Self Organizing Maps - Basic

Ciera Martinez LCM Data

Required Libraries

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

Self Organizing Maps

The goal of this analysis is to find genes that have co-expression patterns in tissues of just the WT data.

1. most differentiated genes between tissue.

1.pca.R

First read in file that came from mostSigDEgenes.Rmd. This is a list of genes from all DE analysis in WT. They were all cancatenated, then duplicate genes were removed. In addition the mean was calculated from the replicates of each type.

The first step is to get it into the right format. First column being the genes, while the subsequent columns are the different libraries (type).

```
mostDEgenes <- read.csv("../data/allGeneList.csv")

mostDEgenes <- mostDEgenes[c(7, 1, 4)] #keep only needed columns (gene, type, mean)

#Change from long to wide data format
mostDEgene.long <- cast(mostDEgenes, gene ~ type, value.var = mean, fun.aggregate = "mean")  #why did I
```

```
## Using mean as value column.  Use the value argument to cast to override this choice
```

```
mostDEgene.long <- as.data.frame(mostDEgene.long)
scale_data <- as.matrix(t(scale(t(mostDEgene.long[c(2:7)]))))#transformation.

#Principle Component Analysis
pca <- prcomp(scale_data, scale=TRUE)

summary(pca)
```

```
## Importance of components:
##                          PC1   PC2   PC3   PC4   PC5      PC6
## Standard deviation     1.399 1.116 1.058 0.962 0.868 9.21e-16
## Proportion of Variance 0.326 0.208 0.186 0.154 0.126 0.00e+00
## Cumulative Proportion  0.326 0.534 0.720 0.874 1.000 1.00e+00
```

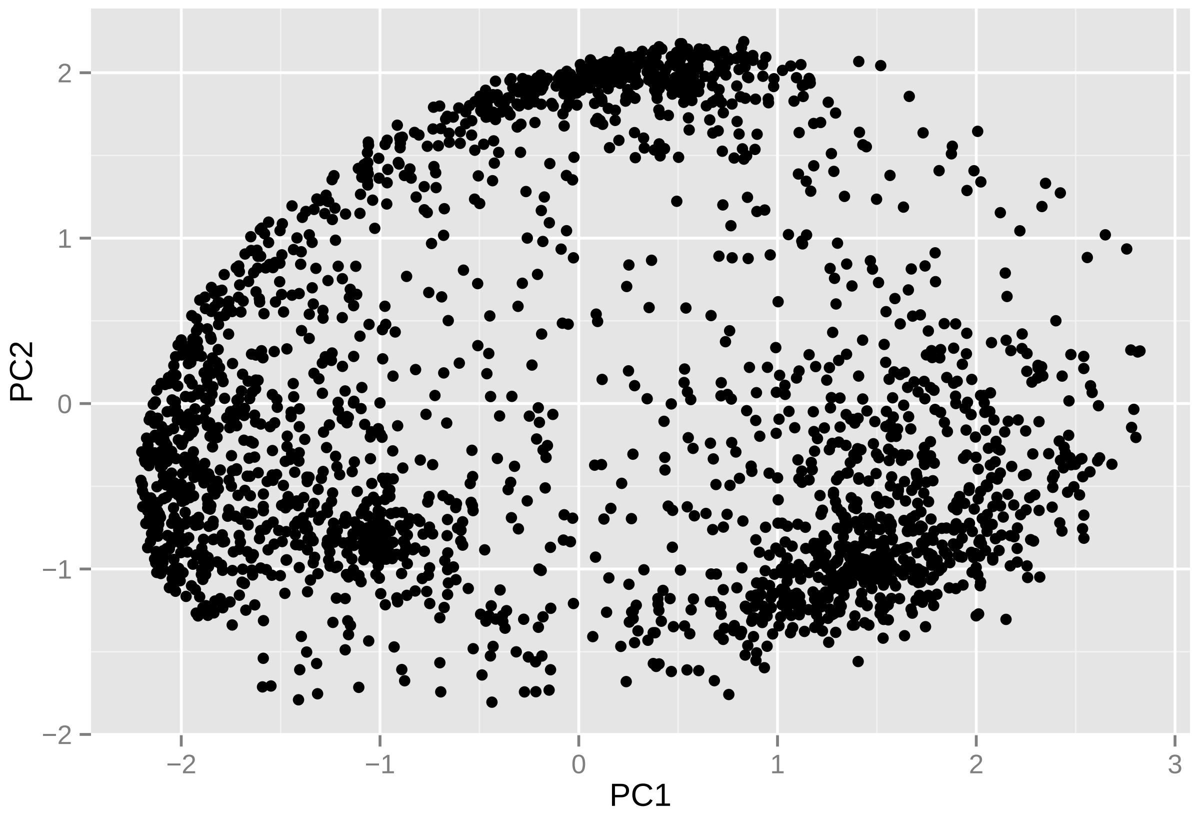
```r
pca.scores <- data.frame(pca$x)

data.val <- cbind(mostDEgene.long, scale_data, pca.scores)
head(data.val)
```

```
##                  gene      Ambr   Aother     Bmbr  Bother    Cmbr  Cother
## 1 Solyc00g005070.1.1   17.1934   4.5236  12.6456   3.462   4.181 105.967
## 2 Solyc00g005080.1.1   16.0215  10.1277   7.5622   8.496   8.413  25.029
## 3 Solyc00g005840.2.1   19.7458  14.0743  11.4830  83.513  15.811  13.981
## 4 Solyc00g005870.1.1    0.1336   0.6414   3.1241   2.779   1.780  12.462
## 5 Solyc00g006470.1.1 2455.5111 605.8620 108.0947 360.482 499.448 390.760
## 6 Solyc00g006670.2.1   66.9760   5.9947   0.6023   7.071  11.065   4.678
##      Ambr  Aother     Bmbr  Bother    Cmbr  Cother     PC1     PC2
## 1 -0.1857 -0.5008 -0.29882 -0.5272 -0.5093  2.0219  0.4124  1.9721
## 2  0.5010 -0.3641 -0.74069 -0.6037 -0.6158  1.8232  0.4094  1.4972
## 3 -0.2381 -0.4399 -0.53216  2.0315 -0.3781 -0.4433 -0.9687 -0.9406
## 4 -0.7372 -0.6256 -0.07972 -0.1557 -0.3751  1.9733  0.2002  2.0773
## 5  2.0024 -0.1524 -0.73231 -0.4383 -0.2764 -0.4030  1.1164 -1.0607
## 6  2.0226 -0.4000 -0.61428 -0.3573 -0.1986 -0.4524  1.3269 -1.0424
##        PC3     PC4     PC5        PC6
## 1  0.62321 0.23135  0.1379 -1.110e-15
## 2  1.14022 0.75216 -0.1592 -1.332e-15
## 3 -0.45945 0.47067  1.6559  1.499e-15
## 4  0.08514 0.01114  0.5230 -9.992e-16
## 5  0.54497 0.93811 -0.4748 -5.551e-16
## 6  0.32412 0.97528 -0.2106 -1.665e-16
```

Visualizing the PCA

Looks to be three major clusters.

```r
p <- ggplot(data.val, aes(PC1, PC2))
p + geom_point()
```



1. Self Organizing Map - (6,6) Large

The size of the map is something that may cause differences in the genes that are clustered. Using a small map size (3,2), I found they cluster in according to tissue type. This makes the interpretation of the results pretty straight forward. My only worry is that the map might not be large enough, considering [1], suggests that you size of the map based on count distribution, the goal being an even distribution, with no "peak" counts in any one cluster while also having no empty clusters.

The only way to see how this is affects what we see is to compare the clusters of the small (3,2) and large map (6,6).

```
names(data.val)
```

```
##  [1] "gene"   "Ambr"   "Aother" "Bmbr"   "Bother" "Cmbr"   "Cother"
##  [8] "Ambr"   "Aother" "Bmbr"   "Bother" "Cmbr"   "Cother" "PC1"
## [15] "PC2"    "PC3"    "PC4"    "PC5"    "PC6"
```

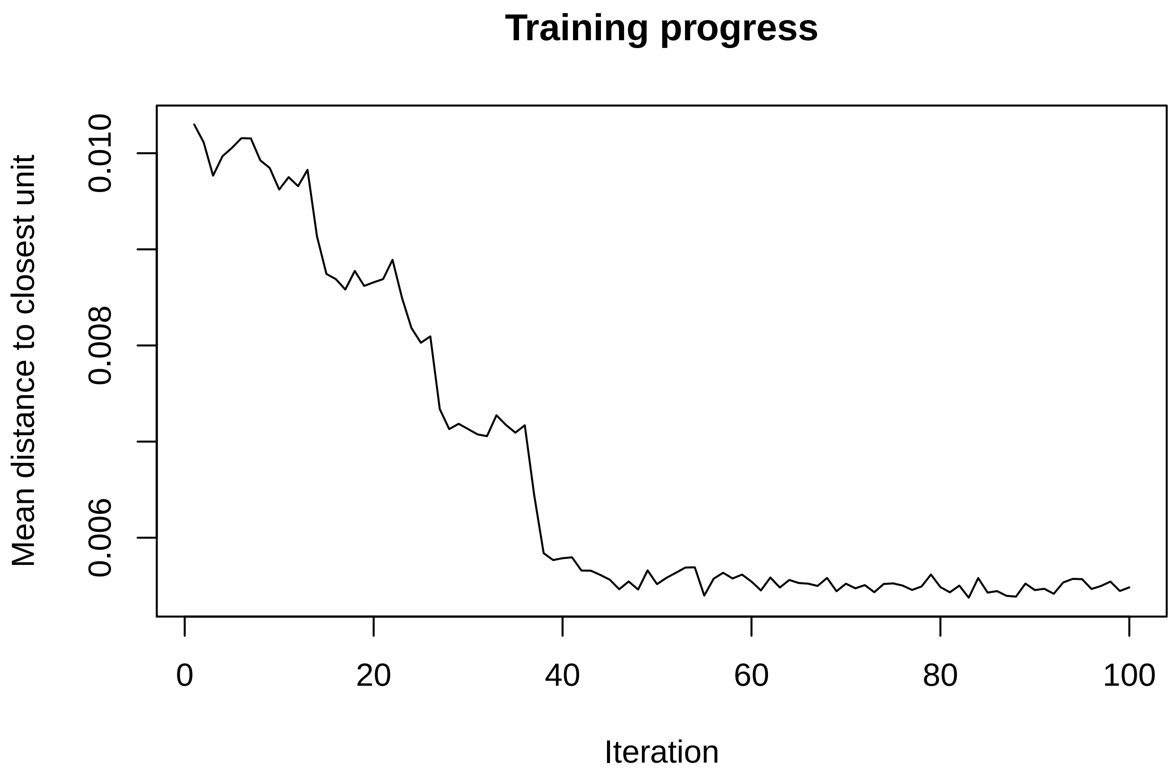
```
som.data <- as.matrix(data.val[,c(9:14)])  #subset only the scaled gene expression values

set.seed(2)

som <- som(data=som.data, somgrid(6,6,"hexagonal")) # This is where you change the size of the map
summary(som)
```

```
## som map of size 6x6 with a hexagonal topology.
## Training data included; dimension is 2249 by 6
## Mean distance to the closest unit in the map: 0.4074
```

Training Plot ("changes") - Large

This shows a hundred iterations. Training decreases with iterations and plateaus at around 40 iterations. Ideally you want the the training to reach a minimum plateau. In the example online, the decrease to this plateau happens slowly with a slow decline to the minimum plateau. I should look into what this sudden drop means.

```
plot(som, type ="changes")
```

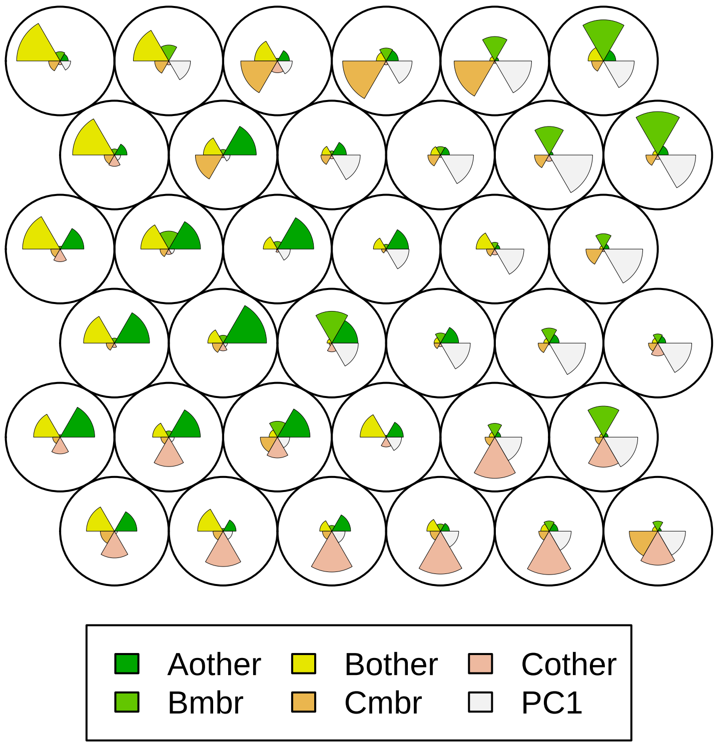


Code Plot - Large

The the code plot shows each cluster and the node wieght vectors or "codes" associated with each node. These are made up of the original normalized values of the original values used to generate the map. You should see patterns of clustering.

The fan chart in the center of the clusters reveals the characteristics that define how the genes were clustered into each particular cluster. For instance if one cluster has only one large fan piece, say for Bother, this is telling us that most of the genes in this cluster were grouped because of similar normalized gene count value of the Bother region. We do not know the degree, it could mean all these genes are up-regulated or down-regulated in the Bother region, but we do not know which at this point.

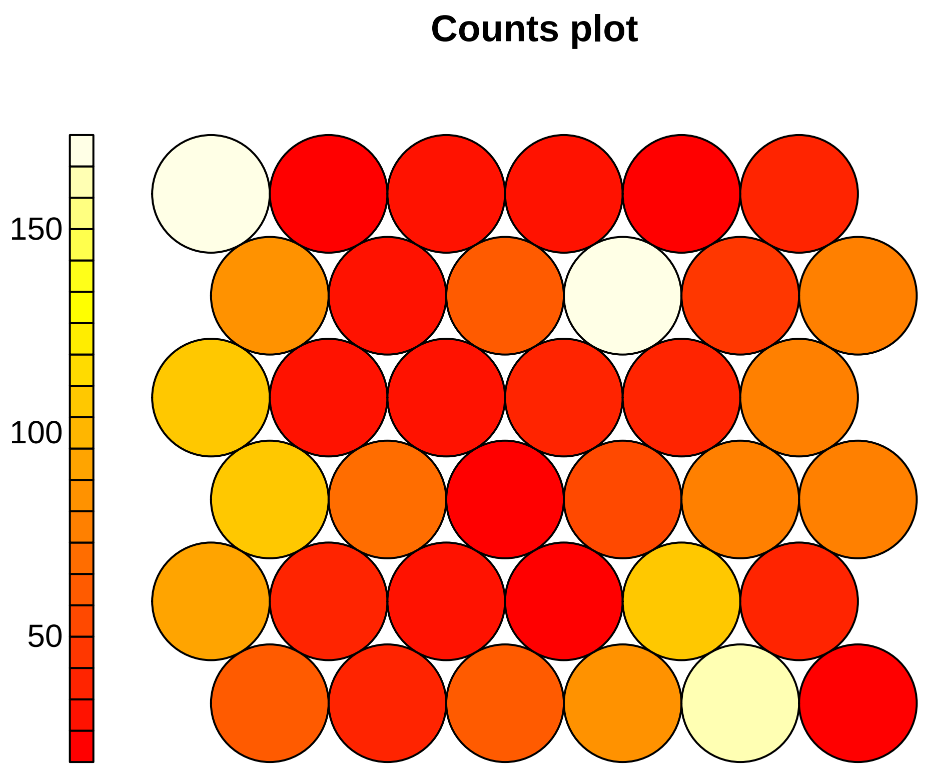
```
plot(som, type = "codes")
```



Count Plot - Large

This tells you how many genes are in each of the clusters. The count plot can be used as a quality check. Ideally you want a uniform distribution. If there are some peaks in certain areas, this means you should likely increase the map size. If you have empty nodes you should decrease the map size [1].

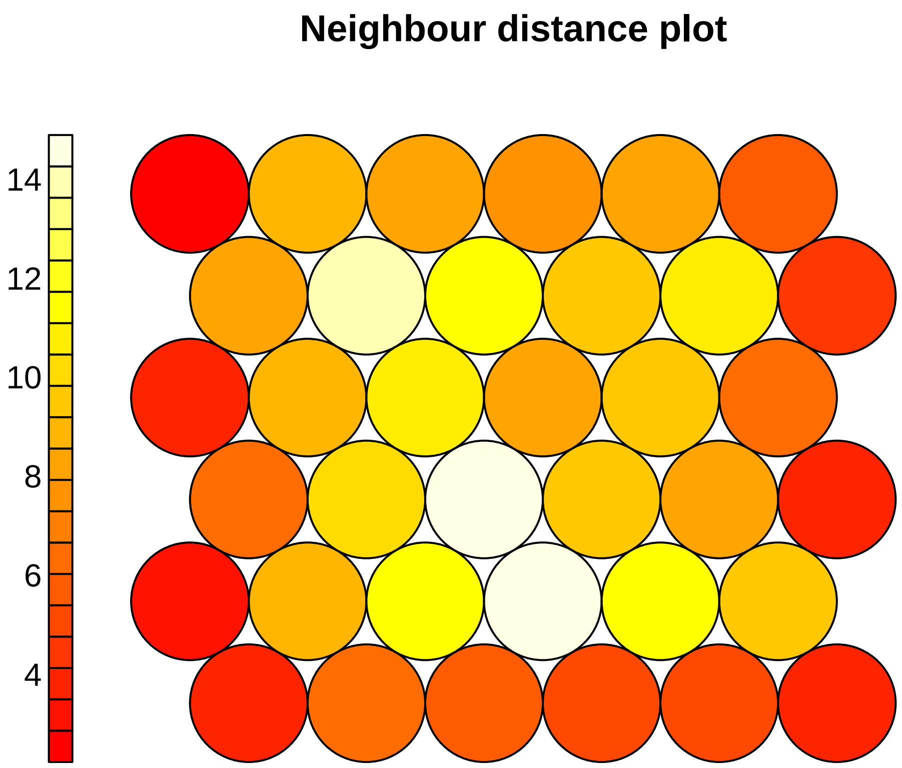
```
plot(som, type = "counts")
```



Distance Neighbour Plot - Large

This is sometimes called the "U-Matrix", it can help identify further clustering. Areas of low neighbour distance indicate groups of nodes that are similar and the further apart nodes indicate natural "borders" in the map.

```r
plot(som, type="dist.neighbours")
```



Heatmaps - large

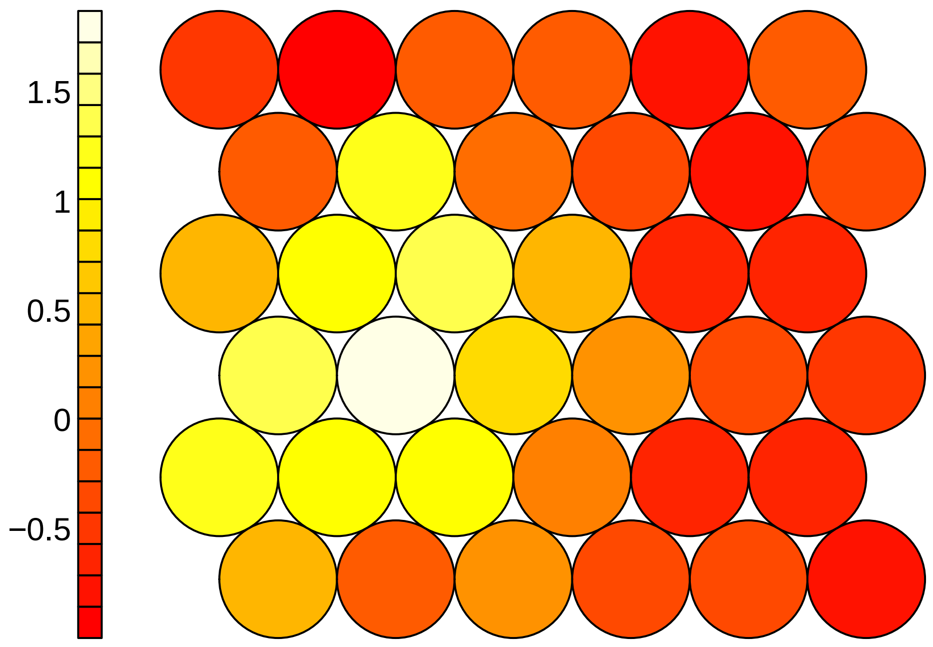
This shows the distribution of each tissue type across the whole SOM.

```
head(som$codes)
```

```
##         Aother    Bmbr  Bother    Cmbr Cother     PC1
## [1,]  0.4882 -1.0325  0.8928 -0.1579 0.9932 -1.7181
## [2,] -0.1423 -0.8477  0.7131 -0.4279 1.5066 -1.0391
## [3,]  0.2481 -0.6481 -0.2393 -0.4800 1.8337 -0.6406
## [4,] -0.4011 -0.5496 -0.1285 -0.4124 1.9570 -0.1185
## [5,] -0.4082 -0.3479 -0.5027 -0.4115 1.9907  0.2739
## [6,] -0.7580 -0.3591 -0.7098  0.8107 1.4232  0.8640
```

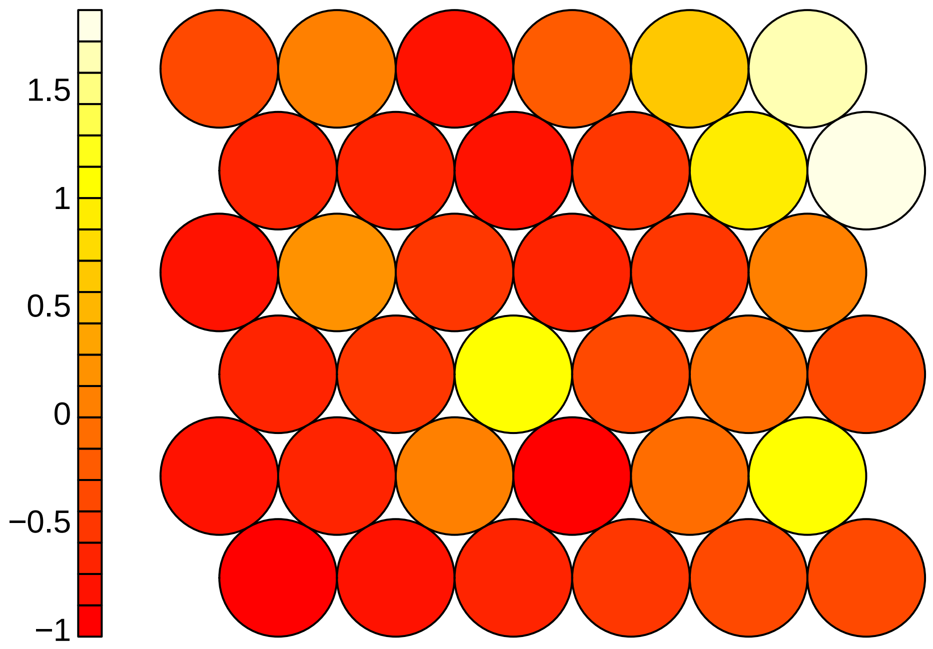
```
som$data <- data.frame(som$data) #changed to dataframe to extract column names easier.

#This is just a loop that plots the distribution of each tissue type across the map.
for (i in 1:6){
  plot(som, type = "property", property = som$codes[,i], main=names(som$data)[i])
  print(plot)
}
```

Aother

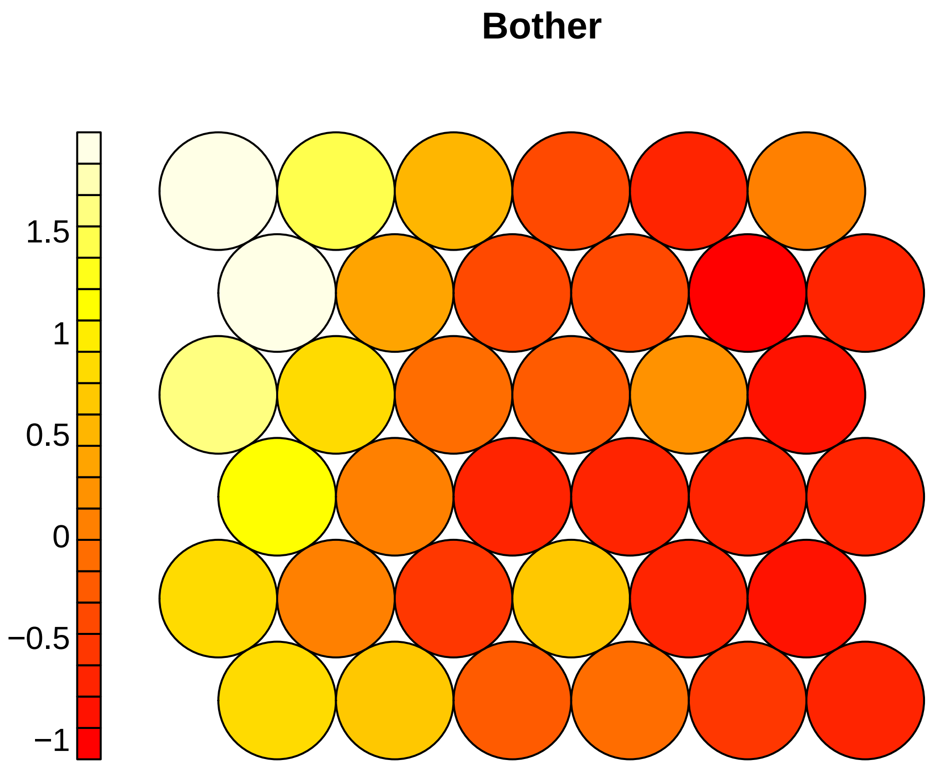


```
## function (x, y, ...)
## UseMethod("plot")
## <bytecode: 0x7fa40237fcd0>
## <environment: namespace:graphics>
```

Bmbr

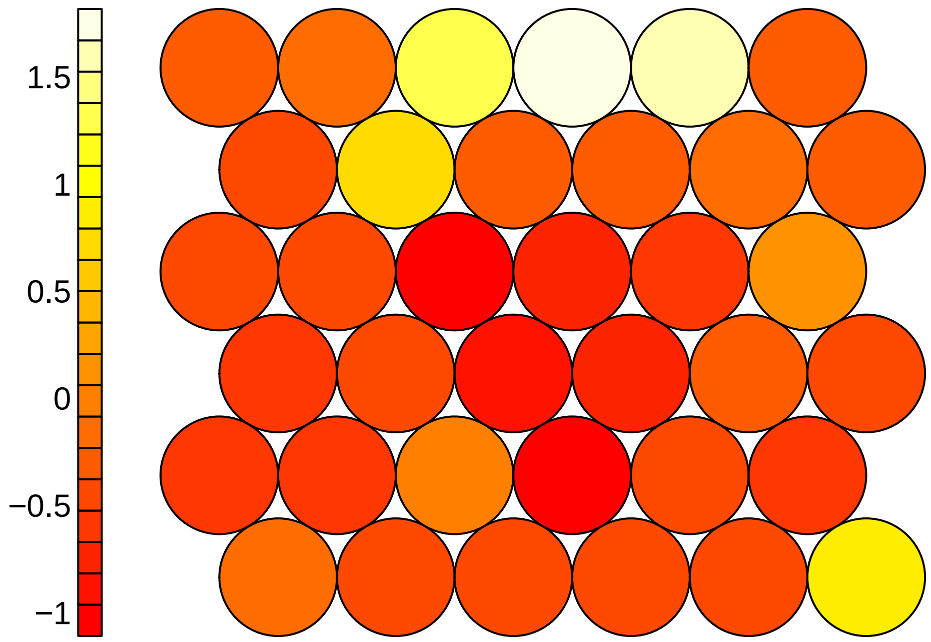


```
## function (x, y, ...)
## UseMethod("plot")
## <bytecode: 0x7fa40237fcd0>
## <environment: namespace:graphics>
```



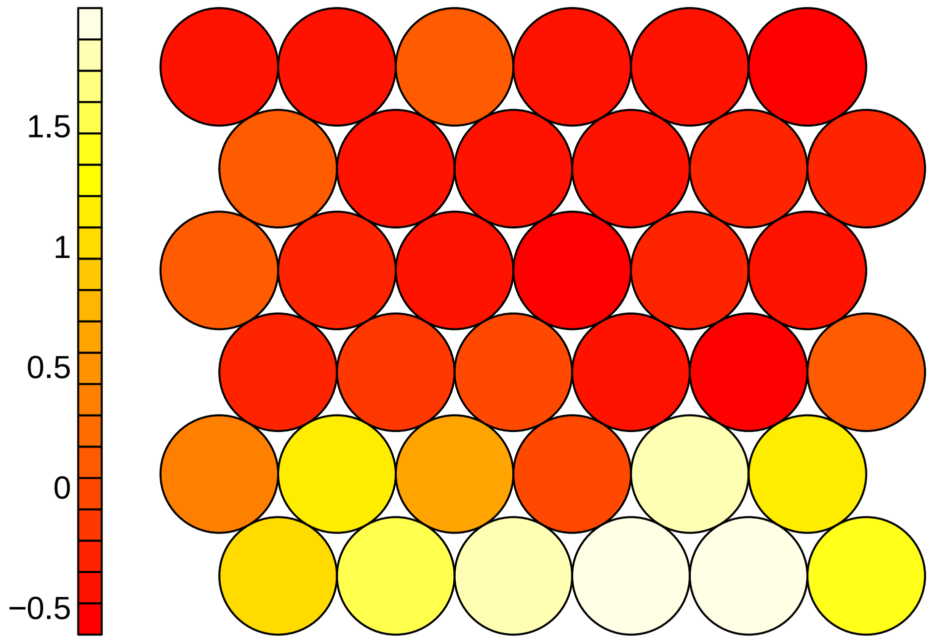
```
## function (x, y, ...)
## UseMethod("plot")
## <bytecode: 0x7fa40237fcd0>
## <environment: namespace:graphics>
```

Cmbr

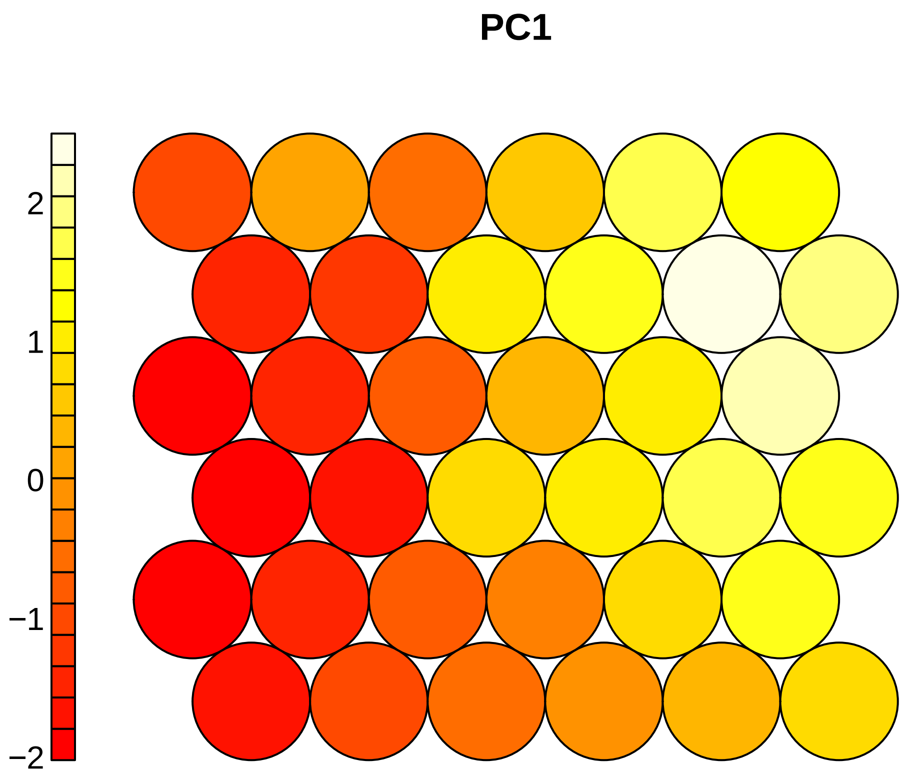


```
## function (x, y, ...)
## UseMethod("plot")
## <bytecode: 0x7fa40237fcd0>
## <environment: namespace:graphics>
```

Cother



```
## function (x, y, ...)
## UseMethod("plot")
## <bytecode: 0x7fa40237fcd0>
## <environment: namespace:graphics>
```



```
## function (x, y, ...)
## UseMethod("plot")
## <bytecode: 0x7fa40237fcd0>
## <environment: namespace:graphics>
```

Clustering Plot - Large

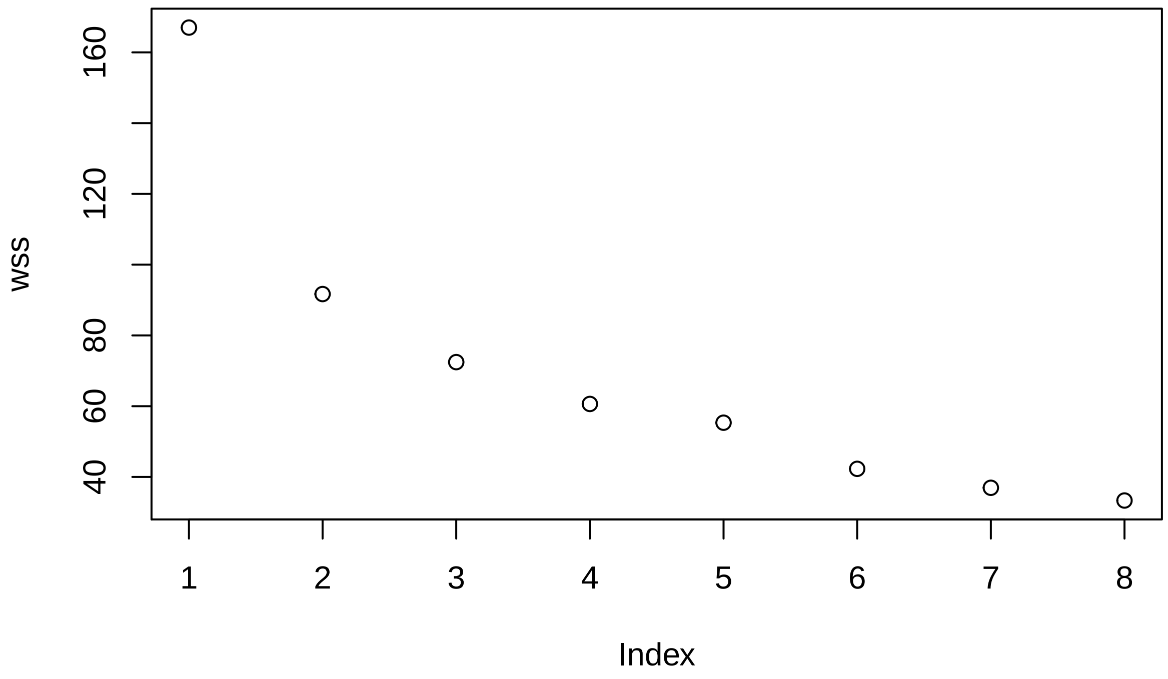
This groups clusters based on similar "metrics". The advice given from [1], suggests that the heatmap should be used to view the overall "story" of the map. Essentiatlly you are taking all the tissue types into account and viewing it all on one heat map.

Estimate of the number of clusters that would be suitable can be ascertained using a kmeans algorithm and examing for an "elbow-point" in the plot of "within cluster sum of squares[1].

```
mydata <- som$codes
head(mydata)
```

```
##       Aother    Bmbr  Bother    Cmbr Cother     PC1
## [1,]  0.4882 -1.0325  0.8928 -0.1579 0.9932 -1.7181
## [2,] -0.1423 -0.8477  0.7131 -0.4279 1.5066 -1.0391
## [3,]  0.2481 -0.6481 -0.2393 -0.4800 1.8337 -0.6406
## [4,] -0.4011 -0.5496 -0.1285 -0.4124 1.9570 -0.1185
## [5,] -0.4082 -0.3479 -0.5027 -0.4115 1.9907  0.2739
## [6,] -0.7580 -0.3591 -0.7098  0.8107 1.4232  0.8640
```

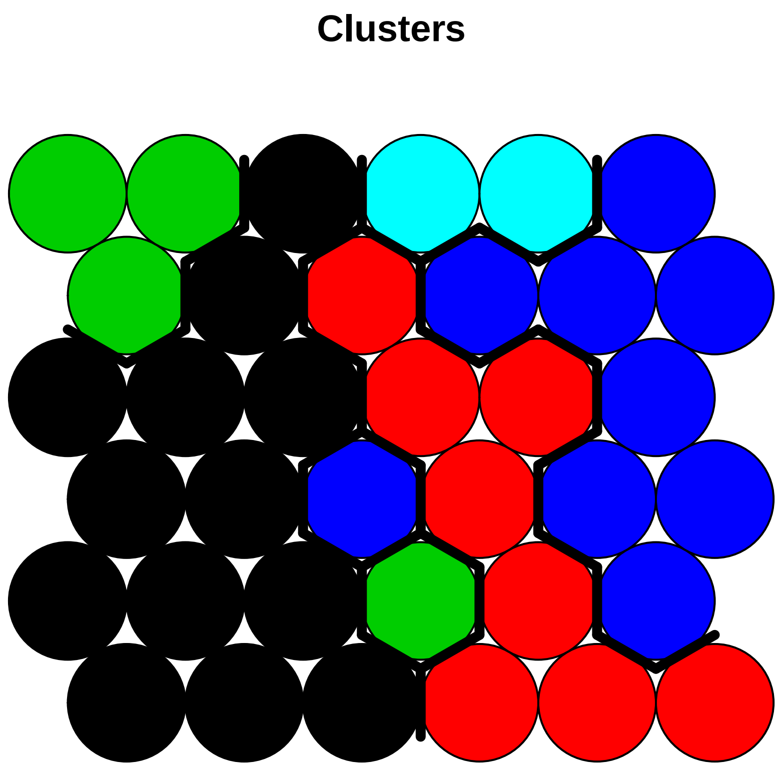
```r
wss <- (nrow(mydata)-1)*sum(apply(mydata,2,var))

for (i in 1:8) {
  wss[i] <- sum(kmeans(mydata, centers=i)$withinss)
}

plot(wss)
```



This is setting the larger clusters that incorporate multiple clusters.

Use hierarchical clustering to cluster the codebook vectors.

```r
som_cluster <- cutree(hclust(dist(som$codes)), 5) #Set cluster #.

# plot these results:
plot(som, type="mapping", bgcol = som_cluster, main = "Clusters")
add.cluster.boundaries(som, som_cluster)
```



```r
# I want to attach the hierchal clusters
#to the larger dataset data.val.
som_clusterKey <- data.frame(som_cluster)
som_clusterKey$unit.classif <- c(1:36)

data.val <- cbind(data.val,som$unit.classif,som$distances)

#Merge data.val with som_clusterKey
##change data.val to match som_cluster key
names(data.val)[20] <- "unit.classif"

data.val <- merge(data.val, som_clusterKey, by.x = "unit.classif" ) #ignore warning, this is what you w
```

```
## Warning: column names 'Ambr', 'Aother', 'Bmbr', 'Bother', 'Cmbr', 'Cother'
## are duplicated in the result
```

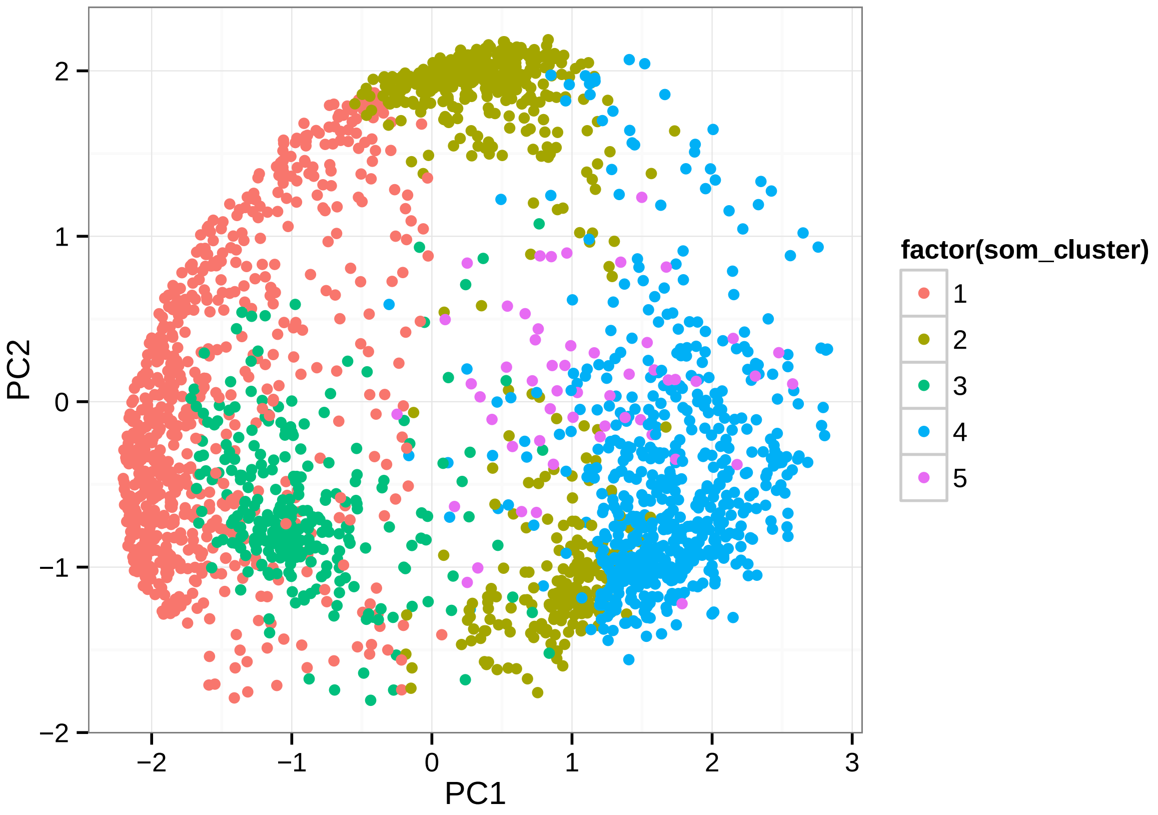
```r
#Make sure that there is just one of each value
#som$unit.classif and distances column.
names(data.val)
```

```
##  [1] "unit.classif"  "gene"          "Ambr"          "Aother"
##  [5] "Bmbr"          "Bother"        "Cmbr"          "Cother"
##  [9] "Ambr"          "Aother"        "Bmbr"          "Bother"
## [13] "Cmbr"          "Cother"        "PC1"           "PC2"
## [17] "PC3"           "PC4"           "PC5"           "PC6"
## [21] "som$distances" "som_cluster"
```

Other Visualization - Large

Visualize by Cluster

```
plot.data <- data.val

p <- ggplot(plot.data, aes(PC1, PC2, colour=factor(som_cluster)))
#notice I am using som_cluster and not unit.classif
p + geom_point() + theme_bw()
```



Visualize by individual clusters - Large

```
sub_cluster <- subset(plot.data, som_cluster =="5")

sub_data <- sub_cluster[,9:14] # just the sample types

names(sub_data)
```

```
## [1] "Ambr.1"   "Aother.1" "Bmbr.1"   "Bother.1" "Cmbr.1"   "Cother.1"
```

```
head(sub_data)
```

```
##        Ambr.1 Aother.1   Bmbr.1 Bother.1 Cmbr.1 Cother.1
## 2162 -0.1294  -0.7154 -0.24138  -0.3381  1.994  -0.5694
## 2163 -0.3762  -0.6806  0.02803  -0.2188  1.960  -0.7120
## 2164 -1.3404   0.6087 -0.17747  -0.2458  1.596  -0.4409
## 2165 -0.4504  -0.2270 -0.33407  -0.5506  2.028  -0.4662
## 2166 -0.5880  -0.4759 -0.58799  -0.4335  1.970   0.1154
## 2167  0.4085  -0.1202 -0.91733  -0.4688  1.805  -0.7070
```

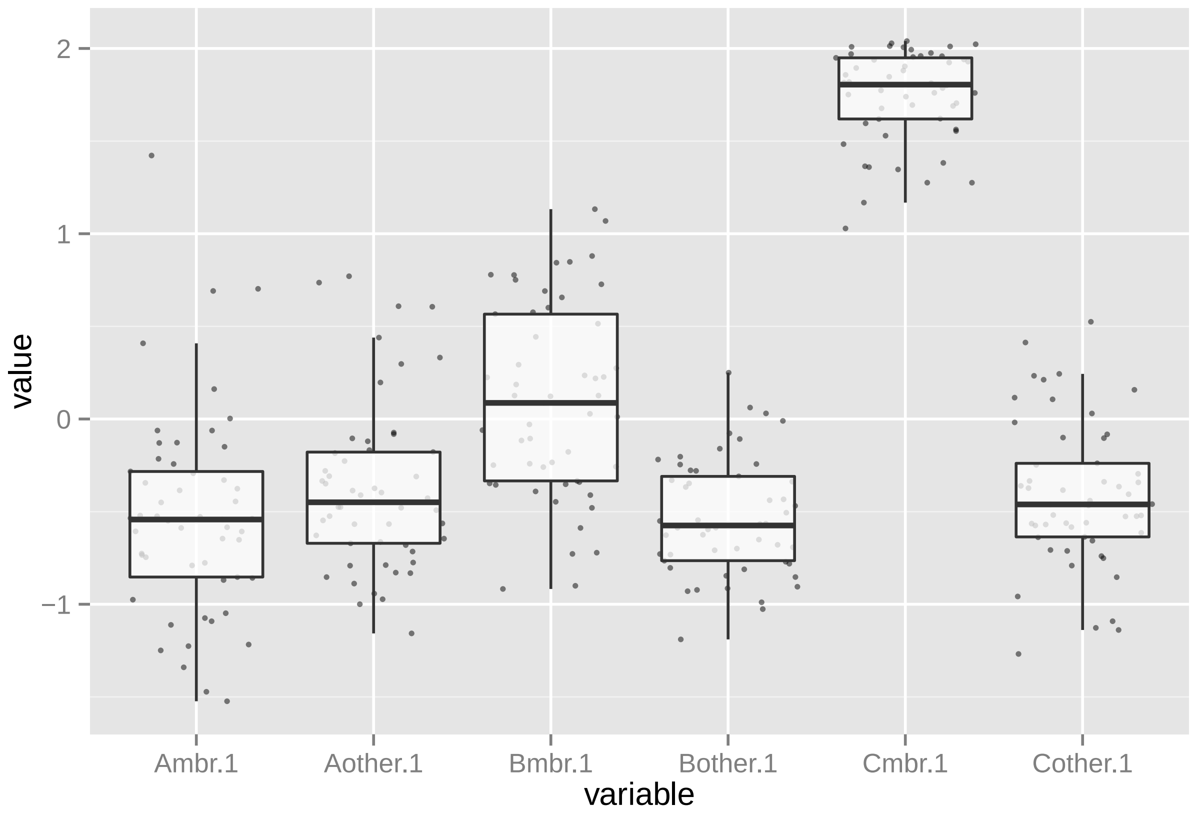
```
m.data <- melt(sub_data)
```

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

```
head(m.data)
```

```
##   variable   value
## 1   Ambr.1 -0.1294
## 2   Ambr.1 -0.3762
## 3   Ambr.1 -1.3404
## 4   Ambr.1 -0.4504
## 5   Ambr.1 -0.5880
## 6   Ambr.1  0.4085
```

```
p <- ggplot(m.data, aes(x=variable, y=value))
p + geom_point(alpha=0.5, position="jitter", size=1) + geom_boxplot(alpha=0.75, outlier.size=0)
```


value
Ambr.1
Aother.1
Bmbr.1
Bother.1
Cmbr.1
Cother.1
variable