Network Analysis of KAPFERER MINE

Introduction

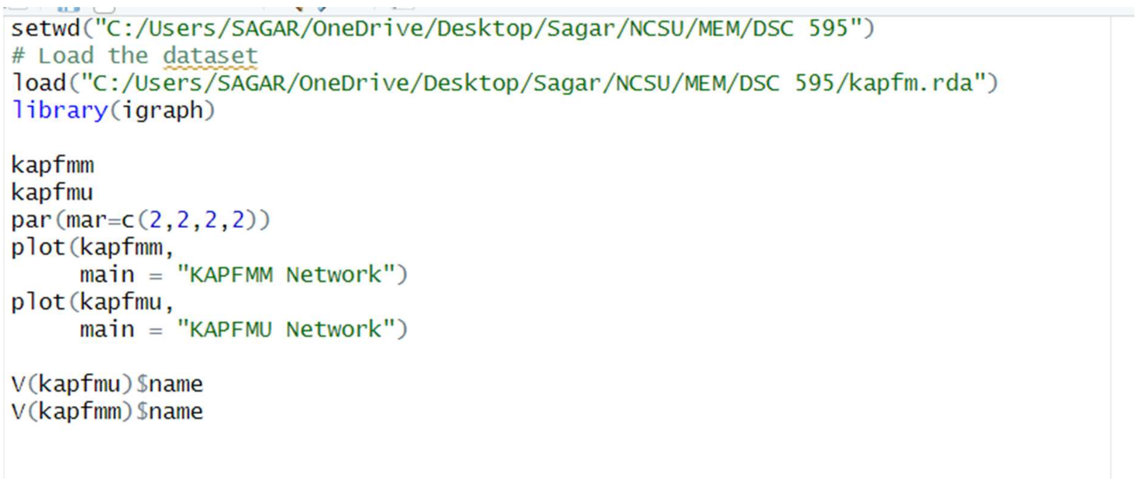
In 1969, anthropologist Bruce Kapferer embarked on an insightful study into the dynamics of conflict among workers in a Zambian mining operation. Through meticulous observation, Kapferer sought to unravel the complexities of workplace relationships and their role in the development and eventual resolution of a dispute that was primarily between two men, Abraham and Donald. This conflict, which saw most workers siding with Abraham, provided a fertile ground for examining the network of interactions that underpin social conflicts.

To dissect the layers of this interaction, Kapferer differentiated between uniplex ties, which are connections based on a single type of relationship depicted in the matrix KAPFMU, and multiplex ties, which are more intricate connections that encompass multiple types of relationships, as illustrated in the matrix KAPFMM. Both matrices are symmetric and binary, offering a structured lens through which the social ties among the miners could be analyzed, laying the groundwork for what would become a pivotal study in network analysis and its application in conflict studies.

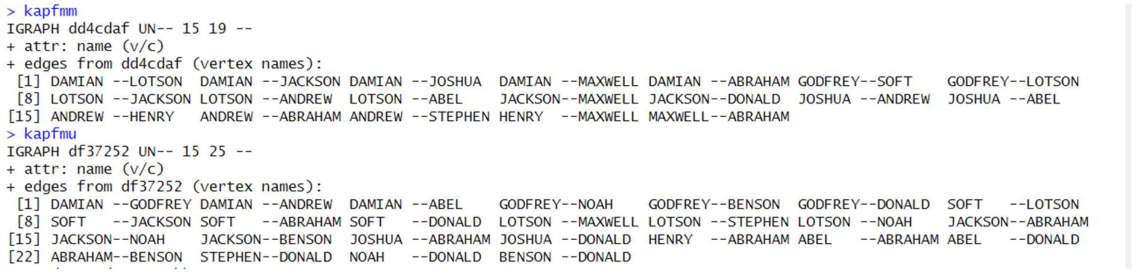
Research questions

1. Examine the dataset to grasp the core characteristics of the network. Determine which type of Network Analysis, whether a Static Network Analysis or a Dynamic Network Analysis would be more appropriate for this.
2.How does the structure of the KAPFMM network differs from the KAPFMU network in terms of centrality, modularity, and community, and how do these measures compare when accounting for the presence of isolates?
3. What are the fundamental structural patterns within the KAPFMM network, and in what ways do they correlate with the network's coreness distribution and assortativity metrics?
4. In what manner does employing block modeling and structural equivalence analysis contribute to a deeper understanding of community structures within the KAPFMM network?

Datasets KAPFMU and KAPFMM
R Code

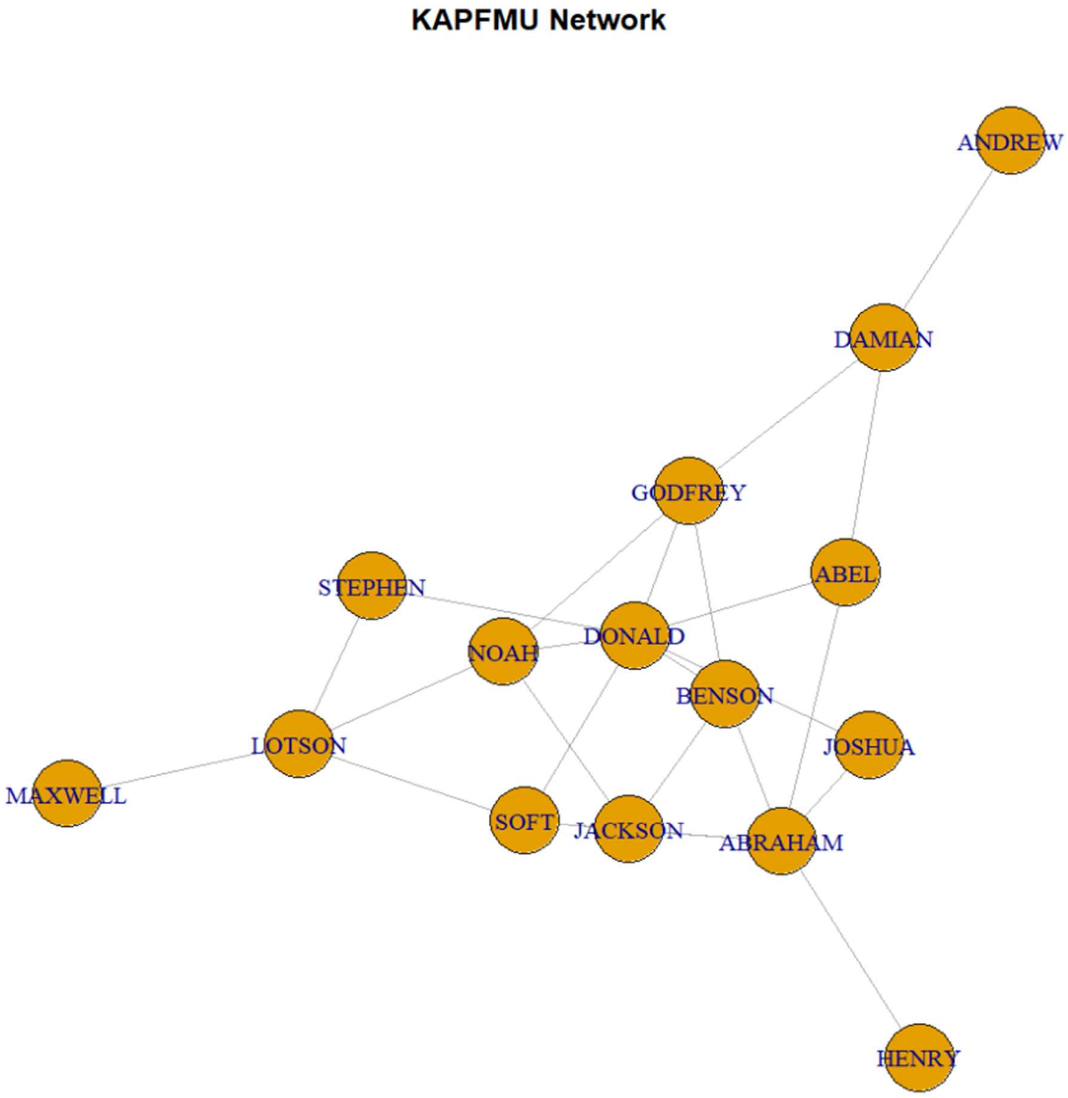


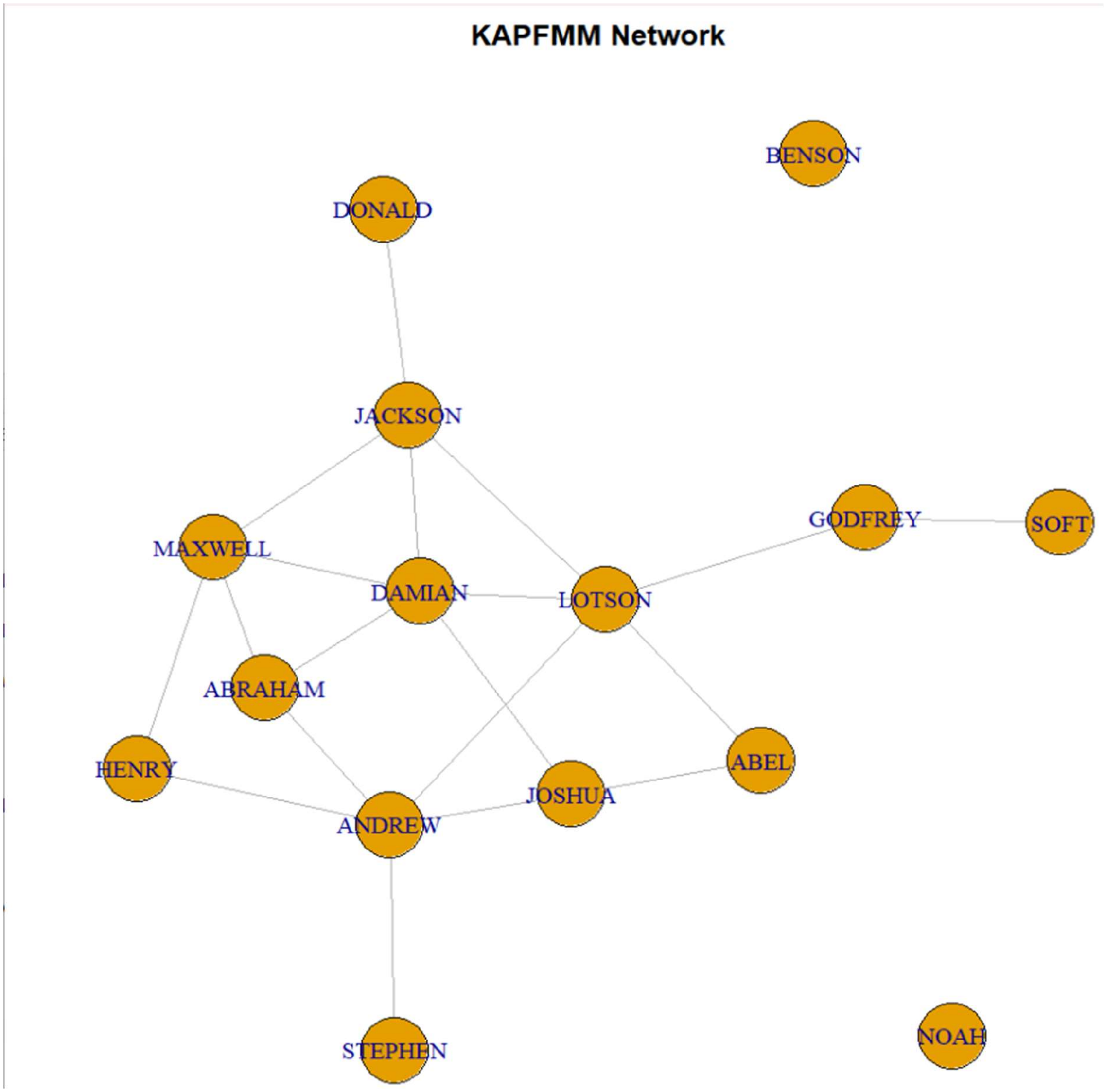
```
setwd("C:/Users/SAGAR/OneDrive/Desktop/Sagar/NCSU/MEM/DSC 595")
# Load the dataset
load("C:/Users/SAGAR/OneDrive/Desktop/Sagar/NCSU/MEM/DSC 595/kapfm.rda")
library(igraph)

kapfmm
kapfmu
par(mar=c(2,2,2,2))
plot(kapfmm,
     main = "KAPFMM Network")
plot(kapfmu,
     main = "KAPFMU Network")

V(kapfmu)$name
V(kapfmm)$name
```



```
> kapfmm
IGRAPH dd4cdaf UN-- 15 19 --
+ attr: name (v/c)
+ edges from dd4cdaf (vertex names):
 [1] DAMIAN --LOTSON  DAMIAN --JACKSON DAMIAN --JOSHUA  DAMIAN --MAXWELL DAMIAN --ABRAHAM GODFREY--SOFT    GODFREY--LOTSON
 [8] LOTSON --JACKSON LOTSON --ANDREW  LOTSON --ABEL    JACKSON--MAXWELL JACKSON--DONALD  JOSHUA --ANDREW  JOSHUA --ABEL
[15] ANDREW --HENRY   ANDREW --ABRAHAM ANDREW --STEPHEN HENRY  --MAXWELL MAXWELL--ABRAHAM
> kapfmu
IGRAPH df37252 UN-- 15 25 --
+ attr: name (v/c)
+ edges from df37252 (vertex names):
 [1] DAMIAN --GODFREY DAMIAN --ANDREW  DAMIAN --ABEL    GODFREY--NOAH    GODFREY--BENSON  GODFREY--DONALD  SOFT   --LOTSON
 [8] SOFT   --JACKSON SOFT   --ABRAHAM SOFT   --DONALD  LOTSON --MAXWELL LOTSON --STEPHEN LOTSON --NOAH    JACKSON--ABRAHAM
[15] JACKSON--NOAH    JACKSON--BENSON  JOSHUA --ABRAHAM JOSHUA --DONALD  HENRY  --ABRAHAM ABEL   --ABRAHAM ABEL   --DONALD
[22] ABRAHAM--BENSON  STEPHEN--DONALD  NOAH   --DONALD  BENSON --DONALD
```

Graph Plots





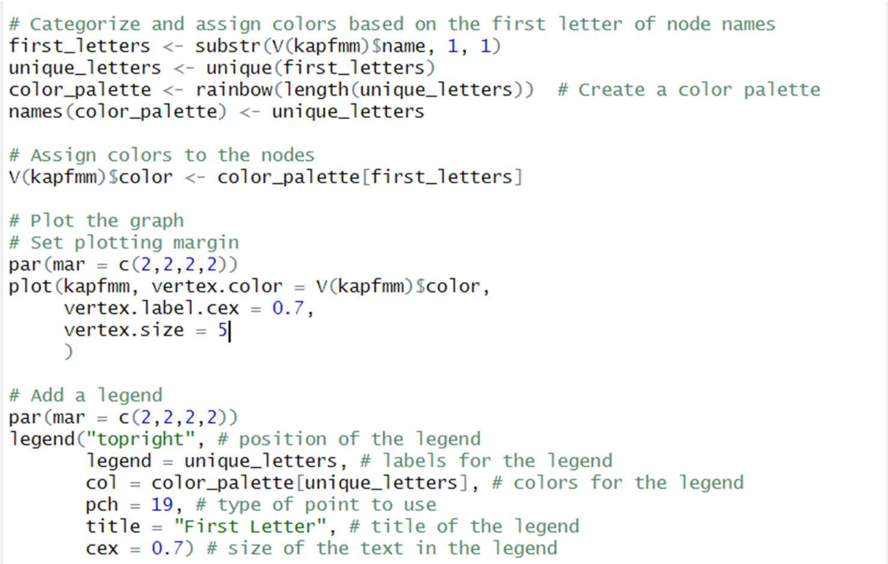
After analysing both the dataset we understand that if the dataset only contains information about the nodes and the edges connecting them without any timestamps or sequence data, it is suited for static network analysis. Static network analysis examines the structure of the network at a single point in time, focusing on properties such as network density, degree distribution, centrality measures, community structures, and other metrics that do not change over time. Whereas if the dataset includes temporal information, such as timestamps on edges indicating when interactions occur or attributes that change over time, then it can be analyzed as a dynamic network. In summary, without timestamps or sequential data, you would default to static network analysis. Therefore we perform static network analysis on the dataset.

After Analyzing the network, we focus on static network analyses such as:

- Network Visualization: Visualize the network to understand its structure.
- Centrality Measures: Calculate various centrality measures to identify important nodes.
- Community Detection: Detect communities within the network.

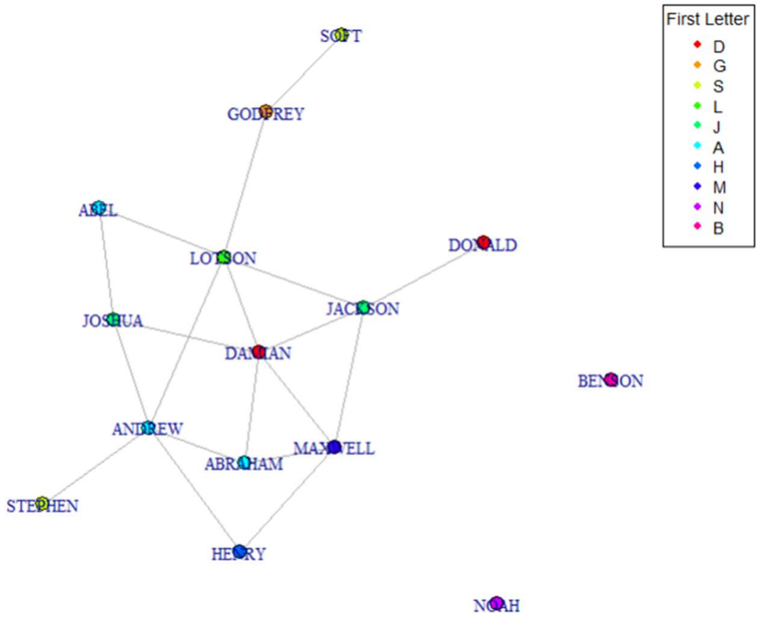
For Network Visualization we categorize and assign colors to KAPFMM network using its first letters of its node names

R Code



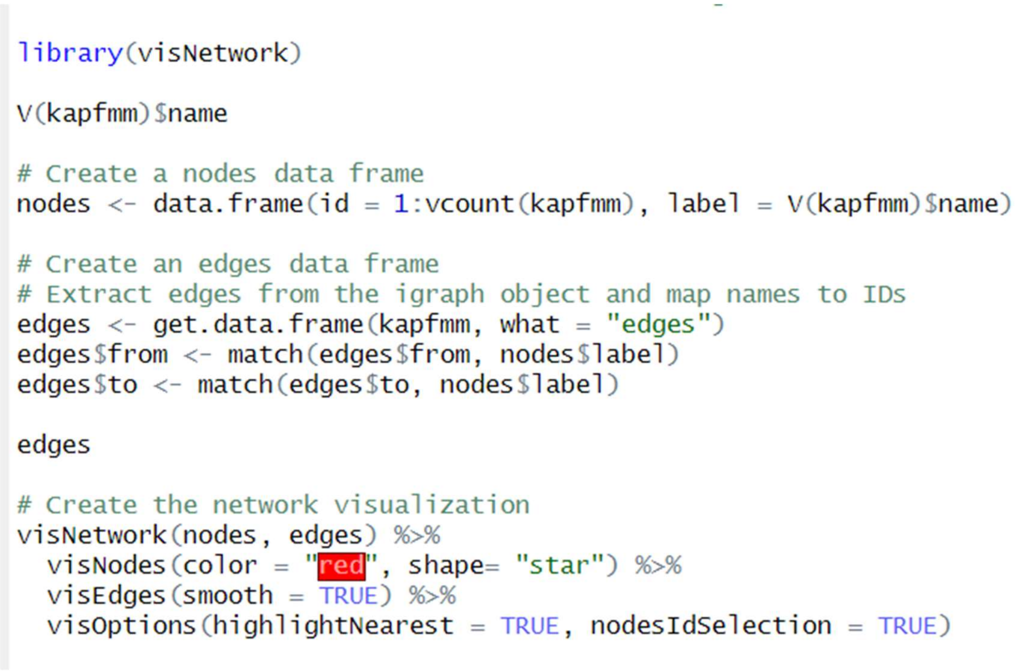
```r
# Categorize and assign colors based on the first letter of node names
first_letters <- substr(V(kapfmm)$name, 1, 1)
unique_letters <- unique(first_letters)
color_palette <- rainbow(length(unique_letters))  # Create a color palette
names(color_palette) <- unique_letters

# Assign colors to the nodes
V(kapfmm)$color <- color_palette[first_letters]

# Plot the graph
# Set plotting margin
par(mar = c(2,2,2,2))
plot(kapfmm, vertex.color = V(kapfmm)$color,
     vertex.label.cex = 0.7,
     vertex.size = 5
     )

# Add a legend
par(mar = c(2,2,2,2))
legend("topright", # position of the legend
       legend = unique_letters, # labels for the legend
       col = color_palette[unique_letters], # colors for the legend
       pch = 19, # type of point to use
       title = "First Letter", # title of the legend
       cex = 0.7) # size of the text in the legend
```

Output



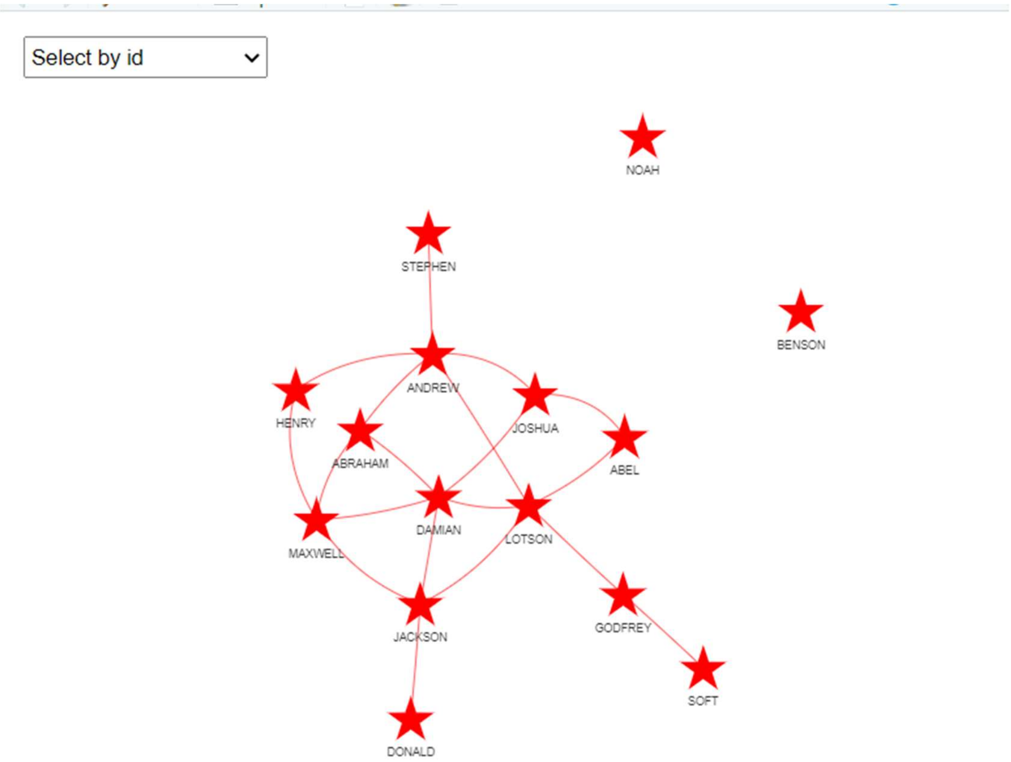
We make the both the network graphs (kapfmm & kapfmu) interactive using visNetwork Library

R Code

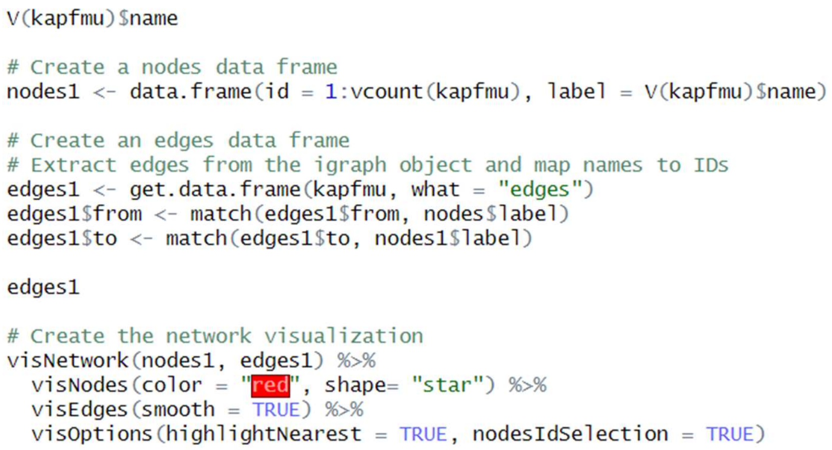


```
library(visNetwork)

V(kapfmm)$name

# Create a nodes data frame
nodes <- data.frame(id = 1:vcount(kapfmm), label = V(kapfmm)$name)

# Create an edges data frame
# Extract edges from the igraph object and map names to IDs
edges <- get.data.frame(kapfmm, what = "edges")
edges$from <- match(edges$from, nodes$label)
edges$to <- match(edges$to, nodes$label)

edges

# Create the network visualization
visNetwork(nodes, edges) %>%
  visNodes(color = "red", shape= "star") %>%
  visEdges(smooth = TRUE) %>%
  visOptions(highlightNearest = TRUE, nodesIdSelection = TRUE)
```

Output Graph plot for KAPFMM

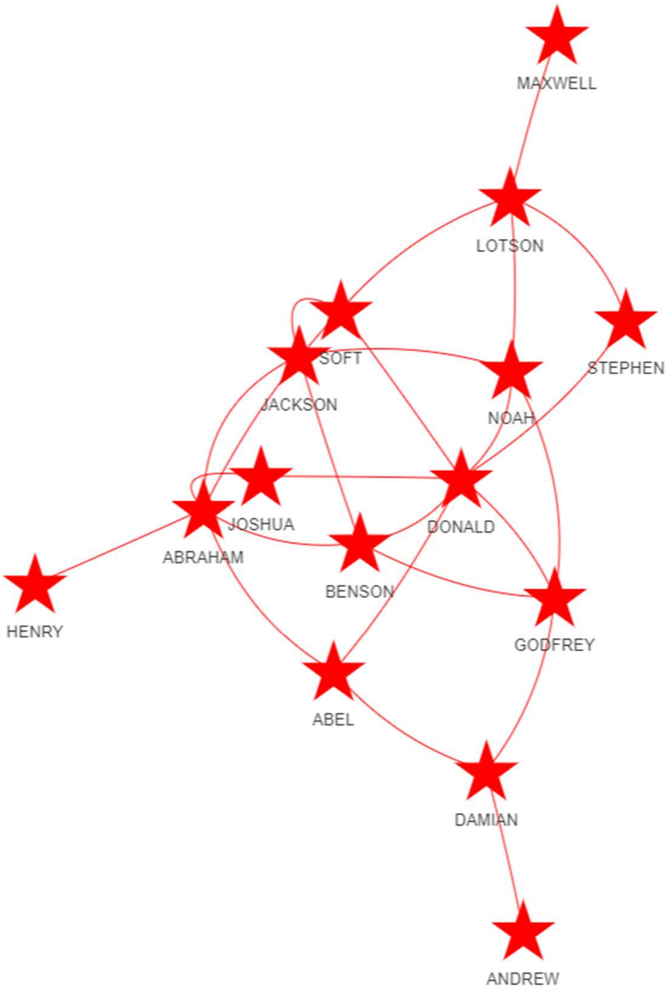


R Code



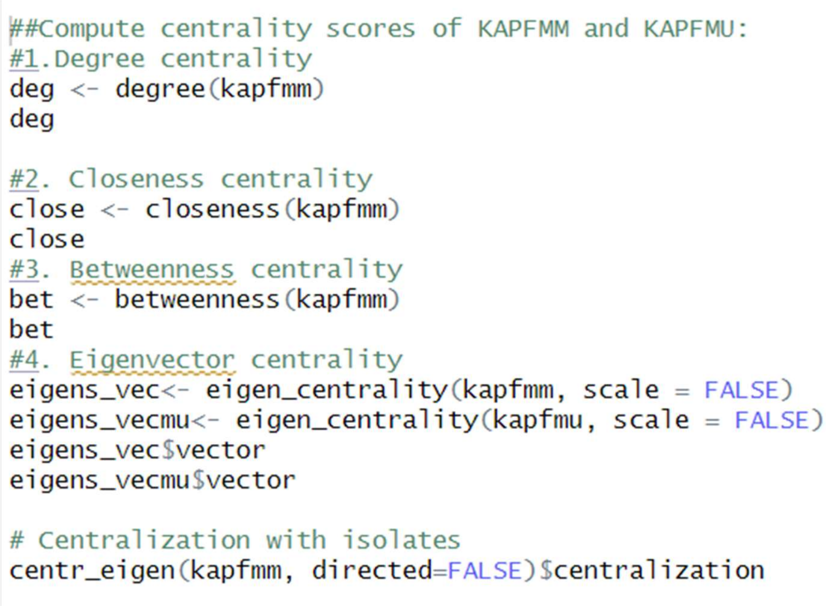
```
V(kapfmu)$name

# Create a nodes data frame
nodes1 <- data.frame(id = 1:vcount(kapfmu), label = V(kapfmu)$name)

# Create an edges data frame
# Extract edges from the igraph object and map names to IDs
edges1 <- get.data.frame(kapfmu, what = "edges")
edges1$from <- match(edges1$from, nodes$label)
edges1$to <- match(edges1$to, nodes1$label)

edges1

# Create the network visualization
visNetwork(nodes1, edges1) %>%
  visNodes(color = "red", shape= "star") %>%
  visEdges(smooth = TRUE) %>%
  visOptions(highlightNearest = TRUE, nodesIdSelection = TRUE)
```

Output Graph plot for KAPFMU



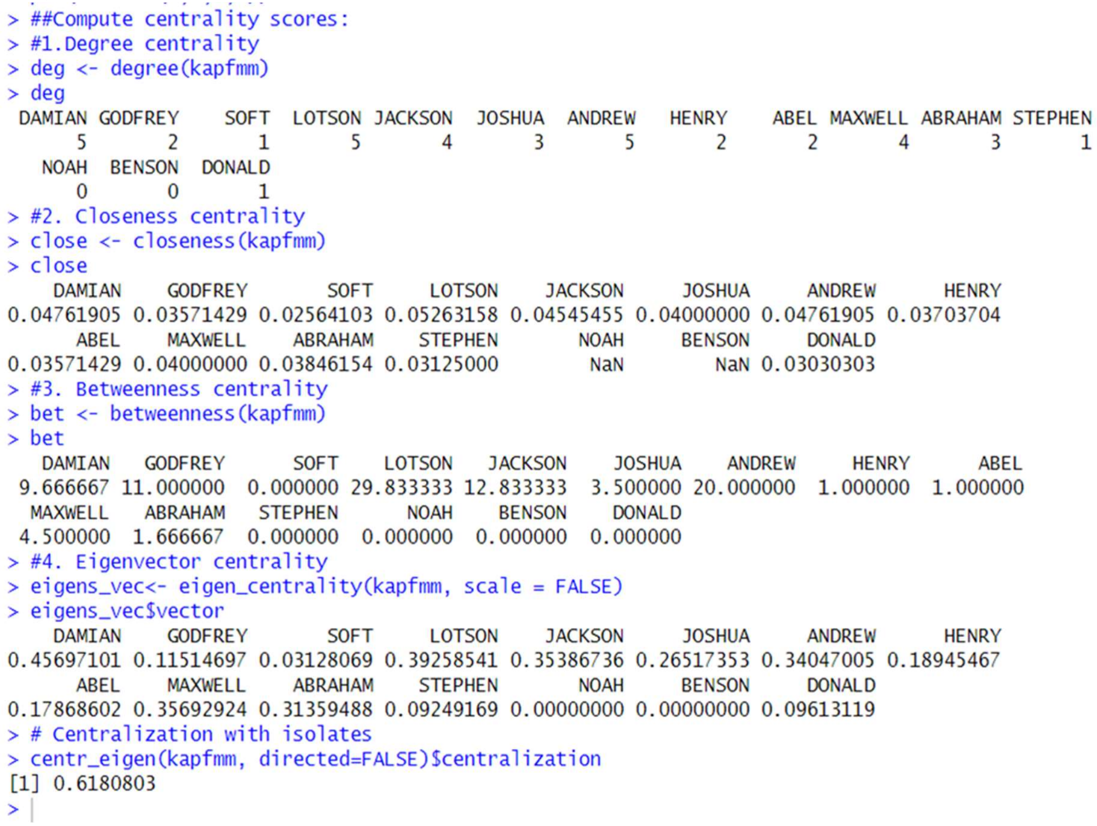
Centrality Measures

R code for KAPFMM



```
##Compute centrality scores of KAPFMM and KAPFMU:
#1.Degree centrality
deg <- degree(kapfmm)
deg

#2. Closeness centrality
close <- closeness(kapfmm)
close
#3. Betweenness centrality
bet <- betweenness(kapfmm)
bet
#4. Eigenvector centrality
eigens_vec<- eigen_centrality(kapfmm, scale = FALSE)
eigens_vecmu<- eigen_centrality(kapfmu, scale = FALSE)
eigens_vec$vector
eigens_vecmu$vector

# Centralization with isolates
centr_eigen(kapfmm, directed=FALSE)$centralization
```

Output

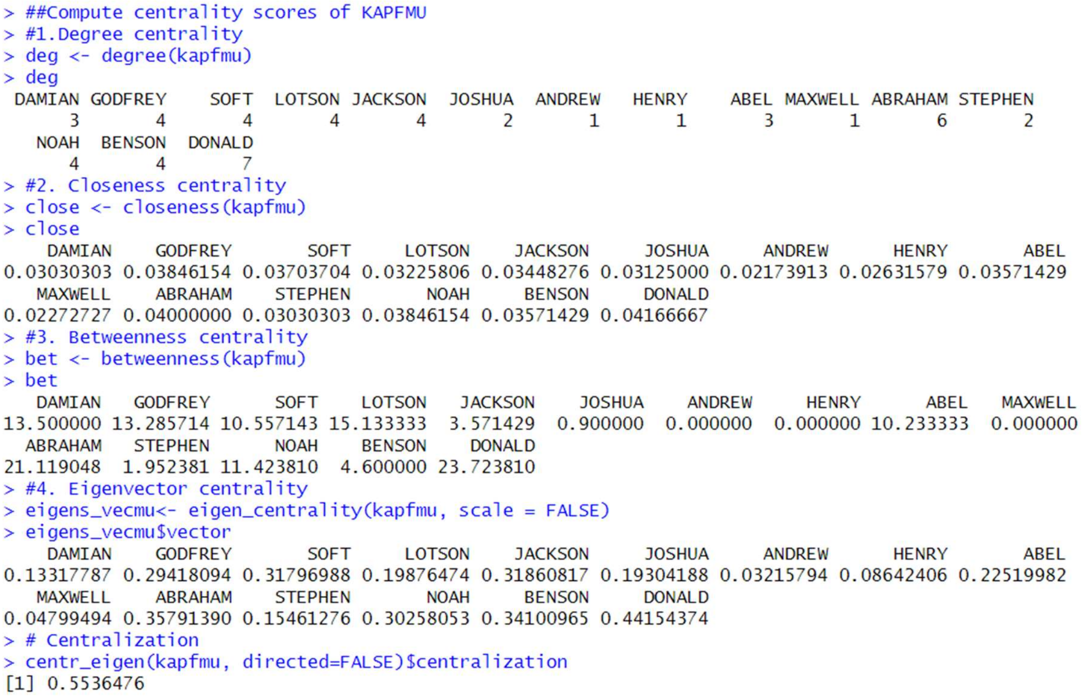


```
> ##Compute centrality scores:
> #1.Degree centrality
> deg <- degree(kapfmm)
> deg
 DAMIAN GODFREY    SOFT  LOTSON JACKSON  JOSHUA  ANDREW   HENRY    ABEL MAXWELL ABRAHAM STEPHEN
      5       2       1       5       4       3       5       2       2       4       3       1
   NOAH  BENSON  DONALD
      0       0       1
> #2. Closeness centrality
> close <- closeness(kapfmm)
> close
    DAMIAN    GODFREY       SOFT     LOTSON    JACKSON     JOSHUA     ANDREW      HENRY
0.04761905 0.03571429 0.02564103 0.05263158 0.04545455 0.04000000 0.04761905 0.03703704
      ABEL    MAXWELL    ABRAHAM    STEPHEN       NOAH     BENSON     DONALD
0.03571429 0.04000000 0.03846154 0.03125000        NaN        NaN 0.03030303
> #3. Betweenness centrality
> bet <- betweenness(kapfmm)
> bet
   DAMIAN   GODFREY      SOFT    LOTSON   JACKSON    JOSHUA    ANDREW     HENRY      ABEL
 9.666667 11.000000  0.000000 29.833333 12.833333  3.500000 20.000000  1.000000  1.000000
  MAXWELL   ABRAHAM   STEPHEN      NOAH    BENSON    DONALD
 4.500000  1.666667  0.000000  0.000000  0.000000  0.000000
> #4. Eigenvector centrality
> eigens_vec<- eigen_centrality(kapfmm, scale = FALSE)
> eigens_vec$vector
    DAMIAN    GODFREY       SOFT     LOTSON    JACKSON     JOSHUA     ANDREW      HENRY
0.45697101 0.11514697 0.03128069 0.39258541 0.35386736 0.26517353 0.34047005 0.18945467
      ABEL    MAXWELL    ABRAHAM    STEPHEN       NOAH     BENSON     DONALD
0.17868602 0.35692924 0.31359488 0.09249169 0.00000000 0.00000000 0.09613119
> # Centralization with isolates
> centr_eigen(kapfmm, directed=FALSE)$centralization
[1] 0.6180803
>
```

Explanation

Degree centrality scores indicate the number of direct connections each node has, with DAMIAN and ANDREW having the highest at 5.
Closeness centrality measures how close a node is to all other nodes in the network, and JACKSON has the highest score, suggesting he can spread information efficiently.
Betweenness centrality identifies nodes that serve as bridges between other nodes; GODFREY and LOTSON have high scores, indicating they control the flow of information.
Eigenvector centrality considers the influence of a node's connections, where DAMIAN scores highest, meaning he is connected to other well-connected nodes.
The centralization score suggests the network has a moderately centralized structure.

R code for KAPFMU with Output



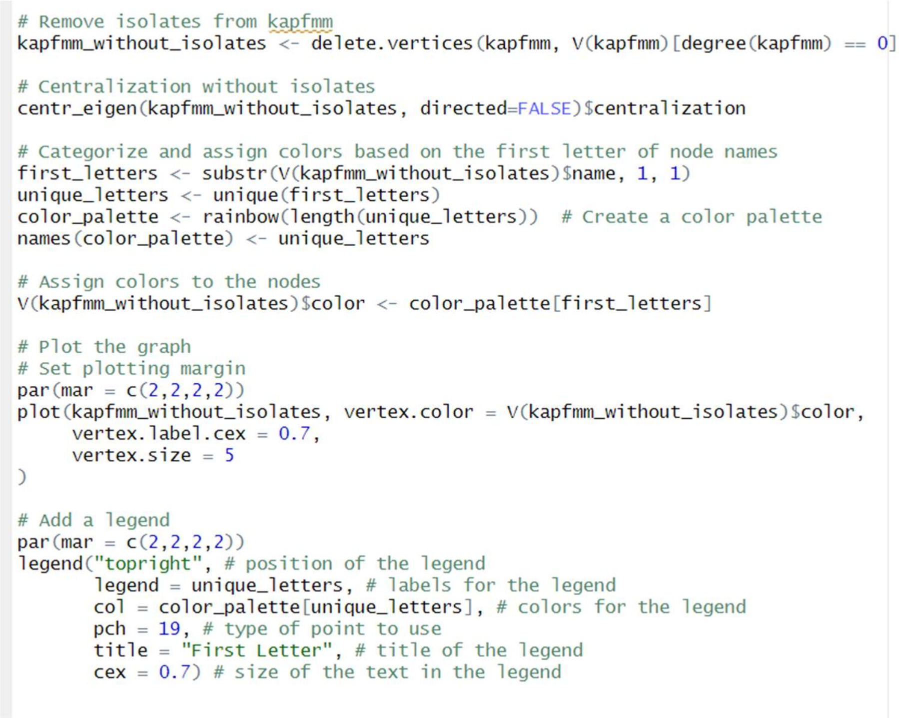
```
> ##Compute centrality scores of KAPFMU
> #1.Degree centrality
> deg <- degree(kapfmu)
> deg
 DAMIAN GODFREY    SOFT  LOTSON JACKSON  JOSHUA  ANDREW   HENRY    ABEL MAXWELL ABRAHAM STEPHEN
      3       4       4       4       4       2       1       1       3       1       6       2
   NOAH  BENSON  DONALD
      4       4       7
> #2. Closeness centrality
> close <- closeness(kapfmu)
> close
    DAMIAN    GODFREY       SOFT     LOTSON    JACKSON     JOSHUA     ANDREW      HENRY       ABEL
0.03030303 0.03846154 0.03703704 0.03225806 0.03448276 0.03125000 0.02173913 0.02631579 0.03571429
   MAXWELL    ABRAHAM    STEPHEN       NOAH     BENSON     DONALD
0.02272727 0.04000000 0.03030303 0.03846154 0.03571429 0.04166667
> #3. Betweenness centrality
> bet <- betweenness(kapfmu)
> bet
   DAMIAN   GODFREY      SOFT    LOTSON   JACKSON    JOSHUA    ANDREW     HENRY      ABEL   MAXWELL
13.500000 13.285714 10.557143 15.133333  3.571429  0.900000  0.000000  0.000000 10.233333  0.000000
  ABRAHAM   STEPHEN      NOAH    BENSON    DONALD
21.119048  1.952381 11.423810  4.600000 23.723810
> #4. Eigenvector centrality
> eigens_vecmu<- eigen_centrality(kapfmu, scale = FALSE)
> eigens_vecmu$vector
    DAMIAN    GODFREY       SOFT     LOTSON    JACKSON     JOSHUA     ANDREW      HENRY       ABEL
0.13317787 0.29418094 0.31796988 0.19876474 0.31860817 0.19304188 0.03215794 0.08642406 0.22519982
   MAXWELL    ABRAHAM    STEPHEN       NOAH     BENSON     DONALD
0.04799494 0.35791390 0.15461276 0.30258053 0.34100965 0.44154374
> # Centralization
> centr_eigen(kapfmu, directed=FALSE)$centralization
[1] 0.5536476
```

Explanation

The centrality measures are computed for the KAPFMU dataset. The degree centrality reveals that Abraham has the highest degree with 6, indicating he has the most direct connections in the network.
In closeness centrality, Donald and Stephen appear to have the lowest values, suggesting they are less central in terms of the shortest paths to all other nodes.
In betweenness centrality, GODFREY and LOTSON have high values, signifying they act as significant intermediaries in the network.
Eigenvector centrality, which indicates the influence of a node, shows Abraham with the highest score, suggesting his connections are to other highly connected nodes.
The centralization score is approximately 0.53, indicating a moderate level of centralization in the network. These measures suggest that **Abraham** is a key figure in this network, with significant direct and indirect influence.

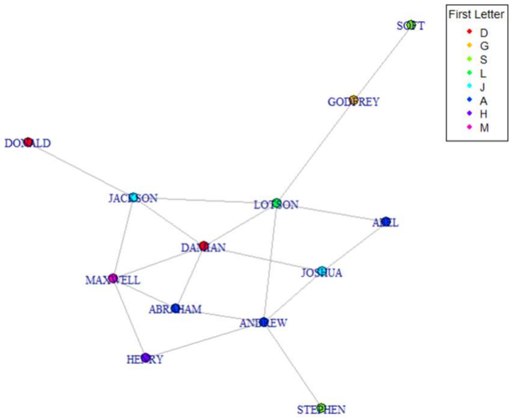
Getting Rid of the Isolates in KAPFMM network

R Code



```r
# Remove isolates from kapfmm
kapfmm_without_isolates <- delete.vertices(kapfmm, V(kapfmm)[degree(kapfmm) == 0])

# Centralization without isolates
centr_eigen(kapfmm_without_isolates, directed=FALSE)$centralization

# Categorize and assign colors based on the first letter of node names
first_letters <- substr(V(kapfmm_without_isolates)$name, 1, 1)
unique_letters <- unique(first_letters)
color_palette <- rainbow(length(unique_letters))  # Create a color palette
names(color_palette) <- unique_letters

# Assign colors to the nodes
V(kapfmm_without_isolates)$color <- color_palette[first_letters]

# Plot the graph
# Set plotting margin
par(mar = c(2,2,2,2))
plot(kapfmm_without_isolates, vertex.color = V(kapfmm_without_isolates)$color,
     vertex.label.cex = 0.7,
     vertex.size = 5
)

# Add a legend
par(mar = c(2,2,2,2))
legend("topright", # position of the legend
       legend = unique_letters, # labels for the legend
       col = color_palette[unique_letters], # colors for the legend
       pch = 19, # type of point to use
       title = "First Letter", # title of the legend
       cex = 0.7) # size of the text in the legend
```

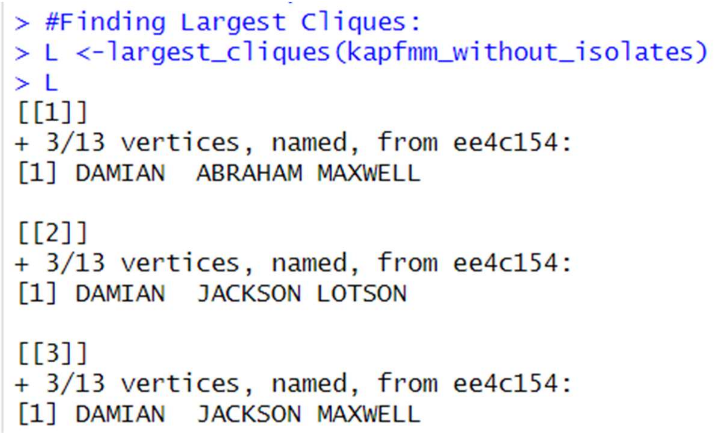
Output



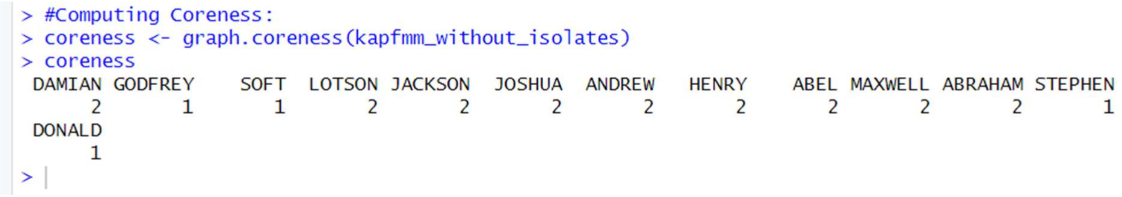
Community Detection

Inorder to understand the community cohesiveness we measure the networks find the largest cliques in the network and their coreness. The largest clique is a measure of the maximum complete subgraph within a larger graph. This provides insights into the strongest mutual relationships within the network. In social networks, individuals with high coreness are those who are critical for spreading information or maintaining the cohesiveness of their community. To measure how well connected the core of the network is coreness helps in identifying the most influential nodes in a network, which are necessary for the connectivity of the network. If nodes with high coreness are removed, the network might fragment into disconnected subnetworks.

R code with Output for KAPFMM



```
> #Finding Largest Cliques:
> L <-largest_cliques(kapfmm_without_isolates)
> L
[[1]]
+ 3/13 vertices, named, from ee4c154:
[1] DAMIAN  ABRAHAM MAXWELL

[[2]]
+ 3/13 vertices, named, from ee4c154:
[1] DAMIAN  JACKSON LOTSON

[[3]]
+ 3/13 vertices, named, from ee4c154:
[1] DAMIAN  JACKSON MAXWELL
```



```
> #Computing Coreness:
> coreness <- graph.coreness(kapfmm_without_isolates)
> coreness
 DAMIAN GODFREY    SOFT  LOTSON JACKSON  JOSHUA  ANDREW   HENRY    ABEL MAXWELL ABRAHAM STEPHEN 
      2       1       1       2       2       2       2       2       2       2       2       1 
 DONALD 
      1 
> 
```

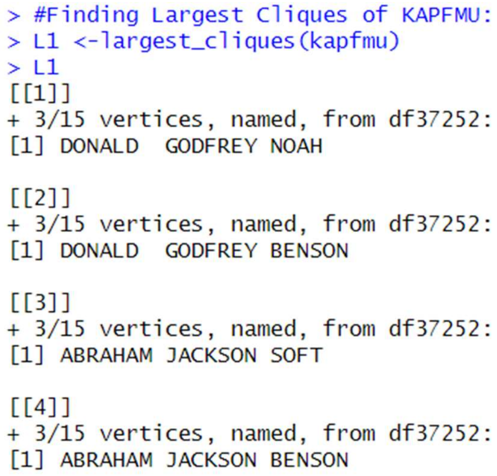
Explanation

We see that the largest cliques found within the network are groups of three nodes that are all directly connected to each other. The cliques mentioned include:

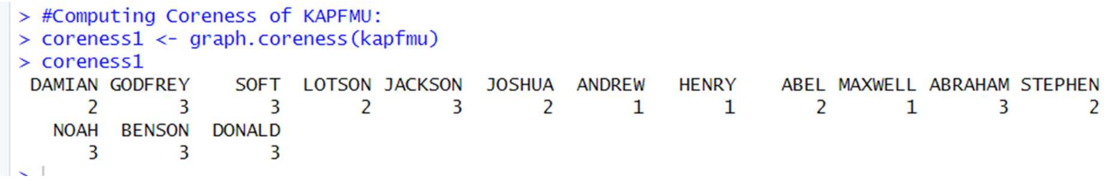
DAMIAN, ABRAHAM, MAXWELL
DAMIAN, JACKSON, LOTSON
DAMIAN, JACKSON, MAXWELL
These cliques indicate tightly-knit subgroups within the network where each member interacts with all the other members.

The coreness measure for each node is given. Coreness in network analysis refers to the degree to which a node is central to the network, with higher coreness values indicating a more central position. Here, DAMIAN has a coreness of 2, which is among the highest, suggesting that he is central to the network's structure. GODFREY, LOTSON, JACKSON, JOSHUA, ANDREW, HENRY, ABEL, MAXWELL, and ABRAHAM also have a coreness of 2, indicating that they too are relatively central within the network. In contrast, SOFT and STEPHEN have coreness values of 1, indicating a more peripheral position.

R code with Output for KAPFMU



```
> #Finding Largest Cliques of KAPFMU:
> L1 <-largest_cliques(kapfmu)
> L1
[[1]]
+ 3/15 vertices, named, from df37252:
[1] DONALD  GODFREY NOAH

[[2]]
+ 3/15 vertices, named, from df37252:
[1] DONALD  GODFREY BENSON

[[3]]
+ 3/15 vertices, named, from df37252:
[1] ABRAHAM JACKSON SOFT

[[4]]
+ 3/15 vertices, named, from df37252:
[1] ABRAHAM JACKSON BENSON
```



```
> #Computing Coreness of KAPFMU:
> coreness1 <- graph.coreness(kapfmu)
> coreness1
 DAMIAN GODFREY    SOFT  LOTSON JACKSON  JOSHUA  ANDREW   HENRY    ABEL MAXWELL ABRAHAM STEPHEN
      2       3       3       2       3       2       1       1       2       1       3       2
   NOAH  BENSON  DONALD
      3       3       3
```

The largest cliques consist of three members each, indicating strong subgroups within the network where every member is connected to every other member within the clique. These cliques are:

DONALD, GODFREY, NOAH
DONALD, GODFREY, BENSON
ABRAHAM, JACKSON, SOFT
ABRAHAM, JACKSON, BENSON

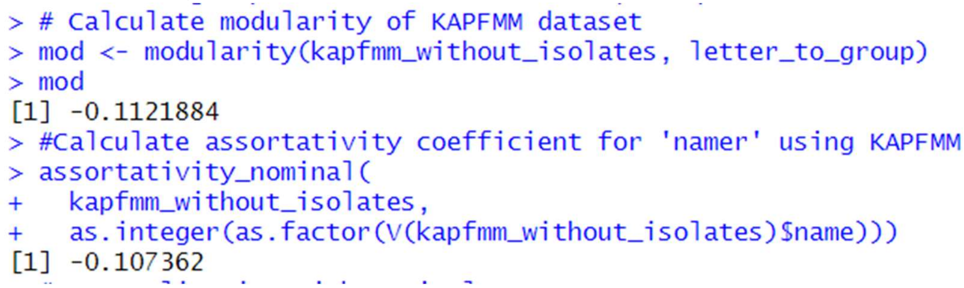
In terms of coreness, NOAH, BENSON, and DONALD have the highest coreness value of 3, suggesting they are central within the network's core structure. GODFREY and SOFT have a coreness value of 3, while DAMIAN, LOTSON, JACKSON, and ABRAHAM have a coreness value of 2. This indicates different levels of centrality, with NOAH, BENSON, and DONALD being the most central. Coreness values can be used to identify influential nodes within the network that are crucial for maintaining the network's connectivity and flow.

We also calculate the modularity and assortativity of the datasets. The calculation of modularity is crucial for detecting community structure in networks, which can be important for understanding the functionality and dynamics of various social systems.

The modularity value ranges from -1 to 1. A high positive value indicates a structure with well-defined communities, whereas a value close to 0 indicates random or uniform interconnections without a clear modularity. Negative values can suggest that the network is not partitioned optimally. The modularity score here is approximately -0.112, which is slightly negative. This suggests that the network does not have a strong community structure.

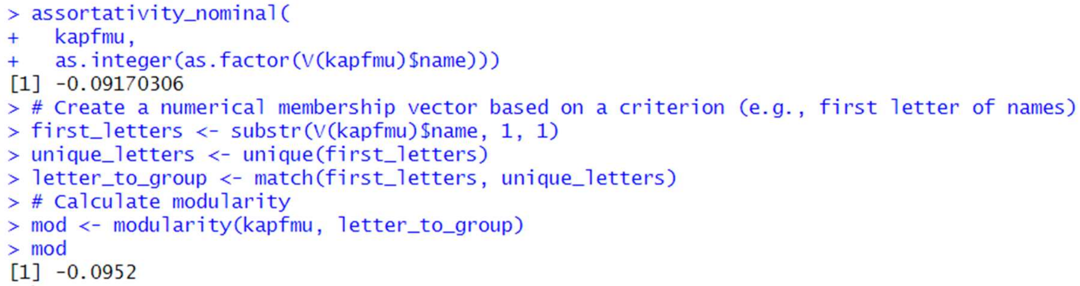
Assortativity measures the similarity of connections in the network with respect to a given node attribute. In social networks, this might be the tendency of individuals to associate with others who are similar to themselves in some way.
The assortativity coefficient is approximately -0.107, which is also negative. This indicates that there's a slight tendency for nodes to be connected to others with dissimilar attributes.

R code with Output for KAPFMM



```
> # Calculate modularity of KAPFMM dataset
> mod <- modularity(kapfmm_without_isolates, letter_to_group)
> mod
[1] -0.1121884
> #Calculate assortativity coefficient for 'namer' using KAPFMM
> assortativity_nominal(
+   kapfmm_without_isolates,
+   as.integer(as.factor(V(kapfmm_without_isolates)$name)))
[1] -0.107362
```

R code with Output for KAPFMM



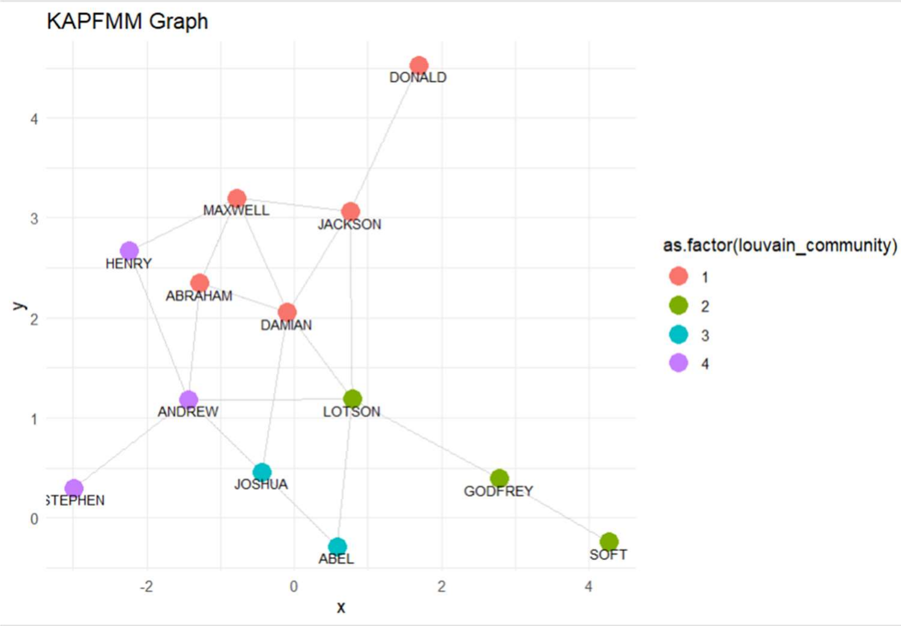
```
> assortativity_nominal(
+   kapfmu,
+   as.integer(as.factor(V(kapfmu)$name)))
[1] -0.09170306
> # Create a numerical membership vector based on a criterion (e.g., first letter of names)
> first_letters <- substr(V(kapfmu)$name, 1, 1)
> unique_letters <- unique(first_letters)
> letter_to_group <- match(first_letters, unique_letters)
> # Calculate modularity
> mod <- modularity(kapfmu, letter_to_group)
> mod
[1] -0.0952
```

Explanation:
Assortativity, with a value of approximately -0.0917 for kapfmu, indicates a tendency for nodes in the network to not be connected to others that are like them in some way. It seems to be evaluating the tendency of nodes to connect with others that do not share similar attributes, possibly the first letter of their names.

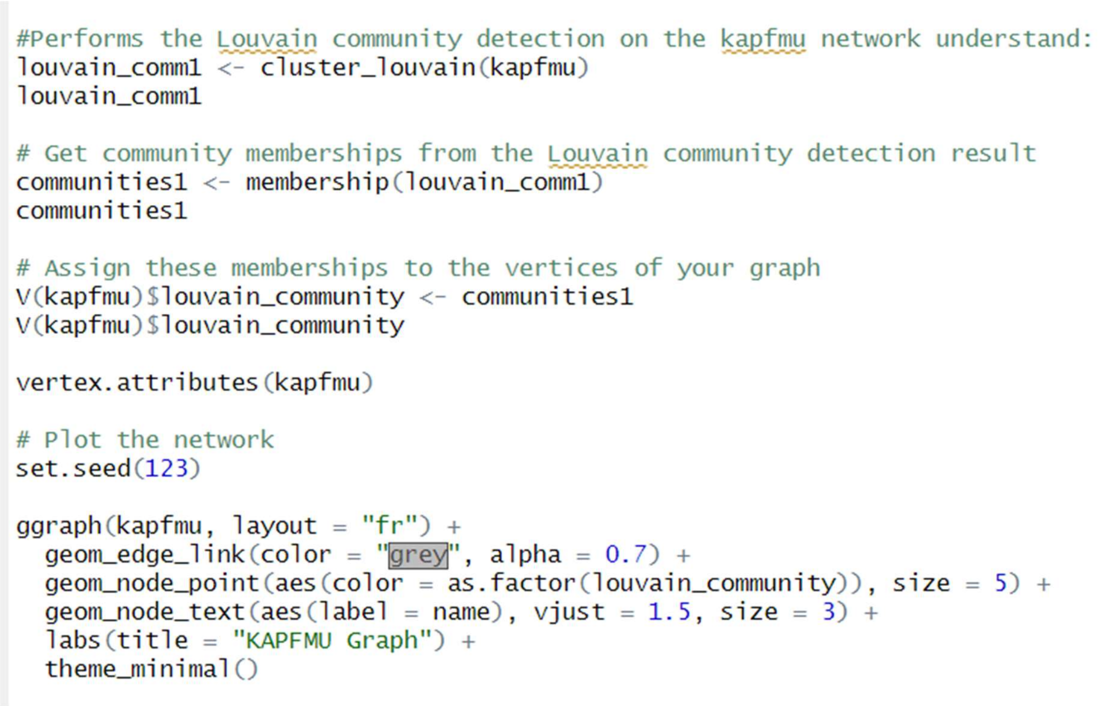
The modularity score of -0.0952 suggests a lack of a clear division of the network into modules or communities based on the given criterion. In network analysis, a negative modularity value can indicate that the network does not have a strong community structure as defined by the chosen categorization, in this case, the first letter of names. This could imply that the network is fairly mixed or homogeneous in terms of the attribute used for grouping.

Louvain Community Detection Algorithm

The communities identified by the Louvain community detection algorithm are sets of nodes that are densely interconnected but sparsely connected to other dense sets within the network.
These communities can be crucial for understanding the structural and functional properties of the network.

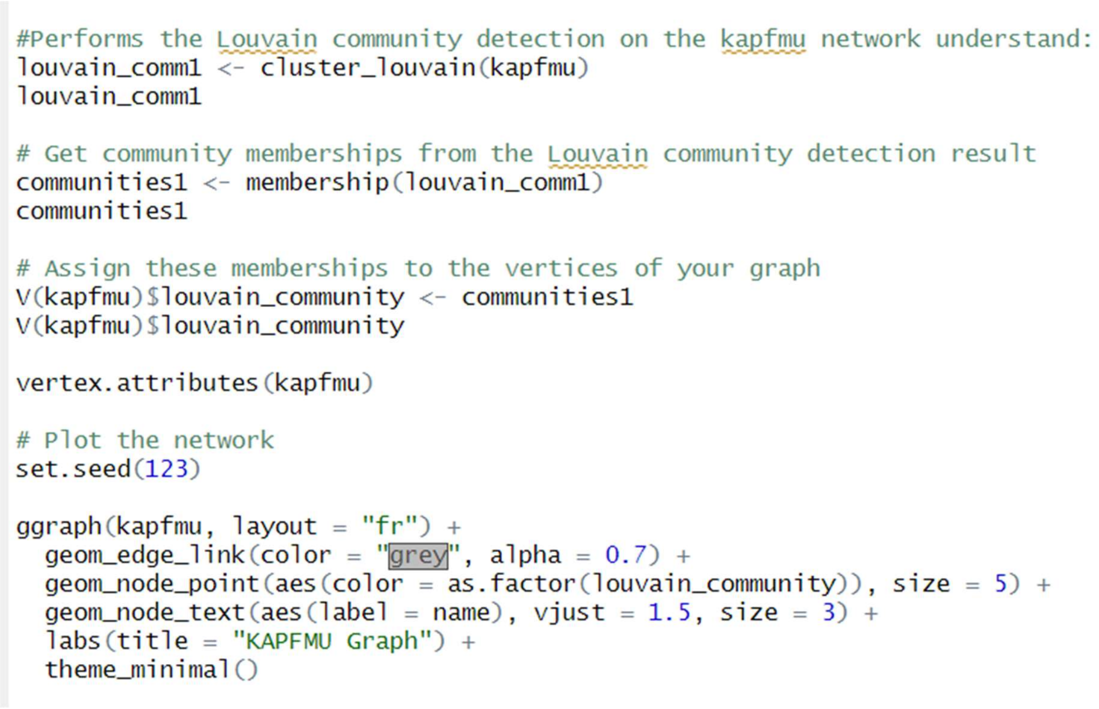


R Code for kapfmu

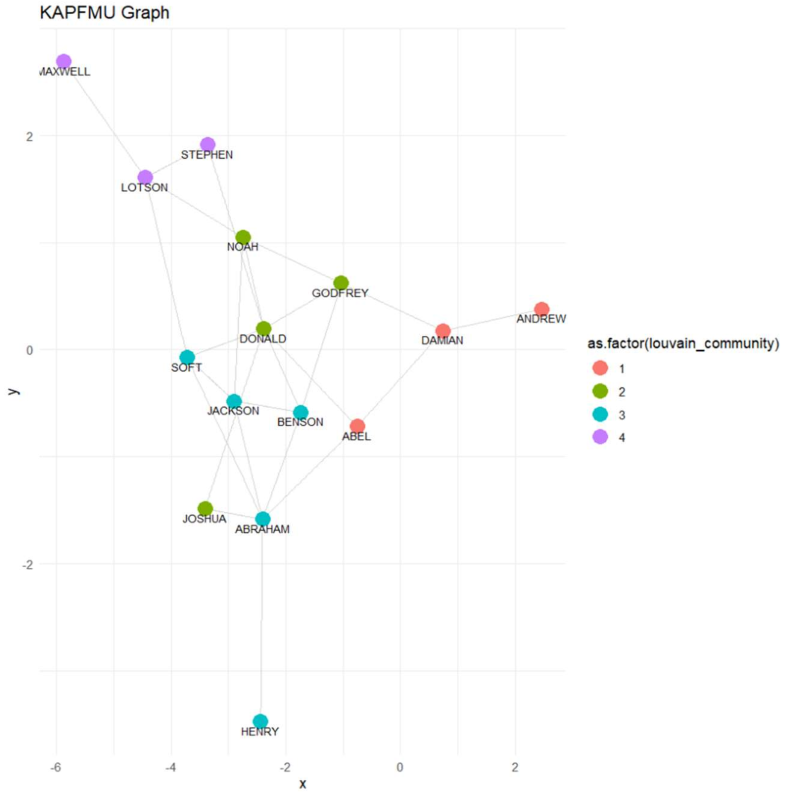


```
#Performs the Louvain community detection on the kapfmu network understand:
louvain_comm1 <- cluster_louvain(kapfmu)
louvain_comm1

# Get community memberships from the Louvain community detection result
communities1 <- membership(louvain_comm1)
communities1

# Assign these memberships to the vertices of your graph
V(kapfmu)$louvain_community <- communities1
V(kapfmu)$louvain_community

vertex.attributes(kapfmu)

# Plot the network
set.seed(123)

ggraph(kapfmu, layout = "fr") +
  geom_edge_link(color = "grey", alpha = 0.7) +
  geom_node_point(aes(color = as.factor(louvain_community)), size = 5) +
  geom_node_text(aes(label = name), vjust = 1.5, size = 3) +
  labs(title = "KAPFMU Graph") +
  theme_minimal()
```

Output



```
#Performs the Louvain community detection on the kapfmu network understand:
louvain_comm1 <- cluster_louvain(kapfmu)
louvain_comm1

# Get community memberships from the Louvain community detection result
communities1 <- membership(louvain_comm1)
communities1

# Assign these memberships to the vertices of your graph
V(kapfmu)$louvain_community <- communities1
V(kapfmu)$louvain_community

vertex.attributes(kapfmu)

# Plot the network
set.seed(123)

ggraph(kapfmu, layout = "fr") +
  geom_edge_link(color = "grey", alpha = 0.7) +
  geom_node_point(aes(color = as.factor(louvain_community)), size = 5) +
  geom_node_text(aes(label = name), vjust = 1.5, size = 3) +
  labs(title = "KAPFMU Graph") +
  theme_minimal()
```



The two graphs represent the results of the Louvain community detection algorithm applied to the KAPFMU and KAPFMM datasets, respectively. The nodes are colored according to the community they belong to, as identified by the algorithm.

In the KAPFMU graph, we see that the nodes are grouped into four communities, indicated by different colors. There appears to be one outlier node, Henry, who is relatively far from other nodes, possibly indicating a peripheral position in the network.

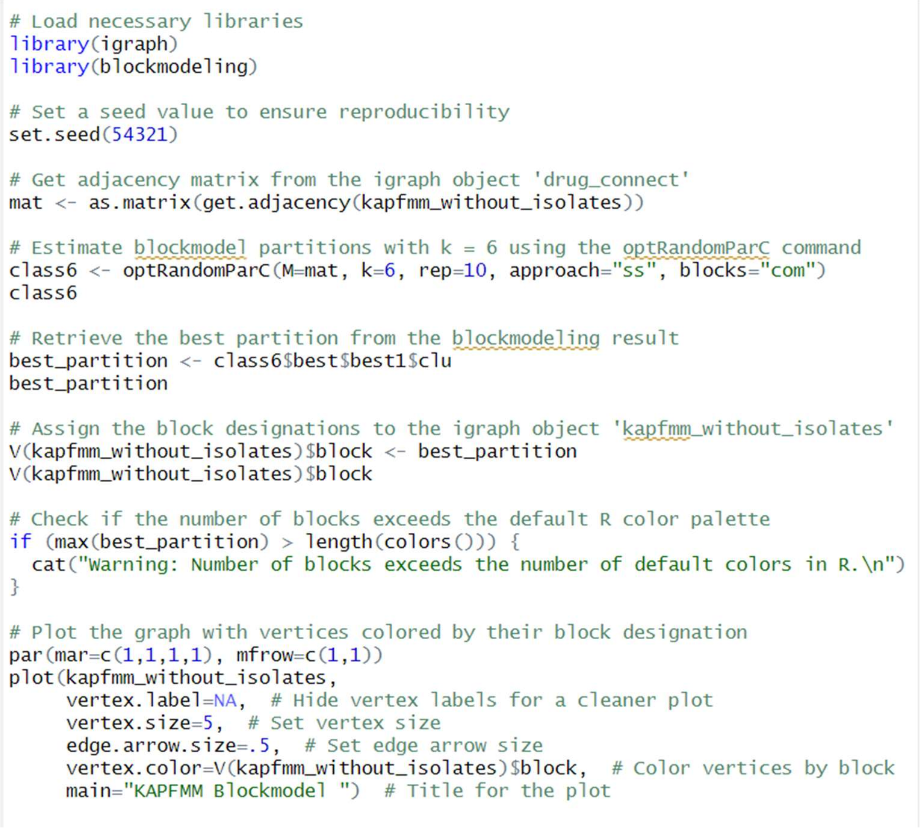
The KAPFMM graph also shows nodes grouped into four communities, but the distribution is different. Here, we notice that the nodes are more evenly spread out, suggesting a more distributed network structure with no clear outliers like in the KAPFMU graph.

The communities in both graphs suggest that within these datasets, there are subgroups of nodes that are more densely interconnected with each other than with the rest of the network.

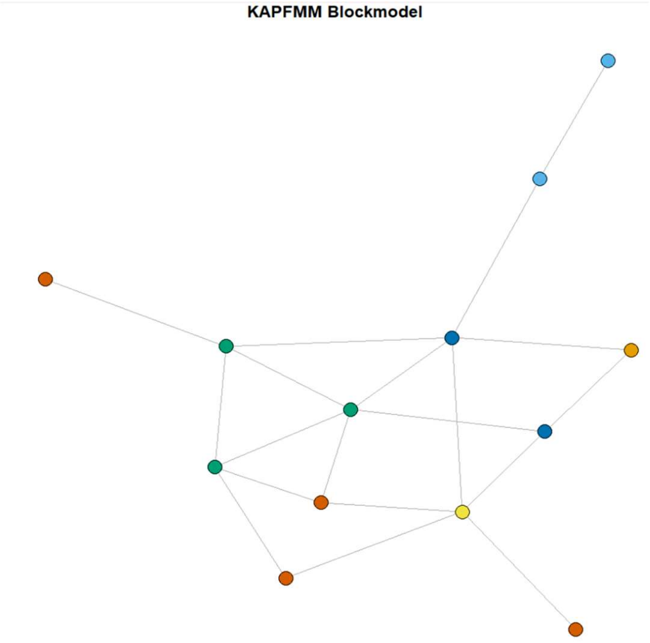
Blockmodeling

Blockmodeling is a technique used in network analysis that aims to simplify the complex structure of a network by dividing it into blocks based on the roles or positions of nodes within the network, rather than on individual attributes or pairwise connections.
It simplifies the representation of the network, making it easier to visualize and understand large networks.
It can reveal important structures such as hierarchies, core-periphery structures, and cohesive subgroups that might be important in the functioning of the network.

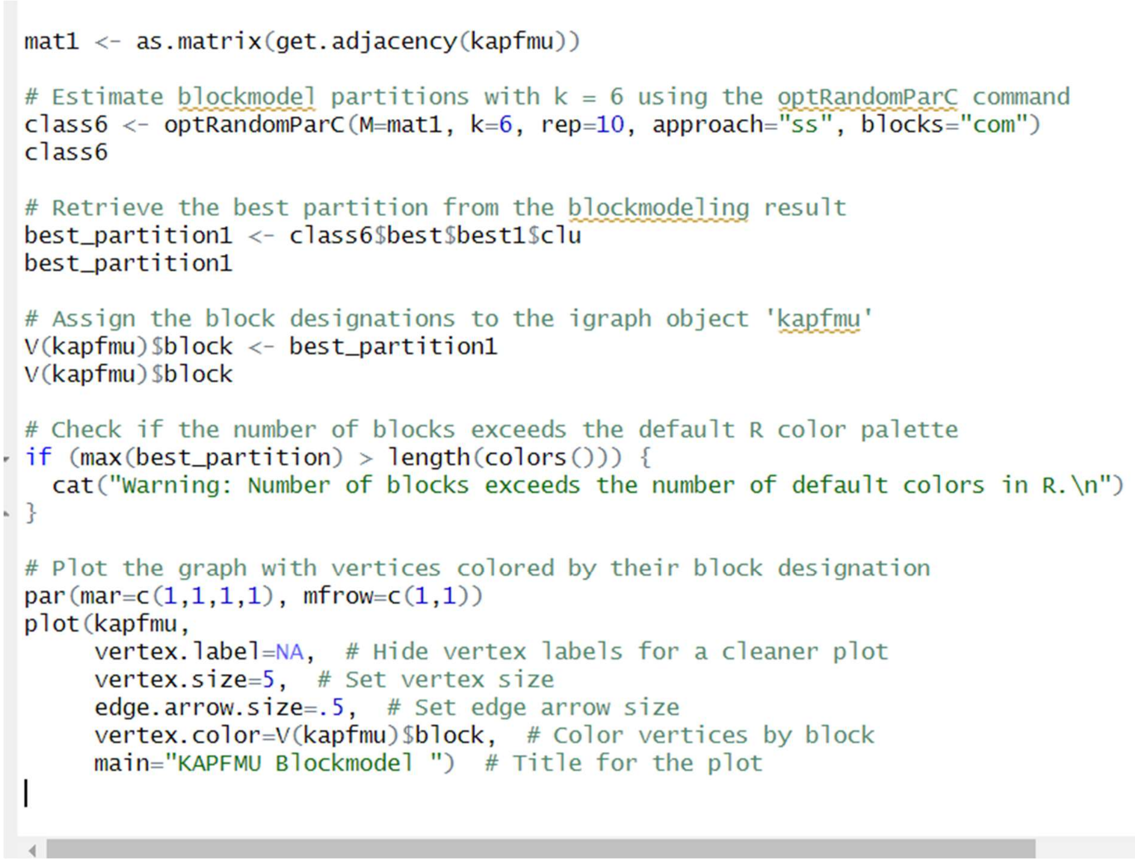
R code for blockmodeling kapfmm



```r
# Load necessary libraries
library(igraph)
library(blockmodeling)

# Set a seed value to ensure reproducibility
set.seed(54321)

# Get adjacency matrix from the igraph object 'drug_connect'
mat <- as.matrix(get.adjacency(kapfmm_without_isolates))

# Estimate blockmodel partitions with k = 6 using the optRandomParC command
class6 <- optRandomParC(M=mat, k=6, rep=10, approach="ss", blocks="com")
class6

# Retrieve the best partition from the blockmodeling result
best_partition <- class6$best$best1$clu
best_partition

# Assign the block designations to the igraph object 'kapfmm_without_isolates'
V(kapfmm_without_isolates)$block <- best_partition
V(kapfmm_without_isolates)$block

# Check if the number of blocks exceeds the default R color palette
if (max(best_partition) > length(colors())) {
  cat("Warning: Number of blocks exceeds the number of default colors in R.\n")
}

# Plot the graph with vertices colored by their block designation
par(mar=c(1,1,1,1), mfrow=c(1,1))
plot(kapfmm_without_isolates,
     vertex.label=NA,  # Hide vertex labels for a cleaner plot
     vertex.size=5,  # Set vertex size
     edge.arrow.size=.5,  # Set edge arrow size
     vertex.color=V(kapfmm_without_isolates)$block,  # Color vertices by block
     main="KAPFMM Blockmodel ")  # Title for the plot
```

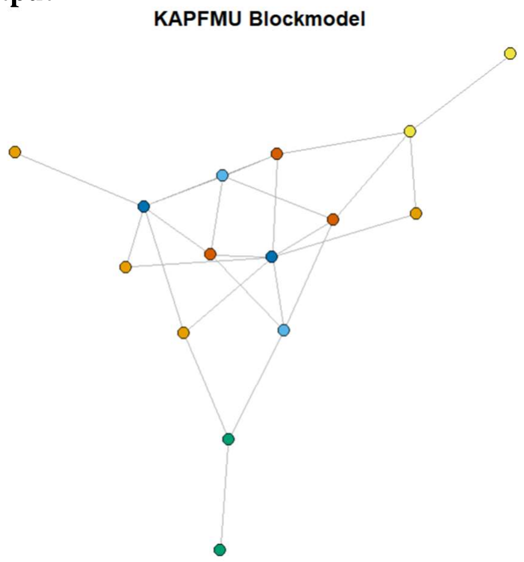


R code for blockmodeling kapfmu



```
mat1 <- as.matrix(get.adjacency(kapfmu))

# Estimate blockmodel partitions with k = 6 using the optRandomParC command
class6 <- optRandomParC(M=mat1, k=6, rep=10, approach="ss", blocks="com")
class6

# Retrieve the best partition from the blockmodeling result
best_partition1 <- class6$best$best1$clu
best_partition1

# Assign the block designations to the igraph object 'kapfmu'
V(kapfmu)$block <- best_partition1
V(kapfmu)$block

# Check if the number of blocks exceeds the default R color palette
if (max(best_partition) > length(colors())) {
  cat("Warning: Number of blocks exceeds the number of default colors in R.\n")
}

# Plot the graph with vertices colored by their block designation
par(mar=c(1,1,1,1), mfrow=c(1,1))
plot(kapfmu,
     vertex.label=NA,  # Hide vertex labels for a cleaner plot
     vertex.size=5,  # Set vertex size
     edge.arrow.size=.5,  # Set edge arrow size
     vertex.color=V(kapfmu)$block,  # Color vertices by block
     main="KAPFMU Blockmodel ")  # Title for the plot
```

Output



References:

Doreian, P. (1979). *On the evolution of group and network structure, Social Networks*. *Volume 2*(Issue 3), 235–252. https://doi.org/https://doi.org/10.1016/0378-8733(79)90016-9.

Qing, H., & Wang, J. (2023). *Community detection for weighted bipartite networks, Knowledge-Based Systems*. *Volume 274*. https://doi.org/https://doi.org/10.1016/j.knosys.2023.110643

Kapferer B. (1969). Norms and the manipulation of relationships in a work context. In J Mitchell (ed), Social networks in urban situations. Manchester: Manchester University Press.

Doreian P. (1974). On the connectivity of social networks. Journal of Mathematical Sociology, 3, 245-258.