

Weighted Gene Co-expression Network Analysis (WGCNA) R Tutorial, **Part B** Module Eigengene, Survival time, and Proliferation

Steve Horvath, Paul Mischel

Correspondence: shorvath@mednet.ucla.edu, <http://www.ph.ucla.edu/biostat/people/horvath.htm>

This is part B of a self-contained R software tutorial. The first few pages are very similar to those of part A, but here we focus on studying the brown module and relating individual genes to survival outcome. Thus, the reader will be able to reproduce all of our findings. This document also serves as a tutorial to weighted gene co-expression network analysis. Some familiarity with the R software is desirable but the document is fairly self-contained.

This tutorial and the data files can be found at the following webpage:

<http://www.genetics.ucla.edu/labs/horvath/CoexpressionNetwork/ASPMgene>

More material on weighted network analysis can be found here

<http://www.genetics.ucla.edu/labs/horvath/CoexpressionNetwork/>

Abstract

The data and biological implications are described in part A and in

- *Horvath S, Zhang B, Carlson M, Lu KV, Zhu S, Felciano RM, Laurance MF, Zhao W, Shu, Q, Lee Y, Scheck AC, Liao 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.*

Patient samples:

Brain Cancer (GBM) microarray data involving the 3600 most connected genes. This file contains 2 sets: the first 55 arrays are the training set. The next 65 patients are the validation set. Columns corresponding to the 65 validation samples start with Set65

Rows are genes, columns are patients. First 4 lines contain patient information (died=1), etc.

Contents part B (the beginning overlaps with part A)

- *) Weighted brain cancer network construction based on ***3600*** most connected genes
- *) Gene significance and intramodular connectivity in data sets I and II
- *) Module Eigengene and its relationship to individual genes
- *) Regressing survival time on individual gene expression and the module eigengene

Generation of weighted gene coexpression network:

Forms the topic of GBMtutorialPartAHorvath.doc which can be found at our webpage
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.

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

<http://www.genetics.ucla.edu/labs/horvath/CoexpressionNetwork/ASPMgene>
Download the zip file containing:
1) R function file: "NetworkFunctions.txt", which contains several R functions
needed for Network Analysis.
2) The data files and this tutorial

Unzip all the files into the same directory.

The user should copy and paste the following script into the R session.

Text after "#" is a comment and is automatically ignored by R.

Set the working directory of the R session by using the following command.

Please adapt the following path. Note that we use / instead of \ in the path.

```
setwd("C:/Documents and Settings/shorvath/My  
Documents/ADAG/PaulMischel/GBMnetworkpaper/Webpage/BrainCancerTutorial")
```

read in the R libraries

```
library(MASS) # standard, no need to install
```

```
library(class) # standard, no need to install
```

```
library(cluster)
```

```
library(sma) # install it for the function plot.mat
```

```
library(impute)# install it for imputing missing value
```

```
library(Hmisc)# install it for the C-index calculations
```

```

library(survival)

# Memory
# check the maximum memory that can be allocated
memory.size(TRUE)/1024
# increase the available memory
memory.limit(size=4000)

# Please adapt the following path to wherever the file NetworkFunctions.txt
# Please adapt the following path to wherever the file NetworkFunctions.txt
source("C:/Documents and Settings/shorvath/My Documents/RFunctions/NetworkFunctions.txt")

# the following contains expression data of the 3600 most connected genes (see part A)
# across the 55 GBM sample data set and the 65 GBM sample set.
dat0=read.csv("GBM3600Set55and65andClinicalInformation.csv")

# the following data frame contains
# the gene expression data: columns are genes, rows are arrays (samples)
datExprdataOne =data.frame(t(dat0[-c(1:4),18:72]))
datExprdataTwo=data.frame( t(dat0[-c(1:4),73:137]))
names(datExprdataOne)=as.character( dat0$gbm133a[-c(1:4)])
names(datExprdataTwo)=as.character( dat0$gbm133a[-c(1:4)])

# these data frames contain the clinical data of the patients
datClinicaldataOne=dat0[1:4,18:72]
datClinicaldataTwo=dat0[1:4,73:137]
datClinicaldataOne =data.frame(t(dat0[c(1:4),18:72]))
names(datClinicaldataOne)=as.character(dat0[1:4,1])
datClinicaldataTwo=data.frame( t(dat0[c(1:4),73:137]))
names(datClinicaldataTwo)=as.character(dat0[1:4,1])

# for (i in c(1:dim(datExprdataOne)[[1]])) { datExprdataOne[i,]=as.numeric(datExprdataOne[i,])}
# for (i in c(1:dim(datExprdataTwo)[[1]])) { datExprdataTwo[i,]=as.numeric(datExprdataTwo[i,])}
# this data frame contains information on the probesets
datSummary=dat0[-c(1:4),c(1:17)]

dim(datExprdataOne)
dim(datExprdataTwo)
dim(datSummary)
rm(dat0);collect_garbage()

```

Quotes from an unrecognized statistician:

"What is best let alone, that accursed thing is not always what least allures."

"Aye, aye! and I'll chase him round Good Hope, and round the Horn, and round the Norway Maelstrom, and round perdition's flames before I give him up."

"...to the last I grapple with thee; from hell's heart I stab at thee; for hates sake I spit my last breath at thee."
 Captain Ahab in "Moby Dick" by Melville

#SOFT THRESHOLDING

#As described in tutorial A, we use the following power for the power adjacency function.

beta1=6

```
DegreedataOne=SoftConnectivity(datExprdataOne,power=beta1)-1
```

```
DegreedataTwo= SoftConnectivity(datExprdataTwo,power=beta1)-1
```

Let's create a scale free topology plot for the dataOne and dataTwo network

```
par(mfrow=c(2,2))
```

```
ScaleFreePlot1(DegreedataOne, AF1=paste("data set I, power=",beta1), truncated1=F);
```

```
ScaleFreePlot1(DegreedataTwo, AF1=paste("data set II, power=",beta1), truncated1=F);
```

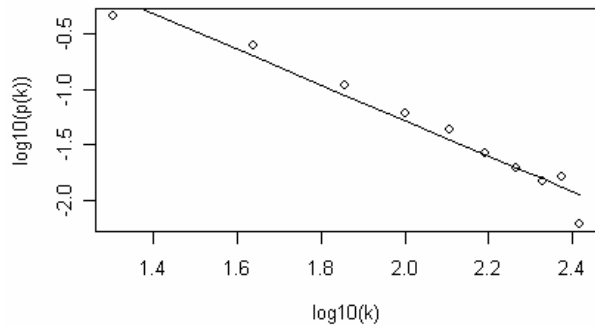
this relates the whole network connectivity measures in the 2 networks

```
scatterplot1(DegreedataOne, DegreedataTwo,xlab1="3600 genes, connectivity in data I",ylab1="k  
in data II",title1="k (data I) vs k (data II)",cex1=1)
```

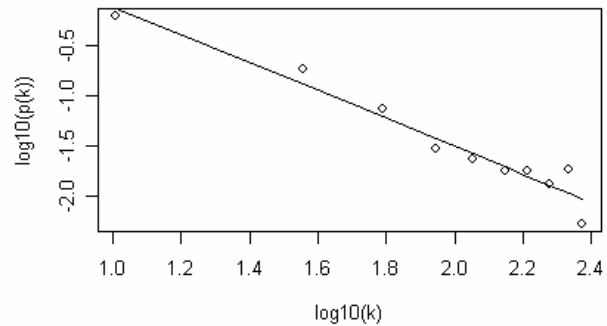
In contrast to the plot in tutorial part A which involves 8000 genes,

these plots are for the network comprised of 3600 genes

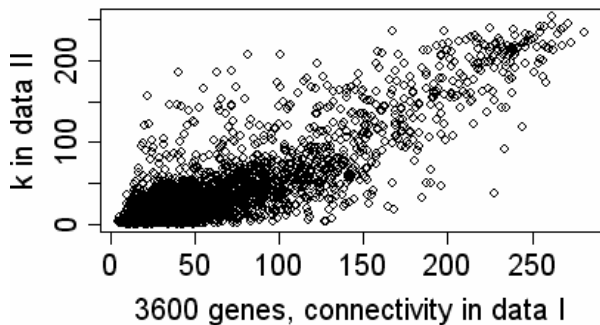
data set I, power= 6 , scale R^2= 0.95 , slope= -1.6



data set II, power= 6 , scale R^2= 0.94 , slope= -1.39



k (data I) vs k (data II) cor= 0.73 p= <10^{-20}

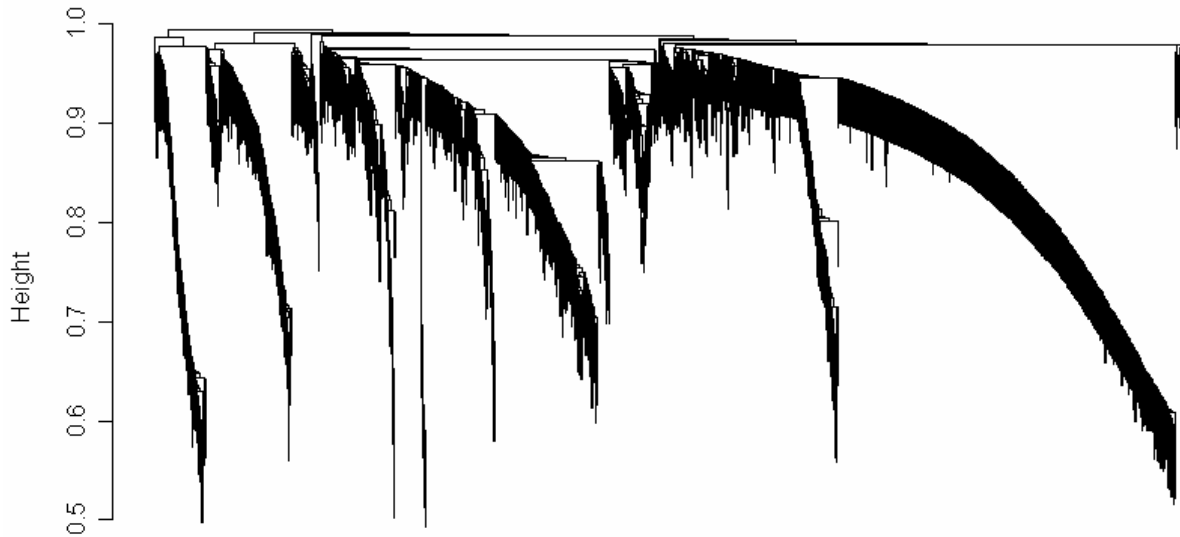


```

# Module Detection (detailed in part A!)
# To group genes with coherent expression profiles into modules, we use average linkage
# hierarchical clustering, which uses the topological overlap measure as dissimilarity.
# This code allows one to restrict the analysis to the most connected genes,
# which may speed up calculations when it comes to module detection.
DegCut = 3601 # number of most connected genes that will be considered
DegreeRank = rank(-DegreedataOne)
vardataOne=as.vector(apply(datExprdataOne,2,var))
vardataTwo= as.vector(apply(datExprdataTwo,2,var))
# Since we want to compare the results between data sets I and II we restrict the analysis to
# the most connected probesets with non-zero variance in both data sets
restDegree = DegreeRank <= DegCut & vardataOne>0 &vardataTwo>0
# The following code computes the topological overlap matrix
dissGTOMdataOne=TOMdist1(abs(cor(datExprdataOne[,restDegree],use="p"))^beta1)
collect_garbage()
hierGTOMdataOne = hclust(as.dist(dissGTOMdataOne),method="average");
collect_garbage()
par(mfrow=c(1,1))
plot(hierGTOMdataOne,labels=F,main="Dendrogram, data set I")

```

Dendrogram, data set I



```

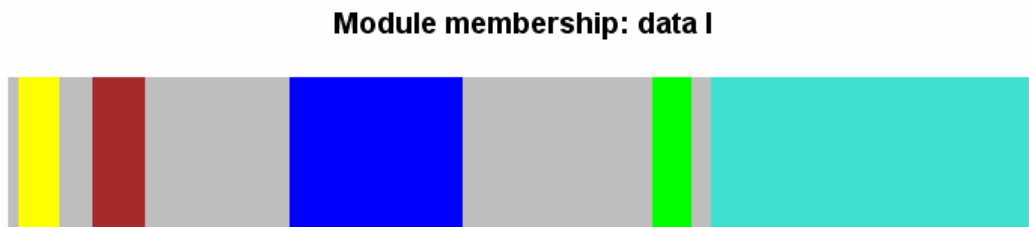
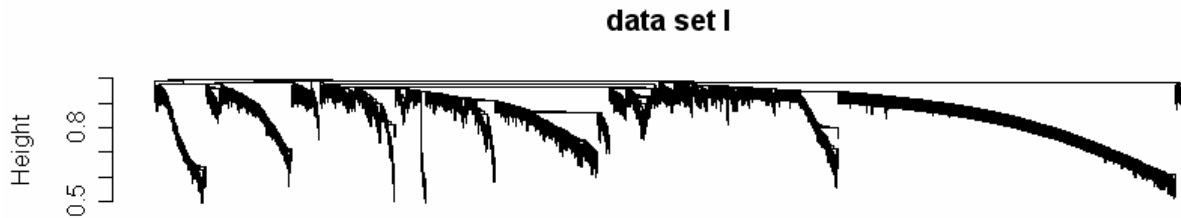
as.dist(dissGTOMdataOne)
hclust (*, "average")

```

```

# 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.
# But in other applications smaller modules may also be of interest.
colorhdataOne=as.character(modulecolor2(hierGTOMdataOne,h1=.94, minsize1=125))
par(mfrow=c(2,1))
plot(hierGTOMdataOne, main="data set I", labels=F, xlab="", sub="");
hclustplot1(hierGTOMdataOne,colorhdataOne, title1="Module membership: data I")
# COMMENT: The colors are assigned based on module size. Turquoise (others refer to it as cyan)
# colors the largest module, next comes blue, etc. Just type table(colorhdataOne) to figure out
# which color corresponds to what module size.

```



This is Figure 1a in our article

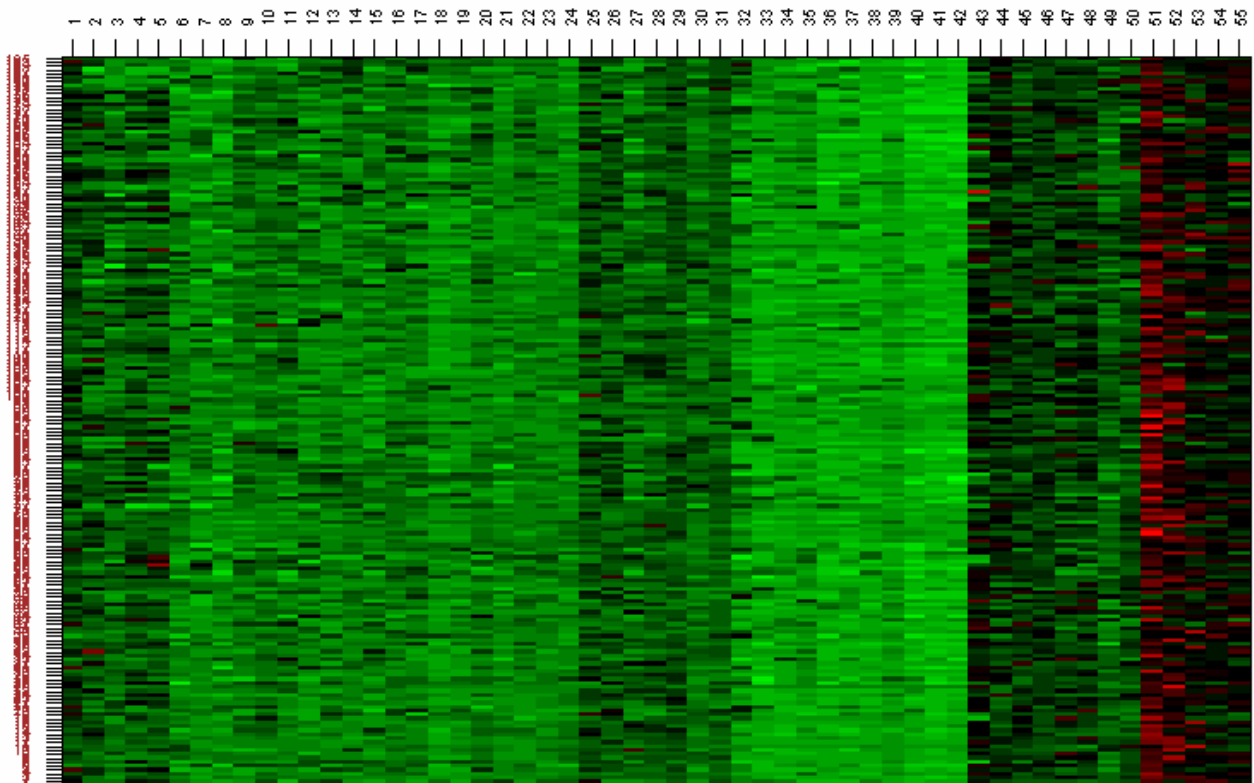
Quote: Two archetypes have defined America's sense of destiny. One is the self-made man, who believes he will get rich through his own hard work. The other is the gambler, who believes that with the the next turn of the cards, providence will deliver the Main Chance. Jackson Learns "Something for Nothing: luck in America".

```
# NOW WE DEFINE THE GENE SIGNIFICANCE VARIABLE,
# which equals minus log10 of the univariate Cox regression p-value for predicting survival
# on the basis of the gene expression info
```

```
# Here we define the prognostic gene significance measures in the data set I and data set II
GSdataOne=-log10(datSummary$Set55Coxpvalue)[restDegree]
GSdataTwo=-log10(datSummary$Set65Coxpvalue)[restDegree]
```

```
# 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(1,1), mar=c(1, 2, 4, 1))
which.module="brown"
ClusterSamples=hclust(dist(datExprdataOne[,restDegree][,colorhdataOne==which.module]
),method="average")
# for this module we find
plot.mat(t(scale(datExprdataOne[ClusterSamples$order,restDegree][,colorhdataOne==which.module
]),nrgcols=30,rlabels=T, clabels=T,rcols=which.module,
title=paste("dataOne set, heatmap",which.module,"module") )
55 set, heatmap brown module
```



```

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

# The function DegreeInOut computes the whole network connectivity kTotal,
# the within module connectivity (kWithin). kOut=kTotal-kWithin and
# and kDiff=kIn-kOut=2*kIN-kTotal

AlldegreesdataOne=DegreeInOut(abs(cor(datExprdataOne[,restDegree],use="p"))^beta1,colorhdat
aOne)

names(AlldegreesdataOne)
[1] "kTotal" "kWithin" "kOut" "kDiff"

# The function WithinModuleAnalysis1 relates the connectivities kTotal, kWithin etc to the gene
# gene significance information within each module.
# Output: first column reports the p-value of the Spearman correlation
# test between the connectivity measure and the node significance (gene significance).
# The second column contains the Spearman correlation between connectivity
# and node significance. The remaining columns list Spearman correlations.

WithinModuleAnalysis1(AlldegreesdataOne,GSdataOne,colorhdataOne)

Excerpt
couleur: brown
      variable NS.CorPval NS.cor kTotal kWithin kOut kDiff
kTotal kTotal 1.9e-08 0.40 1.00 0.910 0.420 0.59
kWithin kWithin 7.1e-19 0.59 0.91 1.000 0.074 0.86
kOut kOut 7.0e-06 -0.32 0.42 0.074 1.000 -0.38
kDiff kDiff 1.1e-25 0.67 0.59 0.860 -0.380 1.00

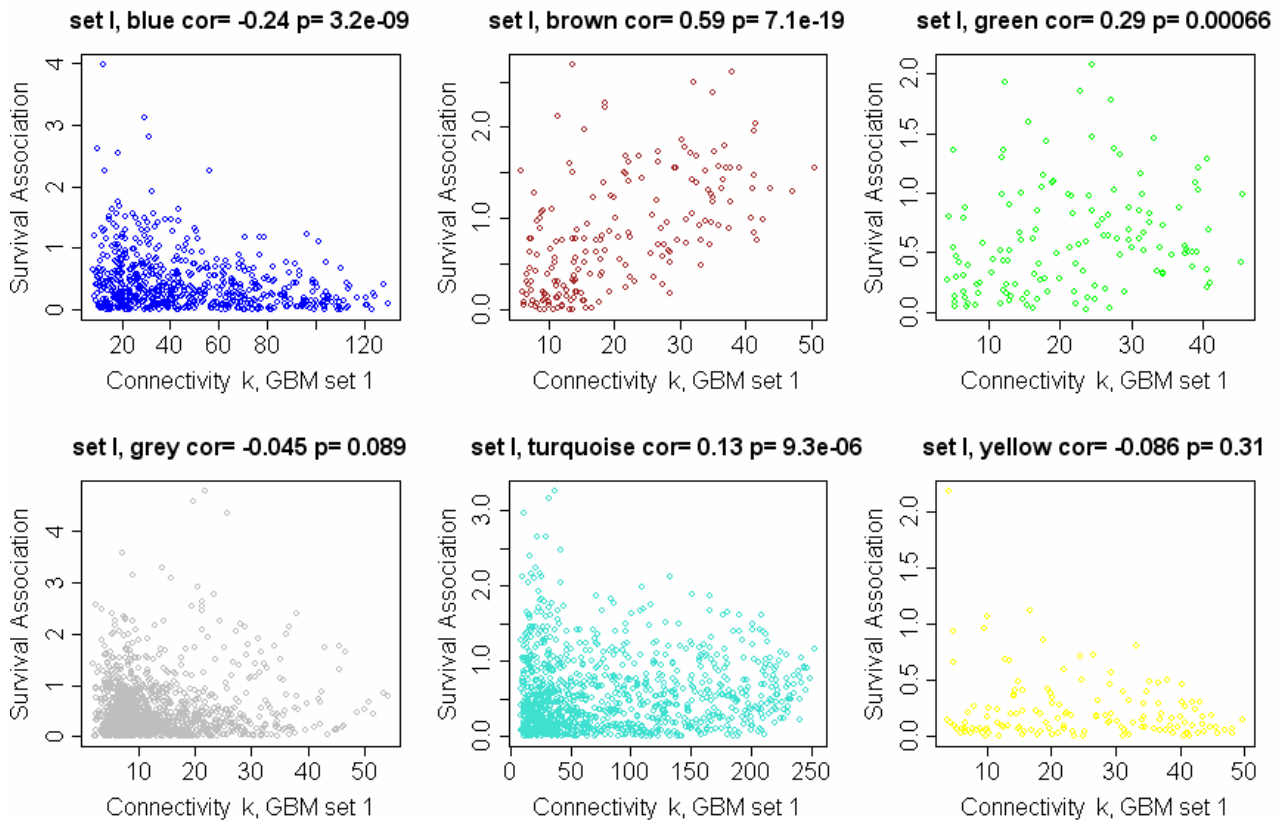
# Comments: Warning messages can be ignored at this point, they simply point out
# that the Spearman test p-values may be incorrect due to ties in: cor.test (Spearman correlation)
# The prefix "NS" means node significance (here gene significance).
# For the brown module we find that kWithin is significantly correlated with
# gene significance (p=7.1e-19, corresponding Spearman correlation= 0.59)
# Note that kWithin and kTotal are highly correlated (Spearman correlation=0.91).

```

```

# The following plots show the gene significance vs intramodular connectivity
# in the data set I
colorlevels=levels(factor(colorhdataOne))
par(mfrow=c(2,3))
for (i in c(1:length(colorlevels) ) ) {
  whichmodule=colorlevels[[i]];restrict1=colorhdataOne==whichmodule
  scatterplot1(AllddegreesdataOne$kWithin[restrict1],
  GSdataOne[restrict1],col1=colorhdataOne[restrict1],xlab1="Connectivity k, GBM set
  1",ylab1="Survival Association, -log(p)",title1= paste("set I,", whichmodule))
}

```

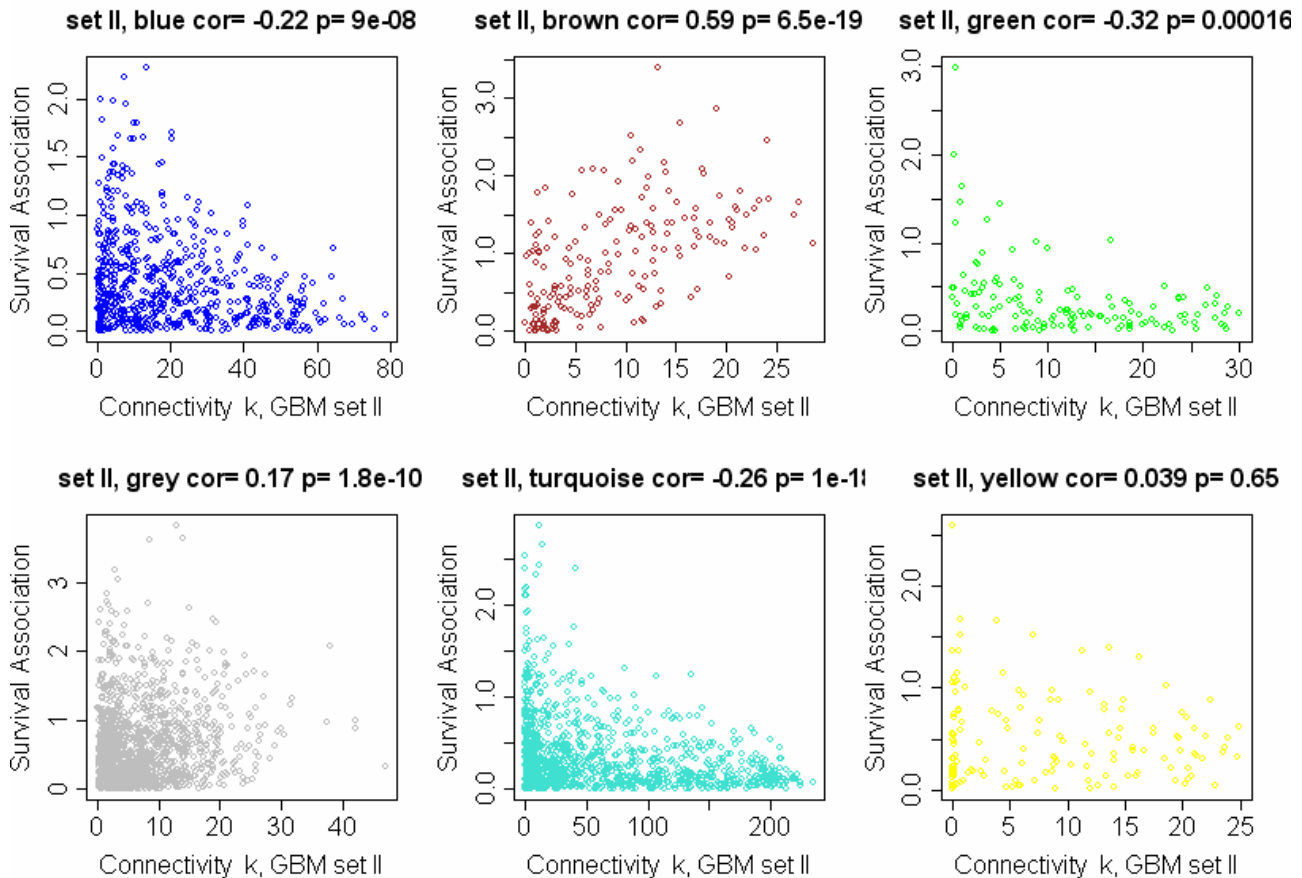


If you don't see a relationship between GS and connectivity in your data,
and, more importantly, if the following is not significant
ModuleEnrichment1(GSdataOne,colorhdataOne)
then check whether you read in the data correctly. Here is quote from an unrecognized statistician #who accidentally
permuted the relationship between datExpr and GS.
Quote:
#"I cudda had class! I cudda been a contender! I cudda been somebody, instead of a bum which is
what I am." Marlon Brando, On the Waterfront

```
# Now we repeat the within module analysis in data set II
AlldegreesdataTwo=DegreeInOut(abs(cor(datExprdataTwo[,restDegree],use="p"))^beta1,colorhdataOne)
WithinModuleAnalysis1(AlldegreesdataTwo,GSdataTwo,colorhdataOne)
```

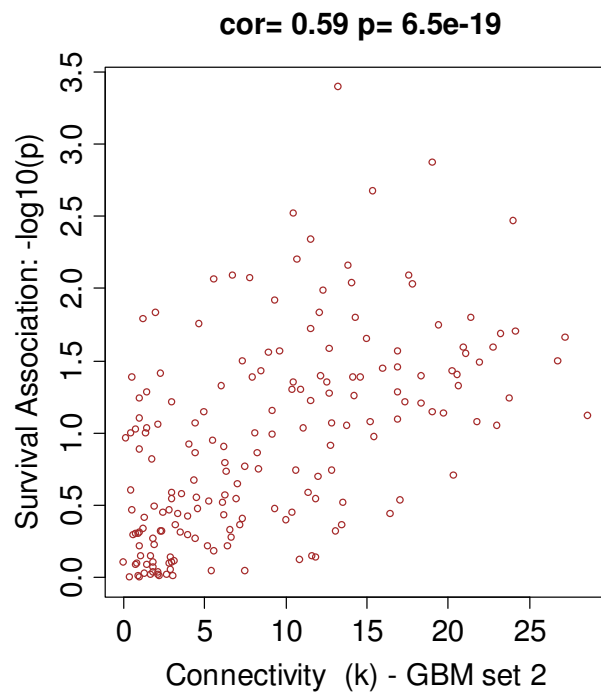
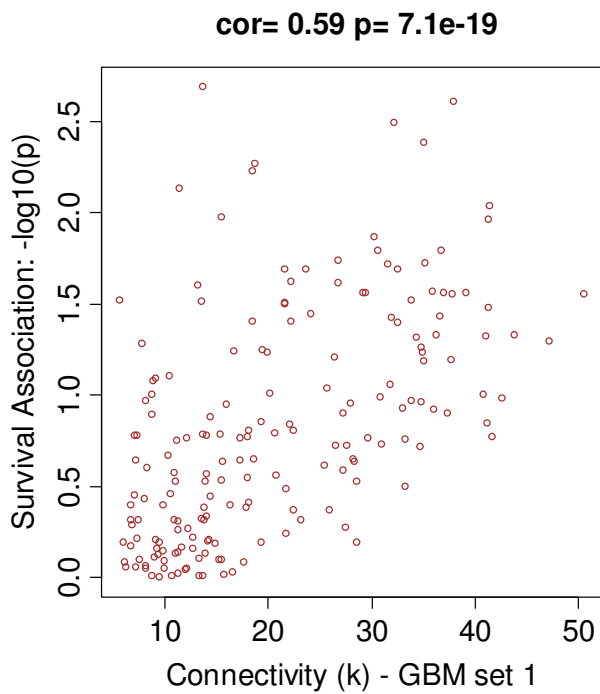
```
#Excerpt
couleur: brown
      variable NS.CorPval NS.cor kTotal kWithin kOut kDiff
kTotal kTotal 5.2e-12 0.480 1.00 0.85 0.60 0.49
kWithin kWithin 6.5e-19 0.590 0.85 1.00 0.20 0.84
kOut kOut 6.6e-01 0.032 0.60 0.20 1.00 -0.32
kDiff kDiff 3.9e-17 0.570 0.49 0.84 -0.32 1.00
```

```
# The following plots show the gene significance vs intramodular connectivity
# in the data set II
colorlevels=levels(factor(colorhdataOne))
par(mfrow=c(2,3))
for (i in c(1:length(colorlevels) ) ) {
  whichmodule=colorlevels[[i]];restrict1=colorhdataOne==whichmodule
  scatterplot1(AlldegreesdataTwo$kWithin[restrict1],
  GSdataTwo[restrict1],col1=colorhdataOne[restrict1],xlab1="Connectivity k, GBM set II",
  ylab1="Survival Association, -log(p)",title1=paste("set II,", whichmodule))
}
```

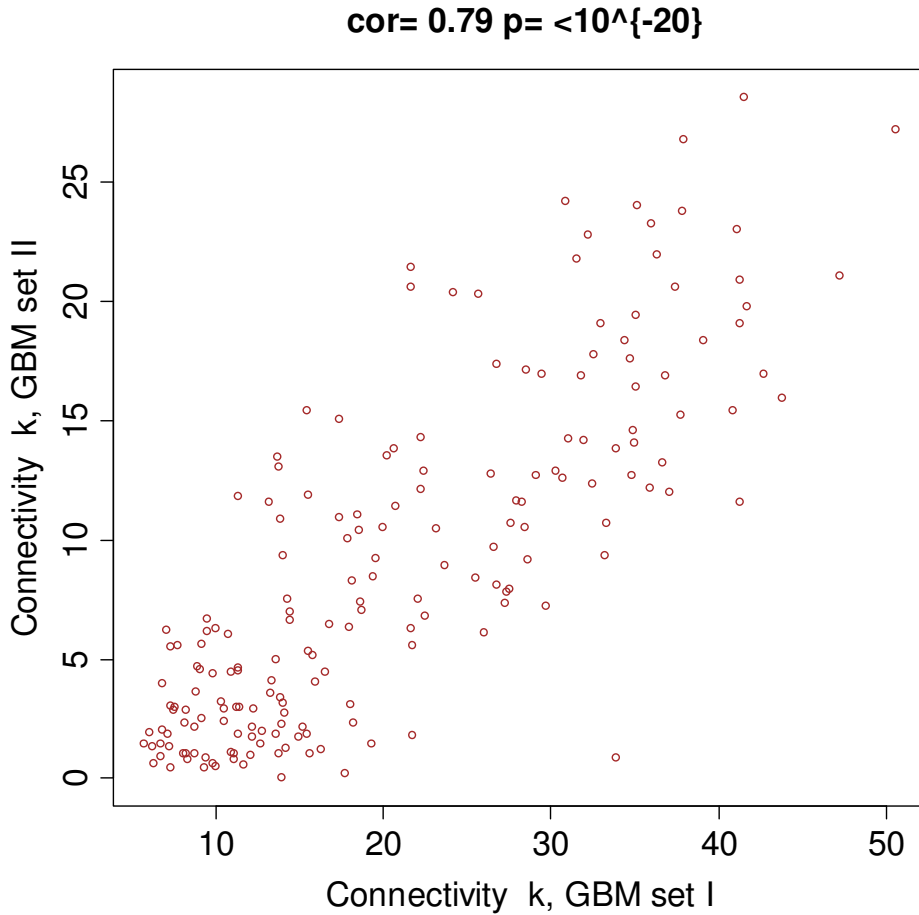


Now we create **Figure 2 a and b**) in the manuscript

```
par(mfrow=c(1,2))  
whichmodule="brown";restrict1=colorhdataOne==whichmodule  
scatterplot1(AlldegreesdataOne$kWithin[restrict1], GSdataOne[restrict1],  
col1=colorhdataOne[restrict1],xlab1="Connectivity (k) - GBM set 1",ylab1="Survival Association:  
-log10(p)",title1="")  
scatterplot1(AlldegreesdataTwo$kWithin[restrict1], GSdataTwo[restrict1],  
col1=colorhdataOne[restrict1],xlab1="Connectivity (k) - GBM set 2",ylab1="Survival  
Association: -log10(p)",title1="")
```



```
# Now we relate the connectivities between set I and II
whichmodule="brown";restrict1=colorhdataOne==whichmodule
scatterplot1(AlldegreesdataOne$kWithin[restrict1], AlldegreesdataTwo$kWithin[restrict1],
coll=colorhdataOne[restrict1],xlab1="Connectivity k, GBM set I",ylab1="Connectivity k, GBM
set II",title1="")
```



```
# This plot shows that the internal structure (as measured by intramodular #connectivity) of the
# brown module is largely preserved
# between the 2 data sets.
```

Module Eigengenes

This code yields the module eigengene for each module

```
PCdataOne=ModulePrinComps1(datExprdataOne[,restDegree],colorhdataOne)[[1]]
```

```
PCdataTwo=ModulePrinComps1(datExprdataTwo[,restDegree],colorhdataOne)[[1]]
```

Here we combine data sets I and II to arrive at a combined module eigengene

```
PCCombined=ModulePrinComps1(rbind(datExprdataOne[,restDegree],datExprdataTwo[,restDegree]),colorhdataOne)[[1]]
```

The following defines the module eigengene of the brown module

```
PCbrowndataOne=PCdataOne$PCbrown
```

```
PCbrowndataTwo=PCdataTwo$PCbrown
```

```
PCbrownCombined=PCCombined$PCbrown
```

#The following output

```
datSummary$Gene.Symbol[restDegree][colorhdataOne=="brown"]
```

the following lists the 10 most connected genes in the brown module.

```
topNumberHubs=10
```

```
datframe=data.frame(GeneSymbol=datSummary$Gene.Symbol[restDegree],AlldegreesdataOne)[colorhdataOne=="brown",]
```

```
datframe[rank(-datframe$kWithin)<= topNumberHubs,c(1,3)]
```

	GeneSymbol	kWithin
201292_at	TOP2A	50.60778
202870_s_at	CDC20	41.27654
203213_at	CDC2	42.69955
203358_s_at	EZH2	41.72014
206364_at	KIF14	41.30686
210052_s_at	TPX2	41.49522
218355_at	KIF4A	43.80523
218585_s_at	RAMP	41.24564
219918_s_at	ASPM	41.13956
222077_s_at	RACGAP1	47.23920

Quote

We must delight in each other; make others' condition our own; rejoice together; mourn together; labor and suffer together...always having before our eyes our commission and community in the work, as members of the same body...For we must consider that we shall be as a city upon a hill...The eyes of all people are upon us. From a 1630 Sermon by John Winthrop. Read at President Reagan's funeral.

```

# The following vectors contain the gene expression profiles of the top 10 hub genes
TOP2AdataOne= datExprdataOne[,datSummary$Gene.Symbol=="TOP2A"]
TOP2AdataTwo= datExprdataTwo[,datSummary$Gene.Symbol=="TOP2A"]
TOP2ACombined=c(TOP2AdataOne, TOP2AdataTwo)
RACGAP1dataOne= datExprdataOne[,datSummary$Gene.Symbol=="RACGAP1"]
RACGAP1dataTwo= datExprdataTwo[,datSummary$Gene.Symbol=="RACGAP1"]
RACGAP1Combined=c(RACGAP1dataOne, RACGAP1dataTwo)
KIF4AdataOne= datExprdataOne[,datSummary$Gene.Symbol=="KIF4A"]
KIF4AdataTwo= datExprdataTwo[,datSummary$Gene.Symbol=="KIF4A"]
KIF4ACombined=c(KIF4AdataOne, KIF4AdataTwo)
TPX2dataOne= datExprdataOne[,datSummary$Gene.Symbol=="TPX2"]
TPX2dataTwo= datExprdataTwo[,datSummary$Gene.Symbol=="TPX2"]
TPX2Combined=c(TPX2dataOne, TPX2dataTwo)
CDC2dataOne= datExprdataOne[,datSummary$Gene.Symbol=="CDC2"]
CDC2dataTwo= datExprdataTwo[,datSummary$Gene.Symbol=="CDC2"]
CDC2Combined=c(CDC2dataOne, CDC2dataTwo)
EZH2dataOne= datExprdataOne[,datSummary$Gene.Symbol=="EZH2"]
EZH2dataTwo= datExprdataTwo[,datSummary$Gene.Symbol=="EZH2"]
EZH2Combined=c(EZH2dataOne, EZH2dataTwo)
CDC20dataOne= datExprdataOne[,datSummary$Gene.Symbol=="CDC20"]
CDC20dataTwo= datExprdataTwo[,datSummary$Gene.Symbol=="CDC20"]
CDC20Combined=c(CDC20dataOne, CDC20dataTwo)
KIF14dataOne= datExprdataOne[,datSummary$Gene.Symbol=="KIF14"]
KIF14dataTwo= datExprdataTwo[,datSummary$Gene.Symbol=="KIF14"]
KIF14Combined=c(KIF14dataOne, KIF14dataTwo)
RAMPdataOne= datExprdataOne[,datSummary$Gene.Symbol=="RAMP"]
RAMPdataTwo= datExprdataTwo[,datSummary$Gene.Symbol=="RAMP"]
RAMPCombined=c(RAMPdataOne, RAMPdataTwo)
ASPMdataOne= datExprdataOne[,datSummary$Gene.Symbol=="ASPM"]
ASPMdataTwo= datExprdataTwo[,datSummary$Gene.Symbol=="ASPM"]
ASPMCombined=c(ASPMdataOne, ASPMdataTwo)

```

To get the connectivity of Ki67 use the following code

```
AlldegreesdataOne[datSummary$Gene.Symbol[restDegree]=="MKI67",]
```

```

kTotal  kWithin    kOut    kDiff
212022_s_at 23.44682 13.76298 9.68384 4.079146

```

#The following genes are standard markers of proliferation

```

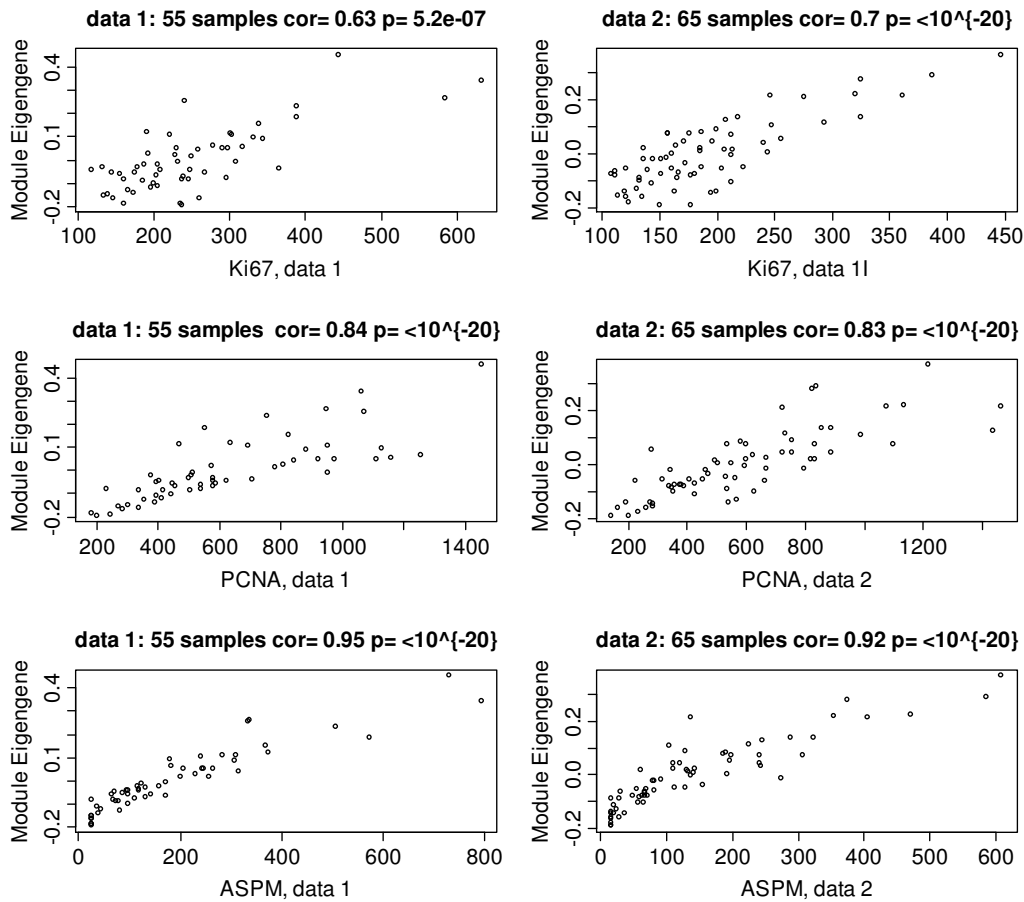
PCNAdataOne= datExprdataOne[,datSummary$Gene.Symbol=="PCNA"]
PCNAdataTwo= datExprdataTwo[,datSummary$Gene.Symbol=="PCNA"]
PCNACombined=c(PCNAdataOne,PCNAdataTwo)
MKI67dataOne= datExprdataOne[,datSummary$Gene.Symbol=="MKI67"]
MKI67dataTwo= datExprdataTwo[,datSummary$Gene.Symbol=="MKI67"]
MKI67Combined=c(MKI67dataOne,MKI67dataTwo)

```

```

#Now we relate the module eigengene to proliferation markers PCNA, KI67 and ASPM
par(mfrow=c(3,2))
scatterplot1(MKI67dataOne,PCbrowndataOne,xlab1="Ki67, data 1",ylab1="Module Eigengene",
title1="data 1: 55 samples")
scatterplot1(MKI67dataTwo,PCbrowndataTwo ,xlab1="Ki67, data 1I",ylab1="Module Eigengene",
title1="data 2: 65 samples")
scatterplot1(PCNAdataOne,PCbrowndataOne,xlab1="PCNA, data 1",ylab1="Module Eigengene",
title1="data 1: 55 samples ")
scatterplot1(PCNAdataTwo,PCbrowndataTwo ,xlab1="PCNA, data 2",ylab1="Module
Eigengene", title1="data 2: 65 samples")
scatterplot1(ASPMdataOne,PCbrowndataOne,xlab1="ASPM, data 1",ylab1="Module Eigengene",
title1="data 1: 55 samples")
scatterplot1(ASPMdataTwo,PCbrowndataTwo ,xlab1="ASPM, data 2",ylab1="Module
Eigengene", title1="data 2: 65 samples")

```



This is Supplementary Figure 1

#Quote

#If you only have a hammer, you tend to see every problem as a nail. – Maslow

#If you have an ax, every problem looks like hours of fun - unknown

Now we relate the marker expression and the module eigengene to the survival outcome in GBM

```
timedataOne=datClinicaldataOne$SurvivalTimeDays
dieddataOne=datClinicaldataOne$died
timedataTwo=datClinicaldataTwo$SurvivalTimeDays
dieddataTwo=datClinicaldataTwo$died
timeCombined=c(timedataOne,timedataTwo)
diedCombined= c(dieddataOne,dieddataTwo)
```

here we regress survival time on the module eigengene

```
summary(coxph(Surv(timeCombined,diedCombined)~PCbrownCombined))$waldtest
test      df      pvalue
6.37000000 1.00000000 0.01161113
```

#here we regress survival time on the top 10 hub genes

```
summary(coxph(Surv(timeCombined,diedCombined)~TOP2ACombined))$waldtest
test      df      pvalue
1.107000e+01 1.000000e+00 8.789649e-04
```

```
summary(coxph(Surv(timeCombined,diedCombined)~RACGAP1Combined))$waldtest
test      df      pvalue
9.340000000 1.000000000 0.002247980
```

```
summary(coxph(Surv(timeCombined,diedCombined)~KIF4ACombined))$waldtest
test      df      pvalue
8.780000000 1.000000000 0.003043759
```

```
summary(coxph(Surv(timeCombined,diedCombined)~TPX2Combined))$waldtest
test      df      pvalue
9.430000000 1.000000000 0.002136835
```

```
summary(coxph(Surv(timeCombined,diedCombined)~CDC2Combined))$waldtest
test      df      pvalue
7.210000000 1.000000000 0.007233725
```

```
summary(coxph(Surv(timeCombined,diedCombined)~EZH2Combined))$waldtest
test      df      pvalue
5.12000000 1.00000000 0.02369322
```

```
summary(coxph(Surv(timeCombined,diedCombined)~CDC20Combined))$waldtest
test      df      pvalue
8.860000000 1.000000000 0.002916241
```

```
summary(coxph(Surv(timeCombined,diedCombined)~KIF14Combined))$waldtest
test      df      pvalue
9.550000000 1.000000000 0.002003858
```

```
summary(coxph(Surv(timeCombined,diedCombined)~RAMPCombined))$waldtest
test      df      pvalue
5.94000000 1.00000000 0.01480329
```

```
summary(coxph(Surv(timeCombined,diedCombined)~ASPMCombined))$waldtest
test      df      pvalue
```

```
7.580000000 1.000000000 0.005889949
```

```
# here we regress survival time on the proliferation markers
```

```
summary(coxph(Surv(timeCombined, diedCombined)~PCNACombined))$waldtest  
test      df      pvalue  
5.36000000 1.00000000 0.02055703
```

```
summary(coxph(Surv(timeCombined, diedCombined)~MKI67Combined))$waldtest  
test      df      pvalue  
2.28000000 1.00000000 0.1309132
```

```
# Here is some code for exporting a summary of the data
```

```
datout=data.frame(datSummary[restDegree,], colorhdataOne,GSdataOne, AlldegreesdataOne,  
GSdataTwo, AlldegreesdataTwo)  
write.table(datout, "datSummaryRestDegree.csv", sep="," ,row.names=F)
```

Quote

I thought the following four rules would be enough, provided that I made a firm and constant resolution not to fail even once in the observance of them. The first was never to accept anything as true if I had not evident knowledge of its being so; that is, carefully to avoid precipitancy and prejudice, and to embrace in my judgment only what presented itself to my mind so clearly and distinctly that I had no occasion to doubt it. The second, to divide each problem I examined into as many parts as was feasible, and as was requisite for its better solution. The third, to direct my thoughts in an orderly way; beginning with the simplest objects, those most apt to be known, and ascending little by little, in steps as it were, to the knowledge of the most complex; and establishing an order in thought even when the objects had no natural priority one to another. And the last, to make throughout such complete enumerations and such general surveys that I might be sure of leaving nothing out. These long chains of perfectly simple and easy reasonings by means of which geometers are accustomed to carry out their most difficult demonstrations had led me to fancy that everything that can fall under human knowledge forms a similar sequence; and that so long as we avoid accepting as true what is not so, and always preserve the right order of deduction of one thing from another, there can be nothing too remote to be reached in the end, or to well hidden to be discovered.
Discours de la Méthode. 1637. Descartes, René (1596-1650)

```
# THE END
```