Network Learning

Our goal is to contruct gene co-expression networks of types WGCNA and Graphical Lasso [from the given gene expression data] and compare them.

```
library(igraph)
```

```
##
## Attaching package: 'igraph'

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union
```

```
library(proxy) # For similarity matrix
```

```
##
## Attaching package: 'proxy'

## The following objects are masked from 'package:stats':
##
##     as.dist, dist

## The following object is masked from 'package:base':
##
##     as.matrix
```

```
e <- read.delim("data/expr_ceph_utah_1000.txt", row.names=1)
head(t(e))[,1:5] # Rows are genes and columns are samples
```

```
##                 GM06986    GM12342    GM07016    GM07017     GM12146
## X207245_at   -2.7317300  0.6767469  0.1738114  0.3578004 -0.05128586
## X201215_at   -0.9004577 -0.9270311  0.6892607  0.8609533 -1.49116258
## X200602_at   -3.1137361 -3.1137361  1.0623280  0.2845590  0.34568798
## X202609_at   -0.8803472  1.6654487  0.0627374 -1.2088007  1.01824344
## X203828_s_at  0.9423669  0.3734892 -0.3160627 -0.3472103  0.08644669
## X213194_at    1.4307468  2.1298571  1.6143684 -0.2839742 -0.57423760
```

Weighted Gene Co-Expression Network Analysis (WGCNA)

The process flow for creating a WGCNA is straightforward. Starting with a **gene expression matrix**, we create a **similarity matrix** by carrying out a similarity measure selection between genes i and j. From the similiarity matrix, we construct an **adjacency matrix** using a threshold measure *tau*. We then plot the graph from the adjacency matrix.

We will use absolute pearson correlation as our measure of similarity between genes i and j, that is $S(i,j) = |corr(i,j)|$. The similarity matrix S is a symmetrix matrix with values in range [0,1] and a unit diagonal.

```
s <- abs(as.matrix(simil(t(e)), FUN=NULL, diag=1)) # Similarity matrix
s[1:5,1:5]
```

```
##              X207245_at X201215_at X200602_at X202609_at X203828_s_at
## X207245_at   1.00000000 0.04304491 0.12104940 0.10970442   0.04130138
## X201215_at   0.04304491 1.00000000 0.10109481 0.17051288   0.09339149
## X200602_at   0.12104940 0.10109481 1.00000000 0.06033583   0.13198330
## X202609_at   0.10970442 0.17051288 0.06033583 1.00000000   0.01280272
## X203828_s_at 0.04130138 0.09339149 0.13198330 0.01280272   1.00000000
```

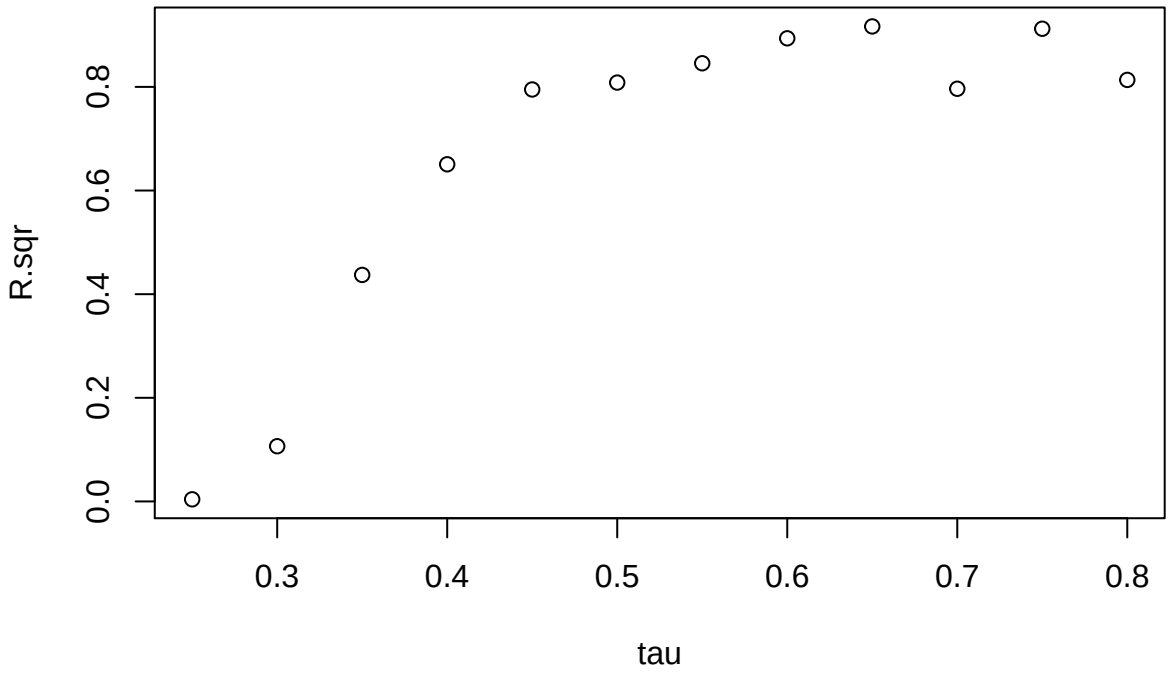
Next, we perform "soft" thresholding via scale-free topology *à la* Zhang and Horvath 2005; since gene co-expression networks have been found to at least approximately display scale-free topology, so thresholding values that do not give rise to networks displaying scale-free topology should not be considered.

```
sft.power.fit <- function(x) {
  a <- s * (s >= x) # Adjacency matrix with threshold x
  g <- graph.adjacency(a, mode="undirected", weighted=TRUE, diag=TRUE) # Construct WGC network

  # Degree of each vertex, not counting self-loops
  d <- degree(g, v=V(g), loops=FALSE)
  # Continuous distribution for linear model fitting
  deg <- 1:max(d)

  # Proportion of nodes with degree d
  d.dist <- degree.distribution(g, v=V(g), loops=FALSE, cumulative=FALSE)
  # Only account for frequencies of non-zero degree vertices
  d.dist <- d.dist[2:length(d.dist)]

  # Remove all zero frequency positions
  non.zero.positions <- which( d.dist!=0 )
  deg <- deg[non.zero.positions]
  d.dist <- d.dist[non.zero.positions]

  # Linear model fitting
  SFT.model <- lm(log(d.dist) ~ log(deg))
  SFT.R.sqr <- summary(SFT.model)$r.squared
  SFT.alpha <- -coef(SFT.model)[[2]]
  return(SFT.R.sqr)
}

tau <- c(0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8)
R.sqr <- sapply(tau, sft.power.fit)
cbind(tau, R.sqr) # Display tau vs R-squared values
```

```
##        tau       R.sqr
##  [1,] 0.25 0.004106849
##  [2,] 0.30 0.106532695
##  [3,] 0.35 0.437114635
##  [4,] 0.40 0.650661556
##  [5,] 0.45 0.794968940
##  [6,] 0.50 0.808327761
##  [7,] 0.55 0.845574982
```

```
##  [8,] 0.60 0.893740743
##  [9,] 0.65 0.916673300
## [10,] 0.70 0.796389129
## [11,] 0.75 0.912218243
## [12,] 0.80 0.813450199
```

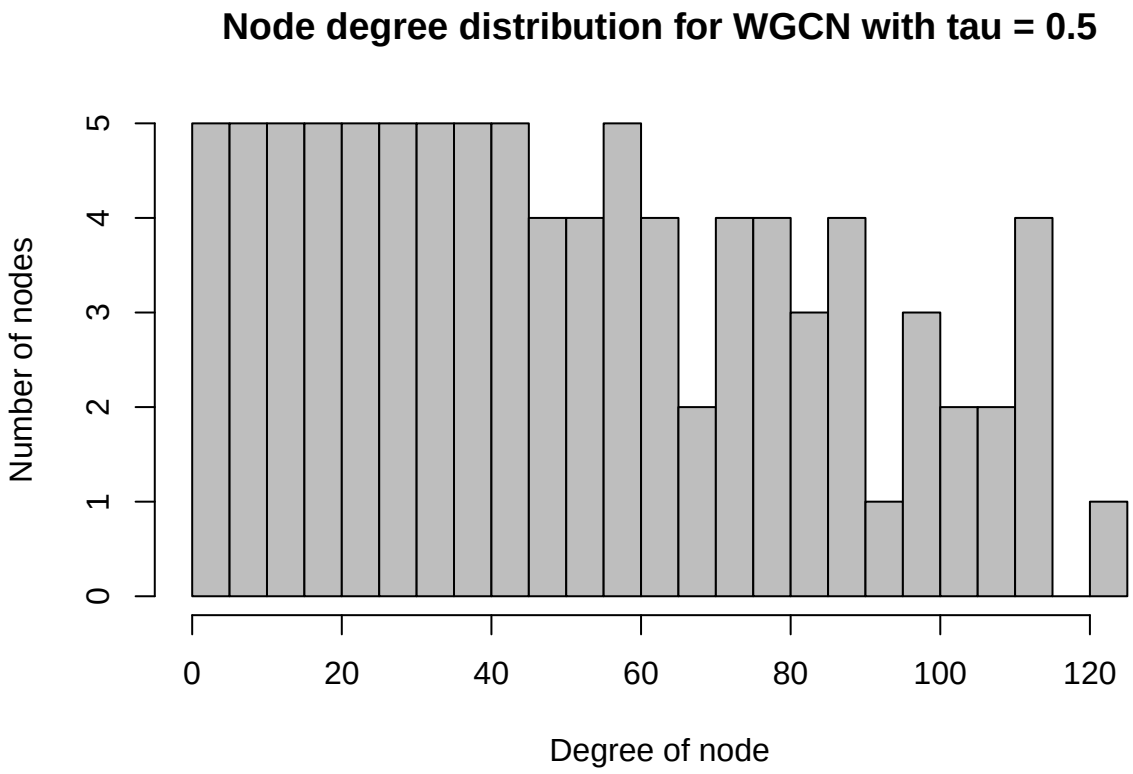
```r
plot(x=tau, y=R.sqr) # Scatter plot
```



The smallest *tau* that produces R-squared greater than or equal to 0.8 is $\boldsymbol{tau = 0.5}$.

Let us plot the node degree distribution plot for our weighted gene co-expression network with $\boldsymbol{tau = 0.5}$.

```r
# Histogram plot of node degrees in wgcna_net
hist(x=deg, breaks=length(deg)/4,
     xlab="Degree of node", ylab="Number of nodes",
     main="Node degree distribution for WGCN with tau = 0.5", col="gray")
```



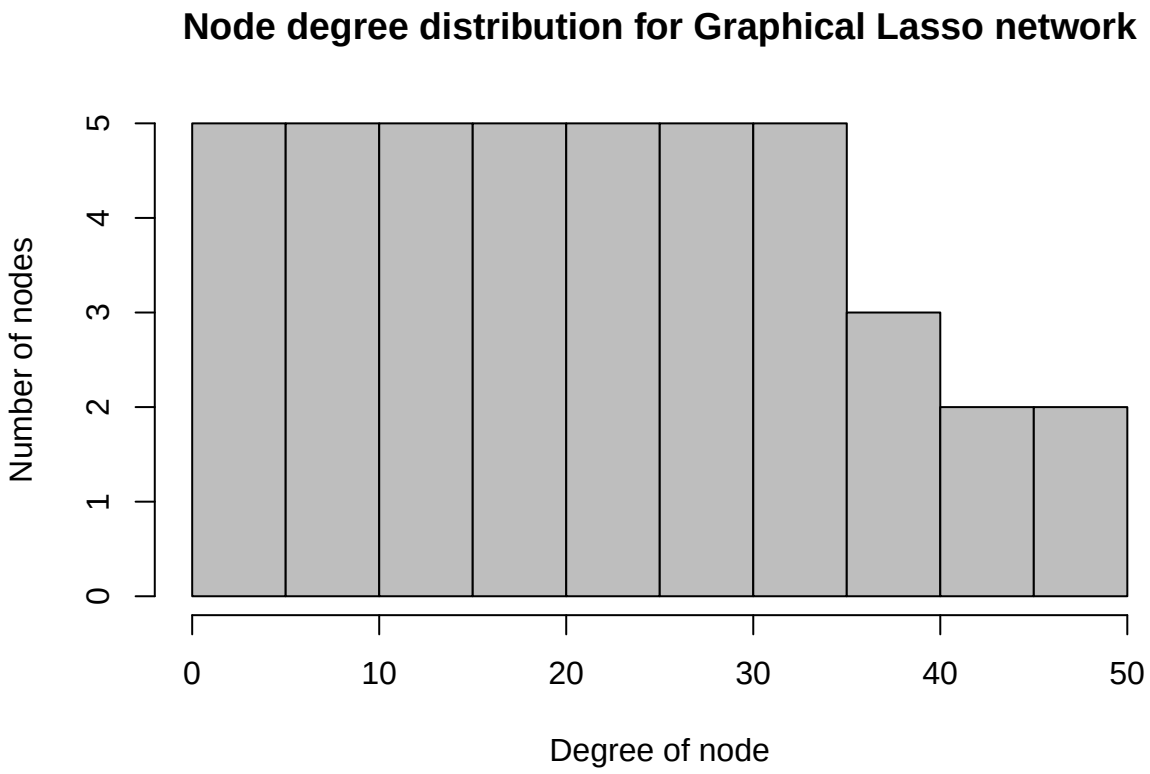
Graphical Lasso

```
c <- cov(e) # Sample covariance matrix
c[1:5,1:5]
```

```
##              X207245_at  X201215_at  X200602_at  X202609_at X203828_s_at
## X207245_at   1.00000000  0.04304491  0.12104940  0.10970442   0.04130138
## X201215_at   0.04304491  1.00000000  0.10109481 -0.17051288   0.09339149
## X200602_at   0.12104940  0.10109481  1.00000000 -0.06033583   0.13198330
## X202609_at   0.10970442 -0.17051288 -0.06033583  1.00000000   0.01280272
## X203828_s_at 0.04130138  0.09339149  0.13198330  0.01280272   1.00000000
```

```
library(glasso)

p <- glasso(c, rho=0.55, thr=1e-4, penalize.diagonal=FALSE) # Precision matrix
colnames(p$wi) <- colnames(e)
rownames(p$wi) <- colnames(e)
glasso_net <- graph.adjacency(p$wi, mode="undirected", weighted=TRUE, diag=TRUE) # Construct glasso net
```

```
# Histogram plot of node degrees in glasso_net
hist(x=deg, breaks=length(deg)/4, freq=TRUE,
     xlab="Degree of node", ylab="Number of nodes",
     main="Node degree distribution for Graphical Lasso network", col="gray")
```



Comparing WGCNA network and Graphical Lasso network

```
gsize(glasso_net) - ncol(e) # Edge count for glasso_net, discounting self-loops
```

```
## [1] 2769
```

```
gsize(wgcna_net) - ncol(e) # Edge count for wgcna_net, discounting self-loops
```

```
## [1] 7690
```

```
# Number of edges common to both networks, discounting self-loops
length(intersection(E(glasso_net), E(wgcna_net))) - ncol(e)
```

```
## [1] 2769
```

All edges present in glasso_net are present in wgcna_net, that is, glasso_net is a subset of wgcna_net. However, the revserse is not true: wgcna_net has some extra edges.

The proportion of edges present in wgcna_net also present in glasso_net is *2769/7690 = 0.360078*. The proportion of edges present in glasso_net also present in wgcna_net is *2769/2769 = 1.0*.

Ideally, we want to separate indirection relationships from direct relationships. Graphical Lasso does so [with the regularization term, encouraging sparsity], but WGCNA does not.