CPTS 591 Elements of Network Science

Jinyang Ruan

011696096

3/15/2021

Assignment 2

Problem 1

The fourth graph I chose is called "Zachary's karate club". It is asocial network of friendships between 34 members of a karate club at a US university in the 1970s.

URL: http://www-personal.umich.edu/~mejn/netdata/

Load 4 real world graphs:

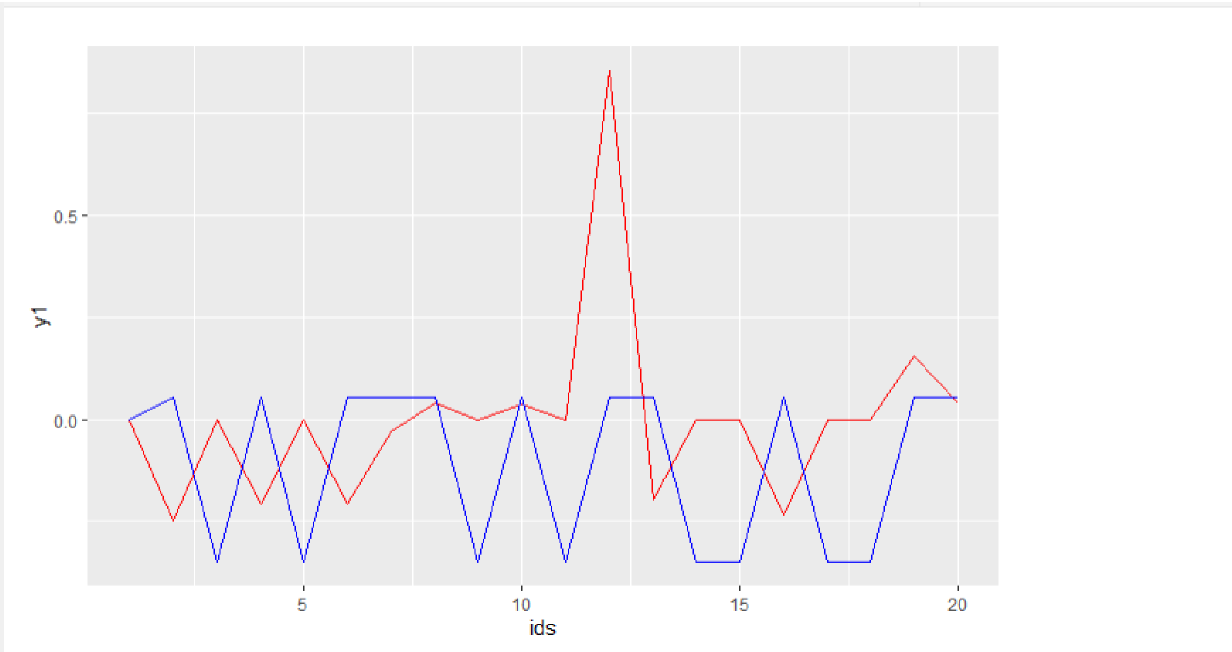
```
```{r load 4 graphs}
political_graph <- read_graph('Political_blogs.gml',format = 'gml')
neural_graph <- read_graph('Neural_network.gml', format = 'gml')
internet_graph <- read_graph('Internet.gml',format = 'gml')
karate_graph <- read_graph('karate.gml',format = 'gml')
```
```

1. Political Blogs Graph:

Highest score nodes in (i) Degree:

```
```{r Political graph-degree}
which(degree(political_graph) == max(degree(political_graph)))
```

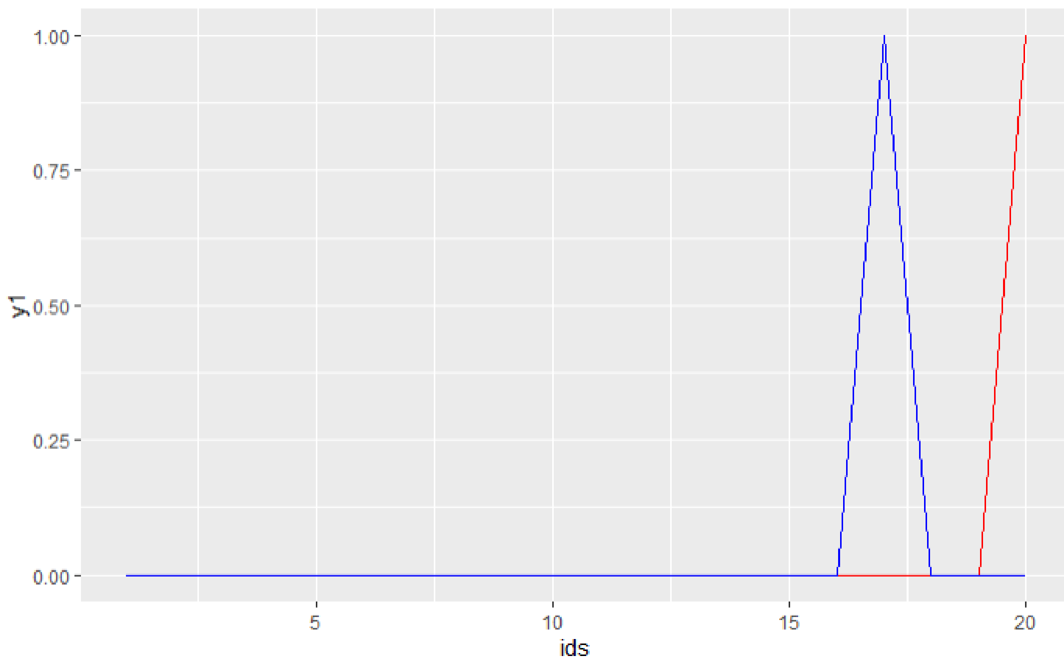
[1] 855
```

(ii) Eccentricity:

```
```{r Political graph-eccentricity}
which(eccentricity(political_graph) == max(eccentricity(political_graph)))
```

[1] 794 1259
```

(iii) Closeness:

```
```{r Political graph-closeness}
which(closeness(political_graph) == max(closeness(political_graph)))
```

At centrality.c:2784 :closeness centrality is not well-defined for disconnected graphsAt centrality.c:2784 :closeness centrality is not well-defined for disconnected graphs[1] 293
```

(iv) Betweenness:

```
```{r Political graph-betweenness}
which(betweenness(political_graph) == max(betweenness(political_graph)))
```

[1] 855
```

(v) Katz index:

```
```{r Political graph-Katz index}
which(alpha.centrality(political_graph, alpha = 0.1) == max(alpha.centrality(political_graph, alpha = 0.1)))
```

[1] 641
```

(vi) PageRank:

```
```{r Political graph-PageRank}
which(page.rank(political_graph)$vector == max(page.rank(political_graph)$vector))
```
```

```
[1] 155
```

(vii) Kleinberg's Authority Score:

```
```{r Political graph-Kleinberg's Authority Score}
which(authority.score(political_graph)$vector == max(authority.score(political_graph)$vector))
```
```

```
[1] 155
```

(viii) Kleinberg's Hub Score:

```
```{r Political graph-Kleinberg's Hub Score}
which(hub.score(political_graph)$vector == max(hub.score(political_graph)$vector))
```
```

```
[1] 512
```

2. Neural Network Graph:

Highest score nodes in (i) Degree:

```
```{r neural graph-degree}
which(degree(neural_graph) == max(degree(neural_graph)))
```
```

```
[1] 45
```

(ii) Eccentricity:

```
```{r neural graph-eccentricity}
which(eccentricity(neural_graph) == max(eccentricity(neural_graph)))
```
```

```
[1]  82 127 129 243 244 296 297
```

(iii) Closeness:

```
```{r neural graph-closeness}
which(closeness(neural_graph) == max(closeness(neural_graph)))
```
```

```
At centrality.c:2784 :closeness centrality is not well-defined for disconnected graphsAt centrality.c:2784 :closeness centrality is not well-defined for disconnected graphs[1] 260
```

(iv) Betweenness:

```
```{r neural graph-betweenness}
which(betweenness(neural_graph) == max(betweenness(neural_graph)))
```
```

```
[1] 178
```

(v) Katz index:

```
```{r neural graph-Katz index}
which(alpha.centrality(neural_graph, alpha = 0.1) == max(alpha.centrality(neural_graph, alpha = 0.1)))
```
```

```
[1] 45
```

(vi) PageRank:

```
```{r neural graph-PageRank}
which(page.rank(neural_graph)$vector == max(page.rank(neural_graph)$vector))
```
```

```
[1] 45
```

(vii) Kleinberg's Authority Score:

```
```{r neural graph-Kleinberg's Authority Score}
which(authority.score(neural_graph)$vector == max(authority.score(neural_graph)$vector))
```
```

```
[1] 45
```

(viii) Kleinberg's Hub Score:

```
```{r neural graph-Kleinberg's Hub Score}
which(hub.score(neural_graph)$vector == max(hub.score(neural_graph)$vector))
```
```

```
[1] 126
```

3. Internet Graph:

Highest score nodes in (i) Degree:

```
```{r Internet graph-degree}
which(degree(internet_graph) == max(degree(internet_graph)))
```
```

```
[1] 4
```

(ii) Eccentricity:

```
```{r Internet graph-eccentricity}
which(eccentricity(internet_graph) == max(eccentricity(internet_graph)))
```
```

```
[1]  9200 16852
```

(iii) Closeness:

```
```{r Internet graph-closeness}
which(closeness(internet_graph) == max(closeness(internet_graph)))
```
```

```
[1] 23
```

(iv) Betweenness:

```
```{r Internet graph-betweenness}
which(betweenness(internet_graph) == max(betweenness(internet_graph)))
```
```

```
[1] 4
```

(v) Katz index:

```
```{r Internet graph-Katz index}
which(alpha.centrality(internet_graph, alpha = 0.1) == max(alpha.centrality(internet_graph, alpha = 0.1)))
```
```

```
[1] 1758
```

(vi) PageRank:

```
```{r Internet graph-PageRank}
which(page.rank(internet_graph)$vector == max(page.rank(internet_graph)$vector))
```
```

```
[1] 4
```

(vii) Kleinberg's Authority Score:

```
```{r Internet graph-Kleinberg's Authority Score}
which(authority.score(internet_graph)$vector == max(authority.score(internet_graph)$vector))
```
```

```
[1] 4
```

(viii) Kleinberg's Hub Score:

```
```{r Internet graph-Kleinberg's Hub Score}
which(hub.score(internet_graph)$vector == max(hub.score(internet_graph)$vector))
```
```

```
[1] 4
```

4. Zachary's Karate Club Graph:

Highest score nodes in (i) Degree:

```
```{r karate graph-degree}
which(degree(karate_graph) == max(degree(karate_graph)))
```
```

```
[1] 34
```

(ii) Eccentricity:

```
```{r karate graph-eccentricity}
which(eccentricity(karate_graph) == max(eccentricity(karate_graph)))
```
```

```
[1] 15 16 17 19 21 23 24 27 30
```

(iii) Closeness:

```
```{r karate graph-closeness}
which(closeness(karate_graph) == max(closeness(karate_graph)))
```
```

```
[1] 1
```

(iv) Betweenness:

```
```{r karate graph-betweenness}
which(betweenness(karate_graph) == max(betweenness(karate_graph)))
```
```

```
[1] 1
```

(v) Katz index:

```
```{r karate graph-Katz index}
which(alpha.centrality(karate_graph, alpha = 0.1) == max(alpha.centrality(karate_graph, alpha = 0.1)))
```
```

```
[1] 34
```

(vi) PageRank:

```
```{r karate graph-PageRank}
which(page.rank(karate_graph)$vector == max(page.rank(karate_graph)$vector))
```
```

```
[1] 34
```

(vii) Kleinberg's Authority Score:

```
```{r karate graph-Kleinberg's Authority Score}
which(authority.score(karate_graph)$vector == max(authority.score(karate_graph)$vector))
```
```

```
[1] 34
```

(viii) Kleinberg's Hub Score:

```
```{r karate graph-Kleinberg's Hub Score}
which(hub.score(karate_graph)$vector == max(hub.score(karate_graph)$vector))
```
```

```
[1] 34
```

Result analysis:

From the above 4 graphs, we can find there are some cases in which the different centrality measures sometimes identify the same node(s) with the highest score in one graph, in other words, different centrality measures could identify that node(s) as the most important. For example, in neural network graph, the node with the index 45 has the highest score in degree, Katz index, PageRank, and Kleinberg's authority score. In the Internet graph, the node with the index 4 has the highest score in degree, betweenness, PageRank, Kleinberg's authority score, and Kleinberg's hub score. In those cases, we can consider those nodes are the most important. However, it will not happen all the time, for example, in political blogs graph, there is not any node(s) which has most highest scores in all centrality measures, so we cannot say there is any node(s) is the most important in this graph.

Problem 2

Generate 2 random graphs with 20 nodes:

````
```{r generate 2 random graphs with node 20}
erg1 <- erdos.renyi.game(20, .1, type = "gnp")
E(erg1)
bag1 <- barabasi.game(20, power = 1.0)
E(bag1)
```
````

```
+ 19/19 edges from 8299add:
 [1]  6-- 8  3-- 9  3--11  2--12  4--12  6--12  2--13 10--13 12--13  3--14 12--16  3--17  5--17 15--17 17--18  2--19  7--19 16--19
[19]  4--20
+ 19/19 edges from 82a87ae:
 [1]  2-> 1  3-> 1  4-> 3  5-> 1  6-> 1  7-> 1  8-> 1  9-> 1 10-> 5 11-> 1 12-> 1 13->10 14-> 1 15->10 16->10 17-> 1 18->16 19-> 3
[19] 20-> 4
```

Generate 2 random graphs with 40 nodes:

````
```{r generate 2 random graphs with node 40}
erg2 <- erdos.renyi.game(40, .05, type = "gnp")
E(erg2)
bag2 <- barabasi.game(40, power = 1.0)
E(bag2)
```
````

```
+ 39/39 edges from 9b2646e:
 [1]  2-- 5  2-- 8  4--10  5--10  1--11  5--13  3--14  1--16 13--17 11--18 12--19  4--21  3--22  8--22 17--24 19--24 12--26 10--27
[19] 26--27  4--28 12--28  8--29 13--29 27--30 11--31 16--31 18--31  9--33 25--33  2--34 22--34 16--36 25--36 35--36  3--37 31--37
[37] 30--38 35--39 12--40
+ 39/39 edges from 9b3674e:
 [1]  2-> 1  3-> 2  4-> 1  5-> 4  6-> 1  7-> 4  8-> 3  9-> 4 10-> 1 11-> 1 12-> 3 13-> 1 14-> 4 15-> 1 16-> 4 17-> 2 18-> 1 19->12
[19] 20->10 21->20 22->14 23->22 24-> 1 25->22 26->14 27-> 7 28->14 29->27 30->16 31-> 4 32->31 33->24 34->26 35->17 36->31 37-> 3
[37] 38->19 39->26 40->19
```

I also presented all edges to check whether they have roughly the same number of edges.

Get the largest connected component for each graph:

````
```{r get the largest connected components for each graph}
is.connected(erg1)
is.connected(bag1)
is.connected(erg2)
is.connected(bag2)

decompose.graph(erg1)
derg1 <- decompose.graph(erg1)
V(derg1[[1]])
V(derg1[[2]])
V(derg1[[3]])

lcderg1 <- derg1[[2]]

decompose.graph(erg2)
derg2 <- decompose.graph(erg2)
V(derg2[[1]])
V(derg2[[2]])
V(derg2[[3]])
V(derg2[[4]])
V(derg2[[5]])
V(derg2[[6]])
V(derg2[[7]])

lcderg2 <- derg2[[1]]
```
````

1. Compute all eigenvalues corresponding eigenvectors of the Laplacian of the graph:

```
```{r 2-1}
eigen1 <- sort(eigen(laplacian_matrix(lcderg1))$values)
eigen2 <- sort(eigen(laplacian_matrix(bag1))$values)
eigen3 <- sort(eigen(laplacian_matrix(lcderg2))$values)
eigen4 <- sort(eigen(laplacian_matrix(bag2))$values)
```
```

2. Compute and plot the table (For easy observation, all results are reserved with 5 valid numbers):

```
```{r 2-2}
options(digits = 5)
data.frame(Graph = c("LCC ER random graph with node 20","LCC BA random graph with node 20", "LCC ER random graph with node 40", "LCC BA
random graph with node 40"),
           n = c(vcount(lcderg1),vcount(bag1),vcount(lcderg2),vcount(bag2)),
           m = c(ecount(lcderg1),ecount(bag1),ecount(lcderg2),ecount(bag2)),
           dmin = c(min(degree(lcderg1)),min(degree(bag1)),min(degree(lcderg2)),min(degree(bag2))),
           dmax = c(max(degree(lcderg1)),max(degree(bag1)),max(degree(lcderg2)),max(degree(bag2))),
           l = c(average.path.length(lcderg1),average.path.length(bag1),average.path.length(lcderg2),average.path.length(bag2)),
           D = c(diameter(lcderg1),diameter(bag1),diameter(lcderg2),diameter(bag2)),
           ccg = c(transitivity(lcderg1, type = "globalundirected"),transitivity(bag1, type = "globalundirected"),transitivity(lcderg2,type =
"globalundirected"),transitivity(bag2,type = "globalundirected")),
           Second_smallest_eigenvalue = c(eigen1[2],eigen2[2],eigen3[2],eigen4[2]),
           Largest_eigenvalue = c(max(eigen1)),max(eigen2),max(eigen3),max(eigen4))
```
```

Graph <chr>	n <int>	m <dbl>	dmin <dbl>	dm... <dbl>	l <dbl>	D <dbl>	ccg <dbl>	Second_smallest_eigenvalue <dbl>	Largest_eigenvalue <dbl>
LCC ER random graph with node 20	11	12	1	5	2.4727	5	0.136364	0.360467	6.2842
LCC BA random graph with node 20	20	19	1	11	1.6364	4	0.000000	1.000000	6.2842
LCC ER random graph with node 40	34	39	1	4	6.1836	16	0.048387	0.024947	6.2842
LCC BA random graph with node 40	40	39	1	9	2.0303	5	0.000000	1.000000	6.2842

4 rows | 1-10 of 13 columns

3. Plot graph for first ER model. Red line represents the eigenvectors corresponding to the largest eigenvalue, blue line represents the eigenvectors corresponding to the second smallest eigenvalue.

```
```{r 2-3(1)}
ids <- array(1:20)
y1 <- eigen(laplacian_matrix(erg1))$vector[,1]
y2 <- eigen(laplacian_matrix(erg1))$vector[,19]
df <- data.frame(ids, y1,y2)
ggplot(df,aes(ids)) +
  geom_line(aes(y=y1),color = "red") +
  geom_line(aes(y=y2),color ="blue")
```
```



Plot graph for BA model. Red line represents the eigenvectors corresponding to the largest eigenvalue, blue line represents the eigenvectors corresponding to the second smallest eigenvalue.

````
```{r 2-3(2)}
ids <- array(1:20)
y1 <- eigen(laplacian_matrix(bag1))$vector[,1]
y2 <- eigen(laplacian_matrix(bag1))$vector[,19]
df <- data.frame(ids, y1,y2)
ggplot(df,aes(ids)) +
  geom_line(aes(y=y1),color = "red") +
  geom_line(aes(y=y2),color ="blue")
```
````



Observation:

In the ER model, the eigenvectors are totally different when we choose the largest eigenvalue and the second smallest eigenvalue. However, in the BA model, most eigenvectors are 0, few eigenvectors are 1, when we correspond to different eigenvalues, the eigenvectors are almost the same.

Problem 3

1. PageRank in biology & bioinformatics: Gene Rank, Protein Rank, Iso Rank. I personally very interested in biology. I think it is a mysterious field. I did not expect that PageRank could be applied to this field. All advanced theories or algorithms might help human-beings achieve the truth of our body closer.

2. PageRank in social networks: Buddy Rank, Twitter Rank. Take a simple example, such as the recommendation system on Facebook, which provides us with a lot of data to recommend relevant people. It can find many interesting connections between things through data.

3. PageRank in literature: Book Rank. Online libraries become more and more popular, so Book Rank becomes necessary in our life. It is very useful when we are learning new knowledge or finding references. I can easily find related articles and books.