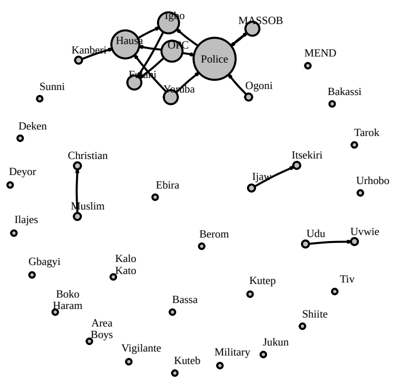
# Homework

```r
load('nigeria.rda')

years = sort(unique(nigeria$year))
groups = with(nigeria, intersect(sender, receiver))
n = length(groups)
adjmat = matrix(0,nrow = n, ncol = n, dimnames = list(groups, groups))

nigerialist = lapply(years, function(t) {
  slice = nigeria[nigeria$year==t,]
  positivecases = slice[slice$conflict==1,]
  for(i in 1:nrow(positivecases)){
    sender= as.character(positivecases$sender[i])
    receiver= as.character(positivecases$receiver[i])
    adjmat[sender, receiver]=1
  }
  return(as.network.matrix(adjmat))
})
names(nigerialist) <- years
```

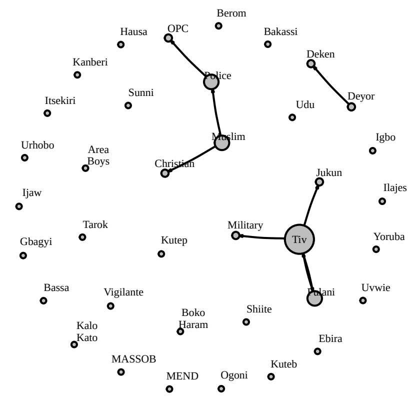
```
par(mfrow = c(1,2))
for (i in 1:length(years)) {
  g <- nigerialist[[i]]
  g1 <- asIgraph(g)
  deg <- degree(g1, mode="all")
  V(g1)$size <- deg*4
  dist <- ((-1.4)*(V(g1)$size-min(V(g1)$size)))/(max(V(g1)$size)-min(V(g1)$size))+1.4
  plot(g1, edge.arrow.size=.08, edge.arrow.width=0.8, edge.curved=.05, edge.color="black
     vertex.label=labels, vertex.label.color="black",
     vertex.label.dist=dist, vertex.label.cex = .3, vertex.color="grey",
     main=years[i],layout=layout_with_kk)
  if ((i %% 2) == 0) par(mfrow = c(1,2))

  #degree
  deg_total[i] <- paste(groups[which( deg == max(deg) )], collapse = ' ')
  deg_total[i] <- gsub("\nMilitia", "", deg_total[i])
  deg_total[i] <- gsub("\n\\(Nigeria\\)", "", deg_total[i])
  #degree in
  deg_in <- degree(g1, mode="in")
  deg_in_total[i] <- paste(groups[which( deg_in == max(deg_in) )], collapse = ' ')
  deg_in_total[i] <- gsub("\nMilitia", "", deg_in_total[i])
  deg_in_total[i] <- gsub("\n\\(Nigeria\\)", "", deg_in_total[i])
  #degree out
  deg_out <- degree(g1, mode="out")
  deg_out_total[i] <- paste(groups[which( deg_out == max(deg_out) )], collapse = ' ')
  deg_out_total[i] <- gsub("\nMilitia", "", deg_out_total[i])
  deg_out_total[i] <- gsub("\n\\(Nigeria\\)", "", deg_out_total[i])
```

```
}
```

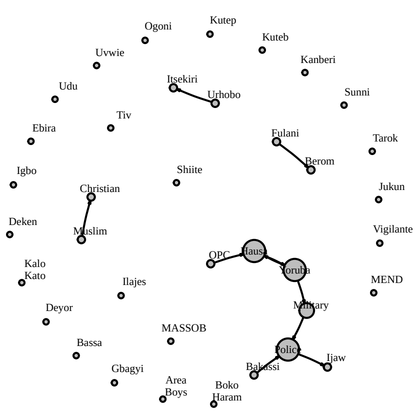
## 2000



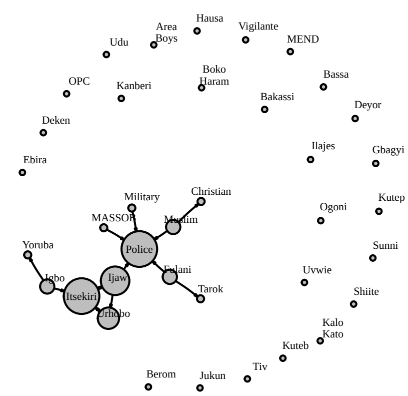

## 2001



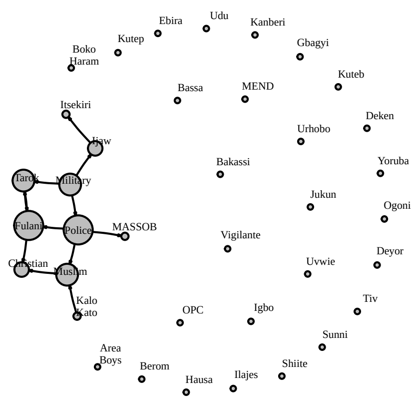

## 2002



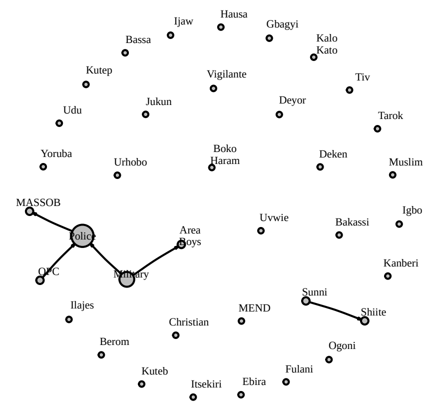

## 2003



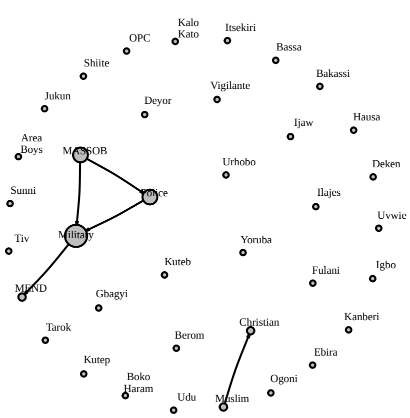
## 2004



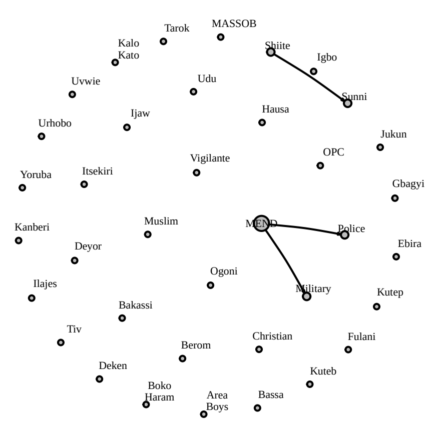

## 2005



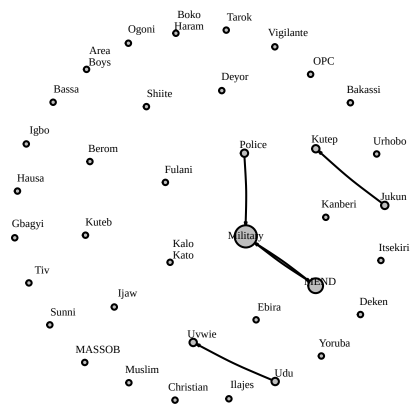

## 2006



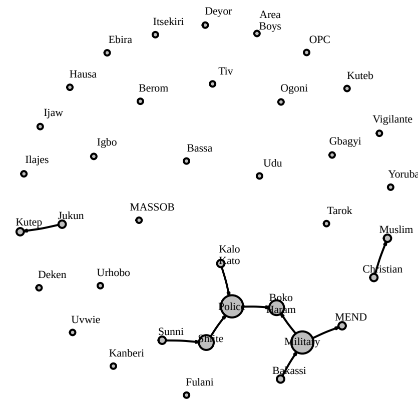

## 2007



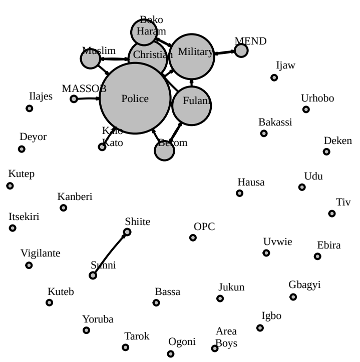
## 2008



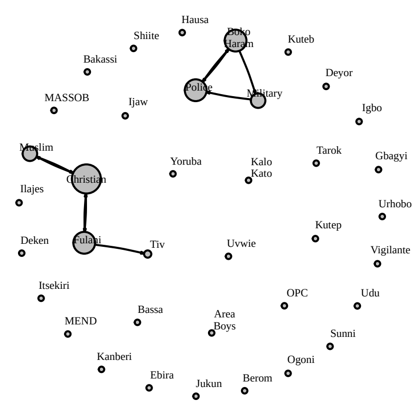

## 2009



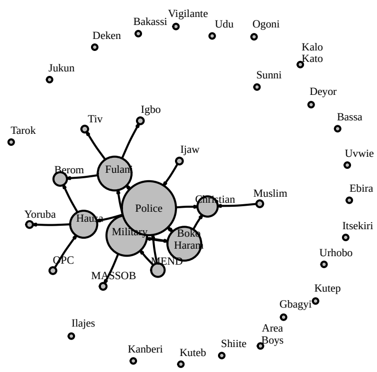

## 2010



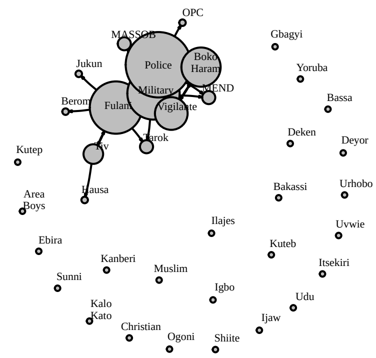

## 2011



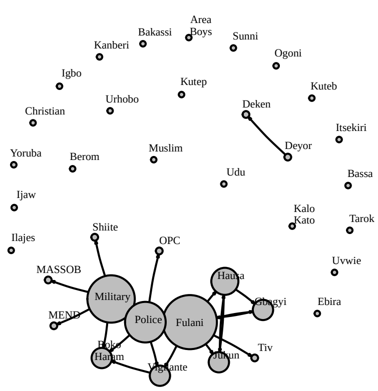
## 2012



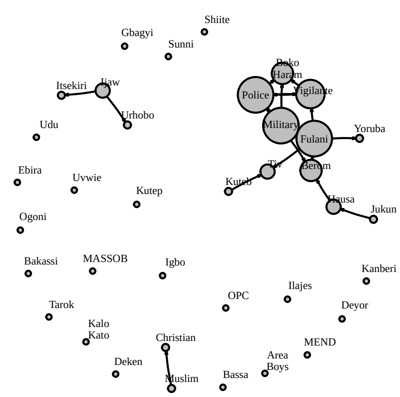

## 2013



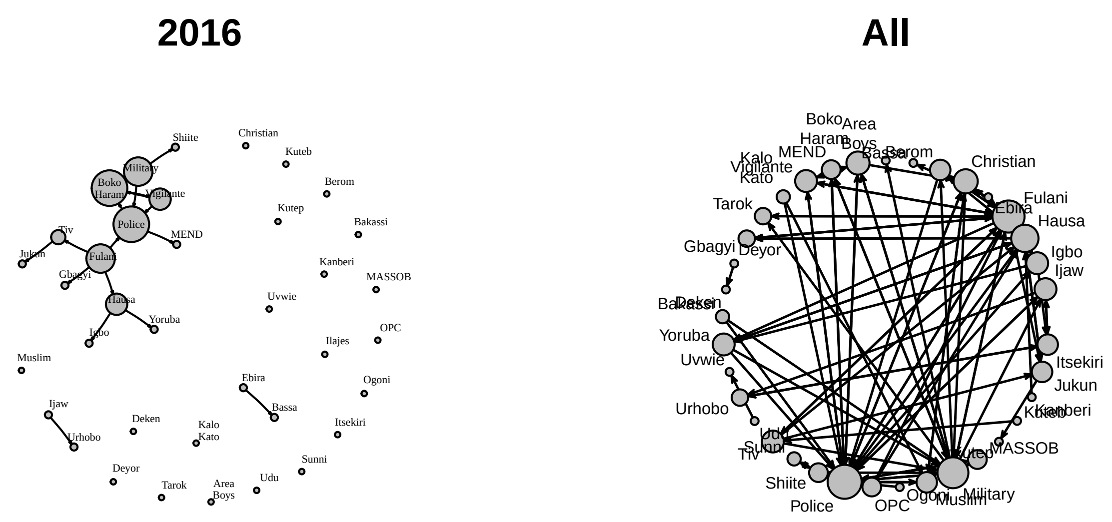

## 2014




## 2015



```
plot(nigeriaDyn, label = labels, main="All", label.cex=0.5, mode="circle", vertex.cex=lo
     label.col="black", vertex.col="grey", edge.col="black")
```




In the graphs the size of the nodes is based on the global degree and in most of years the actor with higher degree is the military or the police. By the degree it is clear that Police is the actor more influential, and only the Fulani Militia seems equal influential according to the out-degree measure. Degree seems the best measurment giving that interaction is based on attacks. However if we look at eigenvector centrality still the Police would be the most influential actor.

Degree seems to be the best way to measure influence when looking year by year as there are many isolated nodes which makes distances or eigenvector centrality less usefull in this case. As seen bellow only in 2001, 2007, 2011 and 2014 the Police or the Military were not the most influential actors. By looking at the in-degree or the receiver, it seems that most of the time the Police was the most influential in most years. Looking at the out-degree the military was the most influential actor.

```r
#Degree
deg_all <- sna::degree(nigeriaDyn)
paste(groups[which( deg_all == max(deg_all) )], collapse = ' ')

## [1] "Police\n(Nigeria)"

deg_in_all <- sna::degree(nigeriaDyn, cmode="indegree")
paste(groups[which( deg_in_all == max(deg_in_all) )], collapse = ' ')

## [1] "Police\n(Nigeria)"

deg_out_all <- sna::degree(nigeriaDyn, cmode="outdegree")
paste(groups[which( deg_out_all == max(deg_out_all) )], collapse = ' ')

## [1] "Fulani\nMilitia Police\n(Nigeria)"

#Eigenvector centrality
eigen <- sna::evcent(nigeriaDyn)
paste(groups[which( eigen == max(eigen) )], collapse = ' ')

## [1] "Police\n(Nigeria)"

#Degree year by year
as.data.frame(cbind(years, "Higher Degree (all)"=deg_total))

##    years   Higher Degree (all)
## 1   2000                Police
## 2   2001                   Tiv
## 3   2002   Hausa Police Yoruba
## 4   2003       Itsekiri Police
## 5   2004         Fulani Police
```

```
## 6   2005                 Police
## 7   2006               Military
## 8   2007                   MEND
## 9   2008               Military
## 10  2009        Military Police
## 11  2010                 Police
## 12  2011              Christian
## 13  2012                 Police
## 14  2013                 Police
## 15  2014                 Fulani
## 16  2015 Fulani Military Police
## 17  2016     Police Boko\nHaram
```

```
as.data.frame(cbind(years, "Higher Degree (in)"=deg_in_total))
```

```
##    years                                     Higher Degree (in)
## 1   2000                                                 Police
## 2   2001 Christian Fulani Jukun Military OPC Police Tiv Deken
## 3   2002                                           Hausa Police
## 4   2003                                                 Police
## 5   2004                          Christian Fulani Muslim Tarok
## 6   2005                                                 Police
## 7   2006                                               Military
## 8   2007                                  Military Police Sunni
## 9   2008                                               Military
## 10  2009                                     Police Boko\nHaram
## 11  2010                                                 Police
## 12  2011                                       Christian Police
```

```
## 13  2012                                              Police
## 14  2013                                              Fulani
## 15  2014                                 Fulani Boko\nHaram
## 16  2015                                  Berom Boko\nHaram
## 17  2016                                 Police Boko\nHaram
```

```
as.data.frame(cbind(years, "HigherDegree (out)"=deg_out_total))
```

```
##    years             HigherDegree (out)
## 1   2000                            OPC
## 2   2001                            Tiv
## 3   2002                         Yoruba
## 4   2003                           Ijaw
## 5   2004                Military Police
## 6   2005                       Military
## 7   2006                         MASSOB
## 8   2007                           MEND
## 9   2008 Jukun Military Police Udu MEND
## 10  2009                       Military
## 11  2010                  Fulani Police
## 12  2011   Christian Fulani Boko\nHaram
## 13  2012                Military Police
## 14  2013                         Police
## 15  2014                       Military
## 16  2015                Fulani Military
## 17  2016                         Fulani
```

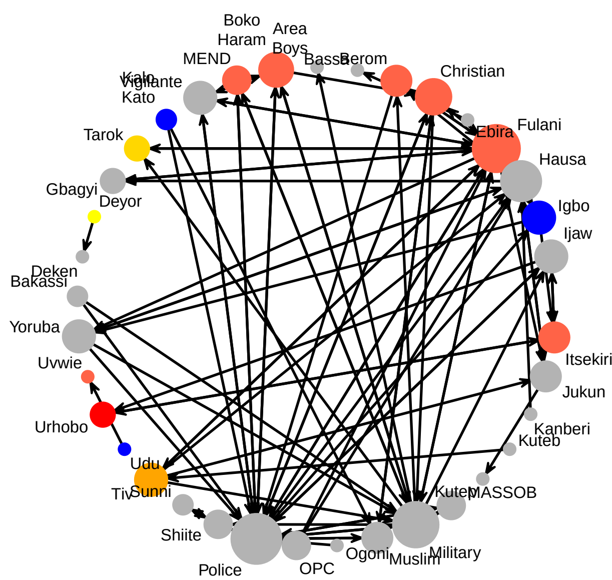
As seen bellow, k=7 seems to be the best number of groups according to both the AUC (PR) and AUC(ROC).

```r
set.seed(1234)

cross <- function(data, f=10, k=2) {
    set.seed(1234)
    folds <- createFolds(groups, k=f, returnTrain = T)
    tot_pr <- c()
    tot_roc <- c()
    for (i in 1:f) {
        nigeriaDyn2 <- data
        network::delete.vertices(nigeriaDyn2,(1:n)[-folds[[i]]])
        eclusts <- equiv.clust(nigeriaDyn2)
        BlockM <- blockmodel(nigeriaDyn2, eclusts, k=k)
        member <- BlockM$block.membership[BlockM$order.vec]

        nigeriaDyn2%v%"member" <- member
        m <- btergm(as.network.networkDynamic(nigeriaDyn2) ~ edges +
                            gwesp(.5, fixed = TRUE) + nodecov("member"))
        #probs <- edgeprob(m)

        g <- gof(m, statistics = rocpr, nsim = 50)
        tot_pr <- c(tot_pr, g$`Tie prediction`$auc.roc)
        tot_roc <- c(tot_pr, g$`Tie prediction`$auc.pr)




    }
    return(list(PR=mean(tot_pr), ROC=mean(tot_roc)))
```

```
}



pr_results <- c()
roc_results <- c()
for (k in 2:10) {
  cross_results <- cross(nigeriaDyn, k=k)
  pr_results <- c(pr_results, cross_results$PR)
  roc_results <- c(roc_results, cross_results$ROC)
}
data.frame(k=2:10,PR=pr_results, ROC=roc_results)

##    k        PR       ROC
## 1  2 0.5235649 0.4809065
## 2  3 0.5371697 0.4923879
## 3  4 0.5502108 0.5044855
## 4  5 0.5399445 0.4963676
## 5  6 0.5561239 0.5117770
## 6  7 0.5787657 0.5327670
## 7  8 0.5518961 0.5065237
## 8  9 0.5367710 0.4937206
## 9 10 0.5307155 0.4878298
```

```
eclusts <- equiv.clust(nigeriaDyn)
BlockM <- blockmodel(nigeriaDyn, eclusts, k=7)
member_or <- BlockM$block.membership[BlockM$order.vec]
```

```
nigeriaDyn%v%"member" <- member_or
#nigeriaDyn %v% "member"
nigeriaDyn %v% "col" <- c("gray70", "tomato", "gold", "yellow", "blue", "red", "orange")

plot(nigeriaDyn, label = labels, label.cex=0.5, mode="circle", vertex.cex=log(deg_all)+1
     label.col="black", vertex.col="col", vertex.border="col", edge.col="black")
```




The first logical hypothesis would be that if a reciprocal tie is present then the odds of a tie would be higher. If an actor attacks, the odd of a retaliation should be higher. For this, it is important to include the term mutual in the ERGM. Including an attribute variable of whether the actor is the Police or the Military may be another important variable to include. It should be expected no attacks between them. Another hypothesis could be that the more an actor has 2 stars the more likely that actor would attack others. This may be important given that two popular actors are in consideration, the police and the military. If two actors attack another third, it should be more likely for them not to attack themselves. For this, it is included the term triangles. And we may discount each additional tie by including the term gwidegree.

As expected the coefficient for mutual is positive so it is very likely retaliation among actors. The strong negative coefficient for the group-homophily term shows that the police and the military do not attack themselves. The triangle couldn't be estimated because there are few triangles in this network.

```
nigeria1 <- nigerialist[[1]]
nigeria1%v%"gov" <- ifelse(groups=="Police\n(Nigeria)"| groups=="Military\n(Nigeria)",1,
m = ergm(nigeria1 ~ edges + mutual+ nodematch("gov") + istar(2)+ triangle+gwidegree(deca
summary(m)
```

```
##
## ==========================
## Summary of model fit
## ==========================
##
## Formula:   nigeria1 ~ edges + mutual + nodematch("gov") + istar(2) + triangle +
##     gwidegree(decay = 0.5, fixed = TRUE)
##
## Iterations:  2 out of 20
##
## Monte Carlo MLE Results:
##                  Estimate Std. Error MCMC % p-value
## edges             0.04503    2.62266      0   0.9863
```

```
## mutual         2.37950    1.11987      0  0.0338 *
## nodematch.gov -1.09842    0.47448      0  0.0208 *
## istar2        -0.73410    0.96313      0  0.4461
## triangle          -Inf    0.00000      0  <1e-04 ***
## gwidegree     -4.80796    2.85555      0  0.0925 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##      Null Deviance: 1847  on 1332  degrees of freedom
##  Residual Deviance:  NaN  on 1326  degrees of freedom
##
## AIC: NaN    BIC: NaN    (Smaller is better.)
##
##  Warning: The following terms have infinite coefficient estimates:
##   triangle
```

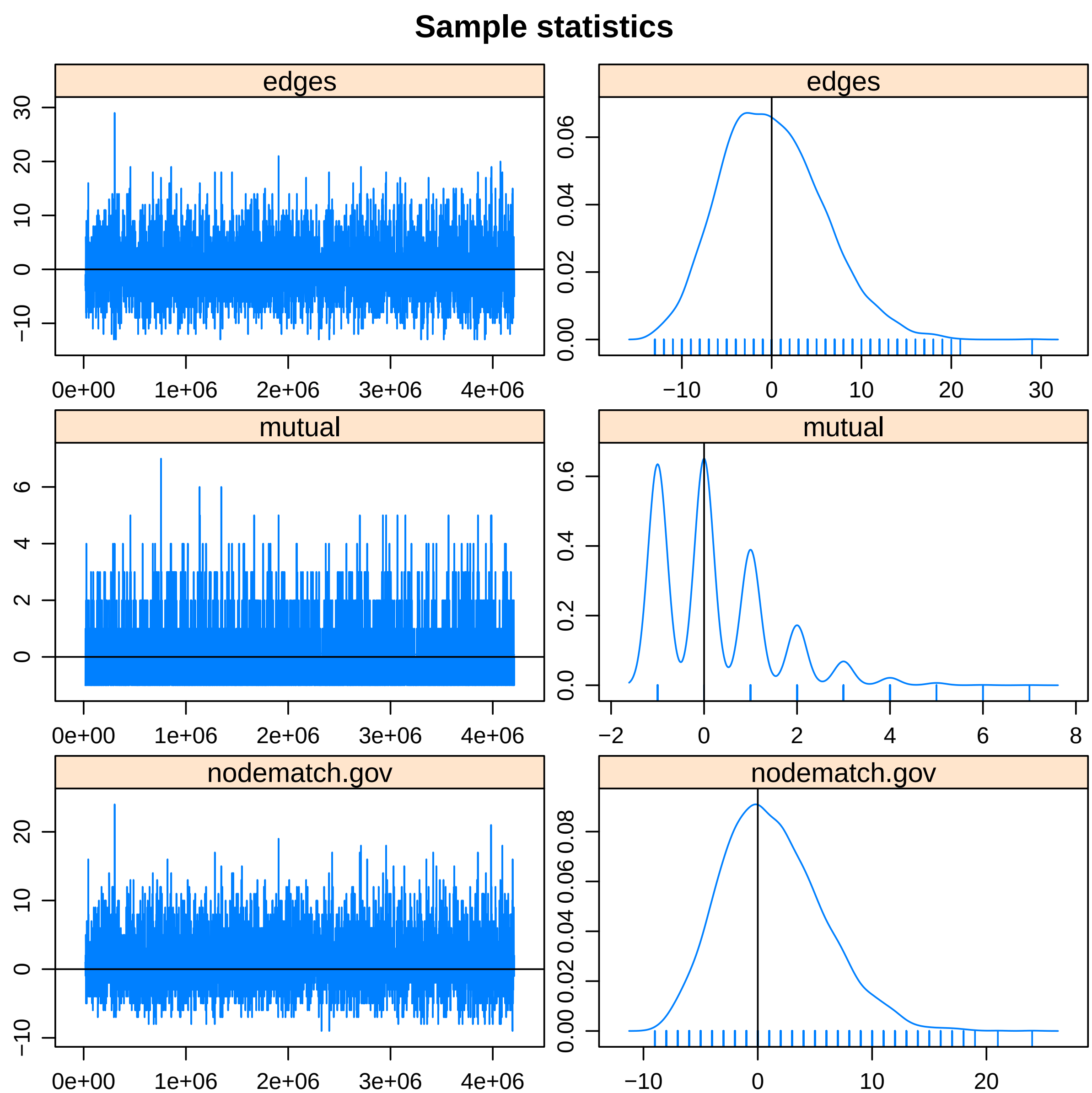
Bellow it is shown the MCM diagnostics, it seems well-mixed and with stationary chains.

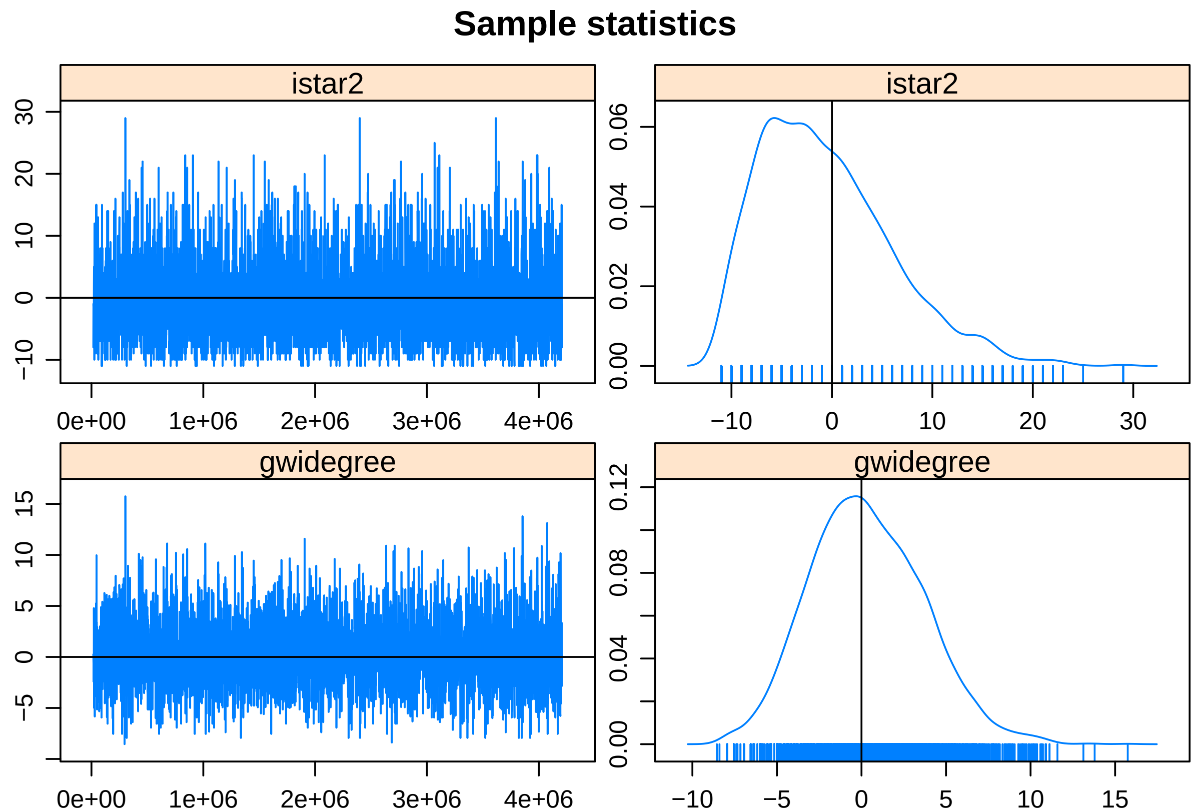
```
mcmc.diagnostics(m)

## Sample statistics summary:
##
## Iterations = 16384:4209664
## Thinning interval = 1024
## Number of chains = 1
## Sample size per chain = 4096
##
## 1. Empirical mean and standard deviation for each variable,
```

```
##    plus standard error of the mean:
##
##                  Mean    SD Naive SE Time-series SE
## edges          0.1099 5.627  0.08792        0.09474
## mutual         0.2222 1.200  0.01875        0.02110
## nodematch.gov  1.3191 4.415  0.06898        0.06898
## istar2        -0.6309 6.537  0.10214        0.11122
## gwidegree      0.3317 3.359  0.05248        0.05565
##
## 2. Quantiles for each variable:
##
##                  2.5%    25%     50%   75%  97.5%
## edges         -10.000 -4.000  0.0000 4.000 12.000
## mutual         -1.000 -1.000  0.0000 1.000  3.000
## nodematch.gov  -6.000 -2.000  1.0000 4.000 11.000
## istar2        -10.000 -6.000 -2.0000 3.000 15.000
## gwidegree      -5.848 -2.061  0.1548 2.571  7.168
##
##
## Sample statistics cross-correlations:
##                   edges    mutual nodematch.gov    istar2 gwidegree
## edges         1.0000000 0.6451016     0.8536075 0.8567962 0.9430548
## mutual        0.6451016 1.0000000     0.4661134 0.5527570 0.6085496
## nodematch.gov 0.8536075 0.4661134     1.0000000 0.6453335 0.8515740
## istar2        0.8567962 0.5527570     0.6453335 1.0000000 0.6452543
## gwidegree     0.9430548 0.6085496     0.8515740 0.6452543 1.0000000
##
```

```
## Sample statistics auto-correlation:
## Chain 1
##                  edges       mutual nodematch.gov       istar2
## Lag 0     1.0000000000  1.000000000    1.00000000  1.000000000
## Lag 1024  0.0745023118  0.117637222    0.01497454  0.084838496
## Lag 2048 -0.0016078931  0.012019314   -0.01324165 -0.000231895
## Lag 3072 -0.0112532712 -0.011026355   -0.01067052 -0.002124739
## Lag 4096  0.0002175683  0.024690048   -0.02199732  0.019639235
## Lag 5120 -0.0233379084 -0.008498069   -0.02003264 -0.022915736
##              gwidegree
## Lag 0     1.0000000000
## Lag 1024  0.0584582952
## Lag 2048 -0.0002112205
## Lag 3072 -0.0090114816
## Lag 4096 -0.0135061043
## Lag 5120 -0.0211166244
##
## Sample statistics burn-in diagnostic (Geweke):
## Chain 1
##
## Fraction in 1st window = 0.1
## Fraction in 2nd window = 0.5
##
##         edges        mutual nodematch.gov        istar2     gwidegree
##       0.12778      -0.45316      -0.39710       0.28642      -0.01315
##
## Individual P-values (lower = worse):
```

```
##          edges         mutual nodematch.gov          istar2      gwidegree
##      0.8983269      0.6504361      0.6912923      0.7745529      0.9895095
## Joint P-value (lower = worse):  0.8791296 .
```






```
##
## MCMC diagnostics shown here are from the last round of simulation, prior to computati
```