Decoding Connection

Workshop in Network Data Analysis

Beatrice Franzolini

July 08, 2025

Introduction

In this lab, we will analyze a social network using R. We will cover basic network **descriptive statistics**, compute **centrality measures** (degree, betweenness, closeness) and **transitivity measures** (local and global), and then estimate a **stochastic block models (SBM)** for community detection.

We will use the **igraph** package for core network analysis, **blockmodels** to estimate SBM, and the **igraphdata** package to load a sample network dataset.

The network we analyze is the celebrated Zachary's **Karate Club** network, originally documented by Wayne W. Zachary in 1977. It captures friendship ties among 34 members of a university karate club observed from 1970–1972, where an edge between two members indicates they consistently interacted outside of formal club meetings and practices. During the study period, a dispute between the club president ("John A.") and the instructor ("Mr. Hi") escalated, ultimately causing the club to split into two factions.

For more details, see: https://en.wikipedia.org/wiki/Zachary%27s_karate_club

Data Source: Mark Newman, Network Data http://www-personal.umich.edu/~mejn/netdata/.

Reference W. W. Zachary, An information flow model for conflict and fission in small groups, Journal of Anthropological Research 33, 452-473 (1977).

Packages Needed: `igraph`, `igraphdata`, `blockmodels` and `LaplacesDemon`. Install them if not already installed:

```
# Install required packages if not already installed
packages <- c("igraph", "igraphdata", "blockmodels", "LaplacesDemon")
for(pkg in packages) {
  if(!requireNamespace(pkg, quietly = TRUE)) {
    install.packages(pkg)
  }
}
```

```
library(igraph)
library(igraphdata)
data("karate")
g = karate
```

1. Basic Network Descriptive Statistics

Let's compute some measures of structure and size

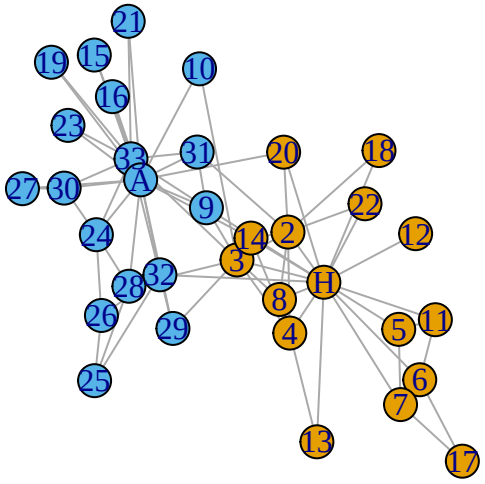
```
#nodes labels
V(g)
```

```
## + 34/34 vertices, named, from 4b458a1:
##  [1] Mr Hi    Actor 2  Actor 3  Actor 4  Actor 5  Actor 6  Actor 7  Actor 8
##  [9] Actor 9  Actor 10 Actor 11 Actor 12 Actor 13 Actor 14 Actor 15 Actor 16
## [17] Actor 17 Actor 18 Actor 19 Actor 20 Actor 21 Actor 22 Actor 23 Actor 24
## [25] Actor 25 Actor 26 Actor 27 Actor 28 Actor 29 Actor 30 Actor 31 Actor 32
## [33] Actor 33 John A
```

```
is_directed(g)
```

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

```
set.seed(0)
plot(g)
```




```
num_of_nodes = vcount(g) #num of nodes
num_of_edges = ecount(g) #num of edges
```

```
#network density:
edge_density(g)
```

```
## [1] 0.1390374
```

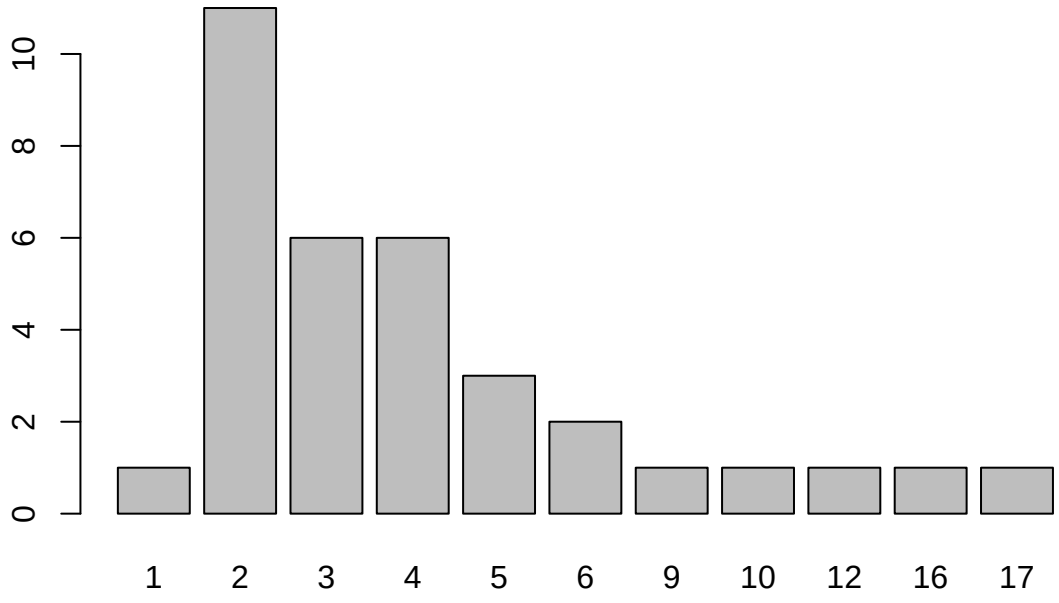
```
num_of_edges / choose(num_of_nodes, 2)
```

```
## [1] 0.1390374
```

```
#degree distribution:
deg = degree(g)
#degree(g, mode = "in")  or degree(g, mode = "out")
summary(deg)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   4.588   5.000  17.000
```

```
degree_dist = table(deg)
barplot(degree_dist)
```



2. Centrality Measures

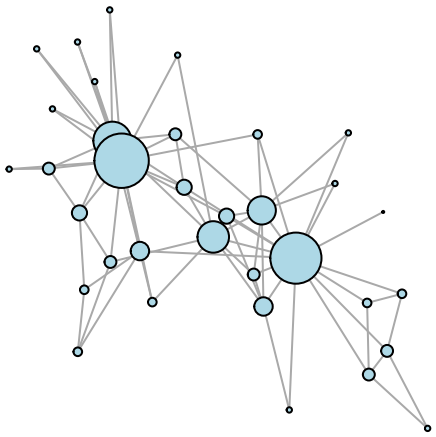
Let's investigate centrality to detected most influential nodes

Degree Centrality

```
deg = degree(g) #already computed
order(deg, decreasing=TRUE)[1:5]
```

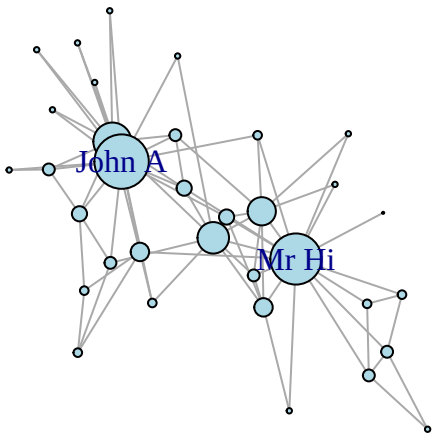
```
## [1] 34  1 33  3  2
```

```
V(g)$degree = deg

set.seed(0)
plot(g,
     vertex.size  = ((V(g)$degree - min(V(g)$degree)) /
                        (max(V(g)$degree) - min(V(g)$degree)) *25 + 1),
     vertex.label = NA,
     vertex.color = "lightblue",
     edge.arrow.size   = 0.3,
     main = "Node Size proportional to Degree")
```




```
#same plot with the label for John A and Mr Hi
lbls = rep(NA_character_, vcount(g))
lbls[c(1, 34)] = V(g)$name[c(1, 34)]

set.seed(0)
```

```
plot(g,
     vertex.size  = ((V(g)$degree - min(V(g)$degree)) /
                       (max(V(g)$degree) - min(V(g)$degree)) *25 + 1),
     vertex.label = lbls,
     vertex.color = "lightblue",
     edge.arrow.size   = 0.3,
     main = "Node Size proportional to Degree")
```



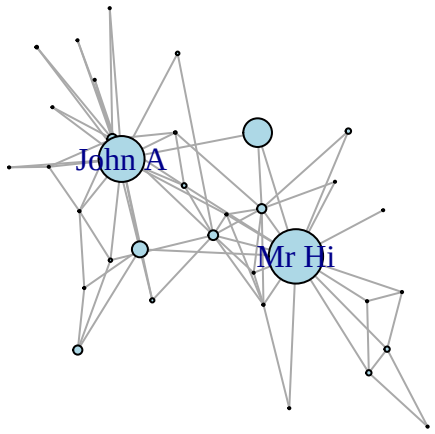
Betweenness Centrality

```
bet = betweenness(g)
order(bet, decreasing=TRUE)[1:5]
```

```
## [1]  1 34 20 32 33
```

```
V(g)$betweenness = bet

set.seed(0)
plot(g,
     vertex.size  = ((V(g)$betweenness - min(V(g)$betweenness)) /
                       (max(V(g)$betweenness) - min(V(g)$betweenness)) *25 + 1),
     vertex.label = lbls,
```

```
     vertex.color = "lightblue",
     edge.arrow.size   = 0.3,
     main = "Node Size proportional to Betweenness")
```



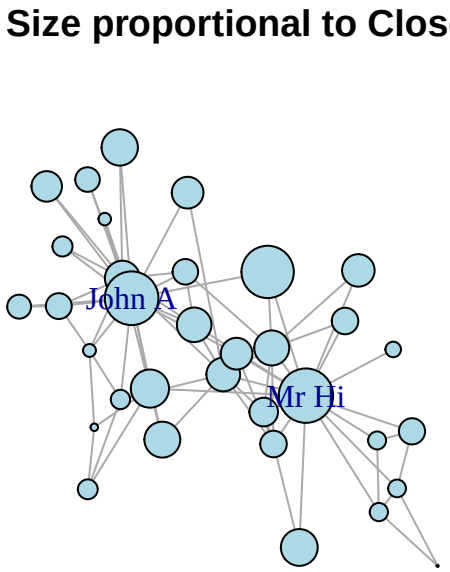
Closeness Centrality

```
clo = closeness(g, mode = "total")
order(clo, decreasing=TRUE)[1:5]
```

```
## [1]  1 34 20 32 13
```

```
#node_with_na_closeness = which(is.na(clo))
#neighbors(g, node_with_na_closeness)
#clo[node_with_na_closeness] = 0

V(g)$closeness = clo

set.seed(0)
plot(g,
     vertex.size  = ((V(g)$closeness - min(V(g)$closeness)) /
                       (max(V(g)$closeness) - min(V(g)$closeness)) *25 + 1),
     vertex.label = lbls,
```

```
     vertex.color = "lightblue",
     edge.arrow.size   = 0.3,
     main = "Node Size proportional to Closeness")
```



3. Transitivity (Clustering)

The transitivity or clustering coefficient gives a sense of how locally "dense" the network is—i.e., whether a friend of your friend is also your friend.

We will compute both the **global transitivity** (overall triangle density) and **local transitivity** (per-node clustering).

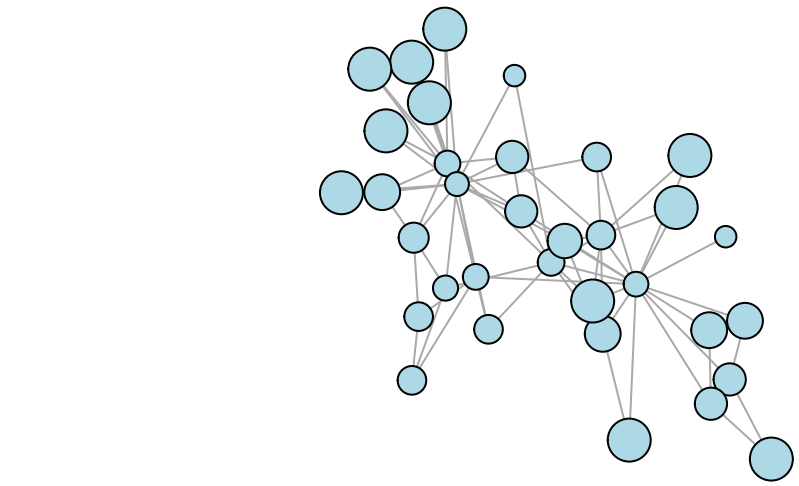
```
# Local clustering coefficient for each node
local_clustering = transitivity(g, type = "local")
summary(local_clustering)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##  0.0000  0.3333  0.5000  0.5879  1.0000  1.0000       1
```

```
node_with_na_clustering = which(is.na(local_clustering))
neighbors(g, node_with_na_clustering)
```

```
## + 1/34 vertex, named, from 4b458a1:
## [1] Mr Hi
```

```
local_clustering[node_with_na_clustering] = 0

# Visualize: Node size proportional to local clustering coefficient
V(g)$local_clust = local_clustering
set.seed(0)
plot(g,
     vertex.size = ((V(g)$local_clust - min(V(g)$local_clust, na.rm=TRUE)) /
                    (max(V(g)$local_clust, na.rm=TRUE) - min(V(g)$local_clust, na.rm=TRUE)) + 1)*10,
     vertex.color = "lightblue",
     vertex.label = NA,
     edge.arrow.size   = 0.3,
     main = "Node Size proportional to Local Clustering Coefficient")
```




```
# Average clustering coefficient
mean(local_clustering)
```

```
## [1] 0.5706385
```

```
# Global clustering coefficient: proportion of closed triplets
transitivity(g, type = "global")
```

```
## [1] 0.2556818
```

The **local clustering coefficient** measures how tightly connected each node's neighbors are, and its average (here 0.57) reflects how "cliquish" a typical node's neighborhood is. In contrast, the **global clustering**

coefficient (0.26) measures the overall proportion of closed triplets (triangles) in the entire network. The fact that the global value is much lower suggests that, although many nodes are embedded in tightly connected local groups, the network as a whole is not densely interconnected. This is a common feature in social networks, where individuals often form small, highly clustered communities, but connections across communities are sparser.

Summing up so far

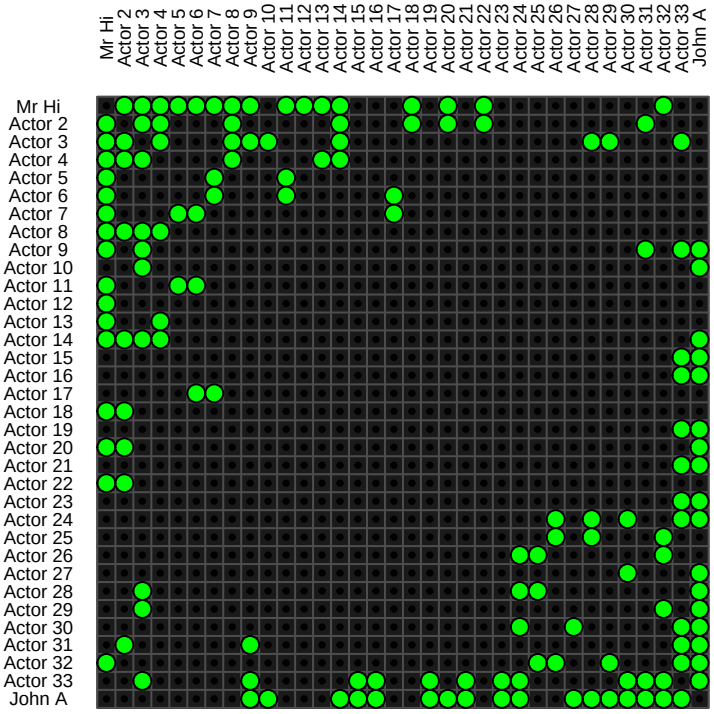
Through our descriptive analysis of Zachary's Karate Club network, we gained insight into both the overall structure and the roles of individual members. The network is moderately sparse, with a density of about 14%, and its degree distribution is right-skewed, indicating that a few individuals are significantly more connected than others. Centrality measures revealed that nodes 1 and 34 are the most central according to degree, betweenness, and closeness—consistent with their known leadership roles in the real-world split of the club. The clustering analysis showed a high average local clustering coefficient (~0.57), indicating that individuals tend to form tightly connected local groups. However, the lower global clustering coefficient (~0.26) suggests that the network is not globally cohesive, but rather made up of overlapping communities.

4. Stochastic Block Models (SBM)

```
library(blockmodels)
library(LaplacesDemon) #to plot adj matrix
A = as.matrix(as_adjacency_matrix(g))
isSymmetric(A)
```

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

```
plotMatrix(A)
```

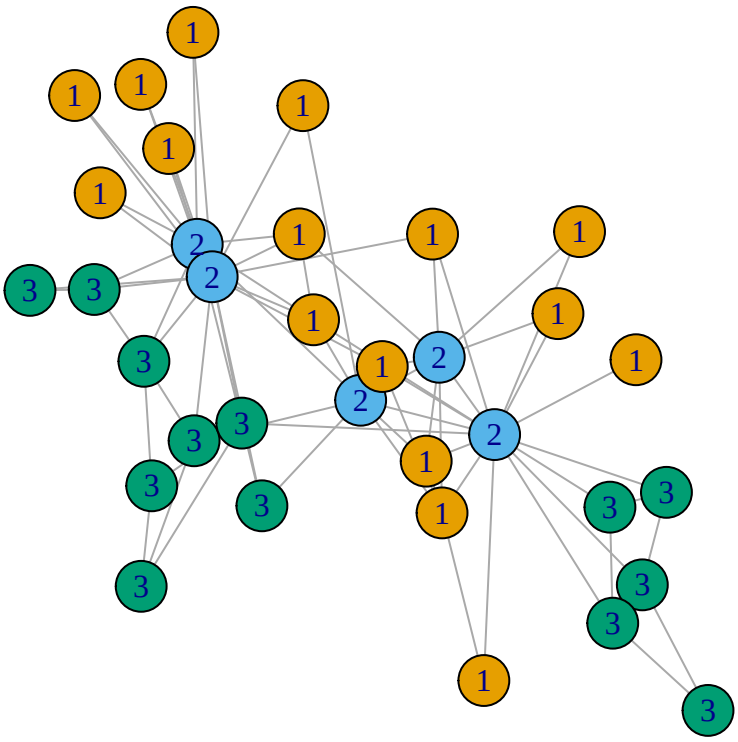



```
sbm_model = BM_bernoulli("SBM", A, verbosity = 0, plotting = "")
sbm_model$estimate()
```

```
K_opt = which.max(sbm_model$ICL)

Z = sbm_model$memberships[[K_opt]]$Z
block_assignments = apply(Z, 1, which.max)
V(g)$block = block_assignments
table(V(g)$block)
```

```
##
##  1  2  3
## 16  5 13
```

```
set.seed(0)
plot(g, vertex.color=V(g)$block, vertex.label=V(g)$block,
     edge.arrow.size   = 0.3,
     main=paste("SBM Detected", K_opt, "Blocks"))
```




```
sbm_model$model_parameters[[K_opt]]$pi
```

```
##              [,1]      [,2]        [,3]
## [1,] 0.034331940 0.4679185 0.001297862
## [2,] 0.467918523 0.4946149 0.244698939
## [3,] 0.001297862 0.2446989 0.188699680
```

Conclusion

In this lab, we have analyzed the Karate network, computed centrality and transitivity measures, and estimated community structure with stochastic block models.

You can now try yourself with the following network

`UKfaculty` = Friendship network of a UK university faculty (directed!).

Edges arise from a survey asking each faculty member to list colleagues they consider personal friends. An edge i to j exists if i named j, regardless of whether j named i

```
data("UKfaculty")
g = UKfaculty

#....
```