# Affiliation network analysis of real-world data in R

**Manuel S. González Canché**
msgc@email.arizona.edu
University of Arizona

**Cecilia Rios Aguilar**
Cecilia.Rios-Aguilar@cgu.edu
Claremont Graduate University

**INCHER, University of Kassel**

June 25, 2012

## Contents

# Presentation

The purpose of this document is to explain the process followed to conduct analysis of real-world affiliation networks in R. By "real-world" data we mean matrices containing data with extremely high dimensionality, reaching thousands of rows and columns. Because of this, several data mining techniques will be implemented in this tutorial.

The importance of data mining in R is that the process of extracting relevant information somehow implies deleting non-relevant information. In this sense, the manner how non-relevant information is defined becomes vital. We will be clear when describing the selection criteria to conduct the analysis.

This document is divided in three main sections, the first corresponds to a very brief contextualization of R and the data used; the second addresses the issue of dealing with high-dimensional data and prepares the affiliation network to obtain relevant information, the third accounts for the process of transforming the two-mode network to a co-membership network, as explained in the presentation.

**Note:** This document is for reference only, please do not copy any command from here. The code contained is in R language and requires "perfection." Copying and pasting from this document is likely to contain noise and consequently the functions won't work, or won't work properly. Instead use the text document called "R function Affiliation Network.R" provided during the session which will be described later in this tutorial.

# 1 Contextualization

**What is real-world or High-dimensional data?**

- Term adopted to describe massive amounts of data
- Requires more computer and packages power
- Requires to adopt **data mining** techniques

  > *The science and art of finding hidden structure in large amounts of data, dropping waste while ensuring that valuable information is kept.*

- We will analyze student's affiliations to virtual communities.
- Actors=4,064 and Communities=4,445
- Almost impossible to be handled by UCINET with regular computers.

The dataset comes from real data extracted from a Facebook application that creates virtual communities within community colleges across the United States of America. Each community college invites students to form part of this virtual community and we have requested authorizations to students to use this dataset. All identifiable information has been removed.

## 1.1 Why R?

Even though R could be difficult to learn

- Has THE MOST advanced techniques in many disciplines, including SNA.
- If UCINET does not crash when trying to read the data, the outcome is practically unusable.

**Shortest R history ever**

- R is an open source language and environment for statistical computing and graphics.
- A group of grad students in (multidisciplinary) statistical methodology did not want to pay for the package S and decided to build their own platform.
- R, like S, is designed around a true computer language, allowing users to continue creating functions and packages.
- R has more than 3,898 packages... and counting.

**R Basics**
We will be using the following commands (and more):

```
Everything after # is a comment and will be ignored
<- is used to assign values to objects
Objects can be data frames, matrices, arrays, vectors, scalars
ls() #Tells what objects are in the current workspace
setwd(‘‘path to your folder’’) #Tells R where to read/save data
dta<-read.csv("Data.csv", header=TRUE, sep=",")#Reads csv files
dim(dta) #Tells us dimensions (No. of rows and columns)
colnames(dta)[1]<-"id" #Changes the name of column 1 to id
rownames(dta)<-dta$id #Adds the value of id to actors
install.packages("package name")#Connects to an R repository
library("package name")#Makes its functions available
dta<-t(dta) #Transposes object dta
```

# 2 Analysis of two-mode, real-world data

Before running the R function we will need to locate our files as follows:

1. Locate the databases containing all the information used in this tutorial, they are called “User_to_Community_Memberships.csv” and “Community_export”
2. Copy and paste the datasets to a desired folder
3. Copy the path of the folder containing these datasets

## 2.1 R function

Depending on where you have placed the dataset the path will need to be modified accordingly. Also, Mac and Pc will have small differences in terms of how to write paths, for instance:

Possible paths:

```
#Sets workding directory
setwd("C:/Users/MSGC/Documents/Ph. D/Dropbox/Germany PHUDCFILY
/TricDataWorkshop") #Pc

setwd("/Users/msgc/Dropbox/Dropbox/Germany PHUDCFILY
/TricDataWorkshop") #mac
```

Please follow the next directions:

1. Open R.
2. Remember to modify the path based on your computer's configuration.
3. Copy and paste one of the previous paths (depending on whether you are using a mac or Pc).
4. Once you are sure about having correctly modified the path, paste it in R and hit enter. This command told R where to read the data.

The following command will import our previously modified *.csv into the R environment. Remember, this dataset needs to be located in the folder indicated by our previous path.

1. To read the data from csv format located in our working directory
   `dta<-read.csv("User_to_Community_Memberships.csv", header=TRUE, sep=","))`
2. To get the dimensions of our data frame (No. of rows and columns) we type
   `dim(dta)`
3. Changes the name of column 1 to id
   `colnames(dta)[1]<-"id"`
4. Adds the value of id to actors `rownames(dta)<-dta$id`
   `dim(dta)`
5. Removes first column to get squared matrix
   `dta<-dta[-1]`
   `dim(dta)`
6. Shows the data
   `fix(dta)`
7. Transposes the matrix
   `dta<-t(dta)`

8. Reads the second dataset which contains the names of communities
```
names<-read.csv("Community_export.csv", header=T)
dim(names)
```

9. Creates an object containing the names to be used in our plot
```
names1<-names$Title
class(names1)
length(names1)
rownames(dta)<-names1
```

10. Here we start playing with numbers of community members required to make the plot.

    This is a data mining process given that only communities with x number of members will be used.

    The first option tells R to keep only communities with at least 1 member.
```
dtm2<-dta[rowSums(dta) != 0,]
dtm2<-dta[rowSums(dta) > 10,]
dtm2<-dta[rowSums(dta) > 150,]
dtm2<-dta[rowSums(dta) > 200,]
dtm2<-dta[rowSums(dta) > 310,]
dtm2<-dta[rowSums(dta) > 330,]
dtm2<-dta[rowSums(dta) > 340,]
dtm2<-dta[rowSums(dta) > 350,]
dtm2<-dta[rowSums(dta) > 390,]
dtm2<-dta[rowSums(dta) > 400,]
dtm2<-dta[rowSums(dta) > 415,]
dim(dtm2)
```

11. Here we tell R to transppose our matrix again, so that we have our original matrix back
```
dtm2<-t(dtm2)
```

12. To tell R to keep only participants who belong to at least 1 community, but we can change this criterion as well to increase the minimum number of communities to be kept.
```
dtm2<-dtm2[rowSums(dtm2) != 0,] dim(dtm2)
```

13. To see the names of the communities
```
colnames(dtm2)
```

14. If we need to make the names shorter, we can do so
```
colnames(dtm2)<-c("Music","Fam & Friends","Laugh'g",
"Watch'g Movies","Mak'g Friends","Meet'g People","Movies",+
"Comedy","Sleep'g", "Road Trips", "Travel", "Barbecues",
"Hispanic", "2013","Stay'g Up Late","Vacation", "Pets",+
"Reading","Work'g out","Family Guy")
```

15. Installing the package to be used for SNA
```
install.packages("igraph")
```

16. Loading the package
```
library(igraph)
```

17. Creating a two-mode graph object using our matrix (dtm2)
```
 g <- graph.incidence(dtm2, mode=c("all"))
```

18. UCINET problem plotting the data is universal, even R has difficulties doing so
```
plot(g)
```

19. The following steps will be used to address this issue and make sense of the sociogram
We need to get vectors accounting for the length of rows and columns
```
nActors <- nrow(dtm2)
nComms <- ncol(dtm2)
idx.Actors <- 1:nActors
idx.Comms <- (nActors1):(nActors+nComms)+
V(g)$degree <- degree(g)
V(g)$size <- 1.4*V(g)$degree/max(V(g)$degree)
```

20. The following two command will render a warning message because R will detect that the length of the criteria we are assigning is different, that is Ok in this case because we are fixing the problem
```
V(g)$color[idx.Comms] <- rgb(1, 0, 0, .4)
V(g)$color[idx.Actors] <- rgb(0, 1, 0, .4)
V(g)$frame.color <- NA
```

21. Adding names to vertices (actors)
```
V(g)$label <- V(g)$name
```

22. Adding color to vertices (actors)
```
V(g)$label.color[idx.Actors] <- rgb(0, 0, 0, 0.5)
```

23. Adding names to columns (communities)
```
V(g)$label.color[idx.Comms] <- "red"
```

24. Adding weights to names based on degree centrality, this is just for convenience, we can use eigenvector or betweenness degree measures as well.
```
V(g)$label.cex <- 1.4*V(g)$degree/max(V(g)$degree)  1+
pdf("COMM2TRIC2MOdePHUDCFILY.pdf",width=34,height=35)
```

25. Kamada-Kawai layout allows for a better distribution of the map.
```
plot(g, layout=layout.kamada.kawai)
title(main="Top 16 communities",
sub= "TRIC, communities with 416 or more members",
col.main="Black", cex.sub=3.5,cex.main=3.5,font.sub=2)
dev.off()
```

# 3 Transforming two-mode to co-membership mode

In this section we will see how to transform affiliation networks to co-membership networks. This step in one sense is a measure of the strenght of ties since it will render information about the number of communities students belong to, and the number of other students are co-attending to these communities.

We will continue with the use of our virtual community dataset, which should be in R workspace. Specifically, we will work with the affiliation network containing 2,092 actors who belong to the 16 most important virtual communities. The R object that contains this dataset is called dtm2.

It is also important to mention that we will assign weight to the edges (lines or ties) so that we have a sense of the co-membership of participants.

## 3.1 R function

1. We need it to be a matrix
   ```
   dtm2<-as.matrix(dtm2)
   ```

2. Now we can do the matrix multiplication $A * A^T$ to get co-memberships
   ```
   dtm2<- dtm2 %*% t(dtm2)
   #dtm2<- t(dtm2) %*% dtm2 #For communities overlap
   ```

3. Checking if this worked
   ```
   dim(dtm2)
   ```

4. Getting the graph object
   ```
   g <- graph.adjacency(dtm2, mode = "undirected")
   ```

5. Now we need to transform the graph so that multiple edges become an attribute `(E(g)$weight)` of each unique edge:
   ```
   E(g)$weight <- count.multiple(g)
   ```

6. Simplify removes loops (me, selecting me) and multiple edges (subjects selecting themselves more than once)
   ```
   g <- simplify(g)
   ```

7. Adding names to vertices (actors)
   ```
   V(g)$label <- V(g)$name
   ```

8. Getting degree centrality (this improves size)
   ```
   V(g)$degree <- degree(g)
   ```

9. Making sociogram reproducible by setting initial conditions of layout algorithm
   ```
   set.seed(3952)
   ```

10. Managing the size of labels
    ```
    V(g)$label.cex <-((V(g)$degree+1)^1.5)*
    .001#5 * V(g)$degree / max(V(g)$degree)+.5
    ```

11. Label color
```
V(g)$label.color <- "black"
```

12. No frame
```
V(g)$frame.color <- NA
V(g)$color<-rgb(0,0,0,255/17,maxColorValue=255)
```

13. Managing size of node
```
V(g)$size <- 6 * V(g)$degree/max(V(g)$degree)+1
```

14. Creating variable to give variation to weight of lines
```
egam<-(log(E(g)$weight)+.4)/max(log(E(g)$weight)+.4)
E(g)$color<-rgb(47,79,79,255/4,maxColorValue=255)
```

15. Adding weight to lines or ties
```
E(g)$width <- egam
```

16. Creating pdf
```
pdf("TRICONEMODEUSERPHUDCFILY.pdf",width=24,
height=25, bg=rgb(255, 127, 0,255/1,maxColorValue=255))
plot(g, layout=layout.kamada.kawai)
title(main="Top 2092 community members",
sub="TRIC, Community members", col.main="black",
col.sub="black", cex.sub=3.5,cex.main=3.5,font.sub=2)
dev.off()
```

**Warning:** The code just saw will produce a 20 megabytes pdf that is just not useful. As before, we may want to concentrate on actors with very active participation. To clean the data we want to run the following code:

1. Exploring the dimensionality of dtm2
```
dim(dtm2)
```

2. Now since dtm2 is a matrix, we can get its diagonal, remember that this diagonal accounts for the total number of communities students are affiliated to.
```
fix(dtm2)
```

3. We can get the summary statistics of this diagonal which is 2,092
```
summary(diag(dtm2))
```

4. Now we are ready to remove participants getting a reasonable cut off point, like more than ten communities at least
dtm3<-dtm2[diag(dtm2) >10, diag(dtm2) >10]

5. Now we are ready to replicate the ploting, using dtm3 instead!

```
#Replicating the sociogram
g <- graph.adjacency(dtm3, mode = "undirected")
E(g)$weight <- count.multiple(g)
```

```
g <- simplify(g)
V(g)$label <- V(g)$name
V(g)$degree <- degree(g)
set.seed(3952)
# layout1 <- layout.fruchterman.reingold(g)
# plot(g, layout=layout1)
# plot(g, layout=layout.kamada.kawai)
V(g)$label.cex <- ((V(g)$degree+1)^1.5)*.001
V(g)$label.color <- "black"#rgb(255, 127, 0,255/1,maxColorValue=255)
V(g)$frame.color <- NA
V(g)$color <- rgb(0, 0, 0,255/17,maxColorValue = 255)
V(g)$size <- 6 * V(g)$degree / max(V(g)$degree)+ 1
egam <- (log(E(g)$weight)+.4) / max(log(E(g)$weight)+.4)
E(g)$color <- rgb(47, 79, 79,255/4,maxColorValue = 255)
E(g)$width <- egam
pdf("C:\\Users\\MSGC\\Documents\\Ph. D\\Dropbox\\Cloud of words
phudcfily\\NewDataGatesPHUDCFILY\\gates_text_data\\TRICNEWPHUDCFILY
\\TRICONEMODEUSERPHUDCFILY.pdf",width=24,height=25,
bg=rgb(255, 127, 0,255/1,maxColorValue=255))
plot(g, layout=layout.kamada.kawai)
title(main="Top 148 community members", sub="TRIC,
Participants belonging to at least 11 communities", col.main="black",
col.sub="black", cex.sub=3.5,cex.main=3.5,font.sub=2)
dev.off()
```