Igraph: An introduction

Sebastian Robledo Ph.D.(c)

R language has become very popular between researchers because it let them do sophisticated statistical calculations to a lot of data and with the support of a huge community.

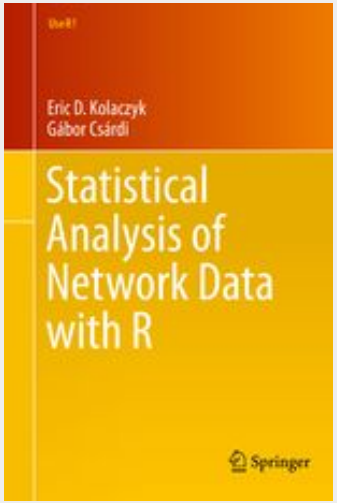
Purpose

The purpose of this session is to introduce the participants to the main technical concepts of network data analysis in R using igraph, for example: how to read data from R, how to convert this data to a graph object and how to do some descriptive analysis.

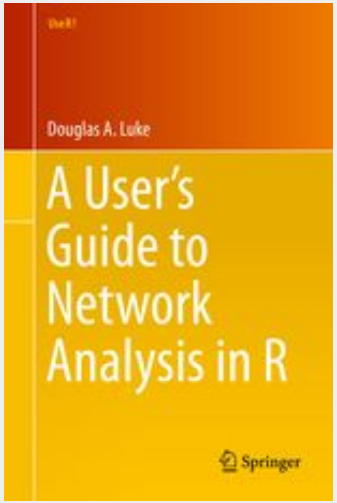
Methodology

Just run the code!
The objective is to focus On the main features of igraph (what can we do with it?) and avoid the thecnical problems.

Statistical Analysis of Network Data with R



A User's Guide to Network Analysis in R



Useful links

Triads Go Boink!

Network Analysis and Visualization with R and igraph

Social Network Analysis Labs in R and SoNIA

Examples...

Datasets

Longitudinal Network Data Sources

Network Datasets

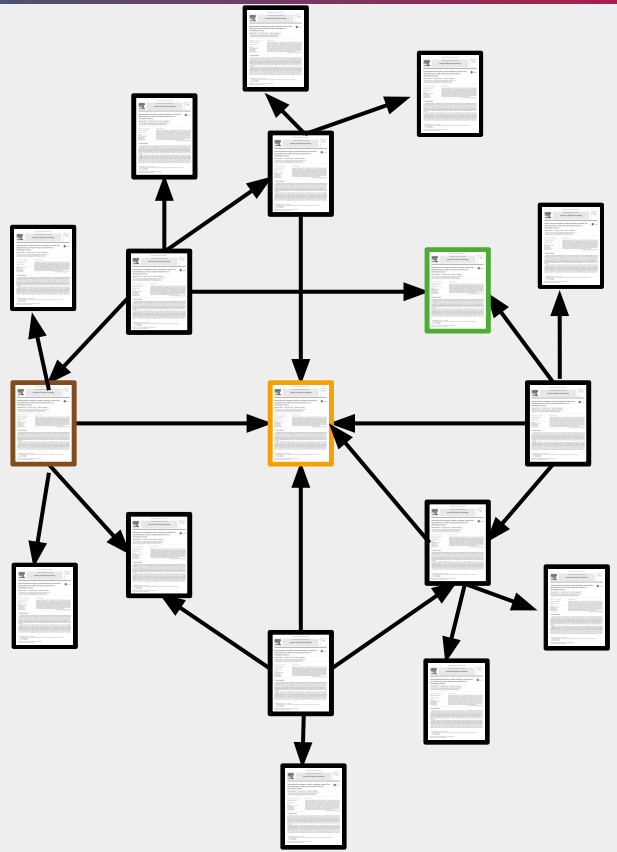
The Koblenz Network Collection

Stanford Large Network Dataset Collection

Question

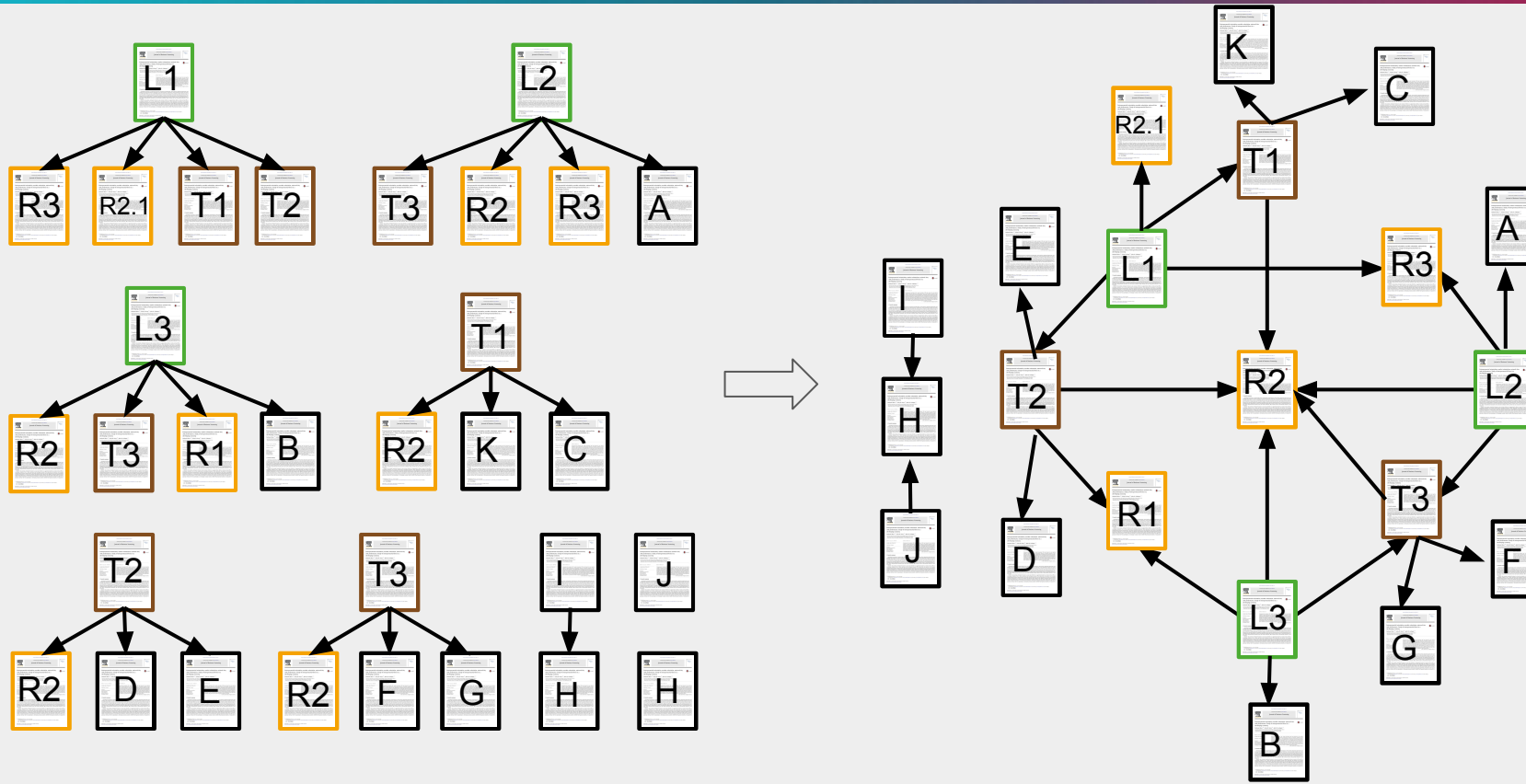
What are the most important papers in a research topic using graph theory?

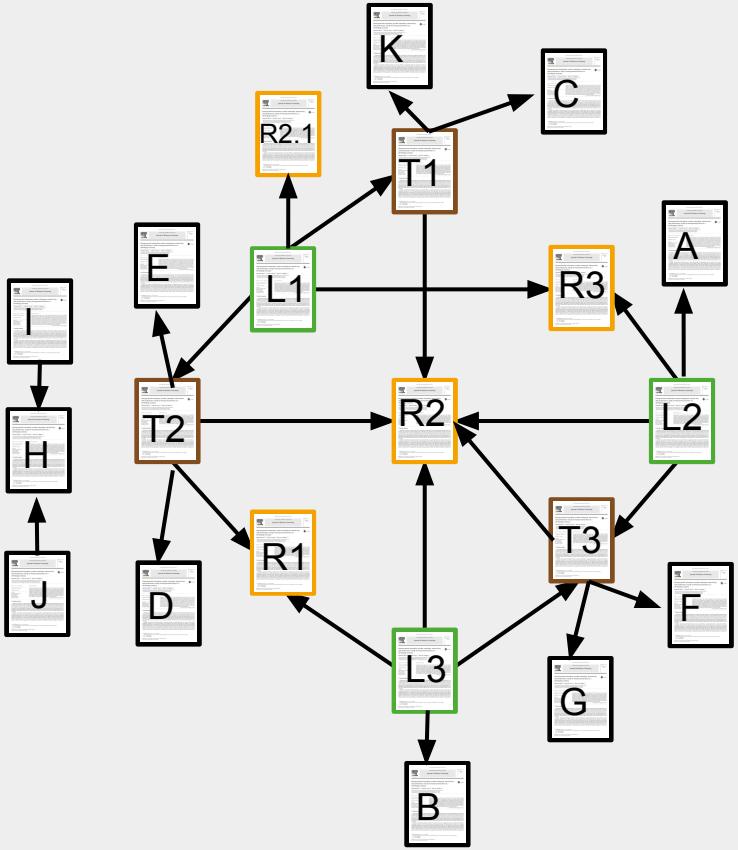
1. High in-degree and 0 out-degree — Seminals
2. High Betweeness — Structurals
3. High out-degree and 0 in-degree — Current



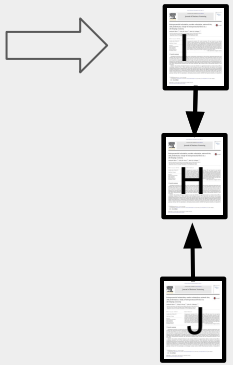
Data

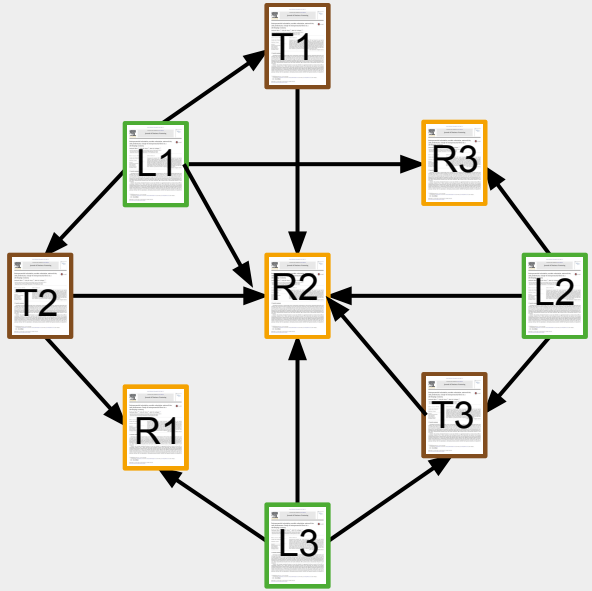
Web of Science	TITLE: (social networks) AND TOPIC: (organization)	306 papers


L1
R3
R2.1
T1
T2
L2
T3
R2
R3
A
L3
R2
T3
R1
B
T1
R2
K
C
T2
R2
D
E
T3
R2
F
G
I
J
H
H
I
H
J
K
C
R2.1
T1
E
L1
R3
A
T2
R2
L2
D
R1
T3
F
L3
G
B

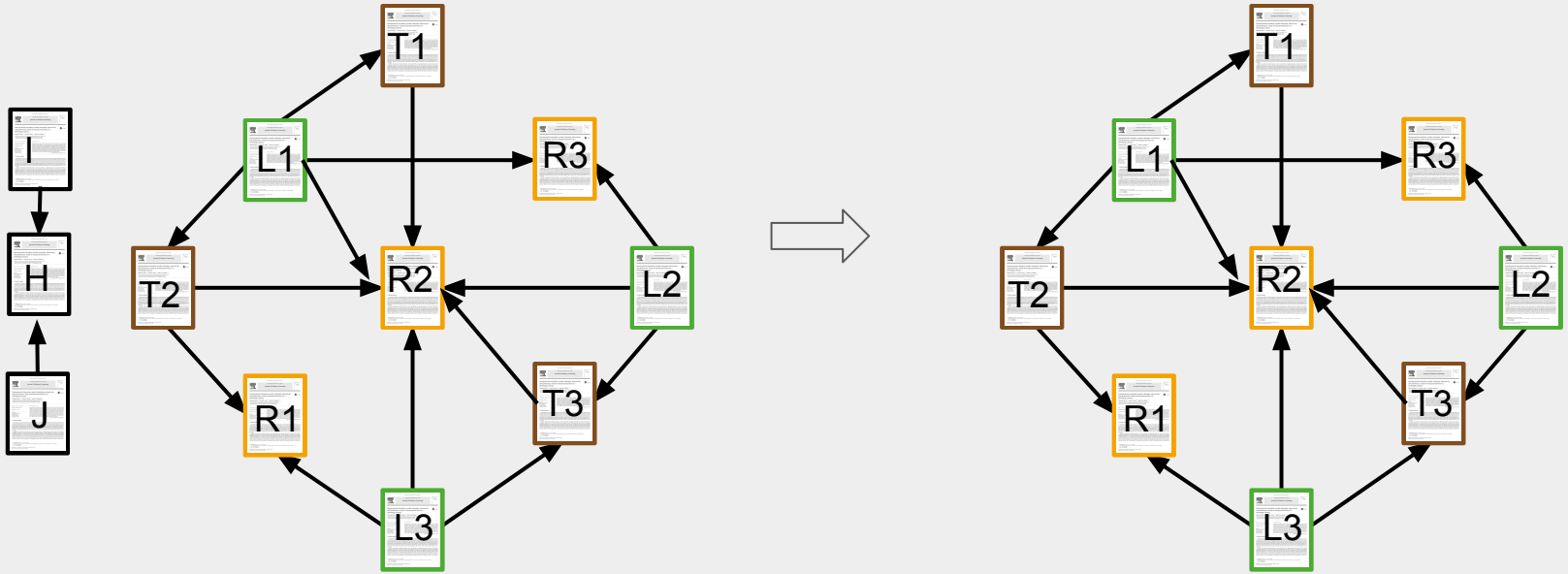


Deleting nodes with in-degre = 1 and out-degree = 0





Extracting the giant component



Deleting in and out degree attributes

```
net.tidied.2 <- delete_vertex_attr(net.tidied.1, "outdegree" )
net.tidied.3 <- delete_vertex_attr(net.tidied.2, "indegree" )
net.tidied <- net.tidied.3
summary(net.tidied)
```

```
IGRAPH DN-- 1062 3115 --
+ attr: name (v/c), label (v/c)
```

Global Properties

Density

```
graph.density(net.tidied)
```

Transitivity

```
transitivity(net.tidied, type = "global")
```

Diameter

```
diameter(net.tidied, directed = TRUE, weights = NA)
```

Centralization

```
centr_degree(net.tidied, mode = "all")$centralization
```

Local Properties

Degree: in and out

```
V(net.tidied)$indegree <- degree(net.tidied, mode = "in")
V(net.tidied)$outdegree <- degree(net.tidied, mode = "out")
V(net.tidied)$degree <- degree(net.tidied, mode = "all")
```

Betweenness

```
V(net.tidied)$bet <- betweenness(net.tidied)
```

Bonacich

```
V(net.tidied)$bonacich <- power_centrality(net.tidied)
```

Transitivity

```
V(net.tidied)$transitivity <- transitivity(net.tidied, type = "local")
head(as_data_frame(net.tidied, what = "vertices"))
```

Subgroups and communities

Based on greedy optimization of modularity

```
community <-
cluster_fast_greedy(as.undirected(net.tidied))
V(net.tidied)$community <- community$membership
```

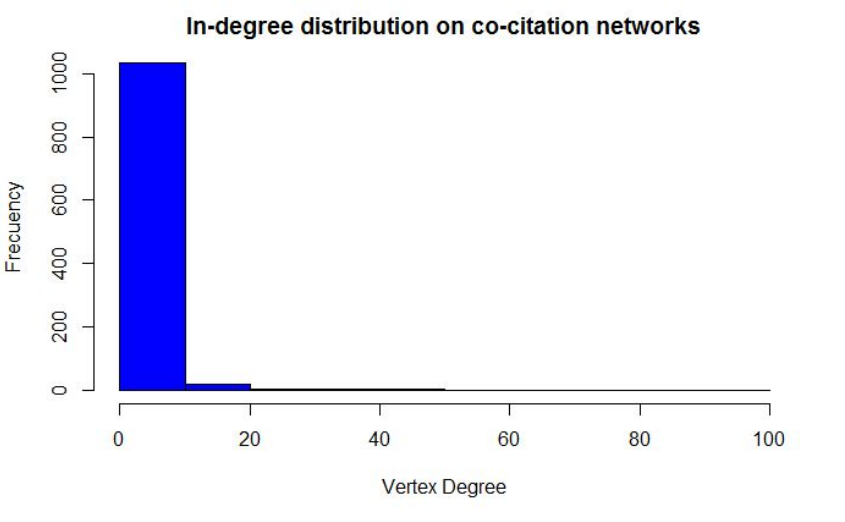
Coreness

```
V(net.tidied)$coreness <- coreness(net.tidied)
```

Topological Properties

Degree Distribution

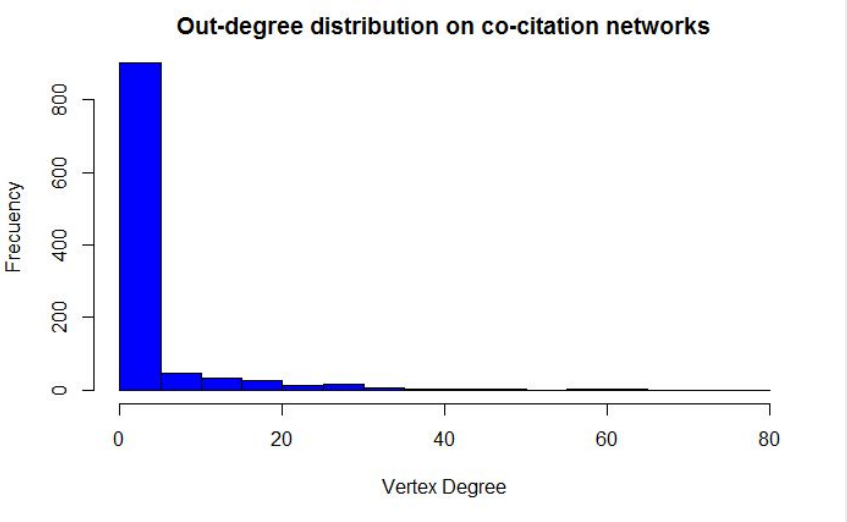
```
hist(degree(net.tidied, mode = "in"), col="blue",
    main = "In-degree distribution on co-citation networks",
    xlab = "Vertex Degree", ylab = "Frecuency")
```



Topological Properties

Degree Distribution

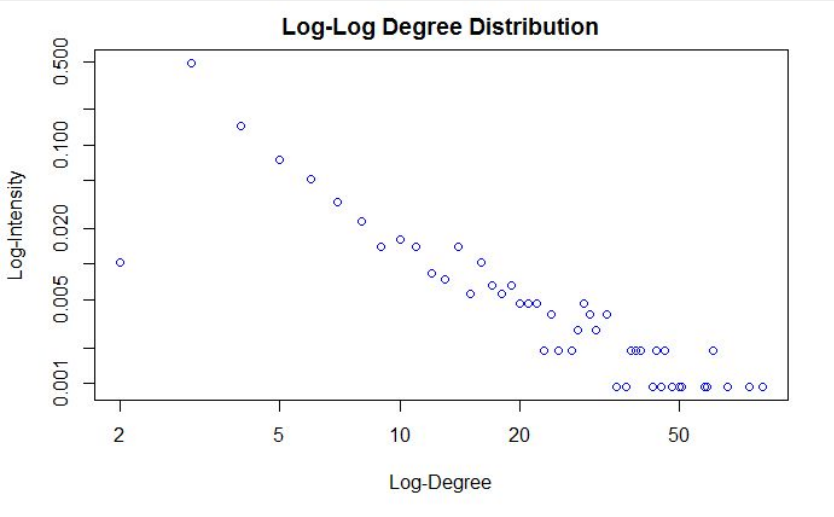
```
hist(degree(net.tidied, mode = "out"), col="blue",
    main = "Out-degree distribution on co-citation networks",
    xlab = "Vertex Degree", ylab = "Frecuency")
```



Topological Properties

Log - Log Degree Distribution

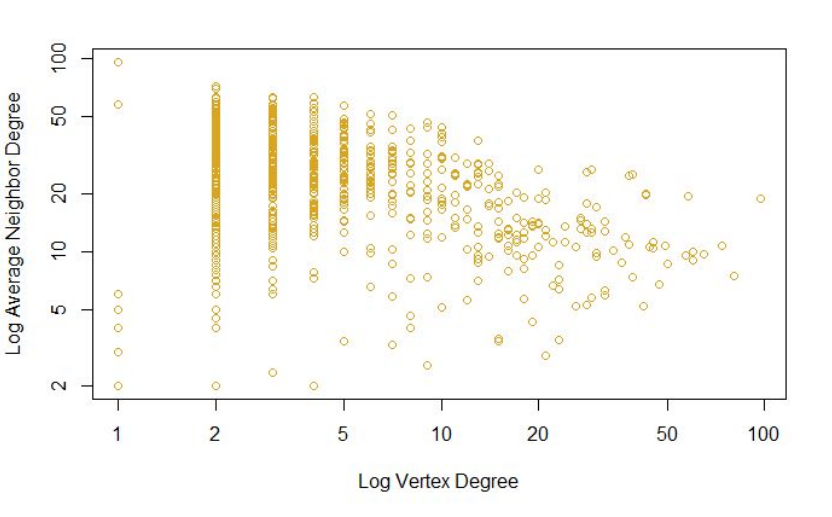
```
d.net.tidied <- degree(net.tidied, mode = "all")
dd.net.tidied <- degree.distribution(net.tidied)
d <- 1:max(d.net.tidied)
ind <- (dd.net.tidied != 0)
plot(d[ind], dd.net.tidied[ind], log = "xy", col = "blue",
    xlab = c("Log-Degree"), ylab = c("Log-Intensity"),
    main = "Log-Log Degree Distribution")
```



Topological Properties

Log - Log Average Neighbor Degree

```
a.nn.deg.net.tidied <- graph.knn(net.tidied,V(net.tidied))$knn
plot(d.net.tidied, a.nn.deg.net.tidied,
     log="xy", col="goldenrod",
     xlab=c("Log Vertex Degree"),
     ylab=c("Log Average Neighbor Degree"))
```



Identifidying key actors in the network: Seminal, Structural, and Current papers

```
df.1 <- as_data_frame(net.tidied, what = "vertices")
df.1 <- df.1[, c("label", "outdegree", "indegree", "bet")]
seminals <- df.1[df.1$outdegree == 0,]
seminals <- head(seminals[order(seminals$indegree,
                                rev(seminals$indegree),
                                decreasing = TRUE), "label"], 10)
structurals <- head(df.1[order(df.1$bet, rev(df.1$bet),
                               decreasing = TRUE), "label"], 10)
currents <- df.1[df.1$indegree == 0,]
currents <- head(currents[order(currents$outdegree,
                                rev(currents$outdegree),
                                decreasing = TRUE), "label"], 10)
key.papers <- data.frame(seminals, structurals, currents, stringsAsFactors = FALSE)
```

Identifidying key actors in the network: Seminal, Structural, and Current papers

	seminals	structurals	currents
1	Wasserman S., 1994, Social Network Anal, P249	Cross R, 2002, Calif Manage Rev, V44, P25	Farine Dr, 2015, J Anim Ecol, V84, P1144, Doi 10.11...
2	Scott J., 1991, Social Network Anal, P92	Borgatti Sp, 2009, J Supply Chain Manag, V45, P5, Doi...	Conway S, 2014, Brit J Manage, V25, P102, Doi 10.11...
3	Freeman Lc, 1979, Soc Networks, V1, P215, Doi 10.1...	Burt Rs, 2013, Annu Rev Psychol, V64, P527, Doi 10....	Rose Pe, 2015, Anim Welfare, V24, P123, Doi 10.712...
4	Borgatti S. P., 2002, Ucinet Windows Softw	Fattore G, 2009, Health Policy, V92, P141, Doi 10.10...	El Louadi M, 2008, Knowl Man Res Pract, V6, P199, D...
5	Granovetter Ms, 1973, Am J Sociol, V78, P1360, Doi 1...	Hu C, 2008, Int J Hosp Manag, V27, P302, Doi 10.10...	Ho Y, 2013, Asia Pac J Manag, V30, P1265, Doi 10.1...
6	Burt R. S., 1992, Structural Holes Soc	Kasper C, 2009, Primates, V50, P343, Doi 10.1007/s...	Diez-vial I, 2014, Knowl Man Res Pract, V12, P276, D...
7	Borgatti Sp, 2003, J Manage, V29, P991, Doi 10.1016...	James R, 2009, Behav Ecol Sociobiol, V63, P989, Doi ...	Sutanto J, 2011, Long Range Plann, V44, P421, Doi 1...
8	Hanneman Ra, 2005, Intro Social Network	Yousefi-nooraie R, 2012, Bmc Health Serv Res, V12, ...	Garcia Md, 2016, Rev Esp Investig Soc, P23, Doi 10.5...
9	Borgatti Sp, 2009, Science, V323, P892, Doi 10.1126...	Makagon Mm, 2012, Appl Anim Behav Sci, V138, P15...	Khan Gf, 2016, Commun Assoc Inf Sys, V39, P367
10	Borgatti Sp, 2005, Soc Networks, V27, P55, Doi 10.10...	Allen J, 2007, R&d Manage, V37, P179, Doi 10.1111/...	Zheng X, 2016, Int J Proj Manag, V34, P1214, Doi 10....

```
library(igraph)
library(sand)
```

Create a graph object g with N = 7 vertices

```
g <- graph.formula(1-2, 1-3, 2-3, 2-4, 3-5, 4-5, 4-6, 4-7, 5-6, 6-7)
```

To see the vertices

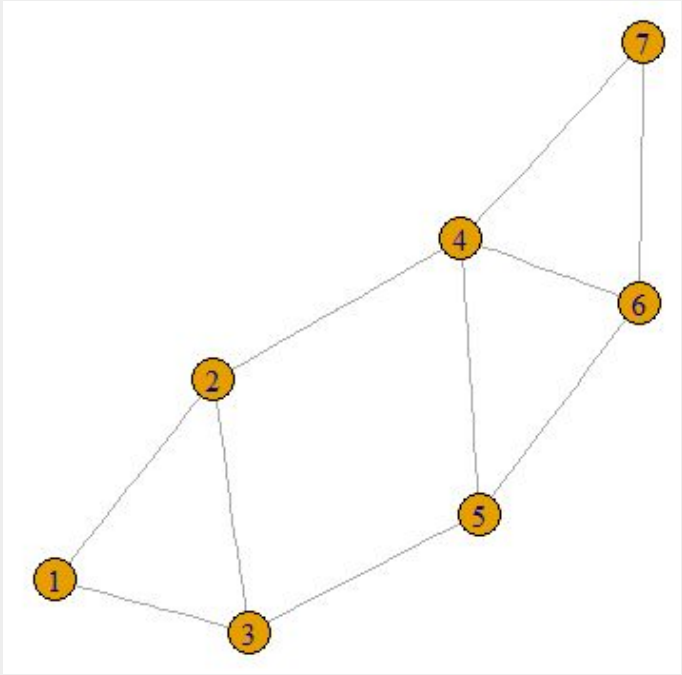
V(g)

```
> V(g)
+ 7/7 vertices, named:
[1] 1 2 3 4 5 6 7
```

To see the edges

E(g)

```
> E(g)
+ 10/10 edges (vertex names):
 [1] 1--2 1--3 2--3 2--4 3--5 4--5 4--6 4--7 5--6 6--7
```

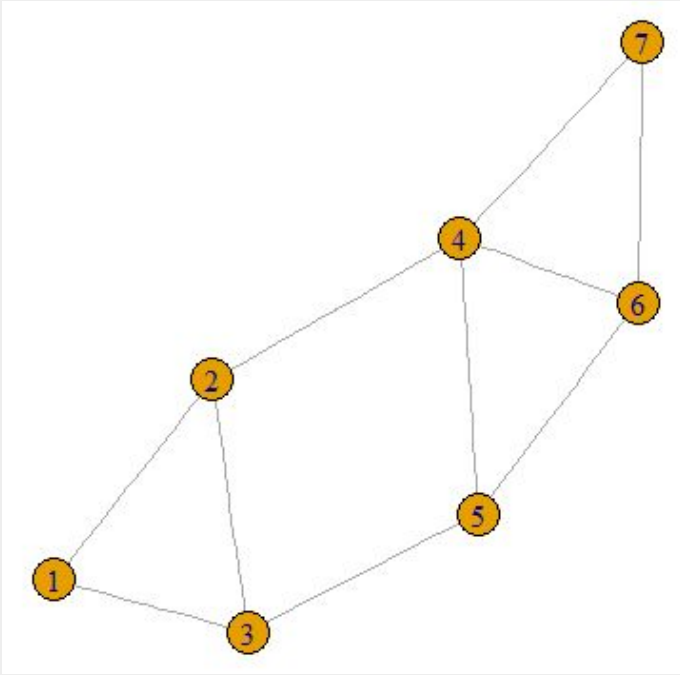


More compressed format

str(g)

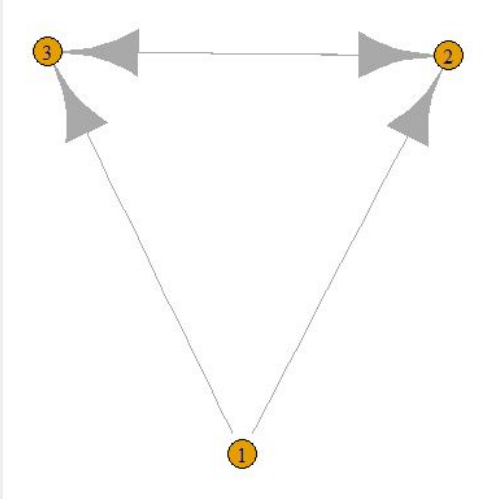
```
> str(g)
IGRAPH UN-- 7 10 --
+ attr: name (v/c)
+ edges (vertex names):
1 -- 2, 3
2 -- 1, 3, 4
3 -- 1, 2, 5
4 -- 2, 5, 6, 7
5 -- 3, 4, 6
6 -- 4, 5, 7
7 -- 4, 6
```

UN : Undirected
UNW: Undirected + Weighted

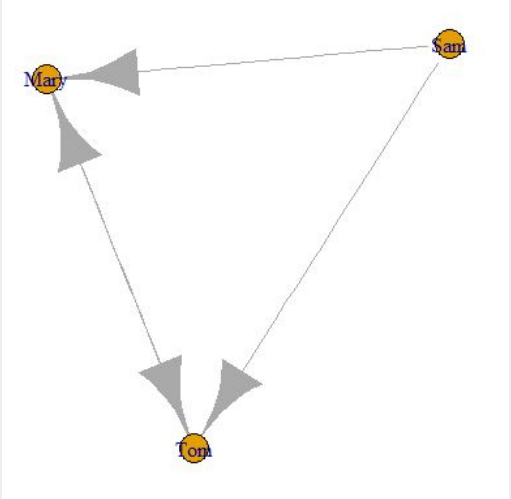


plot(g)

```
dg <- graph.formula(1-+2, 1-+3, 2++3)

plot(dg)
```




```
dg.1 <- graph.formula(Sam-+Mary, Sam-+Tom, Mary++Tom)

plot(dg.1)
```



edgelist <- get.edgelist(g)
View(edgelist)

V1	V2
1	2
1	3
2	3
2	4
3	5
4	5
4	6
4	7
5	6
6	7

matrix <- as.matrix(get.adjacency(g))
View(matrix)

	1	2	3	4	5	6	7
1	0	1	1	0	0	0	0
2	1	0	1	1	0	0	0
3	1	1	0	0	1	0	0
4	0	1	0	0	1	1	1
5	0	0	1	1	0	1	0
6	0	0	0	1	1	0	1
7	0	0	0	1	0	1	0

adjlist <- get.adjlist(g)
adjist

```
> adjlist
$`1`
+ 2/7 vertices, named:
[1] 2 3

$`2`
+ 3/7 vertices, named:
[1] 1 3 4

$`3`
+ 3/7 vertices, named:
[1] 1 2 5

$`4`
+ 4/7 vertices, named:
[1] 2 5 6 7

$`5`
+ 3/7 vertices, named:
[1] 3 4 6

$`6`
+ 3/7 vertices, named:
[1] 4 5 7

$`7`
+ 2/7 vertices, named:
[1] 4 6
```

Consider the subgraph of g induced by the first five vertices.

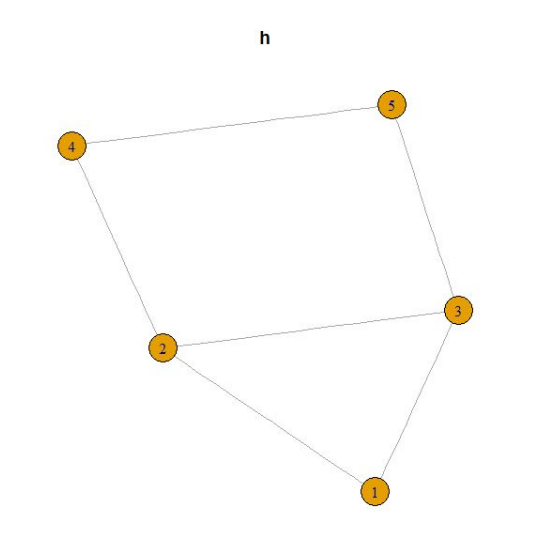
```
h <- induced.subgraph(g, 1:5)
plot(h)
```

Also, we could remove vertices 6 and 7 from g to generate h

```
h <- g - vertices(c(6,7))
```

Recovering g from h by adding these two vertices and the apropiate edges

```
h <- h + vertices(c(6,7))
plot(h)
g <- h + edges(c(4,6),c(4,7),c(5,6),c(6,7))
plot(g)
```



```
dg.2 <- graph.formula(1-+2, 1-+3, 2++3)
plot(dg.1)
```

Adding a name to our graph

```
dg.2$name <- "Toy Graph"
```

Adding names to our nodes

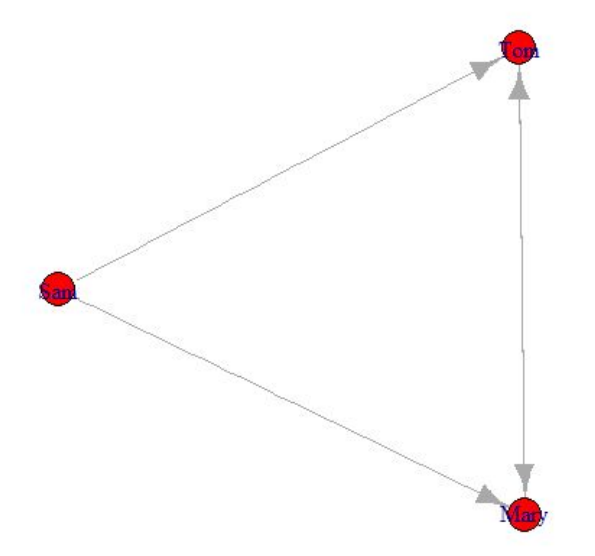
```
V(dg.2)$name <- c("Sam", "Mary", "Tom")
```

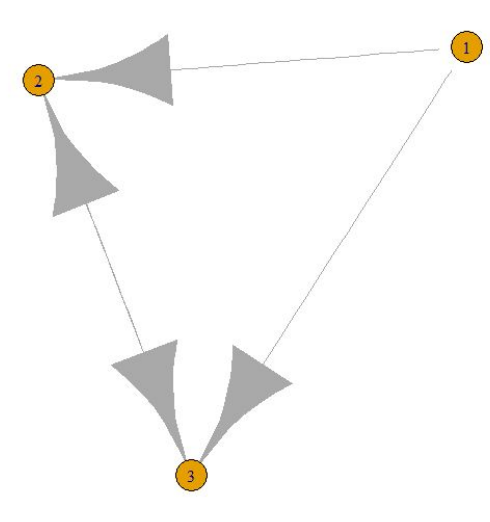
Adding gender to our nodes

```
V(dg.2)$gender <- c("M","F","M")
```

Adding color to our nodes

```
V(dg.2)$color <- "red"
summary(dg.2)
plot(dg.2)
```





library(sand)

View vertices and edges

View(elist.lazega)

View(v.attr.lazega)

Name	Seniority	Status	Gender	Office	Years	Age	Practice	School
V1	1	1	1	1	31	64	1	1
V2	2	1	1	1	32	62	2	1
V3	3	1	1	2	13	67	1	1
V4	4	1	1	1	31	59	2	3

Convert them into a graph object (g.lazega)

g.lazega <- graph.data.frame(elist.lazega, directed="FALSE", vertices=v.attr.lazega)

Checking the values of the graph

vcount(g.lazega)

ecount(g.lazega)
list.vertex.attributes(g.lazega)
summary(g.lazega)

elist.lazega

V1	V2
V1	V17
V2	V7
V2	V16
V2	V17

Uploading vertices and edges

```
vertices <- read.csv("vertices.csv", stringsAsFactors = FALSE)

edges <- read.csv("edges.csv", stringsAsFactors = FALSE)
```

Convert them into a graph object (g.lazega.1)

```
g.lazega.1 <- graph.data.frame(edges, directed = "FALSE",
                               vertices = vertices)
```

Uploading graph object

```
lazega <- read.graph("lazega.graphml", format = "graphml")
summary(lazega)
plot(lazega)
```

Uploading matrix file

```
matrix.lazega <- read.csv("matrix_1.csv", stringsAsFactors = FALSE)

matrix.lazega.1 <- as.matrix(matrix.lazega)

g.matrix.lazega <- graph.adjacency(adjmatrix = matrix.lazega.1, mode = "undirected")
```

```
summary(g.matrix.lazega )
```

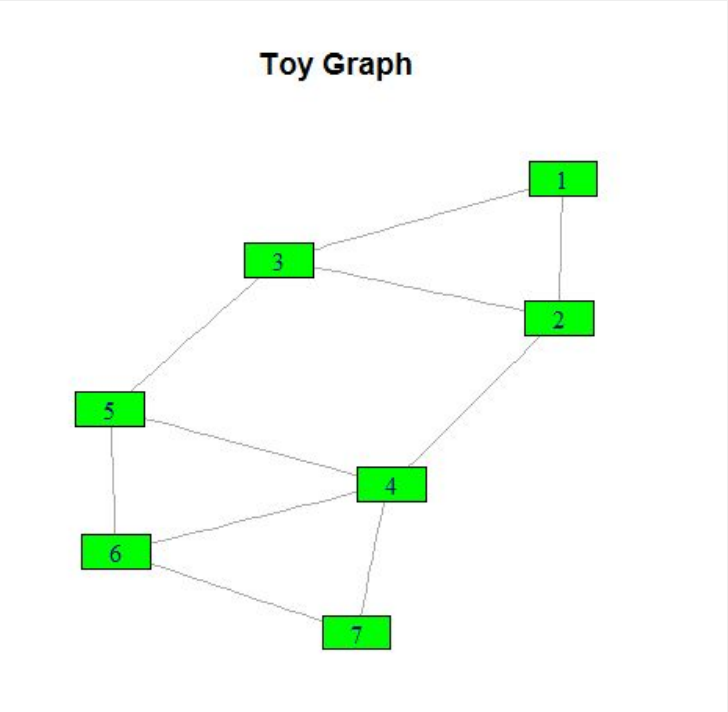
Exporting graph objects as vertices and edges

```
write.csv(v.attr.lazega, "lazega_vertices.csv", row.names = FALSE)

write.csv(elist.lazega, "lazega_edges.csv", row.names = FALSE)
```

Exporting graph objects as graph objects

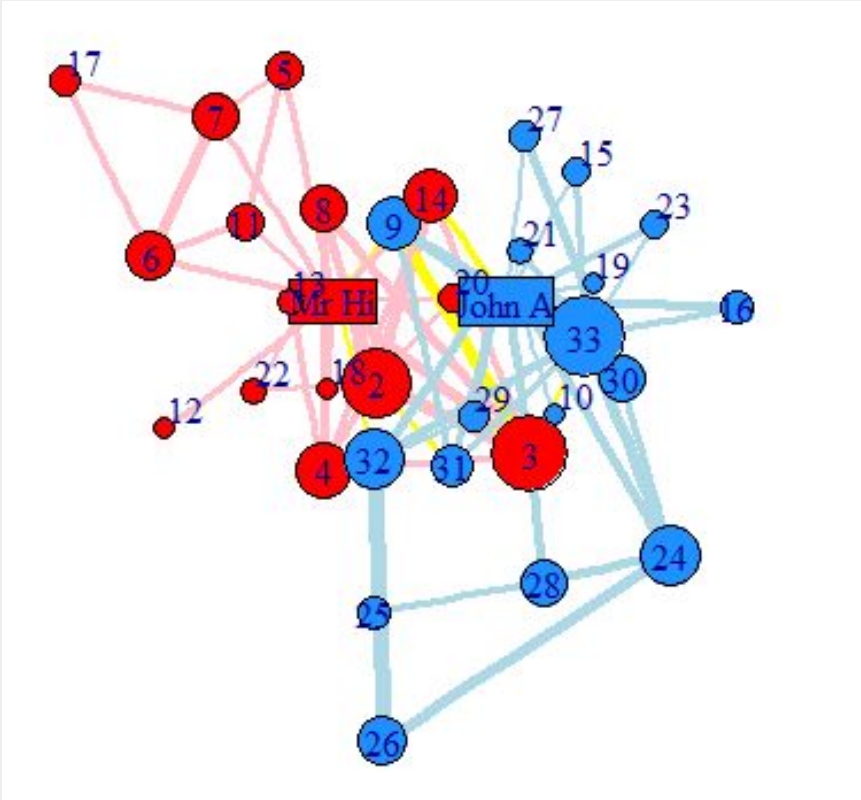
```
write.graph(g.lazega, "lazega.graphml", format = "graphml")
```

```
plot(g, vertex.size = 30, vertex.shape = "rectangle", vertex.color = "green")
title("Toy Graph")
```



```
library(igraphdata)
data(karate)
set.seed(42)
l <- layout.kamada.kawai(karate)
par(mfrow=c(1,1))
plot(karate, layout=l, vertex.label=NA)
V(karate)$label <- sub("Actor ", "", V(karate)$name)
V(karate)$shape <- "circle"
V(karate)[c("Mr Hi", "John A")]$shape <- "rectangle"
V(karate)[Faction == 1]$color <- "red"
V(karate)[Faction == 2]$color <- "dodgerblue"
V(karate)$size <- 4*sqrt(graph.strength(karate))
V(karate)$size2 <- V(karate)$size * .5
```

```
E(karate)$width <- E(karate)$weight
F1 <- V(karate)[Faction==1]
F2 <- V(karate)[Faction==2]
E(karate)[ F1 %--% F1 ]$color <- "pink"
E(karate)[ F2 %--% F2 ]$color <- "lightblue"
E(karate)[ F1 %--% F2 ]$color <- "yellow"
V(karate)$label.dist <-
      ifelse(V(karate)$size >= 10, 0, 0.75)
plot(karate, layout=l)
```


17
5
7
27
15
23
8
14
9
21
11
6
19
Mr Hi
John A
16
33
22
18
2
30
12
29
10
4
32
31
3
24
28
25
26

We are going to do an exploratory analysis of karate dataset

```
karate.vertices <- read.csv("Practice_karate_vertices.csv", stringsAsFactors = FALSE)
karate.edges <- read.csv("Practice_karate_edges.csv", stringsAsFactors = FALSE)
```

View(head(karate.edges))

	from	to	weight
1	Mr Hi	Actor 2	4
2	Mr Hi	Actor 3	5
3	Mr Hi	Actor 4	3
4	Mr Hi	Actor 5	3
5	Mr Hi	Actor 6	3
6	Mr Hi	Actor 7	3

View(head(karate.vertices))

	Faction	name	label	color
1	1	Mr Hi	H	1
2	1	Actor 2	2	1
3	1	Actor 3	3	1
4	1	Actor 4	4	1
5	1	Actor 5	5	1
6	1	Actor 6	6	1

Zachary, Wayne W. "An information flow model for conflict and fission in small groups." Journal of anthropological research 33.4 (1977): 452-473.

Now, we need to create our graph object

```
net.karate <- graph.data.frame(karate.edges, directed = FALSE)
```

```
summary(net.karate)
```

```
IGRAPH UNW- 34 78 --
+ attr: name (v/c), Faction (v/n), weight (e/n)
```

Adding attributes to vertices

```
net.karate.1 <- set.vertex.attribute(net.karate, name = "Faction",
                                     value = karate.vertices$Faction)
net.karate.2 <- set.vertex.attribute(net.karate, name = "label",
                                      value = karate.vertices$label)
net.karate.3 <- set.vertex.attribute(net.karate, name = "color",
                                      value = karate.vertices$color)
net.karate.tidied <- net.karate.3
```

```
summary(net.karate.tidied)
```

```
IGRAPH UNW- 34 78 --
+ attr: name (v/c), Faction (v/n), color (v/n), weight (e/n)
```

Global properties

Density
```
edge_density(net.karate.tidied, loops = FALSE)
```

Transitivity
```
transitivity(net.karate.tidied, type = "global")
```

Diameter
```
diameter(net.karate.tidied, directed = TRUE, weights = NA)
```

Centralization
```
centr_degree(net.karate.tidied, mode = "all")$centralization
```

Local properties

Degree

```
V(net.karate.tidied)$degree <- degree(net.karate.tidied, mode = "all")
```

Betweenness

```
V(net.karate.tidied)$bet <- betweenness(net.karate.tidied)
```

Bonacich

```
V(net.karate.tidied)$bonacich <- power_centrality(net.karate.tidied)
```

Transitivity

```
V(net.karate.tidied)$transitivity <- transitivity(net.karate.tidied, type = "local")
```

```
View(head(as_data_frame(net.karate.tidied, what = "vertices")))
```

Subgroups and communities

Greedy optimization of modularity

```
community <- cluster_fast_greedy(as.undirected(net.karate.tidied))
V(net.karate.tidied)$community <- community$membership

table(V(net.karate.tidied)$community)
```

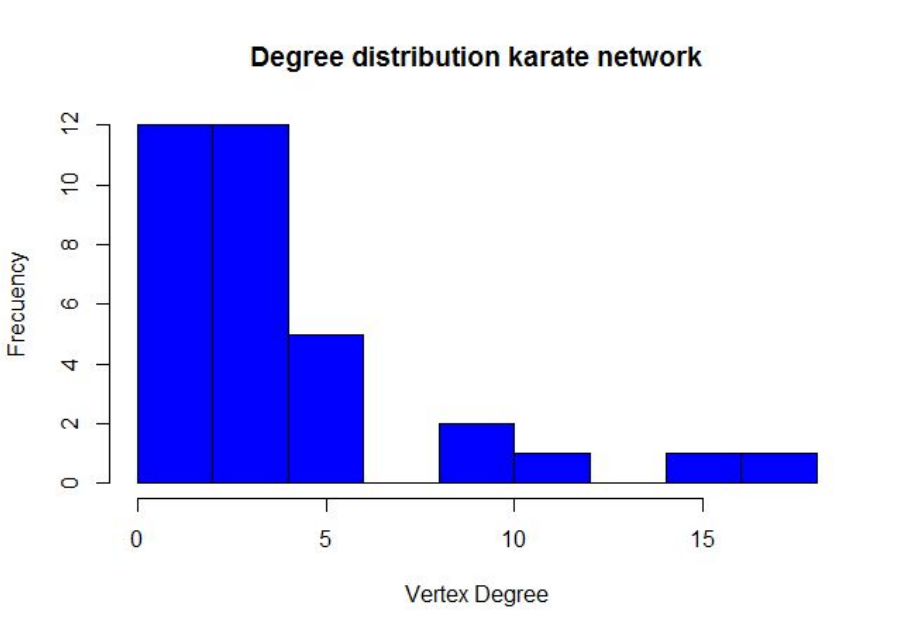
Coreness

```
V(net.karate.tidied)$coreness <- coreness(net.karate.tidied)

table(V(net.karate.tidied)$coreness)
```

Topological Properties

Degree distribution

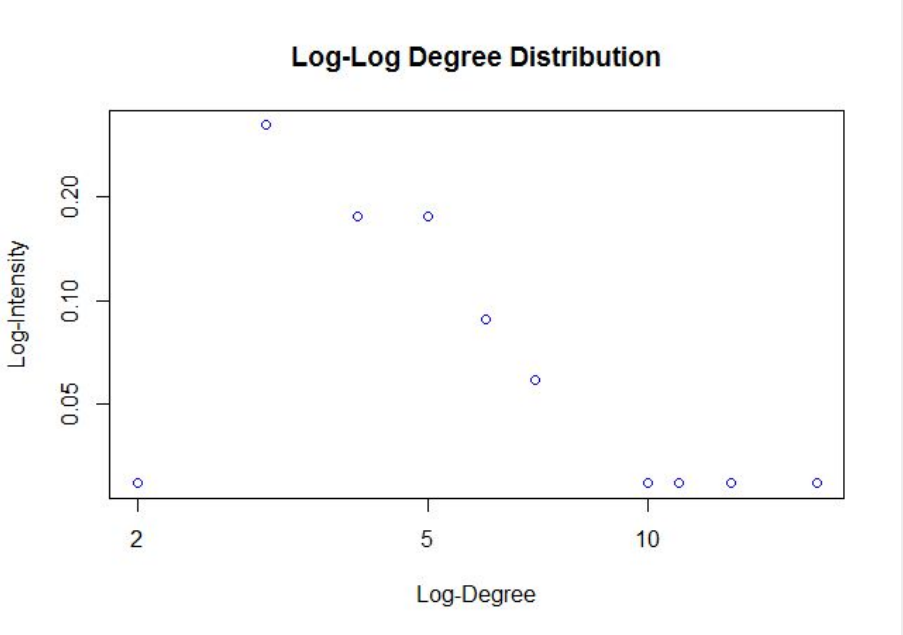
```
hist(degree(net.karate.tidied, mode = "all"), col="blue",
    main = "Degree distribution karate network",
    xlab = "Vertex Degree", ylab = "Frecuency")
```



Topological Properties

Log - log Degree distribution

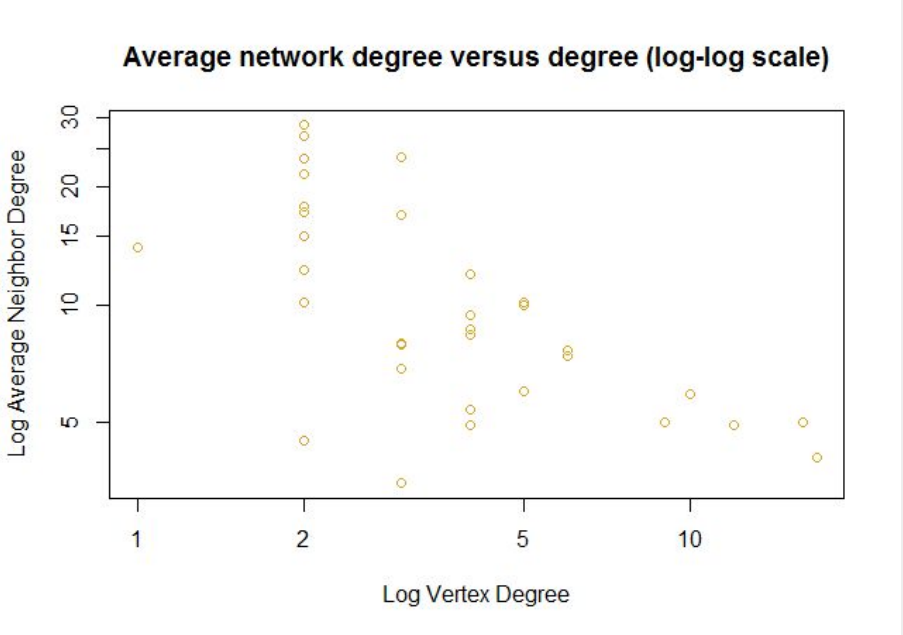
```
d.net.karate.tidied <- degree(net.karate.tidied, mode = "all")
dd.net.karate.tidied <- degree.distribution(net.karate.tidied)

d <- 1:max(d.net.karate.tidied)
ind <- (dd.net.karate.tidied != 0)
plot(d[ind], dd.net.karate.tidied[ind], log = "xy", col = "blue",
    xlab = c("Log-Degree"), ylab = c("Log-Intensity"),
    main = "Log-Log Degree Distribution")
```



Topological Properties

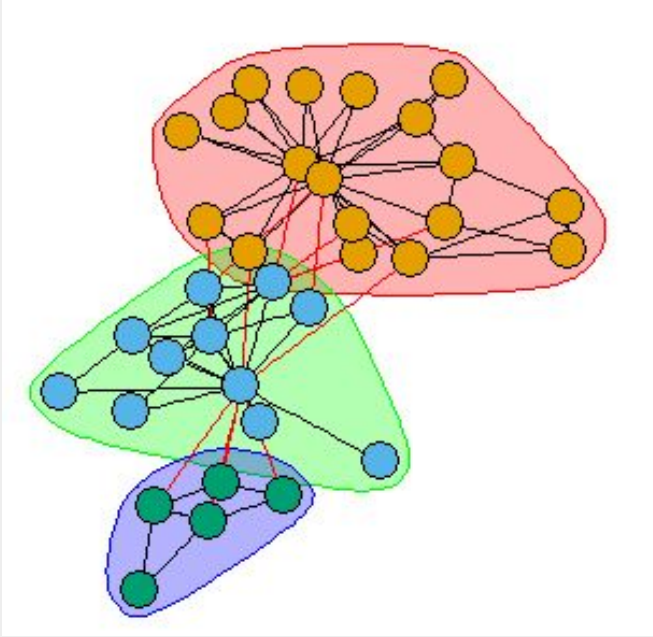
Average network degree versus vertex degree

```
a.nn.deg.net.karate.tidied <- graph.knn(net.karate.tidied,
    V(net.karate.tidied))$knn
plot(d.net.karate.tidied, a.nn.deg.net.karate.tidied,
    log="xy", col="goldenrod",
    xlab=c("Log Vertex Degree"),
    ylab=c("Log Average Neighbor Degree"),
    main =
    "Average network degree versus degree (log-log scale)")
```



Visualization

```
kc <- fastgreedy.community(net.karate.tidied)
plot(kc, net.karate.tidied, vertex.label = NA)
```



THANKS!