Assignment1

Mirco Büchel 22-933-097
Kailin Liu 21-951-470
Qinghao Guan 21-750-260
Zhengxu Li 22-944-482

Task 1: MR-QAP logit regression

(1) Import the data. Build a QAP to test if a friendship nomination is more likely between a pair of students with same gender. (Use the functions as.matrix() and get.node.attr() to extract the adjacency matrix and sex covariate from the sna network object.)

The friendship nomination ties(x_ij) are now the dependent variable. The explanatory variable is in this case the gender homophily Z1. The hypothesis could be written as:

Hp.1 : Z1 = 1 if {i,j} are the same gender, Z1 = otherwise.

The model could be specified in the following formula:

logit[π(x_ij)] = θ0 + θ1Z1

We now encode this in R, we transform the explanatory variable Z1 into a matrix, where each entry is 1 if the two nodes are of the same gender and 0 otherwise. This is realized by the function outer(), then we pass to the netlogit() function the dependent variable encoded in "adj_matrix", explanatory variable "same_Gender", number of repetitions "permutations" and type of null hypothesis, in this case we just use the "qapy" which represents the y-permutation since there is only one independent variable involved. In the next exercise, we will develop a more sophisticated one.

Code and output:

```
# Load the data
> load("friend_net.Rda")
>
> #task 1
> #extract adjancency matrix
> adj_matrix <- as.matrix(friend_net)
> #extract sex covariate from the attribute
> gender <- get.node.attr(friend_net, "sex")
> #set the x variable for first hypothesis
> sameGender <- outer(gender,gender, "==") * 1
```

```
>
> #perform QAP test
> set.seed(1)
> permutations <- 1000
> nl1 <- netlogit(adj_matrix, sameGender, reps = permutations, nullhyp =
"qapy")



> nl1$names <- c("intercept","sameGender")
> summary(nl1)

Network Logit Model

Coefficients:
           Estimate  Exp(b)      Pr(<=b) Pr(>=b) Pr(>=|b|)
intercept  -3.663562  0.02564103 0.995   0.005   0.995
sameGender  2.676175 14.52941176 1.000   0.000   0.000

Goodness of Fit Statistics:

Null deviance: 1289.254 on 930 degrees of freedom
Residual deviance: 675.8591 on 928 degrees of freedom
Chi-Squared test of fit improvement:
       613.3947 on 2 degrees of freedom, p-value 0
AIC: 679.8591     BIC: 689.5295
Pseudo-R^2 Measures:
      (Dn-Dr)/(Dn-Dr+dfn): 0.3974322
      (Dn-Dr)/Dn: 0.475775
Contingency Table (predicted (rows) x actual (cols)):

      0     1
0   786   144
1     0     0

      Total Fraction Correct: 0.8451613
      Fraction Predicted 1s Correct: NaN
      Fraction Predicted 0s Correct: 0.8451613
      False Negative Rate: 1
      False Positive Rate: 0

Test Diagnostics:

      Null Hypothesis: qapy
      Replications: 1000
      Distribution Summary:

       intercept sameGender
Min    -13.87647   -3.03885
```

```
1stQ   -13.09496   -0.88377
Median -12.83052   -0.15813
Mean   -12.85009   -0.01345
3rdQ   -12.60127    0.74929
Max    -11.83781    4.80798
```

Based on the output of the model, we could see that the parameter for x1(gender homophily) is significant different from zero(p value = 0), and the value 2.6762 indicates that the odds of friendship nomination increase by a factor of 14.5294(see the column Exp(b)) when there is gender homophily, holding all other variables constant. Therefore, students with same gender are more likely to give friendship nomination, this hypothesis is supported by our data.

(2) Add to the model in (1) variables to test the following hypotheses simultaneously:
i. Boys are more likely to send friendship nominations than girls
ii. Smokers are more likely to receive friendship nominations than non-smokers.
iii. A friendship nomination is more likely between a pair of students pariticipating in the same activity.

In this case for the three hypothesis we have the following explanatory variables(for every i,j):
i)male gender of the sender
Hp1: $Z_1 = 1$ if gender(i) is male, $Z_1 = 0$ if gender(i) is female
ii) smoking habit of the receiver
Hp2: $Z_2 = 1$ if node j is smoker, $Z_2 = 0$ if node j is not a smoker
iii) homophily concerning the activity
HP3: $Z_3 = 1$ if {i,j} participate the same activity

The testing model follows the following formula
$\text{logit}[\pi(x_{ij})] = \theta_0 + \theta_1 Z_1 + \theta_2 Z_2 + \theta_3 Z_3 + \theta_4 Z_4$
Where the $\theta_4 Z_4$ is the parameter and statistic of the problem 1.1

As for the coding part, we followed the similar idea as before, by constructing 3 matrices "sender_gender", "receiver_smoker"," same_activity"(corresponding to hypothesis 1,2,3 respectively) we have necessary explanatory variables for testing the hypotheses simultaneously. Then we embed these three matrices together with the "sameGender" of problem 1.1 to perform MR-QAP.

It is worth mentioning that we used "qapspp" that represents QAP "semi partialling plus" produce, which could perform in this multivariate case since we want to preserve the ancillarity principle of permutation tests(which requires that the dependence between Z_k and all other explanatory variables be kept intact under permutations).

Code:

```
#task 1.2
> gender <- attributes$sex
> smoker <- attributes$smoke
> activity <- attributes$activity
> nNodes <- length(gender)
>
> sender_gender <- matrix(gender, nNodes, nNodes, byrow= FALSE)
> receiver_smoker <- matrix(smoker, nNodes, nNodes, byrow = TRUE)
> same_activity <- outer(activity, activity, "==") * 1
> nl2 <- netlogit(adj_matrix, list(sameGender, sender_gender,
receiver_smoker, same_activity),
+                 reps = permutations, nullhyp = "qapspp")
>
```

(3) Estimate the model specified in (2). Interpret the coefficients of the model and determine whether the data support the hypotheses listed in (2).

As shown in the output of the model, the coefficients of the boy sender(sender_gender), and smoker receiver(receiver_smoker)are not statistically significant, so there is no evidence from our data to support the hypothesis 1 and 2.

However, if we have a look at the coefficient of sameGender(the one same as the task 1.1) and same_activity, we could see the coefficient of gender homophily is still significant, and it is slightly changed from the previous exercise after including more variables. The coefficient of activity homophily is positive(0.5542) which means that if two students participate in same activity, the odd of friendship nomination increase by a factor of 1.7406(see the column of Exp(b)), holding other variables constant, indicates that students taking the same activities are more likely to have friendship nomination dyads, i.e. our hypothesis 3 is supported by the data.

```
nl2$names <-
c("intercept","sameGender","sender_gender","receiver_smoker","same_activity")
> summary(nl2)

Network Logit Model

Coefficients:
                Estimate   Exp(b)      Pr(<=b) Pr(>=b) Pr(>=|b|)
intercept       -3.5452552  0.02886126 0.001   0.999   0.001
sameGender       2.9092556 18.34313856 1.000   0.000   0.000
sender_gender   -0.5834843  0.55795091 0.066   0.934   0.130
receiver_smoker -0.3962334  0.67284964 0.101   0.899   0.200
same_activity    0.5542016  1.74055079 0.989   0.011   0.016

Goodness of Fit Statistics:

Null deviance: 1289.254 on 930 degrees of freedom
```

```
Residual deviance: 658.3235 on 925 degrees of freedom
Chi-Squared test of fit improvement:
       630.9303 on 5 degrees of freedom, p-value 0
AIC: 668.3235     BIC: 692.4994
Pseudo-R^2 Measures:
      (Dn-Dr)/(Dn-Dr+dfn): 0.4042014
      (Dn-Dr)/Dn: 0.4893763
Contingency Table (predicted (rows) x actual (cols)):

      0     1
0   786   144
1     0     0

      Total Fraction Correct: 0.8451613
      Fraction Predicted 1s Correct: NaN
      Fraction Predicted 0s Correct: 0.8451613
      False Negative Rate: 1
      False Positive Rate: 0

Test Diagnostics:

      Null Hypothesis: qapspp
      Replications: 1000
      Distribution Summary:

        intercept sameGender sender_gender receiver_smoker same_activity
Min    -11.111660  -2.920319     -5.384996       -3.970320     -3.368858
1stQ    -5.535361  -0.873380     -1.134893       -0.912147     -0.866811
Median  -4.393147  -0.081256      0.006914       -0.029449     -0.129742
Mean    -4.334950  -0.017294      0.016208       -0.017409     -0.055439
3rdQ    -3.119086   0.701898      1.234608        0.894934      0.660392
Max      0.652614   4.438090      4.565300        3.594376      4.339958
```

(4) Could you think of another hypothesis that could be tested using QAPs? State your hypothesis and provide the corresponding statistic.

Hypothesis: Hockey players are more likely to send friendship nominations than other players.

Solution:
We conduct the test of hypothesis following the same style as what we did previously, we construct a matrix named “hockey”, where the entry has value 1 if and only if the sender´s activity is hockey, we then embedded this matrix together with the previous one that we used to conduct a MR-QAP test.

```
# Task 1.5
> #mark hockey as 1, other activity as 0
> hockey <- activity
> hockey[hockey == 2 | hockey == 3] <- 0
> sender_hockey <- matrix(hockey, nNodes, nNodes, byrow = FALSE)
> nl3 <- netlogit(adj_matrix, list(sameGender, sender_gender,
receiver_smoker, same_activity, sender_hockey), reps = permutations, nullhyp
= "qapspp")
```

(5) Test the hypothesis formulated in (4) by adding the corresponding variable in the MRQAP specified in (3). Comment on the results.

As shown in the result, the coefficient for the statistic “sender_hockey” is not significant, hence there is no evidence supporting our hypothesis of hockey players being more likely to make nominations is not supported by the data.

Interestingly, we could observe that the other statistics has similar performance as before, namely, the estimate of coefficient remains similar and the significance remains the same(sender_gender and receiver_smoker are still not significant, but same_gender and same_activity are indeed significant).

```
> nl3$names <-
c("intercept","sameGender","sender_gender","receiver_smoker","same_activity",
"sender_hockey")
> summary(nl3)

Network Logit Model

Coefficients:
                Estimate   Exp(b)      Pr(<=b) Pr(>=b) Pr(>=|b|)
intercept       -3.5025291  0.03012111 0.000   1.000   0.000
sameGender       2.9091384 18.34098937 1.000   0.000   0.000
sender_gender   -0.5959850  0.55101953 0.069   0.931   0.136
receiver_smoker -0.3968943  0.67240511 0.084   0.916   0.178
same_activity    0.5450385  1.72467477 0.983   0.017   0.022
sender_hockey   -0.1112732  0.89469433 0.397   0.603   0.797

Goodness of Fit Statistics:

Null deviance: 1289.254 on 930 degrees of freedom
Residual deviance: 658.0745 on 924 degrees of freedom
Chi-Squared test of fit improvement:
       631.1792 on 6 degrees of freedom, p-value 0
AIC: 670.0745     BIC: 699.0856
Pseudo-R^2 Measures:
      (Dn-Dr)/(Dn-Dr+dfn): 0.4042965
      (Dn-Dr)/Dn: 0.4895694
Contingency Table (predicted (rows) x actual (cols)):
      0     1
0   786   144
1     0     0

      Total Fraction Correct: 0.8451613
      Fraction Predicted 1s Correct: NaN
      Fraction Predicted 0s Correct: 0.8451613
      False Negative Rate: 1
      False Positive Rate: 0

Test Diagnostics:

      Null Hypothesis: qapspp
      Replications: 1000
      Distribution Summary:

       intercept sameGender sender_gender receiver_smoker same_activity
sender_hockey
Min    -8.572358  -2.876461     -5.723217       -3.401163     -2.893848     -
4.742213
1stQ   -4.703831  -0.788340     -1.158165       -0.926931     -0.805230     -
1.289579
Median -3.320403  -0.151103      0.033795        0.013541     -0.055279     -
0.019564
Mean   -3.275327  -0.021471      0.041100        0.030999     -0.006313
0.030982
3rdQ   -1.937265   0.658554      1.370835        0.937785      0.711611
1.325435
Max     3.216096   4.575230      5.083418        4.071489      4.455461
5.616662
```

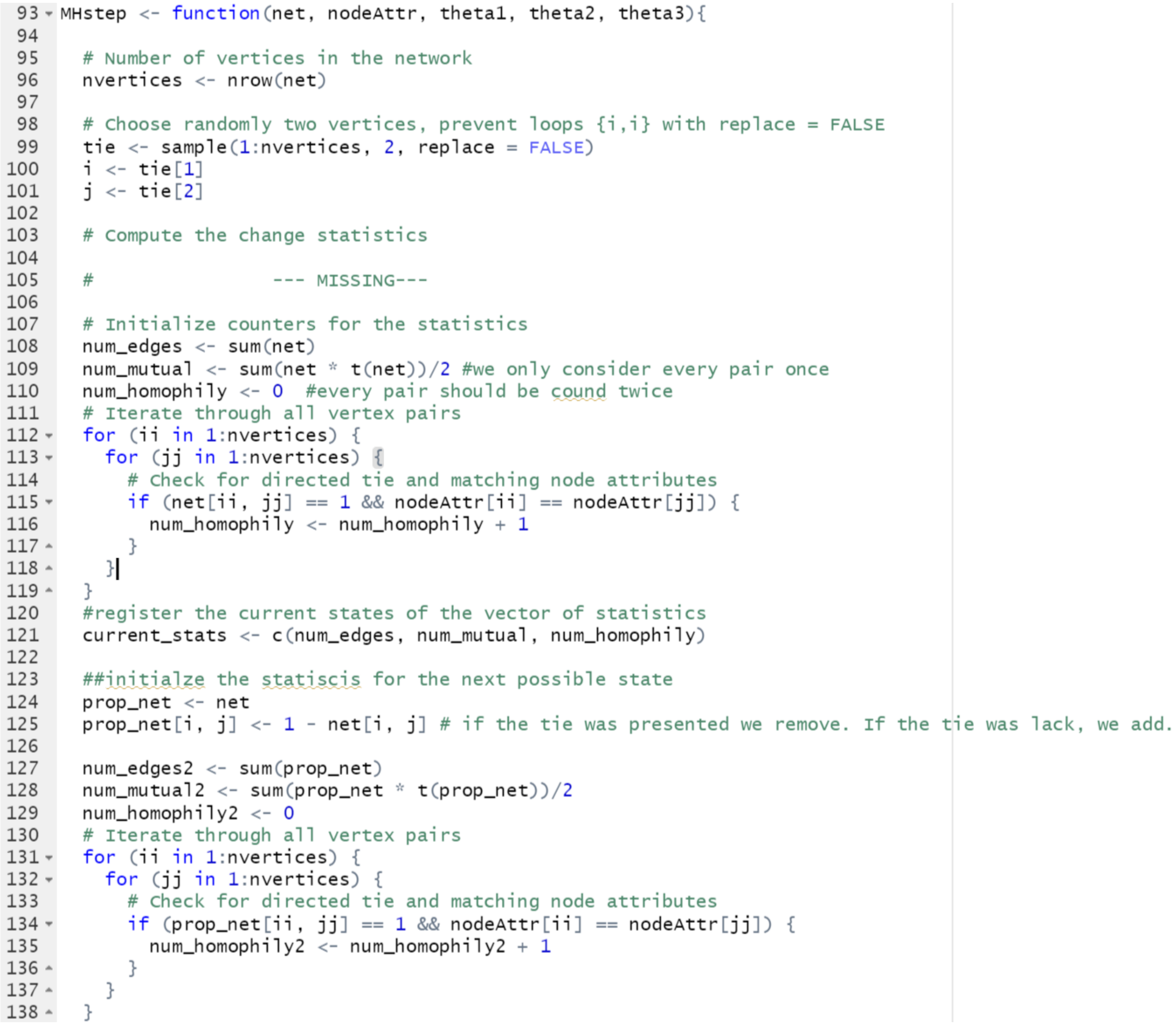
Task 2: Simulation from an ERGM

(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. (Please do not modify existing code even though more efficient solutions can be implemented.)

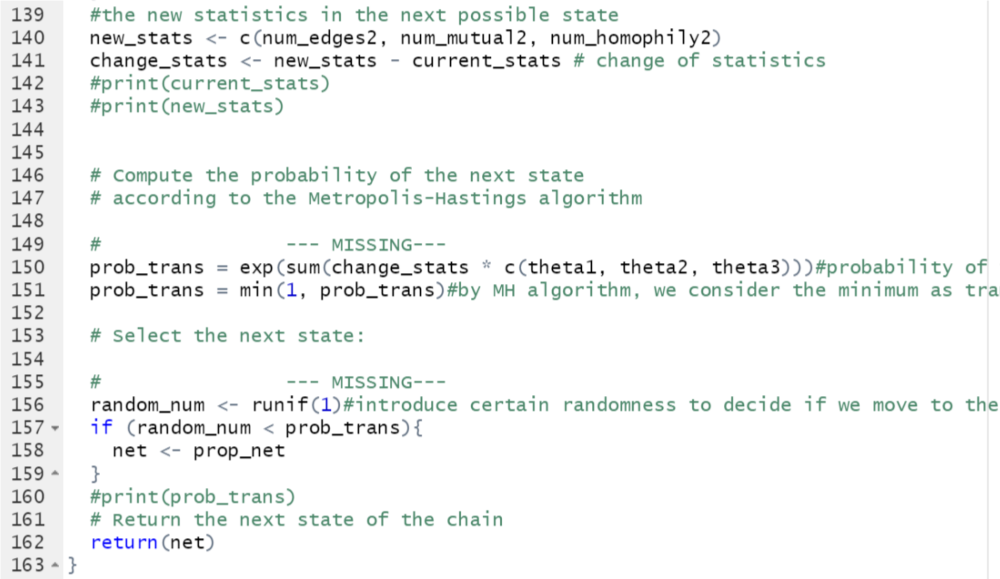
Our code is shown in the following screenshot, with comments explaining every step. Here we only explain the general idea to let you get the intuition.

In the MHstep(): after randomly picking a tie {i,j} we initialized three variables “num_edges”,”num_mutual”,”num_homophily” that store the value of three statistics and embed them into on variable “current_stats”, the calculation of the first two statistics are straightforward, and the third statistics we used two for loops to iterate all i,j pairs to count homophily. Then we toggle the random tie {i,j} are calculate the new statistics following the same method as before and store it in “new_stats”, after that we calculated the change of statistics and plug it in the formula of Metropolis-Hasting to get the transition probability. Then we pick a value “random_num” uniformly at random from 0 to 1, then compare it with transition probability to make decision whether we pass to the next state in MCMC chain or not.

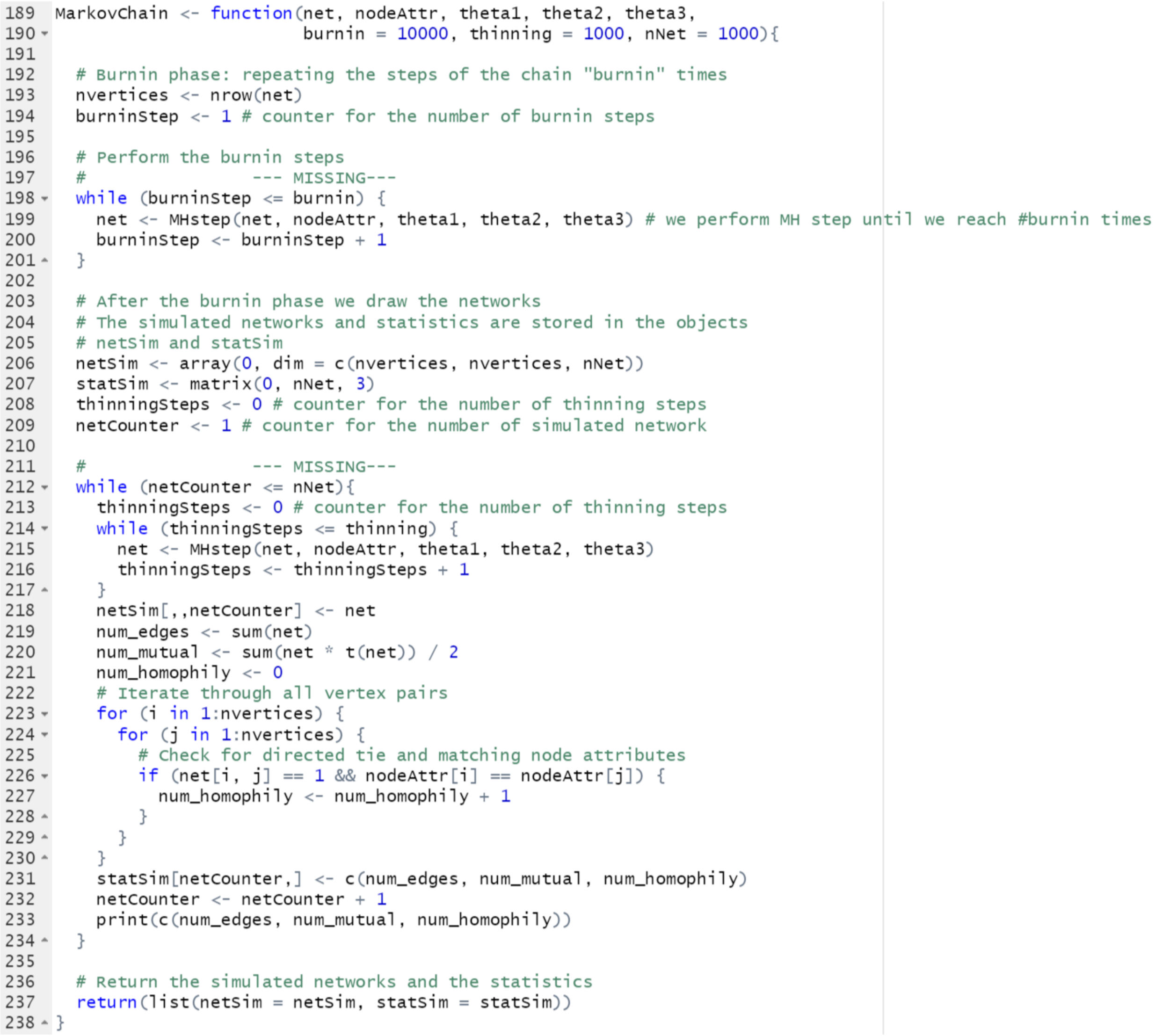
In the Markovchain(): we first run the MHstep() burnin times to throw away some initial iteration at the beginning of MCMC chain’s execution, then we have an outer loop controlled by “nNet”, we only take “nNet” samples of network states to store the simulated network and its corresponding networks' statistics. These samples are made by every “thinning” MHstep()’s function calls in order to reduce autocorrelation, the thinning step is controlled by the inner while loop. Finally, we return the stored sampled networks and statistics.



```
MHstep <- function(net, nodeAttr, 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

  #                --- MISSING---

  # Initialize counters for the statistics
  num_edges <- sum(net)
  num_mutual <- sum(net * t(net))/2 #we only consider every pair once
  num_homophily <- 0  #every pair should be cound twice
  # Iterate through all vertex pairs
  for (ii in 1:nvertices) {
    for (jj in 1:nvertices) {
      # Check for directed tie and matching node attributes
      if (net[ii, jj] == 1 && nodeAttr[ii] == nodeAttr[jj]) {
        num_homophily <- num_homophily + 1
      }
    }
  }
  #register the current states of the vector of statistics
  current_stats <- c(num_edges, num_mutual, num_homophily)

  ##initialze the statiscis for the next possible state
  prop_net <- net
  prop_net[i, j] <- 1 - net[i, j] # if the tie was presented we remove. If the tie was lack, we add.

  num_edges2 <- sum(prop_net)
  num_mutual2 <- sum(prop_net * t(prop_net))/2
  num_homophily2 <- 0
  # Iterate through all vertex pairs
  for (ii in 1:nvertices) {
    for (jj in 1:nvertices) {
      # Check for directed tie and matching node attributes
      if (prop_net[ii, jj] == 1 && nodeAttr[ii] == nodeAttr[jj]) {
        num_homophily2 <- num_homophily2 + 1
      }
    }
  }
```



```r
  #the new statistics in the next possible state
  new_stats <- c(num_edges2, num_mutual2, num_homophily2)
  change_stats <- new_stats - current_stats # change of statistics
  #print(current_stats)
  #print(new_stats)


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

  #                 --- MISSING---
  prob_trans = exp(sum(change_stats * c(theta1, theta2, theta3)))#probability of transition
  prob_trans = min(1, prob_trans)#by MH algorithm, we consider the minimum as transition probability

  # Select the next state:

  #                 --- MISSING---
  random_num <- runif(1)#introduce certain randomness to decide if we move to the next state
  if (random_num < prob_trans){
    net <- prop_net
  }
  #print(prob_trans)
  # Return the next state of the chain
  return(net)
}
```



```r
MarkovChain <- function(net, nodeAttr, 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
  #                 --- MISSING---
  while (burninStep <= burnin) {
    net <- MHstep(net, nodeAttr, theta1, theta2, theta3) # we perform MH step until we reach #burnin times
    burninStep <- burninStep + 1
  }

  # 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

  #                 --- MISSING---
  while (netCounter <= nNet){
    thinningSteps <- 0 # counter for the number of thinning steps
    while (thinningSteps <= thinning) {
      net <- MHstep(net, nodeAttr, theta1, theta2, theta3)
      thinningSteps <- thinningSteps + 1
    }
    netSim[,,netCounter] <- net
    num_edges <- sum(net)
    num_mutual <- sum(net * t(net)) / 2
    num_homophily <- 0
    # Iterate through all vertex pairs
    for (i in 1:nvertices) {
      for (j in 1:nvertices) {
        # Check for directed tie and matching node attributes
        if (net[i, j] == 1 && nodeAttr[i] == nodeAttr[j]) {
          num_homophily <- num_homophily + 1
        }
      }
    }
    statSim[netCounter,] <- c(num_edges, num_mutual, num_homophily)
    netCounter <- netCounter + 1
    print(c(num_edges, num_mutual, num_homophily))
  }

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

(2) With the data from friend net, a member of your research team suggested that plausible estimates of the parameters of the ERGM above for the friendship network are θ1 = −2.76, θ2 = 0.68 and θ3 = 1.21.

(2)i. Use the code developed in (1) to simulate friendship networks from the ERGM with parameters θ1 = −2.76, θ2 = 0.68 and θ3 = 1.21 using as node covariate the gender of the students.

We plug in this set of parameter to the MarkovChain() function that we implemented, the starting point in this case we used a empty matrix,

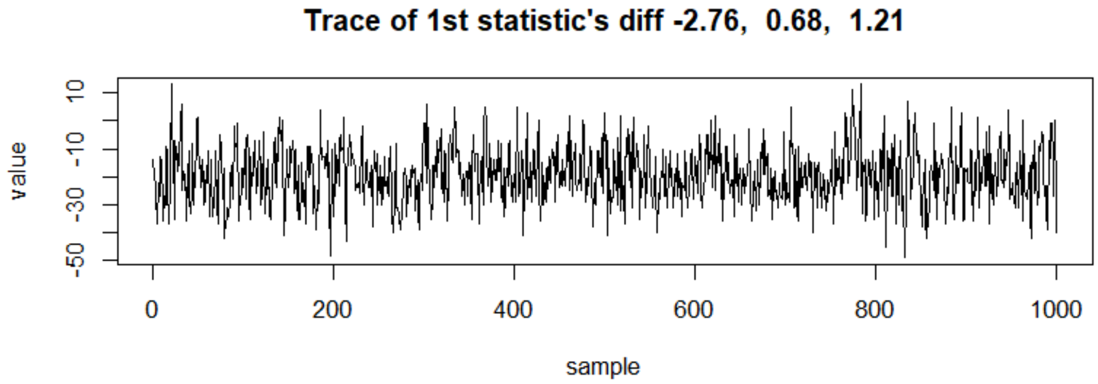
(2)ii. Based on the simulations, do you think that the suggested values of the parameters are plausible estimates? Argue for your answer.

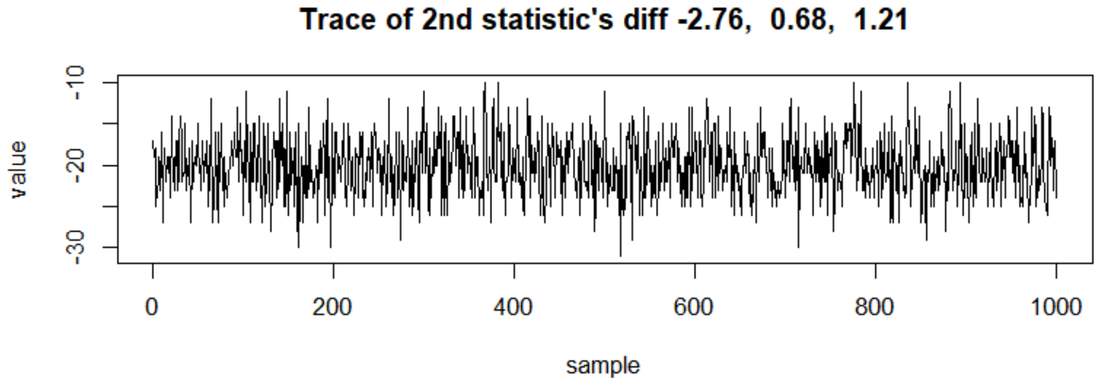
Once we got the returned value from previous part, we made plots of the trace of the difference between the simulated networks' statistics and the observed network's statistics(similar to the idea of library(coda)), we plot differences using the following function(we made three plot, one plot per statistic):

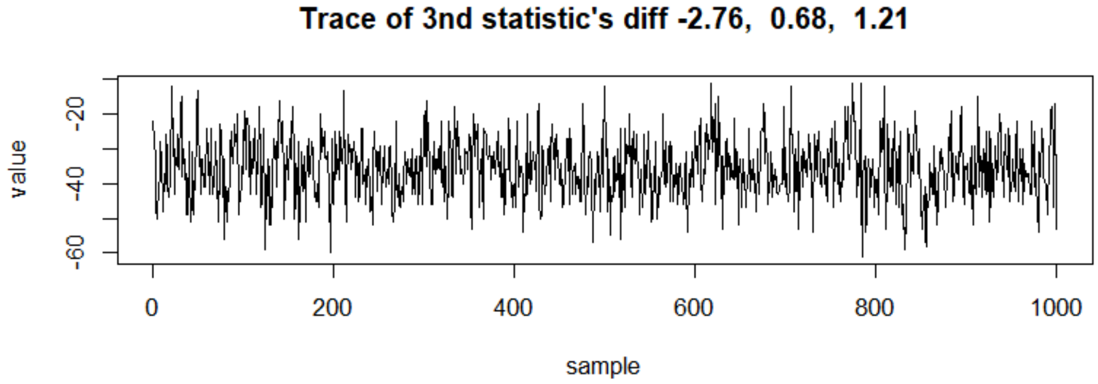
```
plot_trace <- function(MC_res,obs_statistics,mc_param){
  par(xpd = NA,mfrow = c(3, 1))
  plot(MC_res$statSim[,1]-obs_statistics[1],type="l", main=paste("Trace of 1st statistic's
diff",mc_param),xlab="sample",ylab="value")
  plot(MC_res$statSim[,2]-obs_statistics[2],type="l", main=paste("Trace of 2nd statistic's
diff",mc_param),xlab="sample",ylab="value")
  plot(MC_res$statSim[,3]-obs_statistics[3],type="l", main=paste("Trace of 3nd statistic's
diff",mc_param),xlab="sample",ylab="value")
}
```

The trace plots is shown as the following:

```
MC_res1 <- MarkovChain(zero_matrix, gender, -2.76,  0.68,  1.21) # same
plot_trace(MC_res1, obs_statistics, "-2.76,  0.68,  1.21")
```







We could deduce from the trace plot that, in this case, there are fewer edges than the observed network(difference around -20), fewer mutual dyads(around -20) and fewer gender homophily(around -30). So it is not an appropriate guess of the values of the parameters, we should change the value in order to make better simulation.

(3) Guess better estimates of θ1, θ2 and θ3 based on your analysis in (2). Describe the procedure you used to obtain the guessed values. (Please use the code and the analysis in (1), and (2). Obtaining better values using the ergm function is not considered a valid solution.)

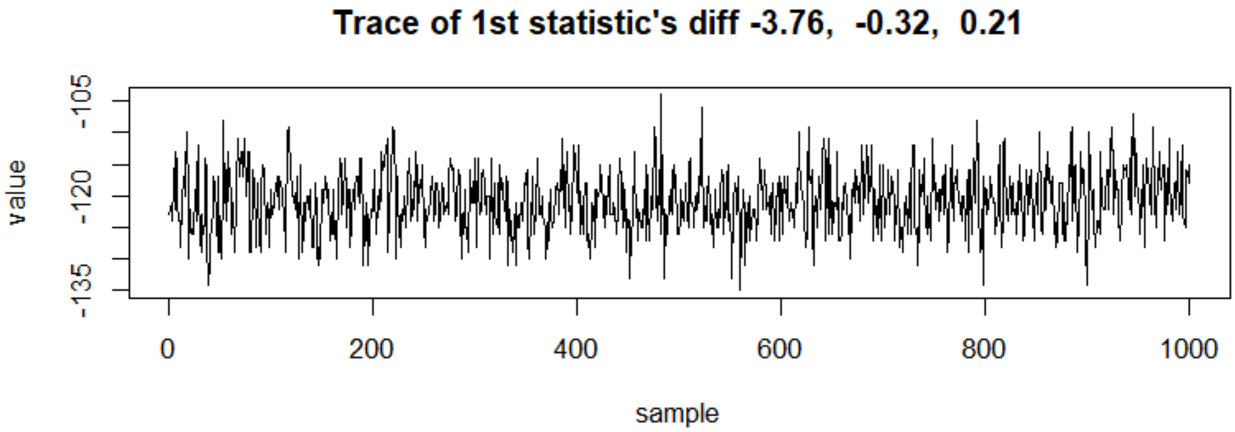
From the previous analysis, we know that the initial guess was not satisfactory. We decided to play with the three parameters, we first tried brute force way, where we tried to change the 3 parameters by one unit(increase or decrease each parameter by one), hence in total 8 additional simulations were made, and we plot the results.

Thanks to the fortune we have, we found out that the configuration c(-3.76, 1.68, 2.21) gave a good estimate of the parameter for the data we have. Compared with the observed value, theta1 is decreased by one unit(from -2.76 to -3.76), theta2 and theta 3 are increased by one unit(from 0.68 and 1.21 to 1.68 and 2.21), the reasoning that we could give would be the following: Since the social network tends to be sparse, and from the previous experience we always find that the parameter of number of edges is usually negative, when we decrease its value, so the expected density could increase. Since coefficient of density could influence the number of mutuals and number of gender homophily, the increase/decrease of the other parameters could not be easily deduced, hence by trial and error(in our case we list all possible combinations of step size 1), we may hopefully arrive at a feasible set of parameters' estimate.

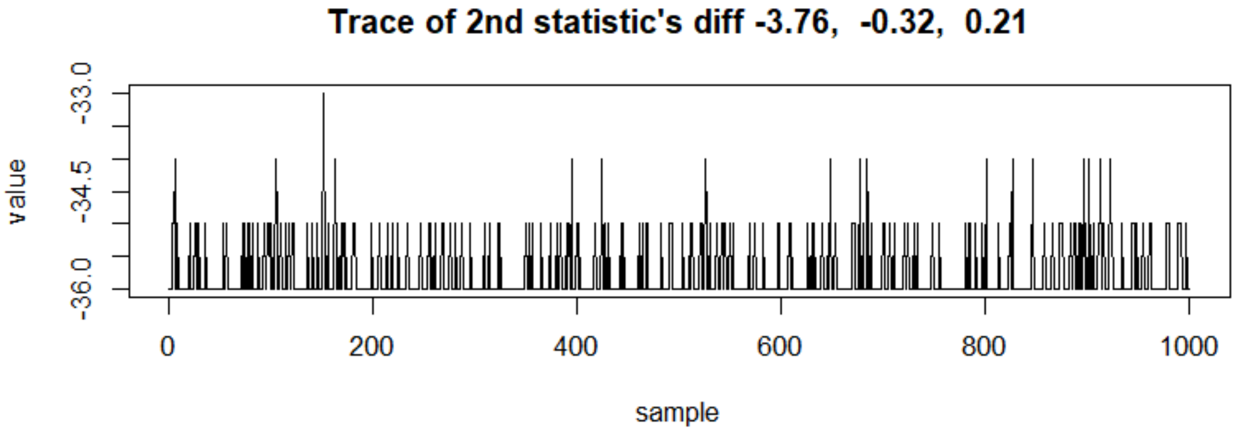
Code for all combination that we tried:

```
MC_res1 <- MarkovChain(zero_matrix, gender, -2.76,  0.68,  1.21) # same
MC_res2 <-MarkovChain(zero_matrix, gender, -3.76,  -0.32,  0.21) # -1,-1,-1
MC_res3 <-MarkovChain(zero_matrix, gender, -3.76,  -0.32,  1.21) #-1,-1,+1
MC_res4 <-MarkovChain(zero_matrix, gender, -3.76,  1.68,  0.21)  # -1, +1, -1
MC_res5 <-MarkovChain(zero_matrix, gender, -3.76,  1.68,  2.21) # -1, +1, +1
MC_res6 <-MarkovChain(zero_matrix, gender, -1.76,  -0.32,  0.21)  # +1, -1, -1
MC_res7 <-MarkovChain(zero_matrix, gender, -1.76,  -0.32,  2.21)  # +1, -1, +1
MC_res8 <-MarkovChain(zero_matrix, gender, -1.26,  1.68,  0.21)  # +1, +1, -1
MC_res9 <-MarkovChain(zero_matrix, gender, -1.26,  1.68,  2.21)  # +1, +1, +1

plot_trace(MC_res1, obs_statistics, "-2.76,  0.68,  1.21")
plot_trace(MC_res2, obs_statistics, "-3.76,  -0.32,  0.21")
plot_trace(MC_res3, obs_statistics, "-3.76,  -0.32,  1.21" )
plot_trace(MC_res4, obs_statistics, "-3.76,  1.68,  0.21")
plot_trace(MC_res5, obs_statistics, "-3.76,  1.68,  2.21")
plot_trace(MC_res6, obs_statistics, "-1.76,  -0.32,  0.21")
plot_trace(MC_res7, obs_statistics, "-1.76,  -0.32,  2.21")
plot_trace(MC_res8, obs_statistics, "-1.26,  1.68,  0.21")
plot_trace(MC_res9, obs_statistics, "-1.26,  1.68,  2.21")
```

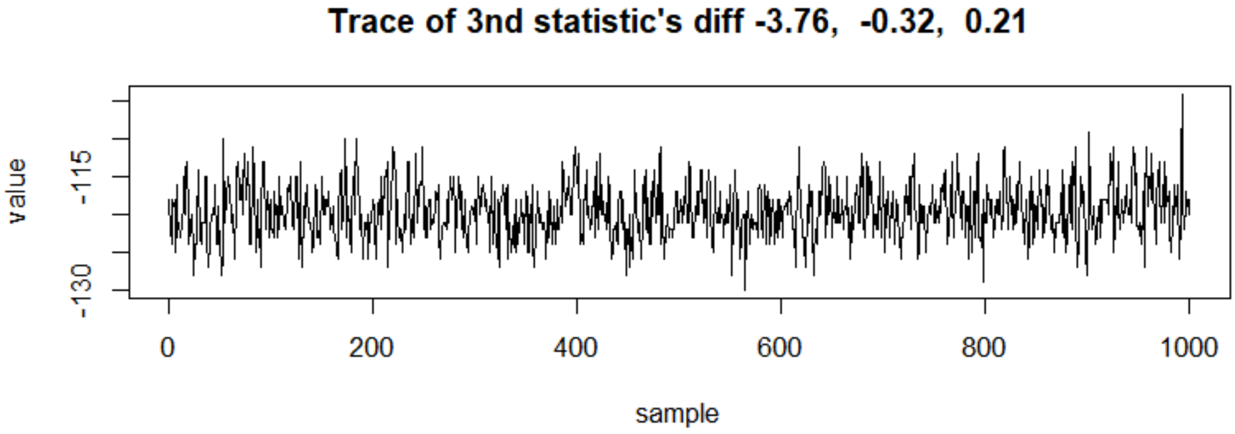
Plots:



Trace of 1st statistic's diff -3.76, -0.32, 0.21

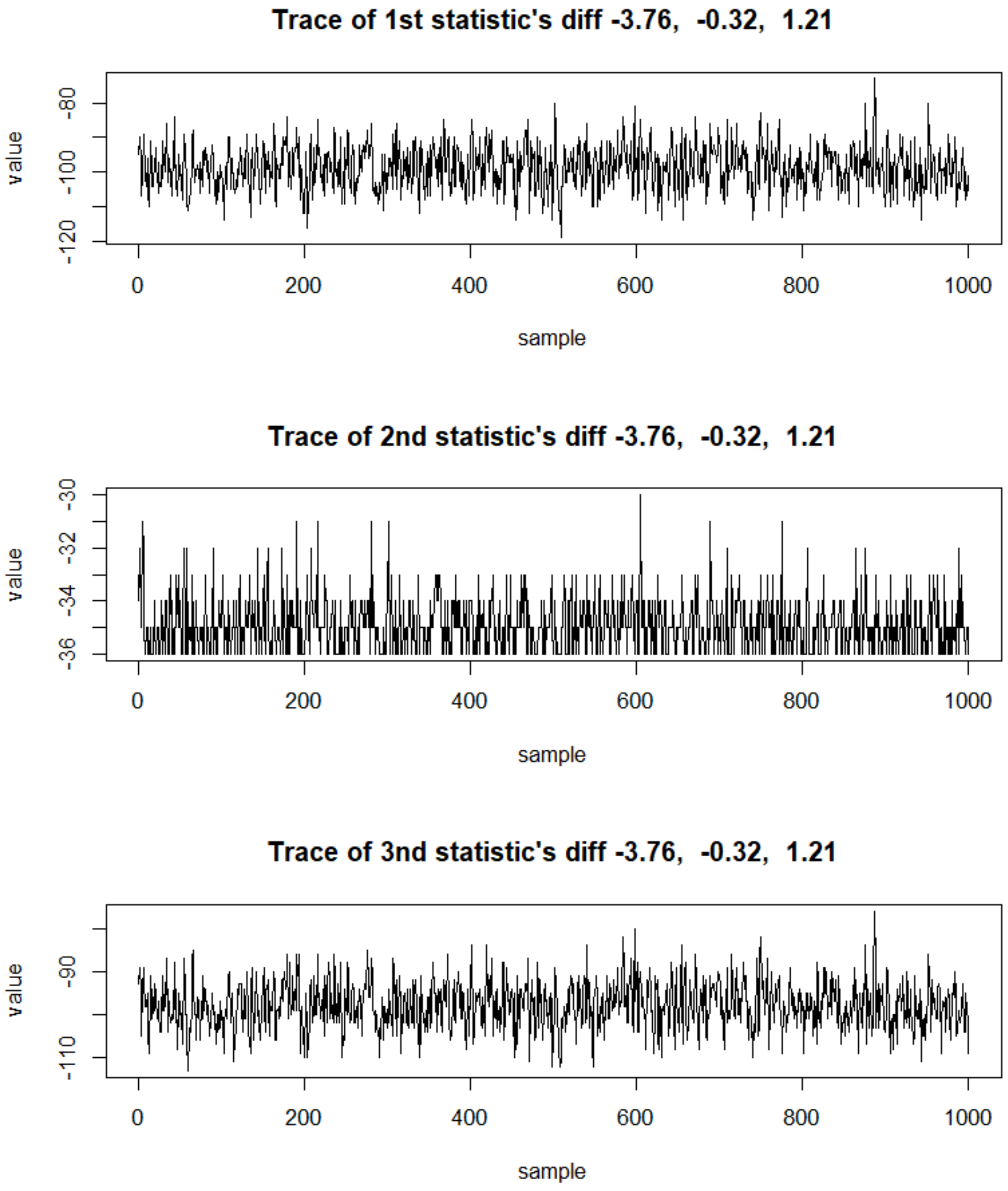


Trace of 2nd statistic's diff -3.76, -0.32, 0.21



Trace of 3nd statistic's diff -3.76, -0.32, 0.21

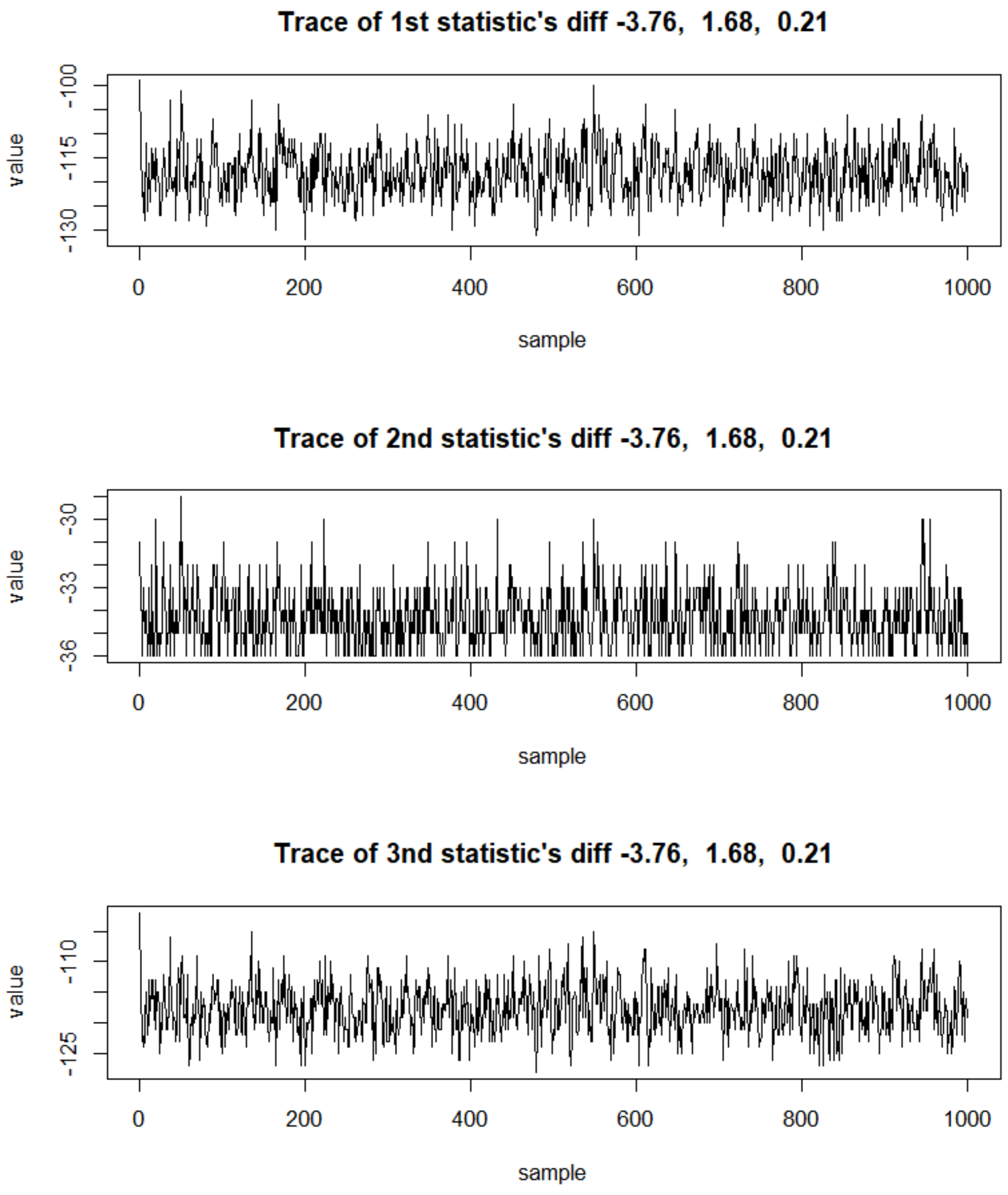
(plot for try 2)


Trace of 1st statistic's diff -3.76, -0.32, 1.21

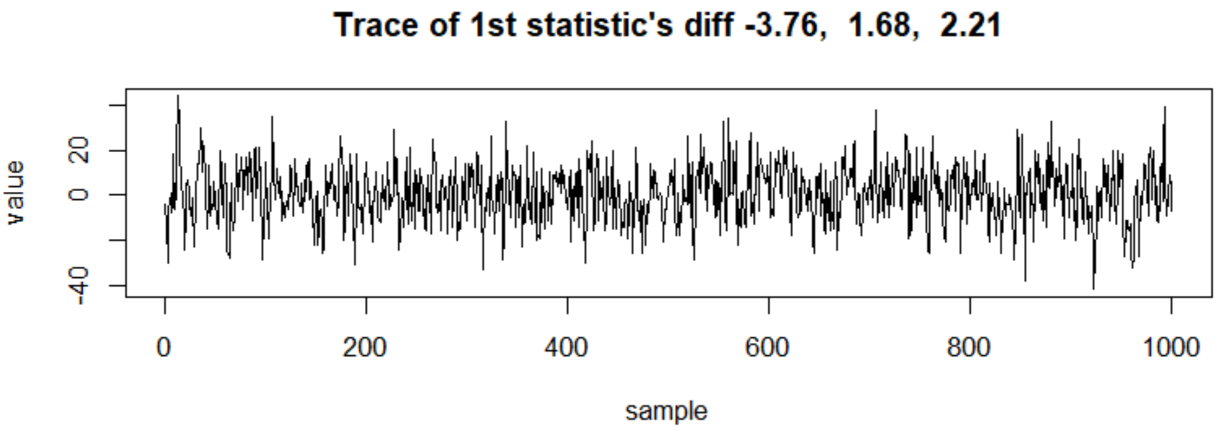
Trace of 2nd statistic's diff -3.76, -0.32, 1.21

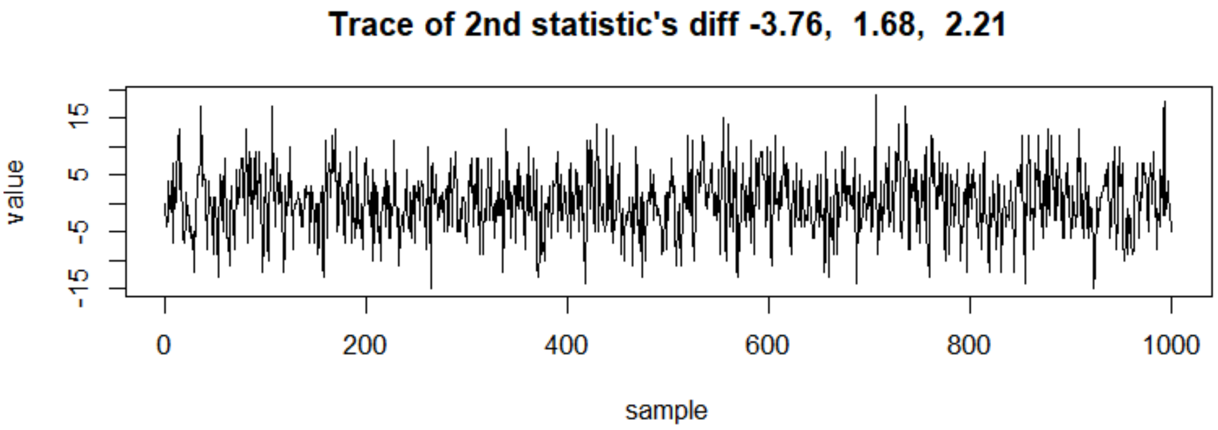
Trace of 3nd statistic's diff -3.76, -0.32, 1.21

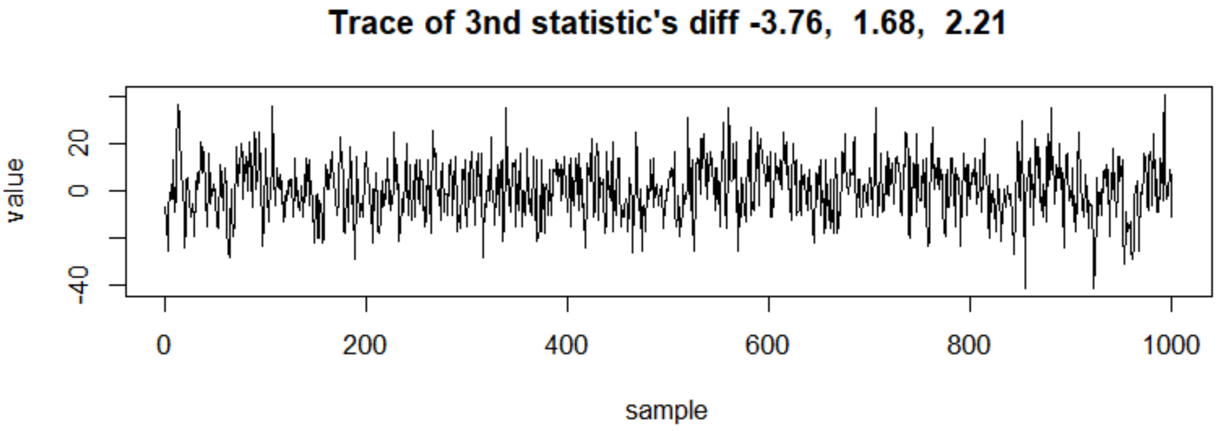
(plot for try 3)



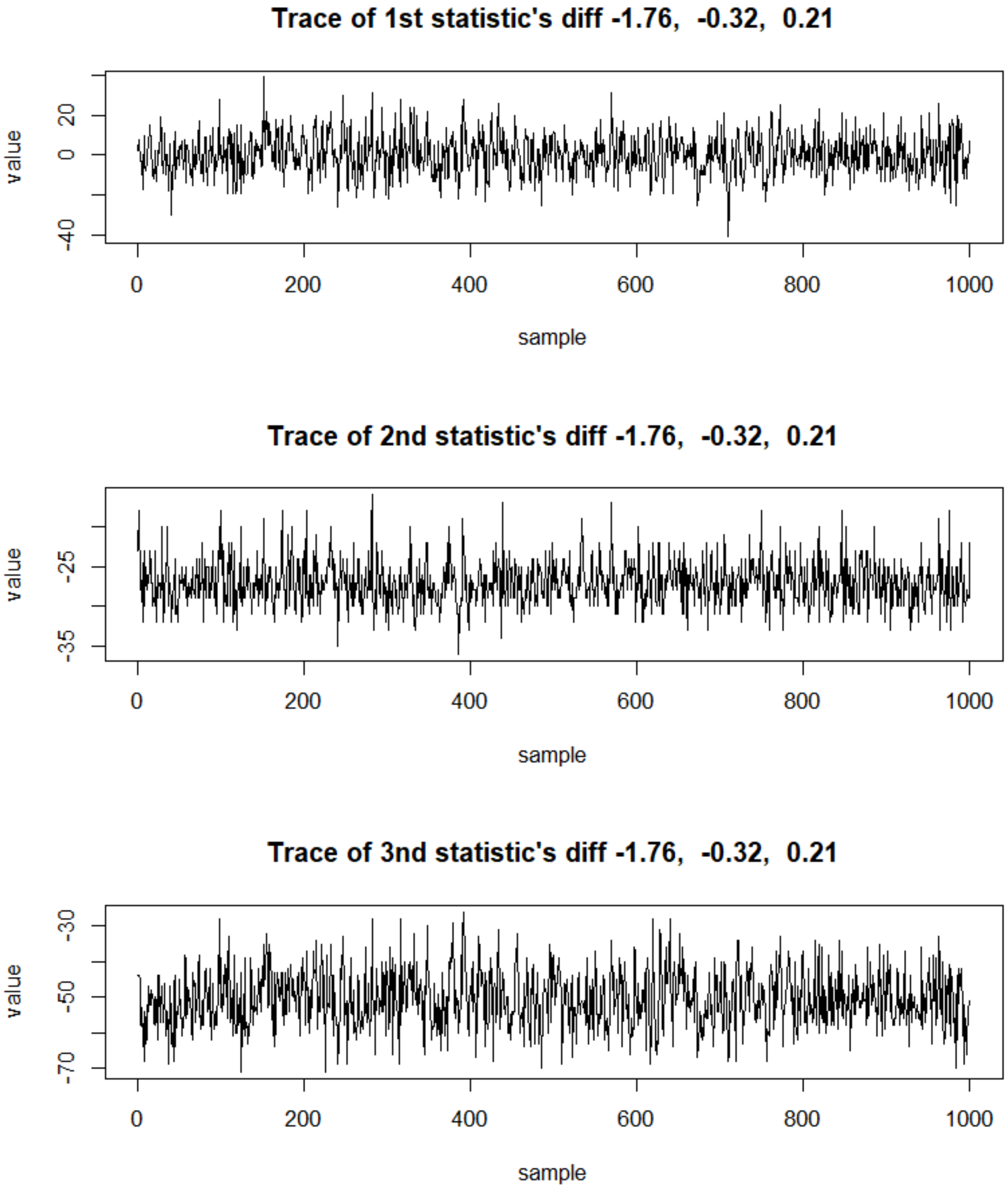
(Plot for try4)







(plot for try 5, correct one! We could see that the difference are fluctuating around zero)

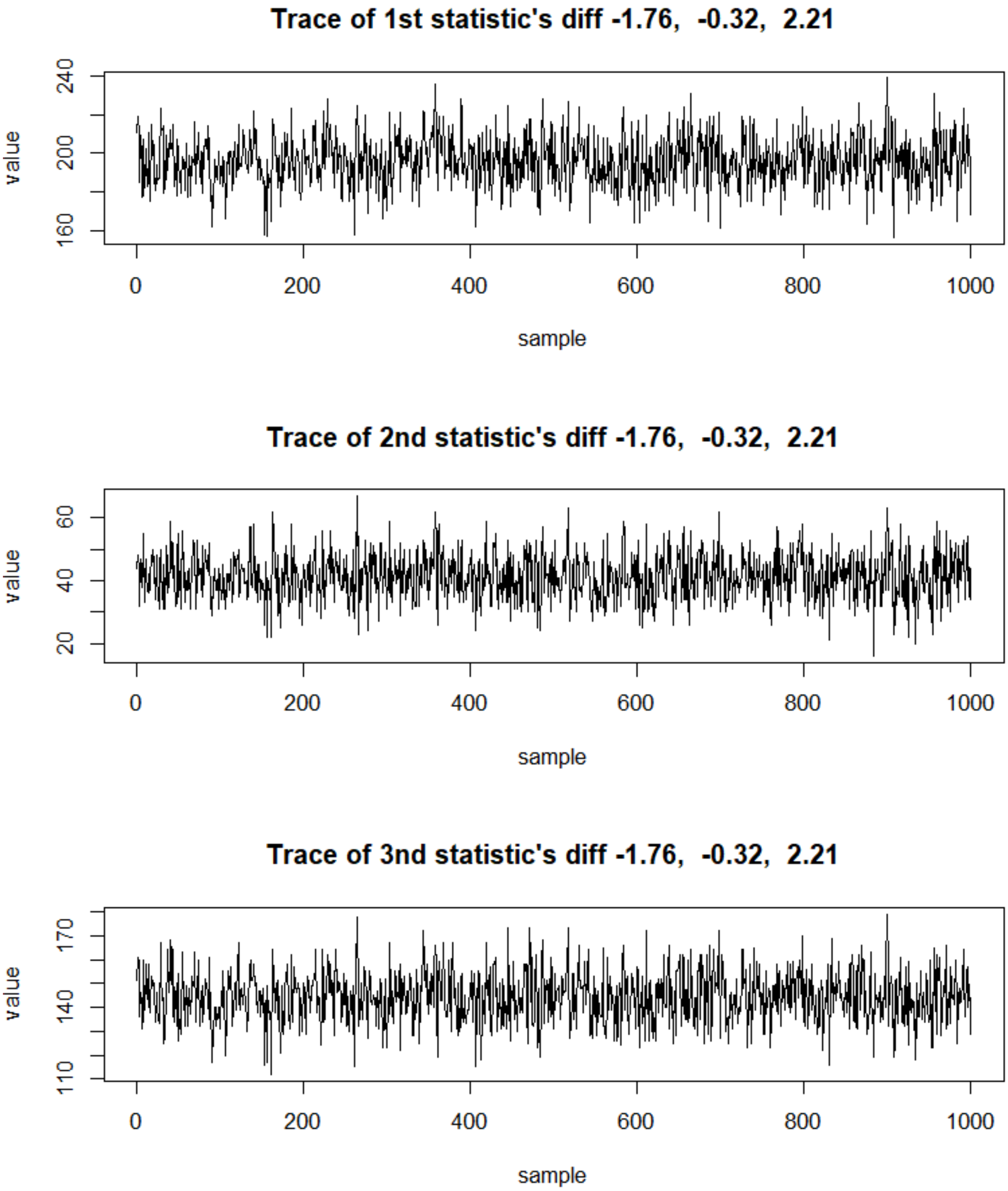


Trace of 1st statistic's diff -1.76, -0.32, 0.21

Trace of 2nd statistic's diff -1.76, -0.32, 0.21

Trace of 3nd statistic's diff -1.76, -0.32, 0.21

(Plot for try 6)

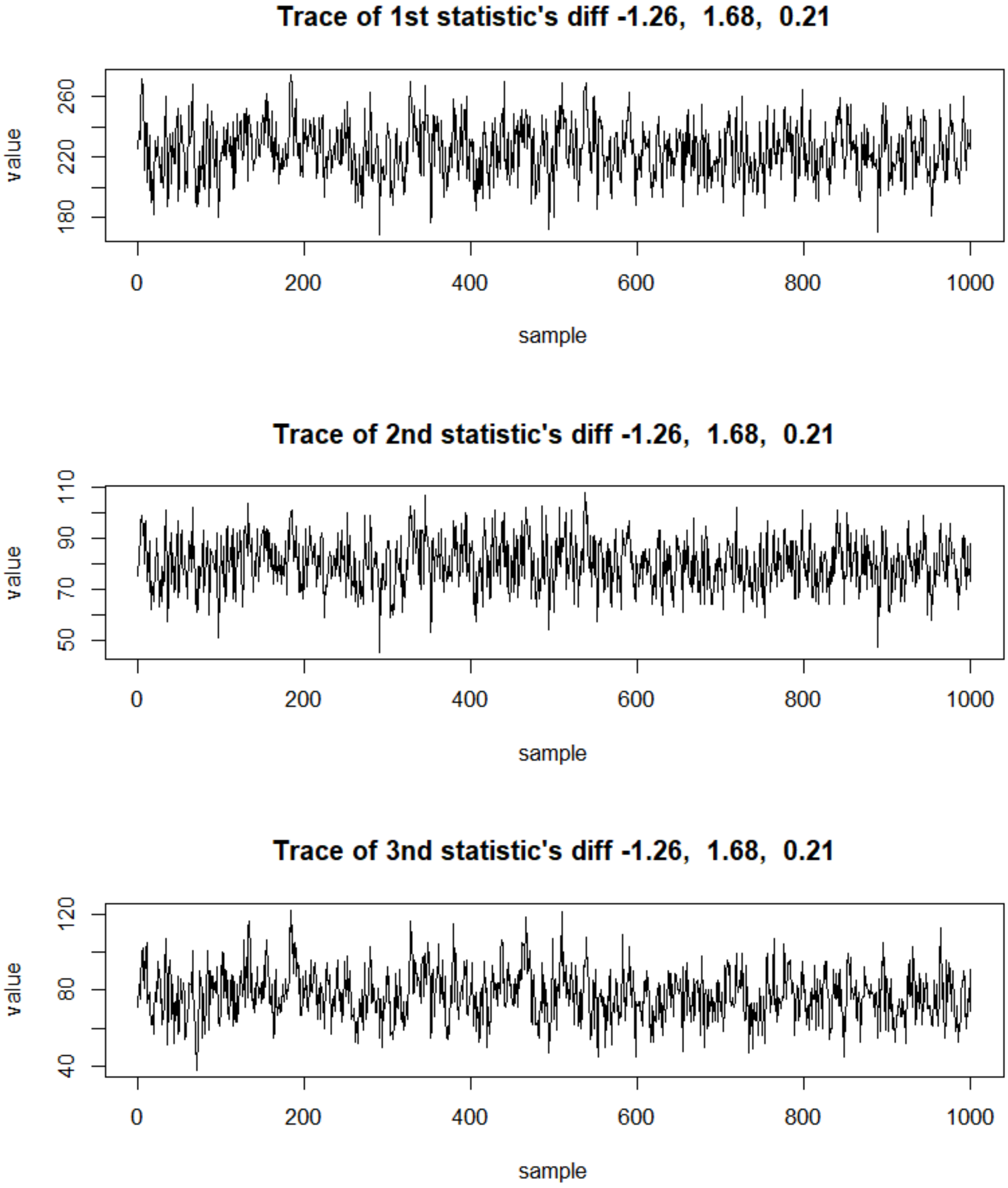


Trace of 1st statistic's diff -1.76, -0.32, 2.21

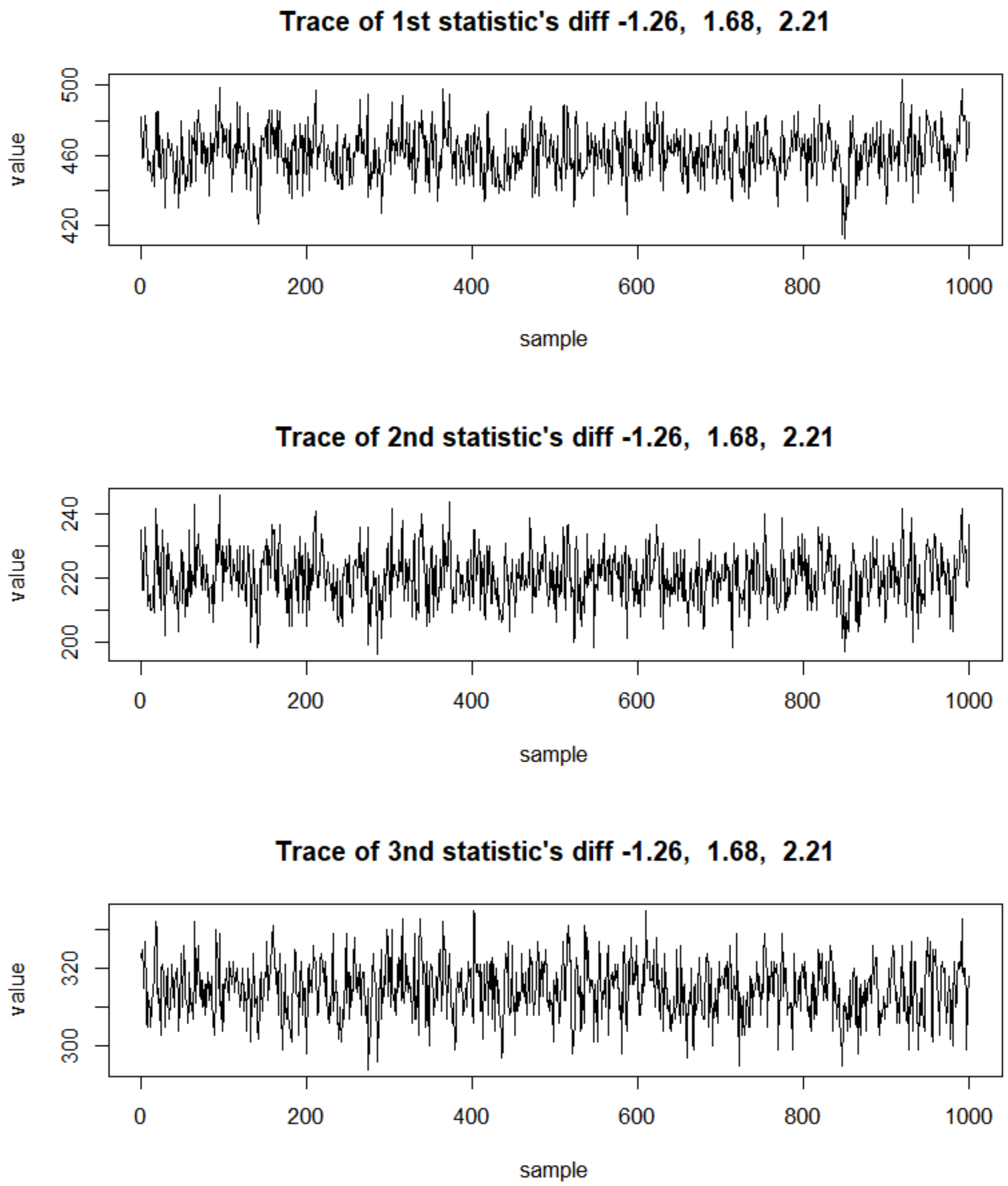
Trace of 2nd statistic's diff -1.76, -0.32, 2.21

Trace of 3nd statistic's diff -1.76, -0.32, 2.21

(Plot for try 7)



(Plot for try 8)



Trace of 1st statistic's diff -1.26, 1.68, 2.21

Trace of 2nd statistic's diff -1.26, 1.68, 2.21

Trace of 3nd statistic's diff -1.26, 1.68, 2.21

(Plot for try 9)

Task 3: Estimation and interpretation of an ERGM

Now we want to analyze the high-school friendship network in friend net.Rda using ERGM.

(1) Estimate an ERGM with an edge and a gender homophily parameter. Compute the conditional probability of observing a tie between two students i and j having the same gender and interpret the result.

Solution:

We create a model using function ergm(), then we pass the parameter “formula” which is “friend_net ~ edges + nodematch(“sex”)” which means that we want to estimate ERGM model with edge and gender homophily parameter. Then we summarize the model using command “summary()”.

As it is shown in the summary of the model, the estimates of parameters of “edges” and “nodematch.sex” are -3.6636 and 2.6762 respectively. Both are significant, the coefficient of edge parameter indicates that the network is sparse, the coefficient of gender homophily suggests that there is evidence for gender homophily, and it is supported by the data.

As for calculation of the conditional probability. We first deduce the change of statistics, when a tie between two students i and j is observed, the number of edges increase by one and so does the gender homophily, hence the vector of change of statistics is c(1,1), we calculate the exp(c(theta1,theta2)) to get the value of odds, and using the formula odds/(1+odds) to finally get the required probability, which is 0.2714286 as shown in the following part.

Code and execution results:

```
> #task 3.1
> set.seed(1)
> #+gwesp(decay = 0.3, fixed = TRUE)
> #This we dont include in the model since the assignment does not explicitly
demand it
> model3.1 <- ergm(friend_net ~ edges + nodematch("sex"))
Starting maximum pseudolikelihood estimation (MPLE):
Obtaining the responsible dyads.
Evaluating the predictor and response matrix.
Maximizing the pseudolikelihood.
Finished MPLE.
Evaluating log-likelihood at the estimate.
> summary(model3.1)
Call:
ergm(formula = friend_net ~ edges + nodematch("sex"))

Maximum Likelihood Results:
```

```
              Estimate Std. Error MCMC % z value Pr(>|z|)
edges          -3.6636     0.3053      0 -11.998   <1e-04 ***
nodematch.sex   2.6762     0.3218      0   8.316   <1e-04 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     Null Deviance: 1289.3  on 930  degrees of freedom
 Residual Deviance:  675.9  on 928  degrees of freedom

AIC: 679.9  BIC: 689.5  (Smaller is better. MC Std. Err. = 0)
> theta1 <- model3.1$coef[1]
> theta2 <- model3.1$coef[2]
> # Computing the probability of a tie not reciprocating an existing tie is
> oSameSex <- exp(theta1+theta2)
> pSameSex <- oSameSex / (1 + oSameSex)
> pSameSex
    edges
0.2714286
```

(2) Add variables to the ERGM specified in (1) to test simultaneously the following hypotheses:
i. A tie is more likely between students when it reciprocates a friendship nomination (reciprocity).
ii. A tie is more likely between students when it closes a transitive two-path (transitivity).
iii. A tie is less likely when the sender has a higher out-degree (social activity)
iv. A tie is more likely when the receiver has a higher in-degree (popularity).

Solution:
In this task we just need to, on top of what we have already included in (1), incorporate more terms into the function ergm()

In order to test simultaneously the required hypothesis, besides the terms that we already included(`friend_net ~ edges + nodematch("sex")`), we include term (`mutual`) for hypothesis 1 which was reciprocity, we include term (`gwesp(decay = 0.3), fixed =TRUE`) for hypothesis 2 which was the closure of transitive two-path, the reason why we don’t include term “ttriple” or “twopath” is because the near degeneracy phenomena that could lead the model unable to estimate correctly, hence we include the “geometrically weighted edgewise shared partner” term for hypothesis 2.

For hypothesis 3, after a long discussion with the professors and other classmates(after going through exogenous variables and other options), we finally chose the gwodegree(decay = 0.3, fixed = TRUE) that take into account the outdegree distribution to measure the social activity of sender in order to test our hypothesis. Similarly, for hypothesis 4 we chose gwidegree(decay = 0.3, fixed = TRUE), the choose of decay parameter was based on trail and error and we consider it is an appropriate value.

Code and results:

```
#task 3.2
> set.seed(1)
> model3.2 <- ergm(friend_net ~ edges + nodematch("sex") + mutual
+                 + gwesp(decay = 0.3, fixed = TRUE)
+                 + gwodegree(decay = 0.3, fixed = TRUE) + gwidegree(decay =
0.3, fixed = TRUE) )
Starting maximum pseudolikelihood estimation (MPLE):
Obtaining the responsible dyads.
Evaluating the predictor and response matrix.
Maximizing the pseudolikelihood.
Finished MPLE.
Starting Monte Carlo maximum likelihood estimation (MCMLE):
Iteration 1 of at most 60:
Optimizing with step length 0.1975.
The log-likelihood improved by 1.9939.
Estimating equations are not within tolerance region.
Iteration 2 of at most 60:
Optimizing with step length 0.3887.
The log-likelihood improved by 2.3794.
Estimating equations are not within tolerance region.
Iteration 3 of at most 60:
Optimizing with step length 0.5609.
The log-likelihood improved by 2.2806.
Estimating equations are not within tolerance region.
Iteration 4 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.4336.
Estimating equations are not within tolerance region.
Iteration 5 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.1564.
Estimating equations are not within tolerance region.
Iteration 6 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.0490.
Convergence test p-value: 0.9440. Not converged with 99% confidence;
increasing sample size.
Iteration 7 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.1015.
```

```
Estimating equations are not within tolerance region.
Iteration 8 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.0472.
Convergence test p-value: 0.7066. Not converged with 99% confidence;
increasing sample size.
Iteration 9 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.0266.
Convergence test p-value: 0.0782. Not converged with 99% confidence;
increasing sample size.
Iteration 10 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.0082.
Convergence test p-value: 0.0365. Not converged with 99% confidence;
increasing sample size.
Iteration 11 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.0147.
Convergence test p-value: 0.0404. Not converged with 99% confidence;
increasing sample size.
Iteration 12 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.0499.
Convergence test p-value: 0.0217. Not converged with 99% confidence;
increasing sample size.
Iteration 13 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.0676.
Convergence test p-value: 0.0208. Not converged with 99% confidence;
increasing sample size.
Iteration 14 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.0032.
Convergence test p-value: 0.0001. Converged with 99% confidence.
Finished MCMLE.
Evaluating log-likelihood at the estimate. Fitting the dyad-independent
submodel...
Bridging between the dyad-independent submodel and the full model...
Setting up bridge sampling...
Using 16 bridges: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 .
Bridging finished.

This model was fit using MCMC.  To examine model diagnostics and check for
degeneracy, use the
mcmc.diagnostics() function.
> summary(model3.2)
```

```
Call:
ergm(formula = friend_net ~ edges + nodematch("sex") + mutual +
    gwesp(decay = 0.3, fixed = TRUE) + gwodegree(decay = 0.3,
    fixed = TRUE) + gwidegree(decay = 0.3, fixed = TRUE))

Monte Carlo Maximum Likelihood Results:

                    Estimate Std. Error MCMC % z value Pr(>|z|)
edges                -5.7517     0.4154      0 -13.847   <1e-04 ***
nodematch.sex         0.9522     0.2019      0   4.717   <1e-04 ***
mutual                0.7871     0.3399      0   2.316   0.0206 *
gwesp.OTP.fixed.0.3   2.1063     0.3172      0   6.639   <1e-04 ***
gwodeg.fixed.0.3      1.3862     0.7269      0   1.907   0.0565 .
gwideg.fixed.0.3      2.6929     1.0683      0   2.521   0.0117 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     Null Deviance: 1289.3  on 930  degrees of freedom
 Residual Deviance:  567.2  on 924  degrees of freedom

AIC: 579.2  BIC: 608.2  (Smaller is better. MC Std. Err. = 0.5104)
```

(3) Estimate the ERGM specified in (2) and comment on the convergence of the algorithm.

We fitted the model3.2(in the previous section), some part of the output shows that:

```
Iteration 14 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.0032.
Convergence test p-value: 0.0001. Converged with 99% confidence.
Finished MCMLE.
```

The model converges in the 14-th iteration, with convergence test p-value 0.0001, the MCMLE algorithm converged with 0.99 confidence.

To further analyse the convergence of the algorithm, we looked at the diagnostics of MCMC simulation, which is shown as follows:

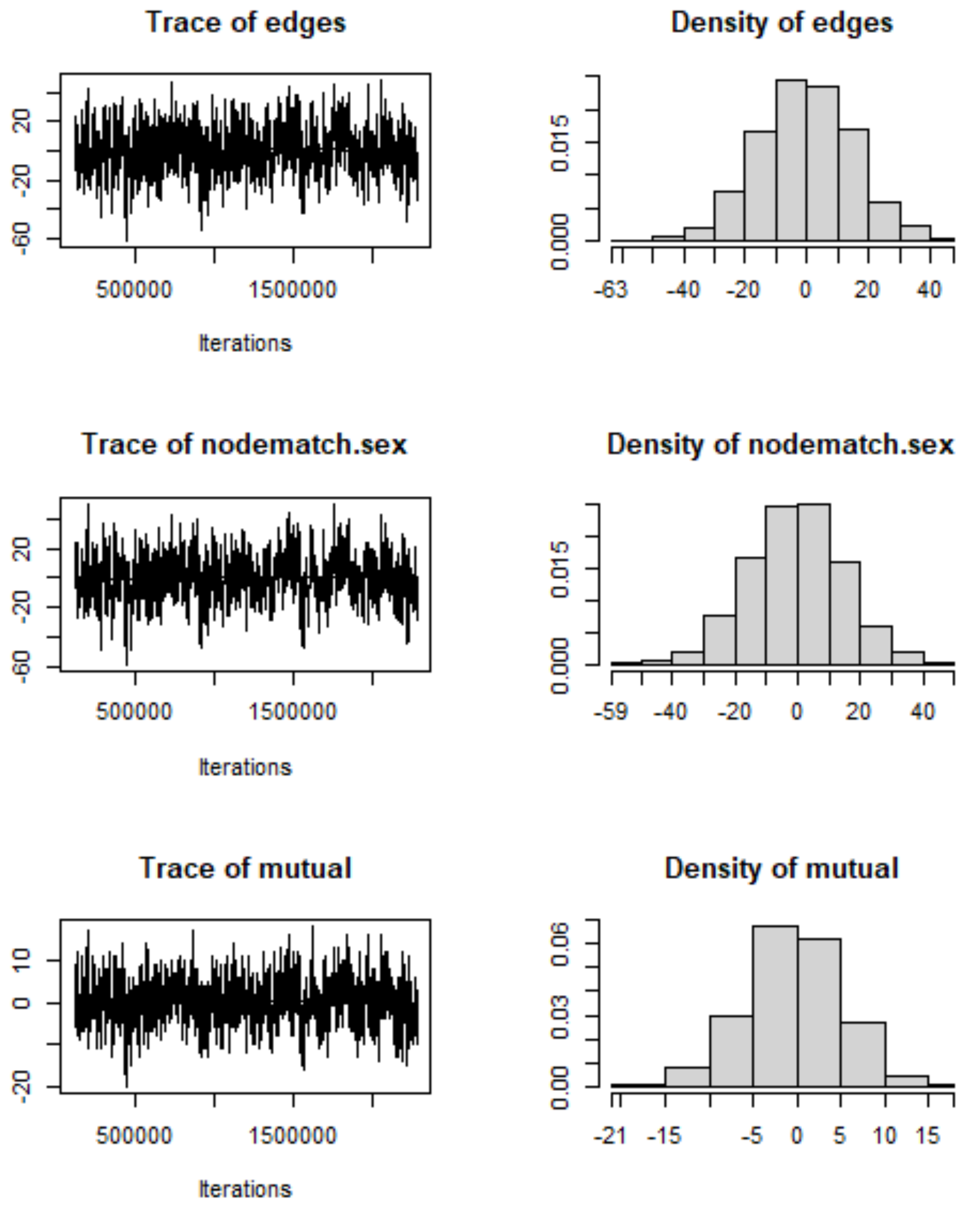
We could observe that the trace of the 6 statistics that we included randomly fluctuate around zero and there is no trend observed, which mean that the difference between the simulated statistics and the observed statistics are close to zero(this could also be reflected by the histogram that indicates the distribution is centered around 0 approximately), thereby suggests that the MCMC chain is mixing/converging well, it is heading into a region of network space where the observed network is located.

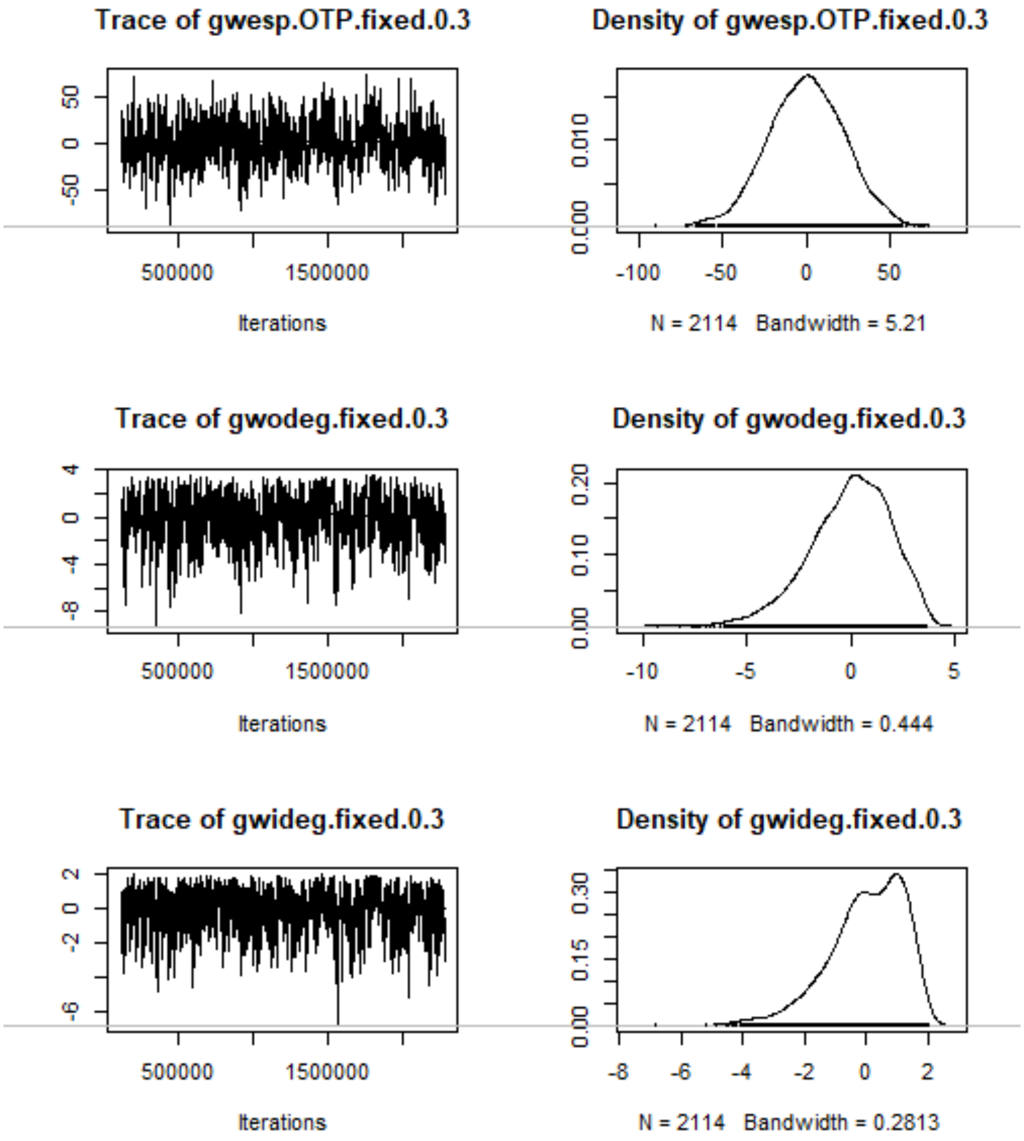
The plot and the output is the following:

```
> # ERGM diagnostics and fit -------------------------------------------------
> ## Model convergence -------------------------------------------------------
> mcmc.diagnostics(model3.2)
Sample statistics summary:

Iterations = 114688:2278400
Thinning interval = 1024
Number of chains = 1
Sample size per chain = 2114

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

                         Mean     SD Naive SE Time-series SE
edges                0.004257 15.505  0.33722        0.86075
nodematch.sex       -0.150426 14.931  0.32474        0.94810
mutual               0.131977  5.411  0.11769        0.30979
gwesp.OTP.fixed.0.3  0.176575 22.728  0.49431        1.30074
gwodeg.fixed.0.3    -0.067380  1.937  0.04212        0.10131
gwideg.fixed.0.3     0.006224  1.227  0.02668        0.06203

2. Quantiles for each variable:

                       2.5%      25%    50%     75%  97.5%
edges               -30.175 -11.0000 0.0000 11.0000 31.000
nodematch.sex       -29.000 -10.0000 0.0000 10.0000 28.000
mutual              -11.000  -3.7500 0.0000  4.0000 11.000
gwesp.OTP.fixed.0.3 -44.230 -14.8219 0.3377 15.6540 44.897
gwodeg.fixed.0.3     -4.392  -1.2673 0.1106  1.3527  3.090
gwideg.fixed.0.3     -2.950  -0.6792 0.1615  0.9743  1.721


Are sample statistics significantly different from observed?
                 edges nodematch.sex    mutual gwesp.OTP.fixed.0.3
gwodeg.fixed.0.3 gwideg.fixed.0.3     (Omni)
diff.      0.004257332    -0.1504257 0.1319773           0.1765751      -
0.06738031      0.006224153         NA
test stat. 0.004946089    -0.1586599 0.4260261           0.1357502      -
0.66510486      0.100342050 10.0602778
P-val.     0.996053608     0.8739368 0.6700888           0.8920188
0.50598343      0.920072775  0.1255486

Sample statistics cross-correlations:
```

```
                        edges nodematch.sex    mutual gwesp.OTP.fixed.0.3
gwodeg.fixed.0.3 gwideg.fixed.0.3
edges               1.0000000     0.9237506 0.7738512           0.9833433
0.6058660        0.6025089
nodematch.sex       0.9237506     1.0000000 0.7881279           0.9374969
0.5979393        0.5802483
mutual              0.7738512     0.7881279 1.0000000           0.8102189
0.5414992        0.4954729
gwesp.OTP.fixed.0.3 0.9833433     0.9374969 0.8102189           1.0000000
0.5589826        0.5675854
gwodeg.fixed.0.3    0.6058660     0.5979393 0.5414992           0.5589826
1.0000000        0.5051831
gwideg.fixed.0.3    0.6025089     0.5802483 0.4954729           0.5675854
0.5051831        1.0000000

Sample statistics auto-correlation:
Chain 1 
             edges nodematch.sex    mutual gwesp.OTP.fixed.0.3
gwodeg.fixed.0.3 gwideg.fixed.0.3
Lag 0    1.0000000     1.0000000 1.0000000           1.0000000
1.0000000        1.0000000
Lag 1024 0.6150289     0.6635840 0.5206011           0.6087970
0.4498078        0.3840673
Lag 2048 0.4372993     0.4882484 0.3700470           0.4301275
0.3346476        0.2898239
Lag 3072 0.2969660     0.3588141 0.2873768           0.2935866
0.2694628        0.2436519
Lag 4096 0.2318648     0.3011238 0.2393146           0.2249724
0.2271929        0.1976040
Lag 5120 0.2153891     0.2799721 0.2176596           0.2132026
0.2214534        0.1730189

Sample statistics burn-in diagnostic (Geweke):
Chain 1 

Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5 

              edges       nodematch.sex              mutual
gwesp.OTP.fixed.0.3    gwodeg.fixed.0.3    gwideg.fixed.0.3 
       -1.382501833        -0.537013449         0.001750347        -
1.167366285        -0.133485635         0.533665532 

Individual P-values (lower = worse):
              edges       nodematch.sex              mutual
gwesp.OTP.fixed.0.3    gwodeg.fixed.0.3    gwideg.fixed.0.3 
          0.1668177           0.5912583           0.9986034
0.2430625           0.8938093           0.5935730 
Joint P-value (lower = worse):  0.01411383 
```

```
Note: MCMC diagnostics shown here are from the last round of simulation,
prior to computation of final parameter
  estimates. Because the final estimates are refinements of those used for
this simulation run, these
  diagnostics may understate model performance. To directly assess the
performance of the final model on
  in-model statistics, please use the GOF command: gof(ergmFitObject,
GOF=~model).
```





(4) Evaluate the goodness of fit of the model according to four different criteria.

In order to perform this task, we use the function gof()

```
> ## Goodness of fit ---------------------------------------------------------
> model3.2gof <- gof(model3.2)
> model3.2gof

Goodness-of-fit for in-degree
```

```
          obs min mean max MC p-value
idegree0    1   0 0.63   4       0.96
idegree1    1   0 2.49  10       0.56
idegree2    3   0 4.36   9       0.78
idegree3    6   0 4.90  10       0.80
idegree4    6   1 4.68   9       0.70
idegree5    4   0 3.96  10       1.00
idegree6    2   0 2.96   7       0.78
idegree7    5   0 2.43   7       0.14
idegree8    1   0 1.71   6       0.98
idegree9    0   0 0.98   4       0.68
idegree10   2   0 0.84   4       0.34
idegree11   0   0 0.42   3       1.00
idegree12   0   0 0.28   2       1.00
idegree13   0   0 0.19   2       1.00
idegree14   0   0 0.04   1       1.00
idegree15   0   0 0.11   2       1.00
idegree16   0   0 0.01   1       1.00
idegree17   0   0 0.01   1       1.00

Goodness-of-fit for out-degree

          obs min mean max MC p-value
odegree0    2   0 1.75   7       1.00
odegree1    1   0 2.44   8       0.64
odegree2    6   1 3.89   7       0.34
odegree3    6   0 4.18  10       0.44
odegree4    4   1 4.39   9       1.00
odegree5    3   0 3.83   9       0.94
odegree6    2   0 2.88   7       0.86
odegree7    0   0 2.66   8       0.14
odegree8    1   0 1.88   6       0.80
odegree9    2   0 1.12   4       0.64
odegree10   2   0 0.76   3       0.38
odegree11   0   0 0.52   3       1.00
odegree12   2   0 0.25   2       0.06
odegree13   0   0 0.24   2       1.00
odegree14   0   0 0.09   1       1.00
odegree15   0   0 0.06   1       1.00
odegree16   0   0 0.03   1       1.00
odegree17   0   0 0.02   1       1.00
odegree18   0   0 0.01   1       1.00

Goodness-of-fit for edgewise shared partner

          obs min  mean max MC p-value
```

```
esp.OTP0   11   2  9.78  20       0.86
esp.OTP1   44  23 45.72  70       0.88
esp.OTP2   35  28 46.19  68       0.30
esp.OTP3   16  11 25.21  45       0.16
esp.OTP4   13   0 10.13  27       0.54
esp.OTP5   12   0  3.84  14       0.02
esp.OTP6   11   0  1.63   7       0.00
esp.OTP7    2   0  0.42   5       0.18
esp.OTP8    0   0  0.08   1       1.00
esp.OTP9    0   0  0.07   2       1.00
esp.OTP10   0   0  0.01   1       1.00
esp.OTP12   0   0  0.01   1       1.00

Goodness-of-fit for minimum geodesic distance

    obs min   mean max MC p-value
1   144 111 143.09 180       1.00
2   173 158 281.52 380       0.02
3   138  65 185.18 321       0.50
4    76   4  99.00 170       0.58
5    70   0  40.61 104       0.36
6    31   0  14.07  60       0.36
7     1   0   4.21  39       0.86
8     0   0   0.98  25       1.00
9     0   0   0.28  19       1.00
10    0   0   0.03   3       1.00
Inf 297   0 161.03 496       0.38

Goodness-of-fit for model statistics

                          obs       min      mean       max MC p-value
edges               144.00000 111.00000 143.09000 180.00000       1.00
nodematch.sex       133.00000 104.00000 131.80000 164.00000       1.00
mutual               36.00000  25.00000  35.59000  50.00000       1.00
gwesp.OTP.fixed.0.3 160.48486 110.19728 159.10480 211.08872       0.94
gwodeg.fixed.0.3     38.08101  32.00028  38.14443  41.31023       0.78
gwideg.fixed.0.3     39.68862  35.29399  39.57727  41.51202       0.98
> par(mfrow = c(2, 2), mar = c(5, 4, 4, 2))
> plot(model3.2gof)
```

For graphical analysis, we did:

```
> par(mfrow = c(2, 2), mar = c(5, 4, 4, 2))
> plot(model3.2gof)
```

The plots include, beside the simulated statistics in the first plot, other auxiliary statistics to evaluate the goodness of fit: out and in-degree distribution, edge wise shared partners and minimum geodesic distance. Lets analyse them one by one

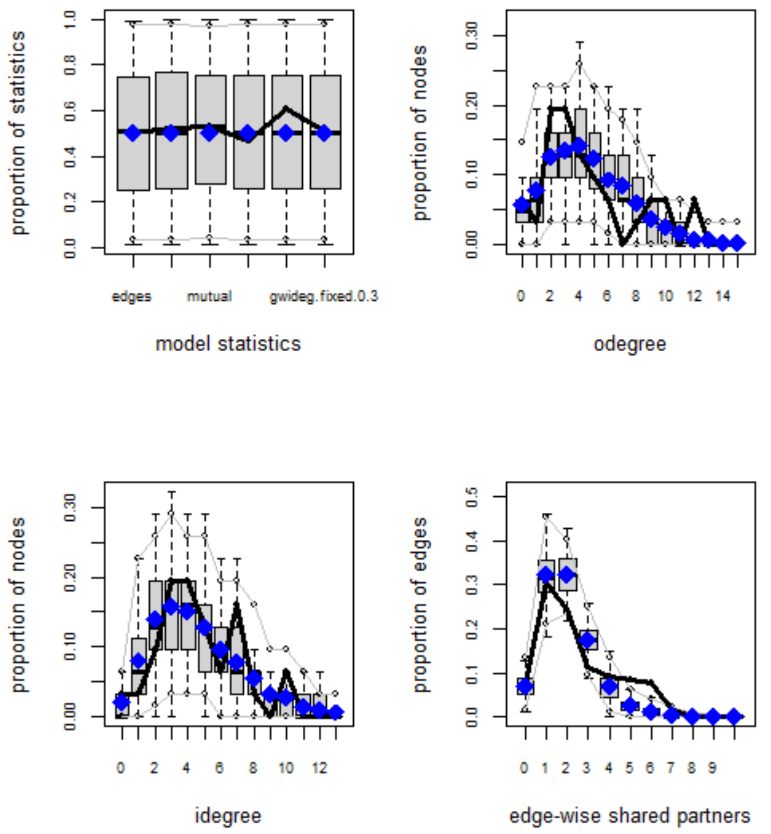
The first plot shows that observed values of these 6 statistics are not extreme in the distribution of the simulated values, in fact, all of 6 statistics are located inside the boxplots(black lines inside the grey boxplots), which is a good indication of the goodness of fit.

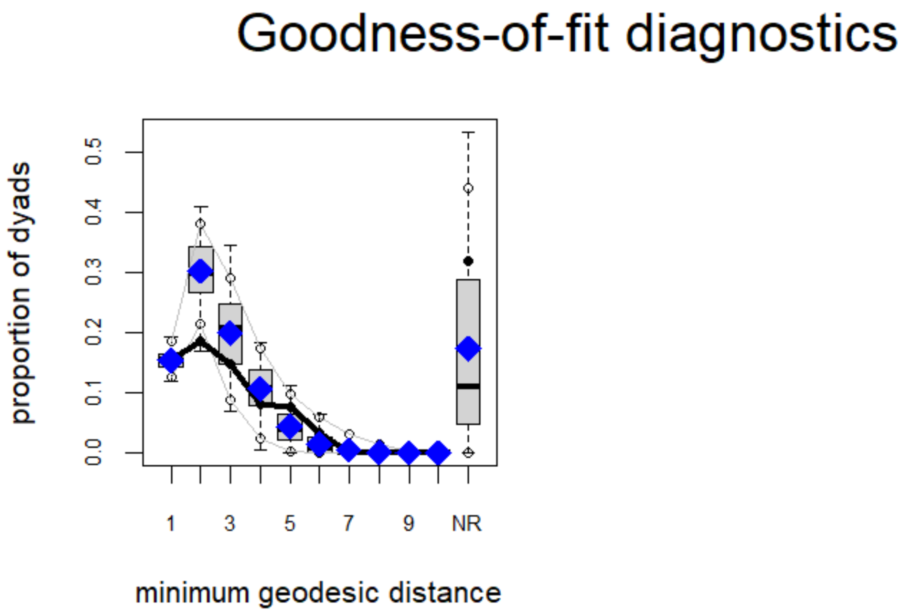
The second plot, we could observe that almost all points are inside the range.

The plot of in-degree indicates that all in-degrees are well represented by the model.

The plot of edge-wise shared partners shows that model explains well with exception in 5 and 6.

The last plot demonstrates the fact that geodesic distance 2 is not well represented by the model.





(5) Interpret the estimated parameters.

Finally, we are gonna to interpret the parameters, with the summary() command, we got the following:

```
> summary(model3.2)
Call:
ergm(formula = friend_net ~ edges + nodematch("sex") + mutual +
    gwesp(decay = 0.3, fixed = TRUE) + gwodegree(decay = 0.3,
    fixed = TRUE) + gwidegree(decay = 0.3, fixed = TRUE))

Monte Carlo Maximum Likelihood Results:

                    Estimate Std. Error MCMC % z value Pr(>|z|)
edges                -5.7517     0.4154      0 -13.847   <1e-04 ***
nodematch.sex         0.9522     0.2019      0   4.717   <1e-04 ***
mutual                0.7871     0.3399      0   2.316   0.0206 *
gwesp.OTP.fixed.0.3   2.1063     0.3172      0   6.639   <1e-04 ***
gwodeg.fixed.0.3      1.3862     0.7269      0   1.907   0.0565 .
gwideg.fixed.0.3      2.6929     1.0683      0   2.521   0.0117 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

     Null Deviance: 1289.3  on 930  degrees of freedom
 Residual Deviance:  567.2  on 924  degrees of freedom
```

```
AIC: 579.2  BIC: 608.2  (Smaller is better. MC Std. Err. = 0.5104)
```

Following the summary, we could interpret the estimated model parameter and conclude the hypothesis one by one:

1- Edge parameter is negative and significant, indicating that the network is sparse.

2- Gender homophily parameter is positive and significant, indicating that the ties with same gender nodes are more likely. This suggests that there is evidence for gender homophily.

3- Reciprocity parameter is positive and significant(under 0.05 significance), indicating that there is evidence for reciprocity and the hypothesis (i.) is hence supported by the data.

4- Geometrically weighted edgewise shared partners (closure of transitive two-path) parameter is positive and significant, indicating that there is evidence for closure of transitive two-path, therefore, the hypothesis (ii.) is supported by the data.

5- Social Activity(`gwodeg.fixed.0.3`) parameter is positive but not significant under 0.05 level of significance(it is significant under 0.1 significance however, but we use 0.05), hence there is no evidence to support hypothesis (iii.)

6- Popularity parameter is positive and significant, indicating that ties are more likely between student when the receiver has a higher in-degree. Therefore, we have evidence for popularity, and hypothesis(iv.) is supported by the data.

Task 4: Comparing ERGM and MR-QAP

(1) Replicate the hypotheses in Task 1(2) using ERGM, with and without the structural terms we specified in Task 3 (2). Comment on the similarity and difference of the results using ERGM compared with those using MRQAP.

The hypotheses in 1.2 were:
i. Boys are more likely to send friendship nominations than girls
ii. Smokers are more likely to receive friendship nominations than non-smokers.
iii. A friendship nomination is more likely between a pair of students participating in the same activity.

In order to test for these hypotheses we add the corresponding terms to the ergm() function.
i) corresponds to nodeofactor(“sex”)
ii) corresponds to nodeifactor(“smoke”)

iii) corresponds to nodematch(“activity”)

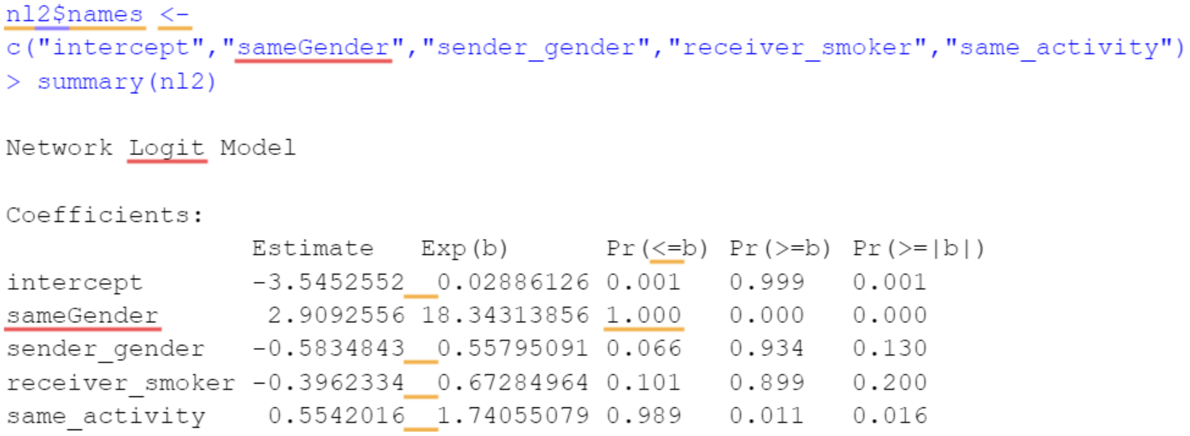
As suggested, the main.method was changed to “Stochastic-Approximation”. The resulting output in the console is omitted in the following.

```
#task 4.1
> set.seed(1)
> #set main.method to "Stochastic Approximation"
> control.ergm(main.method = "Stochastic-Approximation")
...omitted control parameter list...
> #add smoke and activity attributes to friend_net
> smoke = attributes$smoke
> set.vertex.attribute(friend_net, "smoke", smoke)
> activity = attributes$activity
> set.vertex.attribute(friend_net, "activity", activity)
> friend_net
 Network attributes:
  vertices = 31
  directed = TRUE
  hyper = FALSE
  loops = FALSE
  multiple = FALSE
  bipartite = FALSE
  total edges= 144
    missing edges= 0
    non-missing edges= 144

 Vertex attribute names:
    activity sex smoke vertex.names

No edge attributes
>
> #test without terms from 3.2
> model4.1 <- ergm(friend_net ~ edges + nodeofactor("sex") +
nodeifactor("smoke") + nodematch("activity"))
Starting maximum pseudolikelihood estimation (MPLE):
Obtaining the responsible dyads.
Evaluating the predictor and response matrix.
Maximizing the pseudolikelihood.
Finished MPLE.
Evaluating log-likelihood at the estimate.
> summary(model4.1)
Call:
ergm(formula = friend_net ~ edges + nodeofactor("sex") + nodeifactor("smoke")
+
    nodematch("activity"))

Maximum Likelihood Results:
```

```
                        Estimate Std. Error MCMC % z value Pr(>|z|)
edges                    -1.8477     0.1777      0 -10.400   <1e-04 ***
nodeofactor.sex.1         0.1019     0.1924      0   0.529   0.5966
nodeifactor.smoke.1      -0.2622     0.2173      0  -1.206   0.2276
nodematch.activity        0.4060     0.1867      0   2.175   0.0296 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

     Null Deviance: 1289.3  on 930  degrees of freedom
 Residual Deviance:  795.5  on 926  degrees of freedom

AIC: 803.5  BIC: 822.9  (Smaller is better. MC Std. Err. = 0)
```

Here in the ERGM, without using the terms for task 3.2 we could observe in the summary that we have different results now.
The first parameter in this ERGM model, i.e. the parameter for edges(or the intercept in QAP) is still significant and coefficient is negative.
The second parameter parameter for boy sender (see the second row of summary) is not significant, hypothesis 1 can not be supported.
The third parameter(nodeifactor.smoke.1) for receiver_smoker is not significant, the same happened in the QAP, hypothesis 2 can not be supported.
Finally the activity homophily parameter(nodematch.activity) is significant and is positive, just like the QAP. Hypothesis 3 can be supported.

(reminder of the model in MR-QAP)



```
nl2$names <-
c("intercept","sameGender","sender_gender","receiver_smoker","same_activity")
> summary(nl2)

Network Logit Model

Coefficients:
                Estimate   Exp(b)      Pr(<=b) Pr(>=b) Pr(>=|b|)
intercept       -3.5452552  0.02886126 0.001   0.999   0.001
sameGender       2.9092556 18.34313856 1.000   0.000   0.000
sender_gender   -0.5834843  0.55795091 0.066   0.934   0.130
receiver_smoker -0.3962334  0.67284964 0.101   0.899   0.200
same_activity    0.5542016  1.74055079 0.989   0.011   0.016
```

A second model with structural terms of 3.2 is constructed as follows

```
>
> #test with terms from 3.2
> model4.1with <- ergm(friend_net ~ edges + nodeofactor("sex") +
nodeifactor("smoke") + nodematch("activity")
+                      + nodematch("sex") + mutual
+                      + gwesp(decay = 0.3, fixed = TRUE)
+                      + gwodegree(decay = 0.3, fixed = TRUE) +
gwidegree(decay = 0.3, fixed = TRUE))
Starting maximum pseudolikelihood estimation (MPLE):
Obtaining the responsible dyads.
Evaluating the predictor and response matrix.
Maximizing the pseudolikelihood.
Finished MPLE.
Starting Monte Carlo maximum likelihood estimation (MCMLE):
Iteration 1 of at most 60:
Optimizing with step length 0.2277.
The log-likelihood improved by 2.3445.
Estimating equations are not within tolerance region.
Iteration 2 of at most 60:
Optimizing with step length 0.5310.
The log-likelihood improved by 3.1665.
Estimating equations are not within tolerance region.
Iteration 3 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 2.2791.
Estimating equations are not within tolerance region.
Iteration 4 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.3646.
Estimating equations are not within tolerance region.
Iteration 5 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.1669.
Estimating equations are not within tolerance region.
Iteration 6 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.0580.
Convergence test p-value: 0.1632. Not converged with 99% confidence;
increasing sample size.
Iteration 7 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.0318.
Convergence test p-value: 0.0561. Not converged with 99% confidence;
increasing sample size.
Iteration 8 of at most 60:
Optimizing with step length 1.0000.
The log-likelihood improved by 0.0346.
Convergence test p-value: 0.5116. Not converged with 99% confidence;
increasing sample size.
Iteration 9 of at most 60:
Optimizing with step length 1.0000.
```

```
The log-likelihood improved by 0.0303.
Convergence test p-value: < 0.0001. Converged with 99% confidence.
Finished MCMLE.
Evaluating log-likelihood at the estimate. Fitting the dyad-independent
submodel...
Bridging between the dyad-independent submodel and the full model...
Setting up bridge sampling...
Using 16 bridges: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 .
Bridging finished.

This model was fit using MCMC.  To examine model diagnostics and check for
degeneracy, use the
mcmc.diagnostics() function.
> summary(model4.1with)
Call:
ergm(formula = friend_net ~ edges + nodeofactor("sex") + nodeifactor("smoke")
+
    nodematch("activity") + nodematch("sex") + mutual + gwesp(decay = 0.3,
    fixed = TRUE) + gwodegree(decay = 0.3, fixed = TRUE) + gwidegree(decay =
0.3,
    fixed = TRUE))

Monte Carlo Maximum Likelihood Results:

                    Estimate Std. Error MCMC % z value Pr(>|z|)
edges                -5.4524     0.4695      0 -11.614   <1e-04 ***
nodeofactor.sex.1    -0.2991     0.1471      0  -2.034   0.0420 *
nodeifactor.smoke.1  -0.2508     0.1867      0  -1.343   0.1791
nodematch.activity    0.3863     0.1574      0   2.454   0.0141 *
nodematch.sex         1.0969     0.2358      0   4.651   <1e-04 ***
mutual                0.7391     0.3371      0   2.193   0.0283 *
gwesp.OTP.fixed.0.3   1.9688     0.3317      0   5.936   <1e-04 ***
gwodeg.fixed.0.3      0.7257     0.8496      0   0.854   0.3930
gwideg.fixed.0.3      2.0488     1.0120      0   2.024   0.0429 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     Null Deviance: 1289.3  on 930  degrees of freedom
 Residual Deviance:  555.7  on 921  degrees of freedom

AIC: 573.7  BIC: 617.2  (Smaller is better. MC Std. Err. = 0.3654)
```

The MCMC taker longer to converge(compared to not including the task 3.2 terms) since it includes more structural mechanism. Now if we go through the summary to analyze the difference and similarity, we could say:
for the edges parameter it was still significant and negative, same as QAP.

For the boy sender parameter, now it is significant and negative, which means that the boys are less likely to send friendship nominations, hypothesis 1 should be rejected, the true hypothesis is the reverse. Different conclusion as QAP
For the smoker receiver parameter, it is not significant. Hypothesis 2 can not be supported, same conclusion as QAP
For activity homophily , it is significant and positive, hypothesis 3 can be therefore supported, same conclusion as QAP.

(2) Could you think of another hypothesis that could be tested using ERGMs? State your hypothesis and provide the mathematical formula and the graphical representation of the effect that you need to include in the ERGM to test the hypothesis.

Another interesting hypothesis to test could be the following:

> *Students who don’t smoke don’t like to be friends with those that do.*

In the network this would appear as a tie from a node with a “smoke”-attribute of 0 to a node with the attribute value 1. The mathematical formula is

$$\sum_{kl}^{\blacksquare} \blacksquare\, x_{kl}[a_k = 0][a_l = 1]$$

The corresponding keyword would be `nodemix(“smoke”)`, but we would only be interested in the statistic `mix.smoke.0.1`.

```
> #task 4.2
> set.seed(1)
> #add smoke attribute to friend_net
> smoke = attributes$smoke
> set.vertex.attribute(friend_net, "smoke", smoke)
> model4.2 <- ergm(friend_net ~ edges + nodemix("smoke"))
Starting maximum pseudolikelihood estimation (MPLE):
Obtaining the responsible dyads.
Evaluating the predictor and response matrix.
Maximizing the pseudolikelihood.
Finished MPLE.
Evaluating log-likelihood at the estimate.
> summary(model4.2)
Call:
ergm(formula = friend_net ~ edges + nodemix("smoke"))

Maximum Likelihood Results:

              Estimate Std. Error MCMC % z value Pr(>|z|)
```

```
edges           -1.6873     0.1225      0 -13.777   <1e-04 ***
mix.smoke.1.0    0.1666     0.2279      0   0.731    0.465    
mix.smoke.0.1   -0.1172     0.2445      0  -0.479    0.632    
mix.smoke.1.1   -0.4329     0.4491      0  -0.964    0.335    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     Null Deviance: 1289.3  on 930  degrees of freedom
 Residual Deviance:  799.5  on 926  degrees of freedom
 
AIC: 807.5  BIC: 826.9  (Smaller is better. MC Std. Err. = 0)
```

The coefficient mix.smoke.1.0 represents the tendency for non-smokers to form ties with smokers. Although the estimate is positive, it is not statistically significant (p-value = 0.465). The coefficient mix.smoke.0.1 represents the tendency for smokers to form ties with non-smokers. Although the estimate is negative, it is not statistically significant neither (p-value = 0.632). The results show that our hypothesis is rejected by the statistics so the friendship formation has no obvious correlation with people’s smoking habits.