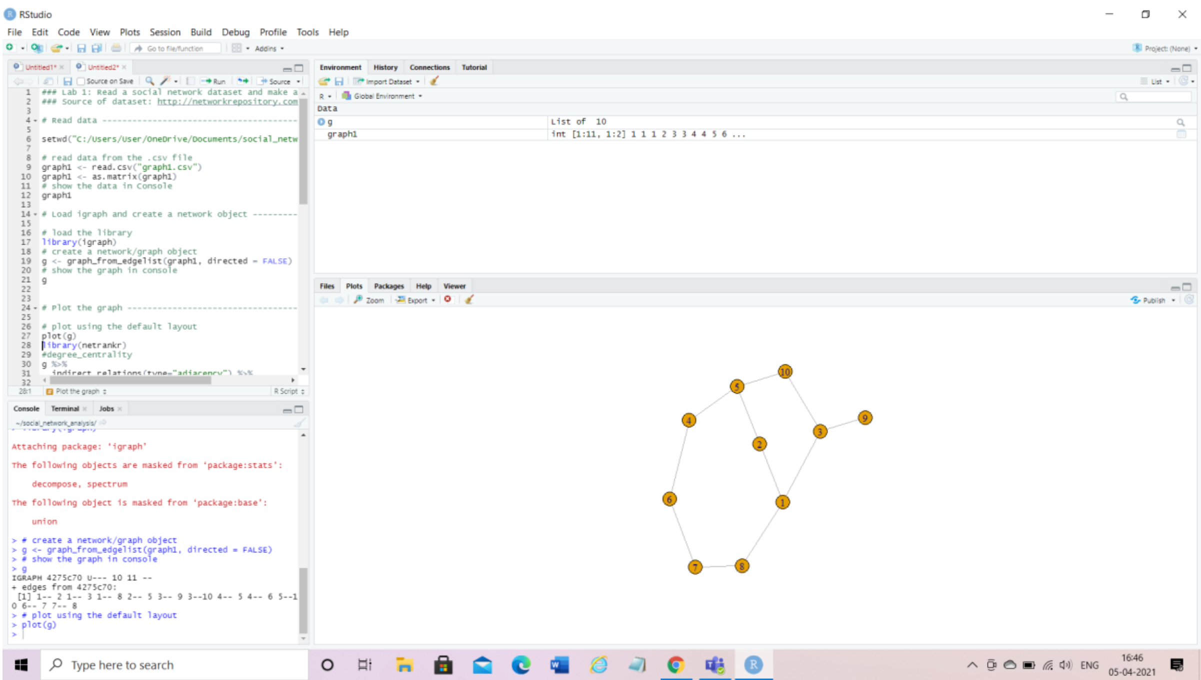
Social Network Analysis

Deepthi V

Two sample graphs are taken and centralities for respective graphs are calculated in RStudio.

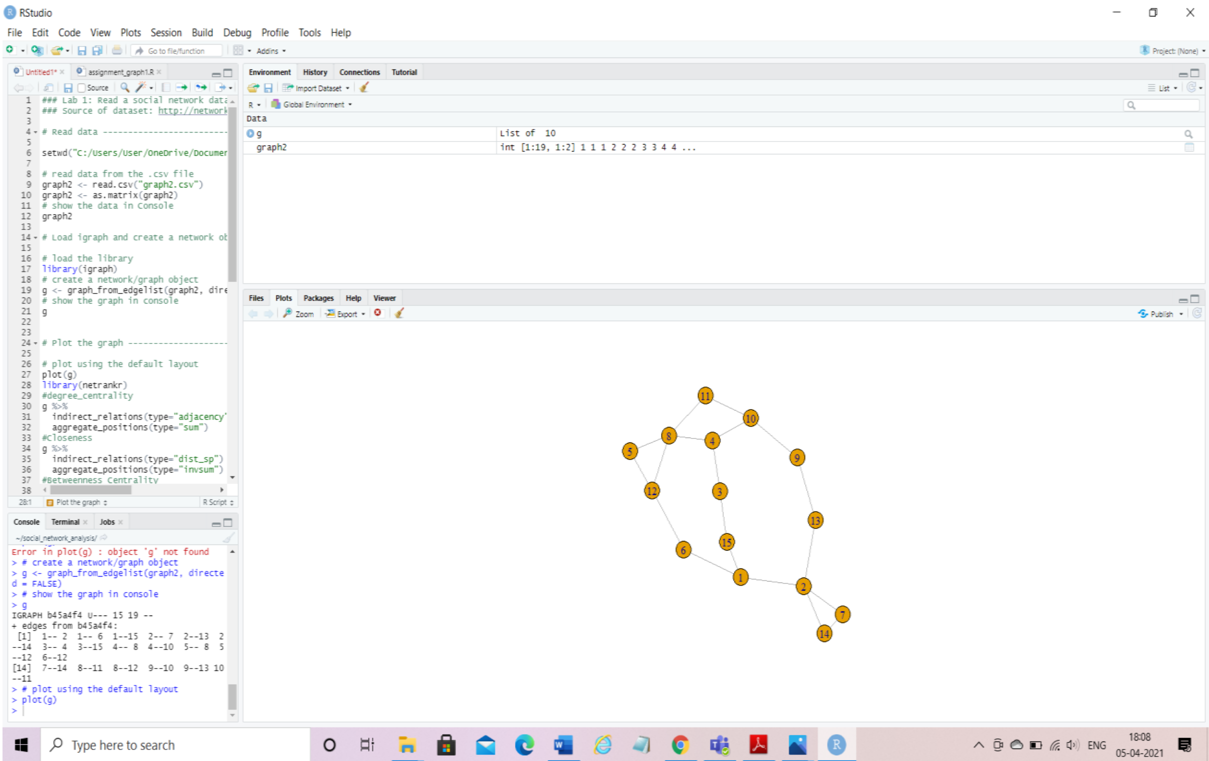
Graph1:

Nodes: 10



Graph 2:

Nodes : 15



Degree Centrality:

Definition:

Degree centrality is defined as the number of links connected to a node.

Formula:

$$C_D(v_k) = \sum_{i=1}^{p} a(v_i, v_k)$$

$a(v_i, v_k) = 1$ only if v_i and v_k are adjacent.

Function used in R:

```
g %>%
  indirect_relations(type="adjacency") %>%
```

```
  aggregate_positions(type="sum")
```

Graph 1 output:

[1] 3 2 3 2 3 2 2 2 1 2

Graph 2 output:

[1] 3 4 2 3 2 2 2 4 2 3 2 3 2 2 2

Closeness centrality:

Definition:

Closeness centrality is a measure of the average shortest distance from each vertex to each other vertex.

Formula:

$$C_c(v_k)^{-1} = \sum_{i=1}^{p} d(v_i, v_k)$$

$d(v_i, v_k)$ = number of edges in geodesic joining v_i and v_k.

Function used in R:

```
g %>%
  indirect_relations(type="dist_sp") %>%
  aggregate_positions(type="invsum")
```

Graph 1 output:

[1] 0.05882353 0.05263158 0.05263158 0.04761905 0.05555556

[6] 0.04166667 0.04347826 0.05000000 0.03703704 0.05000000

Graph 2 output:

1] 0.03030303 0.02857143 0.02631579 0.02777778 0.02272727 0.02777778 0.02127660 0.02702703 0.02564103 0.02702703 0.02439024 0.02702703 0.02564103

[14] 0.02127660 0.02631579

Betweeness centrality:

Definition:

Betweenness centrality measures the extent to which a vertex lies on paths between other vertices.

Formula:

$$b_{ij}(v_k) = \frac{g_{ij}(v_k)}{g_{ij}}$$

$g_{ij}(v_k)$ = number of geodesics containing v_k

Function used in R:

```
g %>%
  indirect_relations(type="depend_sp") %>%
  aggregate_positions(type="sum")
```

Graph 1 output:

[1] 23 6 21 11 19 7 8 13 0 10

Graph 2 output:

[1] 56.666667 63.666667 16.000000 30.666667 0.000000 31.666667 0.000000 33.333333 26.333333 33.333333 8.666667 29.666667 27.000000 0.000000 17.000000

Eigenvector centrality:

Definition:

Eigen vector centrality is a measure of influence of node in a etwork.

Relative scores are assigned to all nodes in the network based on the concept that connections to high-scoring nodes contribute more to the score of the node in question than equal connections to low-scoring nodes.

Formula:

$$x_i(t) = \sum_j A_{ij} x_j(t-1)$$

Function used in R:

```
g %>%
  indirect_relations(type="walks",FUN=walks_limit_prop) %>%
  aggregate_positions(type="sum")
```

Graph 1 output:

[1] 1.3180454 1.0774833 1.2188104 0.7605923 1.2199086 0.5716255

[7] 0.5858409 0.8082912 0.5174435 1.0353533

Graph 2 output:

[1] 0.7586139 0.8002644 0.6397064 1.2321429 1.1010726 0.7562014 0.4632440 1.6992577 0.5570513 1.0217330 0.9976053 1.3039423 0.4976369 0.4632440 0.5126705

Subgraph centrality:

Definition:

Subgraph centrality of a node is a weighted sum of the numbers of all closed walks of different lengths in the network starting and ending at the node.

Formula:

$$SC(v) = \frac{\sum_{k=0}^{\infty} \mu_k(v)}{k!}$$

$\mu_k(v)$ = number of closed walks of length k starting and ending node v in the network.

Function used in R:

```
g %>%
  indirect_relations(type="walks",FUN=walks_exp) %>%
  aggregate_positions(type="self")
```

Graph 1 output:

[1] 3.156213 2.410969 3.103330 2.334679 3.102135 2.281797

[7] 2.281825 2.336337 1.643542 2.408940

Graph 2 output:

[1] 3.220750 4.569829 2.345196 3.398581 2.951876 2.417561 2.868841 4.784767 2.342932 3.262333 2.589298 3.724151 2.416378 2.868841 2.340059

Communicability centrality:

Definition:

Communicability measure makes use of the number of walks connecting every pair of nodes as the basis of a betweenness centrality measure.

Formula:

$$\omega_r = \frac{1}{C}\sum_p \sum_q \frac{G_{prq}}{G_{pq}} \qquad p \neq q,\ p \neq r,\ q \neq r$$

where G_{prq} = number of walks involving node r

G_{pq} = number of closed walks starting at node p ending at node q.

Function used in R:

```
g %>%
  indirect_relations(type="walks",FUN=walks_exp) %>%
  aggregate_positions(type="sum")
```

Graph 1 output:

```
[1] 13.248986 10.587607 12.280389  8.976874 12.481731  7.879132
[7]  7.921923  9.202113  5.637963 10.323290
```

Graph 2 output:

```
[1] 15.63159 18.94302 10.69447 17.36758 13.86650 12.15922 11.68593 21.86233 10.42727
15.31206 13.44402 17.11545 11.35850 11.68593 10.35621
```

Odd subgraph centrality:

Definition:

Subgraph centrality of a node is a weighted sum of the numbers of all closed walks of odd lengths in the network starting and ending at the node.

Formula:

$$C_{S_{odd}} = \frac{\mu_1(i)}{1!} + \frac{\mu_3(i)}{3!} + \frac{\mu_5(i)}{5!} +$$

Function used in R:

```
g %>%
  indirect_relations(type="walks",FUN=walks_exp_odd) %>%
  aggregate_positions(type="self")
```

Graph 1 output:

[1] 0.0220044774 0.0215238107 0.0215364338 0.0009361984

[5] 0.0219980543 0.0004681549 0.0004681552 0.0009362550

[9] 0.0004619552 0.0210681676

Graph 2 output:

[1] 0.0235264131 0.5126862152 0.0010017375 0.0245751064 0.4921585253 0.0225773642 0.4690664011 0.5383026088 0.0005141814 0.0020363583 0.0230981543

[12] 0.5142482791 0.0215759745 0.4690664011 0.0009883623

Even subgraph centrality:

Definition:

Subgraph centrality of a node is a weighted sum of the numbers of all closed walks of even lengths in the network starting and ending at the node.

Formula:

$$C_{s\,even} = \frac{\mu_0(i)}{0!} + \frac{\mu_2(i)}{2!} + \frac{\mu_4(i)}{4!} + \dots$$

Function used in R:

```
g %>%
  indirect_relations(type="walks",FUN=walks_exp_even) %>%
  aggregate_positions(type="self")
```

Graph 1 output:

[1] 3.134208 2.389445 3.081794 2.333743 3.080136 2.281329

[7] 2.281357 2.335400 1.643080 2.387872

Graph 2 output:

[1] 3.197223 4.057142 2.344195 3.374006 2.459717 2.394984 2.399774 4.246464 2.342418 3.260296 2.566199 3.209902 2.394802 2.399774 2.339070

Information centrality:

Definition:

Information centrality calculates the set of all possible paths between two nodes weighted by an information-based value for each path that is derived from the inverse of its length.

Formula:

$$CI(s)^{-1} = nC^{I}_{ss} + \text{trace}(C^{I}) - \frac{2}{n}$$

C^{I}_{ss} = element of s^{th} row and s^{th} column in C^{I}.

Function used in R:

infocent <- sna::infocent(get.adjacency(g,sparse=F))

glimpse(infocent)

Graph 1 output:

num [1:10] 0.951 0.816 0.847 0.756 0.923 …

Graph 2 output:

num [1:15] 0.842 0.772 0.683 0.824 0.644 ...

Alpha centrality:

Definition:

It is an adaptation of eigenvector centrality with the addition that nodes are imbued with importance from external sources.

Formula:

$$x = (I - \alpha A^T)^{-1} e$$

e_j = external important given to nodeJ

α is a parameter

Function used in R:

alpha_centrality(g)

Graph 1 output:

[1] -3 -4 2 0 -2 1 0 -2 3 1

Graph 2 output:

[1] -1.500000e+00 -1.000000e+00 2.220446e-16 -5.000000e-01 -1.110223e-16 -1.000000e+00 2.500000e-01 -5.000000e-01 -1.000000e+00 -1.000000e+00

[11] -5.000000e-01 -5.000000e-01 -1.000000e+00 2.500000e-01 -5.000000e-01

Power centrality:

Definition:

The centrality of each vertex is therefore determined by the centrality of the vertices it is connected to.

Formula:

$$C_i = \sum A_{ij}(\alpha + \beta C_j)$$

α and β are parameters

Function used in R:

power_centrality(g)

Graph 1 output:

[1] -1.5569979 -1.9462474 0.3892495 -0.3892495 -1.1677484

[6] 0.0000000 -0.3892495 -1.1677484 0.7784989 0.0000000

Graph 2 output:

[1] -1.5191091 -1.2152872 -0.6076436 -0.9114654 -0.6076436 -1.2152872 -0.4557327 -0.9114654 -1.2152872 -1.2152872 -0.9114654 -0.9114654 -1.2152872

[14] -0.4557327 -0.9114654

Page Rank centrality:

Definition:

The PageRank algorithm measures the importance of each node within the graph, based on the number incoming relationships and the importance of the corresponding source nodes.

Formula:

$$PR(A) = (1 - d) + d\left(\frac{PR(T_1)}{C(T_1)} + \ldots + \frac{PR(T_n)}{C(T_n)}\right)$$

$d =$ damping factor.

Function used in R:

page_rank(g)

Graph 1 output:

$vector

[1] 0.13018909 0.08875235 0.13570211 0.09178815 0.13011332

[6] 0.09393579 0.09394313 0.09181274 0.05344893 0.09031437

$value

[1] 1

$options

NULL

Graph 2 output:

$vector

[1] 0.07832946 0.10287002 0.05526825 0.07624676 0.05222240 0.05365783 0.05540849 0.09768430 0.05557930 0.07764650 0.05275776 0.07575701 0.05548108

[14] 0.05540849 0.05568235

$value

[1] 1

$options

NULL

Katz centrality:

Definition:

Katz centrality computes the relative influence of a node within a network by measuring the number of the immediate neighbors and also all other nodes in the network that connect to the node under consideration through these immediate neighbors.

Formula:

$$x_i = \alpha \sum_j A_{ij} x_j + \beta$$

A = Adjacency matrix of graph G

$\alpha < \frac{1}{\lambda_{max}}$, is a parameter controls the initial centrality.

Function used in R:

katzcent(g)

Graph 1 output:

[1] 1.392243 1.277394 1.380654 1.263319 1.381695 1.251491

[7] 1.251587 1.264383 1.138065 1.276235

Graph 2 output:

[1] 1.407434 1.524775 1.268857 1.420937 1.295420 1.281939 1.280531 1.542237 1.267753 1.398274 1.294051 1.411960 1.279253 1.280531 1.267629

Bottleneck centrality:

Definition:

It measures the number of shortest paths going through a certain node. Therefore, nodes with the highest betweenness control most of the information flow in the network, representing the critical points of the network. We thus call these nodes the bottlenecks of the network.

Formula:

$$BN(v) = \sum_{s \in V} P_s(v)$$

Function used in R:

bottleneck(g)

Graph 1 output:

[1] 7 2 6 4 6 2 2 5 0 5

Graph 2 output:

[1] 11 12 3 9 0 7 0 9 9 8 4 5 7 0 3

Community centrality:

Definition:

Community centrality weights each community that a node belongs to by how similar that community is to each of the other communities to which the node also belongs. Formula:

$$C_c(i) = \sum_{i \in j}^{N} \left(1 - \frac{1}{m} \sum_{i \in j \cap k}^{m} S(j,k)\right)$$

Function used in R:

communitycent(g)

Graph 1 output:

[1] 1 1 1 1 1 1 1 1 1 1

Graph 2 output:

[1] 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0

Laplacian centrality:

Definition:

Laplacian centrality is a simple centrality measure that can be calculated in linear time. It is defined as the drop in the Laplacian energy (i.e. sum of squares of the eigenvalues in the Laplacian matrix) of the graph when the vertex is removed.

Formula:

$$C_v^L = (\Delta E)_v = d_G^2(v) + d_G(v) + 2\sum_{v_i \in N(v)} d_G(v_i)$$

Function used in R:

laplacian(g)

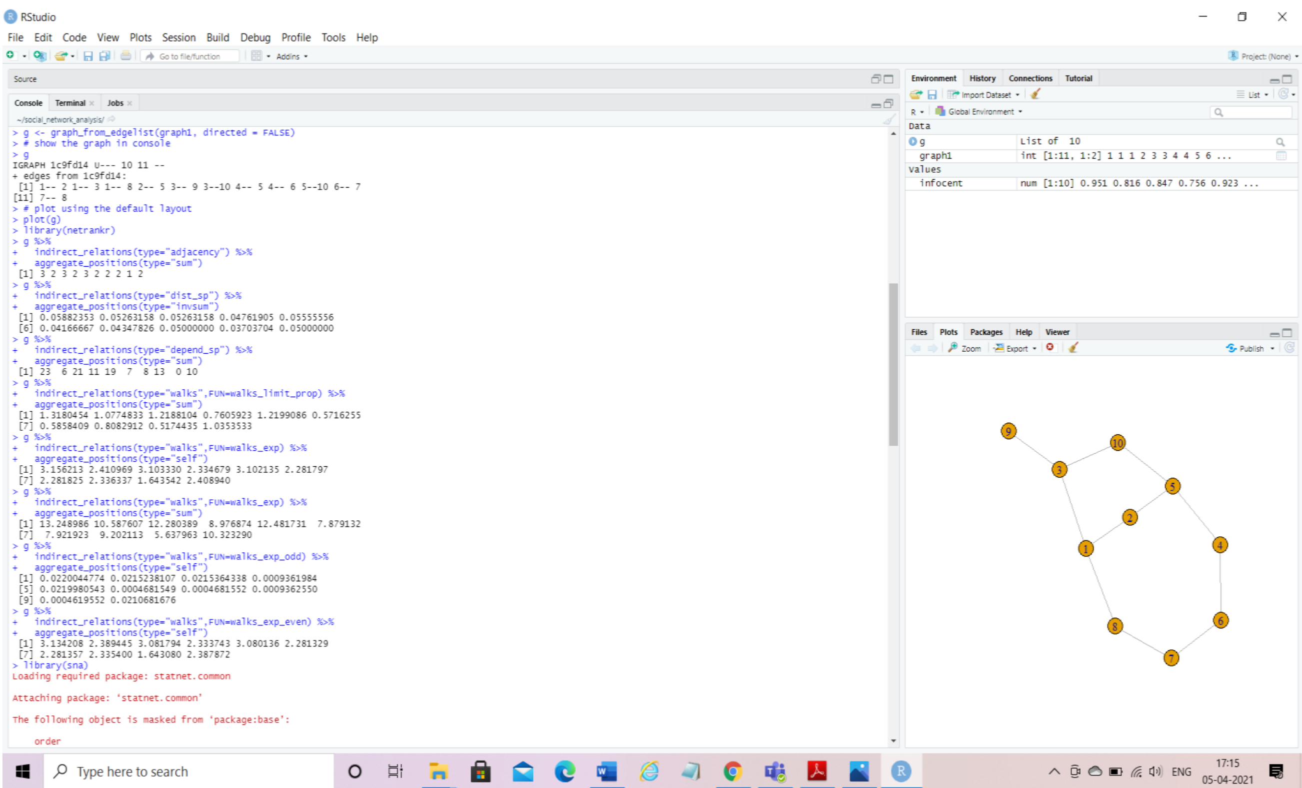
Graph 1 output:

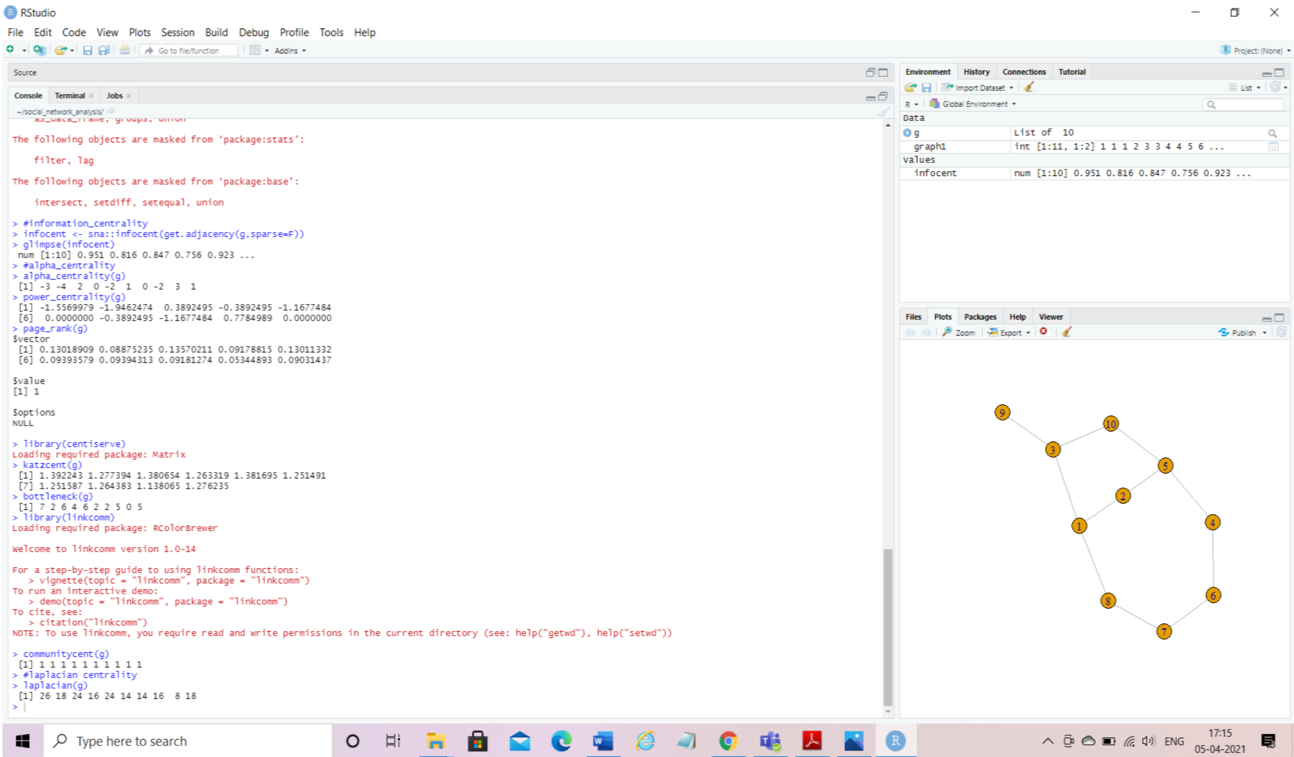
[1] 26 18 24 16 24 14 14 16 8 18

Graph 2 output:

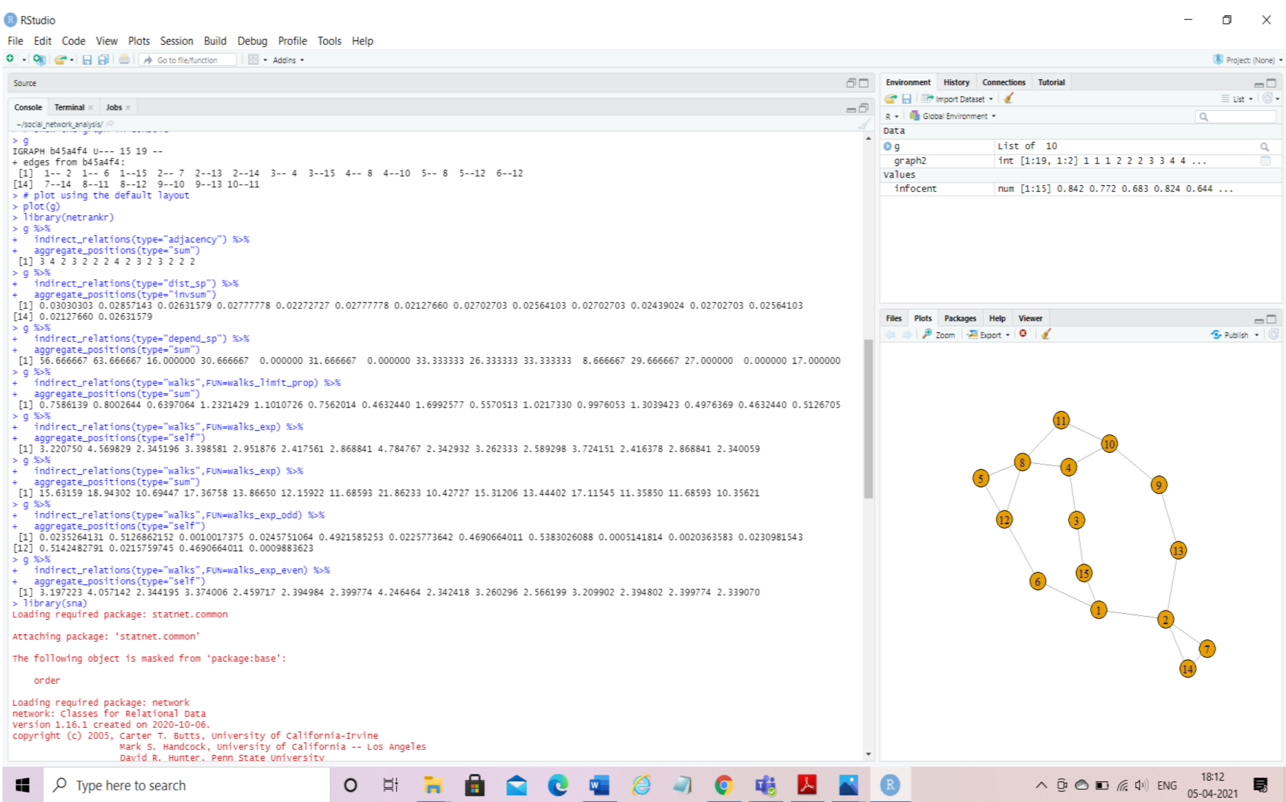
[1] 28 38 16 30 20 18 18 40 16 26 20 28 18 18 16

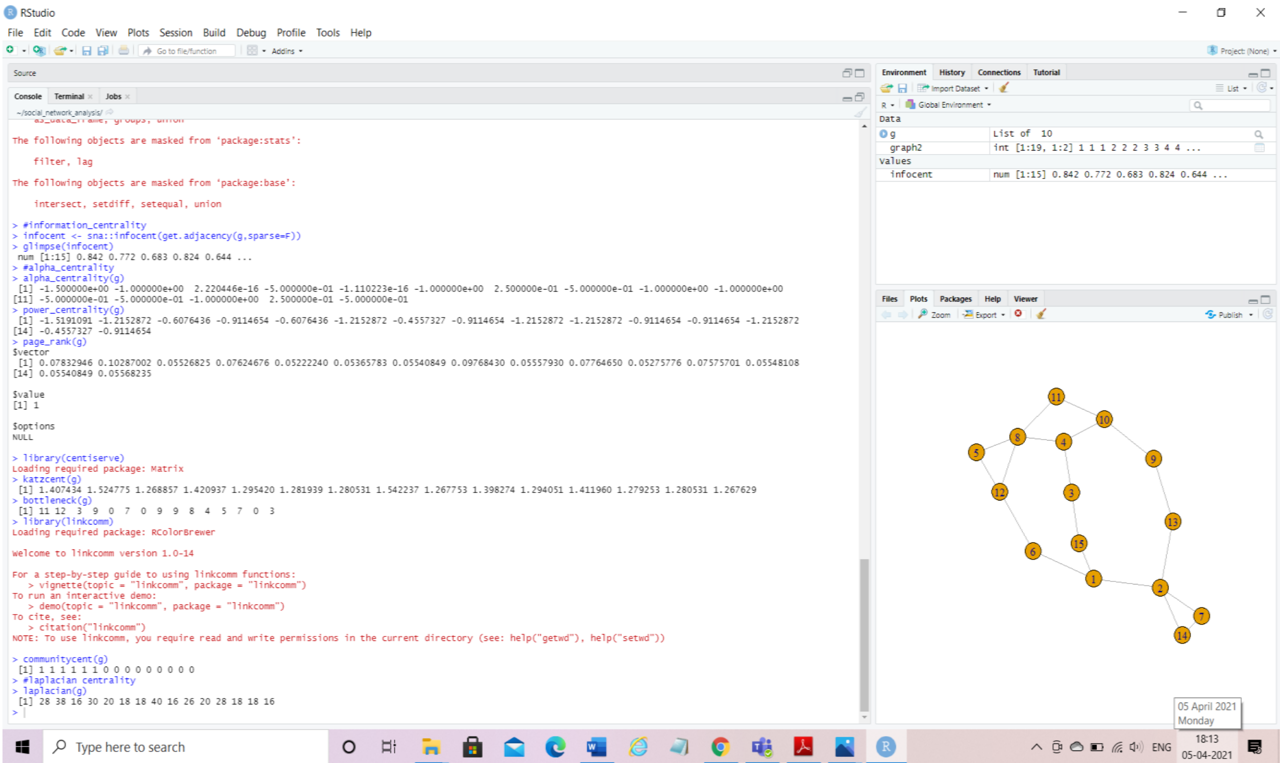
Output screen for graph1:





Output screen for graph2:




RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Console Terminal Jobs
~/social_network_analysis/
The following objects are masked from 'package:stats':
filter, lag
The following objects are masked from 'package:base':
intersect, setdiff, setequal, union
> #information_centrality
> infocent <- sna::infocent(get.adjacency(g,sparse=F))
> glimpse(infocent)
num [1:15] 0.842 0.772 0.683 0.824 0.644 ...
> #alpha_centrality
> alpha_centrality(g)
[1] -1.500000e+00 -1.000000e+00 2.220446e-16 -5.000000e-01 -1.110223e-16 -1.000000e+00 2.500000e-01 -5.000000e-01 -1.000000e+00 -1.000000e+00
[11] -5.000000e-01 -5.000000e-01 -1.000000e+00 2.500000e-01 -5.000000e-01
> power_centrality(g)
[1] -1.5191091 -1.2152872 -0.6076436 -0.9114654 -0.6076436 -1.2152872 -0.4557327 -0.9114654 -1.2152872 -1.2152872 -0.9114654 -0.9114654 -1.2152872
[14] -0.4557327 -0.9114654
> page_rank(g)
$vector
[1] 0.07832946 0.10287002 0.05526825 0.07624676 0.05222240 0.05365783 0.05540849 0.09768430 0.05557930 0.07764650 0.05275776 0.07575701 0.05548108
[14] 0.05540849 0.05568235
$value
[1] 1
$options
NULL
> library(centiserve)
Loading required package: Matrix
> katzcent(g)
[1] 1.407434 1.524775 1.268857 1.420937 1.295420 1.281939 1.280531 1.542237 1.267753 1.398274 1.294051 1.411960 1.279253 1.280531 1.267629
> bottleneck(g)
[1] 11 12 3 9 0 7 0 9 9 8 4 5 7 0 3
> library(linkcomm)
Loading required package: RColorBrewer
Welcome to linkcomm version 1.0-14
For a step-by-step guide to using linkcomm functions:
> vignette(topic = "linkcomm", package = "linkcomm")
To run an interactive demo:
> demo(topic = "linkcomm", package = "linkcomm")
To cite, see:
> citation("linkcomm")
NOTE: To use linkcomm, you require read and write permissions in the current directory (see: help("getwd"), help("setwd"))
> communitycent(g)
[1] 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0
> #laplacian centrality
> laplacian(g)
[1] 28 38 16 30 20 18 18 40 16 26 20 28 18 18 16
Environment History Connections Tutorial
Import Dataset
Global Environment
Data
g List of 10
graph2 int [1:19, 1:2] 1 1 1 2 2 2 3 3 4 4 ...
Values
infocent num [1:15] 0.842 0.772 0.683 0.824 0.644 ...
Files Plots Packages Help Viewer
Zoom Export Publish
11
10
5
8
4
9
12
3
13
15
6
1
2
7
14
05 April 2021
Monday
Type here to search
ENG
18:13
05-04-2021