National Research University Higher School of Economics

Introduction to Network Analysis

Materials to Accompany Lectures 2-3: Network Analysis as a Method

Valentina Kuskova, PhD

Contents

Welcome!

Welcome to the third seminar! Today we are moving into numerical descriptive analytics of network - calculating various network statistics. For this seminar, you will need materials from Lecture 2-3, which will be fully complete by the time this seminar takes place. However, this seminar also provides additional information and explanations to help you get through some of the more difficult definitions in the seminar.

Contents of today's seminar:

- In your Seminar 3 folder, you will find the following files:

1. This file, "Seminar 3" in .pdf format.
2. Dataset "flo.Rdata," which we've seen before; also, in the same format - "madmen.Rdata" and "trade.Rdata.""

- For today's assignment, you will need the following packages (remember, 'R' is case sensitive, make sure, when installing packages, to keep the appropriate case:
  1. 'RColorBrewer' to color our objects
  2. 'network'
  3. 'sna'
  4. 'igraph'. *Please note:* this package may conflict with "network" package, especially on graphics. Therefore, when using one, it is always wise to unload the other (using command 'detach(package:packagename)', or you may get an error in execution.
  5. 'intergraph'
  6. 'CINNA' for calculating centralities
  7. 'knitr' for making pretty output

  Remember that to use a certain package, you need to make sure that the library is installed (using 'install.packages("packagename")' command). To use a loaded package, you call it with 'library.(packagename)' command.
- After completing today's seminar, you should be able to accomplish the following:
  1. Get familiar with other network packages in 'R'.
  2. Learn how to calculate and interpret network characteristics that we have covered in lecture.
  3. Compare and interpret obtained network results. That's right, you are already network analysts!

Seminar 3 assignment. Please answer the questions that are marked as *Assignment questions*. All previous submission rules apply.

Calculating network statistics

While it was a lot of fun drawing graphs (and we will continue to learn new things about them with each seminar), it is important to start generating statistical information about our networks.

Basic network measures

First, load the `sna` package:

```
suppressPackageStartupMessages(library(sna))
```

Basic network measures are the items we talked about in detail in Lecture 2. For this assignment, we go back to our familiar dataset, Florentine Families.

```
# Load data and create networks:
load('flo.Rdata')
flo.marriage<-as.network(flo.marriage)
```

And, onto the code for the quantities we have discussed. We start with basics: counts of ties, dyads, and triads; and network density.

Basic network statistics:

```
# The basics:
network.dyadcount(flo.marriage) #self-explanatory, dyads
```

```
## [1] 240
```

```
dyad.census(flo.marriage) # types and count of dyads
```

```
##      Mut Asym Null
## [1,]  20    0  100
```

```
network.edgecount(flo.marriage) # number of ties
```

```
## [1] 40
```

```
network.density(flo.marriage) #density of an overall network
```

```
## [1] 0.1666667
```

```
triad.census(flo.marriage) # types and counts of triads
```

```
##      003 012 102 021D 021U 021C 111D 111U 030T 030C 201 120D 120U 120C 210 300
## [1,] 324   0 195    0    0    0    0    0    0    0  38    0    0    0   0   3
```

Assignment question 1

This assignment refers to "triadic census" part of Lecture 2-3, so you should be able to answer this question after the Lecture on Friday, February 11th. Given the information we've discussed about triads, what conclusions can you make about the results of the triadic census analysis for the Florentine Family network?

Transitivity

A property very important in social networks, and to a lesser degree in other networks, is transitivity [1]. If refers to the extent to which the relation that relates two nodes in a network that are connected by an edge is transitive. Perfect transitivity implies that, if x is connected (through an edge) to y, and y is connected to z, then x is connected to z as well. It is very rare in real networks, since it implies that each component is a clique, that is, each pair of reachable nodes in the graph would be connected by an edge.

However, partial transitivity is useful. In many networks, particularly social ones, the fact that x knows y and y knows z does not guarantee that x knows z as well, but makes it much more likely. The friend of my friend is not necessarily my friend, but is far more likely to be my friend than some randomly chosen member of the population.

We can quantify transitivity in an undirected network as follows. A loop of length three is a sequence of nodes x, y, z, x such that (x, y), (y, z) and (z, x) are edges of the graph. Notice that this is the smallest loop possible in an undirected network. A path of length two is a sequence of nodes x, y, z such that (x, y) and (y, z) are edges (the edge (z, x) can be present or not). Notice that paths (and loops) are sequences, hence the loop x, y, z, x is different from y, z, x, y. The transitivity coefficient T of a network, also known as clustering coefficient, is the ratio of the number of loops of length three and the number of paths of length two. Hence, it is the frequency of loops of length three in the network.

$T = 1$ implies perfect transitivity, i.e., a network whose components are all cliques. $T = 0$ implied no closed path of length two, which happens for various typologies, such as a tree or a square lattice. Social networks tend to have quite high values of the transitivity coefficient. For instance, the transitivity coefficient of the computer science collaboration network is 0.24. This means that, on average, the chance that two scholars that share a common collaborator wrote a paper together is almost one-fourth. This is a rather high probability, indeed. As a comparison, the transitivity coefficient for a random network of the same size is $9.6 \cdot 10^{-6}$. The transitivity coefficient of a random graph with n nodes and m edges is simply the density

[1] Source: https://www.sci.unich.it/~francesc/teaching/network/transitivity.html

of edges m/m_*, where $m_* = n(n-1)/2$ is the maximum number of edges of a graph of n nodes. However, non-social networks have lower transitivity coefficients. For the Internet, for example, the transitivity is only 0.012, against an expected value, if connections were made at random, of 0.84. This suggests that in the Internet there are forces at work that shy away from the creation of triangles. In some other networks, such as food webs or the Web, transitivity is neither higher nor lower than expected.

```
gtrans(flo.marriage, measure='weak') # transitivity
```

```
## [1] 0.1914894
```

Assignment question 2

Given the explanation of transitivity provided, how do you evaluate the transitivity coefficient obtained with the above command?

Reciprocity

The reciprocity of a directed network reflects the proportion of edges that are symmetrical \footnote{Source: https://campus.datacamp.com/courses/network-analysis-in-r/identifying-special-relationships?ex=4#:~:text=The%20reciprocity%20of%20a%20directed,inter%2Dconnected%20directed%20networks%20are.. That is, the proportion of outgoing edges that also have an incoming edge. It is commonly used to determine how inter-connected directed networks are.That is, the proportion of outgoing edges that also have an incoming edge. It is commonly used to determine how inter-connected directed networks are.

The following description of reciprocity is given by the `help` files on reciprocity: The dyadic reciprocity of a graph is the proportion of dyads which are symmetric; this is computed and returned by grecip for the graphs indicated. (dyadic.nonnull returns the ratio of mutuals to non-null dyads.) Note that the dyadic reciprocity is distinct from the edgewise or tie reciprocity, which is the proportion of edges which are reciprocated. This latter form may be obtained by setting measure="edgewise". Setting measure="edgewise.lrr" returns the log of the ratio of the edgewise reciprocity to the density; this is measure (called r_4 by Butts (2008)) can be interpreted as the relative log-odds of an edge given a reciprocation, versus the baseline probability of an edge. Finally, measure="correlation" returns the correlation between within-dyad edge values, where this is defined by

$$\frac{2\sum_{\{i,j\}}(Y_{ij}-\mu_G)(Y_{ji}-\mu_G)}{(2N_d-1)\sigma_G^2}$$

with Y being the graph adjacency matrix, μ_G being the mean non-loop edge value, σ_G^2 being the variance of non-loop edge values, and N_d being the number of dyads. (Note that this quantity is unaffected by dyad orientation.) The correlation measure may be interpreted as the net tendency for edges of similar relative value (with respect to the mean edge value) to occur within the same dyads. For dichotomous data, adjacencies are interpreted as having values of 0 (no edge present) or 1 (edge present), but edge values are used where supplied. In cases where all edge values are identical (e.g., the complete or empty graph), the correlation reciprocity is taken to be 1 by definition.

Note that grecip calculates values based on non-missing data; dyads containing missing data are removed from consideration when calculating reciprocity scores (except for the correlation measure, which uses non-missing edges within missing dyads when calculating the graph mean and variance).

```
grecip(flo.marriage, measure = 'dyadic') # why not 0?
```

```
## Mut
##   1
```

```
grecip(flo.marriage, measure = 'dyadic.nonnull')
```

```
## Mut
##   1
```

```
grecip(flo.marriage, measure = 'edgewise')
```

```
## Mut
##   1
```

```
grecip(flo.marriage, measure = 'edgewise.lrr')
```

```
##      Mut
## 1.791759
```

```
grecip(flo.marriage, measure = 'correlation')
```

```
## [1] 1
```

Assignment question 3

Here, we showed you how to calculate this measure and its different types, but the information you get will be meaningless. Why?

Geodesic distances

A shortest path, or geodesic path, between two nodes in a graph is a path with the minimum number of edges [2]. If the graph is weighted, it is a path with the minimum sum of edge weights. The length of a geodesic path is called geodesic distance or shortest distance. Geodesic paths are not necessarily unique, but the geodesic distance is well-defined since all geodesic paths have the same length.

One large-scale property of networks is the average geodesic distance between pairs of nodes in the network. One of the most discussed network phenomena is the small-world effect. In most real network the typical geodesic distance is surprisingly short, in particular when compared with the number of nodes of the network.

Because geodesic distance are different for every pair of nodes (you understand why, right?), we actually get a matrix of values that indicate the length of the shortest path between those nodes. Because it's a matrix, it makes sense to assign it to an object to do additional work on it:

```
# The command is simply geodist, but we will assign it to an object:
geo.dist<-geodist(flo.marriage)
class(geo.dist) # Check the structure
```

```
## [1] "list"
```

```
# Ok, it's a list, let's look at it:
summary(geo.dist)
```

```
##        Length Class  Mode
## counts 256    -none- numeric
## gdist  256    -none- numeric
```

We can also summarize individual elements. These summaries give us all relevant information for all nodes.

```
summary(geo.dist$counts)
```

```
##        V1              V2              V3              V4              V5
##  Min.   :0.00   Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000
##  1st Qu.:1.00   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000
##  Median :1.00   Median :1.000   Median :1.000   Median :1.000   Median :1.000
##  Mean   :1.25   Mean   :1.062   Mean   :1.125   Mean   :1.562   Mean   :1.188
##  3rd Qu.:1.25   3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:2.250   3rd Qu.:1.250
##  Max.   :3.00   Max.   :2.000   Max.   :2.000   Max.   :3.000   Max.   :2.000
```

[2] Source: https://www.sci.unich.it/~francesc/teaching/network/geodesic.html

```
##        V6              V7              V8              V9             V10
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.00   Min.   :0.00
##  1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.00   1st Qu.:1.00
##  Median :1.000   Median :1.000   Median :1.000   Median :1.00   Median :1.00
##  Mean   :1.062   Mean   :1.312   Mean   :1.312   Mean   :1.25   Mean   :1.25
##  3rd Qu.:1.000   3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:1.25   3rd Qu.:1.25
##  Max.   :2.000   Max.   :2.000   Max.   :2.000   Max.   :3.00   Max.   :3.00
##       V11             V12             V13             V14
##  Min.   :0.00   Min.   :0.0000   Min.   :0.0000   Min.   :0.00
##  1st Qu.:1.00   1st Qu.:0.0000   1st Qu.:1.0000   1st Qu.:1.00
##  Median :1.00   Median :0.0000   Median :1.0000   Median :1.00
##  Mean   :1.25   Mean   :0.0625   Mean   :0.9375   Mean   :1.25
##  3rd Qu.:2.00   3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:1.25
##  Max.   :2.00   Max.   :1.0000   Max.   :1.0000   Max.   :3.00
##       V15             V16
##  Min.   :0.000   Min.   :0.000
##  1st Qu.:1.000   1st Qu.:1.000
##  Median :1.000   Median :1.000
##  Mean   :1.062   Mean   :1.188
##  3rd Qu.:1.000   3rd Qu.:1.250
##  Max.   :2.000   Max.   :2.000
```

```
summary(geo.dist$gdist)
```

```
##        V1              V2              V3              V4              V5
##  Min.   :0.00   Min.   :0.00   Min.   :  0   Min.   :0.00   Min.   :0.00
##  1st Qu.:2.00   1st Qu.:1.75   1st Qu.:  2   1st Qu.:1.75   1st Qu.:1.75
##  Median :3.00   Median :2.00   Median :  2   Median :2.00   Median :3.00
##  Mean   : Inf   Mean   : Inf   Mean   :Inf   Mean   : Inf   Mean   : Inf
##  3rd Qu.:3.25   3rd Qu.:3.00   3rd Qu.:  3   3rd Qu.:3.25   3rd Qu.:3.25
##  Max.   : Inf   Max.   : Inf   Max.   :Inf   Max.   : Inf   Max.   : Inf
##        V6              V7              V8              V9             V10
##  Min.   :0.00   Min.   :  0   Min.   :  0   Min.   :0.00   Min.   :0.00
##  1st Qu.:2.75   1st Qu.:  1   1st Qu.:  2   1st Qu.:1.00   1st Qu.:3.00
##  Median :3.00   Median :  2   Median :  3   Median :2.00   Median :3.50
##  Mean   : Inf   Mean   :Inf   Mean   :Inf   Mean   : Inf   Mean   : Inf
##  3rd Qu.:4.00   3rd Qu.:  3   3rd Qu.:  4   3rd Qu.:2.25   3rd Qu.:4.25
##  Max.   : Inf   Max.   :Inf   Max.   :Inf   Max.   : Inf   Max.   : Inf
##       V11             V12             V13             V14             V15
##  Min.   :0.00   Min.   :  0   Min.   :0.00   Min.   :0.00   Min.   :  0
##  1st Qu.:1.75   1st Qu.:Inf   1st Qu.:1.75   1st Qu.:2.00   1st Qu.:  1
##  Median :3.00   Median :Inf   Median :2.00   Median :2.50   Median :  2
##  Mean   : Inf   Mean   :Inf   Mean   : Inf   Mean   : Inf   Mean   :Inf
##  3rd Qu.:4.00   3rd Qu.:Inf   3rd Qu.:2.25   3rd Qu.:3.25   3rd Qu.:  3
##  Max.   : Inf   Max.   :Inf   Max.   : Inf   Max.   : Inf   Max.   :Inf
##       V16
##  Min.   :0.00
##  1st Qu.:1.75
##  Median :2.00
##  Mean   : Inf
##  3rd Qu.:3.00
##  Max.   : Inf
```

Now, that's a lot of information, and for large networks especially, it makes reading the output difficult. Therefore, we can run the code for one node at a time:

```
summary(geo.dist$counts[,3])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.000   1.000   1.125   1.000   2.000
```

```
summary(geo.dist$counts[3,])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.000   1.000   1.125   1.000   2.000
```

Assignment question 4

What can you say about the last two commands? Why is the result what it is?

Hint. Remember, we denote the matrix of data as $[r, c]$, where r is the row number and c is the column number (please go back to Seminar 1 where we looked at matrices.) Also, the paths from node A to node B are made *from* row (the first number in the matrix brackets) *to* column (the second number in the matrix brackets. So in the first command, the path is made *to* Node 3, and in the second - *from* Node 3. So what does that mean in the context of an existing network?

We can also calculate the average geodesic distance and some other measures, but `sna` package is not the best for this purpose. `igraph` contains many more commands for distances, so let's switch packages. We'll also need to convert our `network` object into the `igraph` object, and we use the `intergraph` package for it.

Remember, you need to install the `intergraph` package using `install.packages("intergraph")` command first. DO NOT install it from the markdown! It will give you an error message.

```
suppressPackageStartupMessages(library(intergraph))
flo.graph<-asIgraph(flo.marriage)
```

Let's make sure we have an igraph:

```
class(flo.graph)
```

```
## [1] "igraph"
```

Now, we are ready to start calculations. First, let's compute the average geodesic distance. Notice there are two different ways to do so:

```
#detach(package:sna)
suppressPackageStartupMessages(library(igraph))
# Two ways to calculate geodesic distances
mean_distance(flo.graph, directed = FALSE, unconnected = TRUE) #First method
```

```
## [1] 2.485714
```

```
average.path.length(flo.graph) #Second method
```

```
## [1] 2.485714
```

Assignment task 5

Please go to the following help page: https://igraph.org/r/doc/distances.html. Examine various ways to calculate distances in a graph. Calculate *one* measure not shown above for our Florentine Families network. Remember to use the graph object!

Centrality

Centrality plays a paramount role in network analysis. In fact, on the node level, everything starts and ends with centrality. There are over 400 types of centrality, and I encourage you to explore them on your own.))

I will show you a few and tell you where to find more.

We continue to work with `igraph` package, because it has more centralities built-in, and they are easier to extract.

Degree centrality

Degree centrality, as expected, tells us the incoming and outgoing ties in different combinations. There are three most common measures of degree centrality: indegree, outdegree and Freeman's degree. **Indegree centrality** measures how many people are directly connected to the individual. **Outdegree centrality** measures how many people the actor directly connects to.

```
library(igraph) #do not forget to attach the library
indegree <- degree(flo.graph, mode="in")
outdegree<-degree(flo.graph, mode="out")
total<-degree(flo.graph, mode="total") # This is also called Freeman's degree
```

This centrality is intuitive and does not require much explanation.

Betweenness, closeness, eigenvector

Three other common centrality measures - betweenness, closeness and eigenvector - are similarly easily calculated. **Betweenness centrality** measures the number of shortest paths going through a specific vertex; it is returned by the betweenness() function:

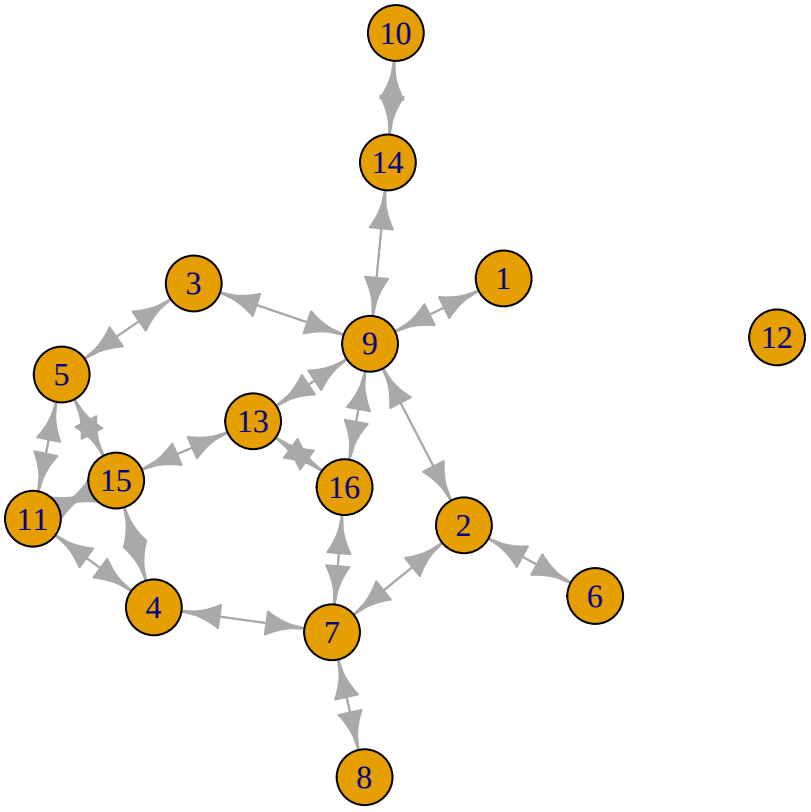
```
between<-betweenness(flo.graph)
between
```

```
##  [1]  0.00000 38.66667 17.00000 19.00000 10.00000  0.00000 46.33333  0.00000
##  [9] 95.00000  0.00000  4.00000  0.00000 20.66667 26.00000 18.66667 16.66667
```

Closeness is the mean geodesic distance between a given node and all other nodes with paths from the given node to the other node. This is close to being the mean shortest path, but geodesic distances give higher values for more central nodes. In a directed network, we can think of in-closeness centrality as the average number of steps one would have to go through to get TO a given node FROM all other reachable nodes in the network. Out-closeness centrality, not surprisingly, measures the same thing with the directionality reversed.

Closeness, as many centralities, is finicky - it only works well for connected graphs. Let's look at our graph:

```
par(mar=c(0,0,0,0))
plot(flo.graph)
```



There is one node that is disconnected from the rest. You should, however, still get closeness centrality to calculate, even though you might get a warning:

```
close<-closeness(flo.graph, mode="in") #For out-closeness, mode will be mode="out."
```

```
## Warning in closeness(flo.graph, mode = "in"): At centrality.c:2784 :closeness
## centrality is not well-defined for disconnected graphs
```

```
#For these data, it's irrelevant as the data are not directed.
close
```

```
##  [1] 0.018518519 0.022222222 0.020833333 0.019607843 0.019230769 0.017241379
##  [7] 0.021739130 0.016949153 0.024390244 0.015384615 0.018518519 0.004166667
## [13] 0.022727273 0.019230769 0.020833333 0.022222222
```

Eigenvector centrality gives greater weight to a node the more it is connected to other highly connected nodes. A node connected to five high-scoring nodes will have higher eigenvector centrality than a node connected to five low-scoring nodes. Thus, it is often interpreted as measuring a node's network importance.

In directed networks, there are 'In' and 'Out' versions. In information flow studies, for instance, In-Eigenvector scores would reflect which nodes are high on receiving information, while Out-Eigenvector scores would reflect which nodes are high on broadcasting information.

For these data, we will simply symmetrize to generate an undirected eigenvector centrality score.Eigenvector centrality takes a couple of steps. Note that, unlike the other centrality measures, evcent() command returns a complex object rather than a simple vector. Thus, we need to first get the evcent() output and then select the eigenvector scores from it.

```
ecentV<-evcent(flo.graph)
eigen<-ecentV$vector
```

```
eigen
```

```
##  [1] 0.3071155 0.5669336 0.4919853 0.6572037 0.6019551 0.1741141 0.6718805
##  [8] 0.2063449 1.0000000 0.1041427 0.6407743 0.0000000 0.7937398 0.3390994
## [15] 0.8272688 0.7572302
```

Looking at all centralities at once

For substantive analysis, we want to compare degrees to each other. The easiest way to do so is by putting them all in one table. But I also want to show you the capabilities of yet another package in R, *knitr*.

What I do first is create a table that will contain the node names (we'll pull them out of the attributes file), and then all centralities we've obtained:

```
node<-flo.att$flo.names # If you look at the attributes file, you'll find the names of families in the
#Now, create the table that we need:
table<-data.frame(node, indegree, outdegree, between, close, eigen)
#Change the header names
names(table)<-c("Family","Indegree","Outdegree","Betweenness","Closeness","Eigenvector")
```

Now that the prep work is done, let's create the nice-looking table:

```
suppressPackageStartupMessages(library(knitr))
```

```
## Warning: package 'knitr' was built under R version 4.0.5
```

```
kable(table, digits=3, caption = "Florentine Families' Network Centralities")
```

Table 1: Florentine Families' Network Centralities

Family	Indegree	Outdegree	Betweenness	Closeness	Eigenvector
ACCIAIUOL	1	1	0.000	0.019	0.307
ALBIZZI	3	3	38.667	0.022	0.567
BARBADORI	2	2	17.000	0.021	0.492
BISCHERI	3	3	19.000	0.020	0.657
CASTELLAN	3	3	10.000	0.019	0.602
GINORI	1	1	0.000	0.017	0.174
GUADAGNI	4	4	46.333	0.022	0.672
LAMBERTES	1	1	0.000	0.017	0.206
MEDICI	6	6	95.000	0.024	1.000
PAZZI	1	1	0.000	0.015	0.104
PERUZZI	3	3	4.000	0.019	0.641
PUCCI	0	0	0.000	0.004	0.000
RIDOLFI	3	3	20.667	0.023	0.794
SALVIATI	2	2	26.000	0.019	0.339
STROZZI	4	4	18.667	0.021	0.827
TORNABUON	3	3	16.667	0.022	0.757

Looks nice, does it not? Now, let's look at centrality correlations.

Correlating centrality measures

There are many ways to manipulate data frame into a matrix to create corerlations from. I will just transform the data frame into a matrix of correlations directly and then just clean it up:

```
# Set dimensions for table of correlations
cor_table<-matrix(nrow=5, ncol=5) #We have five centralities
# Apply the cor.test function to correlate measures in our table
cor_table[lower.tri(cor_table)] <- apply(which(lower.tri(cor_table), arr.ind=TRUE)+1,
1, function(a) cor.test(table[,a[1]], table[,a[2]])$estimate)

# Let's look at what we got:
cor_table
```

```
##           [,1]      [,2]      [,3]      [,4] [,5]
## [1,]        NA        NA        NA        NA   NA
## [2,] 1.0000000        NA        NA        NA   NA
## [3,] 0.8333763 0.8333763        NA        NA   NA
## [4,] 0.7572778 0.7572778 0.5763487        NA   NA
## [5,] 0.9404647 0.9404647 0.6737162 0.7989789   NA
```

Now, that does not look pretty and it has some redundant information. For one, there are no diagonal elements, and the first row and last column are containing empty values (for obvious reasons). We might want to drop them - there is no reason to keep them. Also, we should add column and row names - for a 5x5 table, it may not be as important as for a 12x12 or something of that nature.

```
# Give the same column names as to our original table
colnames(cor_table)<-c("Indegree","Outdegree","Betweenness","Closeness","Eigenvector")
rownames(cor_table)<-c("Indegree","Outdegree","Betweenness","Closeness","Eigenvector")
cor_table<-cor_table[2:5,1:4] # drop the extra row and column
kable(cor_table, caption="Correlations of Florentine Families' Network Centralities")
```

Table 2: Correlations of Florentine Families' Network Centralities

	Indegree	Outdegree	Betweenness	Closeness
Outdegree	1.0000000	NA	NA	NA
Betweenness	0.8333763	0.8333763	NA	NA
Closeness	0.7572778	0.7572778	0.5763487	NA
Eigenvector	0.9404647	0.9404647	0.6737162	0.7989789

The many other centralities

Normally, I would stop my seminar on centralities right about here. However, just a few basic ones are not enough to understand networks. So here we go: here is a glimpse at about one-tenth of other centralities that are out there.

There is a package called `CINNA` (Deciphering Central Informative Nodes in Network Analysis), which calculates some very large number of possible centralities. It's a bit finicky: for example, it would only calculate them on the `igraph` objects, and only for the connected components of the network. On one hand, it makes sense; on another, you have to first break up your network into components and then calculate what you need on individual components. But the results are worth it, so let's give it a try.

```
library(CINNA)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
#First, extract components from the flo.graph network:
comps<-graph_extract_components(flo.graph)
#Let's look at them:
```

```
comps
```

```
## [[1]]
## IGRAPH ace7de5 D--- 15 40 --
## + attr: na (v/l), vertex.names (v/c), na (e/l)
## + edges from ace7de5:
##  [1]  9-> 1  6-> 2  7-> 2  9-> 2  5-> 3  9-> 3  7-> 4 11-> 4 14-> 4  3-> 5
## [11] 11-> 5 14-> 5  2-> 6  2-> 7  4-> 7  8-> 7 15-> 7  7-> 8  1-> 9  2-> 9
## [21]  3-> 9 12-> 9 13-> 9 15-> 9 13->10  4->11  5->11 14->11  9->12 14->12
## [31] 15->12  9->13 10->13  4->14  5->14 11->14 12->14  7->15  9->15 12->15
##
## [[2]]
## IGRAPH ace7de5 D--- 1 0 --
## + attr: na (v/l), vertex.names (v/c), na (e/l)
## + edges from ace7de5:
```

We obviously need the first component, because it contains the large connected component as opposed to the single node. The rest of the work we do with that. There is a command, *"proper_centralities"*, that allows us to see all centralities that could be calculated on a given component. There are many of them, so it's best to give the object a name when we create a list of possible centralities.

```
Comp1<-comps[[1]] #We use the first component to extract centralities from
pr_cent<-proper_centralities(Comp1)
```

```
##  [1] "Alpha Centrality"
##  [2] "Bonacich power centralities of positions"
##  [3] "Page Rank"
##  [4] "Average Distance"
##  [5] "Barycenter Centrality"
##  [6] "BottleNeck Centrality"
##  [7] "Centroid value"
##  [8] "Closeness Centrality (Freeman)"
##  [9] "ClusterRank"
## [10] "Decay Centrality"
## [11] "Degree Centrality"
## [12] "Diffusion Degree"
## [13] "DMNC - Density of Maximum Neighborhood Component"
## [14] "Eccentricity Centrality"
## [15] "Harary Centrality"
## [16] "eigenvector centralities"
## [17] "K-core Decomposition"
## [18] "Geodesic K-Path Centrality"
## [19] "Katz Centrality (Katz Status Index)"
## [20] "Kleinberg's authority centrality scores"
## [21] "Kleinberg's hub centrality scores"
## [22] "clustering coefficient"
## [23] "Lin Centrality"
## [24] "Lobby Index (Centrality)"
## [25] "Markov Centrality"
## [26] "Radiality Centrality"
## [27] "Shortest-Paths Betweenness Centrality"
## [28] "Current-Flow Closeness Centrality"
## [29] "Closeness centrality (Latora)"
## [30] "Communicability Betweenness Centrality"
## [31] "Community Centrality"
```

```
## [32] "Cross-Clique Connectivity"
## [33] "Entropy Centrality"
## [34] "EPC - Edge Percolated Component"
## [35] "Laplacian Centrality"
## [36] "Leverage Centrality"
## [37] "MNC - Maximum Neighborhood Component"
## [38] "Hubbell Index"
## [39] "Semi Local Centrality"
## [40] "Closeness Vitality"
## [41] "Residual Closeness Centrality"
## [42] "Stress Centrality"
## [43] "Load Centrality"
## [44] "Flow Betweenness Centrality"
## [45] "Information Centrality"
## [46] "Dangalchev Closeness Centrality"
## [47] "Group Centrality"
## [48] "Harmonic Centrality"
## [49] "Local Bridging Centrality"
## [50] "Wiener Index Centrality"
```

Now, from this list of about fifty, we can calculate the ones we want (here, I take the first five plus Katz - it has a lot of meaning in interpreting power relationships). Of course, we can turn them into a nice table, and present them to the world. The command is simply *"calculate_centralities"*.

Please remember that this is for the **first component only**. One node, #12, is missing from this component - remember, it was stand-alone, when we graphed it? Therefore, if we want to provide a column with family names, we need to remove node 12 from the list:

```
NewCent<-calculate_centralities(Comp1, include = c(pr_cent[1:5], pr_cent[19]))
NewCent<-as.data.frame(NewCent)
# Remove node 12 from the list of names
# Note how I have to do some acrobatics with data to accomplish this
# Otherwise, R will turn all characters into numbers
node<-c(as.character(flo.att$flo.names[1:11]),as.character(flo.att$flo.names[13:16]))
#Add a column of names to the table
# Make sure names are not converted back to numbers with stringsAsFactors = FALSE command:
NewCent<-data.frame(node,NewCent, stringsAsFactors = FALSE)
names(NewCent)<-c("Family", "Alpha", "Bonacich", "Page Rank", "Average Distance", "BaryCenter","Katz")
kable(NewCent, digits=3, caption = "Florentine Families' Additional Network Centralities")
```

Table 3: Florentine Families' Additional Network Centralities

Family	Alpha	Bonacich	Page Rank	Average Distance	BaryCenter	Katz
ACCIAIUOL	-1	-0.368	0.031	2.375	0.026	1.182
ALBIZZI	-10	-2.022	0.079	1.812	0.034	1.452
BARBADORI	-7	-1.470	0.050	2.000	0.031	1.326
BISCHERI	1	0.000	0.069	2.188	0.029	1.459
CASTELLAN	-6	-1.287	0.069	2.250	0.028	1.436
GINORI	-9	-1.838	0.032	2.625	0.024	1.145
GUADAGNI	0	-0.184	0.098	1.875	0.033	1.555
LAMBERTES	1	0.000	0.031	2.688	0.023	1.156
MEDICI	-2	-0.551	0.146	1.562	0.040	1.823
PAZZI	1	0.000	0.036	3.062	0.020	1.130
PERUZZI	-2	-0.551	0.068	2.375	0.026	1.448
RIDOLFI	8	1.287	0.070	1.750	0.036	1.489

Family	Alpha	Bonacich	Page Rank	Average Distance	BaryCenter	Katz
SALVIATI	0	-0.184	0.061	2.250	0.028	1.295
STROZZI	2	0.184	0.088	2.000	0.031	1.583
TORNABUON	7	1.103	0.071	1.812	0.034	1.487

Seminar assignment 6

1. Take a carefull look at the table with Florentine family centralities. Can you figure out why the name "Medici" is familiar to us, and others - not so much? Which centrality, in your opinion, is the most "responsible"" for this?
2. Interpret correlation matrix of the Florentine family centralities. What do you conclude?
3. From the long list of possible centralities, pick *any* five, plus Bonacich, Page Rank, and Katz, to analyze for Florentine families.
    - Obviously, this should not be random. READ about centralities and find the ones that you would think are appropriate for the Florentine families.
    - Briefly describe the chosen centralities. You will see why I chose Bonacich, Page Rank and Katz for you to learn.
    - Create a nice-looking table of these centralities for Florentine Families.
    - Look at Medici now. Did any of your conclusions (from above) change? What have you learned with these new centralities that was not obvious earlier?

Challenge yourself

For this assignment, we ask you to explore 5-7 additional centralities measures for the Florentine network, summarize them in the table, and describe. What do you observe?

Recommended literature

- Kadushin, C., 2012. Understanding social networks: Theories, concepts, and findings. Oup Usa.
- Robins, G., 2015. Doing social network research: Network-based research design for social scientists. Sage.
- Wasserman, S. and Faust, K., 1994. Social network analysis: Methods and applications.

References

1. Adams, J.S., 1963. Towards an understanding of inequity. The Journal of Abnormal and Social Psychology, 67(5), p.422.
2. Breiger R. and Pattison P. (1986). Cumulated social roles: The duality of persons and their algebras. Social Networks, 8, 215-256.
3. Borgatti, S.P. and Everett, M.G., 1992. Notions of position in social network analysis. Sociological methodology, pp.1-35.
4. Burt, R.S., 1987. Social contagion and innovation: Cohesion versus structural equivalence. American journal of Sociology, 92(6), pp.1287-1335.
5. Burt, R.S., 1992. Structural holes. Harvard university press.

6. Burt, R.S., 2005. Brokerage and closure: An introduction to social capital. Oxford university press.
7. Burt, R.S., Jannotta, J.E. and Mahoney, J.T., 1998. Personality correlates of structural holes. Social Networks, 20(1), pp.63-87.
8. Buskens, V. and Van de Rijt, A., 2008. Dynamics of networks if everyone strives for structural holes. American Journal of Sociology, 114(2), pp.371-407.
9. Freeman, L.C., 1992. The sociological concept of" group": An empirical test of two models. American journal of sociology, 98(1), pp.152-166.
10. Greenberg, J.R. and Greenberg, P.J., 1991. Oedipus and beyond: A clinical theory. Harvard University Press.
11. Haidt, J. and Rodin, J., 1999. Control and efficacy as interdisciplinary bridges. Review of general psychology, 3(4), pp.317-337.
12. Heider, F., 1946. Attitudes and cognitive organization. The Journal of psychology, 21(1), pp.107-112.
13. Homans, George C. "The Human Group. New Brunswick." (1950).
14. Kadushin, C., 2002. The motivational foundation of social networks. Social networks, 24(1), pp.77-91.
15. Kadushin, C. and Jones, D.J., 1992. Social networks and urban neighborhoods in New York City. City & Society, 6(1), pp.58-75.
16. Kalish, Y. and Robins, G., 2006. Psychological predispositions and network structure: The relationship between individual predispositions, structural holes and network closure. Social networks, 28(1), pp.56-84.
17. Kent D. (1978). The rise of the Medici: Faction in Florence, 1426-1434. Oxford: Oxford University Press.
18. Lin, N., 1990. Social resources and social mobility: A structural theory of status attainment. Social mobility and social structure, 3, pp.247-261.
19. Lin, N., 2008. A network theory of social capital. The handbook of social capital, 50(1), p.69.
20. Martin, A.M. and Carey, E., 2009. Person-centred plans: empowering or controlling?. Learning Disability Practice (through 2013), 12(1), p.32.
21. Moody, J., 2001. Race, school integration, and friendship segregation in America. American journal of Sociology, 107(3), pp.679-716.
22. Newman, A.M. and Cooper, J.B., 2010. AutoSOME: a clustering method for identifying gene expression modules without prior knowledge of cluster number. BMC bioinformatics, 11(1), pp.1-15.
23. Podolny, J.M. and Baron, J.N., 1997. Resources and relationships: Social networks and mobility in the workplace. American sociological review, pp.673-693.
24. Robins, G., Pattison, P. and Elliott, P., 2001. Network models for social influence processes. Psychometrika, 66(2), pp.161-189.
25. Sandefur, R.L. and Laumann, E.O., 1998. A paradigm for social capital. Rationality and society, 10(4), pp.481-501.
26. Schwartz, E.L. Dragon (1944). Translated by Y. Machkasov. http://a7sharp9.com/dragon.htm.
27. Weeks, Margaret R., Scott Clair, Stephen P. Borgatti, Kim Radda, and Jean J. Schensul. "Social networks of drug users in high-risk sites: Finding the connections." AIDS and Behavior 6, no. 2 (2002): 193-206.
28. Internet resources for images, information, and other references in the lecture:
    - The map of Moscow rivers: http://www.analytictech.com/mb119/pitts.htm

- The Roman road system network: http://www.fh-augsburg.de/~harsch/Chronologia/Lspost03/Tabula/tab_pe05.html
- The protein molecule: http://www.rcsb.org/
- The real neural network of a roundworm: https://archive.nytimes.com/www.nytimes.com/interactive/2011/06/20/science/brain.html
- The Wine network: https://tinyurl.com/TheWineNetwork
- Internet as a network: http://cheswick.com/ches/map/gallery/index.html;http://cheswick.com/ches/map/gallery/index.html
- The NY Times archive of Dr. J.L.Moreno study https://www.nytimes.com/1933/04/03/archives/emotions-mapped-by-new-geography-charts-seek-to-portray-the.html
- The "Money Network" https://dealbook.nytimes.com/2011/04/07/the-money-network/