INFX 576: Problem Set 5 - Cohesive Groups and Graph Sets*

Avanti Chande

Due: Thursday, February 16, 2017

Collaborators: Jay Chauhan, Gosuddin Siddiqi

Instructions:

Before beginning this assignment, please ensure you have access to R and RStudio.

1. Download the `problemset5.Rmd` file from Canvas. You will also need the data from last week's Problem Set 4 in `problemset4_data.Rdata`.
2. Replace the "Insert Your Name Here" text in the `author:` field with your own full name. Any collaborators must be listed on the top of your assignment.
3. Be sure to include well-documented (e.g. commented) code chucks, figures and clearly written text chunk explanations as necessary. Any figures should be clearly labeled and appropriately referenced within the text.
4. Collaboration on problem sets is acceptable, and even encouraged, but each student must turn in an individual write-up in his or her own words and his or her own work. The names of all collaborators must be listed on each assignment. Do not copy-and-paste from other students' responses or code.
5. When you have completed the assignment and have **checked** that your code both runs in the Console and knits correctly when you click `Knit PDF`, rename the R Markdown file to `YourLastName_YourFirstName_ps5.Rmd`, knit a PDF and submit the PDF file on Canvas.

Setup:

In this problem set you will need, at minimum, the following R packages.

```
# Load standard libraries
library(statnet)
```

```
## Warning: package 'statnet' was built under R version 3.2.5
## Warning: package 'tergm' was built under R version 3.2.5
## Warning: package 'statnet.common' was built under R version 3.2.5
## Warning: package 'ergm' was built under R version 3.2.5
## Warning: package 'network' was built under R version 3.2.5
## Warning: package 'networkDynamic' was built under R version 3.2.5
## Warning: package 'ergm.count' was built under R version 3.2.5
## Warning: package 'sna' was built under R version 3.2.5
```

```
load("problemset5_data.Rdata")
```

*Problems originally written by C.T. Butts (2009)

Problem 1: Cohesive Subgroups

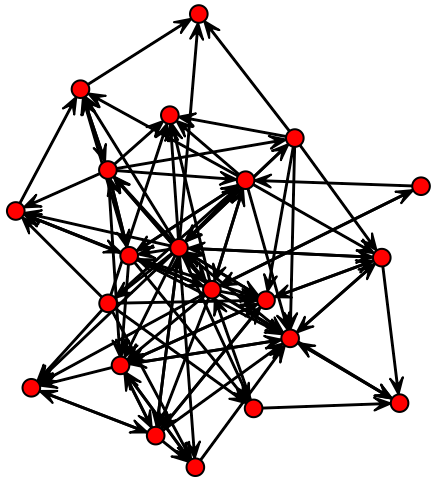
In this problem we use data collected by Krackhardt (1987), `kfr` capturing self-reported friendship ties among 21 managers in a high-tech firm. This data is directed and unvalued, it is possible for i to nominate j as a friend without reciprocation.

(a) Cliques

Using the `clique.census` command, perform the following analyses on `kfr`:

- Obtain a length-tabulation of clique membership by vertex.

```
#Plotting the data
gplot(kfr)
```



```
#Clique Census
kfr1<-clique.census(kfr,tabulate.by.vertex = TRUE)

#Length Tabulation
kfr1$clique.count
```

```
##   Agg v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12 v13 v14 v15 v16 v17 v18 v19
## 1   4  0  0  0  0  0  0  1  0  1   1   0   0   0   0   0   0   0   0   0
## 2   9  2  3  1  1  0  1  0  1  0   0   1   0   1   1   1   1   1   1   1
## 3   6  1  0  0  2  2  0  0  0  0   0   3   3   0   0   1   0   3   0   2
##   v20 v21
## 1   1   0
## 2   0   1
```

```
## 3   0   1
```

```
names(kfr1)
```

```
## [1] "clique.count" "cliques"
```

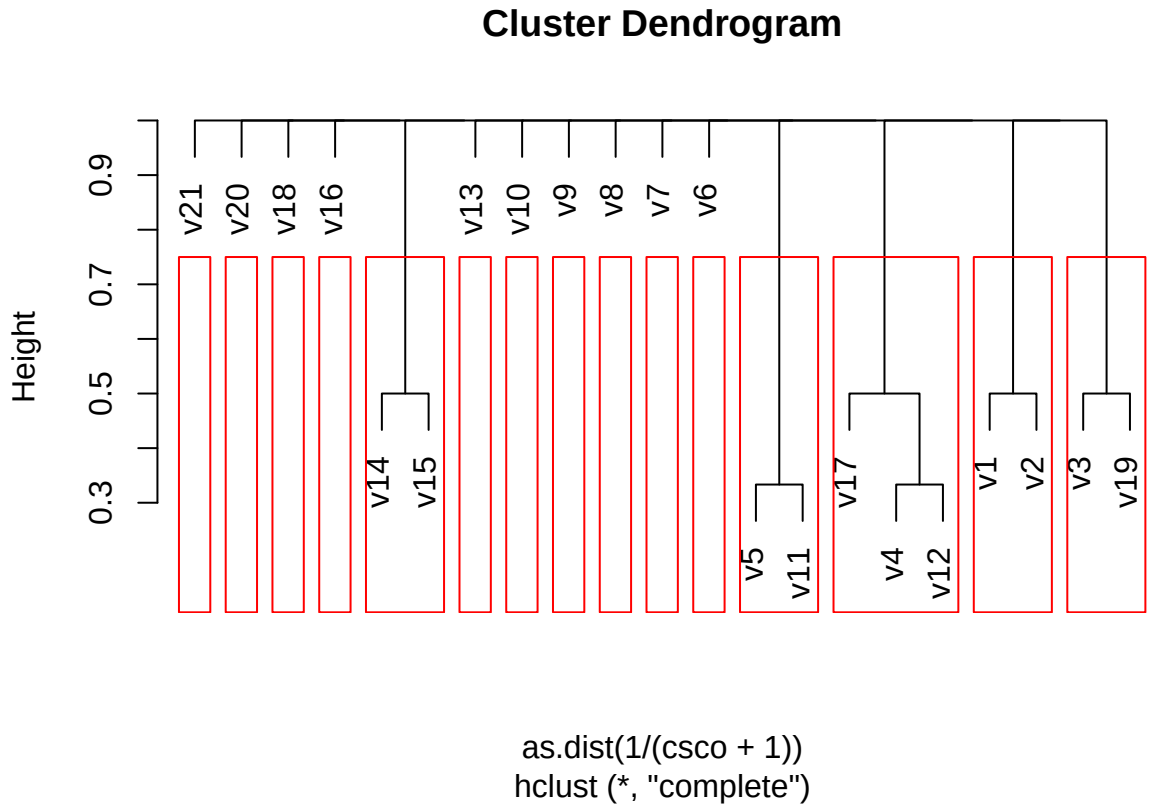
- Obtain the combined clique co-membership matrix.

```
#Combined Clique co-membership matrix
kfr2<- clique.census(kfr,clique.comembership="sum")
kfr2$clique.comemb
```

```
##     v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12 v13 v14 v15 v16 v17 v18 v19 v20
## v1   3  1  0  1  0  0  0  0  0   0   0   1   0   0   0   1   0   0   0   0
## v2   1  3  0  0  0  0  0  0  0   0   0   0   0   0   0   0   0   1   0   0
## v3   0  0  1  0  0  0  0  0  0   0   0   0   0   0   0   0   0   0   1   0
## v4   1  0  0  3  0  0  0  1  0   0   0   2   0   0   0   0   1   0   0   0
## v5   0  0  0  0  2  0  0  0  0   0   2   0   0   0   0   0   1   0   1   0
## v6   0  0  0  0  0  1  0  0  0   0   0   0   0   0   0   0   1   0   0   0
## v7   0  0  0  0  0  0  1  0  0   0   0   0   0   0   0   0   0   0   0   0
## v8   0  0  0  1  0  0  0  1  0   0   0   0   0   0   0   0   0   0   0   0
## v9   0  0  0  0  0  0  0  0  1   0   0   0   0   0   0   0   0   0   0   0
## v10  0  0  0  0  0  0  0  0  0   1   0   0   0   0   0   0   0   0   0   0
## v11  0  0  0  0  2  0  0  0  0   0   4   0   1   0   1   0   1   0   2   0
## v12  1  0  0  2  0  0  0  0  0   0   0   3   0   0   0   0   2   0   0   0
## v13  0  0  0  0  0  0  0  0  0   0   1   0   1   0   0   0   0   0   0   0
## v14  0  0  0  0  0  0  0  0  0   0   0   0   0   1   1   0   0   0   0   0
## v15  0  0  0  0  0  0  0  0  0   0   1   0   0   1   2   0   0   0   1   0
## v16  1  0  0  0  0  0  0  0  0   0   0   0   0   0   0   1   0   0   0   0
## v17  0  0  0  1  1  1  0  0  0   0   1   2   0   0   0   0   4   0   0   0
## v18  0  1  0  0  0  0  0  0  0   0   0   0   0   0   0   0   0   1   0   0
## v19  0  0  1  0  1  0  0  0  0   0   2   0   0   0   1   0   0   0   3   0
## v20  0  0  0  0  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0   1
## v21  0  1  0  0  0  0  0  0  0   0   0   1   0   0   0   0   1   0   0   0
##     v21
## v1    0
## v2    1
## v3    0
## v4    0
## v5    0
## v6    0
## v7    0
## v8    0
## v9    0
## v10   0
## v11   0
## v12   1
## v13   0
## v14   0
## v15   0
## v16   0
## v17   1
## v18   0
## v19   0
## v20   0
## v21   2
```

- Use the clique co-membership matrix to obtain a cohesion-based blockmodel of `kfr`. You may find the commands `hclust` `cutree` and `blockmodel` helpful here. Show the dendrogram (with cutoff value), block image matrix, and block image.

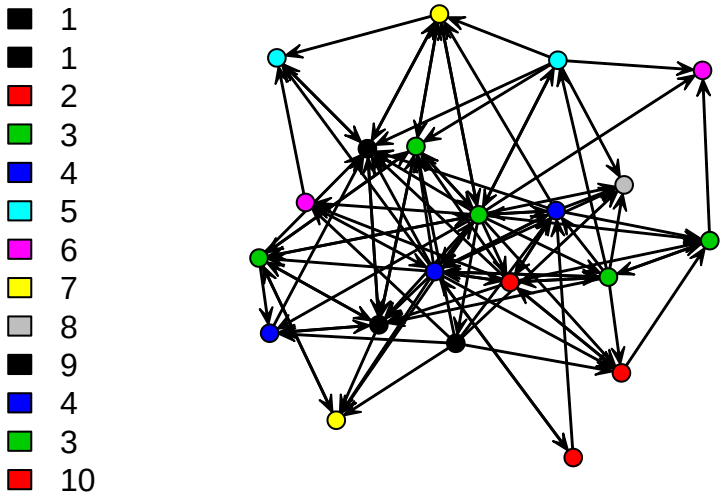
```
#Dendograms
csco<-clique.census(kfr,clique.comembership="sum")$clique.comemb
kfr3<-hclust(as.dist(1/(csco+1)))
plot(kfr3)
rect.hclust(kfr3,h=0.8)                                    # Plot a cutoff point
```



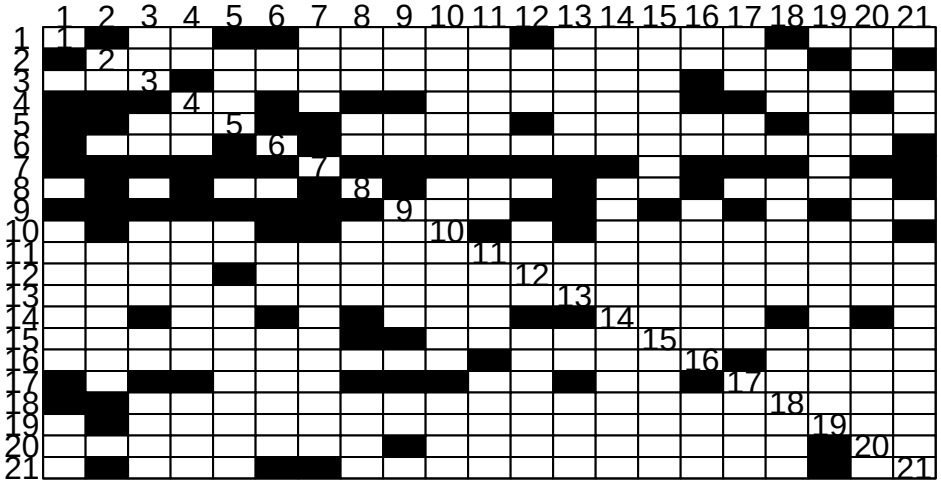

```
ct<-cutree(kfr3,h=0.8)                                     # Cut the clusters
class(ct)
```

```
## [1] "integer"
```

```
gplot(kfr,vertex.col=ct)                                   # Visualize directly
legend("topleft",fill=(ct),legend=ct,bty="n")              # Adding Legend
```




```
plot.sociomatrix(kfr[order(ct),order(ct)])          #Plot the Sociomatrix
```

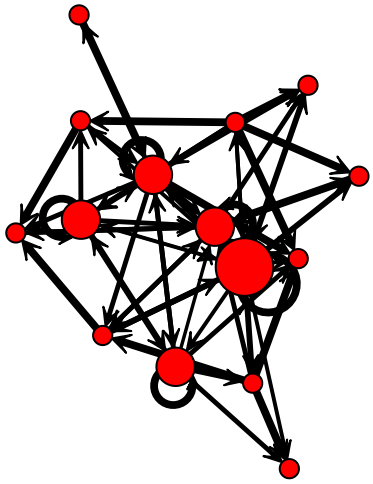


```
#Blockmodel
bm<-blockmodel(kfr,ct)
bm
```

```
##
## Network Blockmodel:
##
## Block membership:
##
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21
##  1  1  2  3  4  5  6  7  8  9  4  3 10 11 11 12  3 13  2 14 15
##
## Reduced form blockmodel:
##
##   1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
##            Block 1   Block 2   Block 3   Block 4   Block 5   Block 6
## Block 1  1.0000000 0.0000000 0.3333333 0.0000000 0.0000000 0.0000000
## Block 2  0.5000000 1.0000000 0.1666667 0.5000000 0.0000000 0.0000000
## Block 3  0.8333333 0.3333333 1.0000000 0.3333333 0.3333333 0.3333333
## Block 4  0.7500000 0.7500000 0.6666667 1.0000000 0.0000000 0.0000000
## Block 5  0.5000000 0.0000000 0.6666667 0.0000000       NaN 1.0000000
## Block 6  0.0000000 0.0000000 0.0000000 0.0000000 0.0000000       NaN
## Block 7  0.0000000 0.0000000 0.3333333 0.0000000 0.0000000 0.0000000
## Block 8  0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
## Block 9  0.0000000 0.5000000 0.3333333 0.5000000 0.0000000 0.0000000
## Block 10 0.0000000 0.0000000 0.0000000 1.0000000 0.0000000 0.0000000
## Block 11 0.2500000 0.5000000 0.0000000 0.5000000 0.5000000 0.5000000
```

```
## Block 12 1.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
## Block 13 0.5000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
## Block 14 0.0000000 0.0000000 0.0000000 0.5000000 0.0000000 0.0000000
## Block 15 0.5000000 0.0000000 0.6666667 0.0000000 0.0000000 0.0000000
##            Block 7   Block 8   Block 9 Block 10  Block 11  Block 12
## Block 1  0.5000000 0.0000000 0.0000000      0.0 0.0000000 0.5000000
## Block 2  0.0000000 0.0000000 0.0000000      0.0 0.7500000 0.0000000
## Block 3  0.6666667 0.3333333 0.3333333      0.0 0.3333333 0.6666667
## Block 4  0.5000000 1.0000000 0.0000000      0.5 0.5000000 0.0000000
## Block 5  0.0000000 1.0000000 0.0000000      0.0 0.0000000 0.0000000
## Block 6  0.0000000 0.0000000 0.0000000      0.0 0.0000000 0.0000000
## Block 7        NaN 0.0000000 0.0000000      0.0 0.0000000 0.0000000
## Block 8  0.0000000       NaN 0.0000000      0.0 0.0000000 0.0000000
## Block 9  1.0000000 1.0000000       NaN      0.0 0.0000000 1.0000000
## Block 10 0.0000000 0.0000000 0.0000000      NaN 0.0000000 0.0000000
## Block 11 0.0000000 0.5000000 0.0000000      0.0 1.0000000 0.0000000
## Block 12 0.0000000 0.0000000 0.0000000      0.0 0.0000000       NaN
## Block 13 0.0000000 0.0000000 0.0000000      0.0 0.0000000 0.0000000
## Block 14 0.0000000 0.0000000 0.0000000      0.0 0.0000000 0.0000000
## Block 15 0.0000000 0.0000000 0.0000000      0.0 0.0000000 0.0000000
##          Block 13  Block 14  Block 15
## Block 1       0.5 0.0000000 0.5000000
## Block 2       0.0 0.5000000 0.0000000
## Block 3       0.0 0.3333333 0.6666667
## Block 4       0.5 0.0000000 0.5000000
## Block 5       0.0 0.0000000 1.0000000
## Block 6       0.0 0.0000000 0.0000000
## Block 7       0.0 0.0000000 0.0000000
## Block 8       0.0 0.0000000 0.0000000
## Block 9       0.0 1.0000000 0.0000000
## Block 10      0.0 0.0000000 0.0000000
## Block 11      0.0 0.0000000 0.0000000
## Block 12      0.0 0.0000000 0.0000000
## Block 13      NaN 0.0000000 0.0000000
## Block 14      1.0       NaN 0.0000000
## Block 15      1.0 0.0000000       NaN
```

```
#Plot the Blockmodel
gplot(bm$block.model,vertex.cex=table(ct),
      edge.lwd=6*bm$block.model, usearrows=TRUE,
  diag=TRUE)
```



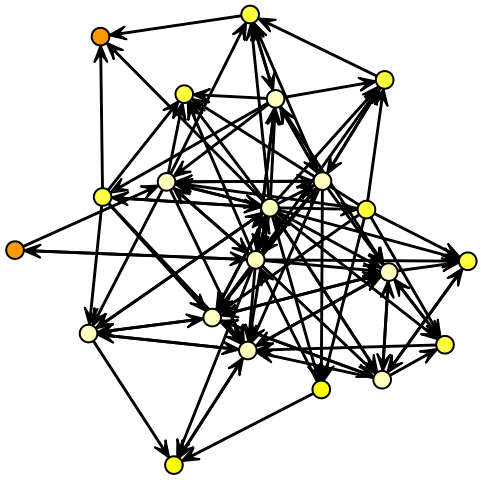
(b) K-Cores

Use the `kcores` command to calculate the total degree k-cores of `kfr`. Visualize the network, indicating by size, shape, or color the core number for each vertex.

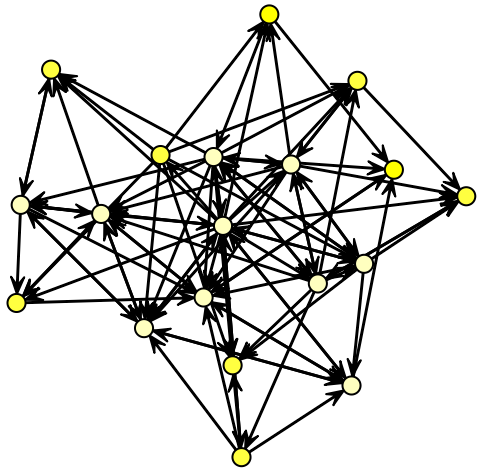
```
kfr4 <- kcores(kfr)

#Tabulating the Output
table(kfr4)
```

```
## kfr4
##  3  5  6  7
##  2  2  7 10
```

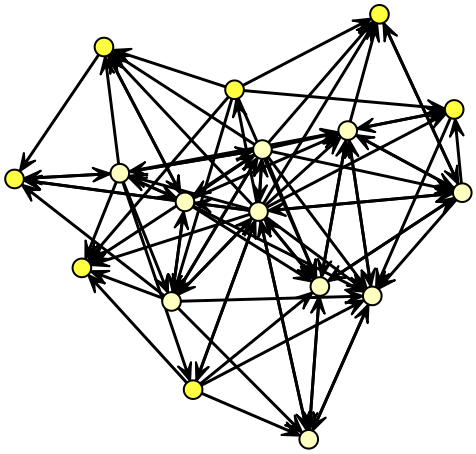
```
#There are 3, 5, 6 and 7 k-cores. Plotting them below.
gplot(kfr,vertex.col=heat.colors(max(kfr4)+1)[kfr4+1])
title(main = "k-core KFR Network")
```



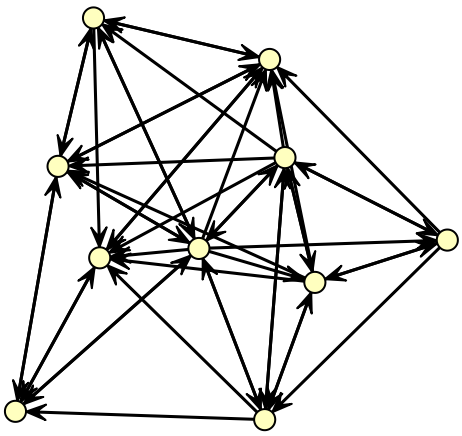

```
# Plotting 5-core network
gplot(kfr[kfr4>3,kfr4>3],vertex.col=heat.colors(max(kfr4[kfr4>3])+1)[kfr4[kfr4>3]+1])
title(main = "5-core KFR Network")
```




```
#Plotting 6-core network
gplot(kfr[kfr4>5,kfr4>5],vertex.col=heat.colors(max(kfr4[kfr4>5])+1)[kfr4[kfr4>5]+1])
title(main = "6-core KFR Network")
```




```
#Plotting 7-core network
gplot(kfr[kfr4>6,kfr4>6],vertex.col=heat.colors(max(kfr4[kfr4>6])+1)[kfr4[kfr4>6]+1])
title(main = "7-core KFR Network")
```



(c) Discussion

Based on the anlaysis in parts (a) and (b), how would you summarize the structure of this network; in particular, how many distinct dense clusters do there appear to be?

Ans. Based on clique census, dendograms and blockmodel, there don't seem to be any dense clusters in this network. There are different colors in the plot for each vertex which proves that there are no particularly dense clusters.

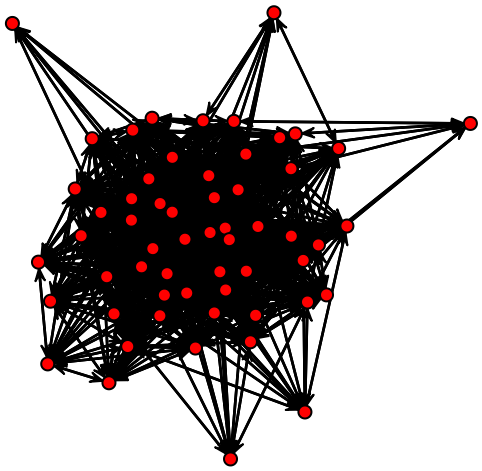
Problem 2: Graph Correlation

Last week, we saw network data from the famous Bernard, Killworth, and Sailer (BKS) studies. These studies examined the issue of accuracy in self-reported network data. Each study involved a group of subjects, each of whom was asked to rank-order all other group members by frequency of interaction. The self-reported interaction frequency was referred to as the "cognitive" ntework by BKS (i.e. the network as understood by the subjects themeselves). During the study period, behavioral information on interaction within the same groups was also collected via trained observers. The network of observed pairwise interaction frequencies was referred to as the "behavioral" network. Accuracy was assessed by comparing the "cognitive" and "behavioral" networks. The BKS studies were controversial and launched a much larger literature on the accuracy of network measurement.

(a) Comparing Networks

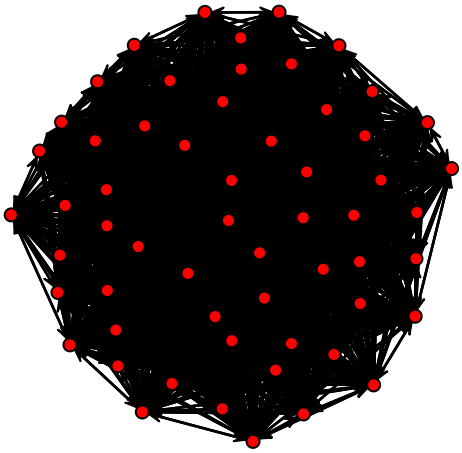
For each of the data objects `bkfrat`, `bkham`, `bkoff` and `bktec` (each itself a list containing the cognitive and behavioral network from a BKS study) perform a QAP test of the correlation between the self-report and the observed structure. Show in each case the test results, including a plot of the QAP replicates.

```
#BKFRAT Networks
gplot(bkfrat$Behavioral)
title(main = "BKFRAT Behavioral")
```




```
gplot(bkfrat$Cognitive)
title(main = "BKFRAT Cognitive")
```

BKFRAT Cognitive



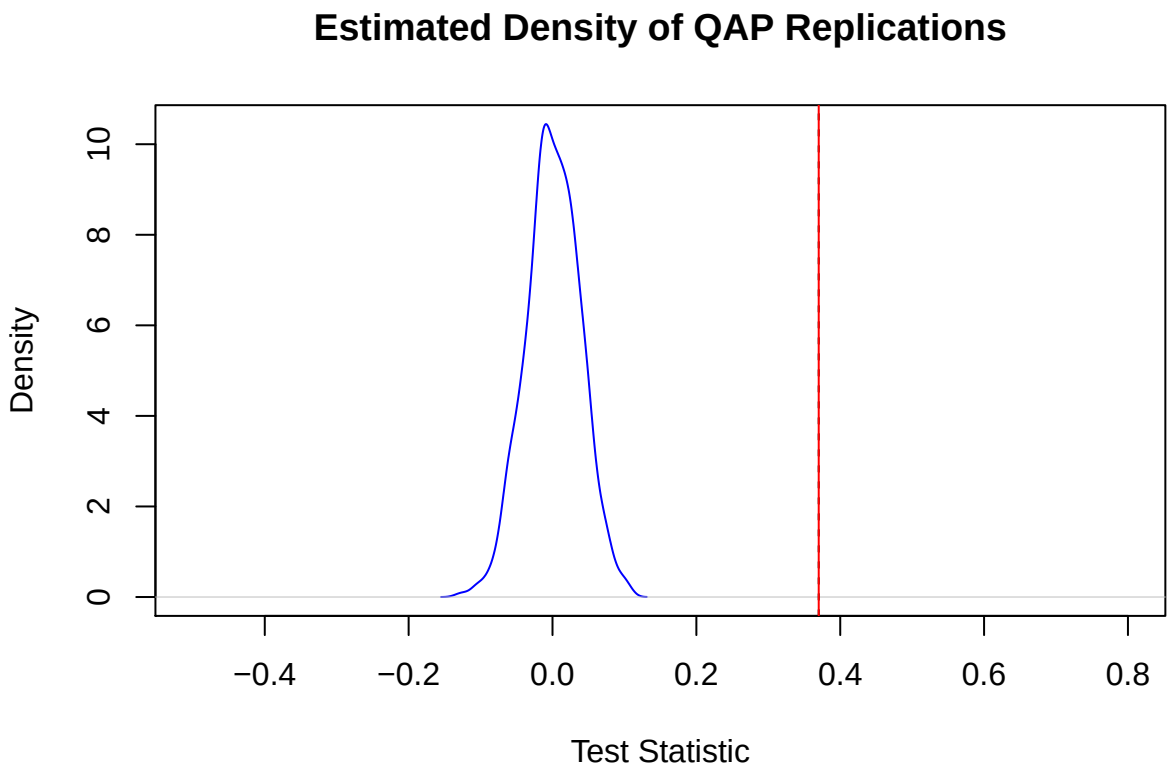
```
#Correlation
bkfrat_cor <- gcor(bkfrat$Cognitive, bkfrat$Behavioral)
bkfrat_cor
```

```
## [1] 0.3700903
```

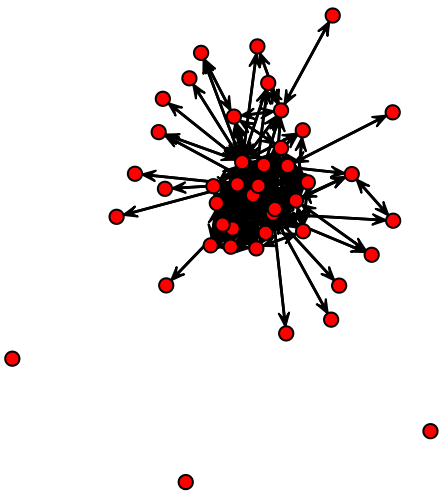
```
# qaptest routine
bkfrat_qt<-qaptest(list(bkfrat$Cognitive, bkfrat$Behavioral),gcor,g1=1,g2=2)
summary(bkfrat_qt)                                          # Examine the results
```

```
##
## QAP Test Results
##
## Estimated p-values:
##  p(f(perm) >= f(d)): 0
##  p(f(perm) <= f(d)): 1
##
## Test Diagnostics:
##  Test Value (f(d)): 0.3700903
##  Replications: 1000
##  Distribution Summary:
##       Min:     -0.1292991
##       1stQ:    -0.02308707
##       Med:     0.00120441
##       Mean:    0.0009938215
##       3rdQ:    0.02738752
##       Max:     0.1054124
```

```
#Comparing with QAP Permutation
plot(bkfrat_qt, xlim=c(-0.5,0.8), col="blue")
abline(v=bkfrat_qt$testval, col="red")
```

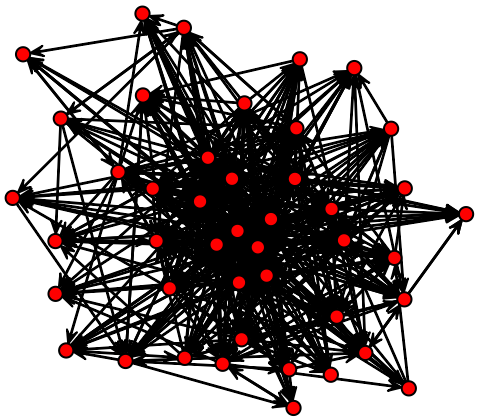



```
#BKHAM
gplot(bkham$Behavioral)
title(main = "BKHAM Behavioral")
```




```
gplot(bkham$Cognitive)
title(main = "BKHAM Cognitive")
```

BKHAM Cognitive



```
#Correlation
bkham_cor <- gcor(bkham$Cognitive, bkham$Behavioral)

bkham_cor
```

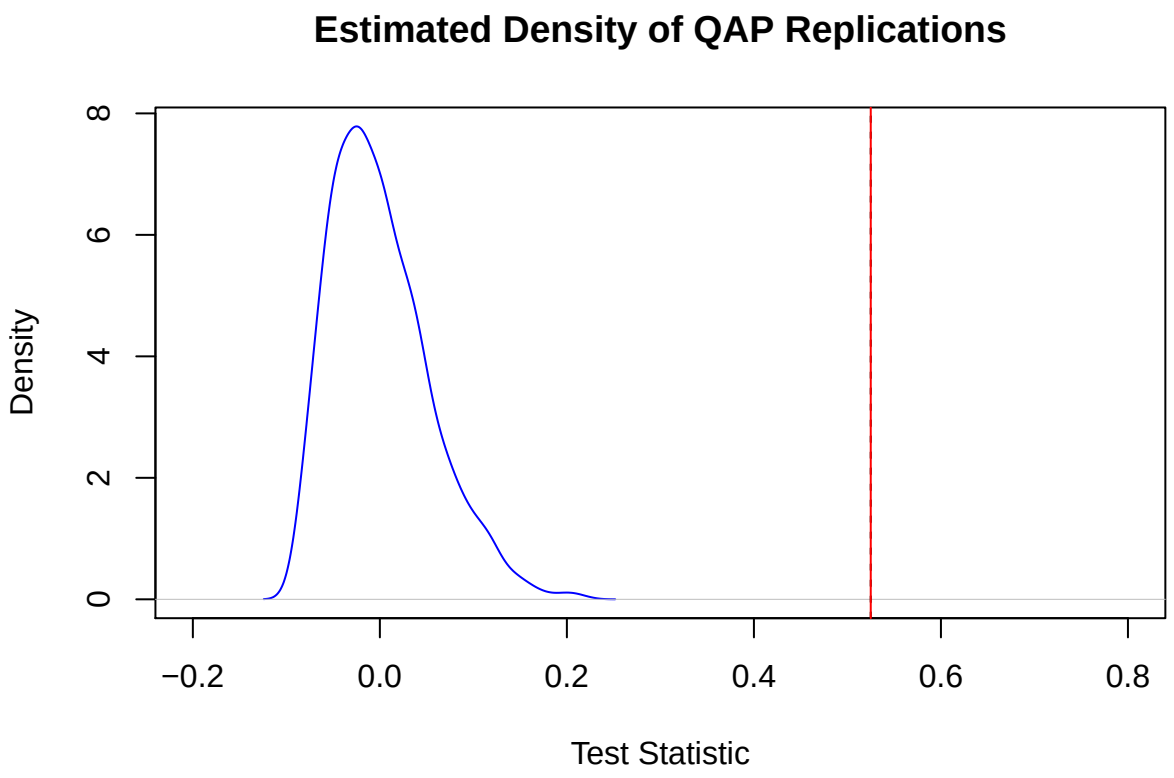
```
## [1] 0.5249309
```

```
# qaptest routine
bkham_qt<-qaptest(list(bkham$Cognitive, bkham$Behavioral),gcor,g1=1,g2=2)
summary(bkham_qt)
```

```
##
## QAP Test Results
##
## Estimated p-values:
##  p(f(perm) >= f(d)): 0
##  p(f(perm) <= f(d)): 1
##
## Test Diagnostics:
##  Test Value (f(d)): 0.5249309
##  Replications: 1000
##  Distribution Summary:
##      Min:     -0.08818271
##      1stQ:    -0.04001332
##      Med:     -0.00698033
##      Mean:    0.0006794055
##      3rdQ:    0.03235207
```

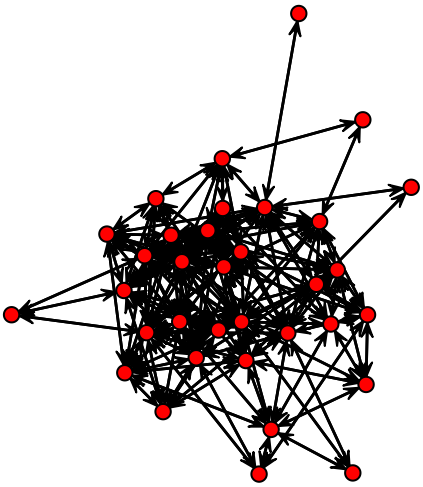
```
##       Max:      0.215703
```

```
plot(bkham_qt, xlim=c(-0.2,0.8), col="blue")

#Comparing with QAP Permutation
abline(v=bkham_qt$testval, col="red")
```



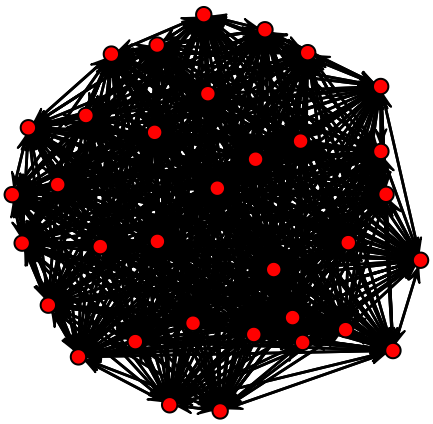

```
#BKTEC
gplot(bktec$Behavioral)
title(main = "BKTEC Behavioral")
```

BKTEC Behavioral



```
gplot(bktec$Cognitive)
title(main = "BKTEC Cognitive")
```

BKTEC Cognitive



```
#Correlation
bktec_cor <- gcor(bktec$Cognitive, bktec$Behavioral)

bktec_cor
```

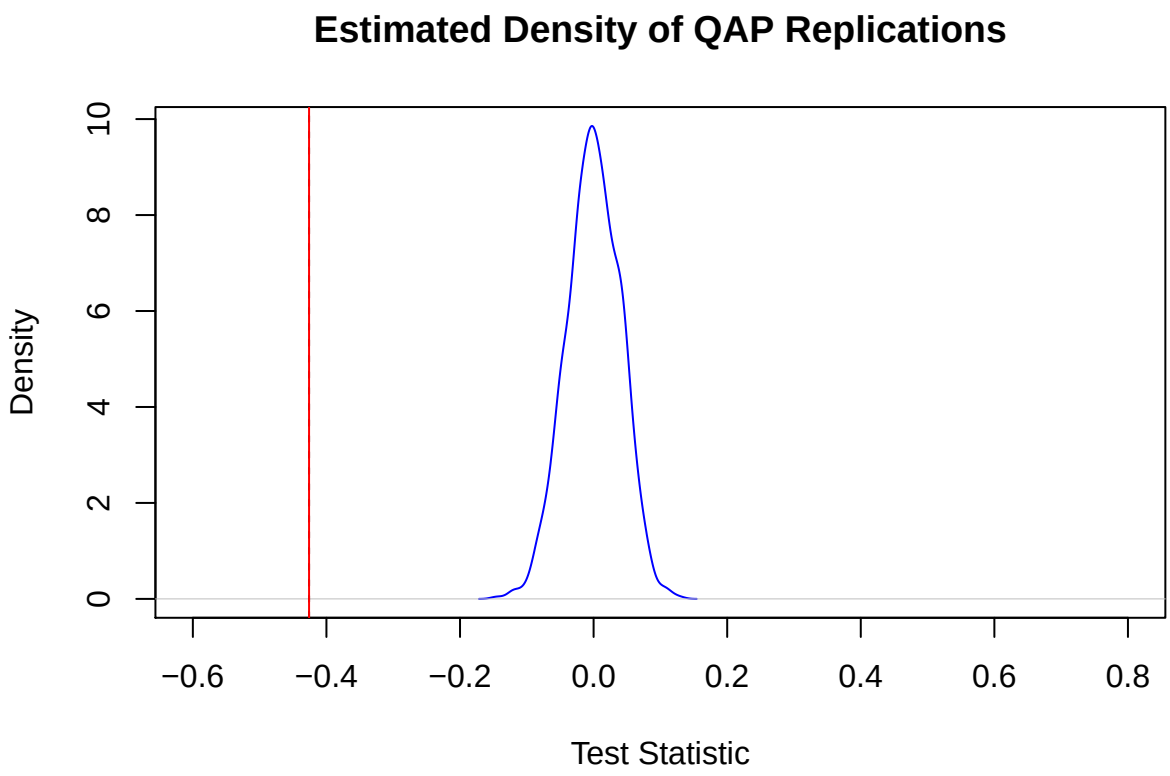
```
## [1] -0.4260065
```

```
# qaptest routine
bktec_qt<-qaptest(list(bktec$Cognitive, bktec$Behavioral),gcor,g1=1,g2=2)
summary(bktec_qt)                                          # Examine the results
```

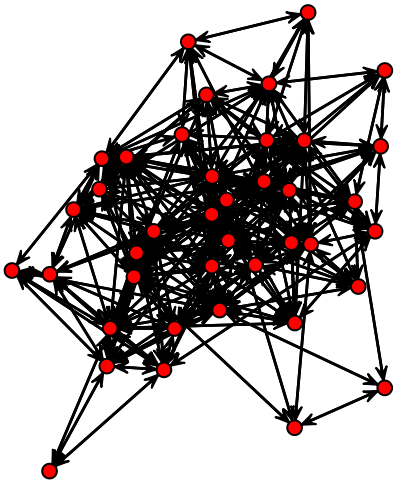
```
##
## QAP Test Results
##
## Estimated p-values:
##  p(f(perm) >= f(d)): 1
##  p(f(perm) <= f(d)): 0
##
## Test Diagnostics:
##  Test Value (f(d)): -0.4260065
##  Replications: 1000
##  Distribution Summary:
##      Min:     -0.1444982
##      1stQ:    -0.02616108
##      Med:     0.000191404
##      Mean:    -0.0001070914
##      3rdQ:    0.02812553
```

```
##       Max:      0.1272462
```

```
#Comparing with QAP Permutation
plot(bktec_qt, xlim=c(-0.6,0.8), col="blue")
abline(v= bktec_qt$testval, col="red")
```

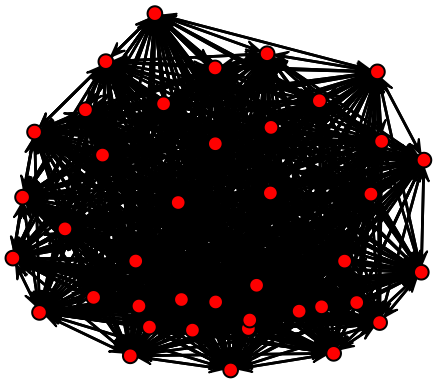



```
#BKOFF
gplot(bkoff$Behavioral)
title(main = "BKOFF Behavioral")
```




```
gplot(bkoff$Cognitive)
title(main = "BKOFF Cognitive")
```

BKOFF Cognitive



```
#Correlation
bkoff_cor <- gcor(bkoff$Cognitive, bkoff$Behavioral)

bkoff_cor
```

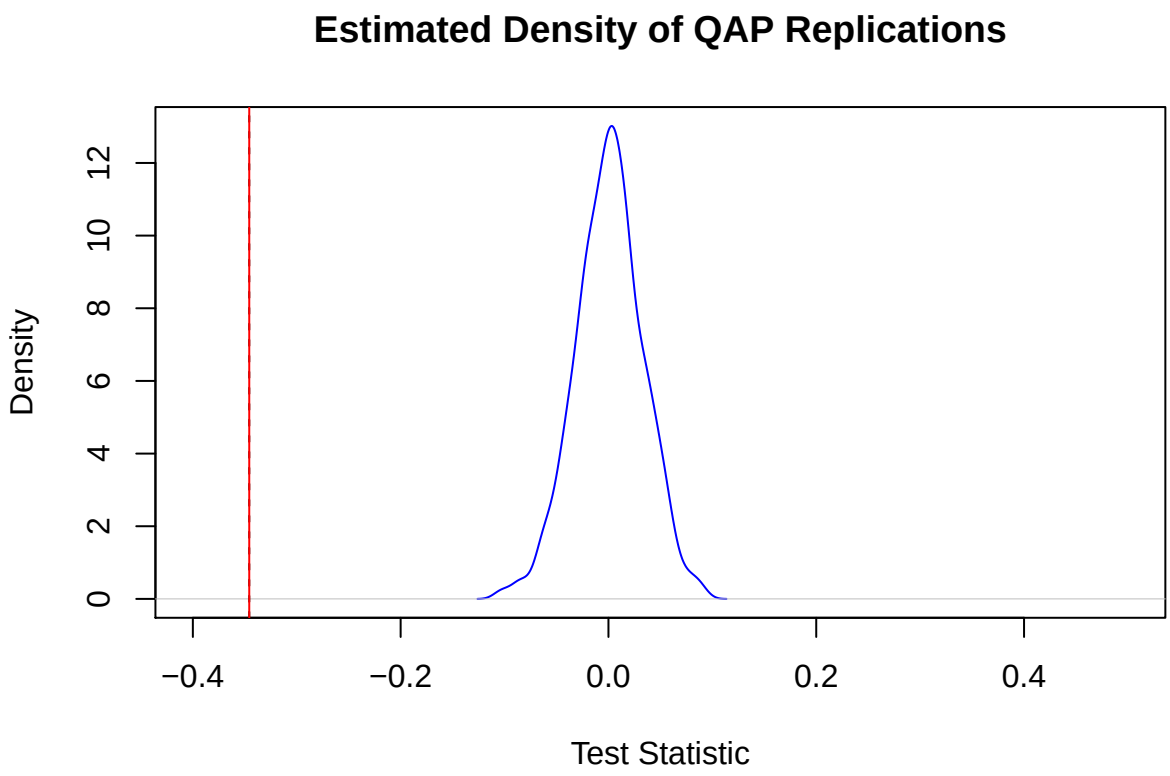
```
## [1] -0.3457147
```

```
# qaptest routine
bkoff_qt<-qaptest(list(bkoff$Cognitive, bkoff$Behavioral),gcor,g1=1,g2=2)
summary(bkoff_qt)                                          # Examine the results
```

```
##
## QAP Test Results
##
## Estimated p-values:
##  p(f(perm) >= f(d)): 1
##  p(f(perm) <= f(d)): 0
##
## Test Diagnostics:
##  Test Value (f(d)): -0.3457147
##  Replications: 1000
##  Distribution Summary:
##      Min:     -0.1049064
##      1stQ:    -0.02093976
##      Med:     0.001227825
##      Mean:    0.0007174458
##      3rdQ:    0.020837
```

```
##       Max:      0.09239301
```

```
#Comparing with QAP Permutation
plot(bkoff_qt, xlim=c(-0.4,0.5), col="blue")
abline(v=bkoff_qt$testval, col="red")
```



(b) Discussion

Use the results from part (a) to provide your own assessment of the extent to which the data does or does not show general agreement between observation and informant report.

Ans. As can be seen from the plots, the networks show significant departure from the QAP test permutations.

For BKFRAT and BKHAM networks, there is an agreement between the cognitive and observational report of interactions. For BKFRAT, correlation co-efficient is 0.37 and for BKHAM, its 0.52.Thus, BKHAM network shows more agreement than the BKFRAT network.

For BKTEC and BKOFF networks, there is a negative association between the cognitive and observational report of interactions. For BKTEC the correlation co-efficient is -0.42 which is lesser than the coefficient for BKOFF network (-0.34). This means that if an inofrmant shows an opposite report of interaction than an observer.

(c) Observation and Networks

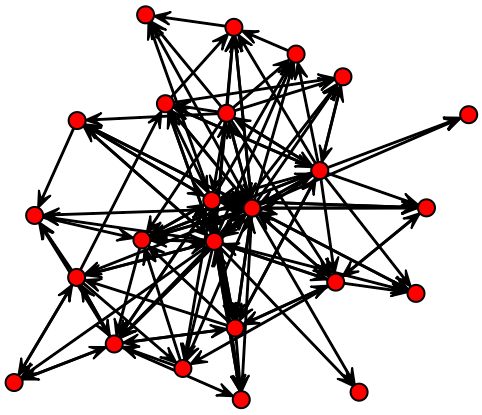
What reliability or validity issues might arise in the BKS studies if the observed report data is taken to be the true "behavioral" network?

Ans. If the observational report is taken to be the true "behavioral" network, there will be a discrepancy in the report of interactions actually occurred. There will not be an actual measure of the interactions and the reports of the observants will be subjective and biased. For example, in the BKTEC and BKOFF networks, it was observed that the cognitive and observational reports were in disagreement.Thus, the reliability of the research conducted on these kind of observations will be poor and might lead incorrect conclusions about network metrics like Betweenness and Degree. Another observer might have a different report on the interactions and might agree with the cognitive report.

Problem 3: Multivariate Analysis of Network Sets

For this problem we will use data on international trade, called `trade` in the data for this problem set. This data captures trade in various types of products/materials among countries. You will want to explore the data before answering these questions, to ensure you understand what is present.

```
#Plotting the Trade data
gplot(trade)
```



```
row.names(trade)
```

```
## [1] "MINERALS"            "CRUDE_MATERIALS"     "FOODS"
## [4] "MANUFACTURED_GOODS"  "DIPLOMATIC_EXCHANGE"
```

```
class(trade[1,,])
```

```
## [1] "matrix"
```

```
colnames(trade)
```

```
##  [1] "ALGERIA "        "ARGENTINA "      "BRAZIL "
```

```
##  [4] "CHINA "          "CZECHOSLOVAKIA " "ECUADOR "
##  [7] "EGYPT "          "ETHIOPIA "       "FINLAND "
## [10] "HONDURAS "       "INDONESIA "      "ISRAEL "
## [13] "JAPAN "          "LIBERIA "        "MADAGASCAR "
## [16] "NEW_ZEALAND "    "PAKISTAN "       "SPAIN "
## [19] "SWITZERLAND "    "SYRIA "          "THAILAND "
## [22] "UNITED_KINGDOM " "UNITED_STATES "  "YUGOSLAVIA"
```

```
str(trade)
```

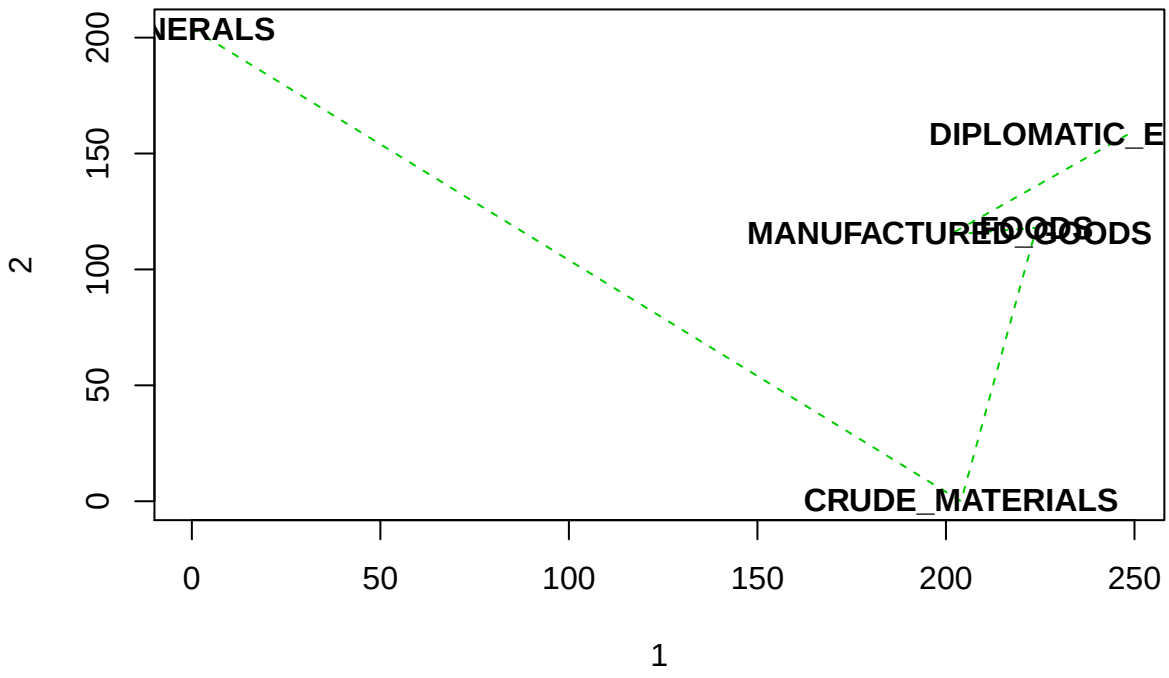
```
##  num [1:5, 1:24, 1:24] 0 0 0 0 0 0 1 1 1 1 ...
##  - attr(*, "dimnames")=List of 3
##   ..$ : chr [1:5] "MINERALS" "CRUDE_MATERIALS" "FOODS" "MANUFACTURED_GOODS" ...
##   ..$ : chr [1:24] "ALGERIA " "ARGENTINA " "BRAZIL " "CHINA " ...
##   ..$ : chr [1:24] "ALGERIA " "ARGENTINA " "BRAZIL " "CHINA " ...
```

```
#its a 3 dimensional array.
```

(a) Clustering

Show a hierarchical clustering of the trade networks, based on the Hamming distance. Compare this with a two-dimensional MDS solution on the same data.

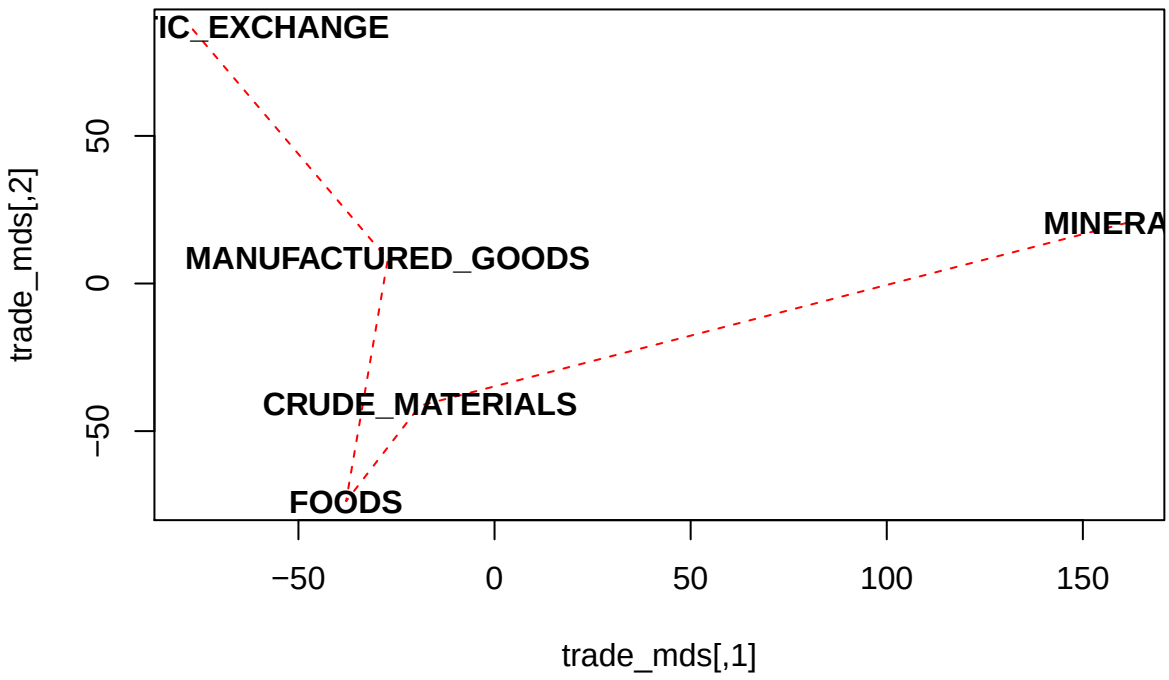
```
#Hamming Distance clustering
trade_hdist <- hdist(trade)

#Plot the Results
plot(trade_hdist, type = 'l', lty=2, col=3)
text(trade_hdist,label=rownames(trade),font=2)
```



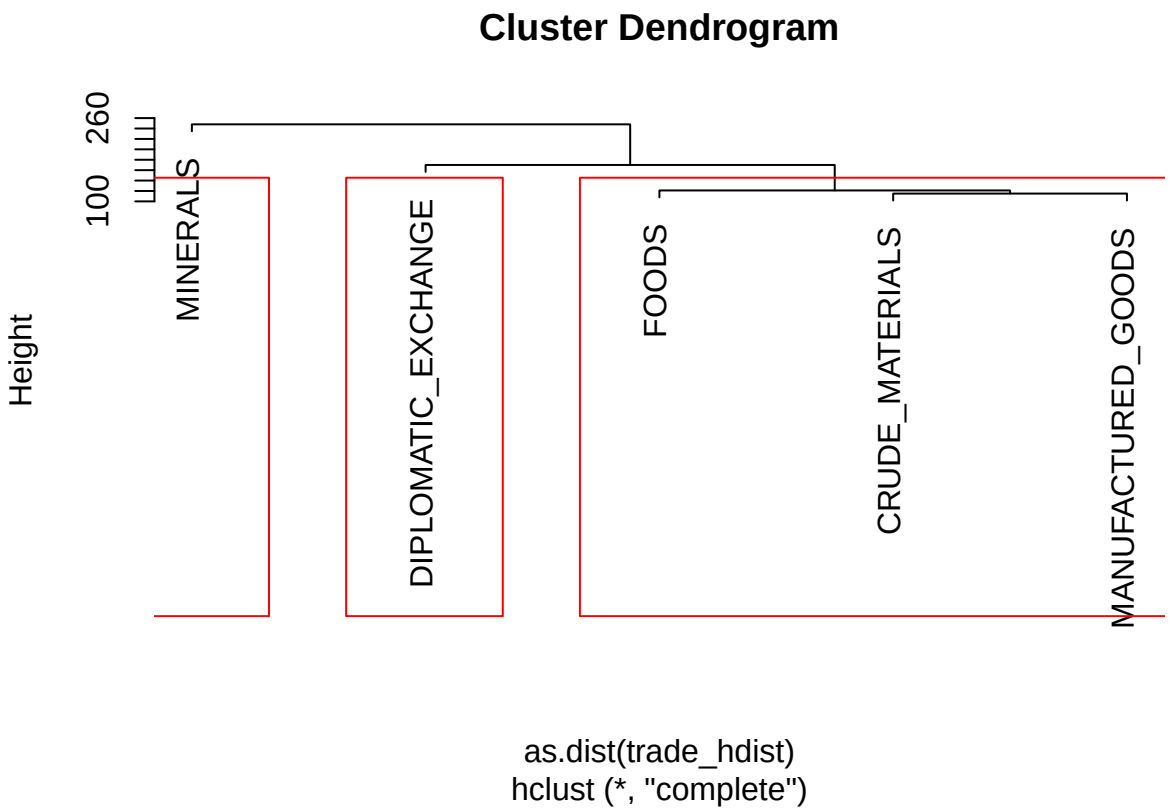

```
#MDS solution
trade_mds <- cmdscale(trade_hdist)
trade_mds
```

```
##        [,1]       [,2]
## 1 161.24239  20.559016
## 2 -18.99459 -41.374438
## 3 -37.84677 -73.752999
## 4 -27.22379   8.134394
## 5 -77.17724  86.434026
```

```
# Plot the results
plot(trade_mds,type="l",lty=2, col=2)
text(trade_mds,label=rownames(trade),font=2)
```




```
# MDS suggests a three-cluster solution; let's try hclust
trade_hc<-hclust(as.dist(trade_hdist))
plot(trade_hc,labels=row.names(trade))
rect.hclust(trade_hc,k=3.5)
```



(b) PCA

Conduct a PCA on the trade networks. How many dimensions are needed to account for the bulk of the variation in these networks? Try using a scree plot to help with this question. Plot the loadings on the first two components; what does this suggest about the undering relationships among the trade networks?

```
#Correlations
trade_cor <- gcor(trade)
trade_cor
```

```
##           1         2         3         4         5
## 1 1.0000000 0.3725626 0.2877321 0.3922966 0.3380220
## 2 0.3725626 1.0000000 0.5670013 0.5775224 0.4165298
## 3 0.2877321 0.5670013 1.0000000 0.5554769 0.3700570
## 4 0.3922966 0.5775224 0.5554769 1.0000000 0.5333617
## 5 0.3380220 0.4165298 0.3700570 0.5333617 1.0000000
```

```
#Eigen vectors
trade_eig<-eigen(trade_cor)
trade_eig
```

```
## $values
## [1] 2.7892765 0.7554745 0.6464812 0.4228567 0.3859110
##
## $vectors
##            [,1]       [,2]       [,3]        [,4]        [,5]
## [1,] -0.3630871  0.8347337  0.3871989 -0.13546147  0.05580884
```

```
## [2,] -0.4801280 -0.2271629  0.2725159  0.76526418  0.24079034
## [3,] -0.4540009 -0.4629607  0.3095938 -0.62712736  0.30068855
## [4,] -0.5018676 -0.1047656 -0.1336908 -0.03169384 -0.84751125
## [5,] -0.4238597  0.1621982 -0.8136895 -0.04156554  0.36085518
```

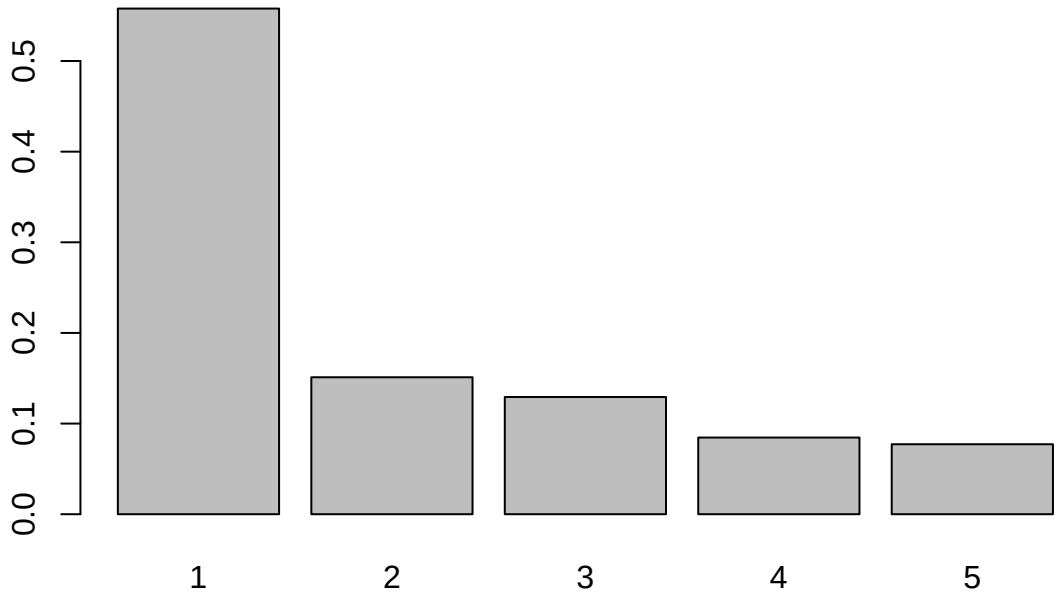
```
# Eigenvalues contain variance explained:
evals<-trade_eig$value                                    # Extract eigenvalues
evals
```

```
## [1] 2.7892765 0.7554745 0.6464812 0.4228567 0.3859110
```

```
evals/sum(evals)
```

```
## [1] 0.55785530 0.15109490 0.12929625 0.08457135 0.07718221
```

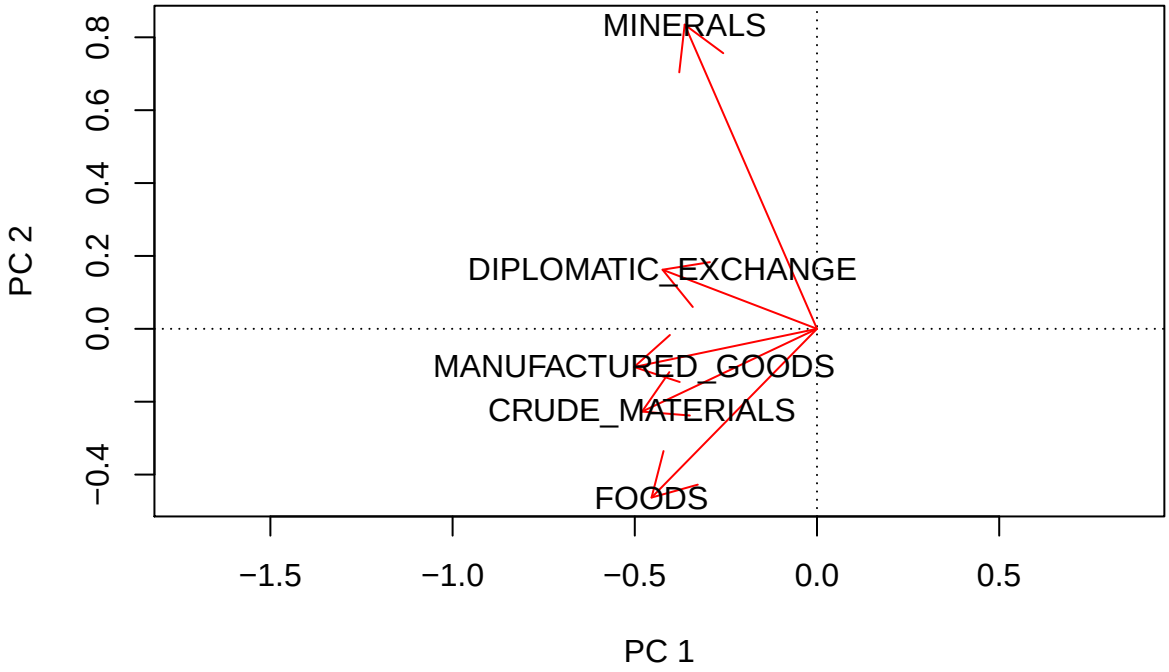
```
# Showing this as a scree plot
barplot(evals/sum(evals),names.arg=1:length(evals))
```




```
load<-trade_eig$vector[,1:3]
rownames(load)<-row.names(trade)
load
```

```
##                          [,1]       [,2]       [,3]
## MINERALS            -0.3630871  0.8347337  0.3871989
## CRUDE_MATERIALS     -0.4801280 -0.2271629  0.2725159
## FOODS               -0.4540009 -0.4629607  0.3095938
## MANUFACTURED_GOODS  -0.5018676 -0.1047656 -0.1336908
## DIPLOMATIC_EXCHANGE -0.4238597  0.1621982 -0.8136895
```

```r
# Plotting the first two components
plot(load[,1:2],type="n",asp=1,xlab="PC 1",ylab="PC 2")
abline(h=0,v=0,lty=3)
arrows(0,0,load[,1],load[,2],col=2)
text(load[,1:2],label=row.names(trade))
```



Ans. As evident in the scree plot, more than 60% of the variance in the network is explained by dimension 1 i.e Minerals. The 2nd dimension explains 15% of the variance. Thus, two dimensions are enough to account for the variation in this network. The variation in the underlying trade network structure can be explained maximally on the basis of mineral trade among the countries.

(c) Discussion

Compare your PCA results to those obtained using MDS. In what ways are they similar? Different?

The MDS structure created a euclidean distance in which trades with similar patterns are grouped approximately. The plot of the MDS solution shows that there the Minerals trade data is far away from the rest of the network; this proves that the Minerals trade data is unique. It is further explained in the PCA network, which shows that the maximum variance in the trade network is explained by the Minerals trade among the 24 countries.