INFX 576: Problem Set 6 - Estimating Networks[*]

Avanti Chande

Due: Thursday, February 23, 2017

Collaborators: Jay Chauhan, Gosuddin Siddiqi

Instructions:

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

1. Download the `problemset6.Rmd` file from Canvas. You will also need the data contained in `problemset6_data.Rdata` and the additional R library `degreenet`.
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_ps6.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
```

```
library(degreenet)
```

```
## Warning: package 'degreenet' was built under R version 3.2.5
```

```
load("problemset6_data.Rdata")
```

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

Problem 1: Perception and Recall of Social Relationships

Pick you favorite social network dataset, this can be data we have encountered in class, data you have collected as part of your own research, or data that was used in one of the readings for the course. Write a short response (3-4 paragraphs) discussing how issues of informant accuracy may or may not affect this data. Be sure to specifically discuss how possible error might be addressed.

Ans.

```r
load('sampson.Rdata')
```

Ans. The data I am going to talk about is the Sampson dataset we've been using in class. The data of particular interest to me is the data on positive affect relations, i.e the LIKE data. Here, each monk was asked if he had positive relations with each of the other monks. Each monk ranked 3/4 as his top choices. Here, a directed edge from monk A to monk B exists if A nominated B among these top choices at any point of time among the three times the data was gathered.

Here, the informant accuracy can gravely affect the data in the following ways: 1. When asked to nominate the people a person likes, the person might get pressurised to name atleast 2 or more people, while the behavioral data might not match the informant's cognitive report. He might not actually 'like' the person he talks to. 2. Secondly, there may be an association of 'liking' to 'talking'. The monks' nomination of other monks in the 'LIKE' dataset may be just a cognitive report of talking to a person. This may affect the actual social behaviour. 3. Another problem that affects the accuracy of this data is that it is a time-aggregated network. The monks are asked to remember who they like/dislike at three different points in time. If they nominate a monk any one time, a dyadic relation is formed. This might not be the actual behavioral structure.

These accuracy errors might possibly be addressed using the sub-graph strucures called cliques with an assumption that the cliques in cognitive and behavioral networks are the same. It is quite possible that a monk i who nominates monk j and monk k, but actually likes monk l and monk m. This would produce an inaccuracy in the dyadic and triadic level of the network structure. However, if i,j,k,l and m form a clique, then monk i's report is a reflection of his interaction with the clique, though not its members. Therefore, a good clique finding algorithm can reduce the noise produced by the informant inaccuracy.

References:

H Russel Bernard, P. D. (2017). Informant Accuracy in Social Network IV.

Problem 2: Modeling Degree Distributions

In the data for this problem set you will find a dataset named `EnronMailUSC1`. This object is the time-aggregated network of emails among 151 employees of Enron Corporation, as prepared by researchers at USC.

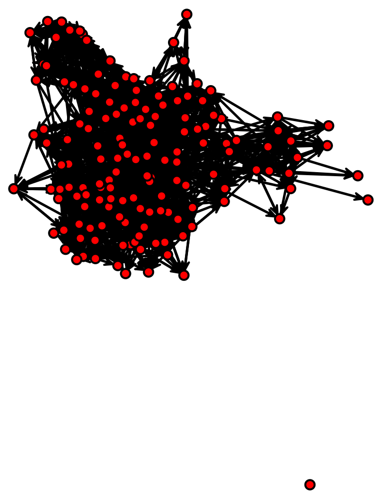
(a) Degree Distribtuions

Begin your investigation by plotting histograms of the indegree, outdegree, and total degree for the Enron email data. Interpret the patterns you see. Do any suggest (or rule out) specific functional form and/or partner formation processes?

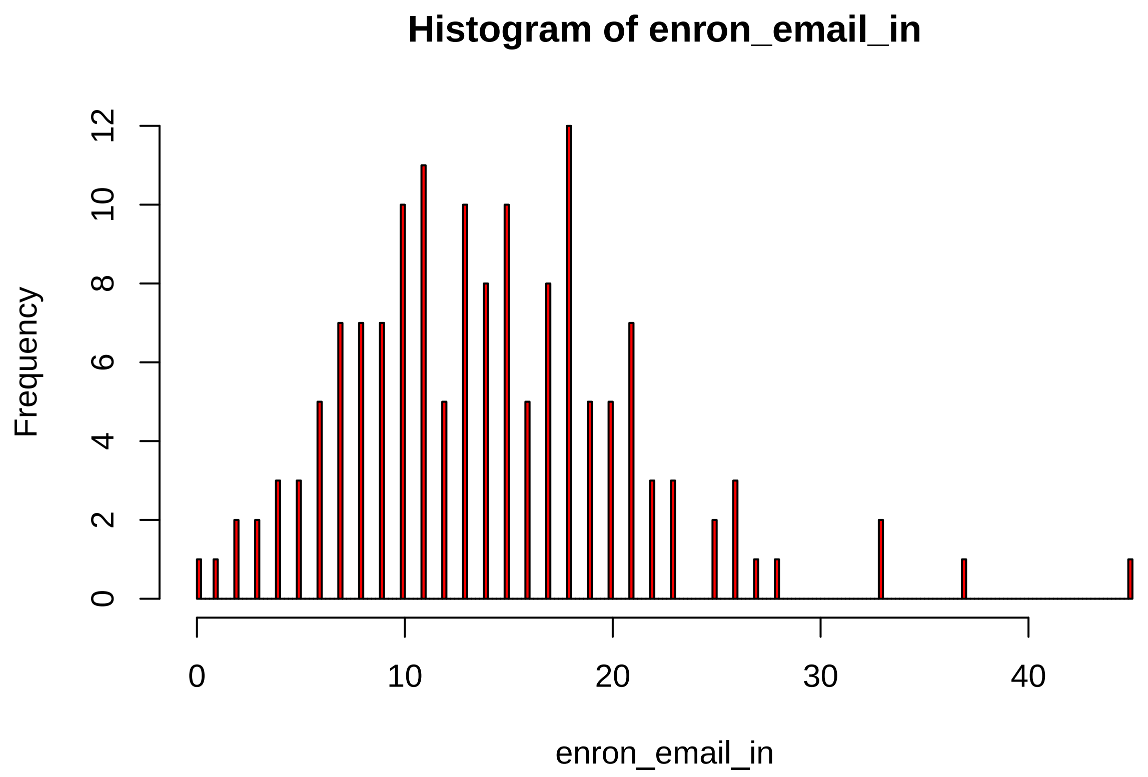
```r
ls()
```

```
## [1] "EnronMailUSC1" "sampson"
```

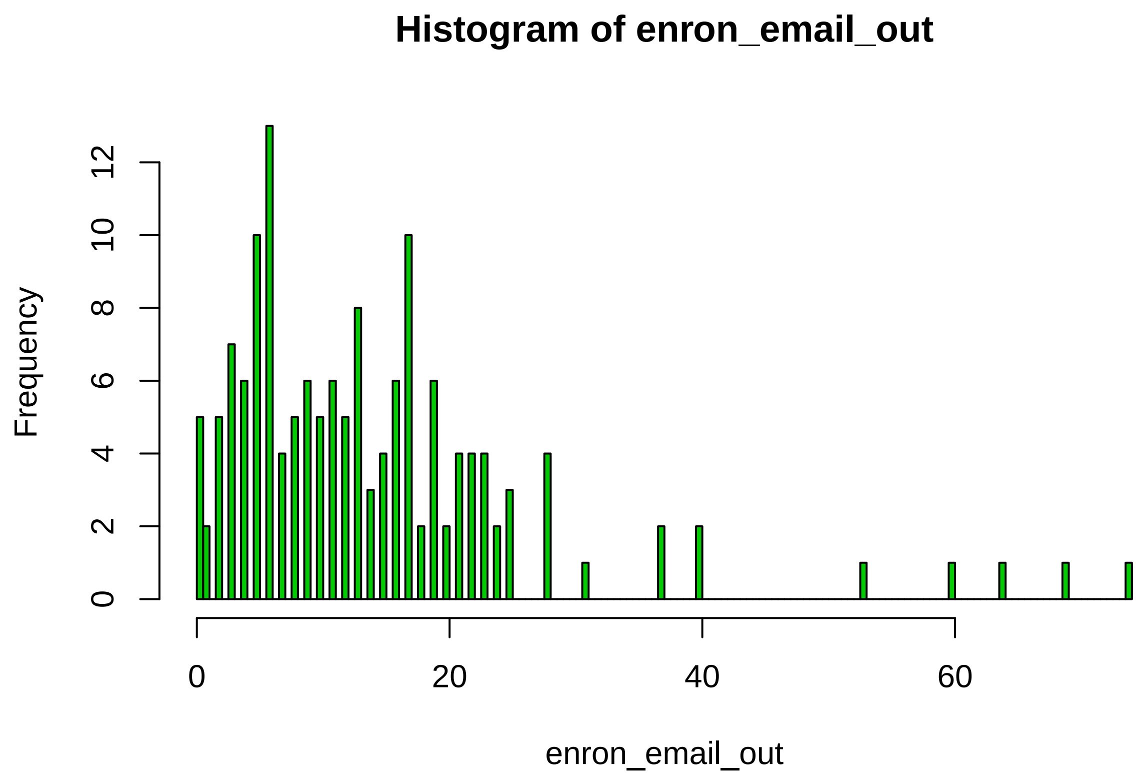
```r
gplot(EnronMailUSC1)
```



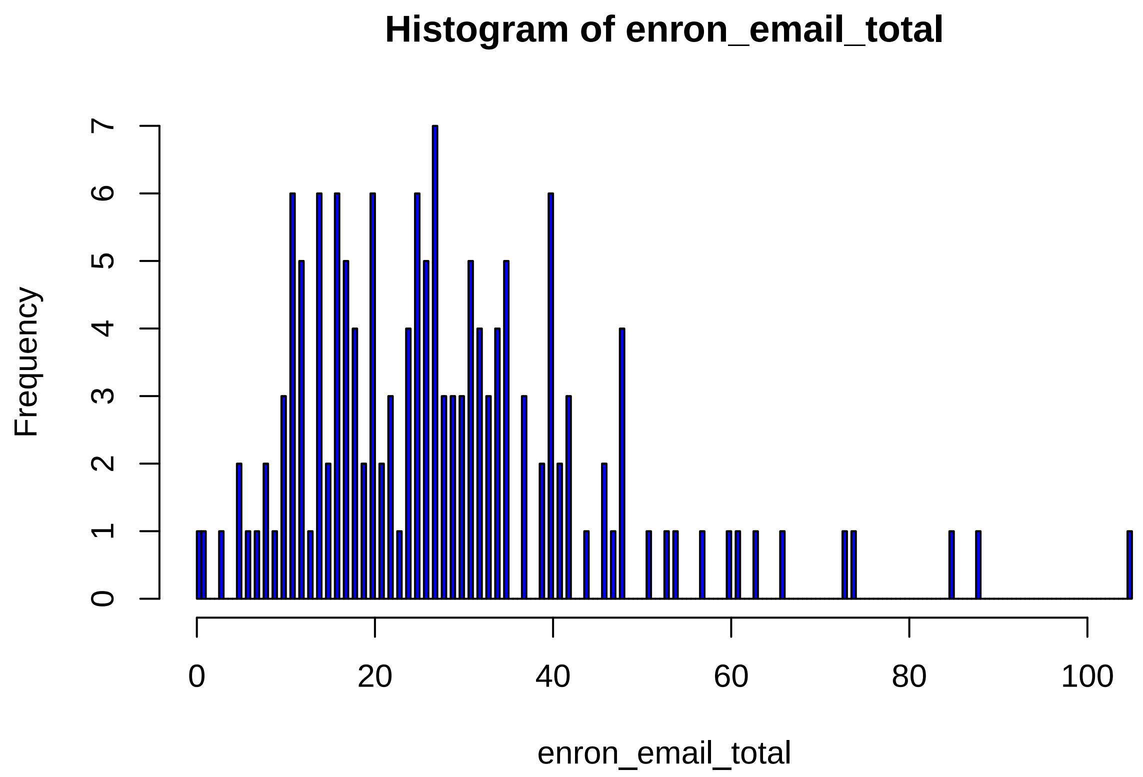
```
enron_email <- EnronMailUSC1

#Indegree
enron_email_in <- degree(enron_email,cmode="indegree")
hist(enron_email_in, col=2, breaks = 200)
```



```
#Outdegree
enron_email_out <- degree(enron_email,cmode="outdegree")
hist(enron_email_out, col=3, breaks = 200)
```



```
#Total Degree
enron_email_total <- degree(enron_email)
hist(enron_email_total, col=4, breaks = 200)
```



From the degree distributions, we can guess the functional forms. Guesses would be as follows: 1. The indegree distribution looks like the Poisson model 2. Outdegree Distribution looks like the Negative Binomial model or Yule/Waring model. 3. The total degree distribution looks like Binomial/Poisson distribution model.

Also, We can rule out some functional forms, for example, Geometic Yule.

(b) Degree Distribution Models

Using the `degreenet` package, fit models to the indegree, outdegree, and total degree distributions for the Enron dataset. Which model provides the best fit in each case in terms of AICC and BIC? In addition to goodness-of-fit information, show the parameters of the best-fitting model.

```r
#Fitting Poisson distribution on Indegree, Outdegree and Total Degree
p1 <- apoimle(enron_email_in)
p2 <- apoimle(enron_email_out)
p3 <- apoimle(enron_email_total)

#Fitting Waring distribution on Indegree, Outdegree and Total Degree
w1 <- awarmle(enron_email_in)
w2 <- awarmle(enron_email_out)
w3 <- awarmle(enron_email_total)

#Fitting Yule distribution on Indegree, Outdegree and Total Degree
y1 <- ayulemle(enron_email_in)
y2 <- ayulemle(enron_email_out)
y3 <- ayulemle(enron_email_total)
```

```
#Fitting Geometric model  on Indegree, Outdegree and Total degree
g1 <- ageomle(enron_email_in)
g2 <- ageomle(enron_email_in)
g3 <- ageomle(enron_email_in)

# Fitting Negative Binomial model  on Indegree, Outdegree and Total degree
nb1 <- anbmle(enron_email_in)
nb2 <- anbmle(enron_email_out)
nb3 <- anbmle(enron_email_total)

# Fitting Negative Binomial model  on Indegree, Outdegree and Total degree
gy1 <- agymle(enron_email_in, guess=c(10,6000))
gy2 <- agymle(enron_email_out, guess=c(10,6000))
gy3 <- agymle(enron_email_total, guess=c(10,6000))

#Calculate the MLE under the Negative Binomial Yule model
?anbymle
nby1 <- anbymle(enron_email_in, guess=c(5,50,0.3))
nby2 <- anbymle(enron_email_out, guess=c(5,500,0.3))
nby3 <- anbymle(enron_email_total, guess=c(5,50,0.3))
```

```
#Testing Goodness of Fit for Indegree
fittab.enron_email_in <-rbind(
  llpoiall(p1$theta, enron_email_in),
  llwarall(w1$theta, enron_email_in),
  llyuleall(y1$theta, x=enron_email_in),
  llgeoall(g1$theta, x=enron_email_in),
  llnball(nb1$theta, x=enron_email_in),
  llgyall(gy1$theta, x= enron_email_in),
  llnbyall(nby1$theta, x=enron_email_in)
)

rownames(fittab.enron_email_in)<-c("Poisson","Waring","Yule","Geometric","NegBinom",
    "GeoYule","NegBinomYule")

#Display the Goodness of Fit table
fittab.enron_email_in
```

```
##              np   log-lik     AICC      BIC
## Poisson       1 -596.9712 1195.969 1198.960
## Waring        3 -556.9045 1119.972 1128.861
## Yule          3 -653.1647 1312.493 1321.381
## Geometric     3 -542.5341 1091.232 1100.120
## NegBinom      3 -502.2345 1010.632 1019.521
## GeoYule       4 -653.1803 1314.635 1326.430
## NegBinomYule  5 -502.9002 1016.214 1030.887
```

```
#Testing Goodness of Fit for Outdegree
fittab.enron_email_out <-rbind(
  llpoiall(p2$theta, enron_email_out),
  llwarall(w2$theta, enron_email_out),
  llyuleall(y2$theta, x=enron_email_out),
  llgeoall(g2$theta, x=enron_email_out),
  llnball(nb2$theta, x=enron_email_out),
```

```r
  llgyall(gy2$theta, x= enron_email_out),
  llnbyall(nby2$theta, x=enron_email_out)
)

rownames(fittab.enron_email_out)<-c("Poisson","Waring","Yule","Geometric","NegBinom",
    "GeoYule","NegBinomYule")

fittab.enron_email_out
```

```
##              np   log-lik     AICC      BIC
## Poisson       1 -999.1225 2000.272 2003.262
## Waring        3 -558.3307 1122.825 1131.713
## Yule          3 -625.1602 1256.484 1265.372
## Geometric     3 -550.3768 1106.917 1115.805
## NegBinom      3 -547.9978 1102.159 1111.047
## GeoYule       4 -625.1598 1258.594 1270.389
## NegBinomYule  5 -625.1602 1260.734 1275.407
```

```r
#Testing Goodness of Fit for Total Degree
fittab.enron_email_total <-rbind(
  llpoiall(p3$theta, enron_email_total),
  llwarall(w3$theta, enron_email_total),
  llyuleall(y3$theta, x=enron_email_total),
  llgeoall(g3$theta, x=enron_email_total),
  llnball(nb3$theta, x=enron_email_total),
  llgyall(gy3$theta, x= enron_email_total),
  llnbyall(nby3$theta, x=enron_email_total)
)

rownames(fittab.enron_email_total)<-c("Poisson","Waring","Yule","Geometric","NegBinom",
    "GeoYule","NegBinomYule")

fittab.enron_email_total
```

```
##              np    log-lik     AICC      BIC
## Poisson       1 -1096.2222 2194.471 2197.462
## Waring        3  -668.4949 1343.153 1352.042
## Yule          3  -789.3682 1584.900 1593.788
## Geometric     3  -698.0450 1402.253 1411.142
## NegBinom      3  -622.9454 1252.054 1260.943
## GeoYule       4  -789.3980 1587.070 1598.865
## NegBinomYule  5  -622.0948 1254.603 1269.276
```

Ans. According to the goodness of fit measured as AICC and BIC, the best model fit seems to be the negative binomial model, since it has the lowest AICC and BIC. It is the best fit model for indegree, outdegree and total degree distributions of the Enron Email network.

```r
#Showing the Parameters of the best fit model - Neg Binomial for Indegree, OutDegree and #Total Degree

#Indegree
print ("Indegree")
```

```
## [1] "Indegree"
```

```r
nb1
```

```
## $theta
## expected stop   prob 1 stop
##    18.1762351     0.2668257
```

```
#Outdegree
print ("Outdegree")
```

```
## [1] "Outdegree"
```

```
nb2
```

```
## $theta
## expected stop   prob 1 stop
##    15.3759296     0.1077503
##
## $asycov
##                expected stop   prob 1 stop
## expected stop   0.362731013 -0.0014139341
## prob 1 stop    -0.001413934  0.0001836278
##
## $se
## expected stop   prob 1 stop
##    0.60227154    0.01355093
##
## $asycor
##                expected stop prob 1 stop
## expected stop     1.0000000  -0.1732477
## prob 1 stop      -0.1732477   1.0000000
##
## $npar
## gamma mean gamma s.d.
##   13.71917   10.65855
##
## $value
## [1] -526.0423
```

```
#Total Degree
print ("Total Degree")
```

```
## [1] "Total Degree"
```

```
nb3
```

```
## $theta
## expected stop   prob 1 stop
##    30.7407884     0.1004351
##
## $asycov
##                expected stop   prob 1 stop
## expected stop  0.3522025032 -0.0002314724
## prob 1 stop   -0.0002314724  0.0001493371
##
## $se
## expected stop   prob 1 stop
##    0.59346651    0.01222035
##
## $asycor
##                expected stop prob 1 stop
## expected stop     1.0000000  -0.0319168
## prob 1 stop      -0.0319168   1.0000000
##
## $npar
## gamma mean gamma s.d.
##   27.65334   15.73792
##
## $value
## [1] -616.9315
```