Prep6

Sebastian Montesinos

Due by midnight, Monday April 4

Reminder: Prep assignments are to be completed individually. Upload a final copy of the .Rmd and renamed .pdf to your private repo, and submit the renamed pdf to Gradescope.

Reading

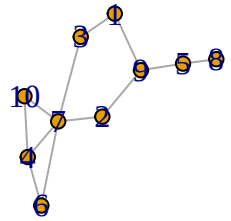
The associated reading for the week is Chapter 20 on Networks.

Note: There is no practice set this week (no Practice6), as you will be finishing the Shiny project for Thursday's class. The next practice, Practice7, will include one network related question, so you'll see that the following week.

1 - Basic Graph Concepts with a Toy Graph

Use the following generated graph to answer the questions that follow. Do NOT change the seed.

```
set.seed(231)
g <- erdos.renyi.game(n = 10, p = 0.3)
plot(g)
```



part a - How many vertices does the graph, g, have? How many edges?

Solution: The graph has 10 vertices and 12 edges.

part b - Compute the diameter of the graph. Then identify a path with that length.

Hint: More than one path may have a length equal to the diameter. Remember that diameter is not the longest path possible. It is the longest of all the shortest paths. To identify a path, list the vertices involved such as 1-2-3 (this is not an actual path in this graph).

Solution: One path with this diameter would be from 8-5-9-2-7-10.

```
diameter(g, weights = NA)
```

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

part c - Compute the degree for vertex 7 without using an R command. Explain what this number represents.

Solution: The degree of vertex 7 is 5. This number represents the number of edges connected to a vertex.

part d - Looking at the plot of the graph, identify a vertex triple - three vertices which are connected by edges. Is your selected vertex triple closed - i.e. is it a triangle?

This is asking you to find a set of vertices that could form a triangle - at least 2 of the 3 potential edges should be there. Determining the fraction of actual triangles out of possible triangles is a measure of what is called the clustering present in the network.

You can list your triple with node numbers. E.G. 1-2-8 (not an actual triple in this graph). The idea would be that 1-2 is an edge and 2-8 is an edge. If 1-8 is also an edge, this would be a triangle.

Solution: 10-7-4 is a vertex triple as all three vertices are connected by edges. This triple is also closed since 10-7, 7-4, and 10-4 are all edges, making it a triangle.

> part e - Imagine you have to walk along this graph. Walking around, what vertices are you most likely to visit often if you move around randomly? (What vertices help connect others a lot?)

No computations necessary here - you don't need to solve this formally - just think through it and give a guess. If you really want, you could look up a formal definition for a random walk on a graph. Basically, at any vertex, you have an equal probability of going to any vertices it has an edge to. For example, in this graph, if you were at vertex 3, you'd have a 50% chance of going to vertex 1 and 50% chance of going to vertex 7.

Solution: I would estimate that vertex 7 would be the most likely one for you to visit, followed by vertex 9 and 4. I'd predict this because these vertices are connected to many other vertices and so there's a high probability you'd travel to them on any given random selection of a new path.

> part f - Compute betweenness centrality for all vertices. Identify the top 3 vertices in terms of betweeness. How do these vertices relate to those you listed in part e?

Solution: The top three vertices in terms of betweeness are 7, then 9, then 2. I was right that two of these three vertices, 7 and 9, would be in the top 3.

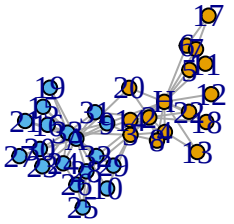
```
centralization.betweenness(g)
```

```
## $res
##  [1]  3.0 12.0  4.0  0.5  8.0  0.0 19.5  0.0 15.0  0.0
##
## $centralization
## [1] 0.4104938
##
## $theoretical_max
## [1] 324
```

2 - Exploring a Social Network

For this question, we'll explore a famous social network example known as "Zachary's Karate Club". This is information about a Karate club from 1970-1972 that split into 2 factions/groups based on a rift between members denoted "A" for "John A" and "H" for "Mr Hi" in the data set (these are not real names). Let's investigate a little bit!

```
# Visualize the network
# Igraph default plot, may not be "pretty"
data(karate)
plot(karate)
```



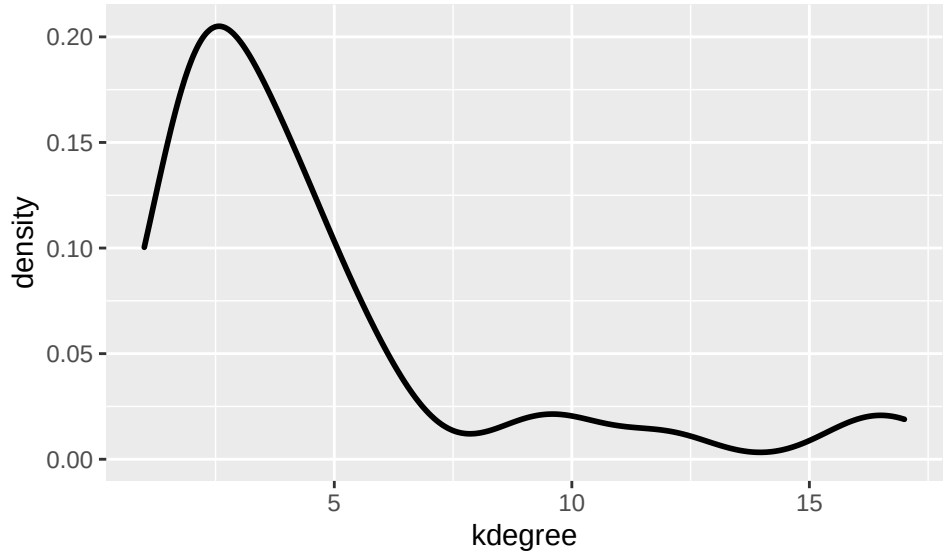
part a - Plot and describe the degree distribution of the karate network.

Hint: ggplot2 requires data to be in a data.frame to plot, so the code below will get you the degree distribution in data set to use, along with faction information for later use.

```
faction <- get.vertex.attribute(karate)$Faction
kdegree <- degree(karate)
karatedata <- data.frame(kdegree, faction)
```

Solution:

```
p <- ggplot(data = karatedata, aes(x = kdegree,
                                   color = faction)) +
  geom_density(size = 1)
p
```



part b - Identify the individuals with the top 5 highest degree values. Do they include "John A" and "Mr Hi"?

Solution: The individuals with the highest degree values are John A, Mr Hi, Actor 33, Actor 3, and Actor 2. They do include the two names individuals.

```
karatedata %>%
  arrange(desc(kdegree)) %>%
  head(5)
```

```
##          kdegree faction
## John A        17       2
## Mr Hi         16       1
## Actor 33      12       2
## Actor 3       10       1
## Actor 2        9       1
```

part c - Determine the eigenvector centrality of all vertices. Identify the individuals with the top 5 eigenvector centrality values. Do they include "John A" and "Mr Hi"?

Hint: The igraph command is eigen_centrality.

Solution:

```
karatedata <- karatedata %>%
  mutate(e_centrality = eigen_centrality(karate)$vector) %>%
  arrange(desc(e_centrality))

karatedata %>%
  head(5)
```

```
##          kdegree faction e_centrality
## John A        17       2    1.0000000
```

```
## Actor 3       10       1    0.9903645
## Actor 33      12       2    0.9125632
## Mr Hi         16       1    0.8578794
## Actor 2        9       1    0.8287662
```

part d - We investigated k-means as a method of clustering observations to find natural groups. Clustering can be performed on networks, though very different methods are needed to do so. Run the code below to perform a clustering analysis on this network. Then use the provided output to assess whether the clusters found match the provided factions assigned in the network by the researcher who studied the karate club. What did you find? Describe the findings in a few sentences.

Note: This clustering analysis may not find just 2 clusters.

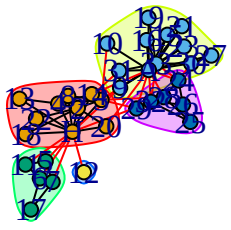
```
# run clustering on karate network
# this is just one example clustering algorithm for a network
kclusters <- leading.eigenvector.community(karate)
# number of clusters
length(kclusters)
```

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

```
# size of each cluster
sizes(kclusters)
```

```
## Community sizes
##  1  2  3  4  5
## 10 12  5  1  6
```

```
#how to plot the solution
plot(kclusters, karate)
```



```r
# Get cluster memberships to compare to faction
karatedata <- karatedata %>%
  mutate(clusters = as.numeric(membership(kclusters)))
mosaic::tally(clusters ~ faction, data = karatedata)
```

```
##         faction
## clusters 1 2
##        1 4 6
##        2 6 6
##        3 3 2
##        4 0 1
##        5 3 3
```

Solution: The KMeans analysis grouped the data in a similar way to original researcher, but used many more clusters. In the KMeans analysis the individuals from faction 2 largely went to clusters 2 and 5 and the individuals from faction 1 largely went to clusters 1 and 3. Therefore, though the KMeans analysis still preserved the original grouping generally, it complicated it by futher dividing each faction into a number of new clusters.