Brain Cancer Microarray Data Weighted Gene Co-expression Network Analysis R Tutorial

Steve Horvath, Bin Zhang, Jun Dong, Tova Fuller, Peter Langfelder

CONTENTS

This is stolen from: [https://horvath.genetics.ucla.edu/html/GeneralFramework/GBMTutorialHorvath.pdf]

This is soft clustering only!!

This document contains function for carrying out the following tasks

A) Assessing scale free topology and choosing the parameters of the adjacency function using the scale free topology criterion (Zhang and Horvath 05)

B) Computing the topological overlap matrix

C) Defining gene modules using clustering procedures

D) Summing up modules by their first principal component (first eigengene)

E) Relating a measure of gene significance to the modules

F) Carrying out a within module analysis (computing intramodular connectivity) and relating intramodular connectivity to gene significance.

G) Miscellaneous other functions, e.g. for computing the cluster coefficient.

Downloading the R software

1) Go to http://www.R-project.org, download R and install it on your computer

After installing R, you need to install several additional R library packages:

To get this tutorial and data files, go to the following webpage [www.genetics.ucla.edu/labs/horvath/CoexpressionNetwork]

Download the zip file containing:

1) R function file: "NetworkFunctions.txt", which contains several R functions needed for Network Analysis.
2) The data file "gbm55old_dchip_14kALL_cox_8000mvgenes2.csv"
3) Of course, this file: "GBMTutorialHorvath.txt"

(I already did this)

```
source("NetworkFunctions.R")
# read in the R libraries
library(MASS) # standard, no need to install
library(class) # standard, no need to install
library(cluster)
#library(impute) # install it for imputing missing value
library(WGCNA)
options(stringsAsFactors = F)
```

read in the 8000 most varying genes (GBM microarray data)

```
dat0=read.csv("gbm55old_dchip_14kALL_cox_8000mvgenes2.csv")
# this contains information on the genes
datSummary=dat0[,1:9]
```

The following data frame contains the gene expression data: columns are genes, rows are arrays (samples)

```
datExpr = t(dat0[,10:64])

no.samples = dim(datExpr)[[1]]
dim(datExpr)
```

```
## [1]   55 8000
```

```
rm(dat0);gc()
```

```
##           used  (Mb) gc trigger  (Mb) limit (Mb) max used  (Mb)
## Ncells 4750343 253.7    8382777 447.7         NA  6287972 335.9
## Vcells 8720055  66.6   16865277 128.7      65536 12137706  92.7
```

To choose a cut-off value,

we propose to use the Scale-free Topology Criterion (Zhang and Horvath 2005). Here the focus is on the linear regression model fitting index (denoted below by scale.law.R.2) that quantify the extent of how well a network satisfies a scale-free topology. The function PickSoftThreshold can help one to estimate the cut-off value when using hard thresholding with the step adjacency function. The first column (different from the row numbers) lists the soft threshold Power The second column reports the resulting scale free topology fitting index R^2 (scale.law.R.2) The third column reports the slope of the fitting line. The fourth column reports the fitting index for the truncated exponential scale free model# Usually we ignore it. The remaining columns list the mean, median and maximum connectivity. To a soft threshold (power) with the scale free topology criterion: aim for reasonably high scale free R^2 (column 2), higher than say .80 and negative slope (around -1, col 4). In practice, we pick the threshold by looking for a "kink" in the relationship between R^2 and power, see below.

Soft thresholding

Now we investigate soft thesholding with the power adjacency function

```
powers1=c(seq(1,10,by=1),seq(12,20,by=2))
RpowerTable=pickSoftThreshold(datExpr, powerVector=powers1)[[2]]
```

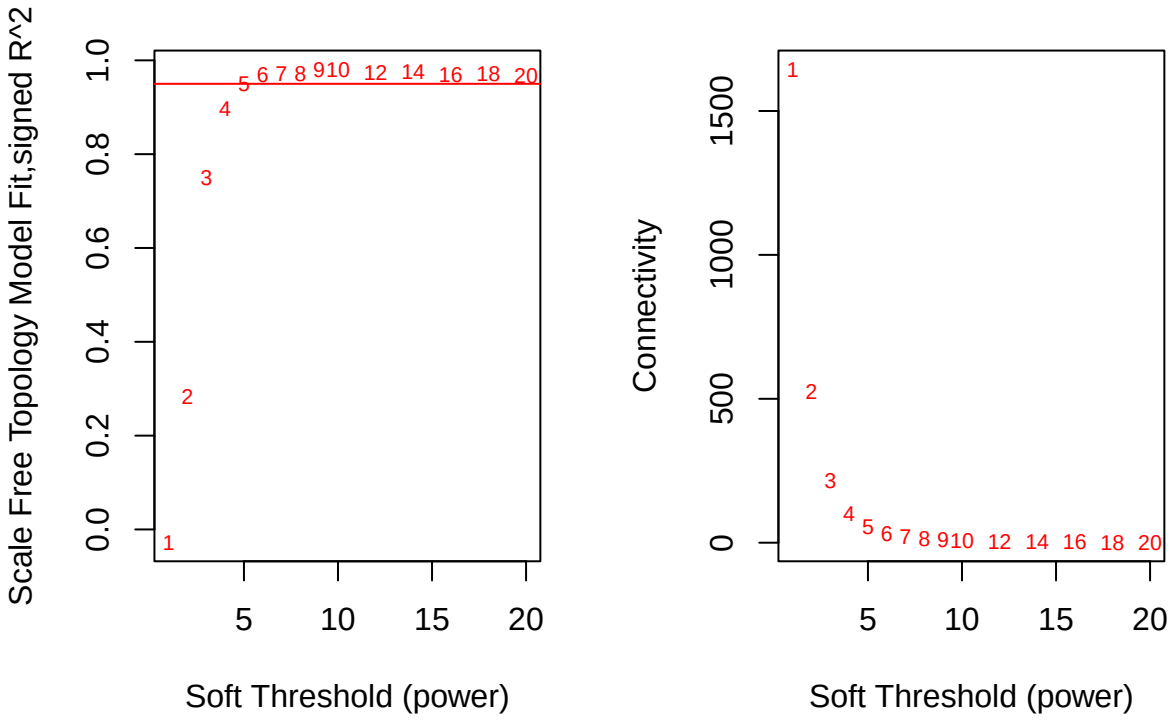
```
##    Power SFT.R.sq  slope truncated.R.sq mean.k. median.k. max.k.
## 1      1   0.0275  0.451          0.979 1640.00  1.60e+03 2700.0
## 2      2   0.2840 -0.851          0.942  527.00  4.79e+02 1310.0
## 3      3   0.7510 -1.420          0.968  214.00  1.74e+02  769.0
## 4      4   0.8980 -1.660          0.975  102.00  7.22e+01  513.0
## 5      5   0.9500 -1.710          0.979   54.90  3.25e+01  373.0
## 6      6   0.9690 -1.660          0.983   32.50  1.58e+01  288.0
## 7      7   0.9720 -1.610          0.980   20.80  8.09e+00  232.0
## 8      8   0.9720 -1.550          0.976   14.10  4.36e+00  193.0
## 9      9   0.9790 -1.500          0.982   10.10  2.43e+00  166.0
## 10    10   0.9810 -1.470          0.984    7.48  1.42e+00  145.0
## 11    12   0.9750 -1.400          0.980    4.52  5.22e-01  114.0
```

```
## 12    14   0.9770 -1.360          0.982    2.97  2.08e-01   92.7
## 13    16   0.9700 -1.340          0.971    2.08  8.94e-02   76.9
## 14    18   0.9720 -1.330          0.980    1.52  4.00e-02   65.3
## 15    20   0.9670 -1.320          0.973    1.14  1.87e-02   56.2
```

```
gc()
```

```
##            used  (Mb) gc trigger   (Mb) limit (Mb)  max used   (Mb)
## Ncells  4775563 255.1    8382777  447.7         NA   6287972  335.9
## Vcells  8775907  67.0  173038957 1320.2      65536 336570879 2567.9
```

```
cex1=0.7
par(mfrow=c(1,2))
plot(RpowerTable[,1], -sign(RpowerTable[,3])*RpowerTable[,2],xlab="
Soft Threshold (power)",ylab="Scale Free Topology Model Fit,signed R^2",type="n")
text(RpowerTable[,1], -sign(RpowerTable[,3])*RpowerTable[,2],
labels=powers1,cex=cex1,col="red")
# this line corresponds to using an R^2 cut-off of h
abline(h=0.95,col="red")
plot(RpowerTable[,1], RpowerTable[,5],xlab="Soft Threshold (power)",ylab="Mean
Connectivity", type="n")
text(RpowerTable[,1], RpowerTable[,5], labels=powers1, cex=cex1,col="red")
```



Note that at power=6, the curve has an elbow or kink, i.e. for this power the scale free topology fit does not improve after increasing the power. This is why we choose beta1=6 Also the scale free topology criterion with a R^2 threshold of 0.95 would lead us to pick a power of 6. Note that there is a natural trade-off between maximizing scale-free topology model fit (R^2) and maintaining a high mean number of connections: parameter values that lead to an R^2 value close to 1 may lead to networks with very few connections. Actually, we consider a signed version of the scale free topology fitting index. Since it is biologically implausible that a networks contains more hub genes than non-hub genes, we multiply R^2 with -1 if the slope of the regression line between log_{10}(p(k)) and log_{10}(k) is positive.

These considerations motivate us to propose the following {scale-free topology criterion} for choosing the

parameters of an adjacency function: Only consider those parameter values that lead to a network satisfying scale-free topology at least approximately, e.g. signed R^2>0.80. In addition, we recommend that the user take the following additional considerations into account when choosing the adjacency function parameter. First, the mean connectivity should be high so that the network contains enough information (e.g. for module detection). Second, the slope of the regression line should be around -1. When considering the power adjacency functions, we find the relationship between R^2 and the adjacency function parameter (beta) is characterized by a saturation curve type of. In our applications, we use the first parameter value where saturation is reached as long as it is above 0.8. Below we study how the biological findings depend on the choice of the power.

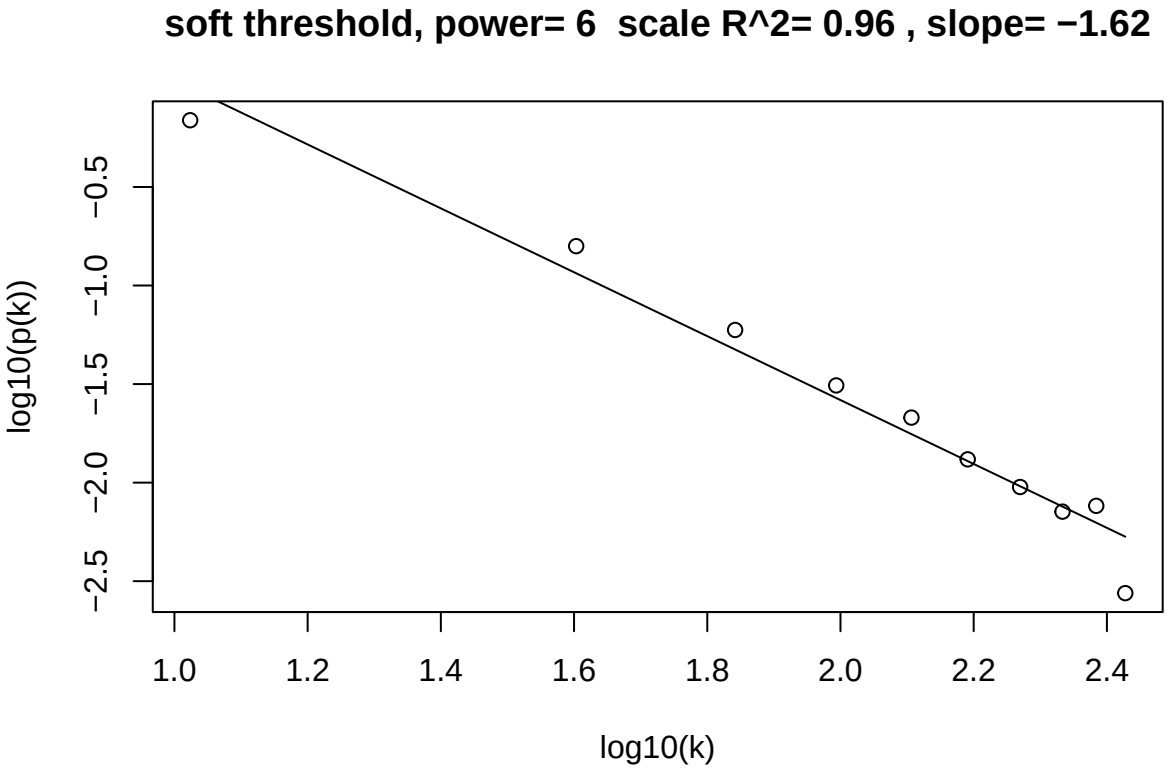
```
beta1=6
Connectivity=softConnectivity(datExpr,power=beta1)-1
```

```
##  softConnectivity: FYI: connecitivty of genes with less than 19 valid samples will be returned as NA
##  ..calculating connectivities..
```

Let's create a scale free topology plot.

The black curve corresponds to scale free topology and the red curve corresponds to truncated scale free topology.

```
par(mfrow=c(1,1))
scaleFreePlot(Connectivity, main=paste("soft threshold, power=",beta1), truncated=F);
```




```
##   scaleFreeRsquared slope
## 1              0.96 -1.62
```

Module Detection

An important step in network analysis is module detetion. Here we use methods that use clustering in combination with the topological overlap matrix. This code allows one to restrict the analysis to the most

connected genes, which may speed up calculations when it comes to module detection.

```
ConnectivityCut = 3600 # number of most connected genes that will be considered

# Incidentally, in the paper by Mischel et al (2005) we considered all 3600 #genes.
ConnectivityRank = rank(-Connectivity)
restConnectivity = ConnectivityRank <= ConnectivityCut

# thus our module detection uses the following number of genes
sum(restConnectivity)
```

```
## [1] 3600
```

```
# Now we define the adjacency matrix for the 3600 most connected genes
ADJ= adjacency(datExpr[,restConnectivity],power=beta1)
gc()
```

```
##            used  (Mb) gc trigger   (Mb) limit (Mb)  max used   (Mb)
## Ncells  4789668 255.8    8382777  447.7         NA   6287972  335.9
## Vcells 21787007 166.3  138431166 1056.2      65536 336570879 2567.9
```

```
# The following code computes the topological overlap matrix based on the
# adjacency matrix.
# TIME: This about a few minutes....
dissTOM=TOMdist(ADJ)
```

```
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
```

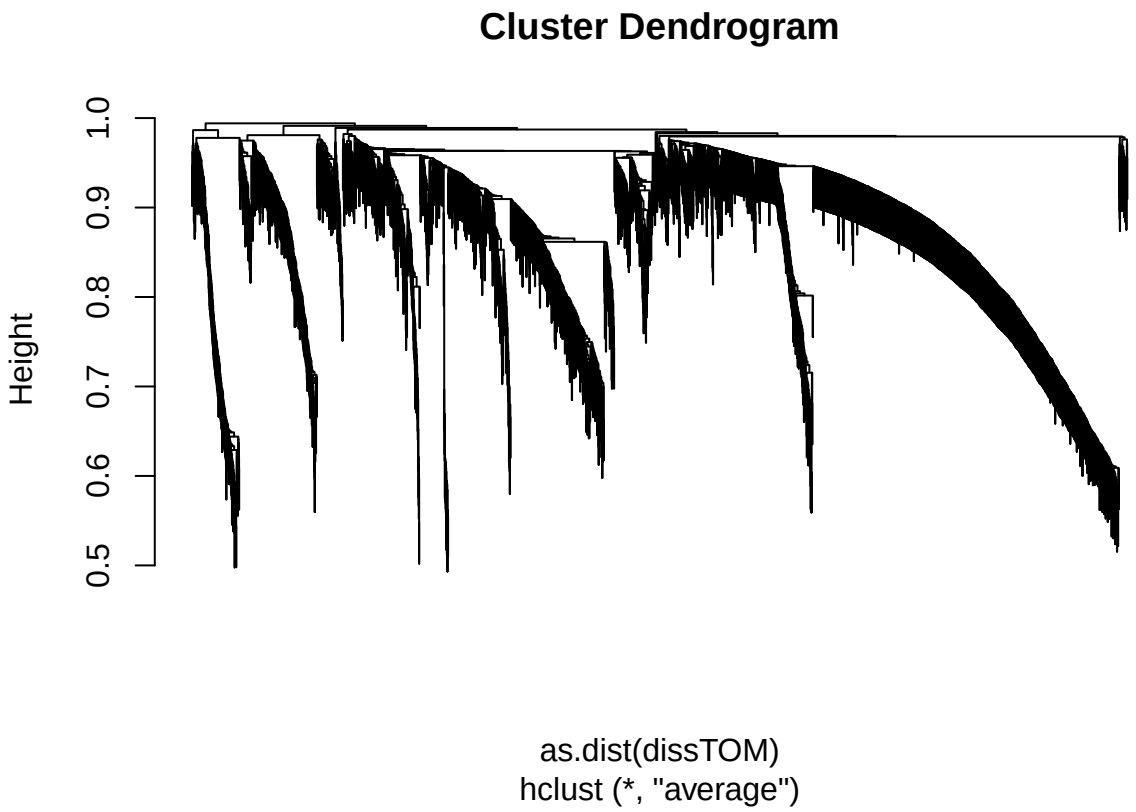
```
gc()
```

```
##            used  (Mb) gc trigger  (Mb) limit (Mb)  max used   (Mb)
## Ncells  4790253 255.9    8382777 447.7         NA   6287972  335.9
## Vcells 34748451 265.2  110744933 845.0      65536 336570879 2567.9
```

Now we carry out hierarchical clustering with the TOM matrix.

This takes a couple of minutes.

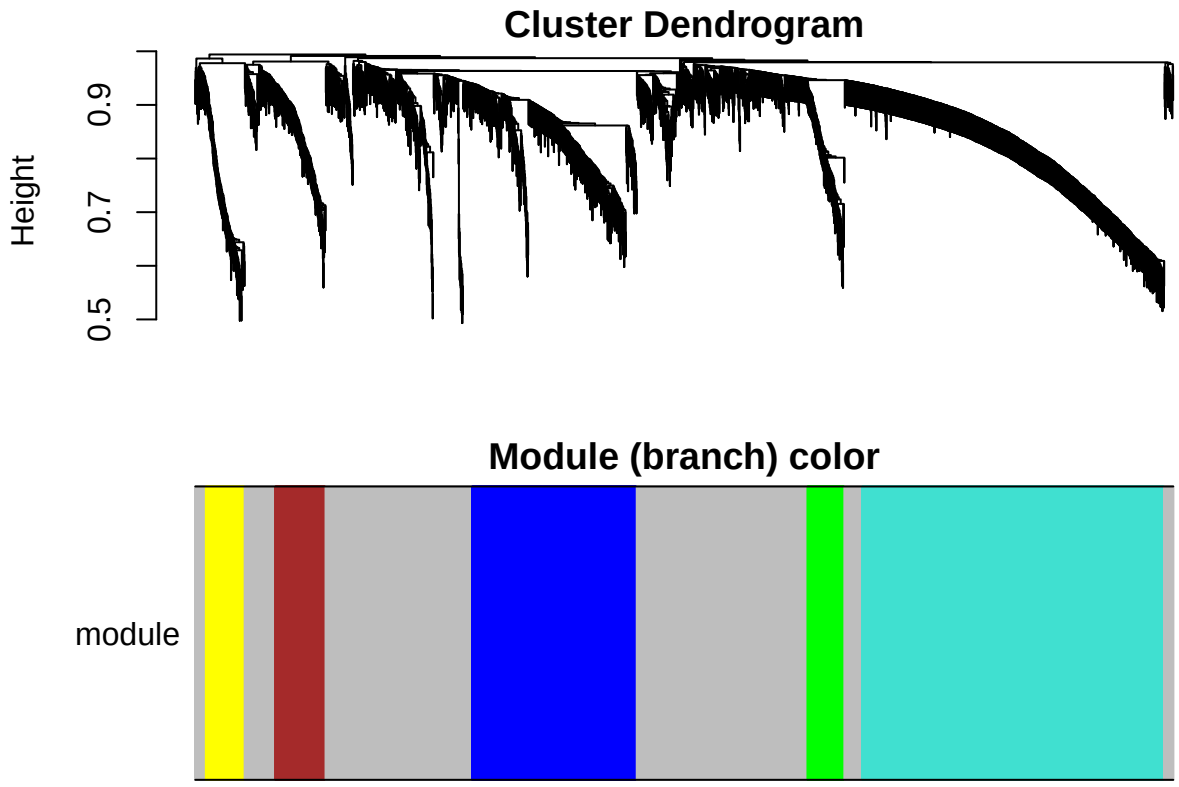
```
hierTOM = hclust(as.dist(dissTOM),method="average");
par(mfrow=c(1,1))
plot(hierTOM,labels=F)
```



According to our definition, modules correspond to branches of the tree. The question is what height cut-off should be used? This depends on the biology. Large heigth values lead to big modules, small values lead to small but tight modules. In reality, the user should use different thresholds to see how robust the findings are.

The function cutreeStatistColor colors each gene by the branches that result from choosing a particular height cut-off. GREY IS RESERVED to color genes that are not part of any module. We only consider modules that contain at least 125 genes.

```
colorh1= cutreeStaticColor(hierTOM,cutHeight = 0.94, minSize = 125)
# The above should be identical to colorh1=datSummary$color1[restConnectivity]
par(mfrow=c(2,1),mar=c(2,4,1,1))
plot(hierTOM, main="Cluster Dendrogram", labels=F, xlab="", sub="");
plotColorUnderTree(hierTOM,colors=data.frame(module=colorh1))
title("Module (branch) color")
```



COMMENTS:

1) The colors are assigned based on module size. Turquoise (others refer to it as cyan) colors the largest module, next comes blue, next brown, etc. Just type table(colorh1) to figure out which color corresponds to what module size.

2) The minimum module size (minsize1=125) is unusually large. As default, we recommend minsize1=50.

3) Here we choose a fixed height cut-off (h1) for cutting off branches. But we have also developed a more flexible method for cutting off branches that adaptively choose a different height for each branch. The resulting dynamic tree cutting algorithm (cutreeDynamic) is desccribed in Langfelder et al (2008).

An alternative view of this is the so called TOM plot that is generated by the function TOMplot

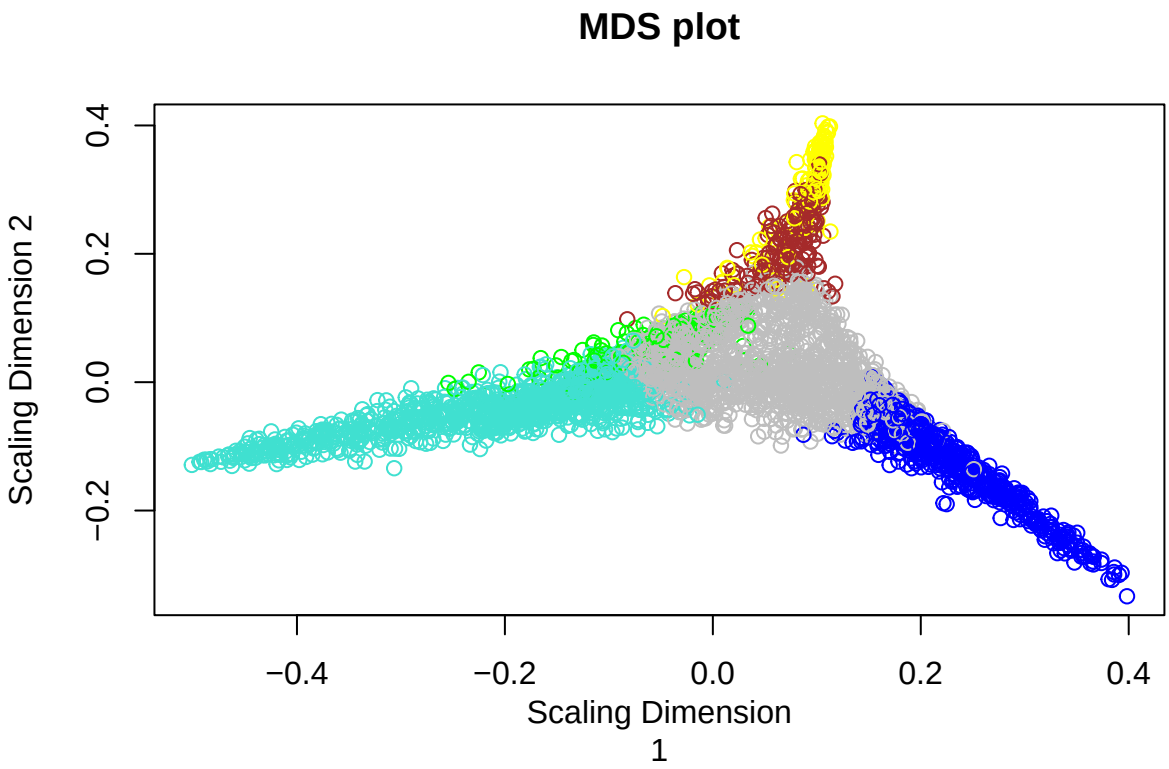
Inputs: TOM distance measure, hierarchical (hclust) object, color

Warning: for large gene sets, say more than 2000 genes this will take a while. I recommend you skip this.

```
TOMplot(dissTOM , hierTOM, colorh1)
```

We also propose to use classical multi-dimensional scaling plots for visualizing the network. Here we chose 2 scaling dimensions. This also takes about 10 minutes...

```
cmd1=cmdscale(as.dist(dissTOM),2)
par(mfrow=c(1,1))
plot(cmd1, col=as.character(colorh1), main="MDS plot",xlab="Scaling Dimension
1",ylab="Scaling Dimension 2")
```



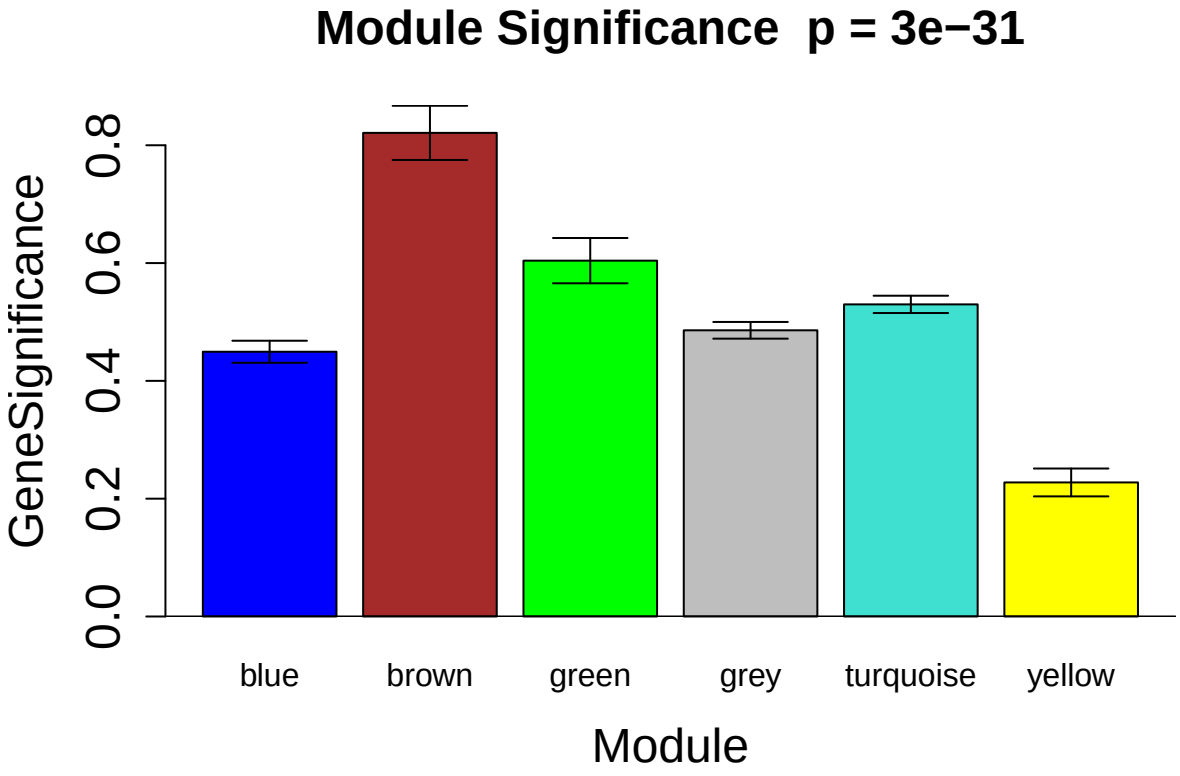
Module significance

Next we define a gene significance variable as minus log10 of the univarite Cox regression pvalue for predicting survival on the basis of the gene epxression info

```
# this defines the gene significance for all genes
GeneSignificanceALL=-log10(datSummary$pCox)
# gene significance restricted to the most connected genes:
GeneSignificance=GeneSignificanceALL[restConnectivity]
```

The function verboseBarplot creates a bar plot that shows whether modules are enriched with essential genes. It also reports a Kruskal Wallis P-value. The gene significance can be a binary variable or a quantitative variable. It also plots the 95% confidence interval of the mean

```
par(mfrow=c(1,1))
verboseBarplot(GeneSignificance,colorh1,main="Module Significance ",
col=levels(factor(colorh1)) ,xlab="Module" )
```



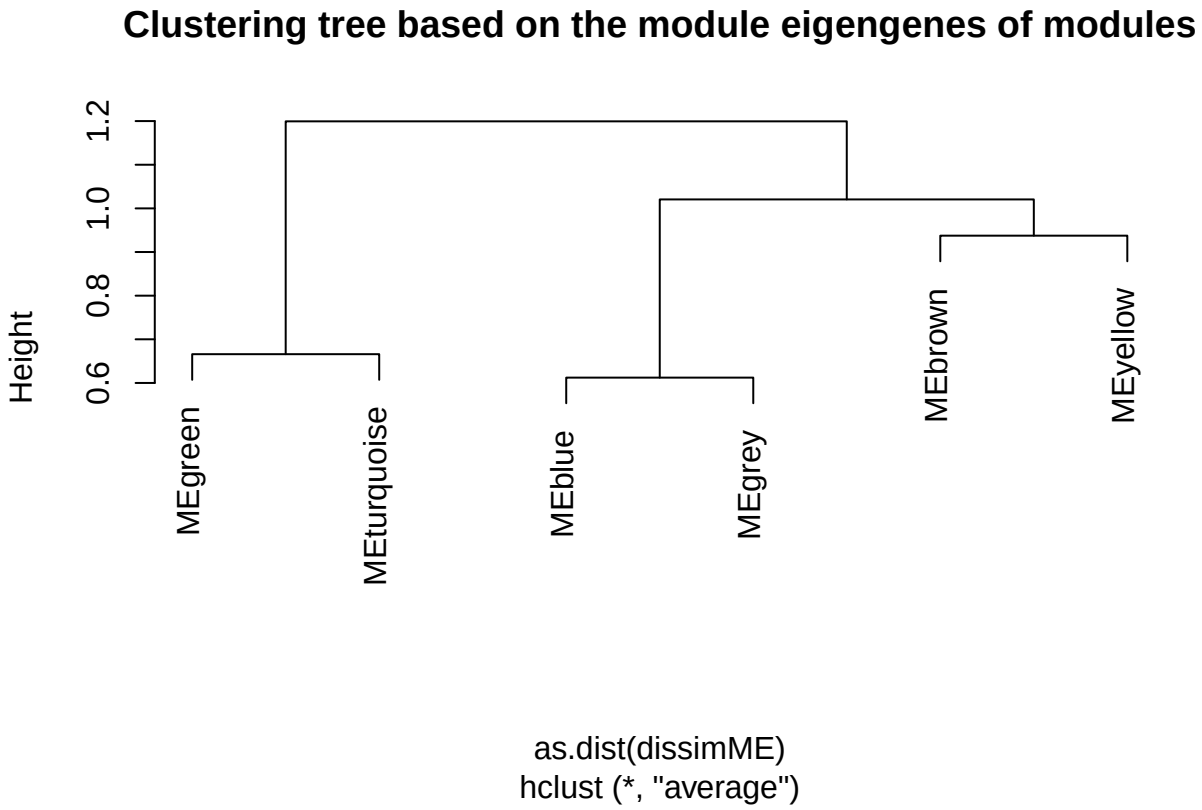
Note that the brown module have a high mean value of gene significance. As aside for the experts, we should mention that the p-value (Kruskal Wallis test) cannot be trusted due to dependence between the genes. The p-value should really be interpreted as a descriptive (not inferential) measure.

To get a sense of how related the modules are one can summarize each module by its first eigengene (referred to as principal components) and then correlate these module eigengenes with each other.

```
datME=moduleEigengenes(datExpr[,restConnectivity],colorh1)[[1]]
```

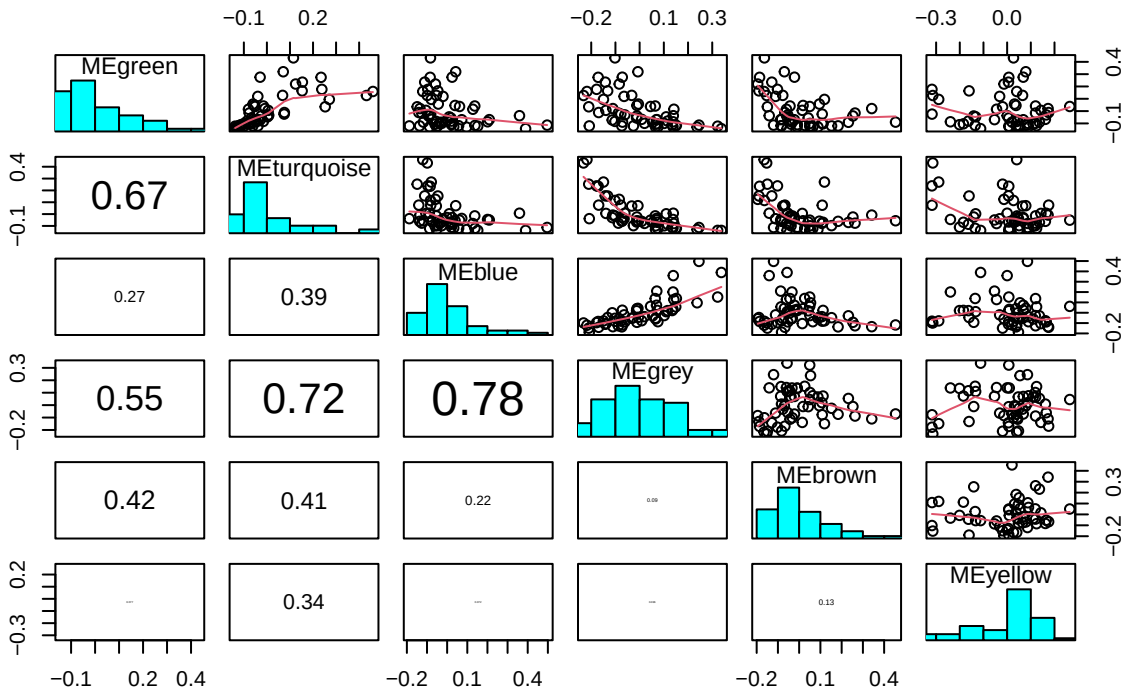
We define a dissimilarity measure between the module eigengenes that keeps track of the sign of the correlation between the module eigengenes.

```
dissimME=1-(t(cor(datME, method="p")))/2
hclustdatME=hclust(as.dist(dissimME), method="average" )
par(mfrow=c(1,1))
plot(hclustdatME, main="Clustering tree based on the module eigengenes of modules")
```



Now we create scatter plots of the samples (arrays) along the module eigengenes.

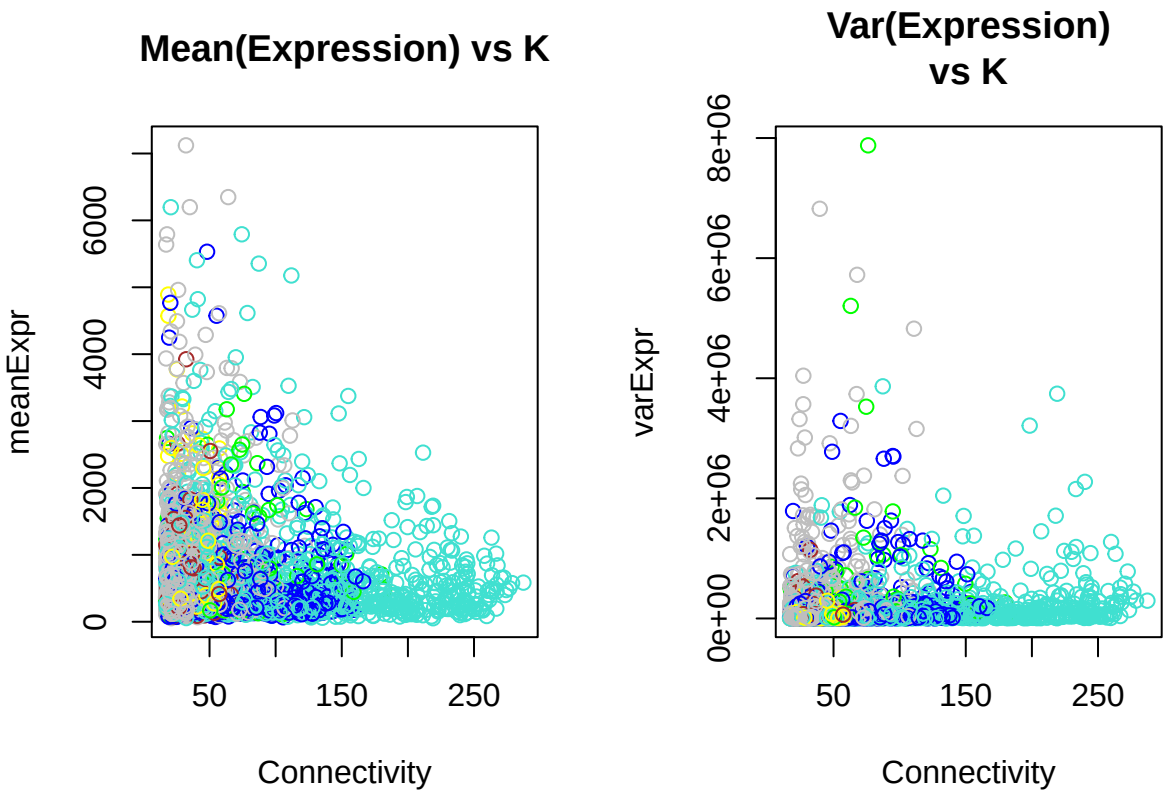
```
datMEordered=datME[,hclustdatME$order]
pairs( datMEordered, upper.panel = panel.smooth, lower.panel = panel.cor,
diag.panel=panel.hist ,main="Relation between module eigengenes")
```



Message: the module eigengenes (first PC) of different modules may be highly correlated. WGCNA can be interpreted as a biologically motivated data reduction scheme that allows for dependency between the resulting components. Compare this to principal component analysis that would impose orthogonality between the components. Since modules may represent biological pathways there is no biological reason why modules should be orthogonal to each other. Aside: If you are interested in networks comprised of module eigengenes, the following article may be of interest: Langfelder P, Horvath S (2007) Eigengene networks for studying the relationships between coexpression modules. BMC Systems Biology 2007, 1:54

To study how connectivity is related to mean gene expression or variance of gene expression we create the following plot.

```
mean1=function(x) mean(x,na.rm=T)
var1=function(x) var(x,na.rm=T)
meanExpr=apply( datExpr[,restConnectivity],2,mean1)
varExpr=apply( datExpr[,restConnectivity],2,var1)
par(mfrow=c(1,2))
plot(Connectivity[restConnectivity],meanExpr, col=as.character(colorh1),
main="Mean(Expression) vs K",xlab="Connectivity")
plot (Connectivity[restConnectivity],varExpr, col= as.character(colorh1), main="Var(Expression)
vs K" ,xlab="Connectivity")
```

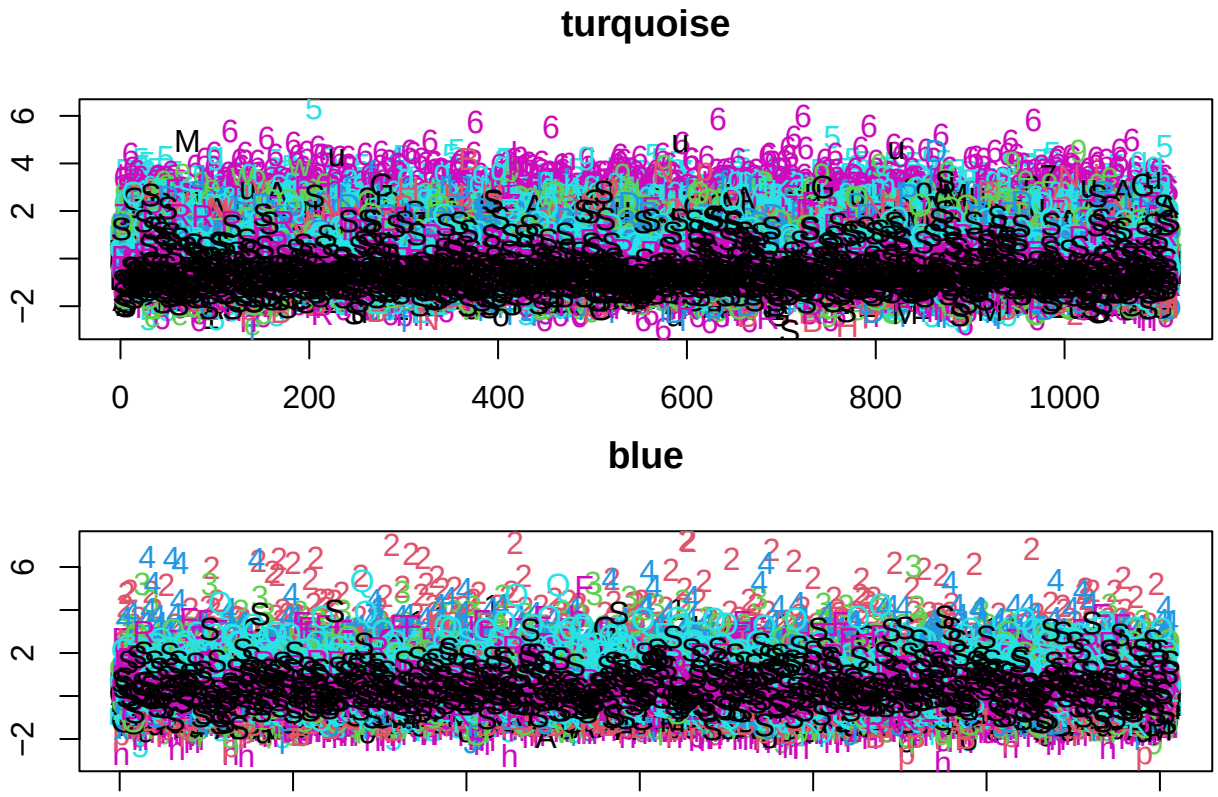


The following produces heatmap plots for each module.

Here the rows are genes and the columns are samples.

Well defined modules results in characteristic band structures since the corresponding genes are highly correlated.

```
par(mfrow=c(2,1), mar=c(1, 2, 4, 1))
ClusterSamples=hclust(dist(datExpr[,restConnectivity] ),method="average")
# for the first (turquoise) module we use
which.module="turquoise"
matplot(t(scale(datExpr[ClusterSamples$order,restConnectivity][,colorh1==which.module ])),nrgcols=30,rla
# for the second (blue) module we use
which.module="blue"
matplot(t(scale(datExpr[ClusterSamples$order,restConnectivity][,colorh1==which.module ])),nrgcols=30,rla
```

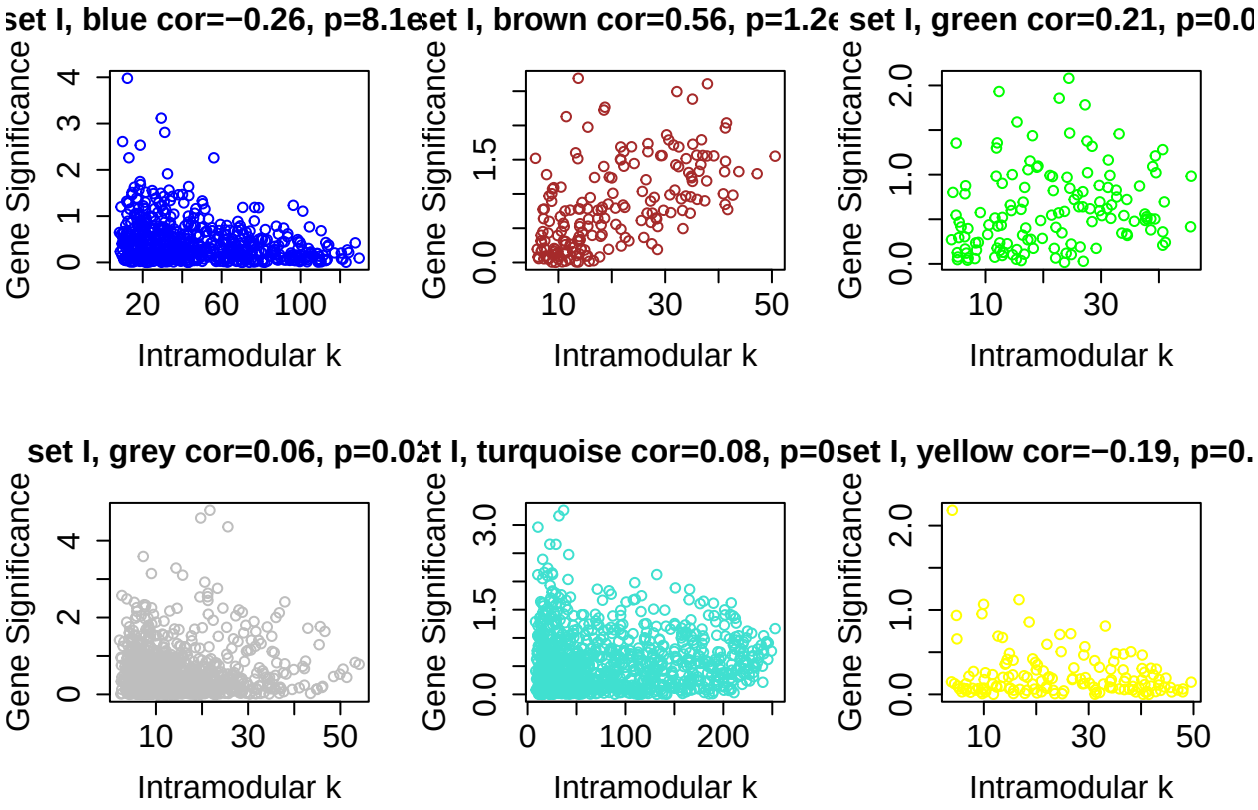


Now we extend the color definition to all genes by coloring all non-module genes grey.

```
color1=rep("grey",dim(datExpr)[[2]])
color1[restConnectivity]=as.character(colorh1)
# The function intramodularConnectivity computes the whole network connectivity kTotal,
# the within module connectivity (kWithin). kOut=kTotal-kWithin and
# and kDiff=kIn-kOut=2*kIN-kTotal
ConnectivityMeasures=intramodularConnectivity(ADJ,colors=colorh1)
names(ConnectivityMeasures)
```

```
## [1] "kTotal"  "kWithin" "kOut"    "kDiff"
```

```
# The following plots show the gene significance vs intramodular connectivity
colorlevels=levels(factor(colorh1))
par(mfrow=c(2,3),mar=c(5, 4, 4, 2) + 0.1)
for (i in c(1:length(colorlevels) ) ) {
whichmodule=colorlevels[[i]];restrict1=colorh1==whichmodule
verboseScatterplot(ConnectivityMeasures$kWithin[restrict1],
GeneSignificance[restrict1],col=colorh1[restrict1],main= paste("set I,",
whichmodule),ylab="Gene Significance",xlab="Intramodular k")
}
```



Generalizing the intramodular connectivity measure to *all* genes. Note that the intramodular connectivity measure is only defined for the genes inside a given module. But in practice it can be very important to measure how connected a given genes is to biologically interesting modules. Toward this end, we define a module eigengene based connectivity measure for each gene as the correlation between a the gene expression and the module eigengene. Specifically,

kMEbrown(i)=cor(x(i), PCbrown)

where x(i) is the gene expression profile of the i-th gene and PCbrown is the module eigengene of the brown module. Note that the definition does not require that the i-th gene is a member of the brown module.

The following data frame contains the kME values corresponding to each module.

```
datKME=signedKME(datExpr, datME)
#Note we have an intramodular connectivity measure for each color.
names(datKME)
```

```
## [1] "kMEblue"      "kMEbrown"     "kMEgreen"     "kMEgrey"      "kMEturquoise"
## [6] "kMEyellow"
```

Note that the intramodular connectivity has been computed for each of the 8000 genes.

```
dim(datKME)
```

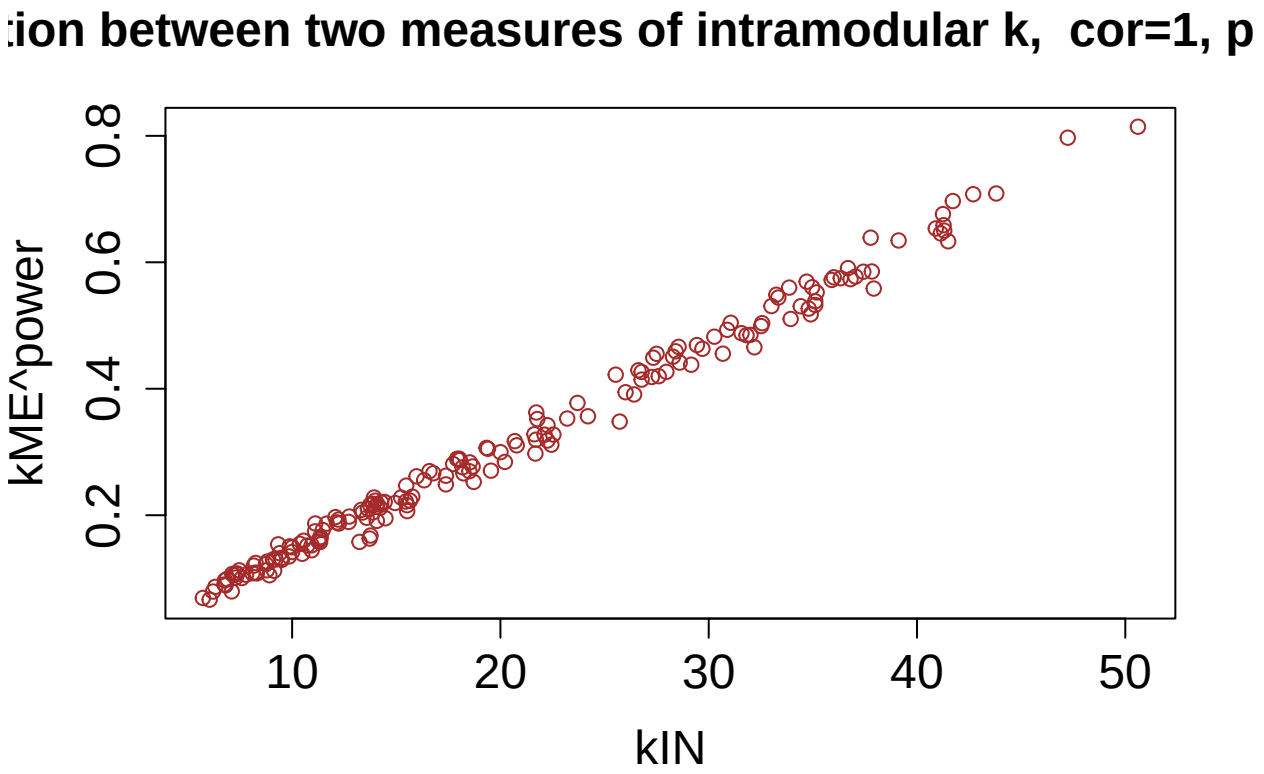
```
## [1] 8000    6
```

```
attach(datKME)
```

Question: how do the kME measure relate to the standard intramodular connectivity?

```
whichmodule="brown"
restrictGenes= colorh1== whichmodule
par(mfrow=c(1,1))
```

```
verboseScatterplot(ConnectivityMeasures$kWithin[ restrictGenes],
(datKME$kMEbrown[restConnectivity][restrictGenes])^beta1 ,xlab="kIN",ylab="kME^power",
col=whichmodule,main="Relation between two measures of intramodular k, ")
```



Note that after raising kME to a power of 6, it is highly correlated with kWithin. A theoretical derivation of this finding can be found in Horvath and Dong (2008).

Question: find genes with high gene significance (Cox-pvalue smaller than 0.05) and high intramodular connectivity in the brown module.

```
attach(datKME)
FilterGenes= GeneSignificanceALL > -log10(0.05) & abs(kMEbrown)>.85
table(FilterGenes)
```

```
## FilterGenes
## FALSE  TRUE
##  7970    30
```

```
datSummary[FilterGenes,]
```

```
##          gbm133a Gene.Symbol LocusLink    pCox HazardRatio HRlower95 HRupper95
## 731   219918_s_at        ASPM    259266 0.04740        1.95     1.100      3.46
## 805    209464_at       AURKB      9212 0.04870        2.21     1.220      4.03
## 1340 214710_s_at       CCNB1       891 0.02790        1.66     0.937      2.94
## 1402 202870_s_at       CDC20       991 0.01090        2.70     1.470      4.96
## 1404   204695_at      CDC25A       993 0.04000        2.09     1.140      3.82
## 1595 204170_s_at        CKS2      1164 0.03720        1.82     1.030      3.22
## 2297 219787_s_at        ECT2      1894 0.02060        2.30     1.260      4.22
## 2630 221591_s_at    FLJ10156     54478 0.01620        1.89     1.070      3.34
## 2671   213007_at    FLJ10719     55215 0.01370        1.92     1.070      3.43
## 2974 202580_x_at       FOXM1      2305 0.02740        1.79     1.020      3.14
```

```
## 3360 218662_s_at      HCAP-G     64151 0.04670        1.70      0.957       3.02
## 3475 208808_s_at       HMGB2      3148 0.02750        1.59      0.906       2.80
## 3921 202503_s_at    KIAA0101      9768 0.02760        1.83      1.030       3.25
## 4127   206364_at       KIF14      9928 0.03340        1.81      1.020       3.20
## 4132   218755_at      KIF20A     10112 0.03750        1.81      1.020       3.20
## 4138   218355_at       KIF4A     24137 0.04720        2.52      1.380       4.61
## 4158 218883_s_at       KLIP1     79682 0.02780        1.73      0.975       3.07
## 4164   219306_at       KNSL7     56992 0.01630        2.25      1.250       4.05
## 4166   204162_at       KNTC2     10403 0.02420        1.82      1.030       3.24
## 4168   201088_at       KPNA2      3838 0.02050        2.19      1.210       3.97
## 4481 203362_s_at      MAD2L1      4085 0.02710        1.57      0.895       2.75
## 5245   218039_at      NUSAP1     51203 0.01880        1.84      1.040       3.27
## 5826 218009_s_at        PRC1      9055 0.00247        2.41      1.320       4.41
## 6010 203554_x_at       PTTG1      9232 0.00413        1.97      1.110       3.49
## 6012   208511_at       PTTG2     10744 0.03050        1.60      0.901       2.84
## 6437   201890_at        RRM2      6241 0.01840        1.75      0.982       3.13
## 7450   201292_at       TOP2A      7153 0.02810        1.92      1.080       3.41
## 7453   219148_at        TOPK     55872 0.01930        2.00      1.130       3.54
## 7481 210052_s_at        TPX2     22974 0.00918        1.89      1.070       3.35
## 7624   202954_at       UBE2C     11065 0.00321        2.09      1.150       3.80
##       HRsd color1
## 731  0.292  brown
## 805  0.306  brown
## 1340 0.292  brown
## 1402 0.311  brown
## 1404 0.309  brown
## 1595 0.291  brown
## 2297 0.308  brown
## 2630 0.290  brown
## 2671 0.297  brown
## 2974 0.288  brown
## 3360 0.293  brown
## 3475 0.287  brown
## 3921 0.292  brown
## 4127 0.292  brown
## 4132 0.292  brown
## 4138 0.307  brown
## 4158 0.293  brown
## 4164 0.299  brown
## 4166 0.292  brown
## 4168 0.303  brown
## 4481 0.286  brown
## 5245 0.293  brown
## 5826 0.307  brown
## 6010 0.292  brown
## 6012 0.293  brown
## 6437 0.296  brown
## 7450 0.292  brown
## 7453 0.291  brown
## 7481 0.291  brown
## 7624 0.306  brown
```

Comments: The ASPM gene colored in red was the focus of the paper Horvath et al (2006) but there are many other interesting genes.

To illustrate the use of the kME measures, we also address the following questions

Question: Screen for significant genes that have a negative correlation with the brown module eigengene

```
FilterGenes= GeneSignificanceALL> -log10(0.05) & -kMEbrown> .5 # notice the red minus sign!
table(FilterGenes)
```

```
## FilterGenes
## FALSE  TRUE
##  7996     4
```

```
datSummary[FilterGenes,]
```

```
##          gbm133a Gene.Symbol LocusLink   pCox HazardRatio HRlower95 HRupper95
## 561  206200_s_at      ANXA11       311 0.0132        1.69     0.967      2.97
## 1618 221042_s_at        CLMN     79789 0.0257        1.60     0.906      2.81
## 5567   214152_at        PIGB      9488 0.0396        1.55     0.884      2.71
## 6682 201811_x_at      SH3BP5      9467 0.0162        2.09     1.180      3.69
##       HRsd    color1
## 561  0.286      grey
## 1618 0.288     green
## 5567 0.286 turquoise
## 6682 0.290 turquoise
```

Question: Screen for significant genes that are close to the brown module and the green module and far away from the yellow module. Answer

```
FilterGenes= GeneSignificanceALL > -log10(0.05) & abs(kMEbrown)>.5 & abs(kMEgreen)>.5
table(FilterGenes)
```

```
## FilterGenes
## FALSE  TRUE
##  7996     4
```

```
datSummary[FilterGenes,]
```

```
##          gbm133a Gene.Symbol LocusLink   pCox HazardRatio HRlower95 HRupper95
## 1422 218399_s_at       CDCA4     55038 0.0438        1.25     0.716      2.18
## 1618 221042_s_at        CLMN     79789 0.0257        1.60     0.906      2.81
## 3244   213170_at        GPX7      2882 0.0199        1.09     0.626      1.91
## 6682 201811_x_at      SH3BP5      9467 0.0162        2.09     1.180      3.69
##       HRsd    color1
## 1422 0.284 turquoise
## 1618 0.288     green
## 3244 0.285      grey
## 6682 0.290 turquoise
```

Question: Screen for significant genes that are close to the brown module and far away from the yellow module. Answer

```
FilterGenes= GeneSignificanceALL > -log10(0.05) & abs(kMEbrown)>.6 & abs(kMEyellow)<.3
table(FilterGenes)
```

```
## FilterGenes
## FALSE  TRUE
##  7949    51
```

How to output the data?

```
datout=data.frame(datSummary, colorNEW=color1, ConnectivityNew=Connectivity,datKME )
write.table(datout, file="OutputCancerNetwork.csv", sep=",", row.names=F)
```

Robustness with regard to the soft threshold

We find that the results of weighted gene co-expression network analysis are highly robust with regard to the soft threshold beta. Here we show some results that demonstrate this point. Now we want to see how the correlation between kWithin and gene significance changes for different SOFT thresholds (powers). This analysis is restricted to the brown module genes.

Also we compare the 2 different connectivity measures: The standard connectivity measure is defined as the row sum of the adjacency matrix. The non-standard connectivity measure (kTOM.IN) is defined as row sum of the topological overlap matrix .

Now we want to see how the correlation between kWithin and gene significance changes for different powers beta within the BROWN module.

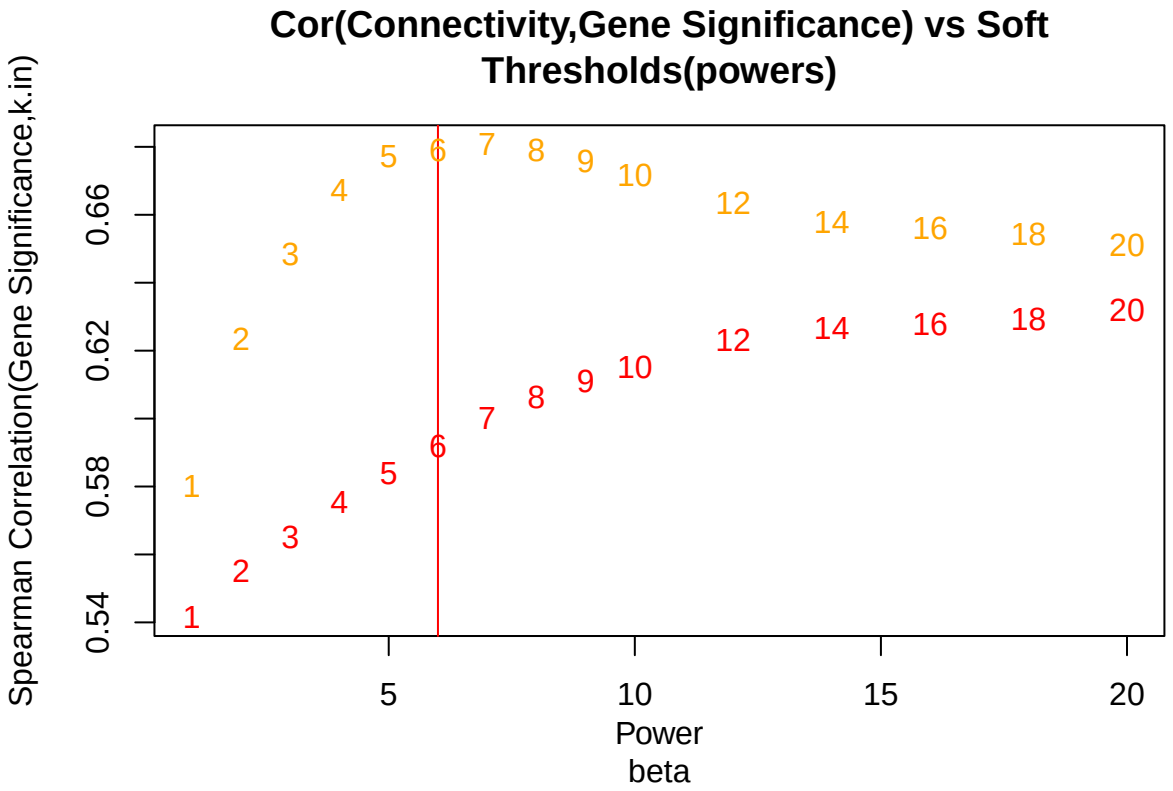
```
corhelp=cor(datExpr[,restConnectivity],use="pairwise.complete.obs")
whichmodule="brown"
datconnectivitiesSoft=data.frame(matrix(666,nrow=sum(colorh1==whichmodule),ncol=length(powers1)))
names(datconnectivitiesSoft)=paste("kWithinPower",powers1,sep="")
for (i in c(1:length(powers1)) ) {
datconnectivitiesSoft[,i]=apply(abs(corhelp[colorh1==whichmodule,
colorh1==whichmodule])^powers1[i],1,sum)}
SpearmanCorrelationsSoft=signif(cor(GeneSignificance[ colorh1==whichmodule],
datconnectivitiesSoft, method="s",use="p"))
# Here we use the new connectivity measure based on the topological overlap matrix
datKTOM.IN=data.frame(matrix(666,nrow=sum(colorh1==whichmodule),ncol=length(powers1)))
names(datKTOM.IN)=paste("omegaWithinPower",powers1,sep="")
for (i in c(1:length(powers1)) ) {
datconnectivitiesSoft[,i]=apply(
1-TOMdist(abs(corhelp[colorh1==whichmodule, colorh1==whichmodule])^powers1[i])
,1,sum)}
```

```
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
```

```
## ..done.
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
## ..connectivity..
## ..matrix multiplication (system BLAS)..
## ..normalization..
## ..done.
```

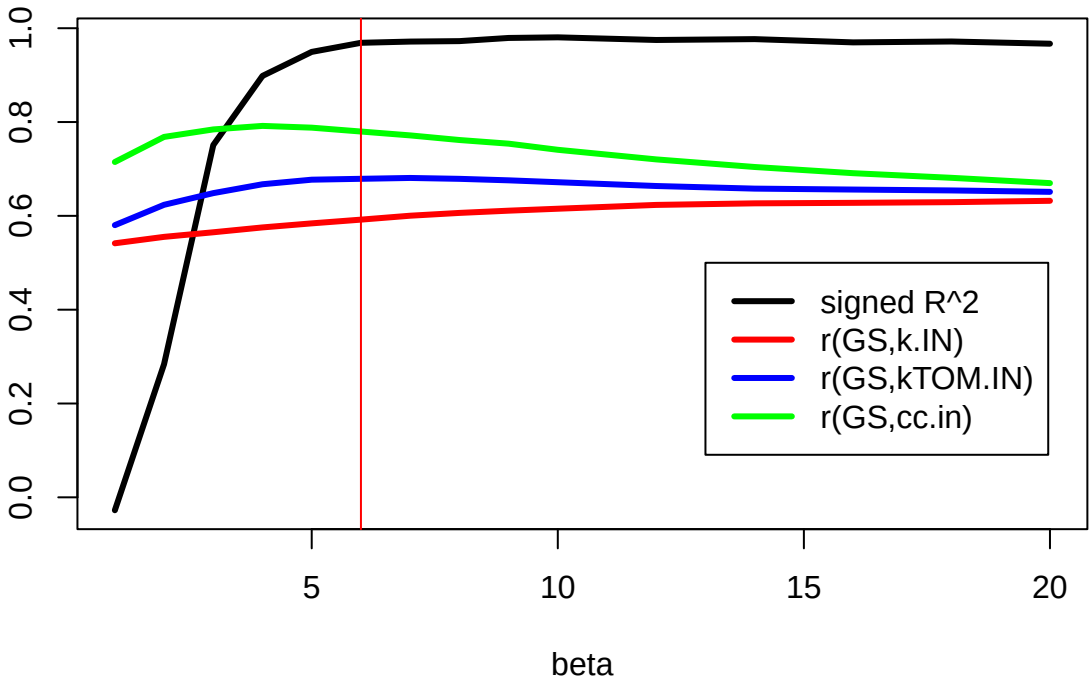
```
SpearmanCorrelationskTOMSoft=as.vector(signif(cor(GeneSignificance[ colorh1==whichmodule],
datconnectivitiesSoft, method="s",use="p")))
par(mfrow=c(1,1), mar=c(5, 4, 4, 2) +0.1)
plot(powers1, SpearmanCorrelationsSoft, main="Cor(Connectivity,Gene Significance) vs Soft
Thresholds(powers)",ylab="Spearman Correlation(Gene Significance,k.in)",xlab="Power
beta",type="n",ylim=range(c(SpearmanCorrelationsSoft,
SpearmanCorrelationskTOMSoft),na.rm=T)
)
text(powers1, SpearmanCorrelationsSoft,labels=powers1,col="red")
# this draws a vertical line at the tau that was chosen by the
# scale free topology criterion.
abline(v=6,col="red")
```

```
points(powers1, SpearmanCorrelationskTOMSoft, type="n")
text(powers1, SpearmanCorrelationskTOMSoft,labels=powers1,col="orange")
```



Now we define the intramodular clustering coefficient (also see the section below)

```
datCCinSoft=data.frame(matrix(666,nrow=sum(colorh1==whichmodule),ncol=length(powers1)))
names(datCCinSoft)=paste("CCinSoft",powers1,sep="")
for (i in c(1:length(powers1)) ) {
datCCinSoft[,i]= clusterCoef(abs(corhelp[colorh1==whichmodule,
colorh1==whichmodule])^powers1[i])
}
SpearmanCorrelationsCCinSoft=as.vector(signif(cor(GeneSignificance[ colorh1==whichmodule],
datCCinSoft, method="s",use="p")))
dathelpSoft=data.frame(signedRsquared=-sign(RpowerTable[,3])*RpowerTable[,2], corGSkINSoft
=as.vector(SpearmanCorrelationsSoft), corGSwINSoft=
as.vector(SpearmanCorrelationskTOMSoft),corGSCCSoft=as.vector(SpearmanCorrelationsCCinSoft))
matplot(powers1,dathelpSoft,type="l",lty=1,lwd=3,col=c("black","red","blue","green"),ylab="",xlab="beta
abline(v=6,col="red")
legend(13,0.5, c("signed R^2","r(GS,k.IN)","r(GS,kTOM.IN)","r(GS,cc.in)"),
col=c("black","red","blue","green"), lty=1,lwd=3,ncol = 1, cex=1)
```



Comment:

the intramodular cluster coefficient (green line) achieves the highest correlation with the gene significance. The TOM based intramodular connectivity kTOM.in (blue line) is superior to the standard connectivity measure k.in (red line) in this application. The vertical line corresponds to the power picked by the scale free topology criterion. The scale free topology criterion leads to near optimal biological signal when using kTOM.IN.

CAVEAT: It is worth mentioning that in other real data sets k.in outperforms cc.in and kTOM.IN.

Computation of the cluster coefficient in the weighted network.

The clustering coefficient measures the cliquishness of a gene. Many references use this concept. For our definition of the clustering coefficient in weighted networks consult Zhang and Horvath (2005) and Dong and Horvath (2007).

Here we study how the clustering coefficient depends on the connectivity.

Since this is computationally intensive (around 15 minutes), we recommend to skip it.

```
CC= clusterCoef(ADJ)
gc()
```

Now we plot cluster coefficient versus connectivity for all genes

```
par(mfrow=c(1,1),mar=c(2,2,2,1))
plot(Connectivity[restConnectivity],CC,col=as.character(colorh1),xlab="Connectivity",ylab="Clust
er Coefficient")
```

This compute the correlation between cluster coefficient and connectivity within each module.

```
restHub= Connectivity[restConnectivity]>0
by(data.frame(CC=CC[restHub], k=Connectivity[restConnectivity][restHub]),
INDICES=colorh1[restHub],FUN=cor)
```