Organizational Network Analysis Using R

Dr. Tony Johnson and Dr. Ian McCulloh

August 10, 2016

Executive Summary

This tutorial reviews some basic concepts in Organizational Network Analysis (ONA). It provides sample code in R to calculate some general measures and produce basic visualization. ONA can be defined as the study of complex human systems through the mapping and analysis of social relationships between people, groups, and/or organizations. It can be used by the operations research analyst to reveal the structure, composition, and characteristics of existing networks. With this approach the savvy analyst is prepared to study and refine the communications and information flow of friendly networks in order to build resiliency while disrupting and fragmenting opposing networks in order to build disunity and ultimately collapse. (Bell and Conjar 2016)

Organizational Network Analysis (ONA)-Examples

The purpose of this document is to help operations research analysts understand how to use Organizational Network Analysis (ONA), and what can be achieved by using R as a tool. This tutorial will also make use of a downloadable R package called **statnet**. The package is evoked with the commands

```
install.package(statnet)
library(statnet)
```

This document provides background and an initial approach to using ONA for assessing and making recommendations to improve the effectiveness and performance of an organization. It is not meant to be a detailed technical guide, however, and it is assumed that the reader has some familiarity with community detection within networks. Fortunately, software code to conduct basic network analysis exists in R. The examples provided here are drawn from a simulated data set of over 7000 emails from a private research laboratory working in the cyber security industry.

The example organization consists of a sector of 250 employees with diverse backgrounds ranging from computer science and electrical engineering to business administration and sociology. The employees are divided into four departmental groups or DGs. Each DG has around 60 employees. Thus, a sub-grouping can be drawn from the sector based upon DG affiliation. The subgroup is further organized into sections with a group supervisor and assistant group supervisor overseeing the section supervisors and three group staff members responsible for administration, strategy, and quality control. The roughly 60 other employees are divided among the different sections.

Data Preparation

Email data can be gathered from a Microsoft exchange server and is most useful if it contain columns that are arranged in the following format.

```
pid1    pid2    num_email
105     1010    17
105     1044    11
1037    101     2
```

and

```
pid     dept_group
1010    AAA-001
1011    AAA-002
1047    AAA-005
```

The first file is called "emailedges.csv" and the second file is called "nodes.csv." The nodes file provides an anonymous table of attributes for the nodes listed in the emailedges.csv.

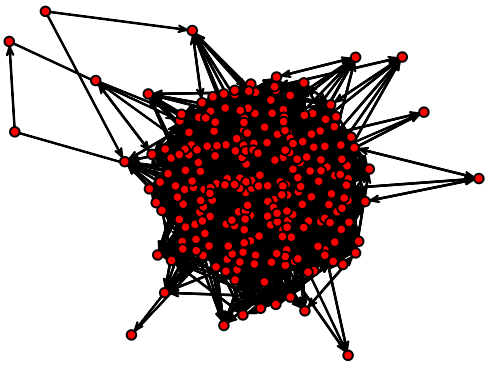
Data Import

While R can read excel **.xls** and **.xlsx** files, sometimes these file types can cause problems because of hidden codes embedded in them. Comma separated files (.csv) are much easier to import. It is best to save these files as **.csv** files before reading them into R. The best way to do this is with the R command `read.csv`. The command allows us read our comma separated nodes and edges into two separate R object known as data frames. A data frame is an R object used to store list of vectors of equal length. Here in our example we will use the `read.csv` command and the **sep** and **header** options. There are many other options that can be used as well. See the official R-manual page on `read.csv` to learn more: http://cran.r-project.org/doc/manuals/R-data.html.

```
## Load data frames data.a and data.b
data.a <- read.csv("ONA-Data/nodes.csv",sep=",", header=T)
data.b <- read.csv("ONA-Data/emailedges.csv",sep=",", header=T)
```

The Network

Let us now examine the entire network of email traffic between the 250 employees. It is difficult to derive any useful information about the network or its email traffic through visual inspection. R can plot the network as seen in the graph below, but further analysis is needed in order to gain insight about the organization and its characteristics.



ONA Techniques, Tactics and Procedures

The first step in the ONA process is to extract a network of interest for analysis. This would most likely consist of all the employees within a particular group. This subgroup of the sector's 250 employees would be roughly 60 employees. The key now is to select employees based on their DG affiliation from the full data set which includes all individuals and their departmental groups. Here is where the R function `grep` is helpful. The command `grep` searches for matches to patterns within each element of a data frame. In this case the data frame is a character vector containing the DG affiliation of each employee. We can now extract a subgroup.

```
## Parse out subgroups of interest for analysis by using the grep command and
## keying on a node attribute that, in this case, identifies the department.
subgroup1<-data.a[grep("AAA-001", data.a[,2]),1]
subgroup2<-data.a[grep("AAA-002", data.a[,2]),1]
```

We now write a custom function to receive as input a **subgroup** of employees and the **edges** from the full network, which represent the sector's email traffic. The function then creates a dichotomous network containing only the vertices of the given subgroup. Further refinement of the number of edges in the smaller sub-network are established by the **cutoff_value**, which establishes at a certain threshold for number of emails that constitute and edge in the network. We call our custom function extract_subnet.

```
extract_subnet<-function(subgroup, edges, cutoff_value){
## This function takes a subgroup of nodes from a larger network
## and creates a dichotomous network consisting only of nodes from
## within the group at a certain edge threshold
## INPUT:
##     subgroup : list of nodes within the subgroup
##        edges : dataframe consisting of all edges in the network
##                containing at least three columns
##                source, target, and edge weight
## cutoff_value : scalar value for threshold cutoff point
##
## OUTPUT:
##            g : network object containing all nodes of subgroup
##                and edges only where both the source and the
##                target are from the subgroup
##

df.edges <- edges # Assign all edges to a dataframe

# Use %in% to search and select only nodes within the subgroup
df.edges[,4] <- df.edges$pid1 %in% subgroup # check for source nodes in subgroup
df.edges[,5] <- df.edges$pid2 %in% subgroup # check for target nodes in subgroup

# Reduce is a Common Higher-Order Function in R located in the
# Functional Programming Language. It uses a binary function to successively
# combine the elements of a given vector. It allows us to filter based on a
# given criteria without the use of loops which should be avoided if possible.
# Here, if the source is TRUE and the target is TRUE meaning they are in
# the subgroup then set df.edges col 6 to resolves to TRUE
df.edges[,6] <- Reduce('&', df.edges)

# Selecting rows that meet certain criteria is a lot like SQL.  Here also
# we can disgard the logical columns as we now have edges only where both
# the source and target are from the subgroup
df.edges <- df.edges[df.edges$V6,][,1:3]
# Use same convention to select only those rows which meet the cutoff_value
df.edges <- df.edges[df.edges$num_emails >= cutoff_value,]

# We can easily count the number of edges left using the nrow function.
#  num_edges <- nrow(df.edges)

# Build the network object g
```

```
node_names<-as.character(subgroup)
num_nodes<-as.numeric(length(subgroup))
# Initialize the object
g<-network.initialize(num_nodes,directed = TRUE)
network.vertex.names(g)<-node_names # Assign the node names
g[,] <- 0
g[df.edges[,1:2]] <- 1 # Attach the edges

return(g)
##
## end function extract_subnet
}
```

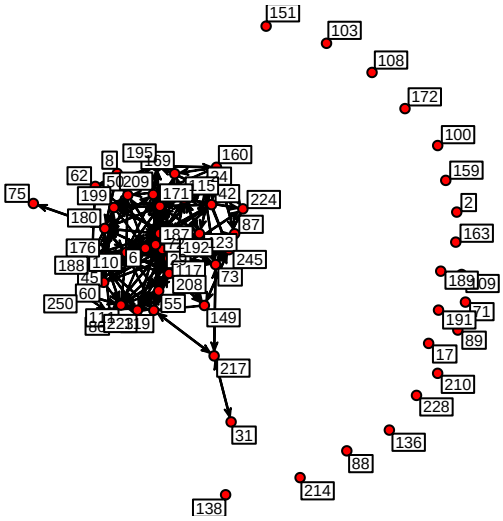
The function **extract_subnet** extracts a group network from the larger network. The cutoff threshold is a value that specifies the number of emails that must be sent between individuals to qualify as a relationship in the network. We know that the number of emails between individuals is more indicative of their personal email behavior rather than the strength of a relationship. The goal, then, of using emails for network data is to identify the likely presence of regular communication. Setting a threshold to reduce data complexity to a level where network structure can be observed provides an estimate of organizational collaboration. Below are three such views. The naming convention is designed to include the **cutoff_value**.

```
subnet.5 <- extract_subnet(subgroup2, data.b, 5)
subnet.11 <- extract_subnet(subgroup2, data.b, 11)
subnet.18 <- extract_subnet(subgroup2, data.b, 18)
```

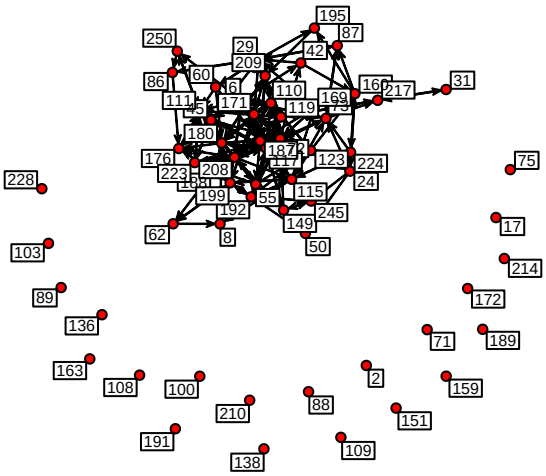
Basic Visual Analysis

Now that we have a way to extract a subgroup nodes along with the corresponding edges between them, we can visualized the sub-network in R by using the `plot` command. Note that one of the final tasks accomplished by our custom function **extract_subnet** was to create a network object **g** which defines the resulting network in terms that the `plot` command understands. This preconditioning makes visualizing the network a simple one line command. The plots shown below display various views of the network. Note that preferences to display labels or place boxes around labels are controlled by `displaylabels=TRUE, boxed.labels=TRUE`. You can also adjust the size of the labels by adding the `label.cex` option. Other visualization options will be discussed later. For a detailed review of the `plot` command and its related options see https://cran.r-project.org.

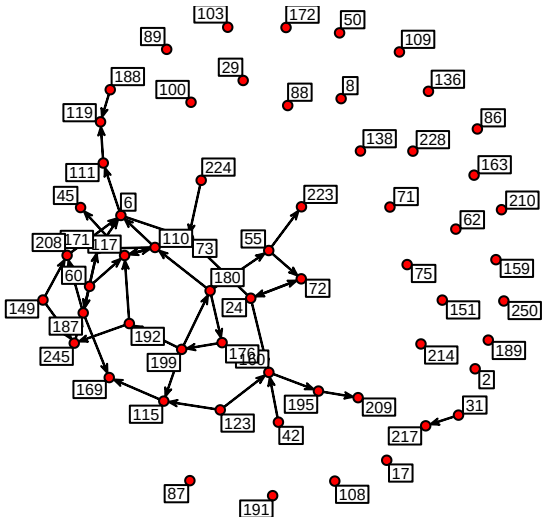
```
plot(subnet.5, displaylabels=TRUE, boxed.labels=TRUE, label.cex=0.5)
```



```
plot(subnet.11, displaylabels=TRUE, boxed.labels=TRUE, label.cex=0.5)
```



```
plot(subnet.18, displaylabels=TRUE, boxed.labels=TRUE, label.cex=0.5)
```



Analysis

The visualization of the nodes on the network immediately provide some interesting insights without any additional computations. If we choose to examine, say, ***subnet.18***, we can get a summary of its characteristic using the `summary.network` command.

```
summary.network(subnet.18, print.adj = FALSE)
```

```
## Network attributes:
##   vertices = 60
##   directed = TRUE
##   hyper = FALSE
##   loops = FALSE
##   multiple = FALSE
##   bipartite = FALSE
##  total edges = 44
##    missing edges = 0
##    non-missing edges = 44
##  density = 0.01242938
##
## Vertex attributes:
##   vertex.names:
##    character valued attribute
##    60 valid vertex names
##
## No edge attributes
```

We can determine the number of isolated nodes or "isolates" in the ***subnet.18*** graph by counting the number of nodes that have a degree centrality of 0.

```
sum(degree(subnet.5)==0)
```

```
## [1] 20
```

```
sum(degree(subnet.11)==0)
```

```
## [1] 21
```

```
sum(degree(subnet.18)==0)
```

```
## [1] 28
```

Nearly half of the nodes ($28/60 = 46.6\%$) are isolates, meaning that they have no interaction over email with any of the other nodes in the group for the given threshold. These nodes may work offsite in direct support of clients and may not interact with others in their own group. If they do interact with each other, it may be done through non-organizational email addresses. Further investigation reveals that many people in connected clusters have no idea of others working for the same employer. This may be a problem for an organization that desires to be pioneering and share knowledge and resources. Lessons learned in support of one client have limited ability to transfer and benefit other clients. Data complexity (the network is too dense) is not an issue in this organization. This is one example of how ONA with R can be used to quickly analyze organizational data to provide insight into the underlying network. Further analysis will be focused on the full, non-reduced network.

Organizational Structure Analysis

The next example and set of diagnostics apply to the network as it relates to its structure. The ideal structure from which to test an organization is the graph theoretical term *outtree*. (Krackhardt 1994) An *outtree* is a directed graph where every point except the point at the very "top", has one and only one edge pointing to it. If these edges represent, say, authority relationships, then one might see how the graph theoretical *outtree* is more like an *archetypical formal hierarchy* (Simon 1981) or commonly known as an organization chart.

Measures of connectedness, hierarchy, efficiency, and least upper boundedness (LUB) can be calculated using the following commands.

```
group.con <- connectedness(subnet.18)
group.hier <- hierarchy(subnet.18)
group.eff <- efficiency(subnet.18)
group.lub <- lubness(subnet.18)
```

The results of these measures are 0.246, .021, 0.983, 0.527 respectively. Each of these four calculated measures gives an analysis of the structure of the organization based on the number of violations that exist in any particular structural arrangement. **Connectedness**. A violation of connectedness happens when a vertex or node is unable to reach another node within the underlying network. The more people are separated from each other, the more difficult it is for them to organize. If the task facing an organization is routine and require very little flexibility, then this measure may not be essential. However, if the task at hand involves exceptions to the routine standard operating procedures requiring consultation and collaboration, then a lack of connectedness could signal major political divisions that would impede the organizations' ability to adapt. **Hierachy**. A violation of hierarchy exists whenever a symmetric ties are presented in the *reachability* directed network derived from the group. This condition is most prevalent in organizations where status, prestige, and formal authority are prominent. In a formal organization, a high-level employee can reach a subordinate's subordinate, but the opposite is not true as the lower level employee cannot reach his "supervisor's supervisor". This type of *mechanistic* organization will most likely be very formal and status ridden (Shrader, Lincoln, and Hoffman 1989). **Efficiency**. One of the conditions of the *outree* is that there are exactly (N-1) edges or links in the network. Fewer than (N-1) edges and the network would fragment into components. More than (N-1) edges and redundant paths would emerge with multiple cycles between nodes. Thus, a violation of efficiency occurs whenever more links are present than are necessary to keep the network from breaking into components. Since links are not without a social cost to maintain, efficiency is a characterization of how dense the network is beyond that which is absolutely necessary to keep the social group connected to each other. Keep in mind, however, that an organization that is trimmed and tailored to the point that its network efficiency is maximum, also run the risk of being vulnerable to fragmentation because of arbitrary link deletions. On the other hand, low efficiency dense networks are likely to be overburdened with trying to network. This requires them to spend time interacting as opposed to actually doing work. Finally, we have **Least Upper Boundedness** or **LUB**. For a pair of nodes in the network to have a *least upper bound* (LUB) score they must each have access to a common third person in the network organization to whom they can both appeal. In other words, we are looking for the closest boss of two employees who has formal authority to which they can appeal or defer. Violations of LUB occur when employees have multiple supervisors in the network. This is the most complex of the four structural measurements. In some, the LUB condition preserves the concept of *unity-of-command* by ensuring that there is only one *commander-in-chief* or *chief executive* at the top of the organization. High LUB would infer that conflict resolution happens quickly as opposed to organizations that lack high LUB (Doreian 1971).

Recall that an *outtree* is a purely hierarchical structure characterized by having 1s across all four measures. It is important to resists the temptation to combine all measures into a single indicator. Each measure provides a unique, yet comparable insight into the organization. The connectedness is low, because there are so many isolates. The hierarchy is also low, because there are many observed reciprocal ties. The efficiency is high, because there are very few lateral peer-to- peer ties, which may be due, in part, to the high number of isolates as well. The least upper boundedness is mid-range due to an ambiguity in unity of command.

Graph Level Diagnostics

The next set of metrics focus on network (graph) level properties. Note that `library(statnet)` does not have an included function to calculate diameter. The following code provides a function that can be used to calculate the diameter.

```
get_diameter <- function(x){ #measures the diameter of a network x
  g <- geodist(x)
  g1 <- g$gdist
  g1[g1 == (Inf)] = 0
  d <- max(g1)
  return(d)
}
```

There is also no included function for calculating the clustering coefficient. However, the global clustering coefficient can be calculated using the following code,

```
clustcoef<-function(x){ #measure clustering coefficient of network x
  t<-triad.census(x)
  cltriad<-sum(t[9],t[10],t[12],t[13],t[14],t[15],t[16])
  optriad<-sum(t[4],t[5],t[6],t[7],t[8],t[11])
  clustcoef<-cltriad/(cltriad+optriad)
  clustcoef
}
```

Using the above functions, the diameter and clustering coefficient of the network can be calculated using the following commands. Additionally, the density of the network is calculated with the statnet function `gden()`.

```
gn <- subnet.18
get_diameter(gn)
clustcoef(gn)
gden(gn)
```

The results of these measures are 8.000, 0.023, 0.012 respectively. **Diameter** indicates the ability of information to quickly move from one side of the network to the other. The number of isolate nodes can effect the diameter. Also, the diameter measure does not scale well with network size. Given that an individual's awareness of resources and knowledge across the network decays somewhere between 2-3 steps through the network, a diameter greater than or equal to 6 will indicate limits to knowledge and resource exchange. Even if the diameter is within a comfortable range, an organization will be challenged when there exists a large number of isolates.

Clustering coefficient is similar to density in that it indicates the tendency for nodes to cluster around each other. A higher relative network clustering coefficient compared to its density can indicate the presence of large clusters in the network. In other words, the nodes that are not isolates tend to cluster together. Too high of a clustering coefficient indicates a trend towards a fully connected *clique*. This leads toward group-think and lacks the diversity for an agile organization. A large clustering coefficient is also a poor quality of an efficient organization. This is because of a high number of lateral peer-to-peer ties which take time and resource away from purely efficient organizational structures. A low clustering coefficient is good for lean organizations, but poor for agile ones. A moderate clustering coefficient is advisable for agile organizations striving for innovation. The scale of "moderate" is dependent, however, on the context and size of the network.

Density takes into account the number of isolates present. The density in this network is very low, which is the result of the high number of isolates. Density also scales poorly with network size. It also has the

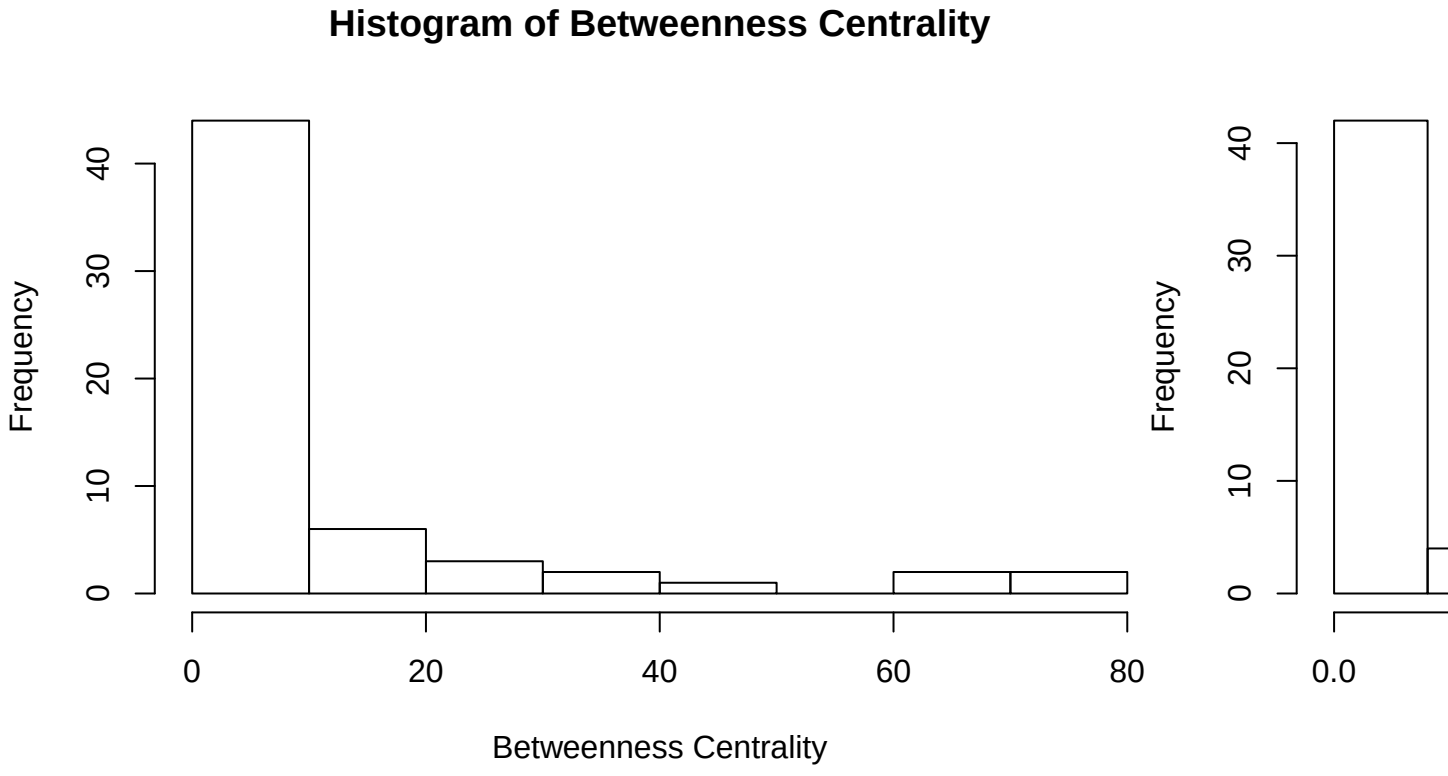
mathematical equivalence to average degree. **Average degree** is equal to the density times the network size. Networks with high average degree can become problematic in that nodes are too busy maintaining network ties to have meaningful interactions or generate useful ideas. Networks with low average degree indicate missed opportunities for knowledge and resource exchange. The range of "good" average degree is dependent upon the context of the network relation and the nature of interaction.

Agent Level Centrality Diagnostics

Centrality measures are important measures of influence and opinion leadership within an organization. The two most important agent-level centrality measures are **betweenness centrality** and **eigenvector centrality**. They can be calculated using the following commands.

```
b<-betweenness(gn)
e<-evcent(gn)
```

Because these are *agent-level* measures, the values are calculated for each node in the network. Therefore, it is impractical to list individual values. However, the distribution of these values can be represented with a histogram.



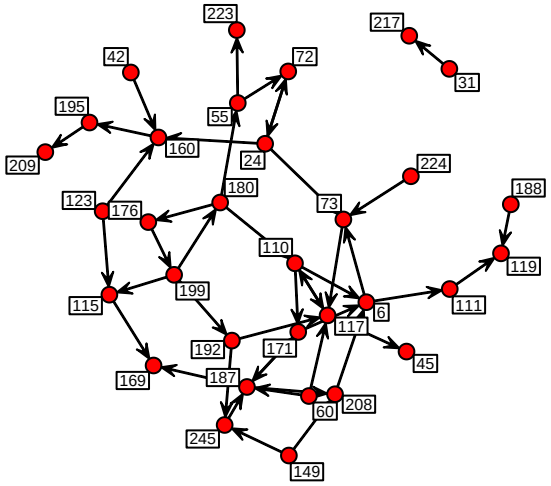
Betweenness centrality provides an ideal measure for informal power and brokerage in relatively sparse networks like the one in this example. When the data complexity becomes high, the network is dense. Since there are many connections, there are less opportunities for nodes to occupy purely brokerage positions, even though they may fulfill this role often in the organization. In these situations, eigenvector centrality may provide a better measure of this influence. This is because it can be interpreted as the frequency in which a node is encountered during a random walk through the network. An unfortunate property of eigenvector centrality, however, is that it tends to only identify brokers in a single network cluster. When there exists multiple clusters, central power brokers in other clusters may not be properly valued.

In order to analyze the network without considering isolates, it is necessary to remove them. This is conducted in two steps: identify the isolates, then remove them. This is completed using the following commands.

```
isol<-isolates(gn) #identify all isolates in the network
small.group<-delete.vertices(gn, isolates(gn)) #rm isol
```

The resulting network can be plotted and observed.

```
gplot(small.group, displaylabels = T, boxed.labels=TRUE, label.cex=0.5, main="No Isolates")
```



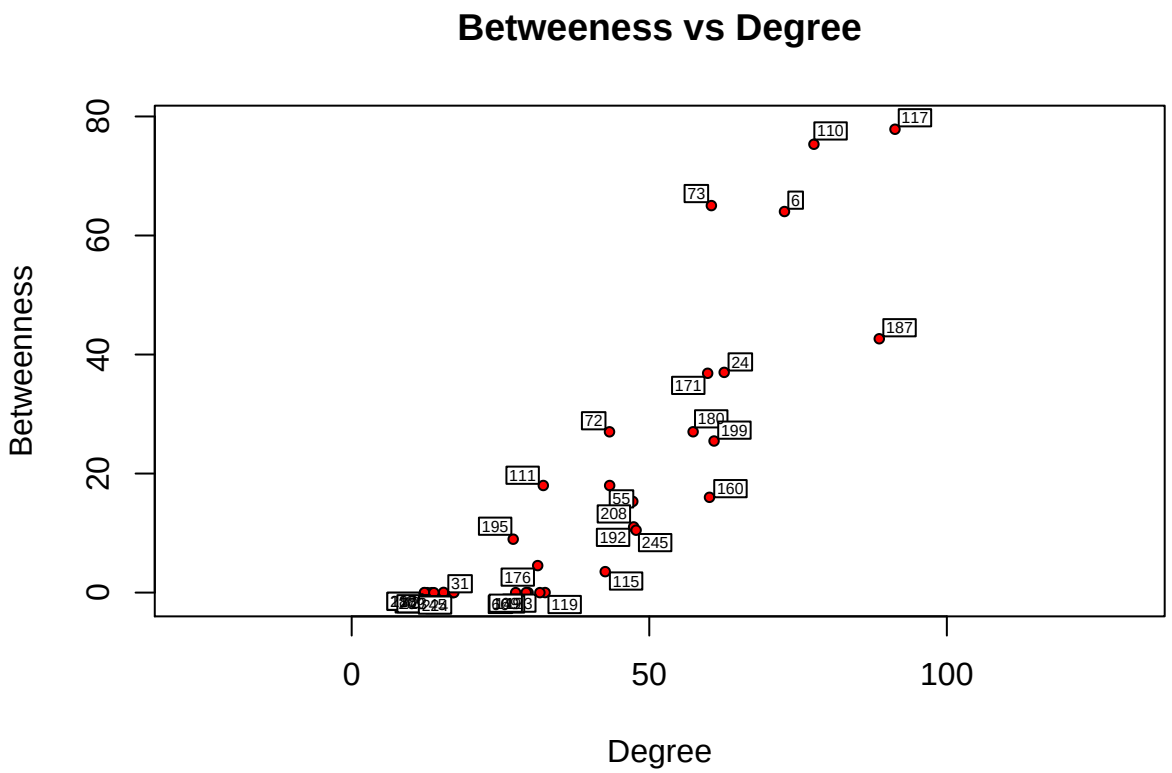
Degree centrality is not particularly interesting on its own. The number of connections that a node has does not really identify its importance apart from the fact that degree centrality is often mathematically correlated with other centrality measures that are often more important. When the measures are correlated, they often conform to intuition. It is when measures are uncorrelated, that ONA can provide interesting, counter-intuitive findings. For example, a node that is high in betweenness centrality, yet low in degree is a power broker or gate keeper. This example does not provide an example of this type of power broker, however an example of how to produce a plot for this type of analysis is provided below. Since this type of layout is not provided in R, a custom layout must be developed. The following code demonstrates how.

```
network.layout.degbet <- function(d, layout.par){
  dd <- degree(d)
  bt <- betweenness(d)
  out <- cbind(dd*15,bt)
  return(out)
}
```

The custom defined layout can be used with the plot command in the following manner.

```
plot(gn,
     mode="degbet",
     displaylabels=TRUE,
     boxed.labels=TRUE,
     label.cex=0.5,
     xlab="Degree",
     ylab="Betweenness",
     suppress.axes = FALSE,
```

```
    edge.col=0,
    main='Betweeness vs Degree'
)
```



Nodes occupying positions in the upper left portion of the plot are of more interest from an informal power broker perspective. For denser networks, eigenvector centrality should be substituted for betweenness centrality. This network shows little by way of informal power brokers.

Visualization

Visualization options are important tools not only for analysis, but for communicating results in sufficient bandwidth to convey rapid understanding to those who must make decisions based on the information. There are seven dimensions of visualization that can be used to convey information: horizontal position, vertical position, size, color, shape, intensity, and texture. Some dimensions are better than others for communicating different types of information. For example, size is more useful for communicating numeric data, while shape is often better for categorical data. It is not cognitively possible to communicate more than seven dimensions of data, without resorting to multiple visualizations or displays of the data. For network analysis, we recommend the following conventions:

Horizontal and vertical position

Horizontal and vertical position is used to maximize the layout of the network. There are multiple layout algorithms in R. A detailed overview of each is beyond the scope of this paper. However, more information about R network layout algorithms can be found at http://kateto.net/network-visualization. ####Size Size is used to represent a numeric node value, such as betweenness centrality, seniority, education level, or some other hypothesized measure of importance. When using centrality as the basis for size, central nodes can sometimes be displayed so large, that the network structure cannot be adequately observed. In this case it is recommended to force a logarithmic scaling to the centrality measures to convey proper understanding. An example scaling function is provided as,

```
node.scale <- function(x1) { #scales cent values for visualization
  x<-log(x1)
  x[x==(-Inf)]=0
  x[x<0.5]=0.5
  return(x)
}
```

Color

Color is used to represent group/cluster affiliation, or some attitudinal variable such as workplace satisfaction. Colors are defined in R as follows,

```
#1=black
#2=red
#3=green
#4=blue
#5=light blue/aqua
#6=pink
#7=yellow
#8=gray
```

Shape

Shape is used to represent a categorical node value, such as gender, job role, location.

Intensity

Intensity is used to focus attention at key network terrain and/or for over time visualization.

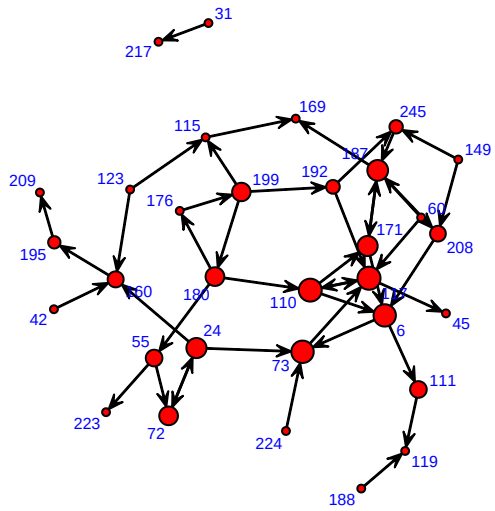
Texture

We should refrain from using texture unless another dimension of data is absolutely necessary. The natural texture of the network itself can often obscure the meaning communicated by other visualization variables.

The figure below provides an example visualization of the **subnet.18** network, where the isolate nodes are removed. The label font size is defined and the color is specified as blue (#4). The color of the nodes are red (#2) and the links are black (#1). The nodes are sized by betweenness, but double scaled using the previously defined function. The layout uses a spring-embedded or force-directed layout algorithm. The supporting command line code is as follows.

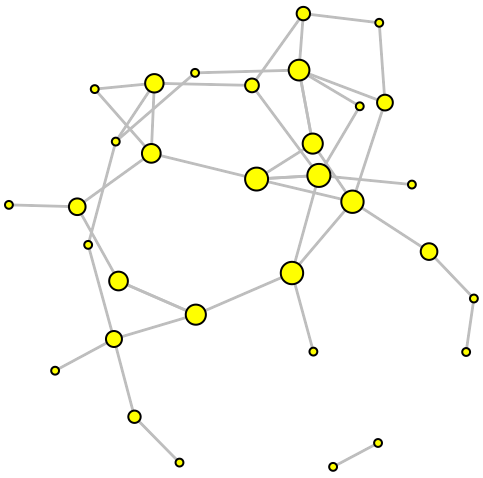
```
gplot(gn,
      label.cex=0.5,
      label.col=4,
      label=network.vertex.names(gn),
      vertex.col=2,
      edge.col=1,
      vertex.cex=node.scale(node.scale(betweenness(gn))),
      main="Size Vertices by Betweenness"
)
```

Size Vertices by Betweenness



Another visualization is to treat the network as if the network were undirected. For this interpretation, the assumption is that if there exists an email from *actor i* to *actor j*, then a relationship is assumed to exist between both *i* and *j*, as well as *j* and *i*. In other words, there is simply a relationship between them and direction does not matter. The figure below is an example of this visualization with supporting code.

```
symmetrize(gn, rule="weak")
gplot(gn,
      usearrows=FALSE,
      vertex.col=7,
      edge.col=8,
      vertex.cex=node.scale(node.scale(betweenness(gn)))
)
```



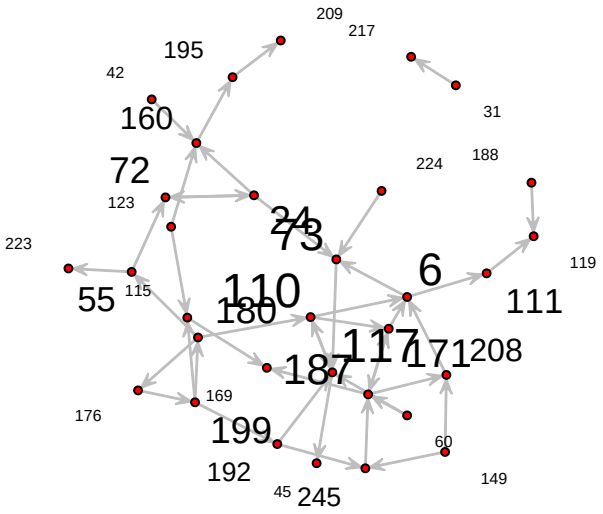
In the above visualization, the labels are omitted, arrows are hidden, and the node and edge colors are modified. These visualizations may lack impact to the reader at this point. This is due to the fact that the reader will have no context by which to understand the organization and the interpersonal dynamics within it. When node labels identify key power brokers in the organization, they often confirm intuition and drive inquiry into areas of the organization that are more effective or problematic. This information will cause changes in behavior among those actors that are exposed to the labeled network diagrams. Some change

may be positive, but others may not. Some actors may experience diminished performance, feeling on the periphery of the network, even if they provide an important function for the organization. For these reasons, it is important to be careful about displaying node labels. Visualization options that hide node labels are important for facilitating discussions about the organization without adversely affecting node behavior.

The network below is an alternate approach to visualization, where the labels are sized based on their centrality value and the edges are gray to provide contrast. Code for this implementation is as follows.

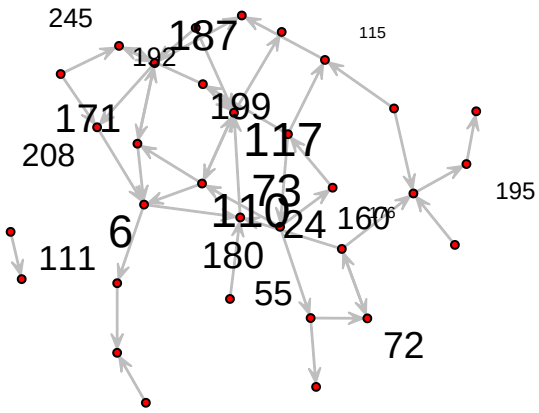
```
gplot(gn,
      label.cex=node.scale(node.scale(betweenness(gn))),
      label.col=1,
      label=network.vertex.names(gn),
      vertex.col=2,
      edge.col=8,
      vertex.cex=0.5,
      main="Size Vertices Labels by Betweenness"
)
```

Size Vertices Labels by Betweenness



It is possible to create a hybrid visualization, where the label values are set to a user defined threshold based on the centrality value of the node. The next figure displays the previous network, however, nodes with a centrality value less than 0.5 are set to display no label. The modified code for this is as follows.

```
bet <- betweenness(gn)
gplot(gn,
      label.cex=node.scale(node.scale(bet)),
      label.col=1,
      label=ifelse(bet>0.5,network.vertex.names(gn),""),
      vertex.col=2,
      edge.col=8,
      vertex.cex=0.5
)
```



Notice that for this implementation, the betweenness centrality scores are stored in the vector, `bet`, for additional manipulation. The logical ifelse command is used to manipulate the threshold. Using this approach, there are many variants that can be implemented for visualization based on the principles described earlier.

Conclusion

In this tutorial, we have reviewed some basic concepts in Organizational Network Analysis (ONA). Sample code in R was provided to get you started on calculating some general measures and produce basic visualization. This tutorial is by no means comprehensive as there are a plethora of measures and analysis that can be done on an organization. With some basic knowledge and practice, much can be derived about an organization using tools freely available in the open source software R.

References

Bell, Christopher P., and Elizabeth Conjar. 2016. https://www.boozallen.com/content/dam/boozallen/documents/01.005.14_NetworkAnalysis_WP-webonly.pdf.

Doreian, Patrick. 1971. *Mathematics and the Study of Social Relations.* Schocken Books.

Krackhardt, David. 1994. "Graph Theoretical Dimensions of Informal Organizations." In *Computational Organization Theory*, edited by Kathleen Carley and Michael Prietula, 89–111. Lawrence Erlbaum Associates Inc.

Shrader, Charles, James Lincoln, and Alan Hoffman. 1989. "The Network Structures of Organizations: Effects of Task Contingencies and Distributional Form." *Human Relations* 42 (1): 43–66. doi:10.1177/001872678904200103.

Simon, H. A. 1981. *The Science of the Artificial.* 2nd ed. MIT Press, Cambridge MA.