +
    geom_point(alpha = .0)
  }
```

geneInClust

This function prints out how many genes in cluster and attaches annotation.

```
annotation1<- read.delim("../data/ITAG2.3_all_Arabidopsis_ITAG_annotations.tsv", header=FALSE)  #Change
colnames(annotation1) <- c("ITAG", "SGN_annotation")
annotation2<- read.delim ("../data/ITAG2.3_all_Arabidopsis_annotated.tsv")
annotation <- merge(annotation1,annotation2, by = "ITAG")
#Only Gene Name and ITAG
annotation <- annotation[,c(1,5)]
```

```
genesInClust <- function(clustNum) {
  sub_cluster <- subset(plot.data, ssom.unit.classif==clustNum)
  sub_data <- as.data.frame(sub_cluster[,2])
  colnames(sub_data) <- "ITAG"
  resultsTable <- merge(sub_data,annotation,by = "ITAG", all.x=TRUE)
  print(nrow(resultsTable))
#  return(resultsTable <- unique(resultsTable))
  return(unique(resultsTable))
  }

genesInClust <- function(clustNum, plot.data, annotation) {
  sub_cluster <- subset(plot.data, ssom.unit.classif==clustNum)
  sub_data <- as.data.frame(sub_cluster[,2])
  colnames(sub_data) <- "ITAG"
  resultsTable <- merge(sub_data,annotation,by = "ITAG", all.x=TRUE)
  print(nrow(unique(resultsTable)))
  return(unique(resultsTable))
  }
```

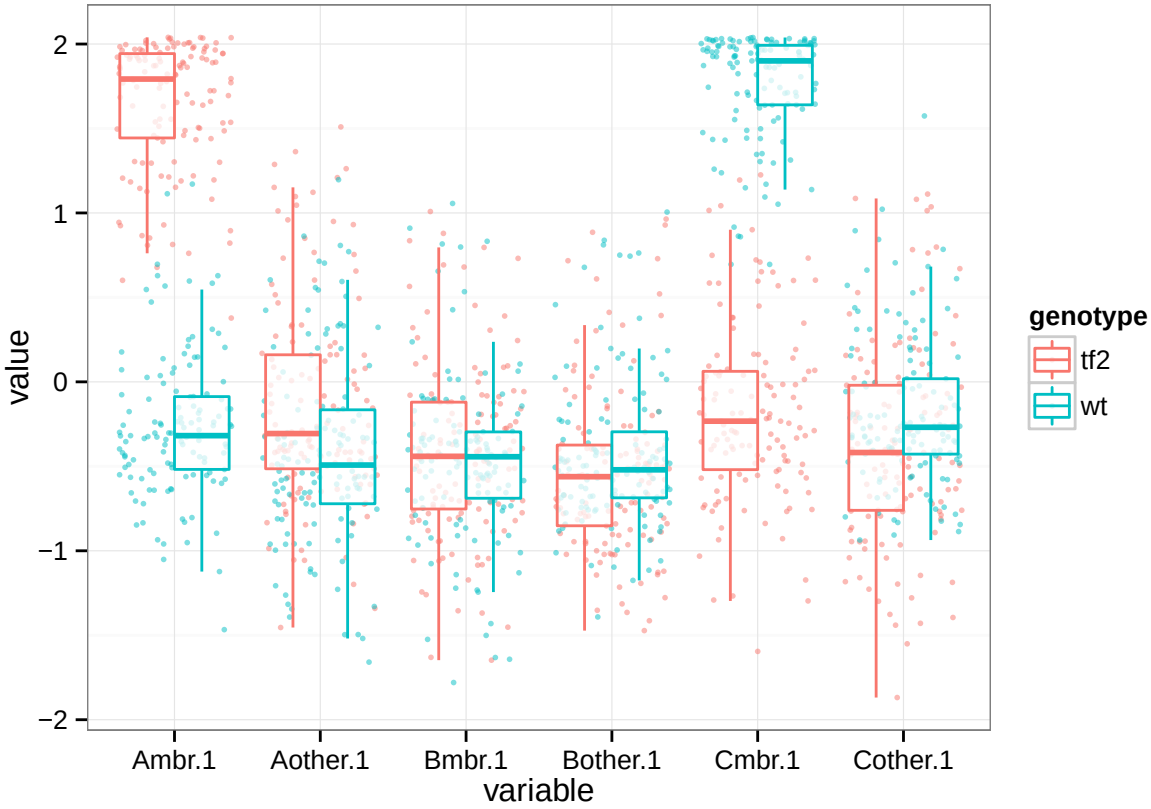
Examples Usage:

Read in data produced from SOM. In this case large superSOM.

```
plot.data <- read.table("../data/ssom.data.analysis5d.txt",header=TRUE)
```

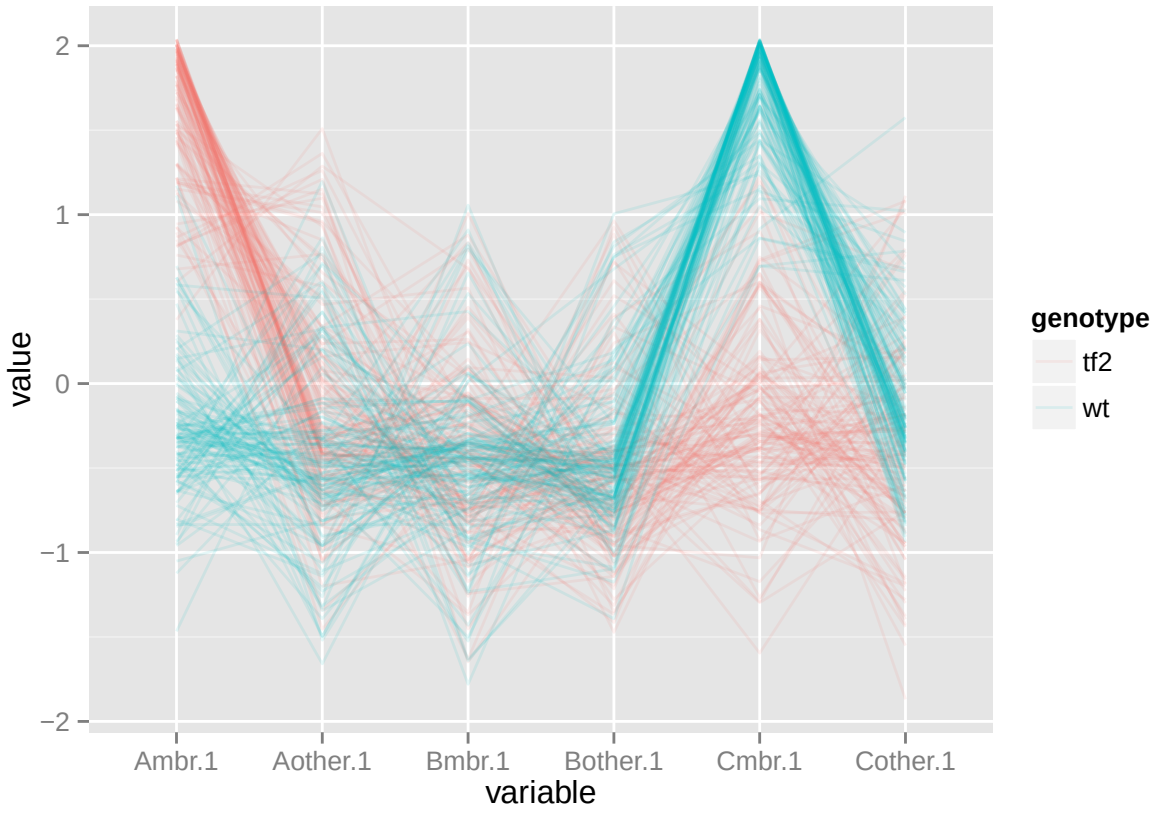
```
clusterVis(4)
```

```
## Using genotype as id variables
```




```
clusterVis_line(4)
```

```
## Using genotype, gene as id variables
```




```
y <- genesInClust(4, plot.data, annotation)
```

```
## [1] 129
```