HW1_Jerome_Wong

Contents

```
library(sna)
```

```
## Loading required package: statnet.common

##
## Attaching package: 'statnet.common'

## The following objects are masked from 'package:base':
##
##     attr, order

## Loading required package: network

##
## 'network' 1.17.1 (2021-06-12), part of the Statnet Project
## * 'news(package="network")' for changes since last version
## * 'citation("network")' for citation information
## * 'https://statnet.org' for help, support, and other information

## sna: Tools for Social Network Analysis
## Version 2.6 created on 2020-10-5.
## copyright (c) 2005, Carter T. Butts, University of California-Irvine
##  For citation information, type citation("sna").
##  Type help(package="sna") to get started.
```

```
library(igraph)
```

```
##
## Attaching package: 'igraph'

## The following objects are masked from 'package:sna':
##
##     betweenness, bonpow, closeness, components, degree, dyad.census,
##     evcent, hierarchy, is.connected, neighborhood, triad.census

## The following objects are masked from 'package:network':
##
##     %c%, %s%, add.edges, add.vertices, delete.edges, delete.vertices,
##     get.edge.attribute, get.edges, get.vertex.attribute, is.bipartite,
##     is.directed, list.edge.attributes, list.vertex.attributes,
##     set.edge.attribute, set.vertex.attribute
```

```
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union
```

```r
library(igraphdata)
knitr::opts_chunk$set(echo = TRUE)
data("UKfaculty")
```

```r
pacakge_list <- c("igraph", "igraphdata")
new_packages <- pacakge_list[!(pacakge_list %in% installed.packages()[,"Package"])]
if(length(new_packages)) install.packages(new_packages)
```

1. Data description and visualization

- In a sentence, what does this data set represent?

```r
?UKfaculty
```

It represents the personal friendship network of a faculty of a UK university.

- What paper, book, or other resource is the source of this data set? It comes from: "Nepusz T., Petroczi A., Negyessy L., Bazso F.: Fuzzy communities and the concept of bridgeness in complex networks. Physical Review E 77:016107, 2008."

- How many vertices and edges does the data set have?

```r
V(UKfaculty)
```

```
## + 81/81 vertices, from 6f42903:
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
## [26] 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
## [51] 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75
## [76] 76 77 78 79 80 81
```

```r
E(UKfaculty)
```

```
## + 817/817 edges from 6f42903:
##   [1] 57->52 76->42 12->69 43->34 28->47 58->51  7->29 40->71  5->37 48->55
##  [11]  6->58 21-> 8 28->69 43->21 67->58 65->42  5->67 52->75 37->64  4->36
##  [21] 12->49 19->46 37-> 9 74->36 62-> 1 15-> 2 72->49 46->62  2->29 40->12
##  [31] 22->29 71->69  4-> 3 37->69  5-> 6 77->13 23->49 52->35 20->14 62->70
##  [41] 34->35 76->72  7->42 37->42 51->80 38->45 62->64 36->53 62->77 17->61
##  [51]  7->68 46->29 44->53 18->58 12->16 72->42 52->32 58->21 38->17 15->51
##  [61] 22-> 7 22->69  5->13 29-> 2 77->12 37->35 18->46 10->71 22->47 20->19
##  [71] 19->31 68->13 49->69 30->63  5->49 53->75 62->57 73->81 29->69 71->40
##  [81] 19->58 49->42 37-> 5 18-> 2 20->80 75->53 15->54 76->58 40->23  5->12
##  [91] 20->54  6->47 51->14 78-> 4 52->49 29->55 27->35 66-> 6 21->29  4->61
## + ... omitted several edges
```

There are 81 vertices and 817 edges

- Is this graph directed or undirected? What other properties can you comment on?

```
is_directed(UKfaculty)
```

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

```
is_weighted(UKfaculty)
```

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

```
is_simple(UKfaculty)
```

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

```
is_connected(UKfaculty)
```

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

The UKfaculty graph is directed, weighted, simple and connected.

- Describe each of the attributes included in this data set. What are they called, what format/type do they have, what are they attributes of (vertices, edges, etc), what do they mean?

```
get_igraph_attrs <- function(igraph){
  if(!is_igraph(igraph)) stop("Not a graph object")
  list(graph_attrs = list.graph.attributes(igraph),
       vertex_attrs = list.vertex.attributes(igraph),
       edge_attrs = list.edge.attributes(igraph))
}

get_igraph_attrs(UKfaculty)
```

```
## $graph_attrs
## [1] "Type"     "Date"     "Citation" "Author"
##
## $vertex_attrs
## [1] "Group"
##
## $edge_attrs
## [1] "weight"
```

```
typeof(V(UKfaculty)$Group)
```

```
## [1] "double"
```

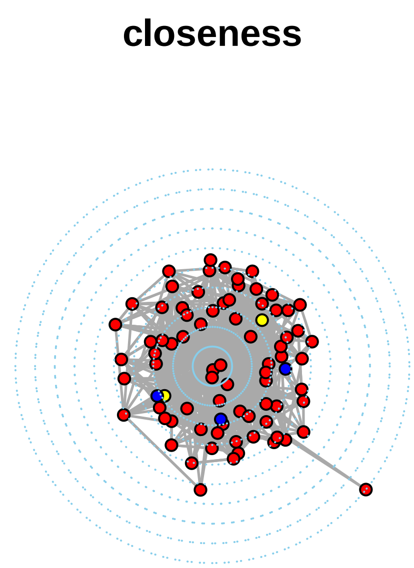
```
typeof(E(UKfaculty)$weight)
```

```
## [1] "double"
```

The vertex attribute "Graph" is in the datatype of "double" The edge attribute "weight" is in the datatype of "double

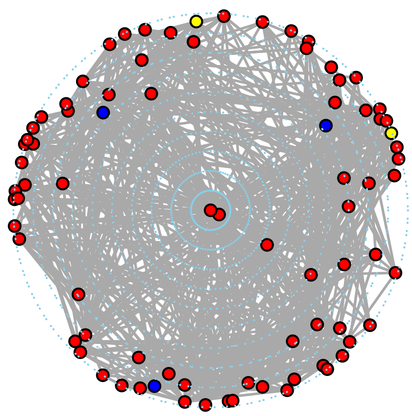
- Compute and comment on two different centrality measures for this data set.

```
A <- get.adjacency(UKfaculty, sparse=FALSE)
g <- as.network.matrix(A)
gplot.target(g, sna::closeness(g),
             main="closeness",
             circ.lab = FALSE, circ.col="skyblue",
             usearrows = FALSE,
             vertex.col=c("blue", rep("red", 32), "yellow"),
             edge.col="darkgray")
```




```
gplot.target(g, sna::betweenness(g),
             main="Betweenness",
             circ.lab = FALSE, circ.col="skyblue",
             usearrows = FALSE,
             vertex.col=c("blue", rep("red", 32), "yellow"),
             edge.col="darkgray")
```

Betweenness



Closeness Centrality is used to find individuals who are best placed to influence the network most quickly Betweeness Centrality is used to find individuals who influence flow around the network by acting as “bridges” between nodes

- Compute and briefly comment on some measure of assortativity for this data set.

```
assortativity.nominal(UKfaculty, V(UKfaculty)$Group, directed=FALSE)
```

```
## [1] 0.7052057
```

```
assortativity.degree(UKfaculty)
```

```
## [1] 0.03805201
```

A high and positive assortivity indicates that people in the different groups tend to associate more with people in the same faction A positive but low degree assortivity indicates that the tendency for nodes of high degree (resp. low degree) in the Ukfaculty to be connected to other high degree nodes is low.

- Compute/explore at least one other aspect of the data (see labs for ideas) and comment on it.

```
?clusters
is_connected(UKfaculty)
```

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

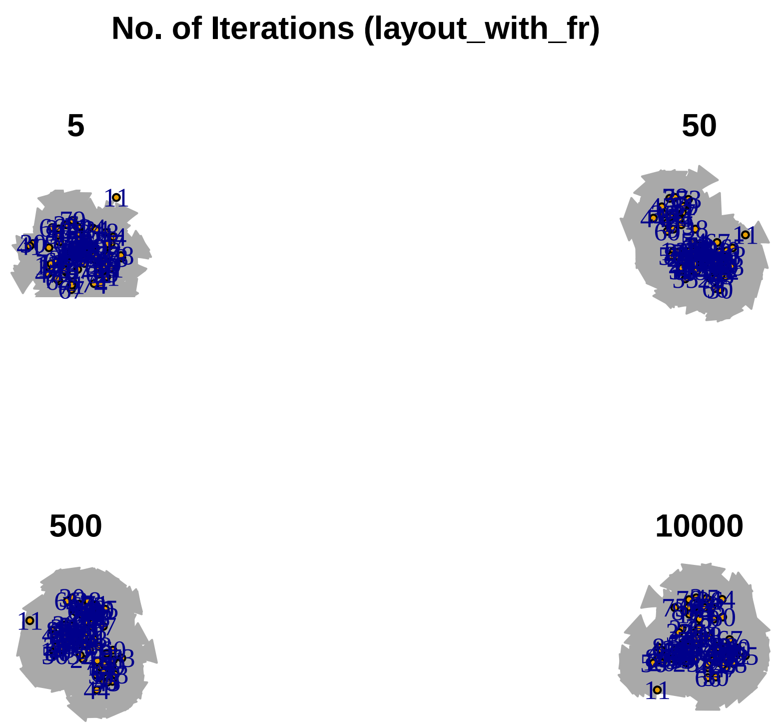
```
clusters(UKfaculty)
```

```
## $membership
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [39] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [77] 1 1 1 1 1
##
## $csize
## [1] 81
##
## $no
## [1] 1
```

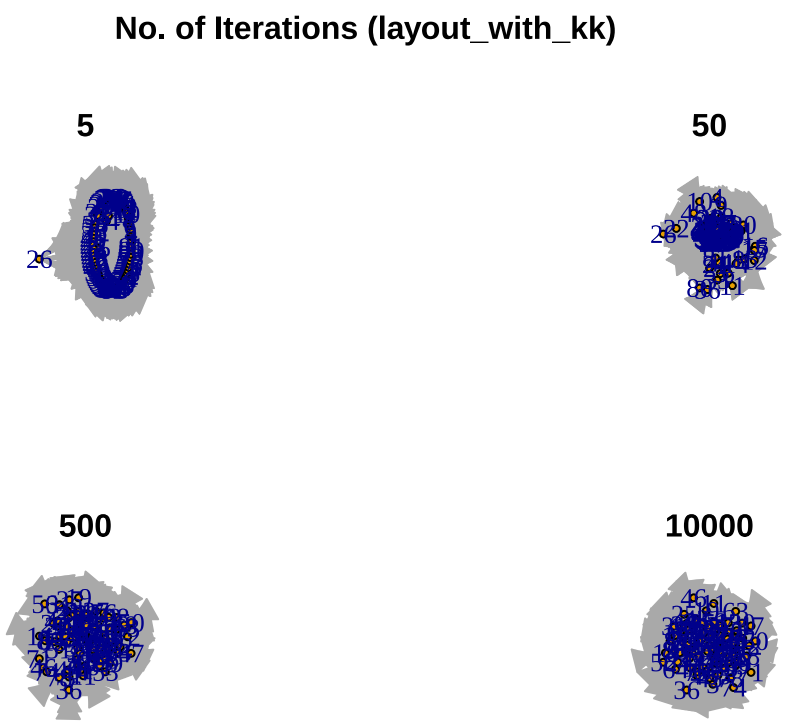
The network is connected. This implies that every member of the UKfaculty is reachable from the other. Therefore, it consists of only one component

Plot the network. Try a few layouts, then choose one you think is informative or appropriate (there is no particular right answer), and comment on the graph structure.

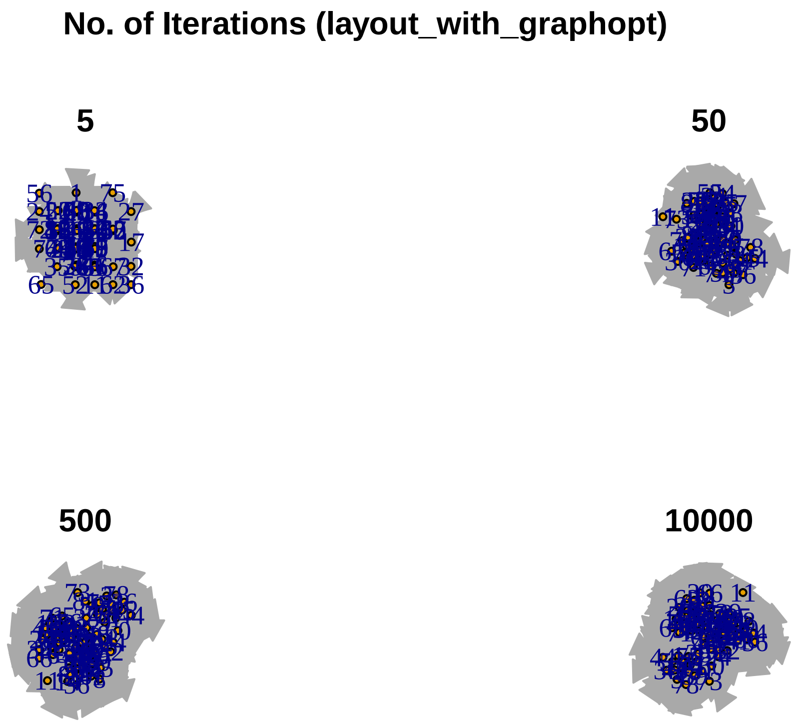
```
par(mfrow=c(2, 2))                            ## par(mfrow=c(2,2)) = 2 row, 2 columns
par(oma=c(0,0,2,0))                           ## adjust outer margin for title

## layout_with_fr
plot.igraph(UKfaculty, layout = layout_with_fr(UKfaculty, maxiter = 5), main = 5)
plot.igraph(UKfaculty, layout = layout_with_fr(UKfaculty, maxiter = 50), main = 50)
plot.igraph(UKfaculty, layout = layout_with_fr(UKfaculty, maxiter = 500), main = 500)
plot.igraph(UKfaculty, layout = layout_with_fr(UKfaculty, maxiter = 10000), main = 10000)
title("No. of Iterations (layout_with_fr)", outer = TRUE)
```




```r
## layout_with_kk
plot.igraph(UKfaculty, layout = layout_with_kk(UKfaculty, maxiter = 5), main = 5)
plot.igraph(UKfaculty, layout = layout_with_kk(UKfaculty, maxiter = 50), main = 50)
plot.igraph(UKfaculty, layout = layout_with_kk(UKfaculty, maxiter = 500), main = 500)
plot.igraph(UKfaculty, layout = layout_with_kk(UKfaculty, maxiter = 10000), main = 10000)
title("No. of Iterations (layout_with_kk)", outer = TRUE)
```



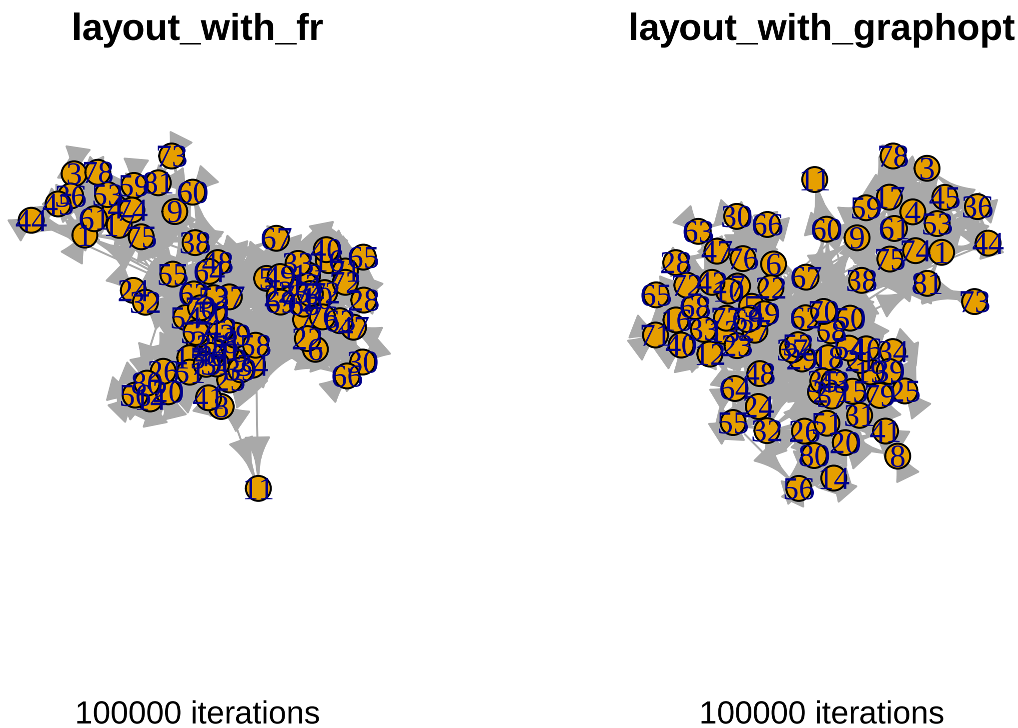

```r
## layout_with_graphopt
plot.igraph(UKfaculty, layout = layout_with_graphopt(UKfaculty, niter = 5), main = 5)
plot.igraph(UKfaculty, layout = layout_with_graphopt(UKfaculty, niter = 50), main = 50)
plot.igraph(UKfaculty, layout = layout_with_graphopt(UKfaculty, niter = 500), main = 500)
plot.igraph(UKfaculty, layout = layout_with_graphopt(UKfaculty, niter = 10000), main = 10000)
title("No. of Iterations (layout_with_graphopt)", outer = TRUE)
```




```
par(mfrow=c(1, 1))
```

layout_with_fr and layout_with_graphopt looks more promising than layout_with_kk. Let's compare those two:

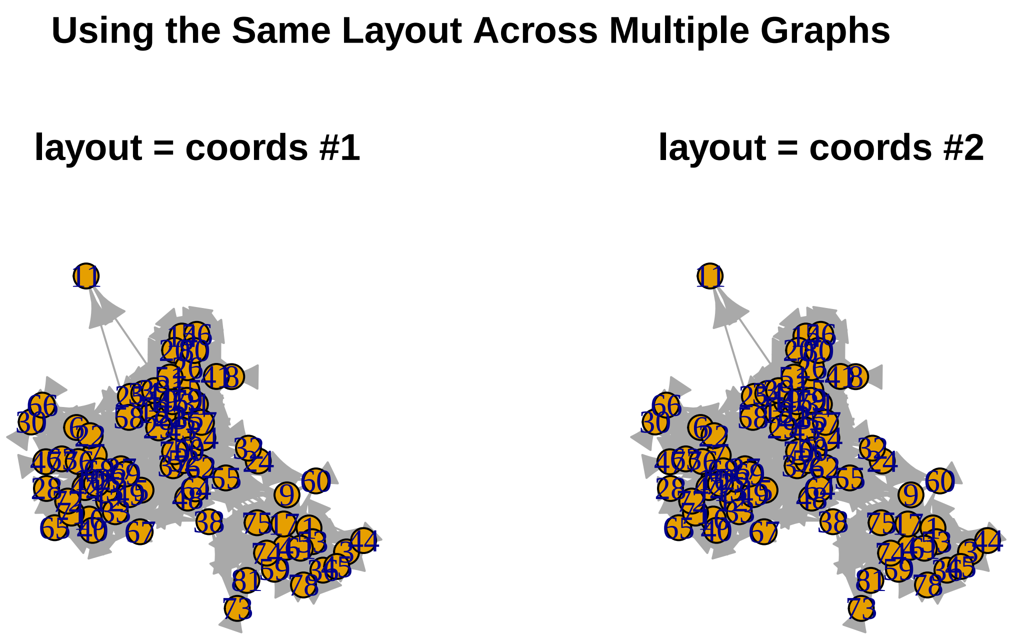
```
par(mfrow=c(1, 2))
par(oma=c(0,0,2,0))

plot.igraph(UKfaculty,
            layout = layout_with_fr(UKfaculty, maxiter = 100000),
            main = "layout_with_fr",
            sub = "100000 iterations")

plot.igraph(UKfaculty,
            layout = layout_with_graphopt(UKfaculty, niter = 100000),
            main = "layout_with_graphopt",
            sub = "100000 iterations")
```




```
par(mfrow=c(1, 1))
```

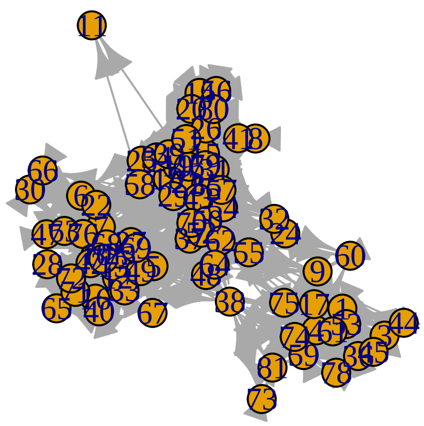
Looks like layout_with_fr have a more defined clustering than layout_with_graphopt.

```
coords <- layout_with_fr(UKfaculty, maxiter = 100000)
par(mfrow=c(1, 2))
par(oma=c(0,0,2,0))

plot.igraph(UKfaculty, layout = coords, main = "layout = coords #1")

plot.igraph(UKfaculty, layout = coords, main = "layout = coords #2")

title("Using the Same Layout Across Multiple Graphs", outer = TRUE)
```

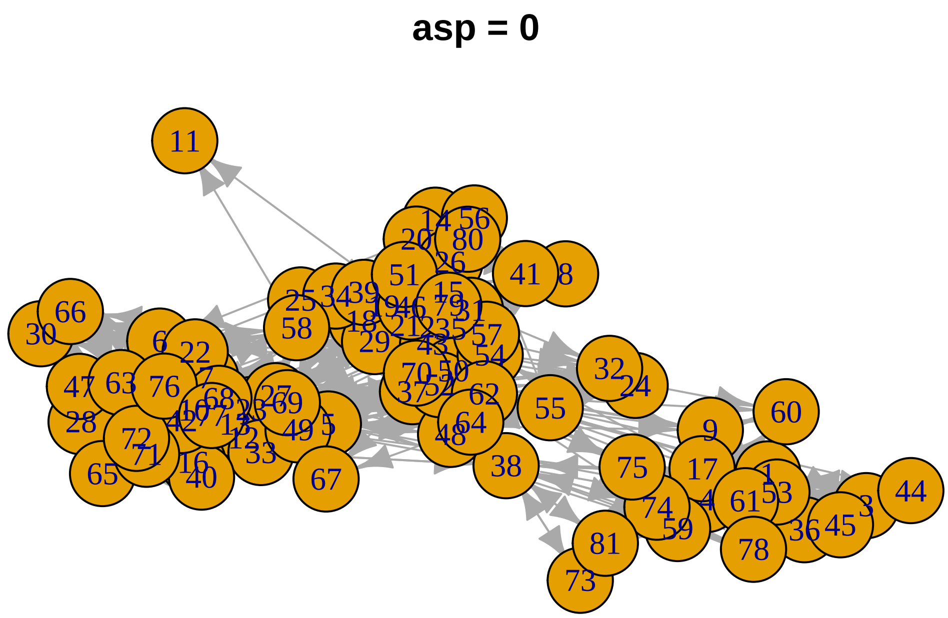



```
par(mfrow=c(1, 1))

plot.igraph(UKfaculty, layout = coords, main = "layout = coords (Single Plot)")
```

layout = coords (Single Plot)



```r
plot.igraph(UKfaculty, layout = coords, asp = 0, main = "asp = 0")
```



The graph has 3 distrinct clusters with node 11 being an outlier.

2. Graph partitioning

Pick two graph partitioning methods (including at least one spectral method), state which two methods you chose, and apply them to this data set. Do the results look similar? Would you draw different conclusions from one partitioning versus the other? Bonus: why might you prefer one over the other?

```
res.UKfaculty = leading.eigenvector.community(UKfaculty)
```

```
## Warning in leading.eigenvector.community(UKfaculty): At community.c:1617 :This
## method was developed for undirected graphs
```

```
length(res.UKfaculty)
```

```
## [1] 5
```

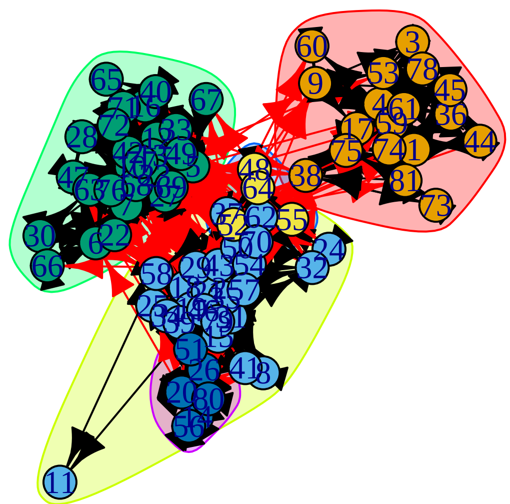
```
modularity(res.UKfaculty)
```

```
## [1] 0.5572471
```

```
membership(res.UKfaculty)
```

```
##  [1] 1 2 1 1 3 3 3 2 1 3 2 3 3 5 2 3 1 2 2 5 2 3 3 2 2 5 3 3 2 3 2 2 3 2 2 1 2 1
## [39] 2 3 2 3 2 1 1 2 3 4 3 2 5 4 1 2 4 5 2 2 1 1 1 2 3 4 3 3 3 3 3 2 3 3 1 1 1 3
## [77] 3 1 2 5 1
```

```
plot(res.UKfaculty, UKfaculty)
```

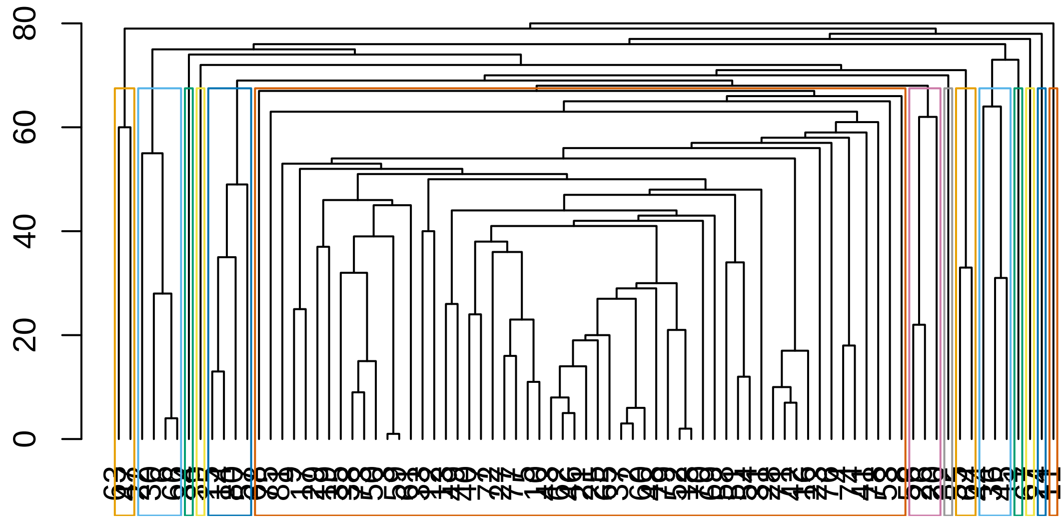


```
ceb <- cluster_edge_betweenness(UKfaculty)
```

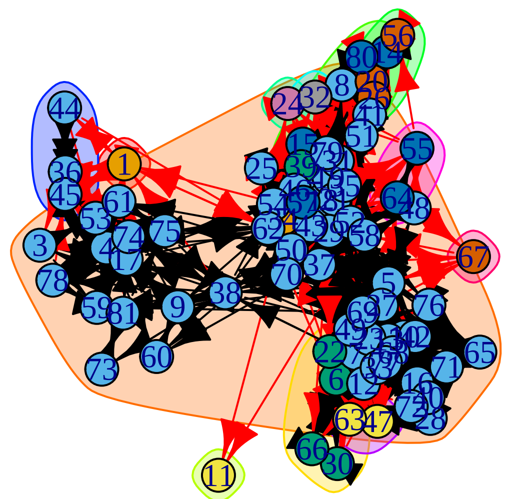
```
## Warning in cluster_edge_betweenness(UKfaculty): At community.c:461 :Membership
## vector will be selected based on the lowest modularity score.
```

```
## Warning in cluster_edge_betweenness(UKfaculty): At community.c:468 :Modularity
## calculation with weighted edge betweenness community detection might not make
## sense -- modularity treats edge weights as similarities while edge betwenness
## treats them as distances
```

```
dendPlot(ceb, mode="hclust")
```



```
plot(ceb, UKfaculty)
```



I used the spectral method and the Girvan–Newman algorithm. The spectral method gives a more defined clustering compared to Girvan-Newman and would be the better choice amongst the two. According to the spectral method, there exist 5 main clusters whereas in Girvan-Newman, there has singular clusters that exist only 1 node.