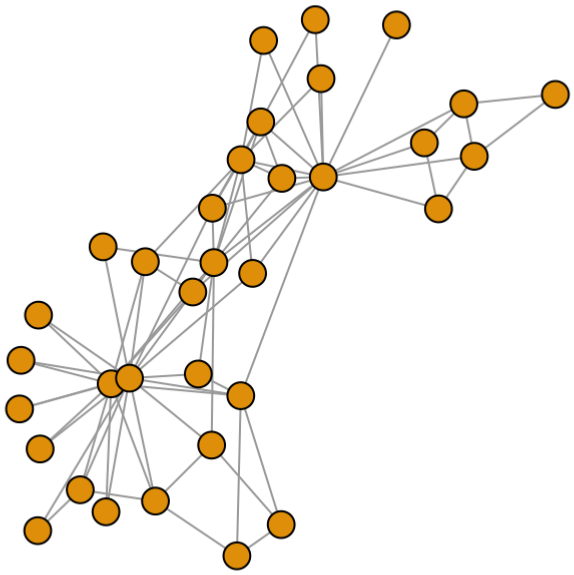
Communities in igraph

Massimo Franceschet

We work with a social network of friendships between 34 members of a karate club at a US university in the 1970s.

```
library(igraph)
library(lsa)
```

```
g = make_graph("Zachary")
coords = layout_with_fr(g)
# plot the graph
plot(g, layout=coords, vertex.label=NA, vertex.size=10)
```



Greedy community detection

```
# greedy method (hiearchical, fast method)
c1 = cluster_fast_greedy(g)

# modularity measure
modularity(c1)
```

```
## [1] 0.3806706
```

```
# modularity matrix
B = modularity_matrix(g, membership(c1))
round(B[1,],2)
```

```
##  [1] -1.64  0.08 -0.03  0.38  0.69  0.59  0.59  0.59  0.49 -0.21  0.69
## [12]  0.90  0.79  0.49 -0.21 -0.21 -0.21  0.79 -0.21  0.69 -0.21  0.79
## [23] -0.21 -0.51 -0.31 -0.31 -0.21 -0.41 -0.31 -0.41 -0.41  0.38 -1.23
## [34] -1.74
```

```
# memberships of nodes
membership(c1)
```

```
##  [1] 1 3 3 3 1 1 1 3 2 3 1 1 3 3 2 2 1 3 2 1 2 3 2 2 2 2 2 2 2 2 2 2 2 2
```

```
# number of communities
length(c1)
```

```
## [1] 3
```

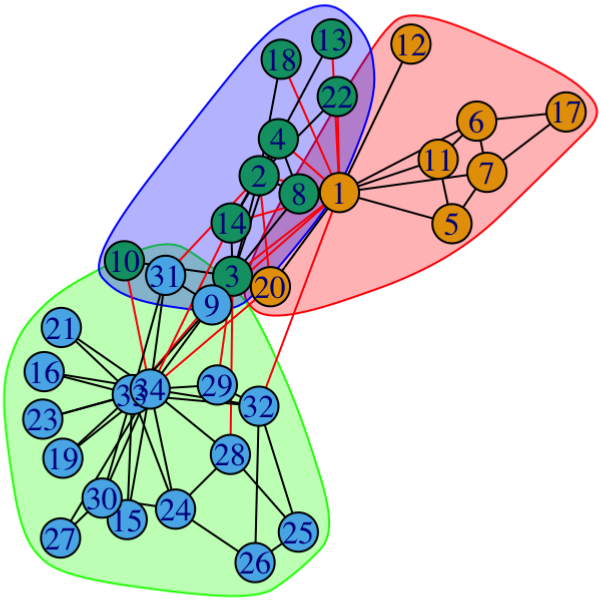
```
# size of communities
sizes(c1)
```

```
## Community sizes
##  1  2  3
##  8 17  9
```

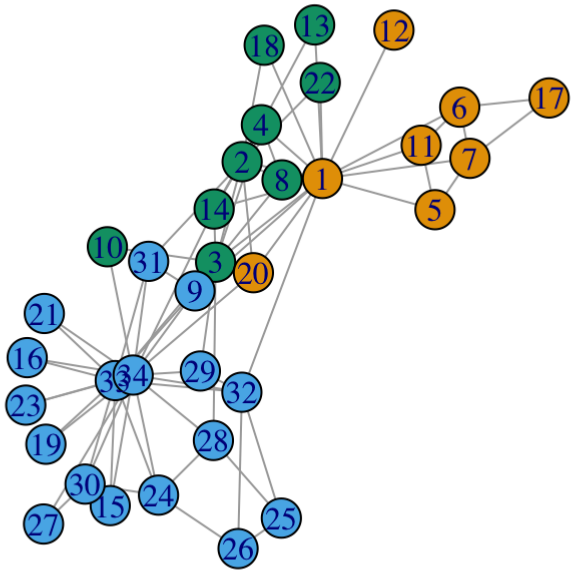
```
# crossing edges
crossing(c1, g)
```

```
##  [1]  TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE  TRUE
## [12]  TRUE  TRUE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE
## [23] FALSE  TRUE FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE FALSE FALSE
## [34] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [45]  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## [56] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [67] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [78] FALSE
```

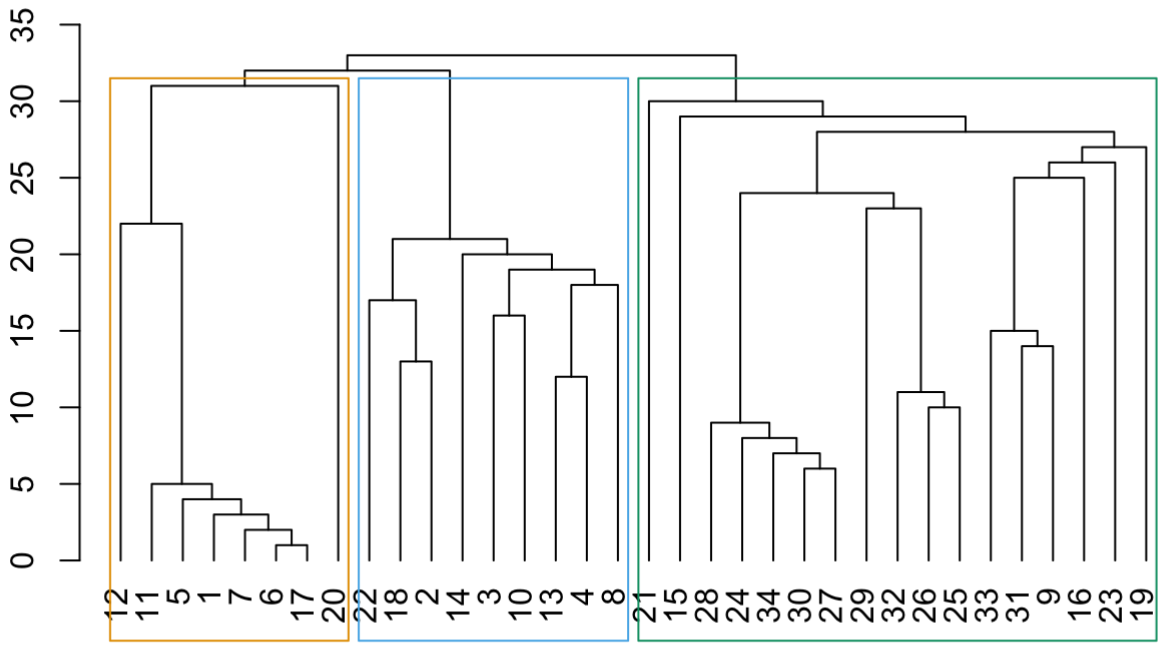
```
# plot communities with shaded regions
plot(c1, g, layout=coords)
```




```
# plot communities without shaded regions
plot(g, vertex.color=membership(c1), layout=coords)
```



```
# plot dendogram
plot_dendrogram(c1)
```

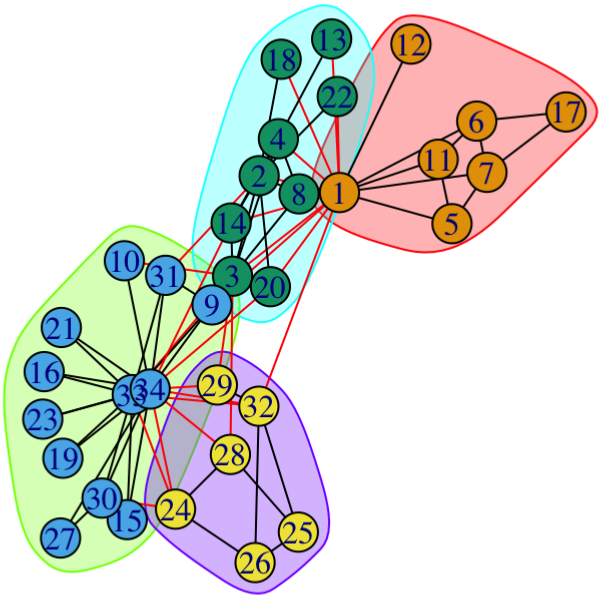


Spectral community detection

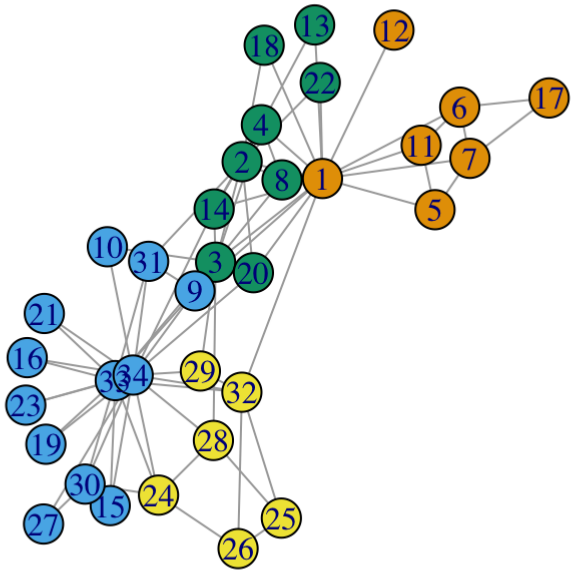
```
# greedy method (hiearchical, fast method)
c2 = cluster_leading_eigen(g)

# modularity measure
modularity(c2)
```

```
## [1] 0.3934089
```

```
# plot communities with shaded regions
plot(c2, g, layout=coords)
```




```
# plot communities without shaded regions
plot(g, vertex.color=membership(c2), layout=coords)
```

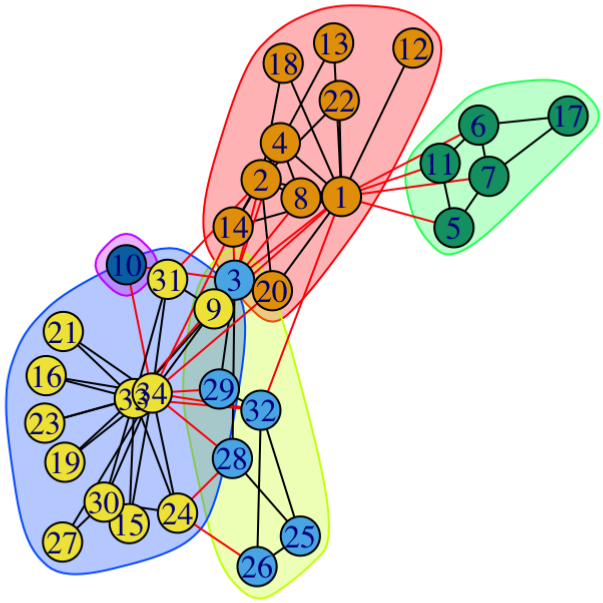


Betweenness community detection

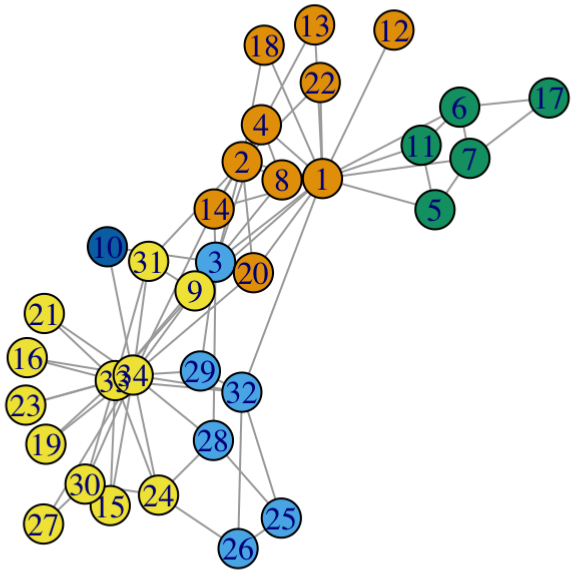
```
# greedy method (hiearchical, fast method)
c3 = cluster_edge_betweenness(g)

# modularity measure
modularity(c3)
```

```
## [1] 0.4012985
```

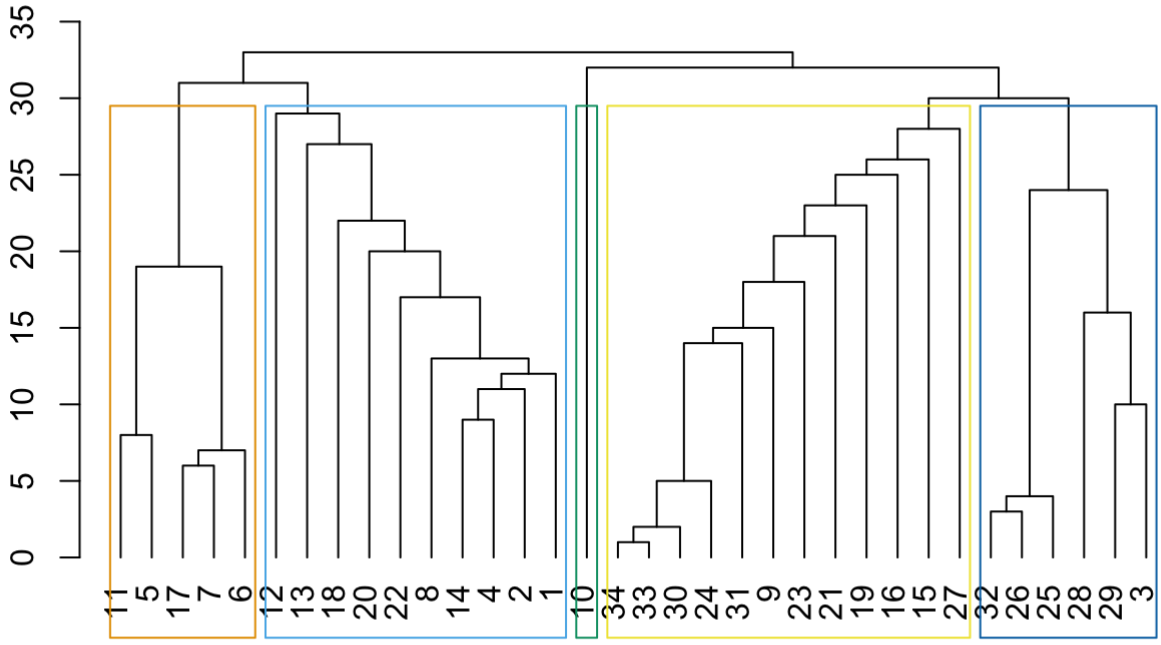
```
# plot communities with shaded regions
plot(c3, g, layout=coords)
```




```
# plot communities without shaded regions
plot(g, vertex.color=membership(c3), layout=coords)
```

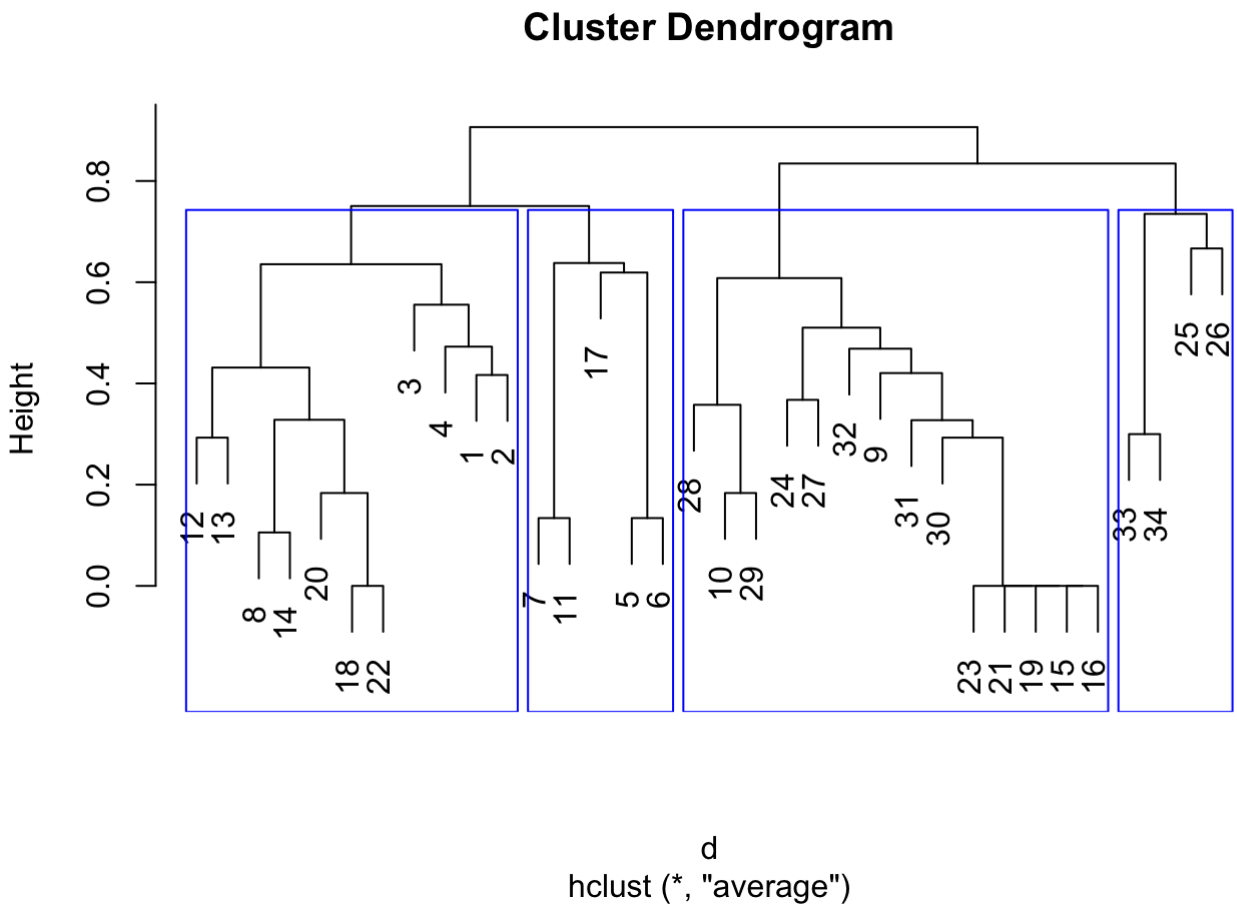



```
# plot dendogram
plot_dendrogram(c3)
```

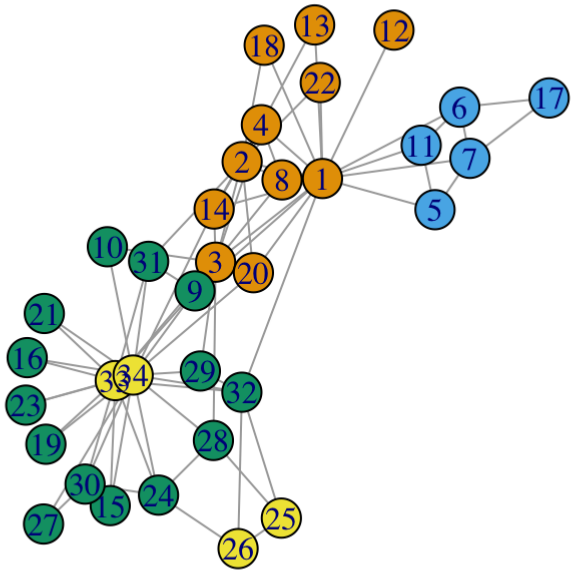


Hiearchical clustering

```
# hierarchical clustering
A = get.adjacency(g, sparse=FALSE)

# cosine similarity
S = cosine(A)

# distance matrix
D = 1-S

# distance object
d = as.dist(D)

# average-linkage clustering method
cc = hclust(d, method = "average")

# plot dendrogram
plot(cc)

# draw blue borders around clusters
clusters.list = rect.hclust(cc, k = 4, border="blue")
```



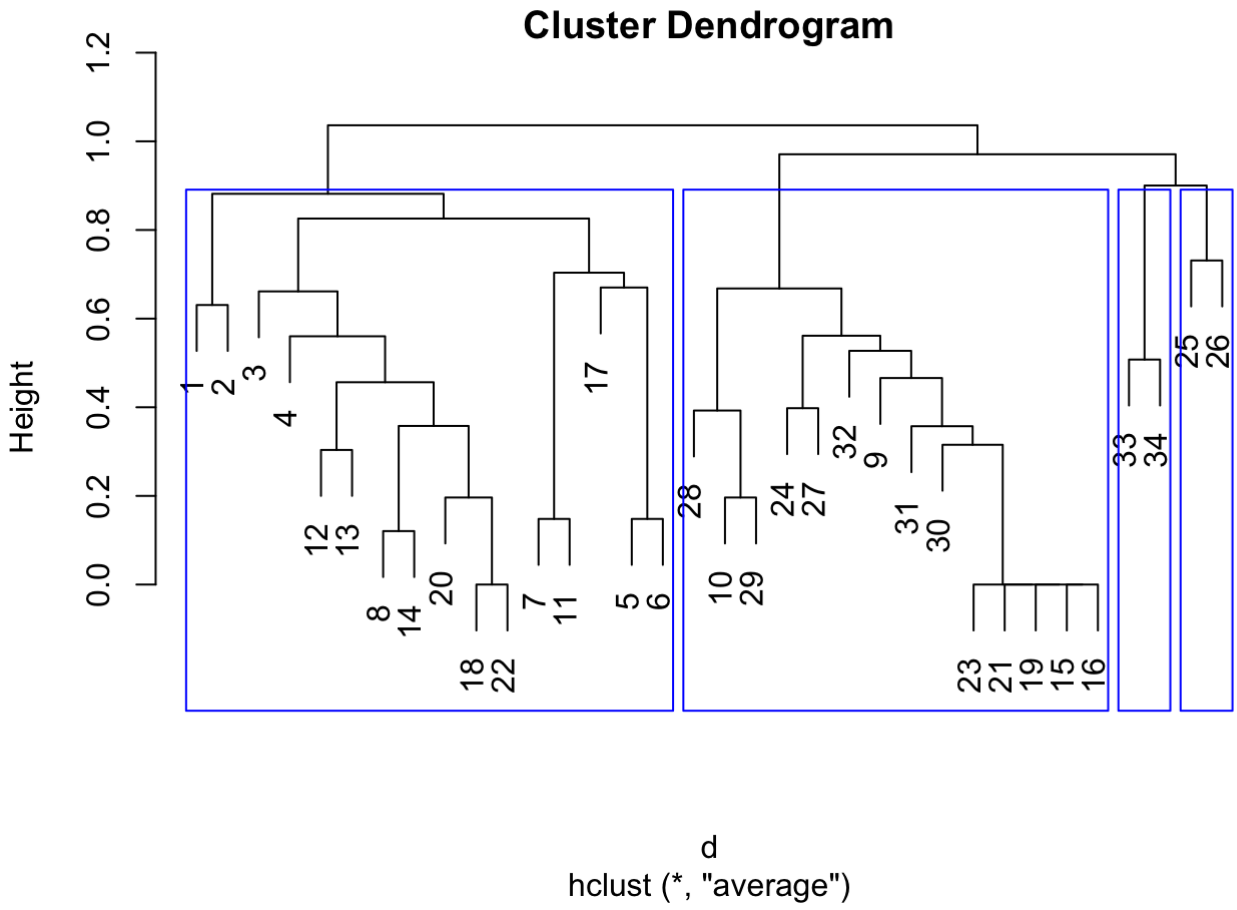

```
# cut dendrogram at 4 clusters
clusters = cutree(cc, k = 4)

# plot graph with clusters
plot(g, vertex.color=clusters, layout=coords)
```



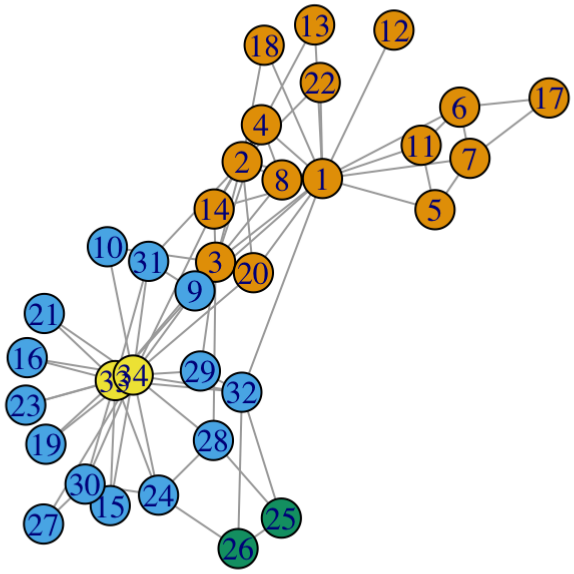

```
# modularity
m1 = modularity(g, clusters)
m1
```

```
## [1] 0.1695431
```

```
# Pearson similarity
S = cor(A, method="pearson")

# distance matrix
D = 1-S

# distance object
d = as.dist(D)

# average-linkage clustering method
cc = hclust(d, method = "average")

# plot dendrogram
plot(cc)

# draw blue borders around clusters
clusters.list = rect.hclust(cc, k = 4, border="blue")
```



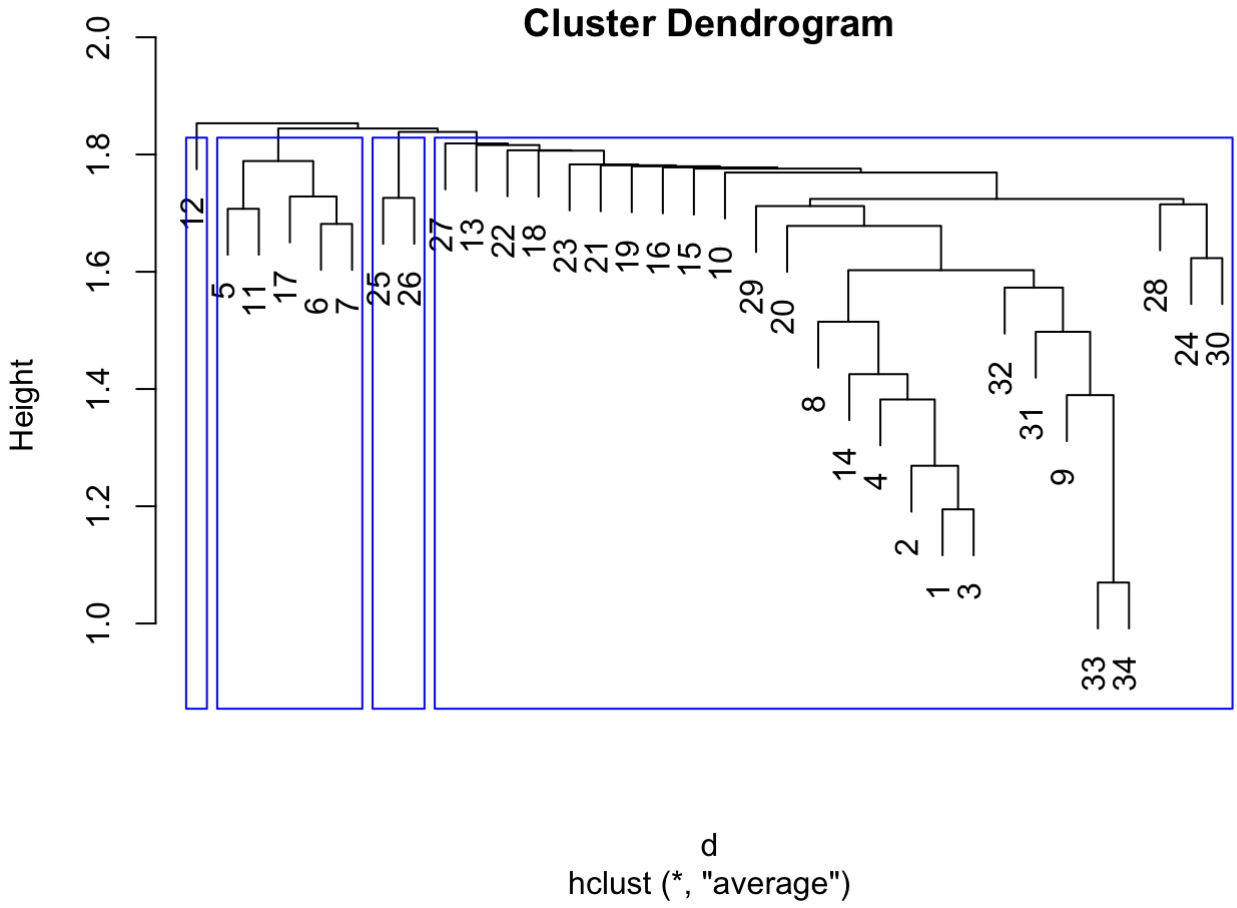

```
# cut dendrogram at 4 clusters
clusters = cutree(cc, k = 4)

# plot graph with clusters
plot(g, vertex.color=clusters, layout=coords)
```



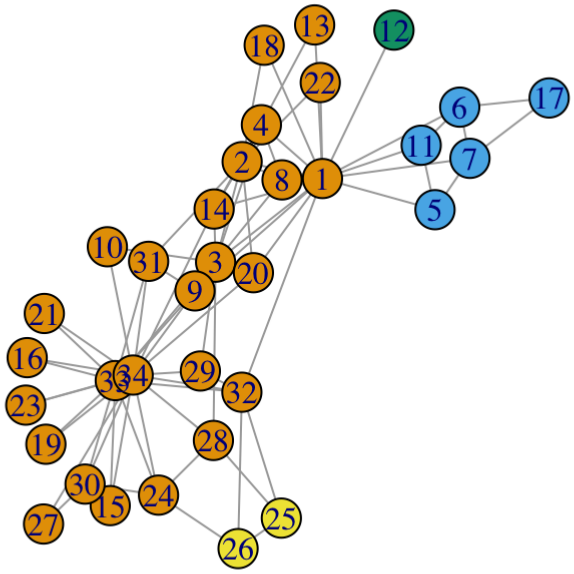

```
# modularity
m2 = modularity(g, clusters)
m2
```

```
## [1] 0.1562295
```

```
# global similarity
I = diag(1, vcount(g));

# leading eigenvalue
l = eigen(A)$values[1]

# 85% of leading eigenvalue
alpha = 0.85 * 1/l

# similarity matrix
S = solve(I - alpha * A)

# largest value
max = max(as.vector(S))

# distance matrix
D = max-S

# set null distance from a node to itself
d = diag(D)
D = D + diag(-d, vcount(g))


# distance object
d = as.dist(D)

# average-linkage clustering method
cc = hclust(d, method = "average")

# plot dendrogram
plot(cc)

# draw blue borders around clusters
clusters.list = rect.hclust(cc, k = 4, border="blue")
```




```
# cut dendrogram
clusters = cutree(cc, k = 4)

# plot graph with clusters
plot(g, vertex.color=clusters, layout=coords)
```




```
# modularity
m3 = modularity(g, clusters)
m3
```

```
## [1] 0.1457101
```

Optimal community detection

```
# greedy method (hiearchical, fast method)
c4 = cluster_optimal(g)

# modularity measure
modularity(c4)
```

```
## [1] 0.4197896
```

```
# plot communities with shaded regions
plot(c4, g, layout=coords)
```