realignment

February 24, 2020

repository: https://github.com/MLBurnham/networks_replication

citation: Gaumont, Noe, Maziyar Panahi, and David Chavalarias. "Reconstruction of the socio-semantic dynamics of political activist Twitter networks—Method and application to the 2017 French presidential election." PloS one 13, no. 9 (2018).

[1]:
```
library(igraph) # plotting
library(dplyr) # data manipulation
library(ggsci) # colors

# experimental packages for community detection
# if installing from Linux see: https://github.com/conda-forge/
 ↪r-devtools-feedstock/issues/4
#devtools::install_github("analyxcompany/resolution")
#library(resolution)
#library(NetworkToolbox)
```

```
Attaching package: 'igraph'

The following objects are masked from 'package:stats':

    decompose, spectrum

The following object is masked from 'package:base':

    union


Attaching package: 'dplyr'

The following objects are masked from 'package:igraph':

    as_data_frame, groups, union

The following objects are masked from 'package:stats':

    filter, lag
```

```
The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

1 Read in data

The data consists of two weighted edge lists representing two periods of time. The initial state is a retweet network of twitter accounts engaging in political discussion prior to the first round of elections. The second is a similar network after the first round of elections.

[2]:
```
a4df <- read.csv('../dataverse_files/A4 File␣
 ↪-Politoscope_RetweetGraph_2017-04-09_2017-04-23_min_3_anon.csv', sep = '')
a5df <- read.csv('../dataverse_files/A5 File -␣
 ↪Politoscope_RetweetGraph_2017-04-24_2017-05-08_min_3_anon.csv', sep = '')
# remove this single row causing a bug in igraph....
a5df <- a5df[-322012,]
```

2 Identify Communities

A nodes political community is the primary attribute of interest. The authors used a louvain algorithm with a resolution of 1 to identify communities. They did not provide their original classification, however, so I used to igraph's implementation to recreate their results.

A sizable portion of nodes do not belong to a clear community. The "sea" is currently defined as anything outside of the top 10 communities, although the authors defined it around candidates and political figures. This will be the next step in replication.

[3]:
```
# create a graph object
a4 <- graph_from_data_frame(d = a4df, directed = FALSE)
a5 <- graph_from_data_frame(d = a5df, directed = FALSE)
# Identify clusters
a4clusters <- cluster_louvain(a4)
a5clusters <- cluster_louvain(a5)
```

[4]:
```
# assign group membership to the cluster attribute of vertices
V(a4)$cluster <- membership(a4clusters)
V(a5)$cluster <- membership(a5clusters)

# create data frame of nodes and their cluster
a4clusterdf <- data.frame(vertex=V(a4)$name, cluster=V(a4)$cluster,␣
 ↪stringsAsFactors=FALSE)
a5clusterdf <- data.frame(vertex=V(a5)$name, cluster=V(a5)$cluster,␣
 ↪stringsAsFactors=FALSE)
```

[5]:
```
# get the 10 largest clusters
a4topclust <- sort(sizes(a4clusters), decreasing = TRUE)[1:10] %>%
                data.frame()
a4topclust <- a4topclust$Community.sizes

# repeat for a5
a5topclust <- sort(sizes(a5clusters), decreasing = TRUE)[1:10] %>%
                data.frame()
a5topclust <- a5topclust$Community.sizes
```

[7]:
```
# create df of top clusters and their assigned colors, then merge with the␣
 ↪cluster df
a4colordf <- data.frame(cluster=as.numeric(as.character(a4topclust)),␣
 ↪color=pal_jco()(10), stringsAsFactors = FALSE) %>%
                right_join(a4clusterdf, by = 'cluster') %>%
                transform(vertex = as.numeric(vertex))
# assign clusters outside of the top 10 to white
a4colordf$color[is.na(a4colordf$color)] <- 'white'
# reorder columns and export as nodes csv
a4colordf <- a4colordf[c('vertex', 'cluster', 'color')]
write.csv(a4colordf, 'a4_nodes.csv', row.names = FALSE)

# repeat for a5
a5colordf <- data.frame(cluster=as.numeric(as.character(a5topclust)),␣
 ↪color=pal_jco()(10), stringsAsFactors = FALSE) %>%
                right_join(a5clusterdf, by = 'cluster') %>%
                transform(vertex = as.numeric(vertex))
a5colordf$color[is.na(a5colordf$color)] <- 'white'
a5colordf <- a5colordf[c('vertex', 'cluster', 'color')]
write.csv(a5colordf, 'a5_nodes.csv', row.names = FALSE)
```

[8]:
```
# recreate graphs with new vertex data
a4 <- graph_from_data_frame(d = a4df, vertices = a4colordf, directed = FALSE)
a5 <- graph_from_data_frame(d = a5df, vertices = a5colordf, directed = FALSE)
```

[9]:
```
# define the 'sea' as nodes outside of the top 10 communities and remove it␣
 ↪from the graph
a4sea <- a4colordf[a4colordf$color == 'white',]$vertex
a4sealess <- delete_vertices(a4, as.character(a4sea))

a5sea <- a5colordf[a5colordf$color == 'white',]$vertex
a5sealess <- delete_vertices(a5, as.character(a5sea))
```

3 Distribution of nodes by community

[10]:
```
print("Number of unique nodes in initial state:")
length(unique(a4colordf$vertex))

print("Number of unique nodes after realignment:")
length(unique(a5colordf$vertex))
```

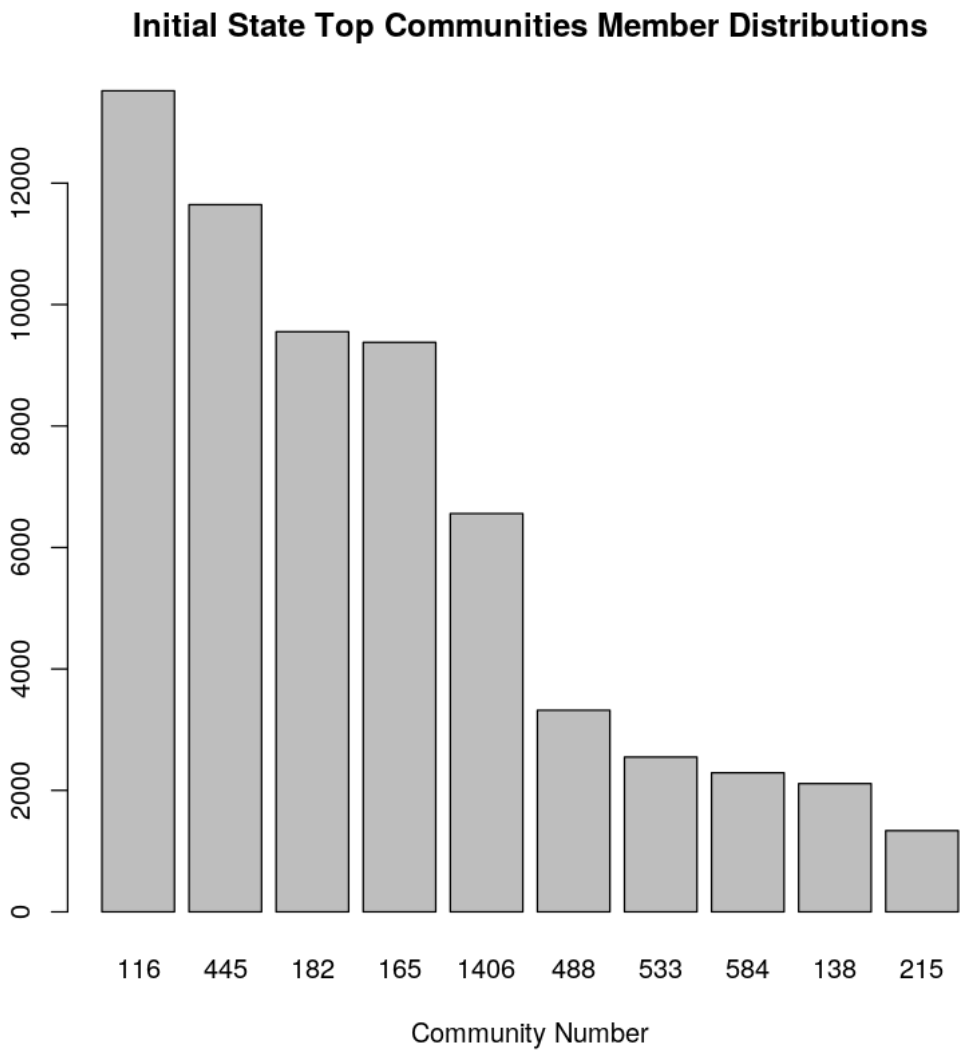
```
[1] "Number of unique nodes in initial state:"
```

69276

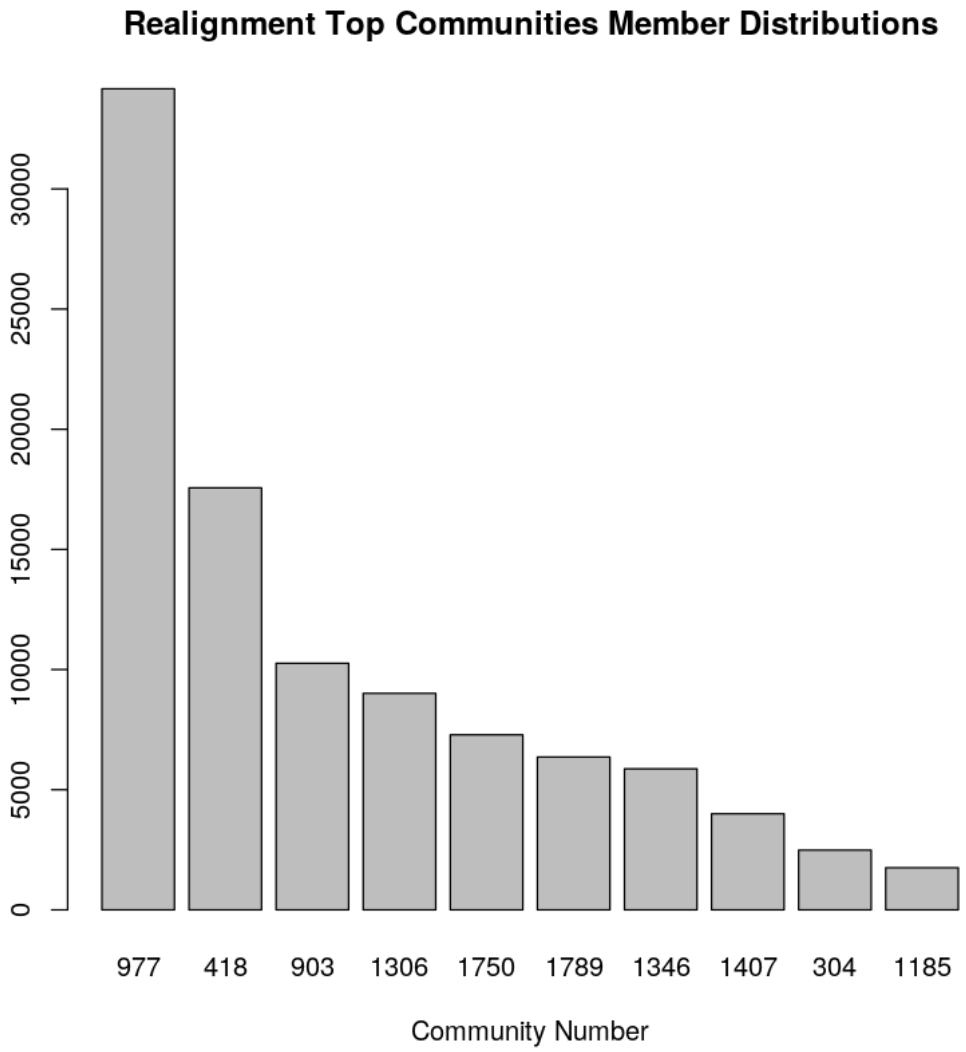
```
[1] "Number of unique nodes after realignment:"
```

115020

[11]:
```
a4top10 <- a4colordf[a4colordf$color != 'white',]
a4table <- table(a4top10$cluster)

barplot(a4table[order(a4table, decreasing=TRUE)],
       main = 'Initial State Top Communities Member Distributions',
       xlab = 'Community Number')
```



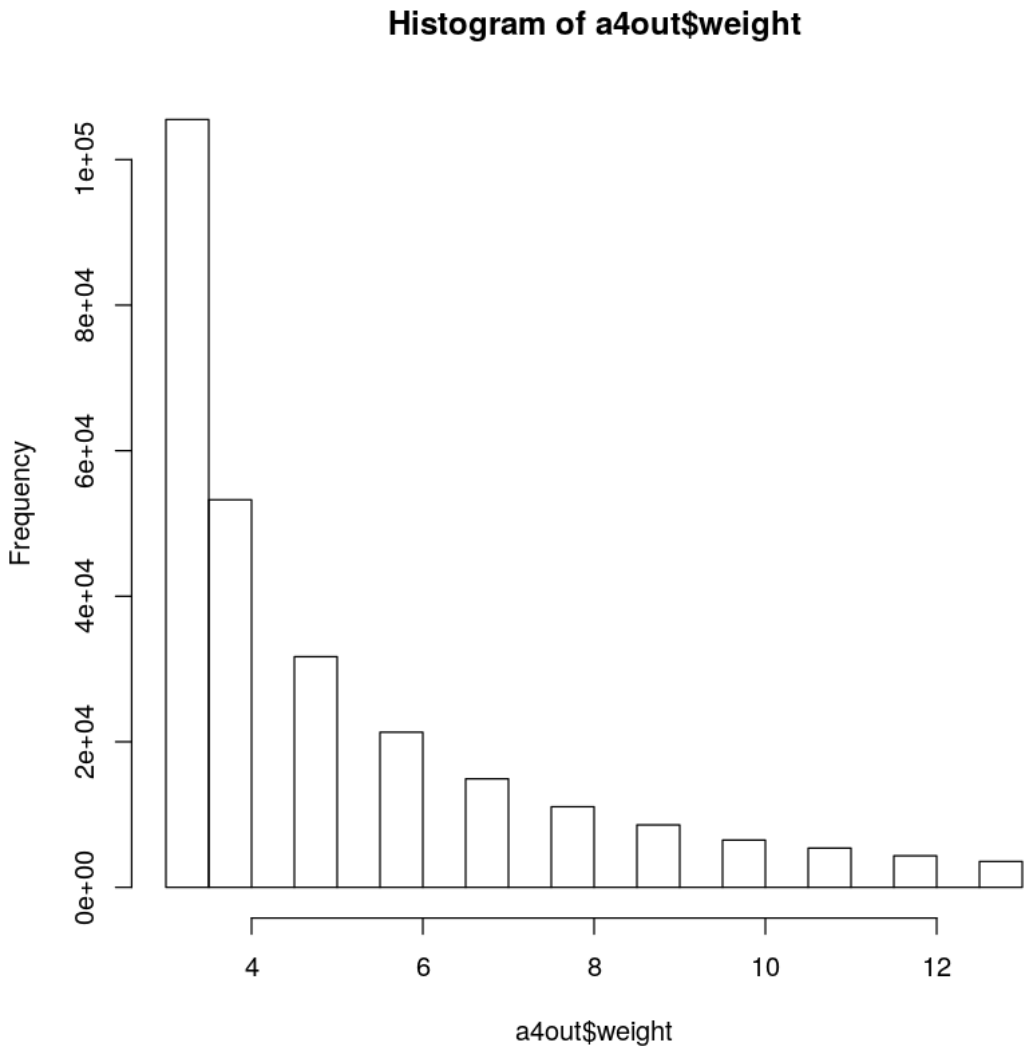
[12]:
```
a5top10 <- a5colordf[a5colordf$color != 'white',]
a5table <- table(a5top10$cluster)

barplot(a5table[order(a5table, decreasing=TRUE)],
       main = 'Realignment Top Communities Member Distributions',
       xlab = 'Community Number')
```

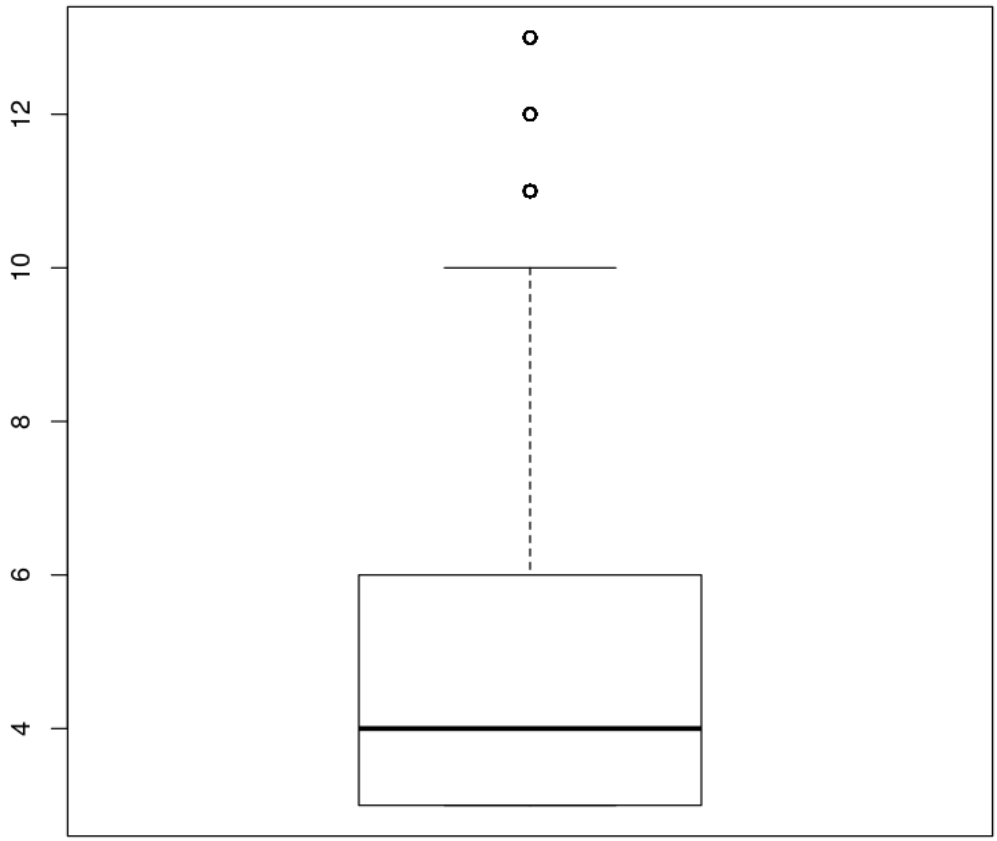


4 Edges Summary

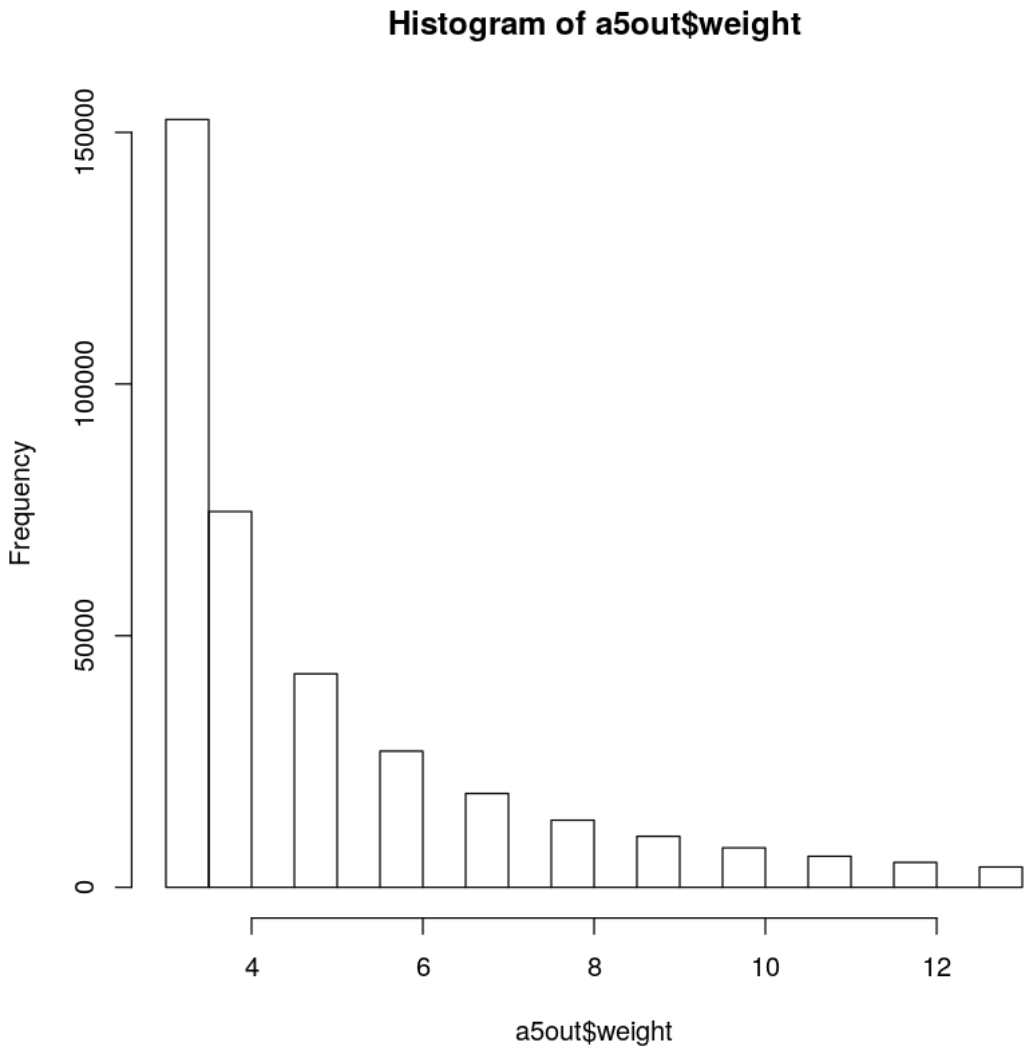
The distribution of edge weights look almost identical between the two time periods

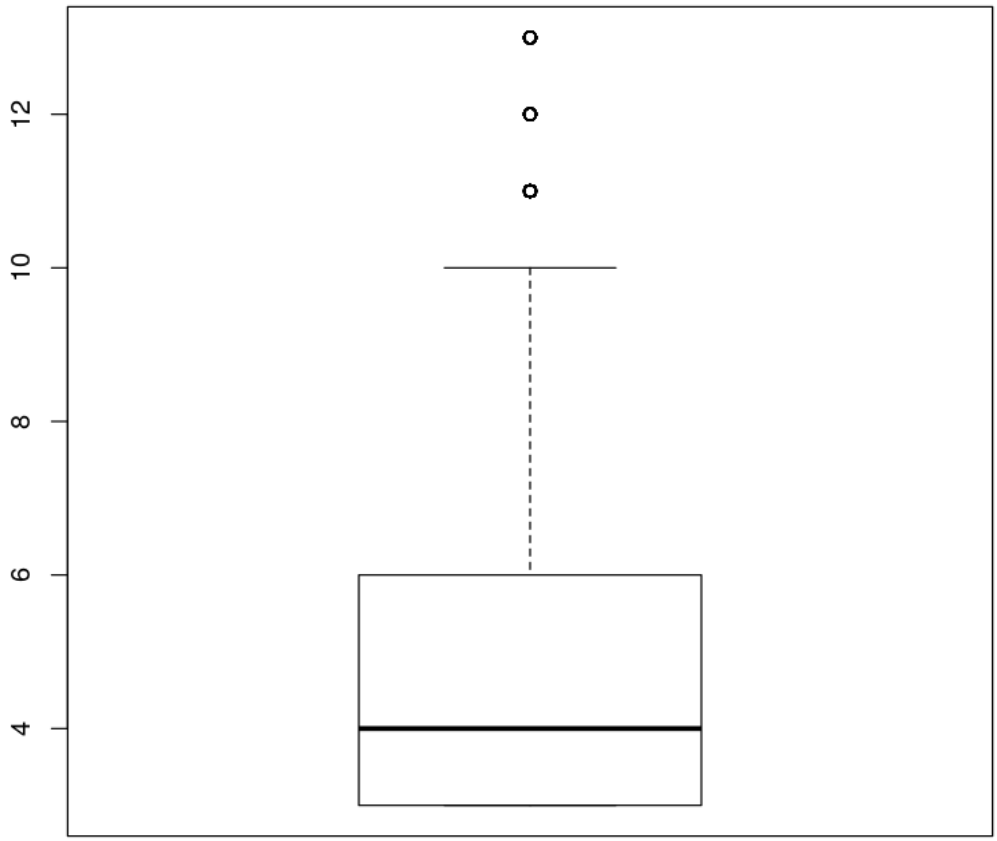
[13]:
```
# Extreme outliers significantly distort visualisations of the distributions
# create a boxplot and histogram of weights, minus outliers
outliers <- boxplot(a4df$weight, plot=FALSE)$out
a4out <- a4df[-which(a4df$weight %in% outliers),]
hist(a4out$weight)
boxplot(a4out$weight)
```


Histogram of a4out$weight
Frequency
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05
4
6
8
10
12
a4out$weight



[14]:
```
# repeat for a5
outliers <- boxplot(a5df$weight, plot=FALSE)$out
a5out <- a5df[-which(a5df$weight %in% outliers),]
hist(a5out$weight)
boxplot(a5out$weight)
```


Histogram of a5out$weight
Frequency
0
50000
100000
150000
4
6
8
10
12
a5out$weight



5 Visualize

igraph replication of the authors graphs representing initial and post election states. The authors use the Atlasforce2 algorithm which is currently unavailable in R. I instead us FR

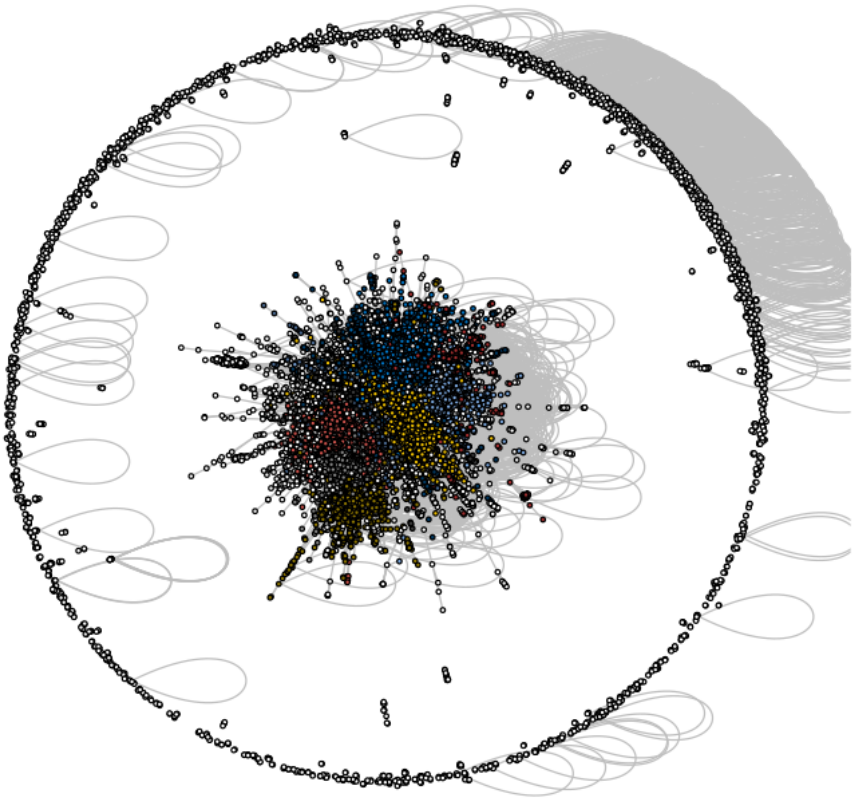
5.1 Initial visualization with the "sea" included.

[239]:
```
# calculate FR layout coordinates, takes 2-4 hours to run for each layout

#a4fr <- layout_with_fr(a4, grid = 'nogrid')
```

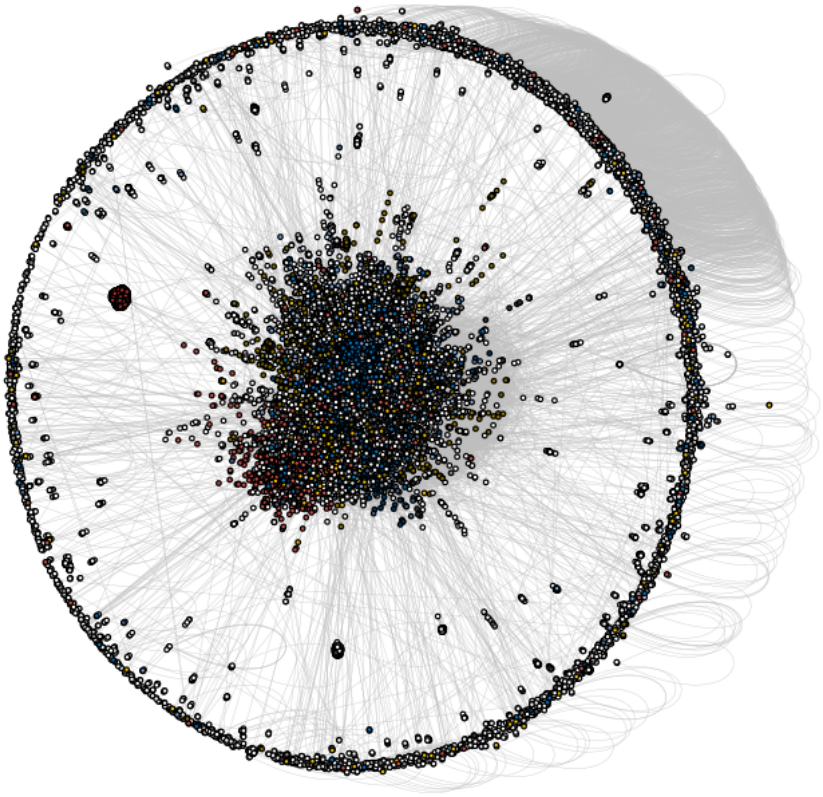
```
#save(a4fr, file = "a4_fr_layout.RData")

#a5fr <- layout_with_fr(a5, grid = 'nogrid')
#save(a5fr, file = "a5_fr_layout.RData")
```

[16]:
```
# loard in graph layout coordinates
load('a4_fr_layout.RData')
load('a5_fr_layout.RData')
```

[17]:
```
# plot a4, before realignment. white nodes = "the sea", nodes that fall outside␣
 ↪of the 10 most populous groups
plot.igraph(a4,
     layout = a4fr,
     vertex.label = NA,
     vertex.size = 1.25,
     vertex.frame.width = .25,
     edge.arrow.size = .1,
     edge.color ='gray',
     edge.widght = .25
    )
```



[18]:
```
# plot a5, after realignment
# edges are now connected to the central graph and more clusters are formed in␣
 ↪the middle
# you can also see colors appearing on the edge.
plot.igraph(a5,
     layout = a5fr, # set layout
     vertex.label = NA,
     vertex.size = 1.25,
     vertex.frame.width = .25,
     edge.color ='gray',
     edge.width = .25
    )
```



5.2 Visualization with the “sea” excluded

[250]:
```
# recalculate coordinates for "sealess" layout

#a4sealessfr <- layout_with_fr(a4sealess, grid = 'nogrid')
#save(a4sealessfr, file = "a4sealess_fr_layout.RData")

#a5sealessfr <- layout_with_fr(a5sealess, grid = 'nogrid')
#save(a5sealessfr, file = "a5sealess_fr_layout.RData")
```

[19]:
```
load('a4sealess_fr_layout.RData')
load('a5sealess_fr_layout.RData')
# remove loops
a4simp <- simplify(a4sealess, remove.multiple=FALSE, remove.loops = TRUE)
a5simp <- simplify(a5sealess, remove.multiple=FALSE, remove.loops = TRUE)
```

[20]:
```
# plot a4sealess
plot.igraph(a4simp,
     layout = a4sealessfr,
     vertex.label = NA,
     vertex.size = 1,
     vertex.frame.color = NA,
     edge.color ='gray',
     edge.width = .25
    )
```

[21]:
```
# plot a5sealess
plot.igraph(a5simp,
     layout = a5sealessfr,
     vertex.label = NA,
     vertex.size = 1,
     vertex.frame.color = NA,
     edge.color ='gray',
     edge.width = .25,
    )
```