Advanced Network Analysis

Centrality

Olga Chyzh [www.olgachyzh.com]

Centrality

- Centrality measures help understand:
  - **"Which node is the most important or central in this network?"**
  - What do you mean by "important"?
  - What do you mean by "center"?
  - Definition of "center" varies by context/purpose
  - The power a person holds in an organization may be inversely proportional to the number of keys on their keyring
    - A janitor has keys to every office, and no power
    - The CEO does not need a key: people always open the door for her

Centrality

Freeman (1979):

"There is certainly no unanimity on exactly what centrality is or on its conceptual foundations, and there is little agreement on the proper procedure for its measurement."

Working example

Florentine marriages (Padgett & Ansell 1993)

```
# load datasets
data(flo) #this dataset is available from the -network- package
# view
dim(flo)
```

```
## [1] 16 16
```

```
rownames(flo)
```

```
##  [1] "Acciaiuoli"   "Albizzi"      "Barbadori"    "Bischeri"     "Castella
##  [6] "Ginori"       "Guadagni"     "Lamberteschi" "Medici"       "Pazzi"
## [11] "Peruzzi"      "Pucci"        "Ridolfi"      "Salviati"     "Strozzi"
## [16] "Tornabuoni"
```

```
flo[1:5,1:5]
```

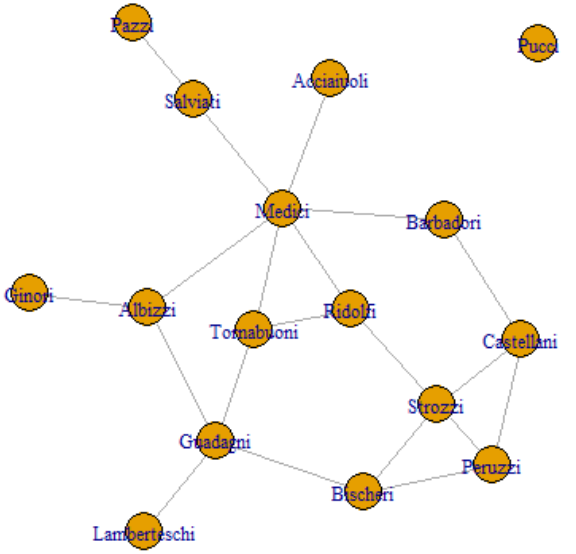
```
##             Acciaiuoli Albizzi Barbadori Bischeri Castellani
## Acciaiuoli          0       0          0        0           0
## Albizzi             0       0          0        0           0
```

Florentine families

```
# convert to igraph object
library(igraph)
g = graph_from_adjacency_matrix(flo, mode='undirected', diag=FALSE)
plot(g)
```

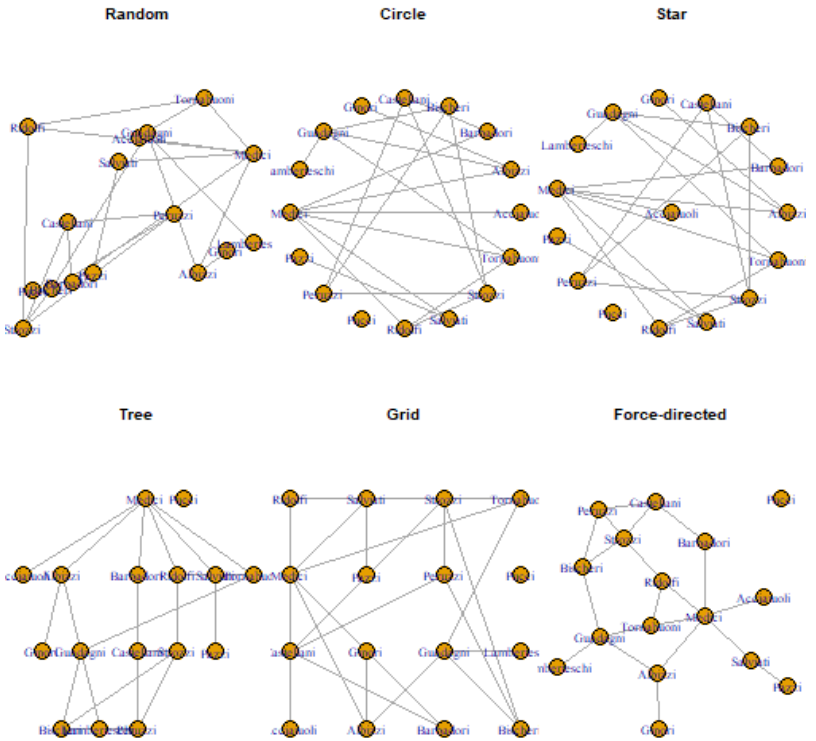
Florentine families

Who looks central?



Florentine families

Does simply plotting with different layouts help?



Popular Measures of Centrality

Well ... let's define centrality:

- Degree
- Closeness
- Betweenness
- Eigenvector

Degree centrality

- **Idea**: The nodes with more connections to others are more central
- How to measure:
  - Undirected degree centrality: $\sum_{j:j\neq i} y_{i,j}$
  - Directed outdegree centrality: $\sum_{j:j\neq i} y_{j,i}$
  - Directed indegree centrality: $\sum_{j:j\neq i} y_{i,j}$
- Though simple, degree is often a highly effective measure of the influence or importance of a node
  - In many situations, people with more connections tend to have more power
- Examples where this might be useful?

Degree centrality

Lets calculate degree centrality in R (using the `igraph` package)

```
##     Medici   Guadagni    Strozzi    Albizzi   Bischeri Castellani
##          6          4          4          3          3          3
```

Degree centrality

Lets calculate degree centrality in R using `igraph`

```
?degree
```

Description

The degree of a vertex is its most basic structural property, the number of its adjacent edges.

Usage

```
degree(graph, v = V(graph), mode = c("all", "out", "in", "total"),
  loops = TRUE, normalized = FALSE)

degree_distribution(graph, cumulative = FALSE, ...)
```

Arguments

`graph`	The graph to analyze.
`v`	The ids of vertices of which the degree will be calculated.
`mode`	Character string, “out” for out-degree, “in” for in-degree or “total” for the sum of the two. For undirected graphs this argument is ignored. “all” is a synonym of “total”.
`loops`	Logical; whether the loop edges are also counted.
`normalized`	Logical scalar, whether to normalize the degree. If `TRUE` then the result is divided by *n-1*, where *n* is the number of vertices in the graph.
`cumulative`	Logical; whether the cumulative degree distribution is to be calculated.
`...`	Additional arguments to pass to `degree`, eg. `mode` is useful but also `v` and `loops` make sense.

Degree centrality

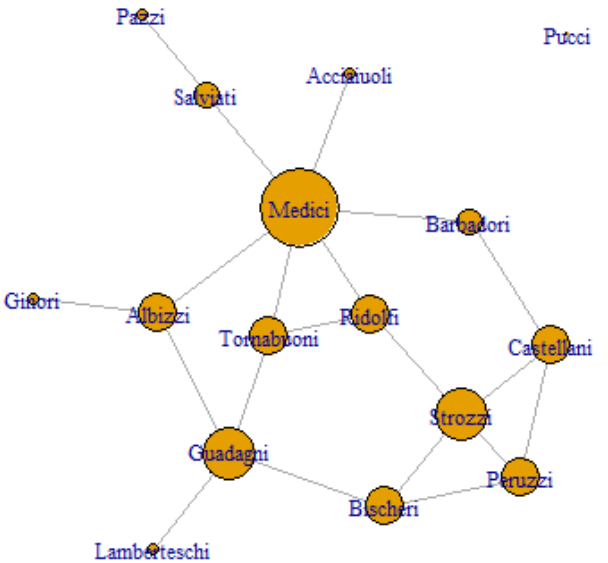
Lets calculate degree centrality in R using `igraph`

```
cDegree = degree(g, mode='in', loops=FALSE)
sort(cDegree, decreasing=TRUE)[1:6]
```

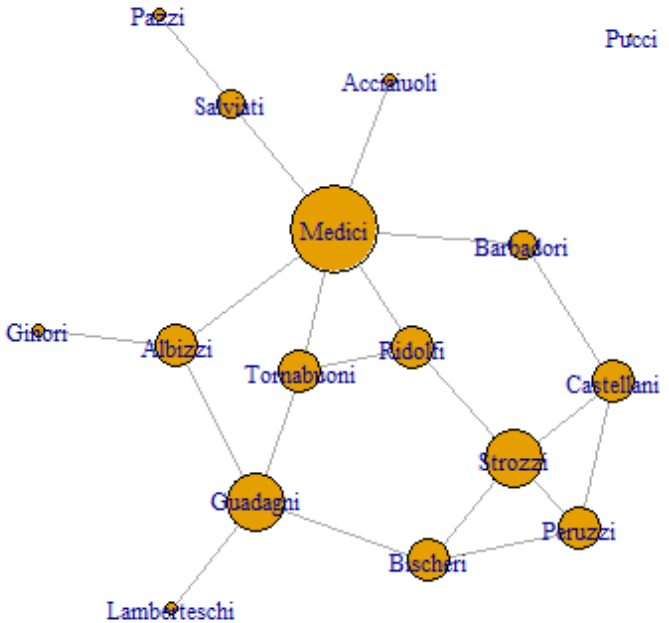
```
##     Medici   Guadagni    Strozzi    Albizzi   Bischeri Castellani
##          6          4          4          3          3          3
```

Now ...

Adjust the size of nodes in the Florentine graph by degree



```
# convert to igraph object
g = graph_from_adjacency_matrix(flo, mode='undirected', diag=FALSE)
set.seed(6886)
plot(g, vertex.size=5*cDegree)
```



Closeness centrality

- **Idea**: If a node is far away from all other nodes, then it should be less central ... or to put it another way, the more central a node, the lower its total distance to all other nodes
- How to measure:
  - (geodesic) distance: $d_{i,j}$ is the minimal path length from i to j
  - closeness centrality: $\frac{1}{\sum_{j:j\neq i} d_{i,j}}$
- Closeness can also be regarded as a measure of how long it will take to spread information from a node to all other nodes sequentially
- This measure won't be useful for disconnected graphs ... why?
- Examples where this might be useful?

Closeness centrality

Lets think about how to calculate this ...

- Start with getting distances between nodes

```
d = distances(g)
d[1:5,1:5]
```

```
##             Acciaiuoli Albizzi Barbadori Bischeri Castellani
## Acciaiuoli          0       2          2        4           3
## Albizzi             2       0          2        2           3
## Barbadori           2       2          0        3           1
## Bischeri            4       2          3        0           2
## Castellani          3       3          1        2           0
```

Closeness centrality

Why? is this

```
d['Pucci',1:5]
```

```
## Acciaiuoli    Albizzi  Barbadori   Bischeri Castellani
##        Inf         Inf         Inf        Inf         Inf
```

Lets fix.

```
d[d==Inf] = NA
```

Now lets get a pseudo-measure of closeness

```
avgPathDistance = sort(apply(d, 1, mean, na.rm=TRUE), decreasing=FALS

#dump pucci
avgPathDistance = avgPathDistance[-1]

# top 6
avgPathDistance[1:6]
```

```
##     Medici    Ridolfi    Albizzi Tornabuoni   Guadagni  Barbadori
##   1.666667   1.866667   1.933333   1.933333   2.000000   2.133333
```

Closeness centrality

Lets calculate this in `igraph`

```
?closeness
```

Description

Cloness centrality measures how many steps is required to access every other vertex from a given vertex.

Usage

```
closeness(graph, vids = V(graph), mode = c("out", "in", "all", "total"),
  weights = NULL, normalized = FALSE)

estimate_closeness(graph, vids = V(graph), mode = c("out", "in", "all",
  "total"), cutoff, weights = NULL, normalized = FALSE)
```

Arguments

`graph`	The graph to analyze.
`vids`	The vertices for which closeness will be calculated.
`mode`	Character string, defined the types of the paths used for measuring the distance in directed graphs. “in” measures the paths *to* a vertex, “out” measures paths *from* a vertex, *all* uses undirected paths. This argument is ignored for undirected graphs.
`weights`	Optional positive weight vector for calculating weighted closeness. If the graph has a `weight` edge attribute, then this is used by default. Weights are used for calculating weighted shortest paths, so they are interpreted as distances.
`normalized`	Logical scalar, whether to calculate the normalized closeness. Normalization is performed by multiplying the raw closeness by *n-1*, where *n* is the number of vertices in the graph.
`cutoff`	The maximum path length to consider when calculating the betweenness. If zero or negative then there is no such limit.

Closeness centrality

Lets calculate this in `igraph`

```
sort(closeness(g), decreasing=TRUE)[1:6]
```

```
##     Medici    Ridolfi    Albizzi Tornabuoni   Guadagni  Barbadori
## 0.02439024 0.02272727 0.02222222 0.02222222 0.02173913 0.02083333
```

Compare with our hack-y calculation:

```
avgPathDistance[1:6]
```

```
##     Medici    Ridolfi    Albizzi Tornabuoni   Guadagni  Barbadori
##   1.666667   1.866667   1.933333   1.933333   2.000000   2.133333
```

Degree vs Closeness

```
sort(degree(g), decreasing=TRUE)[1:6]
```

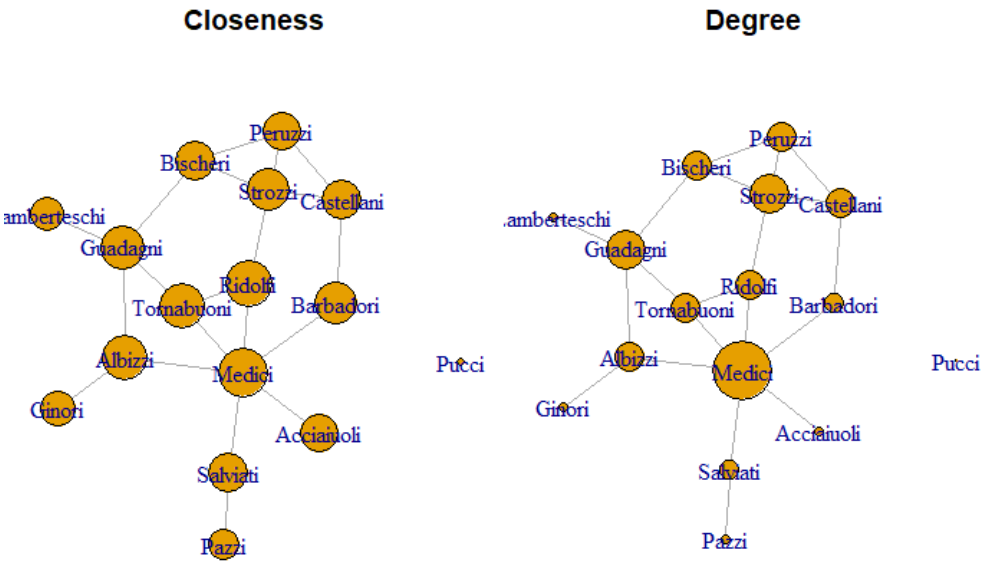
```
##     Medici   Guadagni    Strozzi    Albizzi   Bischeri Castellani
##          6          4          4          3          3          3
```

```
sort(closeness(g), decreasing=TRUE)[1:6]
```

```
##     Medici    Ridolfi    Albizzi Tornabuoni   Guadagni  Barbadori
## 0.02439024 0.02272727 0.02222222 0.02222222 0.02173913 0.02083333
```

Two-way Visualization Comparison

Now plot adjusting node sizes by closeness centrality, and position the previous plot using node size as a function of degree to the left



Two-way Visualization Comparison

```
CloseCent<-closeness(g)
g = graph_from_adjacency_matrix(flo, mode='undirected', diag=FALSE)
set.seed(5574)
LO = layout_with_fr(g) #Layout

par(mfrow=c(1,2)) # plot four figures - 1 rows, 2 columns
plot(g, vertex.size=1000*CloseCent, layout=LO, main="Closeness")
plot(g, vertex.size=5*cDegree, layout=LO, main="Degree")
```

Betweenness

- **Idea**: A node is central if it acts as a bridge to other nodes
- How to measure in words:
  - For each pair of nodes, compute the geodesic distance (shortest path between them)
  - Then for each node, determine the fraction of shortest paths that go through the actor in question
  - End by summing this fraction over all pairs of nodes

Betweenness

- How to measure a bit more formally:
  - Say $g_{j,k}$ equals the number of geodesics between nodes j and k
  - Say $g_{j,k}(i)$ equals the number of geodesics between nodes j and k going through i
  - Then betweenness centrality for actor i: $\sum_{j<k} \frac{g_{j,k}(i)}{g_{j,k}}$

Betweenness

- Simple way to think of $\frac{g_{j,k}(i)}{g_{j,k}}$ is the probability that a "message" from j to k goes through i
  - j and k have $g_{j,k}$ routes of communication
  - i is on $g_{j,k}(i)$ of these routes
  - a randomly selected path contains i with probability $\frac{g_{j,k}(i)}{g_{j,k}}$
- Examples where this might be useful?

Betweenness

Lets calculate this in `igraph`

```
?betweenness
```

Description

The vertex and edge betweenness are (roughly) defined by the number of geodesics (shortest paths) going through a vertex or an edge.

Usage

```
estimate_betweenness(graph, vids = V(graph), directed = TRUE, cutoff,
  weights = NULL, nobigint = TRUE)

betweenness(graph, v = V(graph), directed = TRUE, weights = NULL,
  nobigint = TRUE, normalized = FALSE)

edge_betweenness(graph, e = E(graph), directed = TRUE, weights = NULL)
```

Arguments

`graph`	The graph to analyze.
`vids`	The vertices for which the vertex betweenness estimation will be calculated.
`directed`	Logical, whether directed paths should be considered while determining the shortest paths.
`cutoff`	The maximum path length to consider when calculating the betweenness. If zero or negative then there is no such limit.
`weights`	Optional positive weight vector for calculating weighted betweenness. If the graph has a `weight` edge attribute, then this is used by default. Weights are used to calculate weighted shortest paths, so they are interpreted as distances.
`nobigint`	Logical scalar, whether to use big integers during the calculation. This is only required for lattice-like graphs that have very many shortest paths between a pair of vertices. If `TRUE` (the default), then big integers are not used.
`v`	The vertices for which the vertex betweenness will be calculated.

Betweenness

Lets calculate this in `igraph`

```
sort(betweenness(g), decreasing=TRUE)[1:6]
```

```
##   Medici Guadagni  Albizzi Salviati  Ridolfi Bischeri
## 47.50000 23.16667 19.33333 13.00000 10.33333  9.50000
```

Degree vs Closeness vs Betweenness

```
sort(degree(g), decreasing=TRUE)[1:6]
```

```
##     Medici   Guadagni    Strozzi    Albizzi   Bischeri Castellani
##          6          4          4          3          3          3
```

```
sort(closeness(g), decreasing=TRUE)[1:6]
```

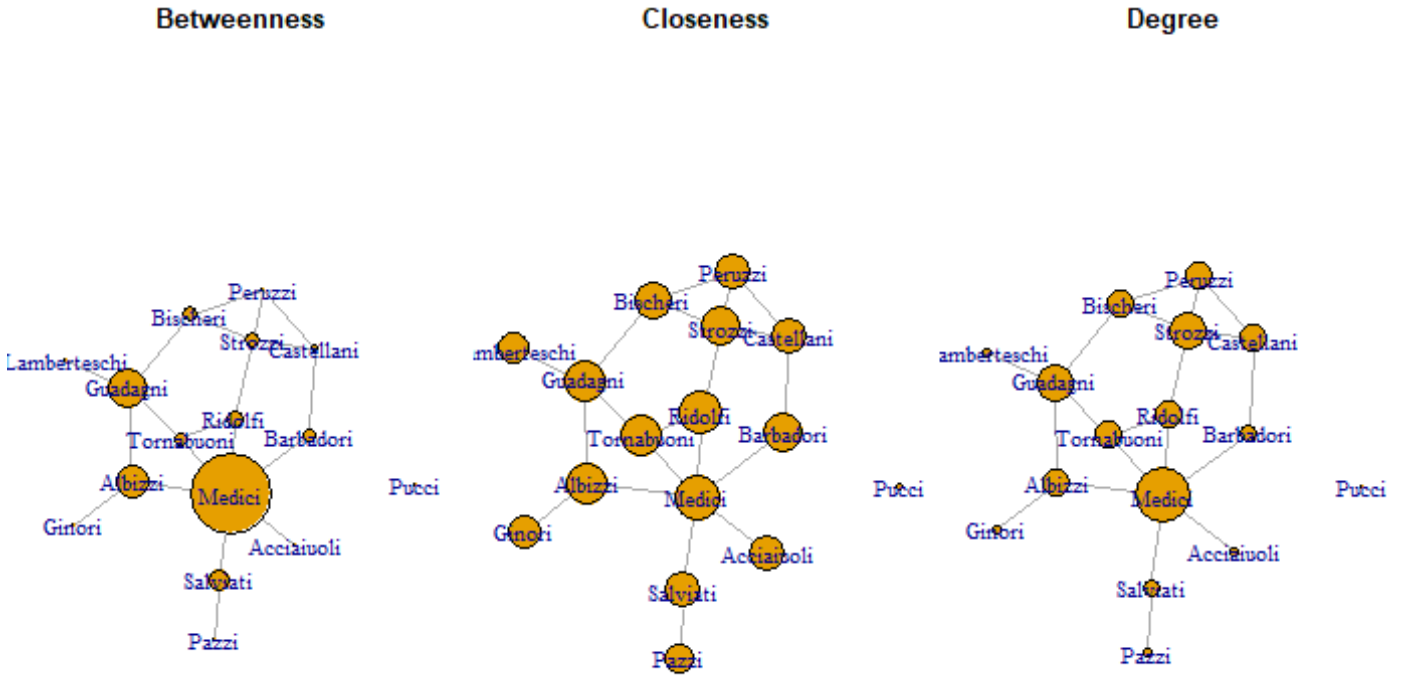
```
##     Medici    Ridolfi    Albizzi Tornabuoni   Guadagni  Barbadori
## 0.02439024 0.02272727 0.02222222 0.02222222 0.02173913 0.02083333
```

```
sort(betweenness(g), decreasing=TRUE)[1:6]
```

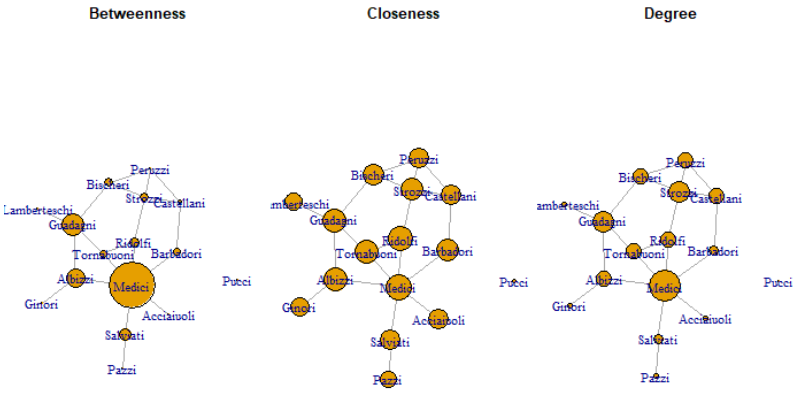
```
##   Medici Guadagni  Albizzi Salviati  Ridolfi Bischeri
## 47.50000 23.16667 19.33333 13.00000 10.33333  9.50000
```

Three-Way Visualization Comparison

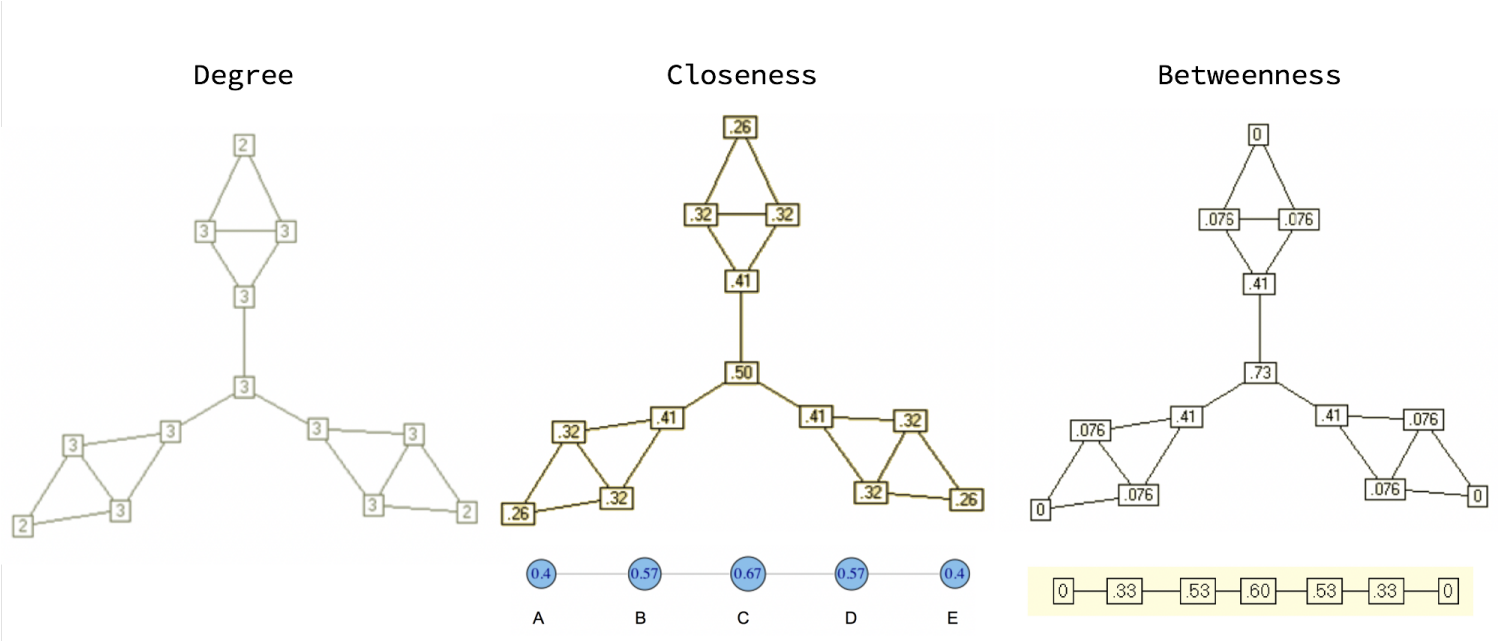
Now plot adjusting node sizes by closeness centrality, and position the previous plot using node size as a function of degree to the left



Three-Way Visualization Comparison



Comparison of these measures (Thanks to Arifuzzaman & Bhuiyan)



Comparison of these measures (Thanks to Arifuzzaman & Bhuiyan)

	Low Degree	Low Closeness	Low Betweenness
High Degree		Embedded in cluster that is far from the rest of the network	Ego's connections are redundant - communication bypasses him/her
High Closeness	Key player tied to important/active alters		Probably multiple paths in the network, ego is near many people, but so are many others
High Betweenness	Ego's few ties are crucial for network flow	Very rare cell. Would mean that ego monopolizes the ties from a small number of people to many others.	

One more ... Eigenvector

- **Idea**: An actor is more central if it is connected to other more central actors
- Eigenvector centrality: centrality of each node is proportional to the sum of the centralities of its neighbors (let c_i^e denote the eigenvector centrality of actor i):
  - $c_i^e = \frac{1}{\lambda} \sum_{j:j \neq i} y_{ij} c_j^e$
- Based on some matrix algebra:
  - $Yc^e = \lambda c^e$
  - Vector c^e satisfying the above equation is an **eigenvector** of Y
- Generally, there are multiple eigenvectors, centrality is taken to be the one corresponding to the largest value of λ
- Examples of where this might be useful?

Trillion dollar application

`Google Describing PageRank`: PageRank relies on the uniquely democratic nature of the web by using its vast link structure as an indicator of an individual page's value. In essence, Google interprets a link from page A to page B as a vote, by page A, for page B. But, Google looks at more than the sheer volume of votes, or links a page receives; it also analyzes the page that casts the vote. Votes cast by pages that are themselves "important" weigh more heavily and help to make other pages "important."

Eigenvector centrality

Lets calculate this in `igraph`

```
?eigen_centrality
```

Eigenvector centrality

Lets calculate this in `igraph`

```
sort(eigen_centrality(g)$vector, decreasing=TRUE)[1:6]
```

```
##     Medici    Strozzi    Ridolfi Tornabuoni   Guadagni   Bischeri
##  1.0000000  0.8272688  0.7937398  0.7572302  0.6718805  0.6572037
```

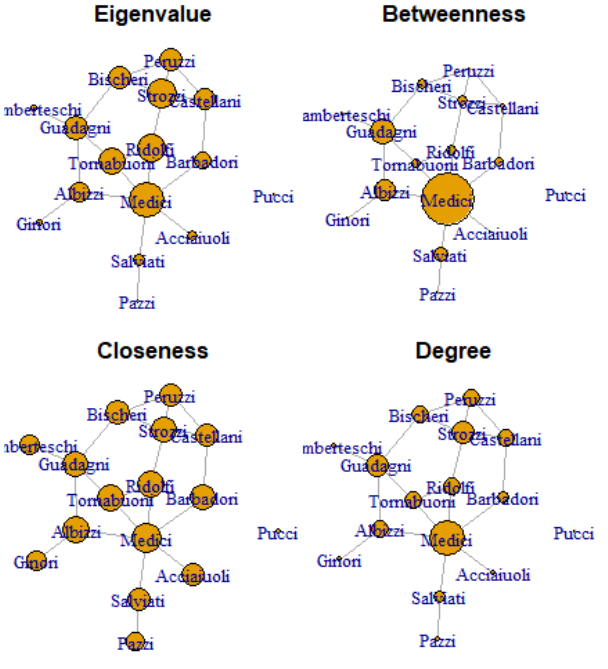
Degree vs Closeness vs Betweenness vs Eigenvector

```
sort(degree(g), decreasing=TRUE)[1:6]
sort(closeness(g), decreasing=TRUE)[1:6]
sort(betweenness(g), decreasing=TRUE)[1:6]
sort(eigen_centrality(g)$vector, decreasing=TRUE)[1:6]
```

```
##     Medici   Guadagni    Strozzi    Albizzi   Bischeri Castellani
##          6          4          4          3          3          3

##     Medici    Ridolfi    Albizzi Tornabuoni   Guadagni  Barbadori
## 0.02439024 0.02272727 0.02222222 0.02222222 0.02173913 0.02083333

##   Medici Guadagni  Albizzi Salviati  Ridolfi Bischeri
## 47.50000 23.16667 19.33333 13.00000 10.33333  9.50000

##     Medici    Strozzi    Ridolfi Tornabuoni   Guadagni   Bischeri
##  1.0000000  0.8272688  0.7937398  0.7572302  0.6718805  0.6572037
```

Four-Way Visualization Comparison

- Now plot adjusting node sizes by eigenvalue centrality, and position the previous plot using node size as a function of degree to the left



```
LO = layout_with_fr(g) #Layout
EigenCent<-eigen_centrality(g)

par(mfrow=c(2,2)) # plot four figures - 2 row, 2 columns
plot(g, vertex.size=30*EigenCent$vector, layout=LO, main="Eigenvalue'
plot(g, vertex.size=BetwCent, layout=LO, main="Betweenness")
plot(g, vertex.size=1000*CloseCent, layout=LO, main="Closeness")
plot(g, vertex.size=5*cDegree, layout=LO, main="Degree")
```

Break-Out Groups

For each of the following networks, think of the best measure of centrality to measure the amount of influence in different contexts.

- countries connected by trade relations
- a network of student friendships on a university campus
- a network of legislators connected by co-sponsorships of bills
- a network of CEOs connected based on their undergraduate institutions

A word from Wasserman & Faust

"..., we do not expect that the most fruitful development in descriptive techniques will be the continued addition of yet another definition of centrality measure or yet another subgroup definition or yet another definition of equivalence. Rather, we expect that careful assessment of the usefulness of current methods in substantive and theoretical applications will be helpful in determining when, and under what conditions, each method is useful (perhaps in conjunction with statistical assumptions). Considerable work also needs to be done on measurement properties (such as sampling variability) of the current measures."

Application: Feld (1991)

Goals:

- Learn to calculate various measures of centrality
- Continue familiarizing yourself with managing network data
- Replicate the analysis by Feld (1991)

Feld (1991) Summary

- Most people have fewer friends than their friends. Why?
- The distribution of the number of friends' friends is pulled to the right by a small number of people with a lot of friends.
- Other examples of the same phenomenon?

Plan

- Step 1: Load the data. We'll be using a dataset named *coleman* from the `sna` package.
    - Notice that these are not the same data as in Feld (1991)

```
library(sna)
data(coleman)
#?coleman
```

Plan

- Step 2: Plot the network in the fall. Notice that our data is a 2x73x73 array.

```
#coleman[1,,] #Fall
#dim(coleman[1,,])
library(igraph)
pGraph<-graph_from_adjacency_matrix(coleman[1,,])
V(pGraph)$label<- NA
plot(pGraph, vertex.size=10, edge.arrow.size=.2,vertex.color="blue" )
```

Now Let's Replicate Feld's Analysis

Table 1 on p. 1466

TABLE 1

A SUMMARY OF THE NUMBERS OF FRIENDS AND THE MEAN NUMBERS OF FRIENDS OF FRIENDS FOR EACH OF THE GIRLS IN FIGURE 1

	Number of Friends (x_i)	Total Number of Friends of Her Friends (Σx_j)	Mean Number of Friends of Her Friends ($\Sigma x_j/x_i$)
Betty	1	4	4
Sue	4	11	2.75
Alice	4	12	3
Jane	2	7	3.5
Pam	3	10	3.3
Dale	3	10	3.3
Carol	2	4	2
Tina	1	2	2
Total	20	60	23.92
Mean	2.5*	3†	2.99*

* For eight girls.
† For 20 friends.

Your Turn

Calculate number of friends (degree centrality) for each person.

```
##   from_id deg
## 1       1   5
## 2       2   2
## 3       3   2
## 4       4   4
## 5       5   2
```

Table 1 Replication Cont'd

1. Calculate individual's degree for Coleman data using dplyr:

```
deg<-as_tibble(coleman[1,,]) %>%
  mutate(from_id=row.names(coleman[1,,])) %>%
  pivot_longer(cols=`1`:`73`,names_to="to_id",values_to="friend") %>%
  group_by(from_id) %>%
  summarise(degree=sum(friend)) %>%
  ungroup()
```

Table 1 Replication Cont'd

1. Calculate total number of friends of friends of each person.
2. Calculate average number of friends of friends.

```
## # A tibble: 69 x 4
##    from_id  ideg tot_fdeg ave_deg
##    <chr>   <dbl>    <dbl>   <dbl>
##  1 1           5       16    3.2
##  2 11          4       16    4
##  3 12          3       13    4.33
##  4 13          4       15    3.75
##  5 14          2        8    4
##  6 15          1        5    5
##  7 16          3       10    3.33
##  8 17          2        3    1.5
##  9 18          3       12    4
## 10 19          5       17    3.4
## # ... with 59 more rows
```

Table 1 Replication Cont'd

1. Calculate total number of friends of friends of each person.
2. Calculate average number of friends of friends.

```
fdeg<-as_tibble(coleman[1,,]) %>%
  mutate(from_id=row.names(coleman[1,,])) %>%
  pivot_longer(cols=`1`:`73`,names_to="to_id",values_to="friend") %>%
  filter(friend==1) %>%
  group_by(from_id)%>%
  mutate(idegree=sum(friend), to_id=to_id) %>%
  left_join(deg, by=c("to_id"="from_id")) %>%
  rename(fdegree=degree)  %>%
  mutate(tot_fdeg=sum(fdegree),ave_fdeg=mean(fdegree)) %>%
  ungroup()

fdeg %>% group_by(from_id) %>% summarise(ideg=first(idegree),tot_fdeg
```

Figure 3 Replication

1. Plot individual's degree distribution for Coleman data:

```
p<-deg %>%
  ggplot() +
  geom_bar(aes(x=degree))+
  xlab("Individual's Degree")+
  theme_classic()
```

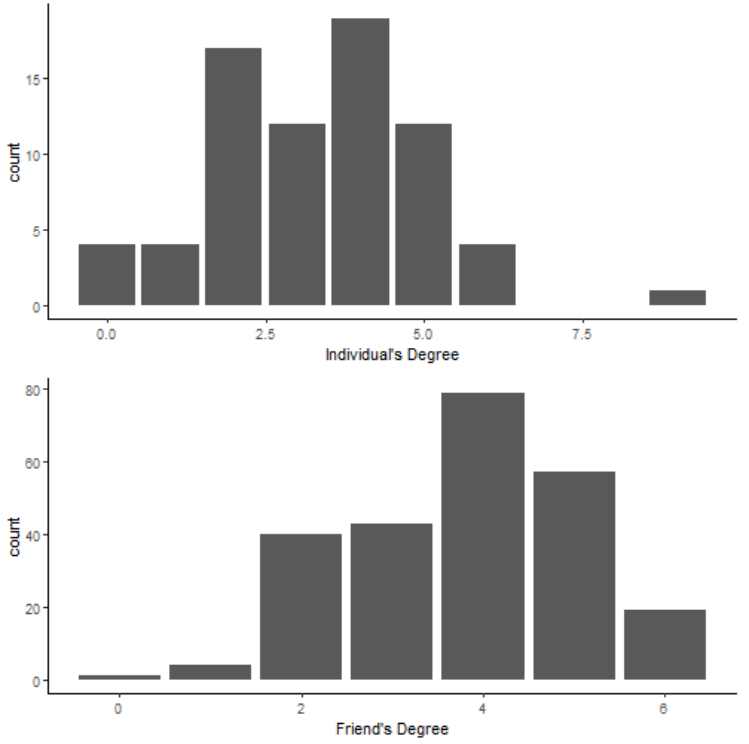
Figure 3 Replication Cont'd

1. Plot friend's degree distribution:

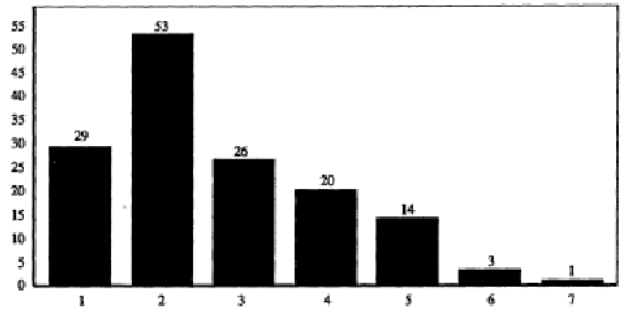
```
p1<-fdeg %>%  ggplot() +geom_bar(aes(x=fdegree))+
  xlab("Friend's Degree")+
  theme_classic()
```

Figure 3 Replication Cont'd

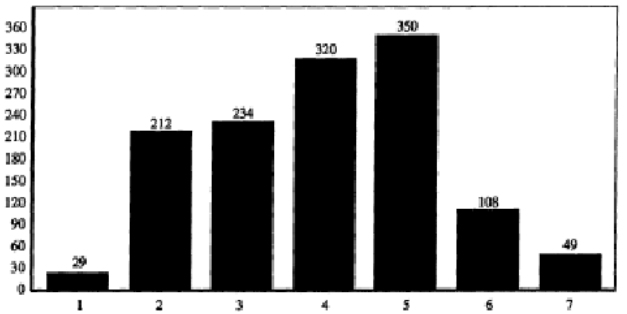
1. Combine the plots



Feld (1991) Figure 3



a) The mean is 2.7.



b) The mean is 3.4

FIG. 3.—(*a*) Distribution of numbers of friends for Marketville girls; (*b*) distribution of number of friends' friends for Marketville girls.