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

Steve Horvath, Bin Zhang, Jun Dong, Tova Fuller, Peter Langfelder
Correspondence: shorvath@mednet.ucla.edu, http://www.ph.ucla.edu/biostat/people/horvath.htm

This R tutorial describes how to carry out a gene co-expression network analysis with the R software.

Content of this tutorial

1) **1) Gene Co-expression Network Construction. We show how to construct unweighted networks using hard thresholding and how to construct weighted networks using soft thresholding. We describe a criterion (scale free topology criterion) for choosing the threshold.**
   2) Module Definition Based on Average Linkage hierarchical clustering with a tree cutting algorithm
   3) Relating modules to prognostic significance for cancer survival time
   Keywords: gene significance, module significance
   4) Relating gene significance to intramodular connectivity.
   5) Generalizing intramodular connectivity to all genes on the array.
   Keyword: module eigengene based connectivity measure
   6) Comparing weighted network results to unweighted network results
   7) Studying the relationship between the clustering coefficicient and intramodular connectivity.

To cite this tutorial or the statistical methods please use the following 2 references

1. *Bin Zhang and Steve Horvath (2005) "A General Framework for Weighted Gene Co-Expression Network Analysis", Statistical Applications in Genetics and Molecular Biology: Vol. 4: No. 1, Article 17* Technical Report and software code at: www.genetics.ucla.edu/labs/horvath/CoexpressionNetwork.
2. *Horvath S, Zhang B, Carlson M, Lu KV, Zhu S, Felciano RM, Laurance MF, Zhao W, Shu, Q, Lee Y, Scheck AC, Liau LM, Wu H, Geschwind DH, Febbo PG, Kornblum HI, Cloughesy TF, Nelson SF, Mischel PS (2006) "Analysis of Oncogenic Signaling Networks in Glioblastoma Identifies ASPM as a Novel Molecular Target", PNAS | November 14, 2006 | vol. 103 | no. 46 | 17402-17407*

The following theoretical reference explores the meaning of coexpression network analysis

- *Horvath S, Dong J (2008) Geometric Interpretation of Gene Co-Expression Network Analysis. PloS Computational Biology. 4(8): e1000117. PMID: 18704157*

The WGCNA R package is described in

- *Langfelder P, Horvath S (2008) WGCNA: an R package for Weighted Correlation Network Analysis. BMC Bioinformatics. 2008 Dec 29;9(1):559. PMID: 19114008*

For the generalized topological overlap matrix as applied to unweighted networks see

- *Yip A, Horvath S (2007) Gene network interconnectedness and the generalized topological overlap measure. BMC Bioinformatics 8:22*

Microarray Data
The data and biological implications are described in Horvath et al 2006
The data were provided by Stan Nelson, who directs the UCLA Microarray core.
Expression of 22,215 probe sets (15,005 unique transcripts) was measured using Affymetrix HG-U133A microarrays, as previously described(16). Data files (cel) were uploaded into the dCHIP program (http://www.dchip.org/) and normalized to the median intensity array. Complete gene expression for datasets 1 and 2 available at:
http://genetics.ucla.edu/labs/horvath/binzhang/Public/Networks/GBM_all_datasets.zip; (dataset 1 = gbm55old_dchipALL_cox.xls; dataset 2 = bm65new_dchipAll_cox.xls. Quantification was performed using model-based expression and the perfect match minus mismatch method implemented in dCHIP.
More detailed descriptions and more detailed tutorials can be found at the following webpage:
http://www.genetics.ucla.edu/labs/horvath/CoexpressionNetwork/ASPMgene/

Methods Outline
The network construction is conceptually straightforward: nodes represent genes and nodes are connected if the corresponding genes are significantly co-expressed across appropriately chosen tissue samples. Here we study networks that can be specified with the following adjacency matrix: $A=[a_{ij}]$ is symmetric with entries in [0,1]. By convention, the diagonal elements are assumed to be zero. For unweighted networks, the adjacency matrix contains binary information (connected=1, unconnected=0). In weighted networks the adjacency matrix contains weights.

The absolute value of the Pearson correlation between expression profiles of all pairs of genes was determined for the 8000 most varying non-redundant transcripts. Then, pioneering the use of a novel approach to the generation of a weighted gene coexpression networks, the Pearson correlation measure was transformed into a connection strength measure by using a power function (connection strength(i,j)=|correlation(i,j)|^β)(Zhang and Horvath 2005). The connectivity measure for each gene is the sum of the connection strengths (correlationβ) between that gene and all the other genes in the network. Gene expression networks, like virtually all types of biological networks, exhibit an approximate scale free topology. A linear regression model fitting index R2 between log p(k) and log(k) was used to determine how well a resulting network fit scale free topology for a range of values. The scale free topplogy criterion (Zhang and Horvath 2005) was used to determine the power,: specifically, a value of =6 was the lowest power that resulted in a scale free topology fit R^2 that was >0.9 (Supplementary Fig. 1).
We will discuss the choice of this power in great detail below and provide several arguments that this choice of power results in a biologically meaningful network.
Most biologists would be very suspicious of a gene co-expression network that does not satisfy scale-free topology at least approximately. Therefore, thresholds (or powers) that give rise to networks that do not satisfy approximate scale-free topology should not be considered.

Detection of hub genes:
To identify hub genes for the network, one may either consider the whole network connectivity (denoted by kTotal) or the intramodular connectivity (kWithin).
We find that intramodular connectivity is far more meaningful than whole network connectivity

Relating hub gene status to gene (prognostic) significance:
For each of the 55 glioblastoma samples patient survival information was available. Since some of the survival times were censored, we used a Cox proportional hazards model(21, 22) to define the prognostic significance of a gene by its univariate Cox regression p-value. More specifically we define the gene significance of each gene as minus log10 of the univariate Cox regression p-value. Thus high values of the gene significance imply that the gene expression is a significant predictor of patient survival, see Mischel et al 2005.

Abstractly speaking, gene significance is any quantitative measure that specifies how biologically significant a gene is. One goal of network analysis is to relate the measure of gene significance (here –log10[Cox p-value]) to intramodular connectivity.

Unweighted networks, hard thresholding
Based on the expression data, the absolute pair-wise (Pearson) correlation coefficient between the expression profiles of each pair of genes is calculated. Then, a network with each node representing one gene is constructed. An edge between two nodes is present if their absolute correlation coefficient exceeds a threshold. We obtain the threshold tau by using the scale-free criterion.

Module Construction
To group genes with coherent expression profiles into modules, we use average linkage hierarchical clustering, which uses the topological overlap measure as dissimilarity.
The topological overlap of two nodes reflects their similarity in terms of the commonality of the nodes they connect to, see [Ravasz et al 2002, Yip and Horvath 2005].
Once a dendrogram is obtained from a hierarchical clustering method, we need to choose a height cutoff in order to arrive at a clustering. It is a judgement call where to cut the tree branches.
The height cut-off can be found by inspection: a height cutoff value is chosen in the dendrogram such that some of the resulting branches correspond to the discrete diagonal blocks (modules) in the TOM plot.

Absolutely no warranty on the code. Please contact SH with suggestions.

CONTENTS
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:
For example to install Hmisc, open R,
go to menu "Packages\Install package(s) from CRAN",
then choose Hmisc. R will automatically install the package.
When asked "Delete downloaded files (y/N)? ", answer "y".
Do the same for some of the other libraries mentioned below. But note that
several libraries are already present in the software so there is no need to re-install them.
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"

Unzip all the files into the same directory.
Set the working directory of the R session by using the following command.
Note that we use / instead of \ in the path.
setwd("C:/Documents and Settings/Steve Horvath/My
Documents/ADAG/LinSong/NetworkScreening/GBM/GeneralFramework")

#Please copy and paste the following script into the R session.
#Text after "#" is a comment and is automatically ignored by 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

 # Download the WGCNA library as a .zip file from
http://www.genetics.ucla.edu/labs/horvath/CoexpressionNetwork/Rpackages/WGCNA/
and choose "Install package(s) from local zip file" in the packages tab

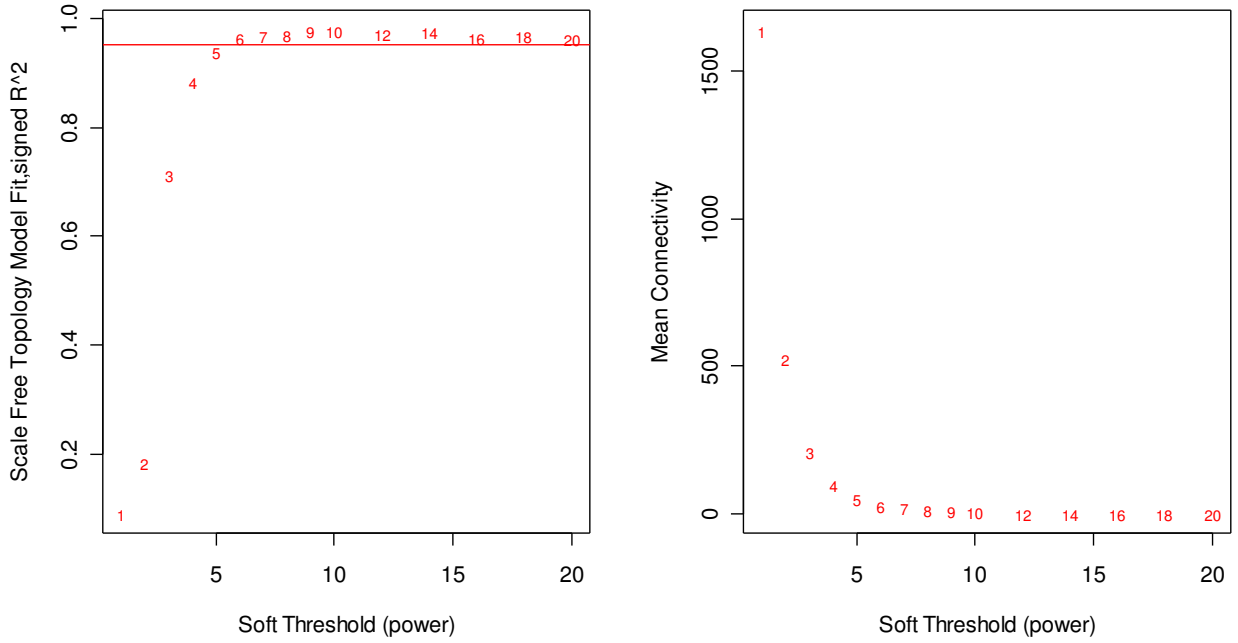
```
library(WGCNA)
options(stringsAsFactors=F)

#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)
rm(dat0);gc()



# 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	scale.law.R.2	slope	truncated.R.2	mean.k.	median.k.	max.k.
1	1	−0.0927	0.463	0.979	1640.00	1.60e+03	2700.0
2	2	0.1880	−0.844	0.942	527.00	4.79e+02	1310.0
3	3	0.7150	−1.410	0.967	214.00	1.74e+02	769.0
4	4	0.8840	−1.650	0.974	102.00	7.22e+01	513.0
5	5	0.9390	−1.710	0.977	54.90	3.25e+01	373.0
6	6	0.9650	−1.660	0.983	32.50	1.58e+01	288.0
7	7	0.9680	−1.610	0.980	20.80	8.09e+00	232.0
8	8	0.9690	−1.550	0.977	14.10	4.36e+00	193.0
9	9	0.9770	−1.490	0.983	10.10	2.43e+00	166.0
10	10	0.9780	−1.460	0.984	7.48	1.42e+00	145.0
11	12	0.9720	−1.400	0.981	4.52	5.22e−01	114.0
12	14	0.9740	−1.360	0.982	2.97	2.08e−01	92.7
13	16	0.9660	−1.340	0.971	2.08	8.94e−02	76.9
14	18	0.9680	−1.330	0.980	1.52	4.00e−02	65.3
15	20	0.9610	−1.320	0.972	1.14	1.87e−02	56.2

```
gc()
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")
```

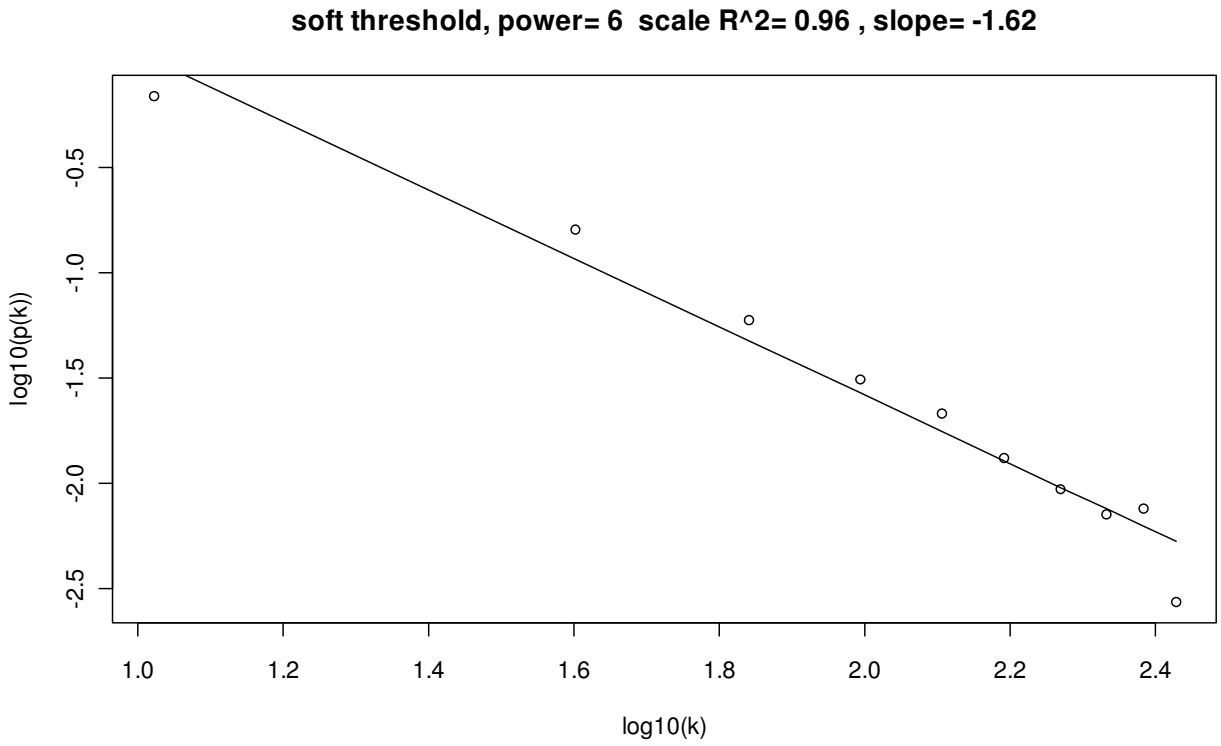



```
# Note that at power=6, the curve has an elbow or kink, i.e. for this power the scale free topology
# fit does not improve after increasing the power. This is why we choose beta1=6
# Also the scale free topology criterion with a R^2 threshold of 0.95 would lead us to pick a power
# of 6.
```

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.

```
# We use the following power for the power adjacency function.
beta1=6
Connectivity=softConnectivity(datExpr,power=beta1)-1
# 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);
```



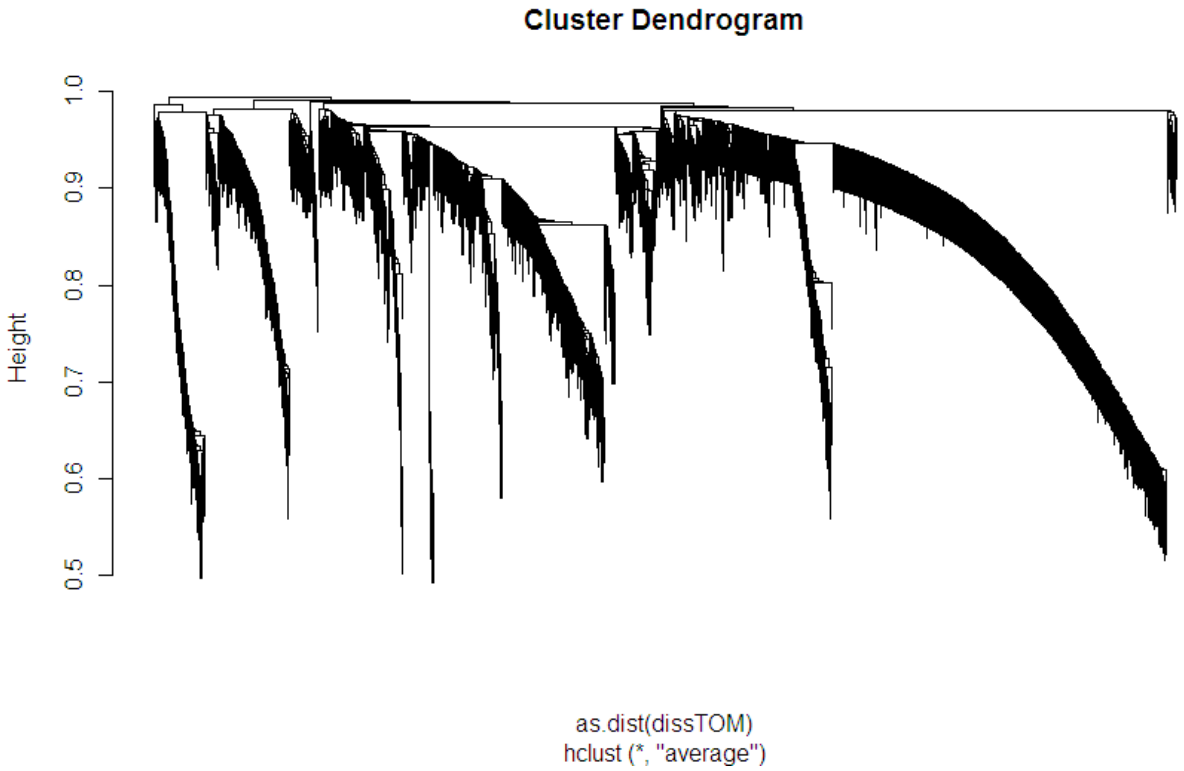
#Module Detection

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)

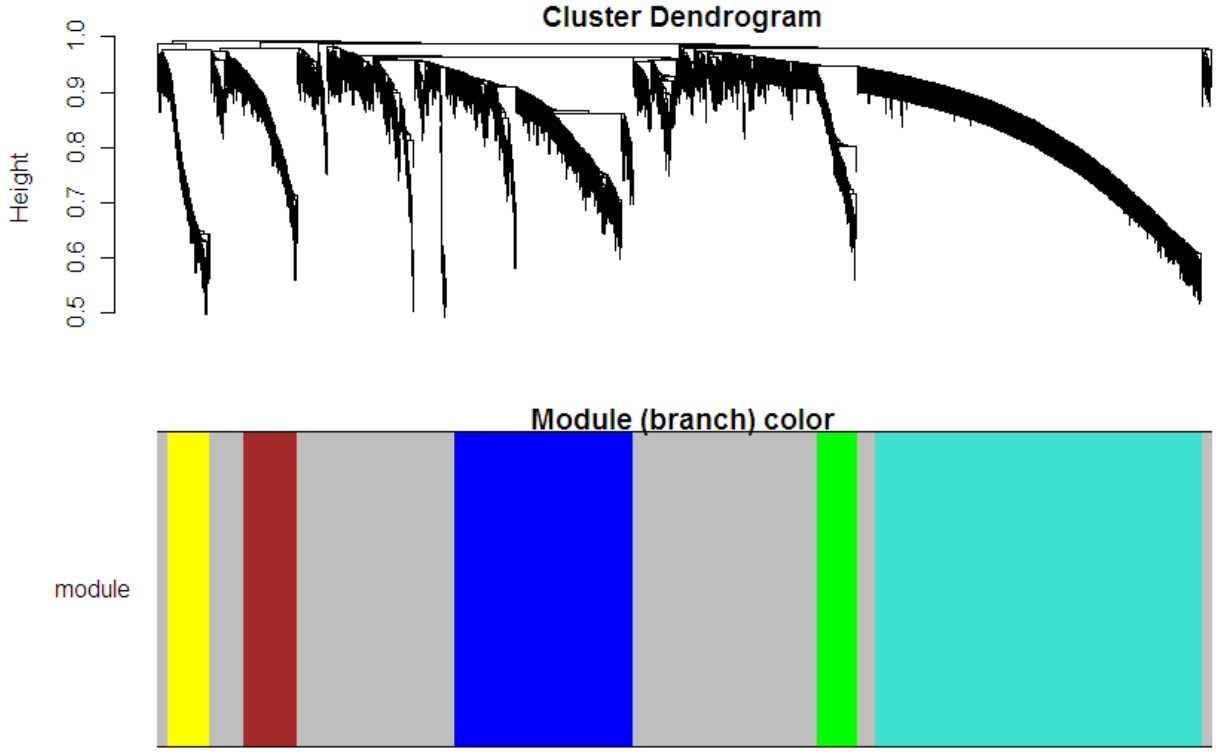
# Now we define the adjacency matrix for the 3600 most connected genes
ADJ= adjacency(datExpr[,restConnectivity],power=beta1)
gc()
# The following code computes the topological overlap matrix based on the
# adjacency matrix.
# TIME: This about a few minutes....
dissTOM=TOMdist(ADJ)
gc()

# 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")
```

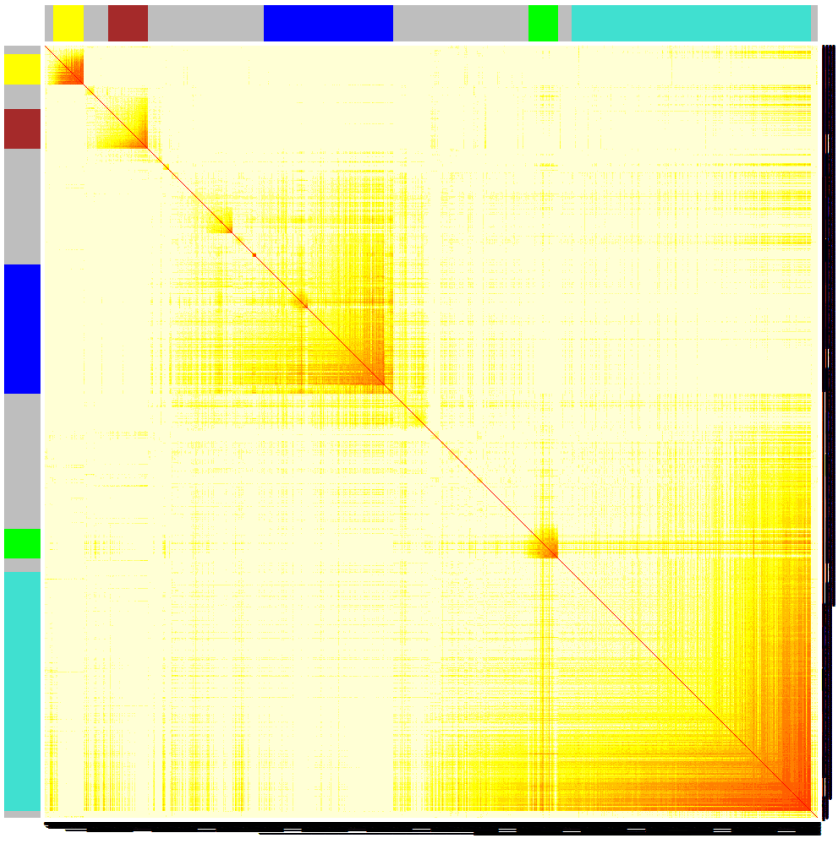


COMMENTS:

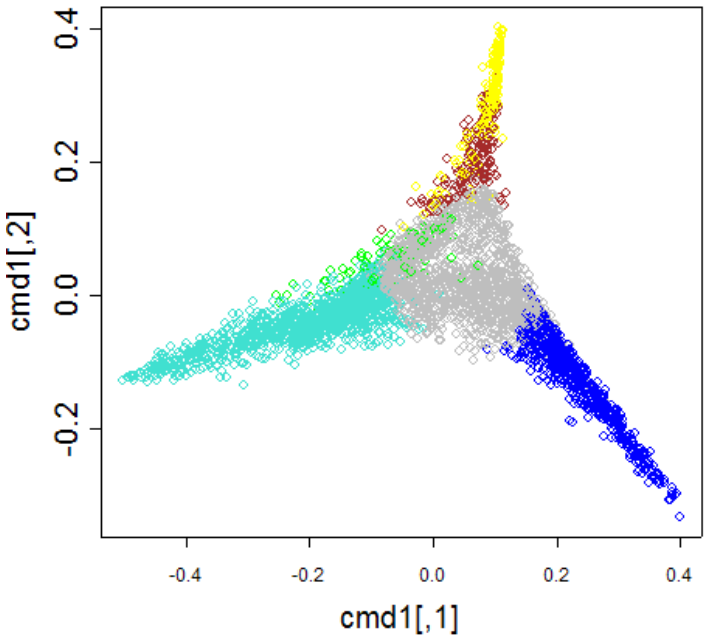
1) The colors are assigned based on module size. Turquoise (others refer to it as cyan) colors the largest module, next comes blue, next brown, etc. Just type table(colorh1) to figure out which color corresponds to what module size.
2) The minimum module size (minsize1=125) is unusually large. As default, 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)

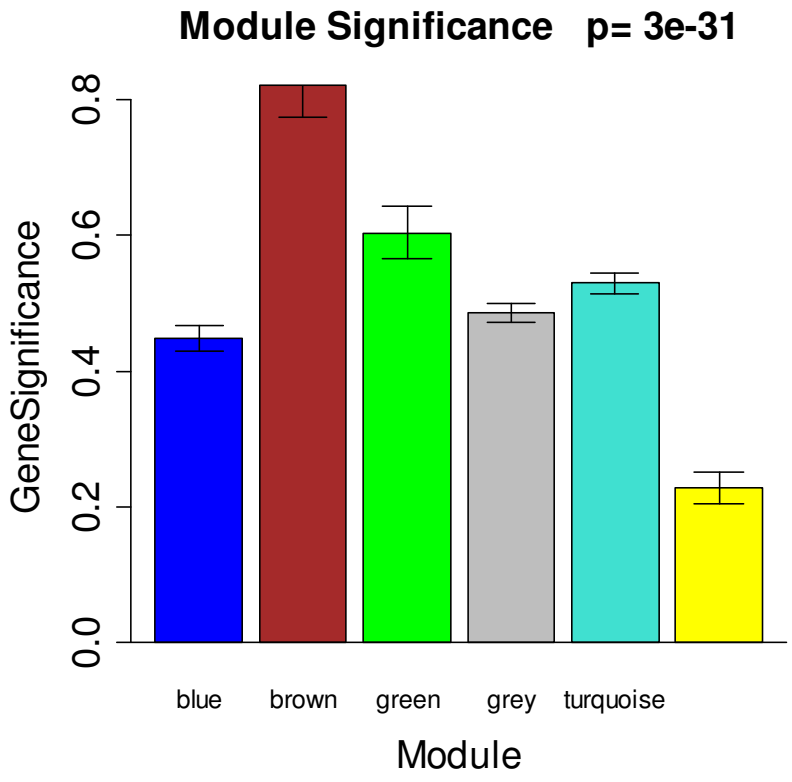


```
# We also propose to use classical multi-dimensional scaling plots
# for visualizing the network. Here we chose 2 scaling dimensions
# This also takes about 10 minutes...
cmd1=cmdscale(as.dist(dissTOM),2)
par(mfrow=c(1,1))
plot(cmd1, col=as.character(colorh1),  main="MDS plot",xlab="Scaling Dimension
1",ylab="Scaling Dimension 2")
```



Module significance
#Next we define a gene significance variable as minus log10 of the univarite Cox regression p-value 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.
# 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" )
```



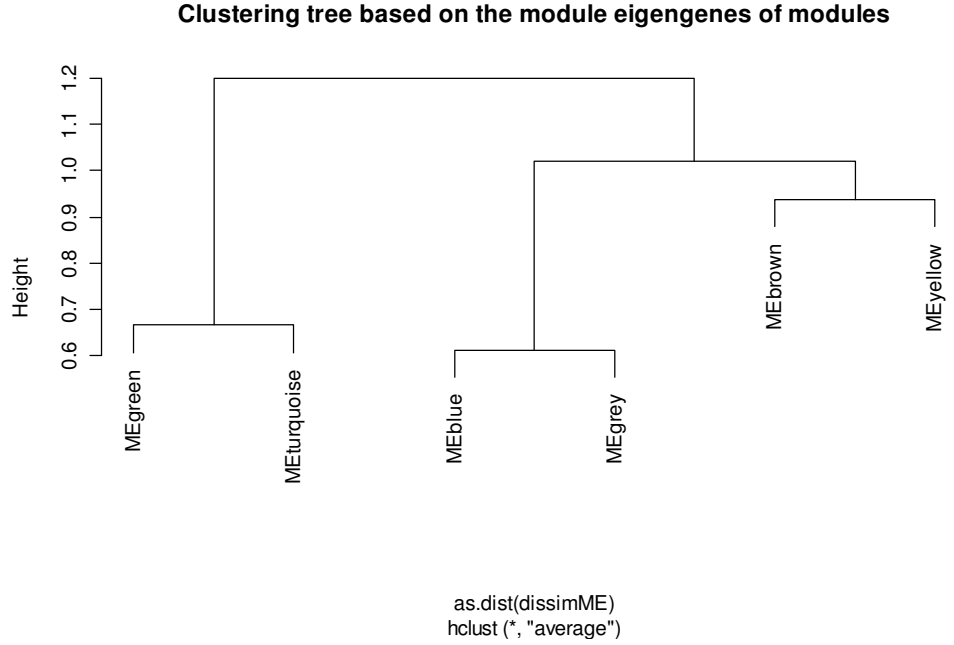
Note that the brown module have a high mean value of gene significance.
As aside for the experts, we should mention that the p-value (Kruskal Wallis test) cannot be trusted due to dependence between the genes. The p-value should really be interpreted as a descriptive (not inferential) measure.

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]]

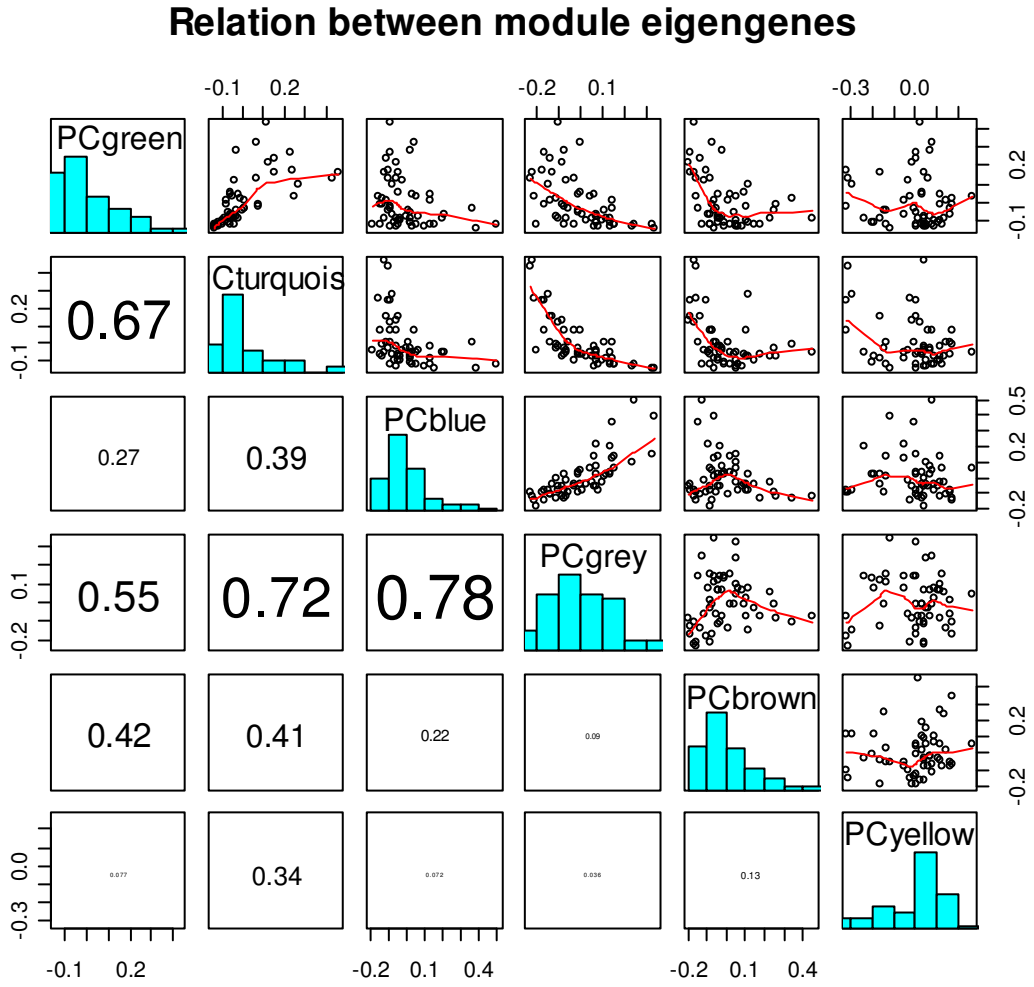
# We define a dissimilarity measure between the module eigengenes that keeps track of the sign of the correlation between the module eigengenes.
dissimME=1-(t(cor(datME, method="p")))/2

hclustdatME=hclust(as.dist(dissimME), method="average")
par(mfrow=c(1,1))
plot(hclustdatME, main="Clustering tree based on the module eigengenes of modules")



Compare this output with the following:
signif(cor(datME, use="p"), 2)

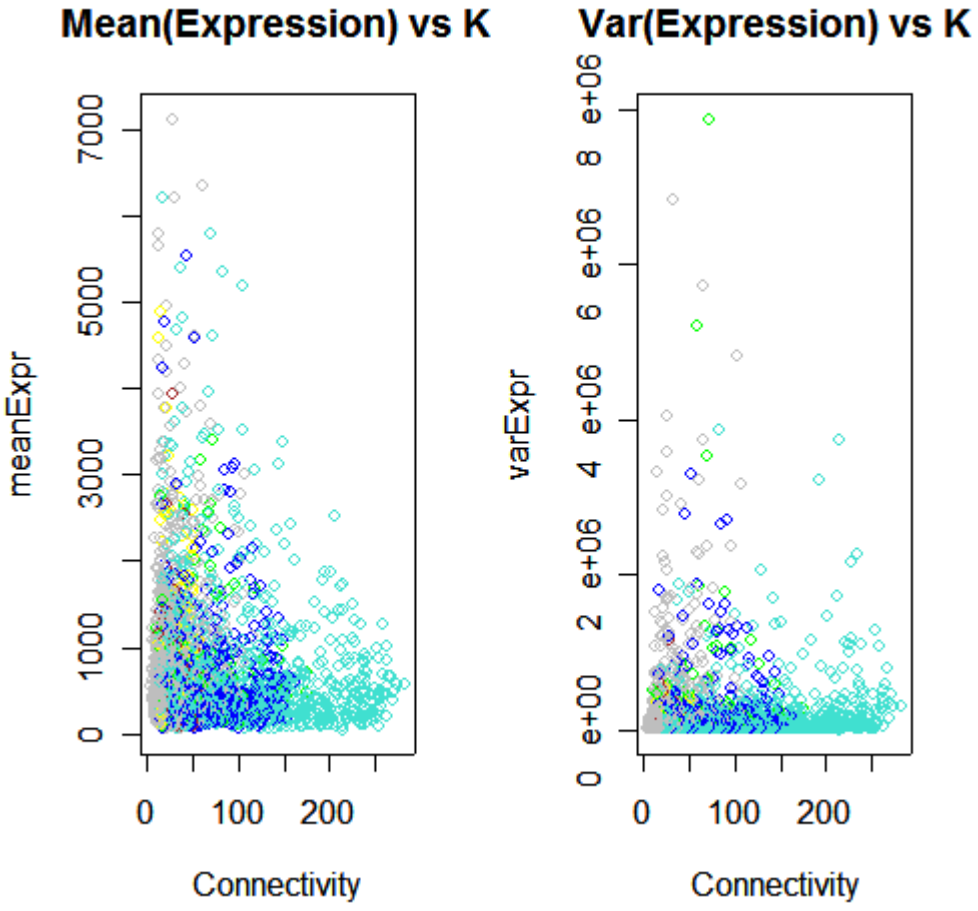
# Now we create scatter plots of the samples (arrays) along the module eigengenes.
datMEordered=datME[,hclustdatME$order]
pairs(datMEordered, upper.panel = panel.smooth,   lower.panel = panel.cor ,
diag.panel=panel.hist ,main="Relation between module eigengenes")



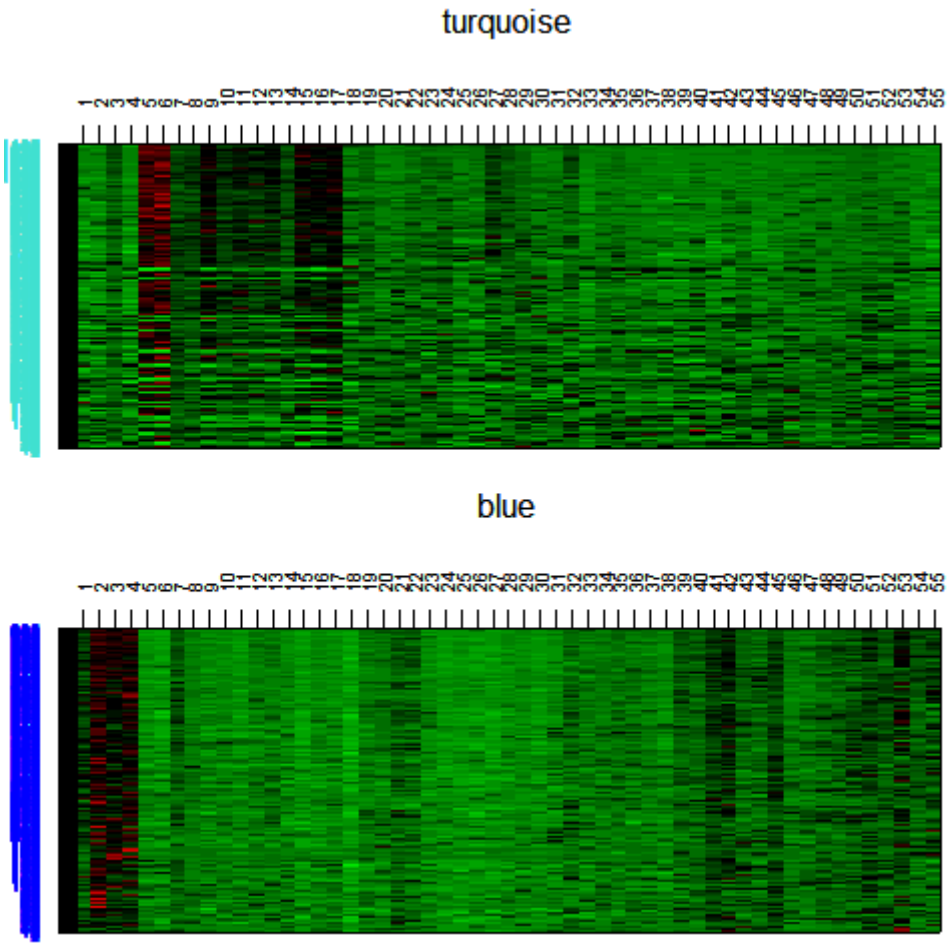
Message: the module eigengenes (first PC) of different modules may be highly correlated. WGCNA can be interpreted as a biologically motivated data reduction scheme that allows for dependency between the resulting components. Compare this to principal component analysis that would impose orthogonality 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 co-expression 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"
plot.mat(t(scale(datExpr[ClusterSamples$order,restConnectivity][,colorh1==which.module ])
),nrgcols=30,rlabels=T, clabels=T,rcols=which.module,
main=which.module )
# for the second (blue) module we use
which.module="blue"
plot.mat(t(scale(datExpr[ClusterSamples$order,restConnectivity][,colorh1==which.module ])
),nrgcols=30,rlabels=T, clabels=T,rcols=which.module,
main=which.module )
```

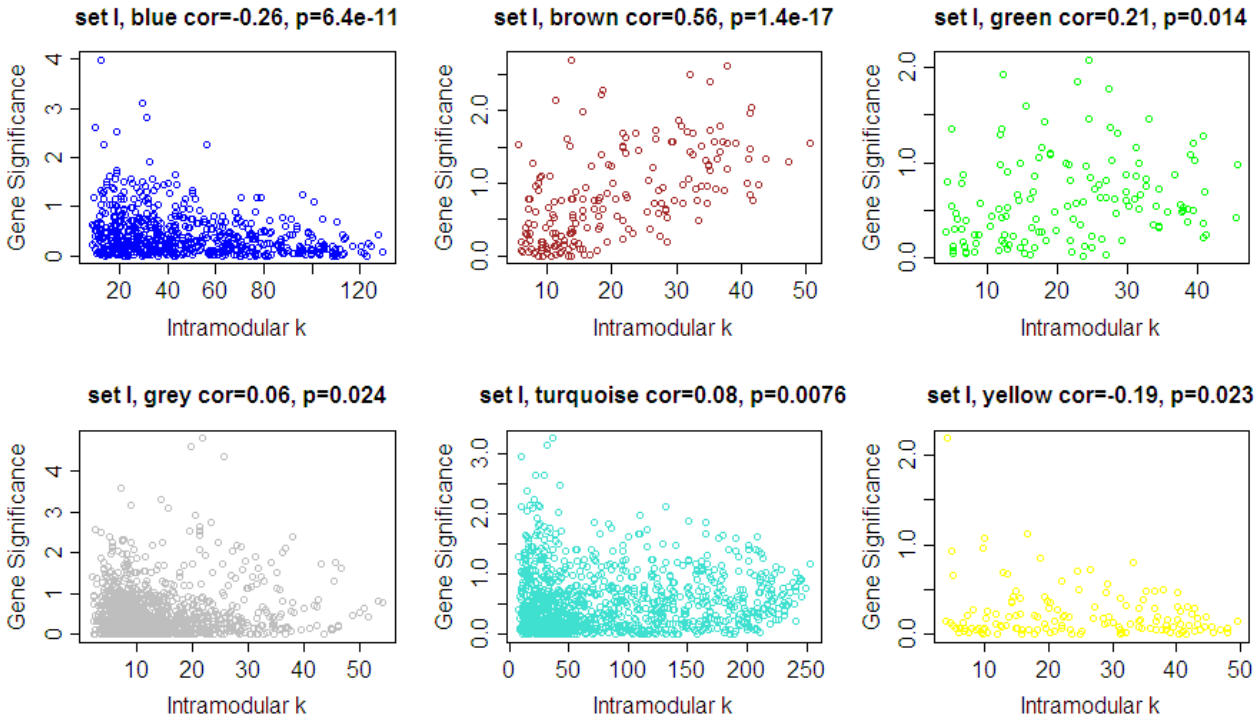


# Now we extend the color definition to all genes by coloring all non-module
# genes grey.
color1=rep("grey",dim(datExpr)[[2]])
color1[restConnectivity]=as.character(colorh1)

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"
```

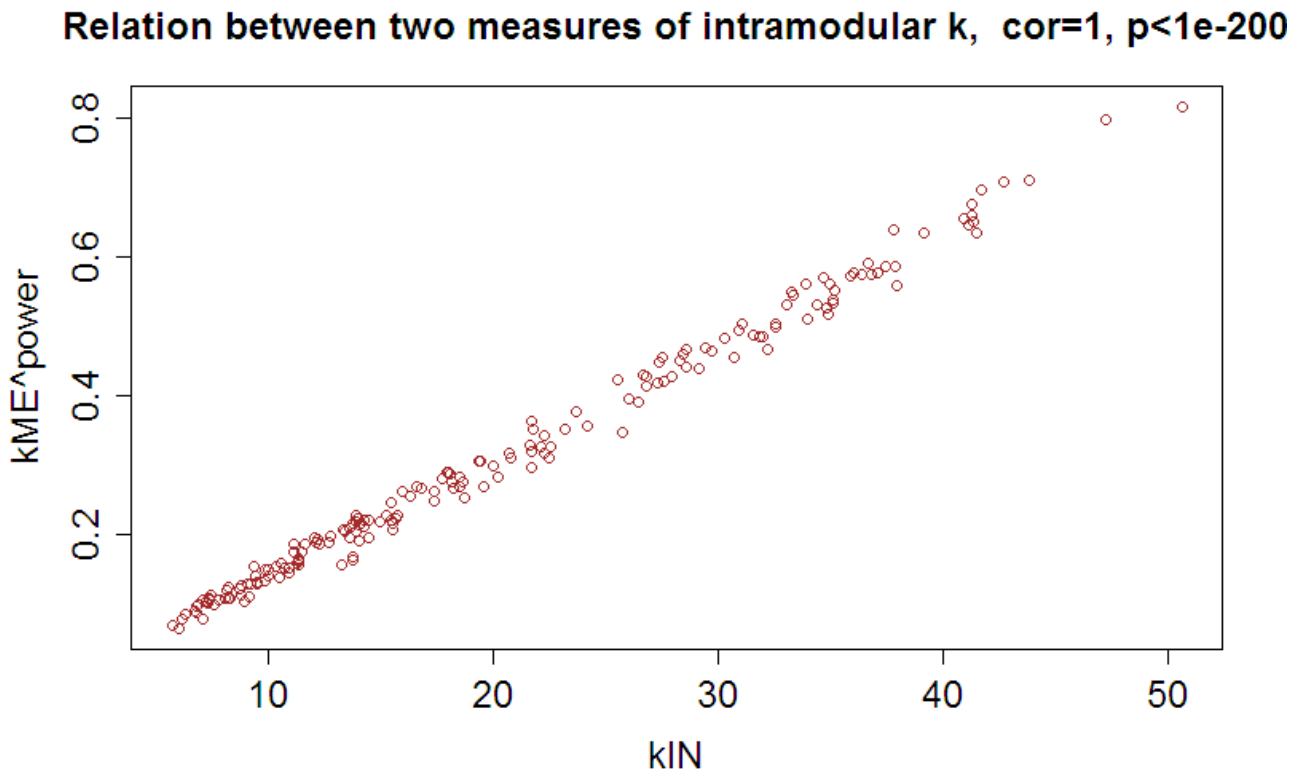
Note that the intramodular connectivity has been computed for each of the 8000 genes.

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)

datSummary[FilterGenes,]

	gbm133a	Gene.Symbol	LocusLink	pCox	HazardRatio	HRlower95	HRupper95	HRsd	color1
731	219918_s_at	ASPM	259266	0.04740	1.95	1.100	3.46	0.292	brown
805	209464_at	AURKB	9212	0.04870	2.21	1.220	4.03	0.306	brown
1340	214710_s_at	CCNB1	891	0.02790	1.66	0.937	2.94	0.292	brown
1402	202870_s_at	CDC20	991	0.01090	2.70	1.470	4.96	0.311	brown
1404	204695_at	CDC25A	993	0.04000	2.09	1.140	3.82	0.309	brown
1595	204170_s_at	CKS2	1164	0.03720	1.82	1.030	3.22	0.291	brown
2297	219787_s_at	ECT2	1894	0.02060	2.30	1.260	4.22	0.308	brown
2630	221591_s_at	FLJ10156	54478	0.01620	1.89	1.070	3.34	0.290	brown
2671	213007_at	FLJ10719	55215	0.01370	1.92	1.070	3.43	0.297	brown
2974	202580_x_at	FOXM1	2305	0.02740	1.79	1.020	3.14	0.288	brown
3360	218662_s_at	HCAP-G	64151	0.04670	1.70	0.957	3.02	0.293	brown
3475	208808_s_at	HMGB2	3148	0.02750	1.59	0.906	2.80	0.287	brown
3921	202503_s_at	KIAA0101	9768	0.02760	1.83	1.030	3.25	0.292	brown
4127	206364_at	KIF14	9928	0.03340	1.81	1.020	3.20	0.292	brown
4132	218755_at	KIF20A	10112	0.03750	1.81	1.020	3.20	0.292	brown
4138	218355_at	KIF4A	24137	0.04720	2.52	1.380	4.61	0.307	brown
4158	218883_s_at	KLIP1	79682	0.02780	1.73	0.975	3.07	0.293	brown
4164	219306_at	KNSL7	56992	0.01630	2.25	1.250	4.05	0.299	brown
4166	204162_at	KNTC2	10403	0.02420	1.82	1.030	3.24	0.292	brown
4168	201088_at	KPNA2	3838	0.02050	2.19	1.210	3.97	0.303	brown
4481	203362_s_at	MAD2L1	4085	0.02710	1.57	0.895	2.75	0.286	brown
5245	218039_at	NUSAP1	51203	0.01880	1.84	1.040	3.27	0.293	brown
5826	218009_s_at	PRC1	9055	0.00247	2.41	1.320	4.41	0.307	brown
6010	203554_x_at	PTTG1	9232	0.00413	1.97	1.110	3.49	0.292	brown
6012	208511_at	PTTG2	10744	0.03050	1.60	0.901	2.84	0.293	brown
6437	201890_at	RRM2	6241	0.01840	1.75	0.982	3.13	0.296	brown
7450	201292_at	TOP2A	7153	0.02810	1.92	1.080	3.41	0.292	brown
7453	219148_at	TOPK	55872	0.01930	2.00	1.130	3.54	0.291	brown
7481	210052_s_at	TPX2	22974	0.00918	1.89	1.070	3.35	0.291	brown
7624	202954_at	UBE2C	11065	0.00321	2.09	1.150	3.80	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)
datSummary[FilterGenes,]

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

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

How to output the data?

datout=data.frame(datSummary, colorNEW=color1, ConnectivityNew=Connectivity,datKME)
write.table(datout, file="OutputCancerNetwork.csv", sep=",", row.names=F)
# The file can now be found in your current working directory (see the setwd command above).
This file should be sent to your clinical collaborators so that they can study the resulting gene list.

Next logical step: carry out a functional enrichment analysis of each module.

For example, input all genes with abs(kMEbrown)>0.8 into

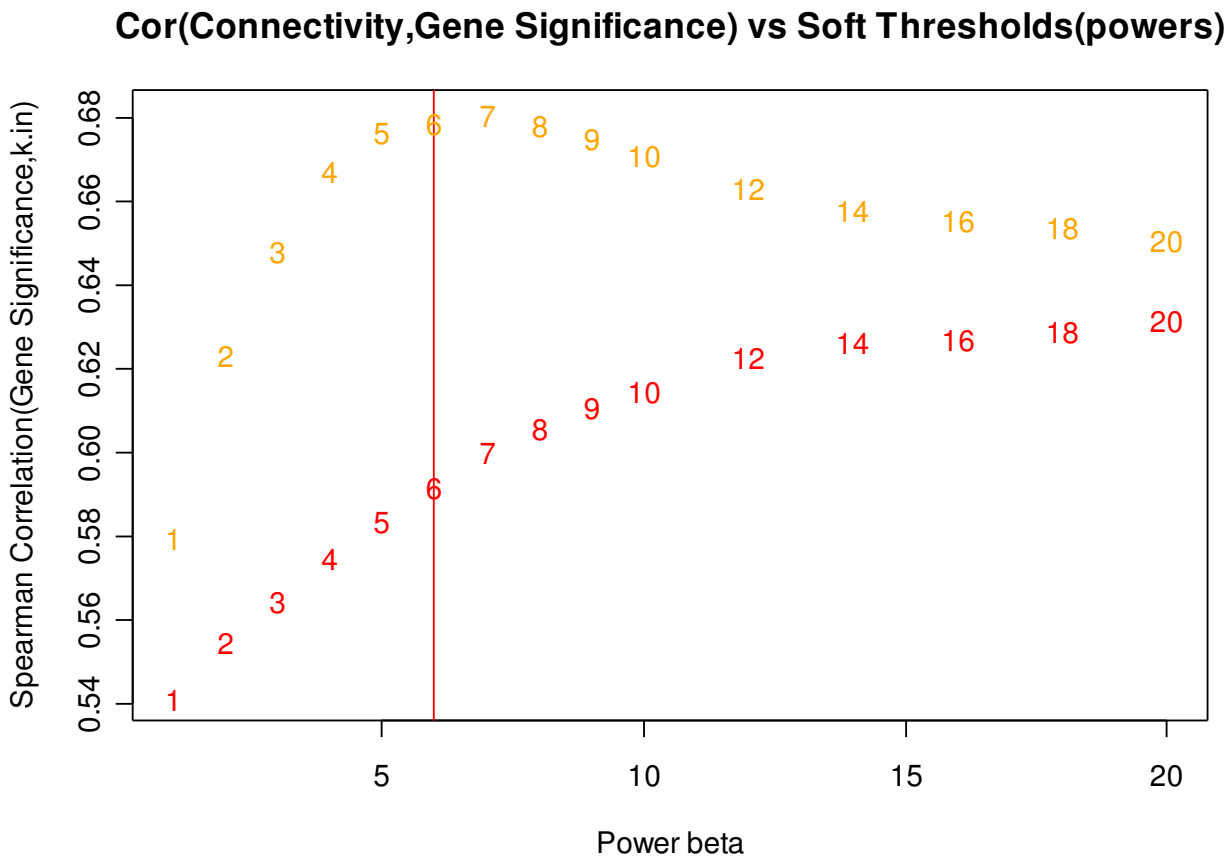
EASE (David)
http://david.abcc.ncifcrf.gov/summary.jsp

The WGCNA library also contains several functions for gene ontology information (e.g. the function GOenrichmentAnalysis) as you can learn from help(WGCNA)

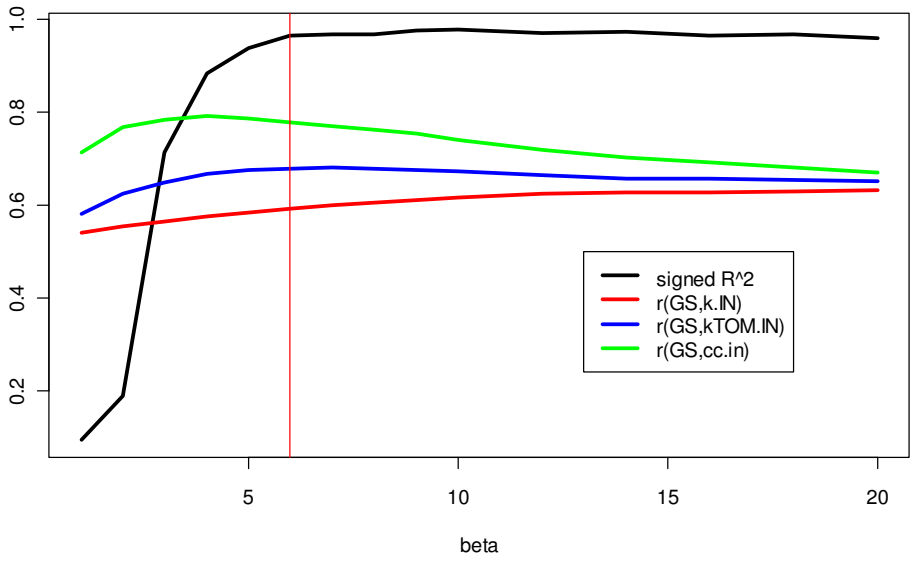
Robustness with regard to the soft threshold (power= beta).
We find that the results of weighted gene co-expression network analysis are highly robust with regard to the soft threshold beta. Here we show some results that demonstrate this point.
Now we want to see how the correlation between kWithin 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(pow
ers1)))
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)}
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(SpearmanCorrelationsCCinSo
ft))
matplot(powers1,dathelpSoft,type="l",lty=1,lwd=3,col=c("black","red","blue","green"),ylab="",xla
b="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)
```

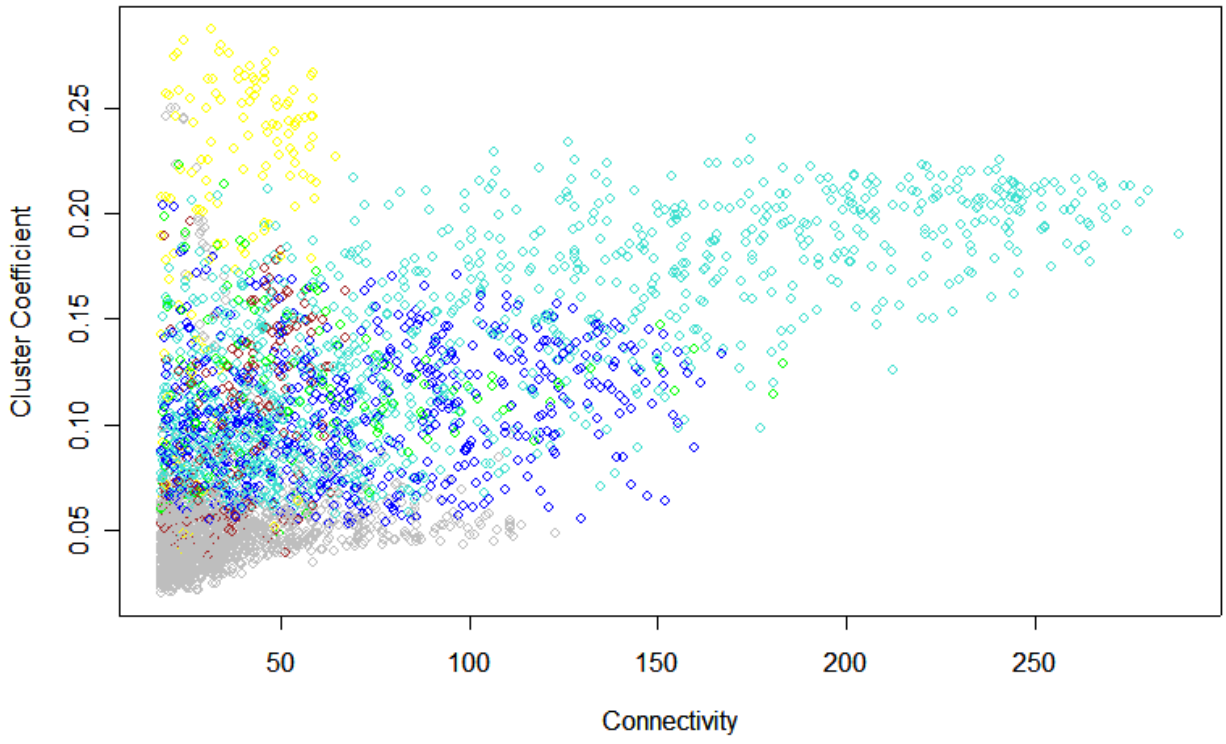


#Comment: the intramodular cluster coefficient (green line) achieves the highest correlation with the gene significance. The TOM based intramodular connectivity kTOM.in (blue line) is superior to the standard connectivity measure k.in (red line) in this application.
The vertical line corresponds to the 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)

```
colorh1: blue
          CC        k
CC 1.000000 0.242394
k  0.242394 1.000000
------------------------------------------------------------------------
colorh1: brown
           CC         k
CC 1.0000000 0.4765162
k  0.4765162 1.0000000
------------------------------------------------------------------------
colorh1: green
            CC          k
CC 1.00000000 0.06135442
k  0.06135442 1.00000000
------------------------------------------------------------------------
colorh1: grey
           CC         k
CC 1.0000000 0.1347935
k  0.1347935 1.0000000
------------------------------------------------------------------------
colorh1: turquoise
          CC        k
CC 1.000000 0.732844
k  0.732844 1.000000
------------------------------------------------------------------------
colorh1: yellow
          CC        k
CC 1.000000 0.484205
k  0.484205 1.000000
```

Unweighted gene co-expression network analysis

Here we choose a hard threshold for dichotomizing the Pearson correlation
matrix

HARD THRESHOLDING

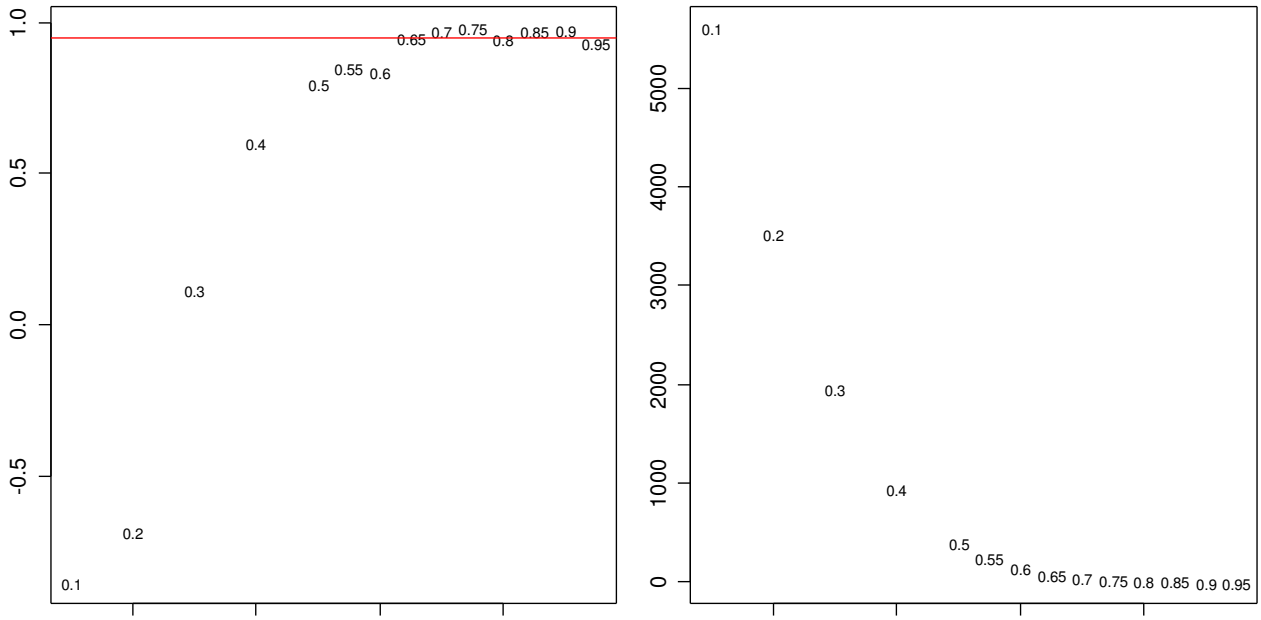
# To construct an unweighted network (hard thresholding),
# we consider the following vector of potential thresholds.
thresholds1= c(seq(.1,.5, by=.1), seq(.55,.95, by=.05))

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 PickHardThreshold can help one to estimate the cut-off value
when using hard thresholding with the step adjacency function.
The first column lists the threshold ("cut"),
the second column lists the corresponding p-value based on the Fisher transform.
The third column reports the resulting scale free topology fitting index R^2.
The fourth column reports the slope of the fitting line.
The fifth 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 pick a hard threshold (cut) with the scale free topology criterion:
aim for high scale free R^2 (column 3), high connectivity (col 6)
and negative slope (around -1, col 4).

RdichotTable=pickHardThreshold(datExpr, thresholds1)[[2]]
gc()

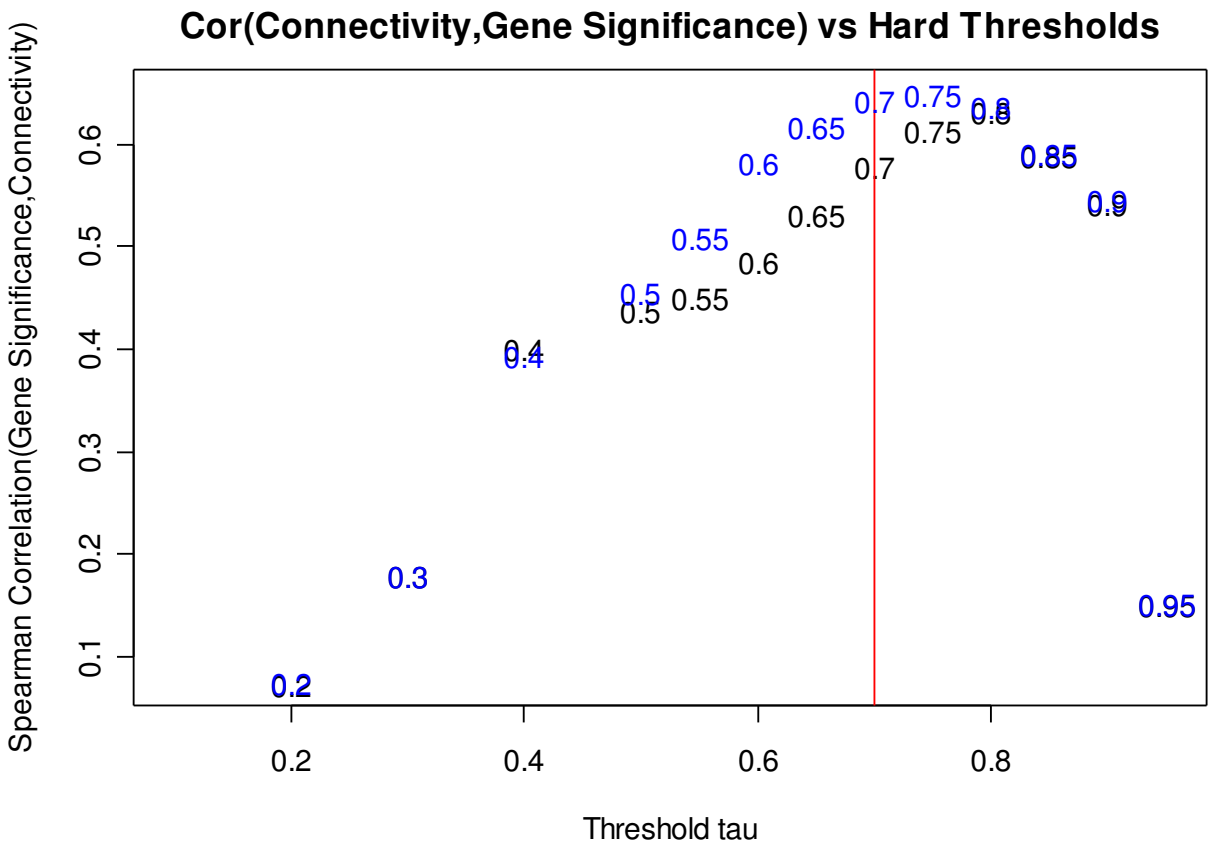
	Cut	p.value	scale.law.R.2	slope.	truncated.R.2	mean.k.	median.k.	max.k.
1	0.10	4.63e-01	0.850	6.5500	0.944	5610.000	5680	6780
2	0.20	1.39e-01	0.682	1.7900	0.953	3530.000	3580	5520
3	0.30	2.47e-02	-0.117	0.0722	0.949	1960.000	1890	4200
4	0.40	2.25e-03	0.603	-0.8360	0.937	947.000	787	2940
5	0.50	8.72e-05	0.795	-1.2100	0.904	395.000	232	1860
6	0.55	1.13e-05	0.850	-1.2300	0.901	243.000	110	1410
7	0.60	1.02e-06	0.837	-1.2400	0.865	145.000	43	1080
8	0.65	5.92e-08	0.952	-1.1100	0.948	85.900	14	795
9	0.70	1.93e-09	0.971	-1.0500	0.968	50.200	4	616
10	0.75	2.88e-11	0.981	-1.0100	0.983	28.900	1	480
11	0.80	1.40e-13	0.946	-1.0300	0.940	15.700	0	383
12	0.85	2.22e-16	0.971	-1.0300	0.969	7.200	0	262
13	0.90	0.00e+00	0.977	-1.0900	0.976	2.170	0	154
14	0.95	0.00e+00	0.933	-1.2600	0.965	0.194	0	47

#Let's plot the scale free topology model fitting index (R^2) versus the cut-off tau. However, the #R^2 values of those cut-offs that lead to a negative slope have been pre-multiplied by -1.
cex1=0.7
gc()
par(mfrow=c(1,2))
plot(RdichotTable[,1], -sign(RdichotTable[,4])*RdichotTable[,3],xlab="Hard Threshold tau",ylab="Scale Free Topology Model Fit,signed R^2", type="n")
text(RdichotTable[,1], -sign(RdichotTable[,4])*RdichotTable[,3] , labels=thresholds1,cex=cex1)
# this line corresponds to using an R^2 cut-off of h
abline(h=0.95,col="red")
plot(RdichotTable[,1], RdichotTable[,6],xlab="Hard Threshold tau",ylab="Mean Connectivity", type="n")
text(RdichotTable[,1], RdichotTable[,6] , labels=thresholds1, cex=cex1)



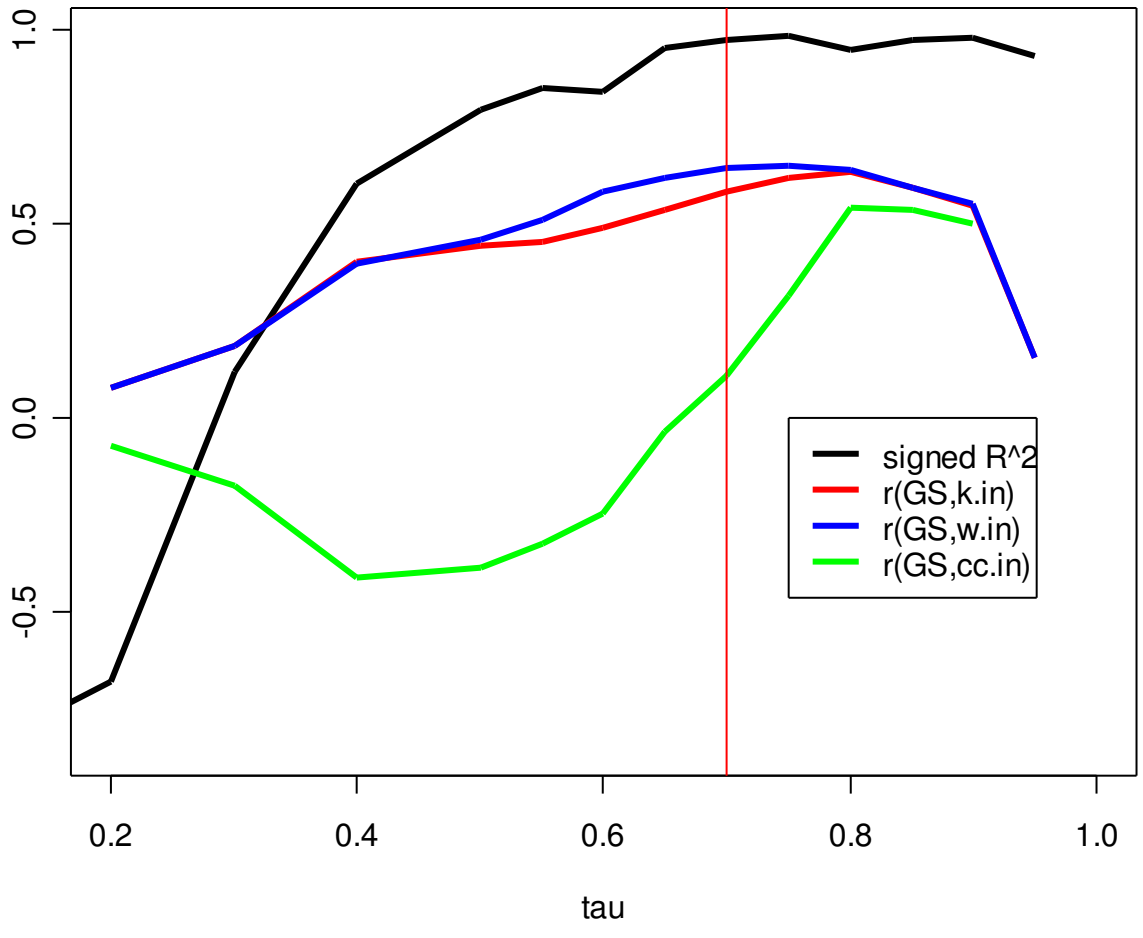
To choose a cut-off value tau, we propose to use the Scale-free Topology Criterion (Zhang and Horvath 2005). Here the focus is on the linear regression model fitting index (denoted as scale.law.R.2) that quantify the extent of how well a network satisfies a scale-free topology. We choose the cut value (tau) of 0.7 for the correlation matrix since this is where the R^2 curve seems to saturates. The red line corresponds to R^2= 0.95. From the above table, we find that the resulting slope looks OK (negative and around -1), and the mean number of connections looks good Below we investigate different choices of tau.

```
# Now we want to see how the correlation between kWithin and gene significance
# changes for different hard threshold values tau within the BROWN module.

corhelp=cor(datExpr[,restConnectivity],use="pairwise.complete.obs")

whichmodule="brown"
# the following data frame contains the intramodular connectivities
# corresponding to different hard thresholds
datconnectivitiesHard=data.frame(matrix(666,nrow=sum(colorh1==whichmodule),ncol=length(thr
esholds1)))
names(datconnectivitiesHard)=paste("kWithinTau",thresholds1,sep="")
for (i in c(1:length(thresholds1)) ) {
datconnectivitiesHard[,i]=apply(abs(corhelp[colorh1==whichmodule,
colorh1==whichmodule])>=thresholds1[i],1,sum)}
SpearmanCorrelationsHard=signif(cor(GeneSignificance[ colorh1==whichmodule],
datconnectivitiesHard, method="s",use="p"))

# Now we define the new connectivity measure omega based on the TOM matrix
# It simply considers TOM as adjacency matrix...
datkTOMINHard=data.frame(matrix(666,nrow=sum(colorh1==whichmodule),ncol=length(threshol
ds1)))
names(datkTOMINHard)=paste("omegaWithinHard",thresholds1,sep="")
for (i in c(1:length(thresholds1)) ) {
datconnectivitiesHard[,i]=apply(
1-TOMdist(abs(corhelp[colorh1==whichmodule,
colorh1==whichmodule])>thresholds1[i]),1,sum)}
SpearmanCorrelationskTOMHard=as.vector(signif(cor(GeneSignificance[
colorh1==whichmodule], datconnectivitiesHard, method="s",use="p")))
# Now we compare the performance of the 2 connectivity measures (k.in and
# kTOM.IN) across different hard thresholds when it comes to predicting
# prognostic genes in the brown module

par(mfrow=c(1,1), mar=c(5, 4, 4, 2) +0.1)
plot(thresholds1, SpearmanCorrelationsHard, main="
Cor(Connectivity,Gene Significance) vs Hard Thresholds",ylab="Spearman Correlation(Gene
Significance,Connectivity)",xlab="Threshold tau", type="n",
ylim=range(c(SpearmanCorrelationsHard, SpearmanCorrelationskTOMHard),na.rm=T))
text(thresholds1, SpearmanCorrelationsHard,labels=thresholds1,col="black")
# this draws a vertical line at the tau that was chosen by the
# scale free topology criterion.
abline(v=0.7,col="red")
points(thresholds1, SpearmanCorrelationskTOMHard, type="n")
text(thresholds1, SpearmanCorrelationskTOMHard,labels=thresholds1,col="blue")
```




```
# Now we define the intramodular cluster coefficient
datCCinHard=data.frame(matrix(666,nrow=sum(colorh1==whichmodule),ncol=length(thresholds1
)))
names(datCCinHard)=paste("CCinHard",thresholds1,sep="")
for (i in c(1:length(thresholds1)) ) {
datCCinHard[,i]= clusterCoef(abs(corhelp[colorh1==whichmodule,
colorh1==whichmodule])>thresholds1[i])}
SpearmanCorrelationsCCinHard=as.vector(signif(cor(GeneSignificance[ colorh1==whichmodule],
datCCinHard, method="s",use="p")))
```

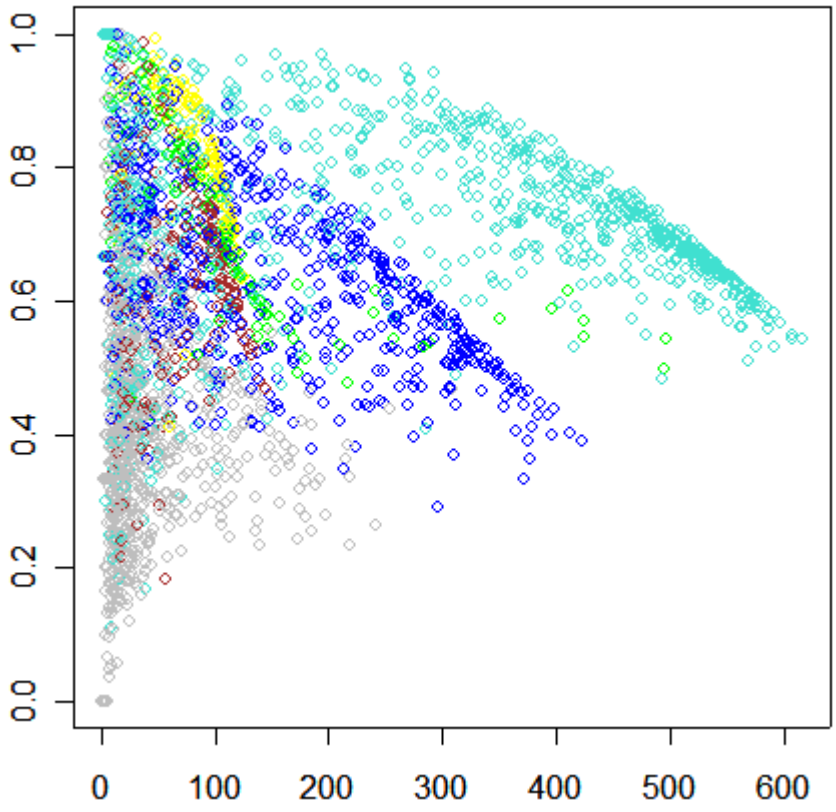
```
# Now we compare the performance of the connectivity measures (k.in,
# kTOM.IN, cluster coefficience) across different hard thresholds when it comes to predicting
# prognostic genes in the brown module

dathelpHard=data.frame(signedRsquared=-sign(RdichotTable[,4])*RdichotTable[,3],
corGSkINHard =as.vector(SpearmanCorrelationsHard), corGSwINHard=
as.vector(SpearmanCorrelationskTOMHard),corGSCCHard=as.vector(SpearmanCorrelationsCCin
Hard))
matplot(thresholds1,dathelpHard,type="l",lty=1,lwd=3,col=c("black","red","blue","green"),ylab=""
,xlab="tau",xlim=c(.2,1))
legend(0.75,0, 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)
abline(v=.7,col="red")
```



#Note that very high or very small threshold values lead to a small correlation,

#i.e. a diminished biological signal. The red line corresponds to the threshold that was picked using #the scale free topology criterion. The scale free topology criterion picked a threshold that leads to #very high significant correlation between node connectivity and gene significance.

```
AdjMatHARD=abs(cor(datExpr[,restConnectivity]))>0.70+0.0
diag(AdjMatHARD)=0
cluster.coefrestHARD= clusterCoef(AdjMatHARD)
ConnectivityHARD= apply(AdjMatHARD,2,sum)
par(mfrow=c(1,1))
plot(ConnectivityHARD,cluster.coefrestHARD,col=as.character(colorh1),xlab="Connectivity",yla
b="Cluster Coefficient" )
```



Now we correlate the cluster coefficient with connectivity by module in the unweighted network

```
restHub=ConnectivityHARD>100
by(data.frame(CC= cluster.coefrestHARD[restHub], k=ConnectivityHARD[restHub]),
INDICES=colorh1[restHub],FUN=cor)
```

```
colorh1[restHub]: blue
           CC          k
CC  1.0000000 -0.5704594
k  -0.5704594  1.0000000
------------------------------------------------------------------------
colorh1[restHub]: brown
          CC         k
CC  1.000000 -0.759792
k  -0.759792  1.000000
------------------------------------------------------------------------
colorh1[restHub]: green
           CC          k
CC  1.0000000 -0.5427443
k  -0.5427443  1.0000000
------------------------------------------------------------------------
colorh1[restHub]: grey
          CC         k
CC  1.000000 -0.205176
k  -0.205176  1.000000
------------------------------------------------------------------------
colorh1[restHub]: turquoise
           CC          k
CC  1.0000000 -0.4295943
k  -0.4295943  1.0000000
------------------------------------------------------------------------
colorh1[restHub]: yellow
           CC          k
CC  1.0000000 -0.9097497
k  -0.9097497  1.0000000
```

Let's compare a summary of soft thresholding to one of hard thresholding

```
#Here are summary statisticis for the different quantities.
apply(dathelpHard,2,summary)
$signedRsquared
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-0.8500  0.6510  0.8915  0.6001  0.9663  0.9810
$corGSkINHard
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
0.07569 0.40200 0.48760 0.43780 0.58110 0.63380 1.00000
$corGSwINHard
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
0.07753 0.39630 0.54800 0.46570 0.61860 0.64920 1.00000
$corGSCCHard
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
-0.41460 -0.27070 -0.05500  0.02667  0.35850  0.53910  2.00000
apply(dathelpSoft,2,summary)
        signedRsquared corGSkINSoft corGSwINSoft corGSCCSoft
Min.            0.0927       0.5416       0.5802      0.6698
1st Qu.         0.9115       0.5797       0.6527      0.7095
Median          0.9660       0.6063       0.6637      0.7539
Mean            0.8344       0.5990       0.6577      0.7414
3rd Qu.         0.9705       0.6250       0.6764      0.7757
Max.            0.9780       0.6321       0.6808      0.7918
```

Comments

- Using the Max. values, we find that when it comes to correlating gene significance with a centrality measure (connectivity or cluster coefficient), the soft intramodular cluster coefficient is most highly correlated with gene significance. Next comes the soft TOM based connectivity (wIN), then kIN. In other applications, the cluster coefficient is *not* the best centrality measure. But please let us know if you find empirical evidence that the cluster coefficient is a good centrality measure. In the latest version of the manuscript Zhang and Horvath (2005) we present a theoretical argument that shows that is a weak positive correlation between intramodular cluster coefficient and intramodular connectivity in *weighted* networks. In contrast, one finds a negative correlation between cluster coefficient and connectivity in unweighted networks (see the plot in the appendix).
- We find that soft thresholding is superior to hard thresholding especially for low values of the scale free topology R^2.
- In our opinion, soft centrality (connectivity) measures are better than hard measures because they are relatively robust with respect to the parameter of the adjacency function. For soft thresholding even choosing a power of beta=1 leads to a good biological signal (correlation). In contrast, choosing a hard threshold of tau=0.2 leads to a much reduced biological signal. Robustness is a very attractive property in this type of analysis since picking parameters of the adjacency function is rather ad-hoc.
- The Scale free topology criterion leads to estimates of the adjacency function that often have good biological signal.

APPENDIX: Constructing an unweighted networks and comparing it to the weighted nework.

Here we study whether the `soft' modules of the unweighted network described above can also be found in the unweighted network

Recall that the soft module assignment in the 3600 most connected genes is given by

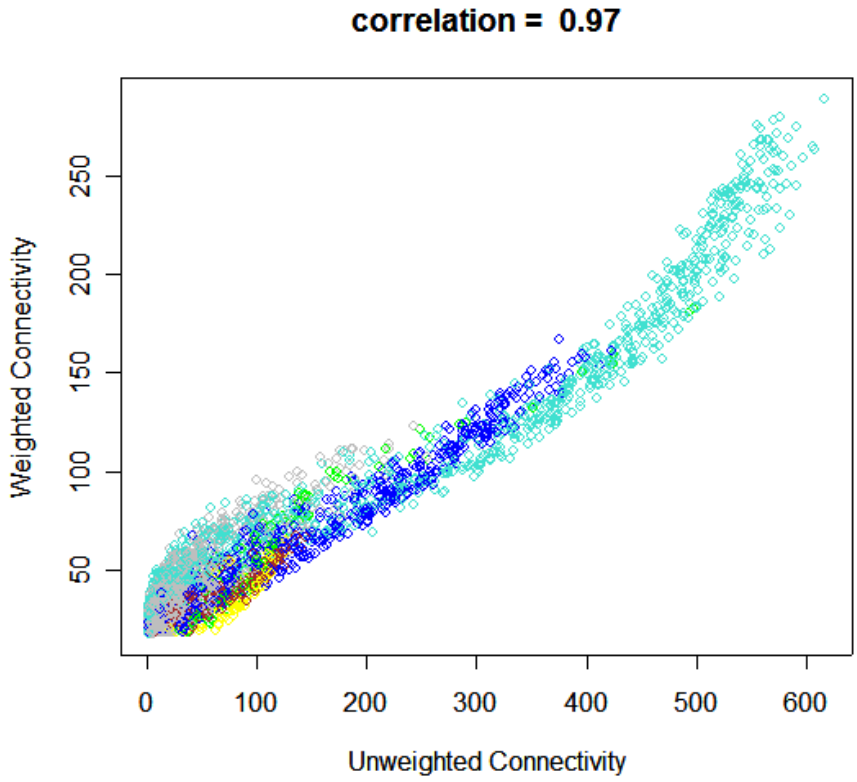
```
colorh1=as.character(datSummary$color1[restConnectivity])
```

#Let's define the adjacency matrix of an unweighted network

```
ADJ= abs(cor(datExpr[,restConnectivity],use="p"))>0.7
gc()
# This is the unweighted connectivity
k=as.vector(apply(ADJ,2,sum))
```

Let's compare weighted to unweighted connectivity in a scatter plot

```
plot(k, Connectivity[restConnectivity],xlab="Unweighted
Connectivity",ylab="Weighted Connectivity",main=paste( "correlation =
",signif(cor(k,Connectivity[restConnectivity]),2)),col=colorh1)
```



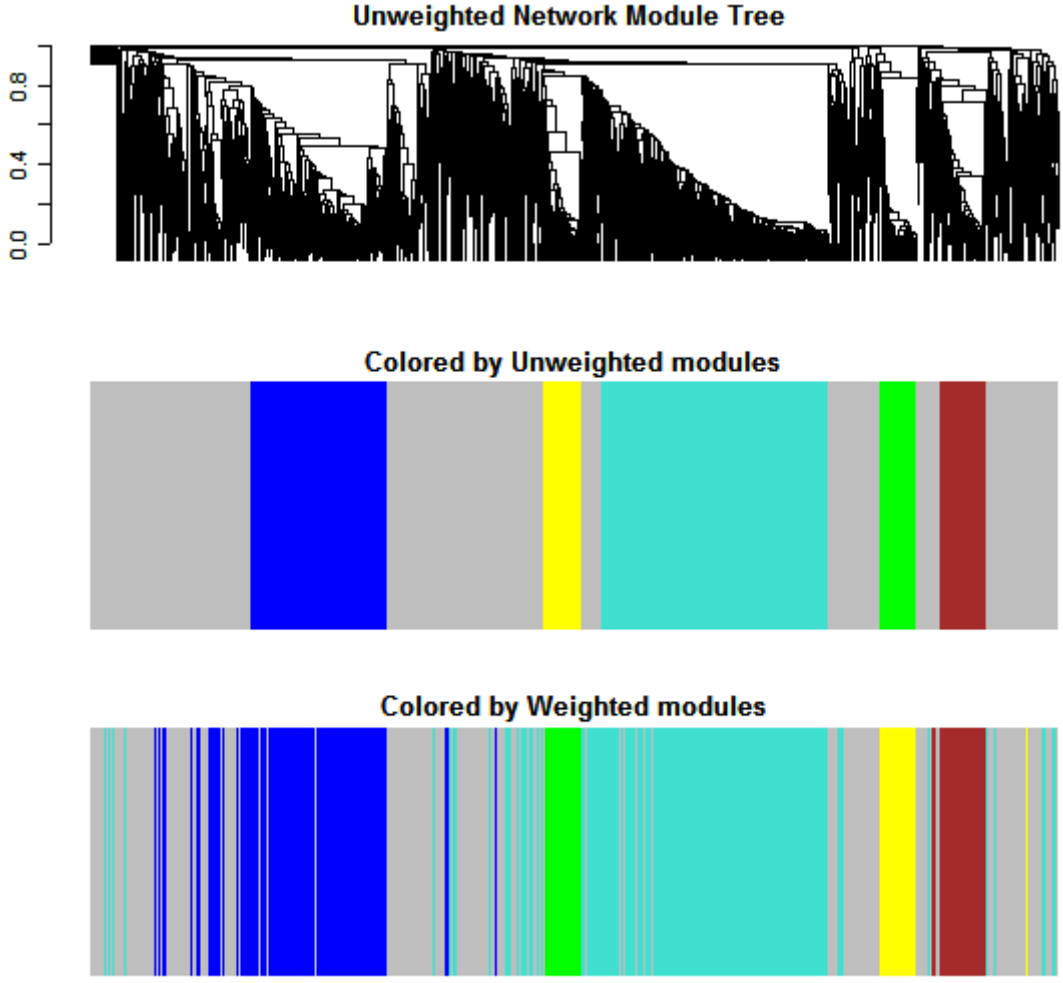
Comments:

- the connectivity measures is highly preserved between weighted and unweighted networks as long as the scale free topology criterion is used for network construction. It is re-assuring that the biological findings will be robust with respect to the network construction method
- The genes with the highest whole network connectivity are contained in the turquoise module, which happens to be the largest module. The second most connected genes are in the blue module, which is the second largest module, etc.

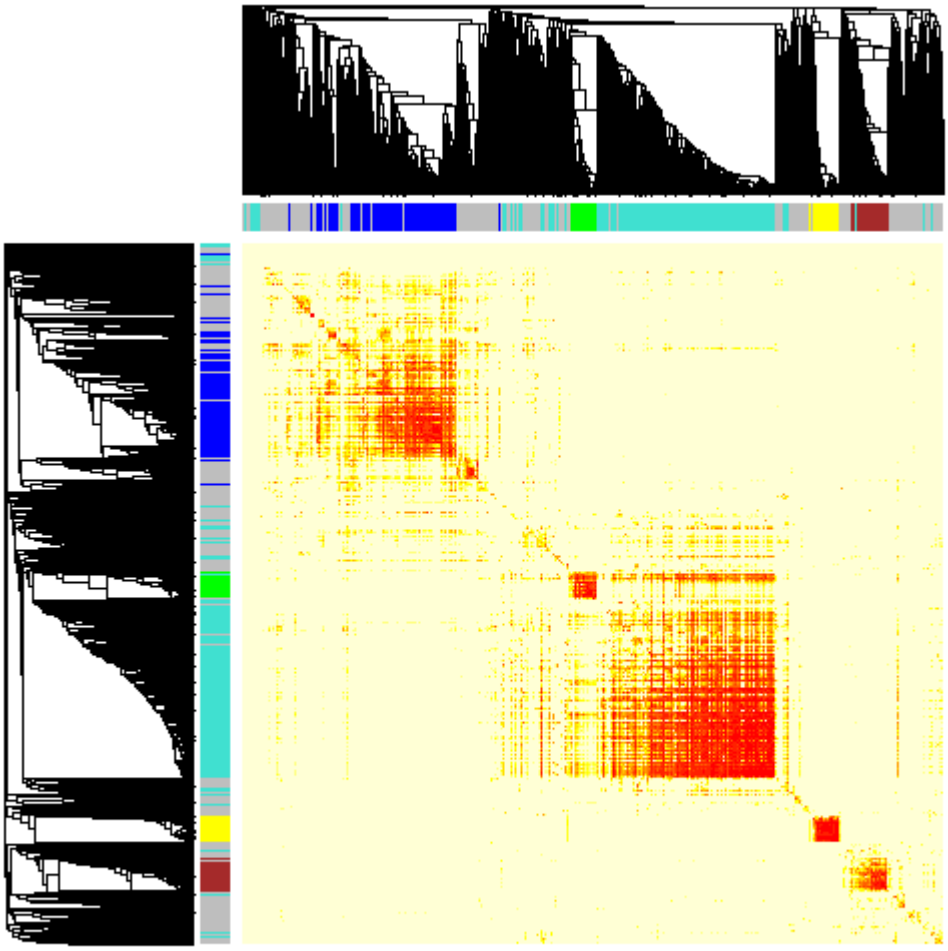
The following code computes the topological overlap matrix based on the

```
# adjacency matrix.
# TIME: Takes about 10 minutes....
dissTOM=TOMdist(ADJ)
gc()

# Now we carry out hierarchical clustering with the TOM matrix. Branches of the
# resulting clustering tree will be used to define gene modules.

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

# By our definition, modules correspond to branches of the tree.
# The function modulecolor2 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.

colorh2=as.character(modulecolor2(hierTOM,h1=.75, minsize1=125))

# The following table shows that there is fairly high agreement
# between the soft and the hard module assignments

table(colorh1,colorh2)
                          colorh2
colorh1     blue brown green grey turquoise yellow
  blue       466     0     0   140     0         0
  brown        0   160     0    25     0         0
  green        0     0     0     5     3       128
  grey        35     7     0  1349    22         5
  turquoise    0     1     0   281   817        13
  yellow       0     0   133    10     0         0
#Rand index to measure agreement between the clusterings
randIndex(table(colorh1,colorh2))
[1] 0.5919202

# Note that the brown module in the weighted network (colorh1) is a little bit
# larger than the corresponding module in the unweighted network. But the point
# is that it is highly preserved. Since this module is of biological interest,
# the good news is that the biological findings are robust with respect to the
# network construction method as long as the scale free topology criterion is
# used to construct the network.
```

```
par(mfrow=c(3,1), mar=c(2,2,2,1))
plot(hierTOM, main="Unweighted Network Module Tree ", labels=F, xlab="",
sub="");
hclustplot1(hierTOM, colorh2, main="Colored by Unweighted modules")
hclustplot1(hierTOM, colorh1, main="Colored by Weighted modules")
```

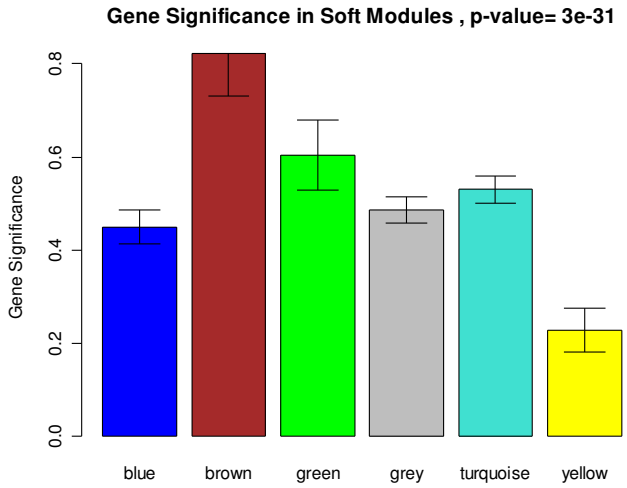


```
# An alternative view of this is the so called TOM plot that is generated by the
# function TOMplot
# Inputs:  TOM  distance measure, hierarchical (hclust) object, color
# Here we use the unweighted module tree but color it by the weighted modules.
TOMplot(dissTOM , hierTOM, as.character(datSummary$color1[restConnectivity]))
gc()
```

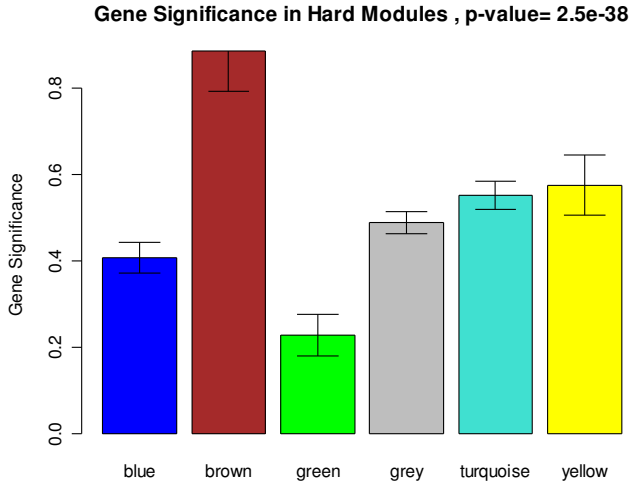


#Comment: module assignment is highly preserved.

verboseBarplot(GeneSignificance,colorh1,main="Gene Significance in Soft Modules")

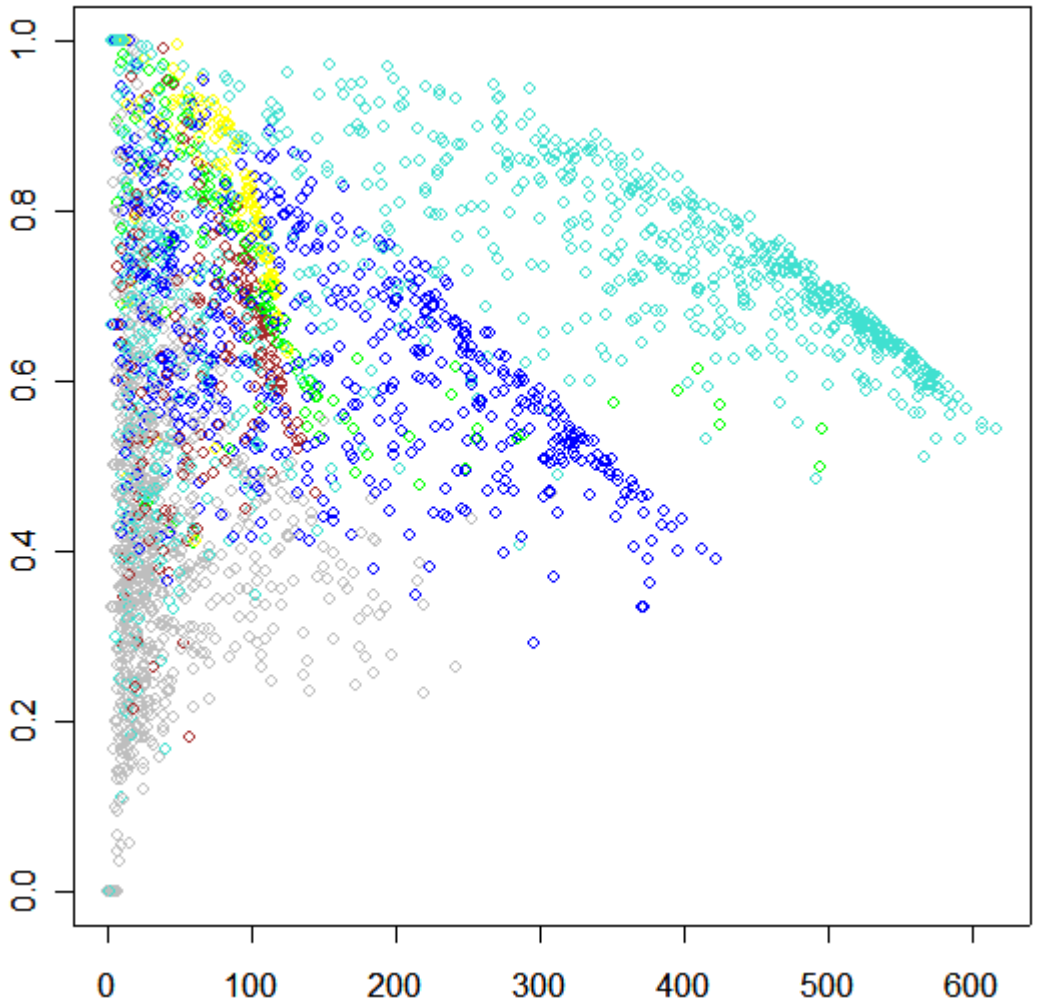


verboseBarplot(GeneSignificance,colorh2,main="Gene Significance in Hard Modules")



Now we compute the cluster coefficient.
#Since this is computationally intensive, we recommend to skip it.
CC= clusterCoef(ADJ)
gc()

plot(k,CC,col= as.character(datSummary$color1[restConnectivity]),xlab="Connectivity (Hard)",ylab="Cluster Coefficient")




```
# Comment: for unweighted networks there is an inverse relationship between
cluster coefficient and connectivity. This is different from the case of
weighted networks. In our opinion, this inverse relationship is an artifact of
hard thresholding, see Zhang and Horvath (2005).
```

# Now we study how intramodule connectivity relates to gene significance
ConnectivityMeasures=intramodularConnectivity(ADJ,colorh2)

```
# The following plots would show the gene significance vs intromodular
# connectivity

par(mfrow=c(2,3))
colorlevels=unique(colorh2)
whichmodule=colorlevels[[1]];restrict1=colorh2==whichmodule
verboseScatterplot(ConnectivityMeasures$kWithin[restrict1],
GeneSignificance[restrict1],col=colorh2[restrict1],main=whichmodule)
whichmodule= colorlevels[[2]];restrict1=colorh2==whichmodule
verboseScatterplot(ConnectivityMeasures$kWithin[restrict1],
GeneSignificance[restrict1],col=colorh2[restrict1],main=whichmodule)
whichmodule= colorlevels[[3]];restrict1=colorh2==whichmodule
verboseScatterplot(ConnectivityMeasures$kWithin[restrict1],
GeneSignificance[restrict1],col=colorh2[restrict1],main=whichmodule)
whichmodule= colorlevels[[4]];restrict1=colorh2==whichmodule
verboseScatterplot(ConnectivityMeasures$kWithin[restrict1],
GeneSignificance[restrict1],col=colorh2[restrict1],main=whichmodule)
whichmodule= colorlevels[[5]];restrict1=colorh2==whichmodule
verboseScatterplot(ConnectivityMeasures$kWithin[restrict1],
GeneSignificance[restrict1],col=colorh2[restrict1],main=whichmodule)
whichmodule= colorlevels[[6]];restrict1=colorh2==whichmodule
verboseScatterplot(ConnectivityMeasures$kWithin[restrict1],
GeneSignificance[restrict1],col=colorh2[restrict1],main=whichmodule)
```

THE END:
To cite the code and methods in this manual, please use

Bin Zhang and Steve Horvath (2005) "A General Framework for Weighted Gene Co-Expression Network Analysis", Statistical Applications in Genetics and Molecular Biology: Vol. 4: No. 1, Article 17