Social Network Analysis Home Assignment 2

Lev Mazaev

due date - 25.05.2019 23:59

Contents

Graph models. Centrality metrics

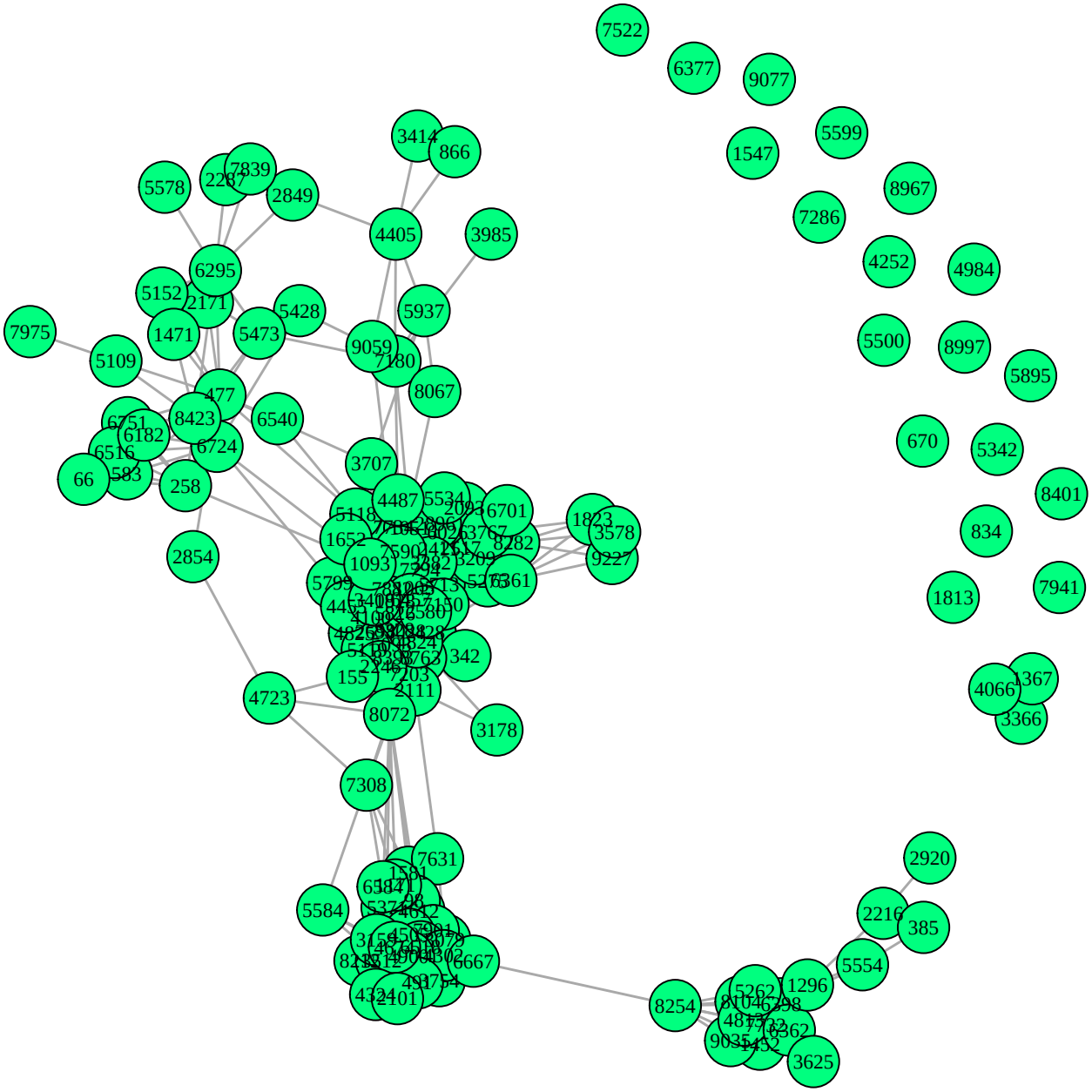
Loading required libraries

```
library(igraph)
library(rgexf)
library(ggplot2)
library(R.matlab)
library(gridExtra)
library(corrplot)
```

Task 1. Your social network

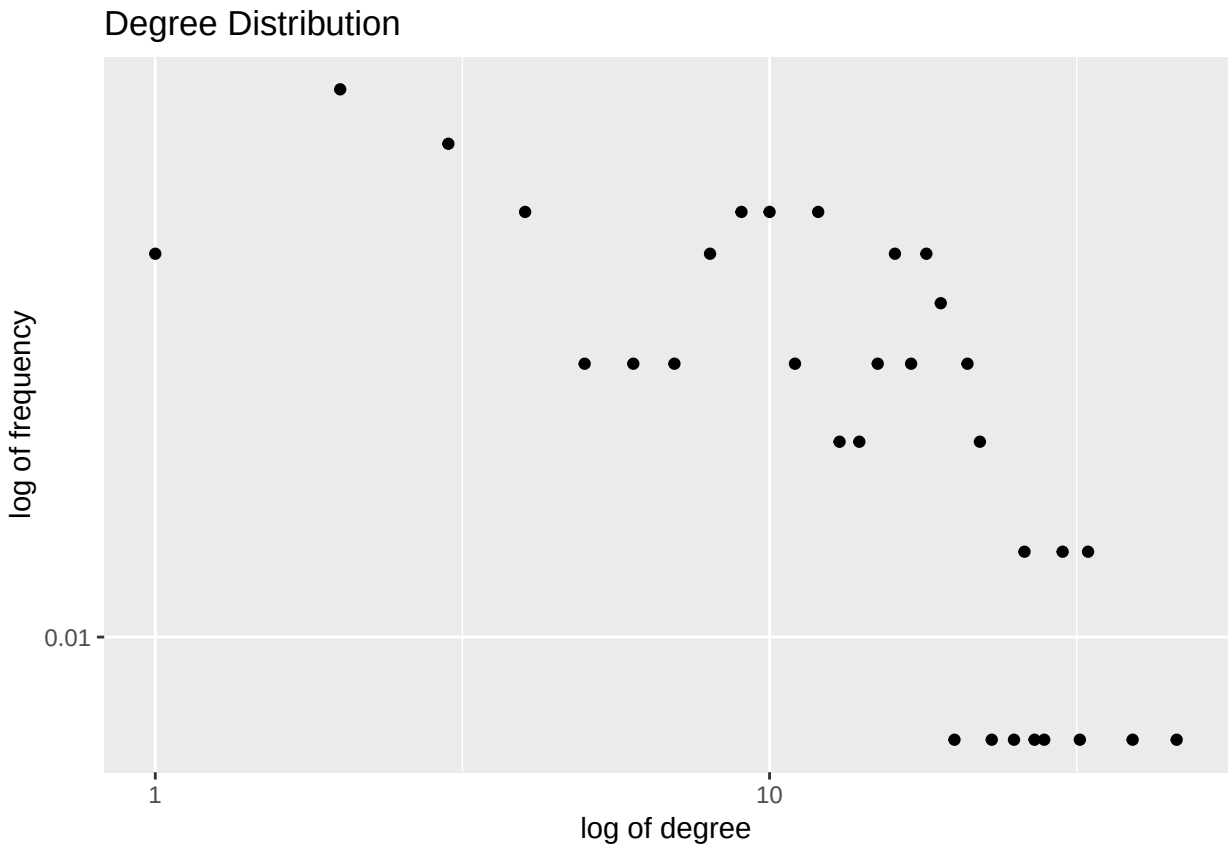
Loading and plotting social network

```
xEN <- read.gexf(x = 'vk_ego_network.gexf')
egoNetwork <- gexf.to.igraph(gexf.obj = xEN)
# anonymization by taking hash of ID
set.seed(42)
dfEgoNet <- data.frame(names = as.integer(xEN$nodes$id) %% sample(1000:10000, 1),
                       stringsAsFactors = FALSE)
egoNetwork <- set_vertex_attr(graph = egoNetwork, name = 'name',
                              value = dfEgoNet$names)
# the mentioned centralities must be computed on the undirected graph
# also ego-networks are usually undirected graphs
egoNetwork <- as.undirected(graph = egoNetwork)
# making it a bit more nice-looking
V(egoNetwork)$color <- 'springgreen'
V(egoNetwork)$size <- 10
V(egoNetwork)$label.cex <- 0.75
V(egoNetwork)$label.color <- 'black'
E(egoNetwork)$width <- 1.5
plot.igraph(egoNetwork, layout = layout.fruchterman.reingold)
```



1. Degree distribution

```
deg <- degree(egoNetwork)
degDist <- degree.distribution(graph = egoNetwork)[-1]
# removing the first entry because it is for degree 0
deg <- sort(unique(deg))[-1]
qplot(x = deg, y = degDist[deg], log = 'xy',
      main = 'Degree Distribution',
      xlab = 'log of degree',
      ylab = 'log of frequency')
```



Is there any correspondence between actual degree distribution of your network and the Power Law distribution? If not, explain why.

Just observing the plot it's hard to say whether the degree is distributed according to the Power Law or not. Let's use `power.law.fit` function.

```
fitPL <- power.law.fit(x = degree(egoNetwork), implementation = 'plfit')
pval <- round(fitPL$KS.p, 4)
fitPL$xmin; fitPL$alpha
```

```
## [1] 17
```

```
## [1] 4.232775
```

The p-value (according to KS test) is 0.9991 which means the degree is distributed according to the Power Law, but what causes doubts is $x_{min} = 17$. Only the second half of possible degrees vector is covered by such fit. Let's run the same for $x_{min} = 1$:

```
fitPL <- power.law.fit(x = degree(egoNetwork), xmin = 1, implementation = 'plfit')
pval <- round(fitPL$KS.p, 4)
```

The p-value is 0 (4 digits rounding), so in this case the degree distribution is significantly different from the fitted Power Law distribution. Also let's take some value in the middle, say $x_{min} = 9$.

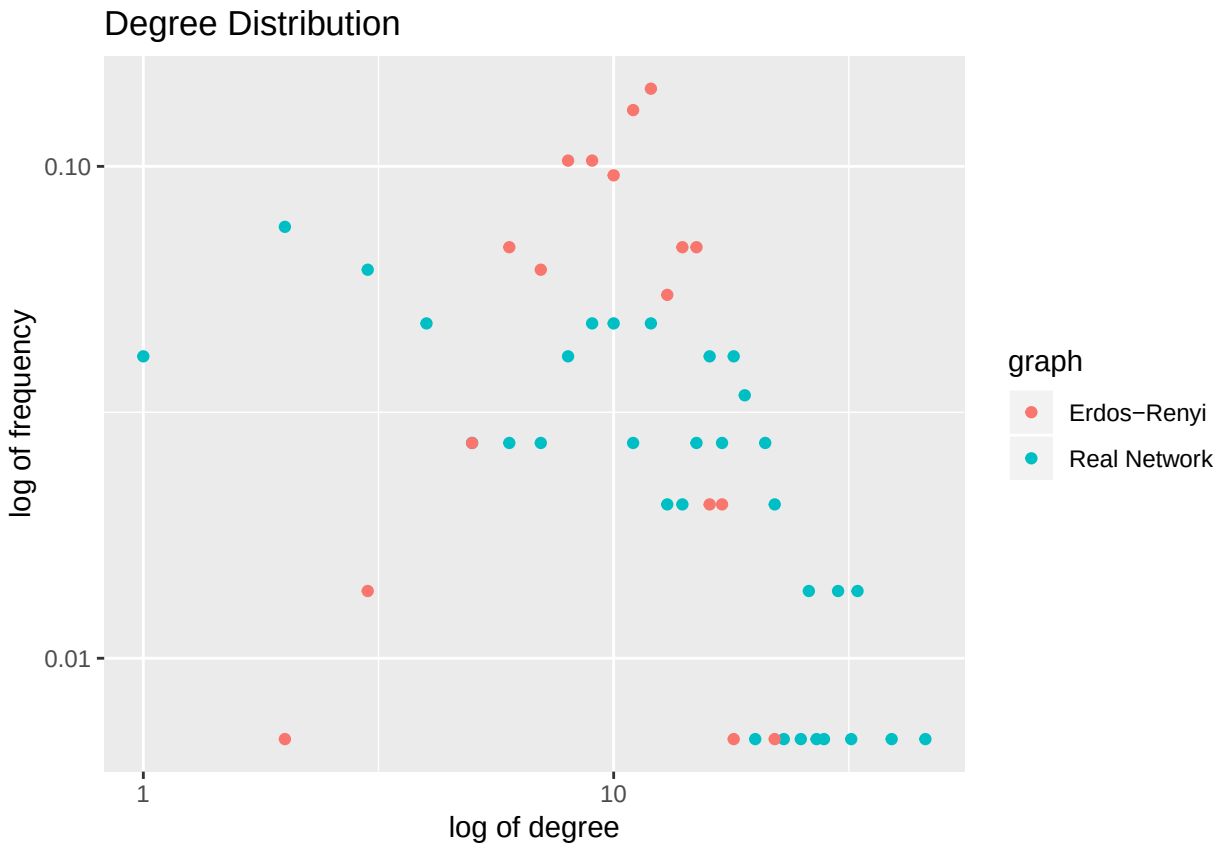
```
fitPL <- power.law.fit(x = degree(egoNetwork), xmin = 9, implementation = 'plfit')
pval <- round(fitPL$KS.p, 4)
```

The p-value is 0.0274. Again, the degree distribution significantly differs from the fitted Power Law distribution.

Thus, we cannot conclude about the degree distribution of this ego-network. Probably the graph is too small to make such inference.

Now, Erdos-Renyi graph matching our network (same number of nodes and same average degree).

```
erGraph <- erdos.renyi.game(n = vcount(egoNetwork),
                            p.or.m = ecount(egoNetwork),
                            type = 'gnm')
degER <- sort(unique(degree(erGraph)))
degER <- degER[degER > 0]
degDistER <- degree.distribution(erGraph)[-1]
df <- data.frame(
  deg = c(deg, degER),
  freq = c(degDist[deg], degDistER[degER]),
  graph = c(
    rep('Real Network', length(deg)),
    rep('Erdos-Renyi', length(degER))
    )
)
ggplot(df, aes(x = deg, y = freq, color = graph)) +
  geom_point() + scale_x_log10() + scale_y_log10() +
  xlab('log of degree') + ylab('log of frequency') +
  ggtitle('Degree Distribution')
```



Observing the plot we can conclude that the degree distribution of Erdos-Renyi random graph considerably differs from the degree distribution of our ego-network. The degree in Erdos-Renyi graph is distributed according to Poisson law.

2. Computing centrality metrics

Firstly, fixing the layout.

```
layOut <- layout.fruchterman.reingold(egoNetwork)
```

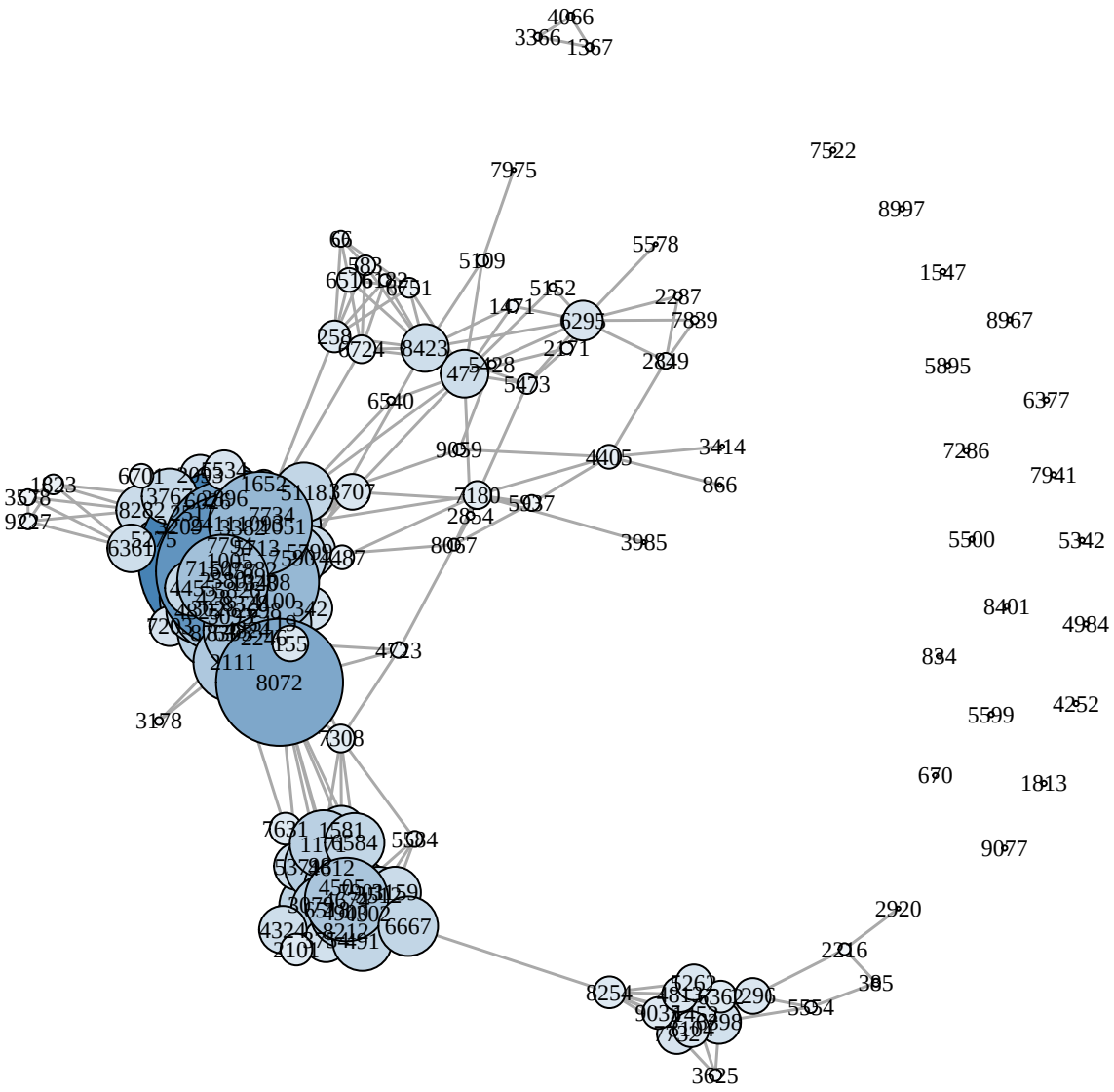
Secondly, setting up the colors.

```
fine <- 4 * vcount(egoNetwork)
colorPalette <- colorRampPalette(c('white', 'steelblue'))
```

Degree Centrality:

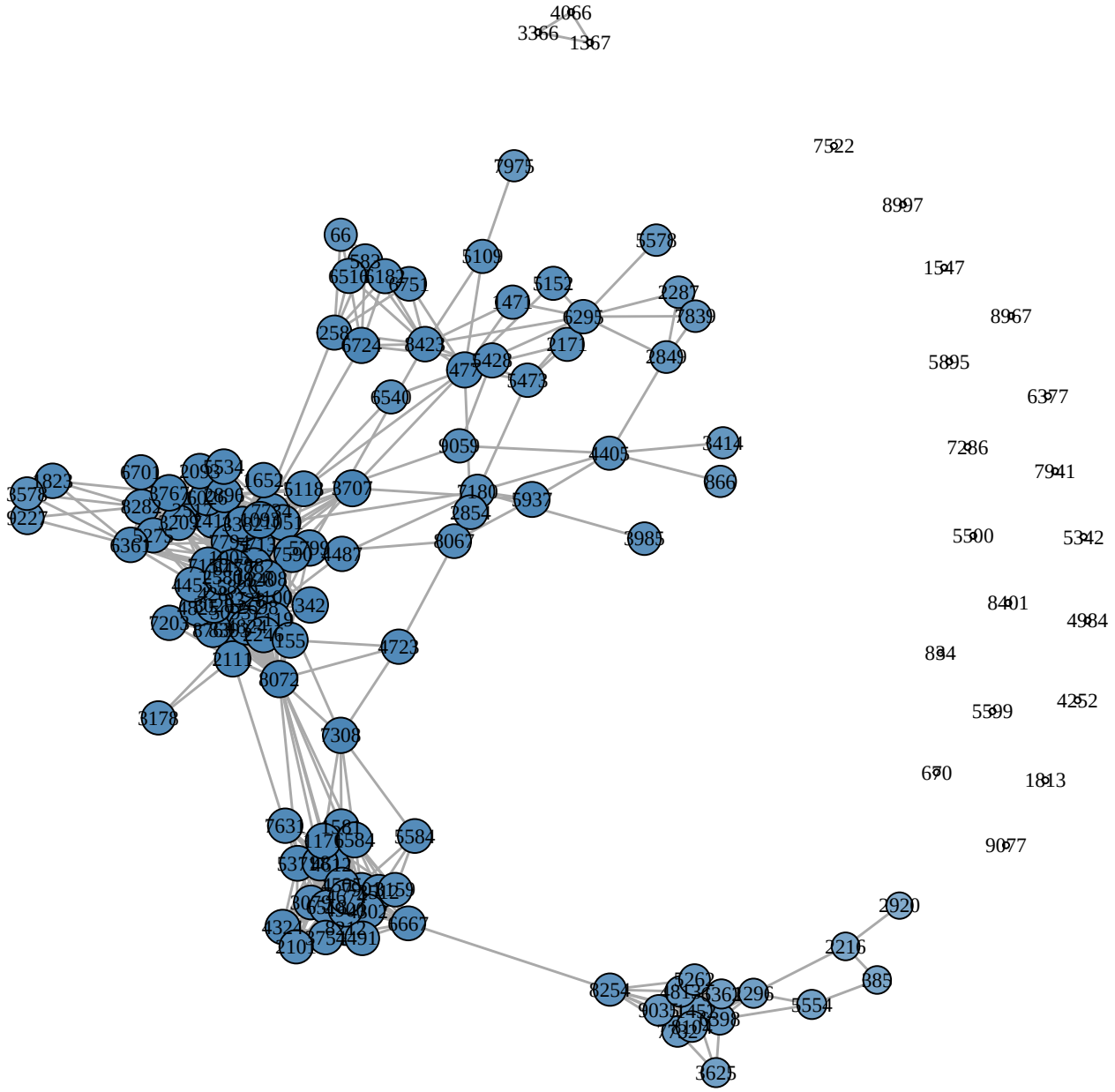
```
dfEgoNet$degree <- degree(graph = egoNetwork)
degColor <- colorPalette(fine)[as.numeric(cut(dfEgoNet$degree, breaks = fine))]
plot.igraph(egoNetwork, layout = layOut,
            vertex.color = degColor,
            vertex.size = dfEgoNet$degree * 0.75,
            main = 'Degree Centrality')
```

Degree Centrality



Closeness Centrality:

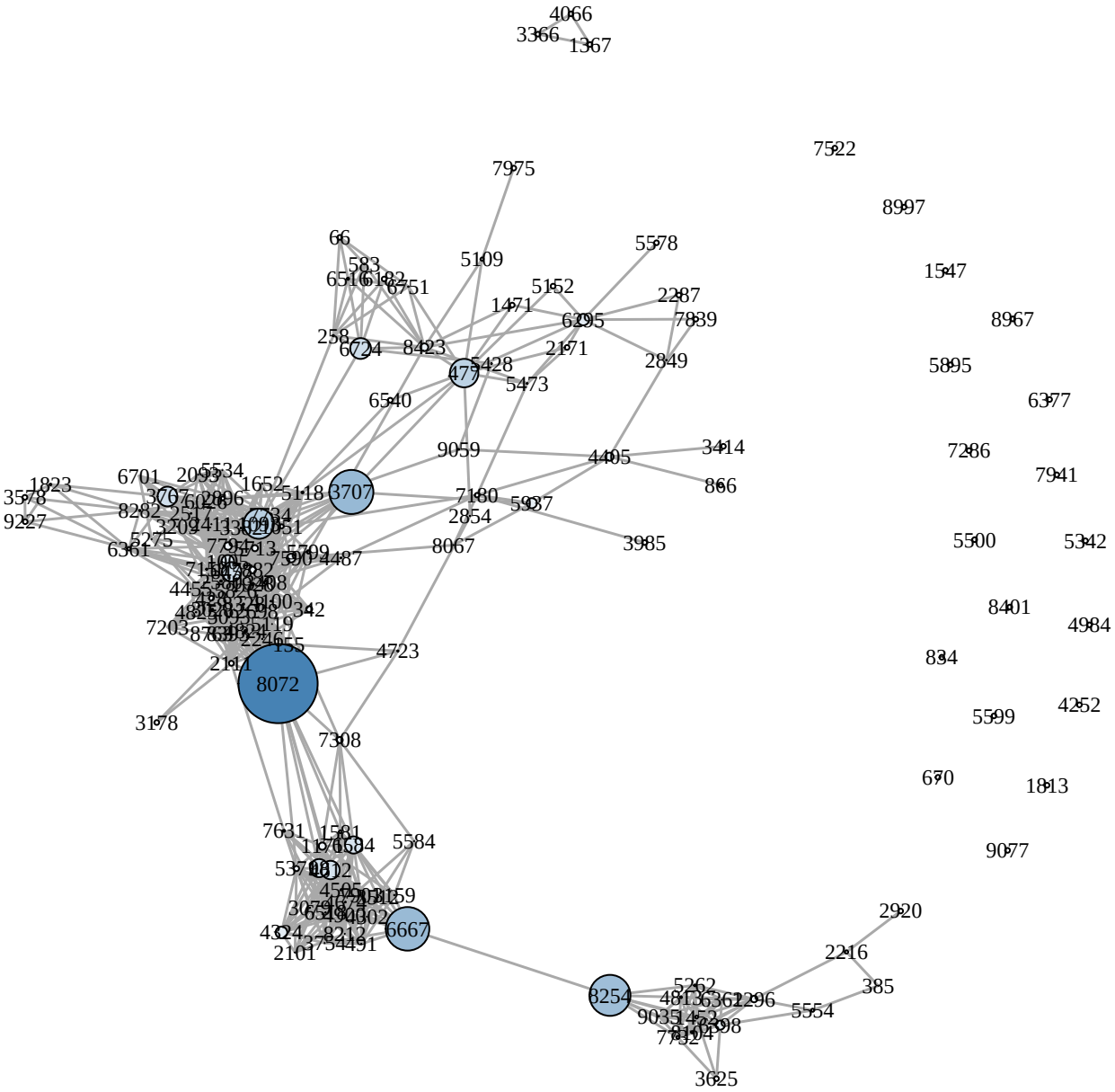
```
dfEgoNet$closeness <- closeness(graph = egoNetwork)
closenessColor <- colorPalette(fine)[as.numeric(cut(dfEgoNet$closeness, breaks = fine))]
plot.igraph(egoNetwork, layout = layOut,
            vertex.color = closenessColor,
            vertex.size = dfEgoNet$closeness * 25000,
            main = 'Closeness Centrality')
```



Betweenness Centrality:

```
dfEgoNet$betweenness <- betweenness(graph = egoNetwork)
betweennessColor <- colorPalette(fine)[as.numeric(cut(dfEgoNet$betweenness, breaks = fine))]
plot.igraph(egoNetwork, layout = layOut,
            vertex.color = betweennessColor,
            vertex.size = dfEgoNet$betweenness * 0.005,
            main = 'Betweenness Centrality')
```

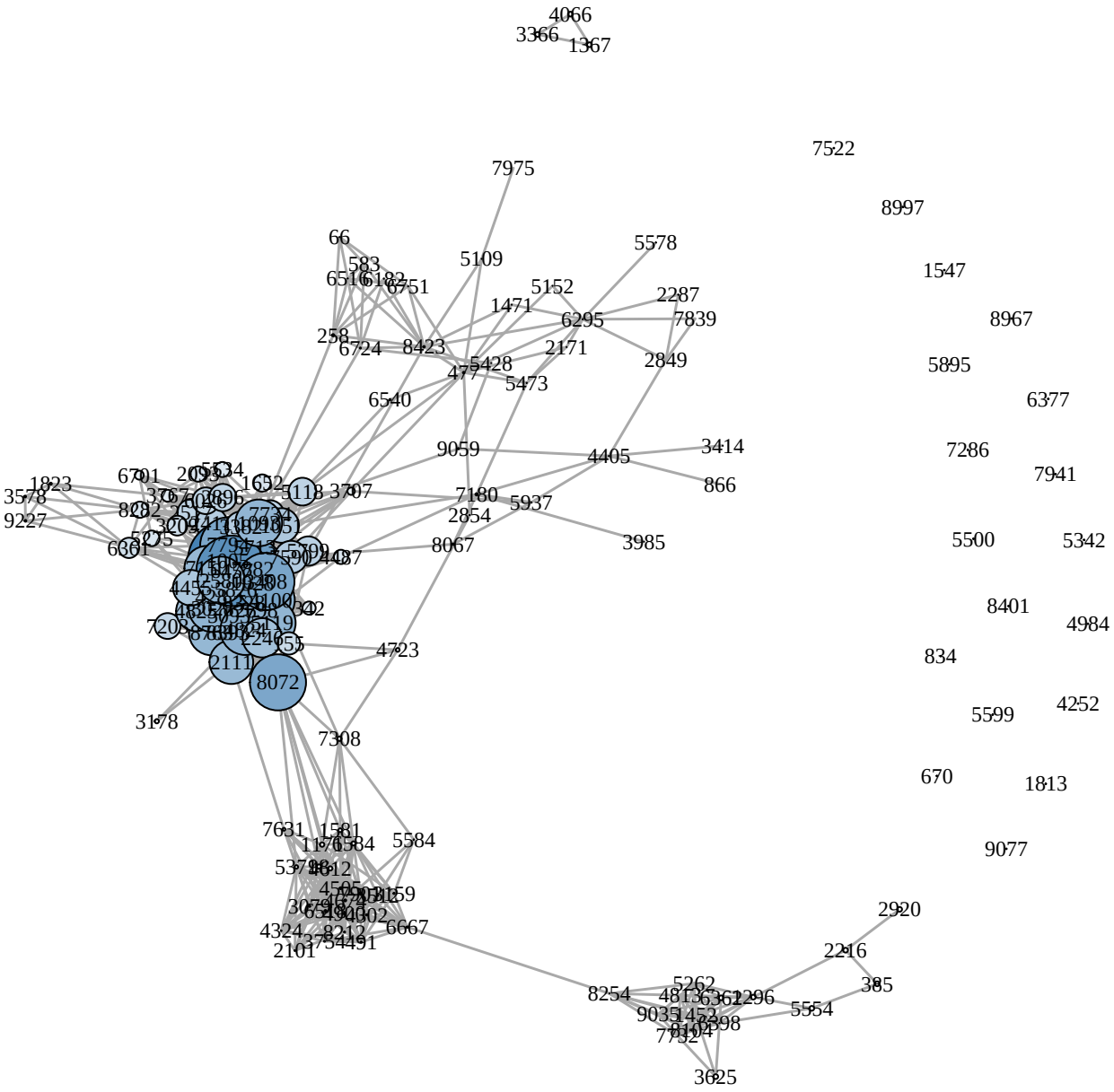
Betweenness Centrality



Eigenvector Centrality:

```
dfEgoNet$ev <- eigen_centrality(graph = egoNetwork)$vector
evColor <- colorPalette(fine)[as.numeric(cut(dfEgoNet$ev, breaks = fine))]
plot.igraph(egoNetwork, layout = layOut,
            vertex.color = evColor,
            vertex.size = dfEgoNet$ev * 15,
            main = 'Eigenvector Centrality')
```

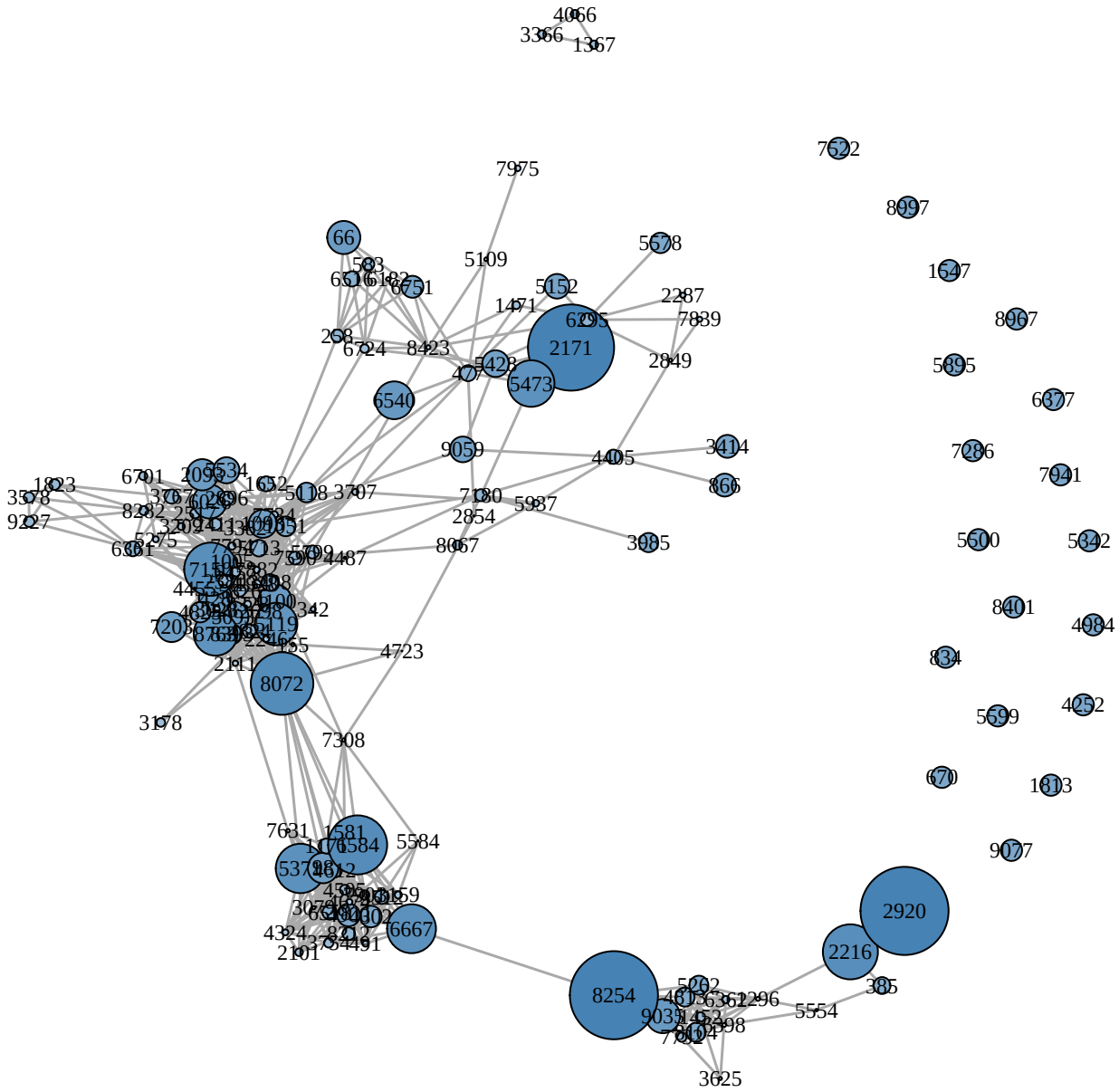
Eigenvector Centrality



Bonacich Power Centrality:

```
dfEgoNet$bon <- bonpow(graph = egoNetwork)
bonColor <- colorPalette(fine)[as.numeric(cut(dfEgoNet$bon, breaks = fine))]
# because of negative values vertex.size doesn't work
# the values must of bon be transformed to positive ones
plot.igraph(egoNetwork, layout = layOut,
            vertex.color = bonColor,
            vertex.size = 4 * exp(dfEgoNet$bon),
```

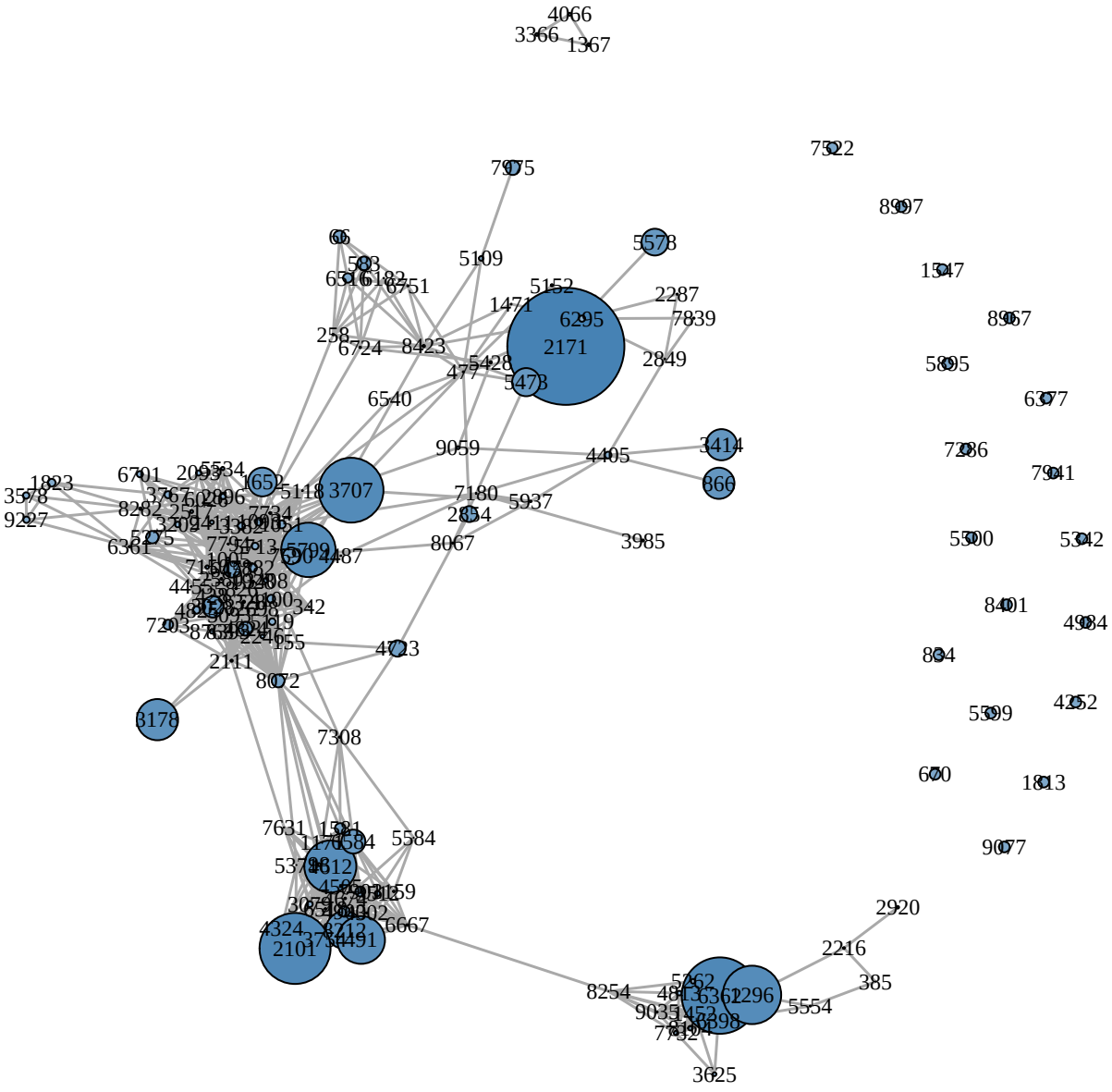
```
            main = 'Bonacich Power Centrality')
```



Alpha Centrality:

```
dfEgoNet$alpha <- alpha_centrality(graph = egoNetwork)
alphaColor <- colorPalette(fine)[as.numeric(cut(dfEgoNet$alpha, breaks = fine))]
# because of negative values vertex.size doesn't work
# the values must of alpha be transformed to positive ones
plot.igraph(egoNetwork, layout = layOut,
```

```
            vertex.color = alphaColor,
            vertex.size = 0.75 * exp(dfEgoNet$alpha),
            main = 'Alpha Centrality')
```



Top 10 nodes for each centrality:

```
top10 <- data.frame(
  Degree = head(dfEgoNet$names[order(dfEgoNet$degree, decreasing = TRUE)], n = 10L),
  Closeness = head(dfEgoNet$names[order(dfEgoNet$closeness, decreasing = TRUE)], n = 10L),
  Betweenness = head(dfEgoNet$names[order(dfEgoNet$betweenness, decreasing = TRUE)], n = 10L),
```

```
  Eigenvector = head(dfEgoNet$names[order(dfEgoNet$ev, decreasing = TRUE)], n = 10L),
  Bonacich = head(dfEgoNet$names[order(dfEgoNet$bon, decreasing = TRUE)], n = 10L),
  Alpha = head(dfEgoNet$names[order(dfEgoNet$alpha, decreasing = TRUE)], n = 10L)
)
knitr::kable(top10)
```

Degree	Closeness	Betweenness	Eigenvector	Bonacich	Alpha
1005	6457	8072	1005	8254	2171
6457	8072	3707	6457	2920	6362
7794	1005	6667	5826	2171	2101
7882	1093	8254	7882	8072	3707
8072	3707	1093	1928	6584	1296
5826	428	477	8328	2216	5799
1928	7590	6724	3408	7150	4612
8328	3408	3767	428	5371	491
428	5713	6457	8072	6667	3178
3408	1928	4612	5093	5473	8212

Comment on your results here - for example, why some nodes have high betweenness centrality and the others have high closeness centrality. Is this what you would expect to see?

All the centralities behave differently. Most nodes located in dense clusters have high closeness because there a lot of nodes around them available at short distance, while remote or isolated nodes have low closeness since there is no path or it's very long. Nodes having high betweenness are usually bridging nodes located between clusters. These nodes have a bridging edge incident to them through which there are many shortest path. All these patterns definitely correspond to what we expect to see.

Task 2. Flickr network

Data contains sparse matrix A and list of user names. This is a denser part of the Flickr photo sharing site friendship graph from 2006. Edge direction corresponds to friendship requests (following). Some of the links are reciprocal, others not

```
flickr <- readMat(con = 'flickr.mat')
fmatrix <- as.matrix(flickr[1]$A)
fnames <- flickr[2]$names
fnames <- gsub(" ", "", fnames)
dFlickr <- data.frame(name = fnames, stringsAsFactors = FALSE)
```

Creating the graph:

```
flickrNet <- graph_from_adjacency_matrix(adjmatrix = fmatrix)
vcount(flickrNet)
```

```
## [1] 15724
```

```
ecount(flickrNet)
```

```
## [1] 510983
```

Computing in- and out-degree centralities, PageRank, Hubs and Authorities for this network:

```
dFlickr$inDegree <- degree(graph = flickrNet, mode = 'in')
dFlickr$outDegree <- degree(graph = flickrNet, mode = 'out')
dFlickr$pageRank <- page_rank(graph = flickrNet)$vector
dFlickr$hubs <- hub_score(graph = flickrNet)$vector
dFlickr$authority <- authority_score(graph = flickrNet)$vector
```

Print top ten names in each ranking:

```
top10 <- data.frame(
  'In-degree' = head(dFlickr$name[order(dFlickr$inDegree, decreasing = TRUE)], 10L),
  'Out-degree' = head(dFlickr$name[order(dFlickr$outDegree, decreasing = TRUE)], 10L),
  'Page Rank' = head(dFlickr$name[order(dFlickr$pageRank, decreasing = TRUE)], 10L),
  'Hubs' = head(dFlickr$name[order(dFlickr$hubs, decreasing = TRUE)], 10L),
  'Authority' = head(dFlickr$name[order(dFlickr$authority, decreasing = TRUE)], 10L),
  stringsAsFactors = FALSE, check.names = FALSE
)
knitr::kable(top10)
```

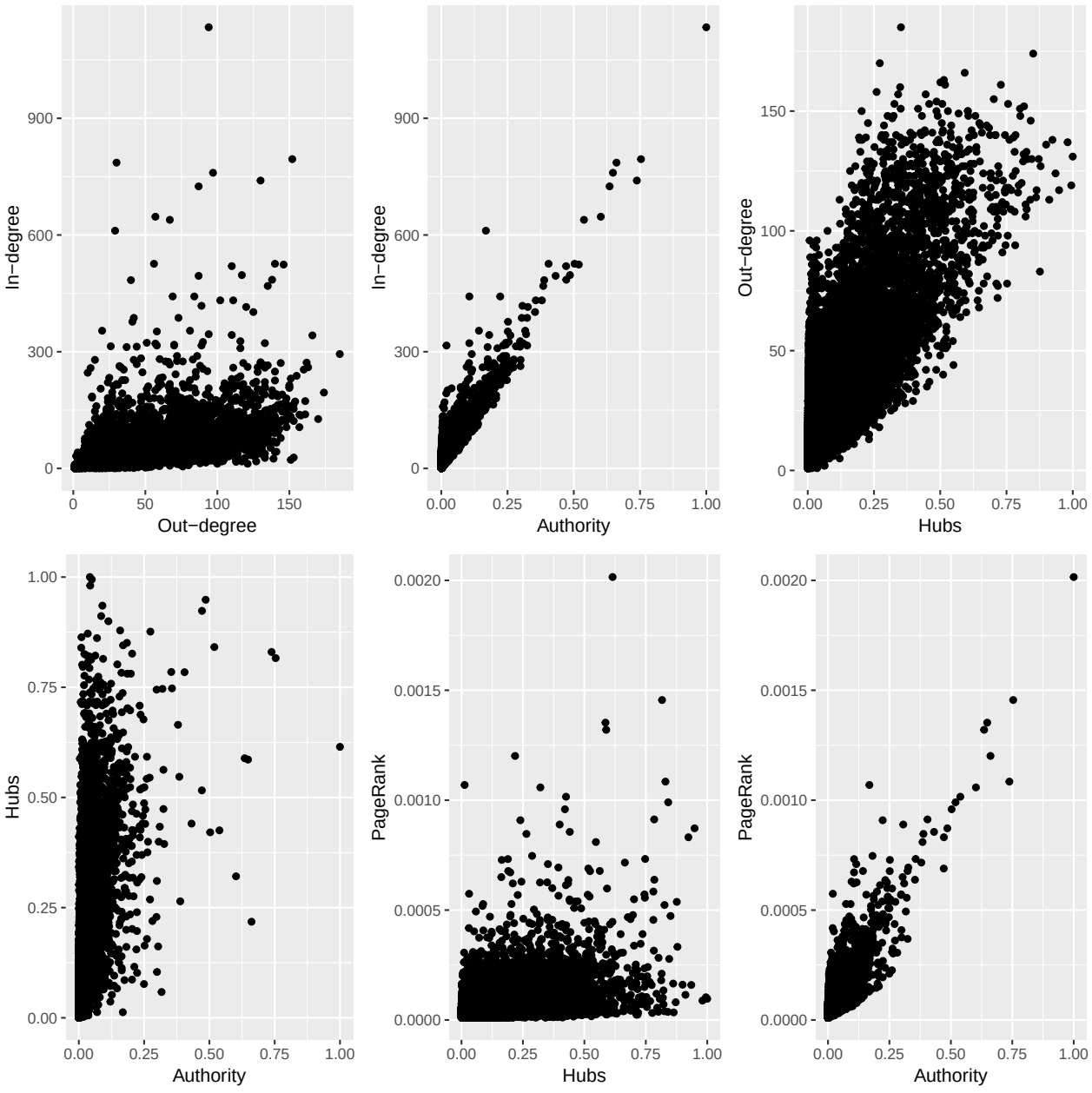
In-degree	Out-degree	Page Rank	Hubs	Authority
awfulsara	anildash	awfulsara	mrpiink	awfulsara
drp	tozzer	drp	automat	drp
Ivan	AtiRanA	antimethod	schizoo23	DrJoanne
antimethod	pixietart	BombDog	lorrainemd	*Ivan*
DrJoanne	jakedobkin	*Ivan*	sgoralnick	antimethod
BombDog	Buntekuh	DrJoanne	*starlet*	BombDog
SimonPais	brainware3000	MaDGi®L • ™	brynfoto	SimonPais
deborahlattimore	Jakes_World	SimonPais	liquidpixel	deborahlattimore
MaDGi®L • ™	maximolly	deborahlattimore	noahstone	cymagen
notraces	AndreiaLopes	cymagen	isherwood	:Nikola

Producing the following plots:

- In-degree centralities versus out-degree centralities
- In-degree centralities versus authorities
- Out-degree centralities versus hubs
- Hubs versus authorities
- PageRank versus hubs
- PageRank versus authorities

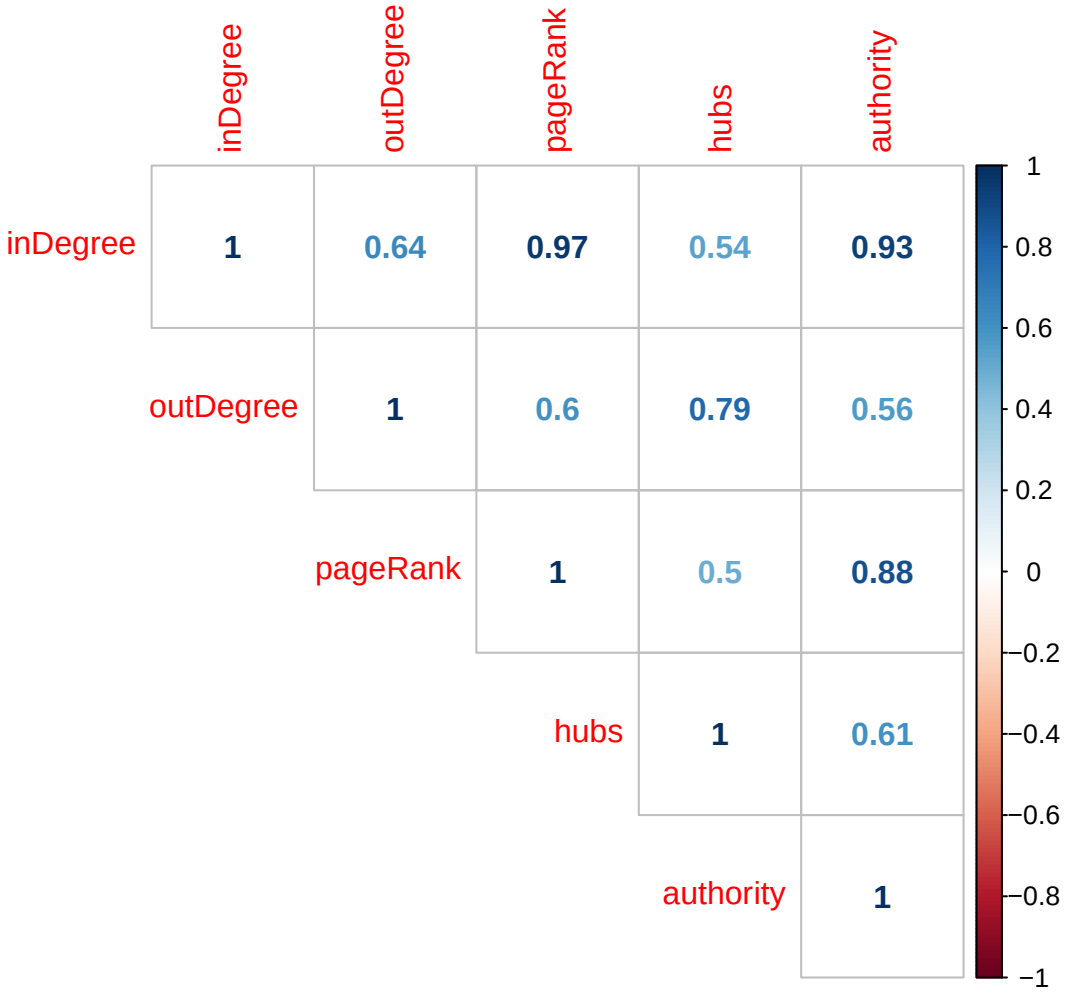
```
q1 <- qplot(x = dFlickr$outDegree, y = dFlickr$inDegree,
      xlab = 'Out-degree', ylab = 'In-degree')
q2 <- qplot(x = dFlickr$authority, y = dFlickr$inDegree,
      xlab = 'Authority', ylab = 'In-degree')
q3 <- qplot(x = dFlickr$hubs, y = dFlickr$outDegree,
      xlab = 'Hubs', ylab = 'Out-degree')
q4 <- qplot(x = dFlickr$authority, y = dFlickr$hubs,
      xlab = 'Authority', ylab = 'Hubs')
q5 <- qplot(x = dFlickr$hubs, y = dFlickr$pageRank,
      xlab = 'Hubs', ylab = 'PageRank')
q6 <- qplot(x = dFlickr$authority, y = dFlickr$pageRank,
```

```
      xlab = 'Authority', ylab = 'PageRank')
grid.arrange(q1, q2, q3, q4, q5, q6,
             nrow = 2, ncol = 3)
```



Additionally, pairwise correlation plot:

```
corrplot(cor(dFlickr[, 2:6]), method = 'number',
         type = 'upper', sig.level = .05,
         p.mat = cor.mtest(dFlickr[, 2:6], conf.level = 0.95)$p)
```




```
# with significance test
```

Comment on the relationships between different centrality metrics

According to the correlation test all possible pairs of these centrality metrics correlate **significantly**, but the magnitude differs between pairs. Among these scatterplots the high correlation ($R > 0.75$) is observed between in-degree and authority centralities (0.932), PageRank and authority centralities (0.879), hubs and out-degree centralities (0.787). Other positive correlations (with lower coefficients) were determined by the test, but visually they aren't noticeable.