Networks Homework 1

William O'Brochta

1 Preparation

```
library(dplyr)

Warning:  package 'dplyr' was built under R version 3.4.4

library(dils)

Warning:  package 'igraph' was built under R version 3.4.4
Warning:  package 'Rcpp' was built under R version 3.4.4

library(igraph)

load('nigeria.rda')
nigeria1<-nigeria[,-4]
nigeria1$sender<-as.character(nigeria1$sender)
nigeria1$receiver<-as.character(nigeria1$receiver)

#Aggregate on sender and receiver regardless of year
nigeria1.1<-nigeria1%>%
  group_by(sender,receiver)%>%
  summarise(conflict=sum(conflict))
nigeria1.1<-as.data.frame(nigeria1.1)

#Should have 1369 entries, but only have 1365
37*37

[1] 1369

#Check out table to see if each militia has 37
#partner sender and receivers

#table(nigeria1.1$sender)
#table(nigeria1.1$receiver)

#Uvwie Militia and Udu Militia are missing one, Boko Haram is missing two,
#in both cases

#Missing receiver Boko Haram
#nigeria1.1[nigeria1.1$sender=='Uvwie\nMilitia',]
```

```r
#Missing receiver Boko Haram
#nigeria1.1[nigeria1.1$sender=='Udu\nMilitia',]

#Missing receiver Uvwie Militia and Udu Militia
#nigeria1.1[nigeria1.1$sender=='Boko\nHaram',]

#Create and add these rows, presuming they have no conflicts
nigeria_extra<-data.frame(c('Uvwie\nMilitia', 'Boko\nHaram', 0),
                          c('Udu\nMilitia', 'Boko\nHaram', 0),
                          c('Boko\nHaram', 'Uvwie\nMilitia', 0),
                          c('Boko\nHaram', 'Udu\nMilitia', 0))
nigeria_extra<-as.data.frame(t(nigeria_extra))
colnames(nigeria_extra)<-c('sender', 'receiver', 'conflict')

nigeria1.2<-rbind(nigeria1.1, nigeria_extra)
rownames(nigeria1.2)<-NULL

#Now, kill the weighted data
nigeria1.2$conflict<-ifelse(nigeria1.2$conflict==0,0,1)

#Finally, turn into adjacency matrix
nigeria_adjMat<-AdjacencyFromEdgelist(nigeria1.2)
nigeria_adjMat2<-nigeria_adjMat$adjacency
rownames(nigeria_adjMat2)<-nigeria_adjMat$nodelist
colnames(nigeria_adjMat2)<-nigeria_adjMat$nodelist

#Check results by observing adjacency matrix.

g = graph_from_adjacency_matrix(nigeria_adjMat2,
                                mode='directed',
                                weighted=NULL,
                                diag=FALSE
)
```

2 Measurements and Community Detection

```r
#Estimate influence via degree and eigenvector centrality
influence_compare<-as.data.frame(cbind(igraph::degree(g),
                         eigen_centrality(g, directed = FALSE)$vector))
colnames(influence_compare)<-c('Degree', 'Eigen')

#Order by Degree and Eigen, determine similarities.
head(influence_compare[order(influence_compare$Degree, decreasing=T),])
```

```
                   Degree  Eigen
Police\n(Nigeria)      28 1.0000
Fulani\nMilitia        22 0.8554
Military\n(Nigeria)    19 0.7567
Hausa\nMilitia         12 0.4018
Christian\nMilitia      8 0.5022
```

```
Boko\nHaram              7 0.4449
```

```
head(influence_compare[order(influence_compare$Eigen, decreasing=T),])
```

```
                    Degree  Eigen
Police\n(Nigeria)       28 1.0000
Fulani\nMilitia         22 0.8554
Military\n(Nigeria)     19 0.7567
Christian\nMilitia       8 0.5022
Boko\nHaram              7 0.4449
Vigilante\nMilitia       6 0.4213
```

We have learned two different ways to measure the influence of an individual actor, degree centrality and eigenvector centrality. In this application, it is a bit less clear which measure is more appropriate. Conflict is often thought of just between two parties, in which case degree centrality would be the best option. However, if I start conflicts with many groups that also start conflicts, I may be a stronger actor because I am able to initiate conflict with another actor who has started a lot of conflicts. If we believe that the number of conflicts started is a measure of actor strength, then my boldness of starting a conflict with a strong actor might make me more influential. I compute both influence statistics above. We see that the top three actors are the same for both statistics. These actors, therefore, can be classified as most influential. Beyond that, there is some disagreement between degree and eigenvector centrality, but only the Hausa Militia and Vigilante Militia are dropped from the top six most influential actors in one of the two measures. The Vigilante Militia is interesting because they initiate fewer conflicts, but these conflicts are relatively more "difficult."

```
library(network, quietly=T)
```

```
Warning:  package 'network' was built under R version 3.4.4
```

```
require(sna)
library(cvTools)
```

```
Warning:  package 'robustbase' was built under R version 3.4.4
```

```
library(PRROC)
```

```
Warning:  package 'PRROC' was built under R version 3.4.4
```

```
#Make network object
g1 <- network(nigeria_adjMat2, directed=TRUE)
eclusts <- equiv.clust(g1)
#plot(eclusts,hang=-1)
#The dendrogram seems to indicate at most k=4 communities

#Set-up number of initiated conflicts
#This is the data we are to use to group
#into communities
g1_degree<-sna::degree(g1, cmode='outdegree')
#Make the data binary
g1_degree<-ifelse(g1_degree!=0,1,0)
```

```
#Create empty df to put in block membership
clustNum<-as.data.frame(matrix(nrow=37, ncol=3, NA))

#Put in block membership and add conflict binary data
set.seed(5)
for(i in 1:3){
g1BlockM <- blockmodel(g1, eclusts, k=i+1)
clustNum[,i]<-g1BlockM$block.membership
}
colnames(clustNum)<-c('k2', 'k3', 'k4')

clustNum<-cbind(clustNum, g1_degree)
clustNum<-as.data.frame(clustNum)


#Set-up cross validation
folds <- cvFolds(NROW(clustNum), K=5)
clustNum$holdoutpred <- rep(0,nrow(clustNum))
clustNum$holdoutpred2 <- rep(0,nrow(clustNum))
clustNum$holdoutpred3 <- rep(0,nrow(clustNum))

set.seed(5)
for(i in 1:4){
  train <- clustNum[folds$subsets[folds$which != i], ]
  validation <- clustNum[folds$subsets[folds$which == i], ]

  newlm <- glm(g1_degree~k2, data=train, family=binomial(link='logit'))
  newpred <- predict(newlm, newdata=validation, type='response')
  clustNum[folds$subsets[folds$which == i], ]$holdoutpred <- newpred

  newlm2 <- glm(g1_degree~k3, data=train, family=binomial(link='logit'))
  newpred2 <- predict(newlm2,newdata=validation, type='response')
  clustNum[folds$subsets[folds$which == i], ]$holdoutpred2 <- newpred2

  newlm3 <- glm(g1_degree~k4, data=train, family=binomial(link='logit'))
  newpred3 <- predict(newlm3,newdata=validation, type='response')
  clustNum[folds$subsets[folds$which == i], ]$holdoutpred3 <- newpred3
}

Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
```

The purpose of this question is to figure out the optimal number of communities. The question required some interpretation and modifications on my part. We need to create binary data that we can use the community membership to predict. Just using whether a node was involved in a conflict (i.e. degree) produces no variation. Thus, I think the question wants us to use whether the node initiated a conflict where there is some variation. If this is wrong, the cross-validation process should still go in the same manner.

The second problem was that ten fold cross-validation left too few observations in the validation dataset (3). I decreased the cross-validation to five fold, and that seemed not to throw errors as often. I am unfamiliar with cross-validation in general, so I am not sure

what impact changing the number of folds might have on the results.

In the cross-validation loop, I establish a training and validation dataset and then run a logit of group membership on the binary conflict initiation data created earlier. I choose to go from two to four groups because the dendrogram does not show much clustering by group membership at all, so anything greater than four groups seemed excessive. I saved the predicted probabilities.

```
#Do AUC(ROC) and AUC(PR)
fg <- clustNum$holdoutpred[clustNum$g1_degree == 1]
bg <- clustNum$holdoutpred[clustNum$g1_degree == 0]
roc <- roc.curve(scores.class0 = fg, scores.class1 = bg, curve = T)
pr <- pr.curve(scores.class0 = fg, scores.class1 = bg, curve = T)

fg <- clustNum$holdoutpred2[clustNum$g1_degree == 1]
bg <- clustNum$holdoutpred2[clustNum$g1_degree == 0]
roc2 <- roc.curve(scores.class0 = fg, scores.class1 = bg, curve = T)
pr2 <- pr.curve(scores.class0 = fg, scores.class1 = bg, curve = T)

fg <- clustNum$holdoutpred3[clustNum$g1_degree == 1]
bg <- clustNum$holdoutpred3[clustNum$g1_degree == 0]
roc3 <- roc.curve(scores.class0 = fg, scores.class1 = bg, curve = T)
pr3 <- pr.curve(scores.class0 = fg, scores.class1 = bg, curve = T)

c(roc$auc, roc2$auc, roc3$auc)

[1] 0.4543 0.4382 0.4382

c(pr$auc.integral, pr2$auc.integral, pr3$auc.integral)

[1] 0.7804 0.7738 0.7738
```

I am not familiar with the AUC(ROC) or AUC(PR). I did some reading online and believe that AUC(ROC) and AUC(PR) are both ways of calculating the false positive and false negative rates and trying to minimize them. The prediction is better when the values of both of these are high. For this case, the AUC(ROC) and AUC(PR) are highest when there are two communities, but the values all seem very close together.

```
#Now select k=2 and put that k here
g1BlockM <- blockmodel(g1, eclusts, k=2)
#bmems <- g1BlockM£block.membership[g1BlockM£order.vec]

#Correct coloring
bmems <- c(rep(1,23), 2, 1, 1, 1, 2, rep(1, 9))

#Match color vector to number of k
colVec <- c("lightblue","yellow")
bcols <- colVec[bmems]
head(cbind(bcols,bmems))

     bcols       bmems
[1,] "lightblue" "1"
```

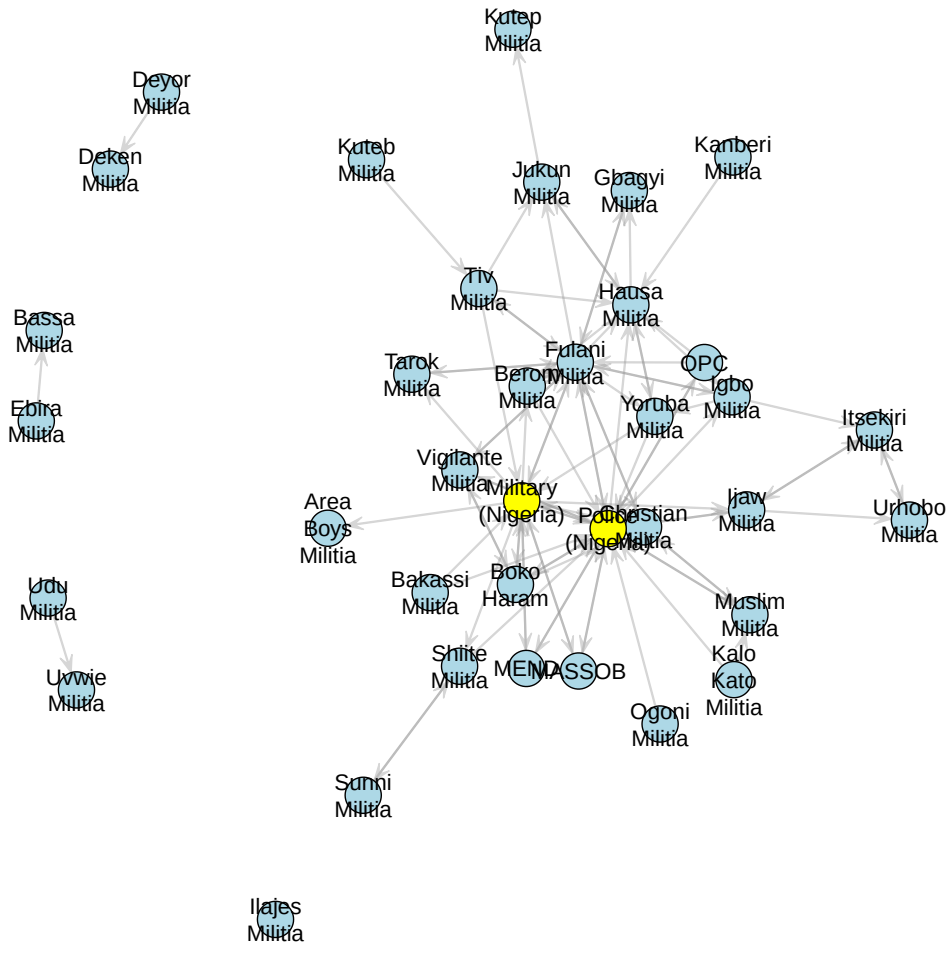
```
[2,] "lightblue" "1"
[3,] "lightblue" "1"
[4,] "lightblue" "1"
[5,] "lightblue" "1"
[6,] "lightblue" "1"

set.seed(5)
#plot(g1,displaylabels=T,
#     vertex.cex=2,label.cex=1,edge.col=rgb(150,150,150,100,maxColorValue=255),
#     label.pos=5,vertex.col=bcols)
```

For some interesting reason, the block membership does not correctly color the nodes as it did in the Star Wars example. I noticed this and colored the nodes manually.

Figure 1 is the network plot with the two communities in different colors. I found that this clustering made some sense because the Police and the Military are the ones who initiated the most conflicts in the dataset. Thus, it is logical to cluster them together. I also tried a number of different values for k manually, and none of them seemed to fit the data well at all. One might hope that the militias on the periphery would be clustered together, but that never seemed to happen.

Figure 1: Two Community Network Plot



ERGM

```
library(statnet)
set.seed(6886)
#m1 = ergm(g1 ~ edges + mutual + gwidegree(decay = 1, fixed = TRUE),
#control=control.ergm(seed=6886,MCMC.samplesize=10000))
#summary(m1)

# Convergence plots
#mcmc.diagnostics(m1)

# Goodness of fit function for ergm
set.seed(6886)
#gofM1 = gof(
# m1,
# GOF=~idegree + odegree + espartners + distance-model
# )

#par(mfrow = c(2, 2))
#plot(gofM1)
```

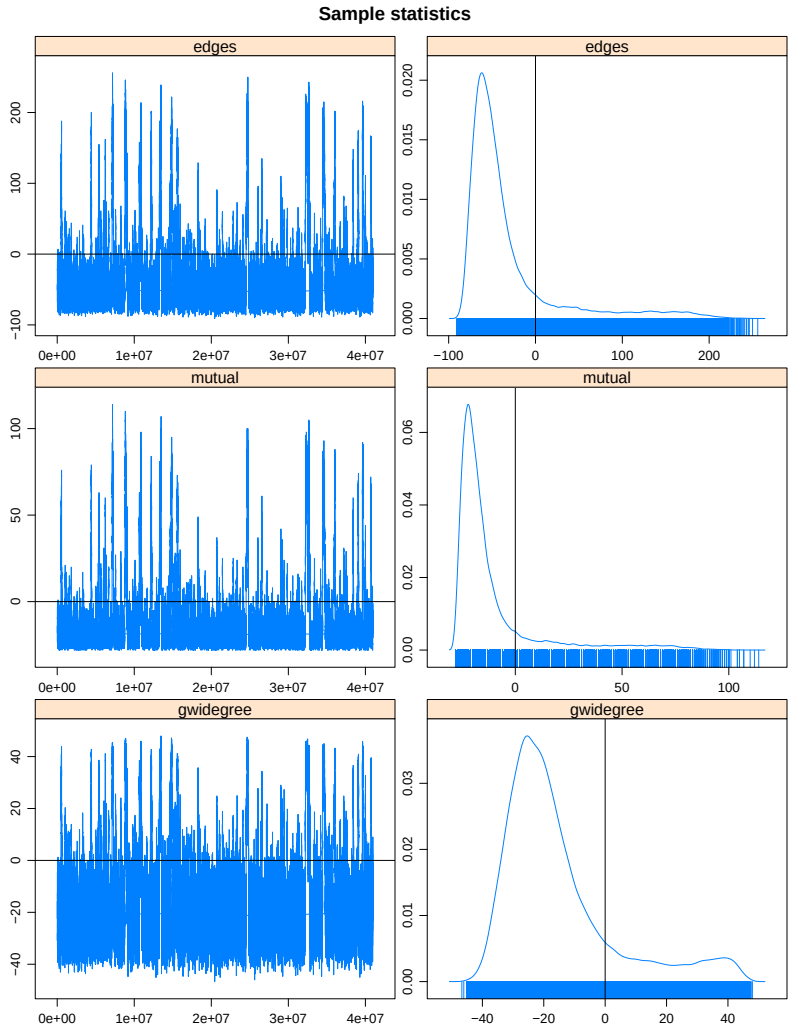
I explored two very simple hypotheses: first, that reciprocity occurs such that a group initiating a conflict with another group is likely to be a recipient of conflict from that same group and second, that conflict is more likely when the degree of the nodes involved is high. I used "mutual" to test for reciprocity, and I found a term "gwidegree" that tests whether conflict is more likely when high degree nodes are involved. I tested a number of other triangle like hypotheses because I observed triangles in the network, but none converged. Table 1 shows the model results and supports both hypotheses. A positive coefficient on "mutual" indicates that conflict is more likely to occur against a group that previously attacked you. Further, the negative coefficient on "gwidegree" indicates that there is an increased likelihood of conflict when high degree nodes are involved.

Table 1: ERGM for Nigeria Data

	Dependent variable:
	g1
edges	-2.561^{***}
	(0.225)
mutual	3.797^{***}
	(0.441)
gwidegree	-2.301^{***}
	(0.262)
Akaike Inf. Crit.	545.007
Bayesian Inf. Crit.	560.591
Note:	*p<0.1; **p<0.05; ***p<0.01

I tested this model for convergence (Figure 2). The initial convergence plot was quite bad in that the MCMC sampler did not explore the full range of values. I used the guidance in the slides to increase sample size, which did improve convergence. I imagine other adjustments are also possible.

Figure 2: MCMC Convergence



The reason I did not worry about convergence more is that the goodness-of-fit diagnostics in Figure 3 are actually great. The model seems to do a very good job fitting the simulations. I did some reading that suggested goodness-of-fit should be prioritized in model selection over convergence plots because convergence plots are unreliable. This is another reason I put more emphasis in the goodness-of-fit diagnostics.

Figure 3: Goodness-of-Fit Diagnostics