# Networking_Assignment_1

October 29, 2024

## 1 Data Preprocessing

Here we first load the dataset:

[1]:
```
# Load the necessary packages
install.packages('ergm')
install.packages('sna')
library(ergm)
library(network)
library(sna)
```

```
Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)

Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)

Loading required package: network


'network' 1.18.2 (2023-12-04), part of the Statnet Project
* 'news(package="network")' for changes since last version
* 'citation("network")' for citation information
* 'https://statnet.org' for help, support, and other information



'ergm' 4.7.1 (2024-10-07), part of the Statnet Project
* 'news(package="ergm")' for changes since last version
* 'citation("ergm")' for citation information
* 'https://statnet.org' for help, support, and other information


'ergm' 4 is a major update that introduces some backwards-incompatible
changes. Please type 'news(package="ergm")' for a list of major
changes.
```

```
Loading required package: statnet.common


Attaching package: ‘statnet.common’


The following objects are masked from ‘package:base’:

    attr, order


sna: Tools for Social Network Analysis
Version 2.8 created on 2024-09-07.
copyright (c) 2005, Carter T. Butts, University of California-Irvine
 For citation information, type citation("sna").
 Type help(package="sna") to get started.
```

Read the information / data from the files given.

[2]:
```
# Reading the adjacency matrices of the networks
list.files()
attributes=read.table("Krackhardt-High-Tech_nodes.txt",header=TRUE)
nodes=as.matrix(read.table("Krackhardt-High-Tech_multiplex.edges",header=FALSE))
colnames(nodes)=c("layerID","IDi","IDj","weight")
```

1. 'Krackhardt-High-Tech_layers.txt' 2. 'Krackhardt-High-Tech_multiplex.edges' 3. 'Krackhardt-High-Tech_nodes.txt' 4. 'sample_data'

Below we build the adjacency matrix for advice, friendship and Reports_to, respectively.

[3]:
```
# Filter out the dataframe for advice, friendship, Reports_to
advice = nodes[(nodes[,1] == 1), ]
friendship = nodes[(nodes[,1] == 2), ]
Reports_to = nodes[(nodes[,1] == 3), ]

# Count the number of the node
nodeID<-attributes[,1]
numID = max(nodeID)

# Friendship adjacency matrix
f.matrix=matrix(0,nrow=numID,ncol=numID)
colnames(f.matrix)=1:ncol(f.matrix)
for( idx in 1:nrow(friendship)){
    f.
 ↪matrix[friendship[idx,'IDi'],friendship[idx,'IDj']]=friendship[idx,'weight']
}
```

```
# Advice adjacency matrix
advice.matrix=matrix(0,nrow=numID,ncol=numID)
colnames(advice.matrix)=1:ncol(advice.matrix)
for( idx in 1:nrow(advice)){
    advice.matrix[advice[idx,'IDi'],advice[idx,'IDj']]=advice[idx,'weight']
}

#extract reporting as adjacency matrix
reports.matrix=matrix(0,nrow=numID,ncol=numID)
colnames(reports.matrix)=1:ncol(reports.matrix)
for( idx in 1:nrow(Reports_to)){
    reports.matrix[Reports_to[idx,'IDi'],Reports_to[idx,'IDj']]=Reports_to[idx,
 ↪'weight']
}
```

# 2 Task 1

## 2.1 Question 1

Build a QAP to test if friendship and advice relations correlate. Use at least 5,000 permutations for reporting the results.

[4]:
```
set.seed(0)
permutations=10000

#build the QAP test
g1=netlogit(advice.matrix,f.matrix, rep=permutations,nullhyp='qapy')
g1$names=c('intercept','friendship')
summary(g1)
```

```
Network Logit Model

Coefficients:
            Estimate   Exp(b)    Pr(<=b) Pr(>=b) Pr(>=|b|)
intercept  -0.3689075 0.6914894 0.0177  0.9838  0.0177
friendship  0.7255825 2.0659341 0.9846  0.0160  0.0326

Goodness of Fit Statistics:

Null deviance: 582.2436 on 420 degrees of freedom
Residual deviance: 568.4124 on 418 degrees of freedom
Chi-Squared test of fit improvement:
         13.83123 on 2 degrees of freedom, p-value 0.0009921717
AIC: 572.4124         BIC: 580.4929
Pseudo-R^2 Measures:
        (Dn-Dr)/(Dn-Dr+dfn): 0.03188159
```

```
        (Dn-Dr)/Dn: 0.02375505
Contingency Table (predicted (rows) x actual (cols)):

         Actual
Predicted     0     1
        0   188   130
        1    42    60

        Total Fraction Correct: 0.5904762
        Fraction Predicted 1s Correct: 0.5882353
        Fraction Predicted 0s Correct: 0.591195
        False Negative Rate: 0.6842105
        False Positive Rate: 0.1826087

Test Diagnostics:

        Null Hypothesis: qapy
        Replications: 10000
        Distribution Summary:

       intercept friendship
Min     -4.11110   -4.68751
1stQ    -2.23715   -0.94629
Median  -1.67982   -0.03266
Mean    -1.69969    0.01118
3rdQ    -1.23271    1.10913
Max      0.67277    4.87127
```

[5]:

```
# Formatting and exporting the results
res1 <- summary(g1)
expRes1 <- cbind(res1$coefficients, exp(res1$coefficients),
res1$se, res1$pgreqabs)
colnames(expRes1) <- c("ESt.", "exp(Est.)", "s.e.", "p-value")
rownames(expRes1) <- res1$names
#write.csv(expRes,"resQAP_Q1.csv")

# Exporting results in tex
library(xtable)
xtable(expRes1,digits=3)
```

A xtable: $2 \times 4$

| | ESt. <dbl> | exp(Est.) <dbl> | s.e. <dbl> | p-value <dbl> |
|---|---|---|---|---|
| intercept | -0.3689075 | 0.6914894 | 0.1140678 | 0.0177 |
| friendship | 0.7255825 | 2.0659341 | 0.2312740 | 0.0326 |

### 2.1.1 Conclusion:

The parameter related to the friendship nomination is significant and positive indicating that a seeking advice tie $A_{ij}$ is more likely when a manager $i$ nominate the second manager $j$ as a friend. Specifically, the odds of manager $i$ seeking advice from manager $j$ when a manager $i$ nominate the second manager $j$ as a friend are 2.066 times greater than the odds of this kind of tie when manager $i$ dose not nominate $j$ as a friend.

## 2.2 Question 2 & 3

Hypotheses: 1. A friendship nomination is more likely between a pair of managers within the same deparment. 2. Senior managers are less likely to nominate friends. 1. A friendship nomination is more likely between a pair of managers of a similar age.

The friendship nomination $X_{ij}$ ties are the dependent variable. The explanatory variables that allow testing the hypotheses above are defined in the following.

The first Hypothese states the relations friendship nomination and working at the same department are associated. The corresponding variable (dyadic covariates) is an indicator functions taking value 1 if two lawyers have the same characteristics, and 0 otherwise.

$$\text{Hp. 1: } Z_{ij} = \begin{cases} 1, & \text{if } \{i, j\} \text{ in the same department.} \\ 0, & \text{otherwise} \end{cases}$$

The second and third Hypotheses concern the dependence of a tie on the seniority of the manager and the differences between ages of the node pair. Thus, these variables take the form:

$$\text{Hp. 2: } Z_{ij} = \text{Tenure}_i \text{Hp. 3: } Z_{ij} = |\text{Age}_i - \text{Age}_j|$$

First, we create vectors containing the values of the explanatory variables and then create the corresponding matrices.

[6]:
```
# Hp. 1: A friendship nomination is more likely between a pair of managers␣
↪within the same deparment
department <- attributes[['nodeDepartment']]
same_department.matrix <- outer(department,department,"==")*1

# Hp. 2: senior managers are less likely to nominate friends.
tenure.matrix = matrix(0, nrow=numID, ncol=numID)
for(i in 1:numID){
  for(j in 1:numID){
    #The value at position (i, j) is tenure[i]
    tenure.matrix[i, j] = attributes[["nodeTenure"]][i]
  }
}

# Hp. 3: a friendship nomination is more likely between a pair of managers of a␣
↪similar age.
```

```
age_dif.matrix = matrix(0, nrow=numID, ncol=numID)
for(i in 1:numID){
  for(j in 1:numID){
    # The value at position (i, j) is |age[i] - age[j]|
    age_dif.matrix[i, j] =
 ↪abs(attributes[["nodeAge"]][i]-attributes[["nodeAge"]][j])
  }
}
```

[7]:
```
zm <- list(advice.matrix, same_department.matrix, tenure.matrix, age_dif.matrix)

g2<-netlogit(f.matrix,zm, rep=permutations,nullhyp='qapspp')
g2$names<-c('intercept', 'advice', 'same_department', 'tenure', 'age_dif')
summary(g2)
```

```
Network Logit Model

Coefficients:
                Estimate    Exp(b)    Pr(<=b) Pr(>=b) Pr(>=|b|)
intercept       -1.97460863 0.1388156 0.0049  0.9951  0.0049
advice           0.78779191 2.1985365 0.9854  0.0146  0.0299
same_department  1.18388201 3.2670323 1.0000  0.0000  0.0000
tenure           0.04876536 1.0499740 0.8707  0.1293  0.2315
age_dif         -0.04722080 0.9538768 0.0315  0.9685  0.0938

Goodness of Fit Statistics:

Null deviance: 582.2436 on 420 degrees of freedom
Residual deviance: 425.0383 on 415 degrees of freedom
Chi-Squared test of fit improvement:
         157.2054 on 5 degrees of freedom, p-value 0
AIC: 435.0383          BIC: 455.2395
Pseudo-R^2 Measures:
        (Dn-Dr)/(Dn-Dr+dfn): 0.2723561
        (Dn-Dr)/Dn: 0.2699993
Contingency Table (predicted (rows) x actual (cols)):

         Actual
Predicted     0     1
        0   306    85
        1    12    17

        Total Fraction Correct: 0.7690476
        Fraction Predicted 1s Correct: 0.5862069
        Fraction Predicted 0s Correct: 0.7826087
        False Negative Rate: 0.8333333
```

```
        False Positive Rate: 0.03773585

Test Diagnostics:

        Null Hypothesis: qapspp
        Replications: 10000
        Distribution Summary:

       intercept    advice same_department    tenure   age_dif
Min    -7.553955 -5.244037       -3.775461 -6.366783 -4.757958
1stQ   -2.248520 -1.015424       -0.796976 -1.758954 -1.163178
Median -0.885677  0.028261       -0.043526 -0.178677 -0.086863
Mean   -0.995801  0.007856        0.001489  0.023970  0.028714
3rdQ    0.373022  1.001290        0.770403  1.660610  1.115361
Max     4.853989  5.098259        4.393634  7.652889  6.113808
```

[8]:

```r
# Formatting and exporting the results
res2 <- summary(g2)
expRes2 <- cbind(res2$coefficients, exp(res2$coefficients),
res2$se, res2$pgreqabs)
colnames(expRes2) <- c("ESt.", "exp(Est.)", "s.e.", "p-value")
rownames(expRes2) <- res2$names
#write.csv(expRes,"resQAP_Q1.csv")

# Exporting results in tex
library(xtable)
xtable(expRes2,digits=3)
```

A xtable: 5 × 4

| | ESt. <dbl> | exp(Est.) <dbl> | s.e. <dbl> | p-value <dbl> |
|---|---|---|---|---|
| intercept | -1.97460863 | 0.1388156 | 0.33380046 | 0.0049 |
| advice | 0.78779191 | 2.1985365 | 0.25125909 | 0.0299 |
| same_department | 1.18388201 | 3.2670323 | 0.26909036 | 0.0000 |
| tenure | 0.04876536 | 1.0499740 | 0.01645553 | 0.2315 |
| age_dif | -0.04722080 | 0.9538768 | 0.01651161 | 0.0938 |

### 2.2.1 Conclusion

The parameters of the same department is significant and positive suggesting that friendship nomination is more likely between a pair of managers within the same deparment. Specifically, the odds of a friendship nomination tie between managers working in the same department are 1.184 times greater than the odds of a tie between managers working in different departments, holding all the other variables constant. Thus, the data supports Hypothesis 1.

The parameters of tenure and age differences are significantly different from 0 at a significance level $\alpha = 0.05$. Thus, the data does not support Hypotheses 2 and 3.

### 2.3 Question 4 & 5

***Question:*** Could you think of another hypothesis that could be tested using QAPs? State your hypothesis and provide the corresponding statistic.

***Solution:*** Yes, here we propose a hypothsis that: **If a person reports to another person as part of his function in organization, then the first person more likely nominate the second as a friend.** The code & statistic are shown below:

[9]:
```
zm1 <- list(advice.matrix, same_department.matrix, tenure.matrix, age_dif.
 ↪matrix,
            reports.matrix)
g3<-netlogit(f.matrix,zm1, rep=permutations,nullhyp='qapspp')
g3$names<-c('intercept', 'advice', 'same_department', 'tenure', 'age_dif',␣
 ↪'report_matrix')
summary(g3)
```

```
Network Logit Model

Coefficients:
                Estimate    Exp(b)    Pr(<=b) Pr(>=b) Pr(>=|b|)
intercept       -1.90160438 0.1493288 0.0061  0.9939  0.0061
advice           0.68802778 1.9897874 0.9631  0.0369  0.0709
same_department  1.03989079 2.8289081 0.9981  0.0019  0.0020
tenure           0.04711714 1.0482448 0.8682  0.1318  0.2441
age_dif         -0.04845630 0.9526990 0.0330  0.9670  0.0928
report_matrix    0.83217434 2.2983106 0.9460  0.0540  0.0992

Goodness of Fit Statistics:

Null deviance: 582.2436 on 420 degrees of freedom
Residual deviance: 422.5441 on 414 degrees of freedom
Chi-Squared test of fit improvement:
         159.6995 on 6 degrees of freedom, p-value 0
AIC: 434.5441         BIC: 458.7856
Pseudo-R^2 Measures:
        (Dn-Dr)/(Dn-Dr+dfn): 0.2754867
        (Dn-Dr)/Dn: 0.274283
Contingency Table (predicted (rows) x actual (cols)):

         Actual
Predicted     0     1
        0   308    85
        1    10    17

        Total Fraction Correct: 0.7738095
        Fraction Predicted 1s Correct: 0.6296296
        Fraction Predicted 0s Correct: 0.783715
```

```
        False Negative Rate: 0.8333333
        False Positive Rate: 0.03144654

Test Diagnostics:

        Null Hypothesis: qapspp
        Replications: 10000
        Distribution Summary:

       intercept    advice same_department    tenure   age_dif report_matrix
Min    -7.505426 -4.835998       -4.037302 -6.444961 -4.712527     -3.044494
1stQ   -2.171582 -1.014912       -0.791513 -1.773822 -1.219133     -0.649187
Median -0.782573 -0.018775       -0.016344 -0.207976 -0.121814     -0.011678
Mean   -0.897303  0.004527        0.006007 -0.008699 -0.011613      0.014251
3rdQ    0.434792  1.020886        0.752871  1.628084  1.070960      0.666118
Max     4.713405  5.656976        4.692464  7.872279  5.791957      3.752562
```

[10]:

```r
# Formatting and exporting the results
res3 <- summary(g3)
expRes3 <- cbind(res3$coefficients, exp(res3$coefficients),
res3$se, res3$pgreqabs)
colnames(expRes3) <- c("ESt.", "exp(Est.)", "s.e.", "p-value")
rownames(expRes3) <- res3$names
#write.csv(expRes,"resQAP_Q1.csv")

# Exporting results in tex
library(xtable)
xtable(expRes3,digits=3)
```

A xtable: 6 × 4

| | ESt.<br><dbl> | exp(Est.)<br><dbl> | s.e.<br><dbl> | p-value<br><dbl> |
|---|---|---|---|---|
| intercept | -1.90160438 | 0.1493288 | 0.33580013 | 0.0061 |
| advice | 0.68802778 | 1.9897874 | 0.25944396 | 0.0709 |
| same_department | 1.03989079 | 2.8289081 | 0.28547655 | 0.0020 |
| tenure | 0.04711714 | 1.0482448 | 0.01655627 | 0.2441 |
| age_dif | -0.04845630 | 0.9526990 | 0.01669396 | 0.0928 |
| report_matrix | 0.83217434 | 2.2983106 | 0.53090641 | 0.0992 |

### 2.3.1 Conclusion

The parameter of report relation is significantly different from 0 at a significance level $\alpha = 0.05$. Thus, the data does not support our Hypothsis mentioned above.

# 3 Task 2: Simulation from an ERGM

## 3.1 Question 1

Some parts of the code are missing as denoted by the chunk code - - - MISSING - - -. Implement these in the R script, and include comments explaining what your code is doing.

The solution is below:

[11]:
```
#function calculating the given statics of an adjacency matrix:
stat=function(net){
    z1=0
    z2=0
    z3=0
    indegree=0
    nvertices <- nrow(net)

    for(k in 1:nvertices){
      for(q in 1:nvertices){
        z1=z1+net[k,q]             #calculating the num of edges
        if(k<q){
          z2=z2+net[k,q]*net[q,k]  #calculating reciprocity
        }
        if(net[q,k]==1){           #calculating indegree of node k
          indegree=indegree + 1
        }
      }
      z3=z3+choose(indegree,2)     #calculating 2-istar and resetting for next␣
 ↪node
      indegree=0
    }
    return(list(z1,z2,z3))
}

# MHstep -------------------------------------------------------------------
#' Simulate the next step of a network in Markov chain using Metropolis-Hasting
#'
#' The function `MHstep` simulates the the Metropolis-Hastings step that defines
#' the Markov chain whose stationary distribution is the ERGM with
#' edge, mutual and nodematch statistics
#'
#' @param net an object of class `matrix`. Adjacency matrix of the network.
#' @param theta1 a numeric value. The value of the edge parameter of the ERGM.
#' @param theta2 a numeric value. The value of the mutual parameter of the ERGM.
#' @param theta3 a numeric value. The value of the istar(2) parameter of the␣
 ↪ERGM.
#'
#' @return next state of the Markov Chain
#'
```

```r
#' @examples
#' MHstep(
#'   matrix(c(0, 1, 0, 0, 0, 0, 1, 1, 0), nrow = 3, ncol = 3),
#'   -log(0.5), log(0.4), log(.8)
#' )
MHstep <- function(net, theta1, theta2, theta3){

  # Number of vertices in the network
  nvertices <- nrow(net)

  # Choose randomly two vertices, prevent loops {i,i} with replace = FALSE
  tie <- sample(1:nvertices, 2, replace = FALSE)
  i <- tie[1]
  j <- tie[2]

  # Compute the change statistics

  net2=net    #creating the matrix with different i->j

  if(net[i,j]==0)
    net2[i,j]=1
  else
    net2[i,j]=0
  #calculating the statistics for the input
  stat1=stat(net)

  #computing the statistics for the changed graph
  stat2=stat(net2)


  # Compute the probability of the next state
  # according to the Metropolis-Hastings algorithm

  # computing both exponential
  stat1=as.double(stat1)
  stat2=as.double(stat2)
  exp1=exp(theta1*stat1[1]+theta2*stat1[2]+theta3*stat1[3])
  exp2=exp(theta1*stat2[1]+theta2*stat2[2]+theta3*stat2[3])

  # computing transition probability

  p=min(1,exp2/exp1)

  # Select the next state:

  #sample with probability p the change of state
  outcome=sample(c(0,1),size=1,prob=c(1-p,p))
```

```r
  if(outcome==1)
    net=net2

  # Return the next state of the chain
  return(net)
}

# Markov Chain simulation -----------------------------------------------------
#' The function MarkovChain simulates the networks from the ERGM with
#' edge, mutual and nodematch statistics
#'
#' @param net an object of class `matrix`. Adjacency matrix of the network.
#' @param theta1 a numeric value. The value of the edge parameter of the ERGM.
#' @param theta2 a numeric value. The value of the mutual parameter of the ERGM.
#' @param theta3 a numeric value. The value of the istar(2) parameter of the␣
 ↪ERGM.
#' @param burnin an integer value.
#'   Number of steps to reach the stationary distribution.
#' @param thinning an integer value. Number of steps between simulated networks.
#' @param nNet an integer value. Number of simulated networks to return as␣
 ↪output.
#'
#' @return a named list:
#'   - netSim: an `array` with the adjancency matrices of the simulated␣
 ↪networks.
#'   - statSim: a `matrix` with the value of the statistic defining the ERGM.
#'
#' @examples
#' MarkovChain(
#'   matrix(c(0, 1, 0, 0, 0, 0, 1, 1, 0), nrow = 3, ncol = 3),
#'   -log(0.5), log(0.4), log(.8)
#' )
MarkovChain <- function(
    net,
    theta1, theta2, theta3,
    burnin = 10000, thinning = 1000, nNet = 1000){

  # Burnin phase: repeating the steps of the chain "burnin" times
  nvertices <- nrow(net)
  burninStep <- 1 # counter for the number of burnin steps

  # Perform the burnin steps
  for(burninStep in 1:burnin){
    net=MHstep(net, theta1, theta2, theta3)
  }

  # After the burnin phase we draw the networks
```

```
  # The simulated networks and statistics are stored in the objects
  # netSim and statSim
  netSim <- array(0, dim = c(nvertices, nvertices, nNet))
  statSim <- matrix(0, nNet, 3)
  thinningSteps <- 0 # counter for the number of thinning steps
  netCounter <- 1 # counter for the number of simulated network

  while(netCounter<=nNet){
    while(thinningSteps<thinning){
      net=MHstep(net, theta1, theta2, theta3)  #performing 1000 transitions
      thinningSteps=thinningSteps+1
    }
    netSim[1:nvertices,1:nvertices,netCounter]=net #saving the current network␣
↪and its statistics
    statSim[netCounter,1:3]=as.double(stat(net))
    netCounter=netCounter+1
    thinningSteps=0                            #resetting the counter
  }

  # Return the simulated networks and the statistics
  return(list(netSim = netSim, statSim = statSim))
}
```

## 3.2 Question 2

A member of your research team suggested that plausible estimates of the parameters of the ERGM above for the advice network are $\theta_1 = -2.76, \theta_2 = 0.68$ and $\theta_3 = 0.05$.

Solution is shown below: 1. Firstly, we build up a advice network to test the simulation result. The output shows that the number of edges, reciprocal dyads and 2-istar from actual Advice network are $190, 45$ and $930$, respectively.

[12]:
```
# Get the number of edges, reciprocal dyads and 2-istar from actual Advice␣
↪network
statRes = stat(advice.matrix)
statRes = as.double(statRes)
paste('Actual number of edges:', statRes[1])
paste('Actual number of reciprocal dyads:', statRes[2])
paste('Actual number of 2-istar:', statRes[3])
```

'Actual number of edges: 190'

'Actual number of reciprocal dyads: 45'

'Actual number of 2-istar: 930'

2. Then we test our model with the given parameters: $\theta_1 = -2.76, \theta_2 = 0.68$ and $\theta_3 = 0.05$. The result is shown below. Here the columns of **MC_simulation$statSim** are the number of edges, reciprocal dyads and 2-istar, respectively. By comparing with the actual number of

edges, reciprocal dyads and 2-istar, all of the results generated by this simulation are much smaller. ***Hence, we don't think that the suggested values of the parameters are plausible estimates.***

[13]:
```
t1=-2.76
t2=0.68
t3=0.05
ad.matrix=matrix(0, numID,numID)

MC_simulation=MarkovChain(ad.matrix,t1,t2,t3)
#MC_simulation$statSim
```

[14]:
```
avr.dens=mean(MC_simulation$statSim[,1])
avr.rec=mean(MC_simulation$statSim[,2])
avr.star=mean(MC_simulation$statSim[,3])
paste('Average number of edges:', avr.dens)
paste('Average number of reciprocal dyads:', avr.rec)
paste('Average number of 2-istar:', avr.star)
```

'Average number of edges: 27.832'

'Average number of reciprocal dyads: 1.646'

'Average number of 2-istar: 18.132'

### 3.3 Question 3

Guess better estimates of $\theta_1, \theta_2$ and $\theta_3$ based on the analysis in Question 2. Describe the procedure you used to obtain the guessed values.

Solution:

As we find out that all of the simulated results above are much smaller than the relevant actual values, we consider to increase the weights $\theta_1, \theta_2$ and $\theta_3$ of these three variables. Below is the code illustrating the strategy we applied to obtain the guessed values:

1. Firstly, we slightly increased $\theta_1, \theta_2$ and $\theta_3$ from $(-2.76, 0.68, 0.05)$ to $(-2.5, 0.8, 0.1)$.

[15]:
```
t1_g1 = -2.5
t2_g1 = 0.8
t3_g1=0.1
ad_g1.matrix=matrix(0,numID,numID)
MC_g1=MarkovChain(ad_g1.matrix,t1_g1,t2_g1,t3_g1)
avr.dens=mean(MC_g1$statSim[,1])
avr.rec=mean(MC_g1$statSim[,2])
avr.star=mean(MC_g1$statSim[,3])
paste('Average number of edges:', avr.dens)
paste('Average number of reciprocal dyads:', avr.rec)
paste('Average number of 2-istar:', avr.star)
```

'Average number of edges: 41.628'

'Average number of reciprocal dyads: 3.76'

'Average number of 2-istar: 42.693'

2. The simulated results of the first guess above show that the average number of edges, reciprocal dyads and 2-istar are all much smaller than the actual results $(190, 45$ and $930)$, we increase $\theta_1, \theta_2$ and $\theta_3$ again to $(-2, 0.9, 0.2)$ for the second guess.

[16]:
```
t1_g2 = -2.0
t2_g2 = 0.9
t3_g2 = 0.2
ad_g2.matrix=matrix(0,numID,numID)
MC_g2=MarkovChain(ad_g2.matrix,t1_g2,t2_g2,t3_g2)
avr.dens=mean(MC_g2$statSim[,1])
avr.rec=mean(MC_g2$statSim[,2])
avr.star=mean(MC_g2$statSim[,3])
paste('Average number of edges:', avr.dens)
paste('Average number of reciprocal dyads:', avr.rec)
paste('Average number of 2-istar:', avr.star)
```

'Average number of edges: 374.407'

'Average number of reciprocal dyads: 169.127'

'Average number of 2-istar: 3186.577'

3. The simulated results of the second guess above show that the average number of edges, reciprocal dyads and 2-istar are all much larger than the actual results $(190, 45$ and $930)$, we decrease the $\theta_1, \theta_2$ and $\theta_3$ from $(-2, 0.9, 0.2)$ to $(-2.2, 0.85, 0.14)$ for the third guess.

[17]:
```
t1_g3 = -2.2
t2_g3 = 0.85
t3_g3 = 0.14
ad_g3.matrix=matrix(0,numID,numID)
MC_g3=MarkovChain(ad_g3.matrix,t1_g3,t2_g3,t3_g3)
avr.dens=mean(MC_g3$statSim[,1])
avr.rec=mean(MC_g3$statSim[,2])
avr.star=mean(MC_g3$statSim[,3])
paste('Average number of edges:', avr.dens)
paste('Average number of reciprocal dyads:', avr.rec)
paste('Average number of 2-istar:', avr.star)
```

'Average number of edges: 77.006'

'Average number of reciprocal dyads: 11.583'

'Average number of 2-istar: 155.65'

4. The simulated results of the third guess above show that the average number of edges, reciprocal dyads and 2-istar are all much smaller than the actual results $(190, 45$ and $930)$, we increase the $\theta_1, \theta_2$ and $\theta_3$ from $(-2.2, 0.85, 0.14)$ to $(-2.13, 0.87, 0.17)$ for the fourth guess.

[18]:
```
t1_g4 = -2.13
t2_g4 = 0.87
t3_g4 = 0.17
ad_g4.matrix=matrix(0,numID,numID)
MC_g4=MarkovChain(ad_g4.matrix,t1_g4,t2_g4,t3_g4)
avr.dens=mean(MC_g4$statSim[,1])
avr.rec=mean(MC_g4$statSim[,2])
avr.star=mean(MC_g4$statSim[,3])
paste('Average number of edges:', avr.dens)
paste('Average number of reciprocal dyads:', avr.rec)
paste('Average number of 2-istar:', avr.star)
```

'Average number of edges: 144.254'

'Average number of reciprocal dyads: 34.278'

'Average number of 2-istar: 571.243'

5. The simulated results of the fourth guess above show that the average number of edges, reciprocal dyads and 2-istar are all slightly smaller than the actual results $(190, 45$ and $930)$, we slightly increase the $\theta_1, \theta_2$ and $\theta_3$ from $(-2.13, 0.87, 0.17)$ to $(-2.1225, 0.8721, 0.1721)$ for the fifth guess.

[19]:
```
t1_g5 = -2.1225
t2_g5 = 0.8721
t3_g5 = 0.1721
ad_g5.matrix=matrix(0,numID,numID)
MC_g5=MarkovChain(ad_g5.matrix,t1_g5,t2_g5,t3_g5)
avr.dens=mean(MC_g5$statSim[,1])
avr.rec=mean(MC_g5$statSim[,2])
avr.star=mean(MC_g5$statSim[,3])
paste('Average number of edges:', avr.dens)
paste('Average number of reciprocal dyads:', avr.rec)
paste('Average number of 2-istar:', avr.star)
```

'Average number of edges: 161.929'

'Average number of reciprocal dyads: 41.89'

'Average number of 2-istar: 713.624'

It seems that the average number of edges, reciprocal dyads and 2-istar for the fifth guess are similar to the actual results $(190, 45$ and $930)$, hence we choose $(\theta_1, \theta_2, \theta_3) = (-2.1225, 0.8721, 0.1721)$ for the final simulation.

# 4 Task 3: Estimation and interpretation of an ERGM