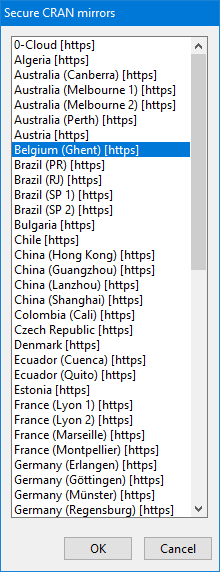
**PRACTICAL 01**

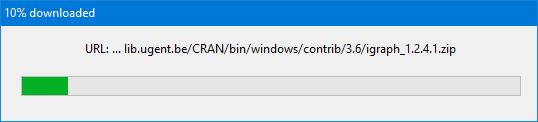
**Aim**: Write a program to compute the following for a given a network: (i) number of edges, (ii) number of nodes; (iii) degree of node; (iv) node with lowest degree; (v) the adjacency list; (vi) matrix of the graph

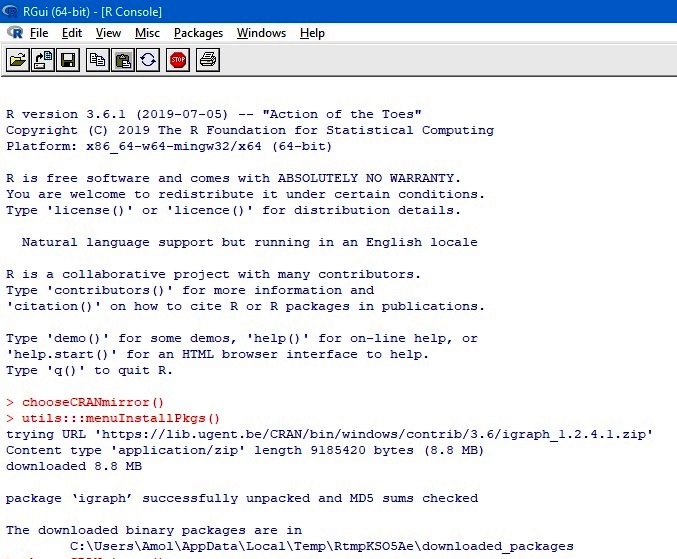
1. Launch **RGui.** Go to: **Packages > Set CRAN Mirrors**.

Select a Mirror-Site.

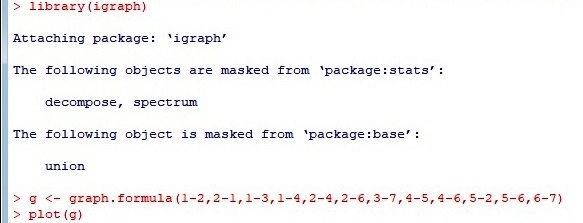


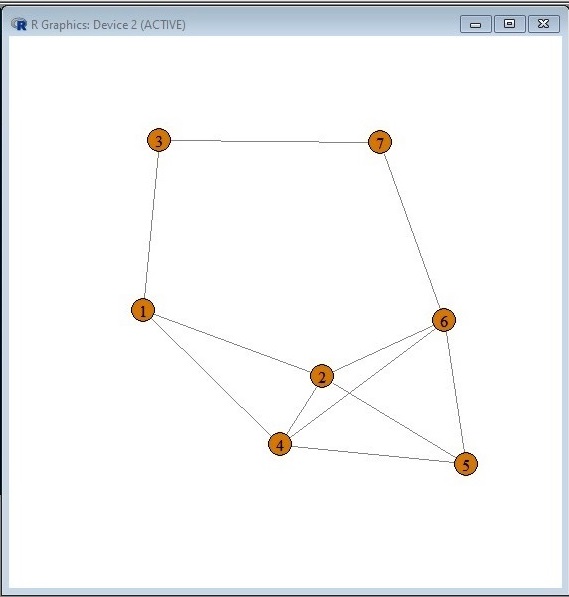
1. To Install a Package, go to: **Packages > Install package(s)…** and select a Package and click OK.



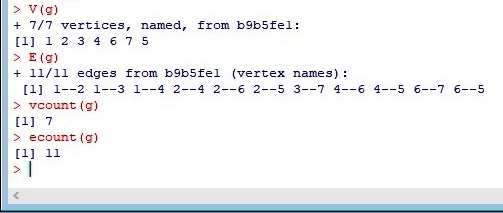


1. Install Packages: **igraph, igraphdata, sna, Dominance -** in the above Manner.
2. Load library **igraph**
3. Formulate and Plot a Graph.





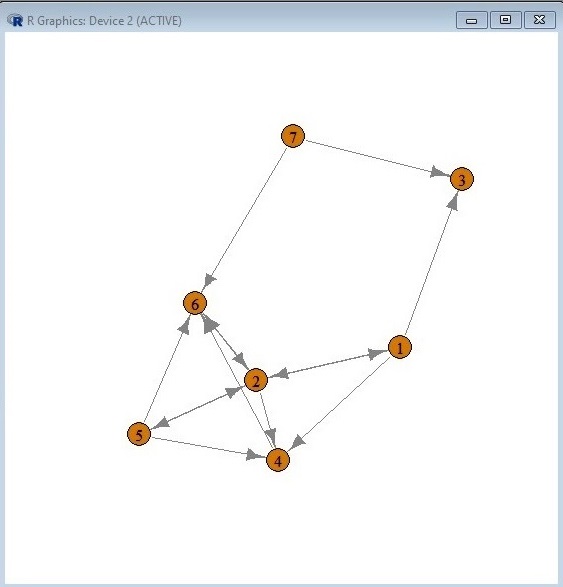
1. Name of Edges & Nodes. The Count of their Vertices (Nodes) and of their Edges, Dyads, Ties.



1. **Degree** of a **Node**

Directed Graph:

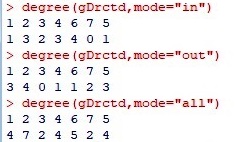




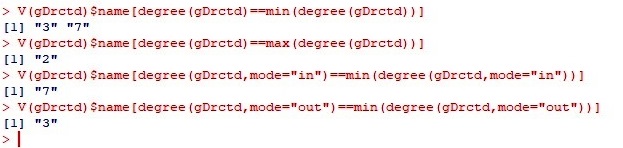
1. Degree of the Graph



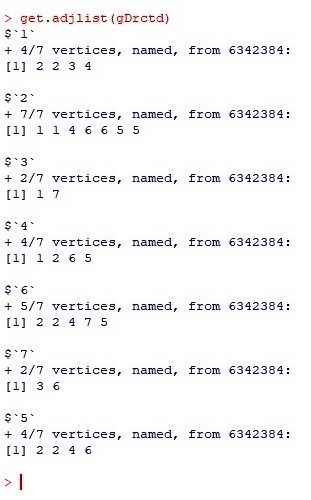
1. In-degree, Out-degree, In-degree & Out-degree both.



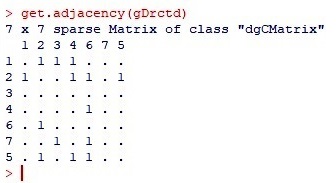
1. Node with the Lowest and Highest Degrees, Lowest In-Degrees & Lowest Out-Degrees.



1. To find **Neighbors** & **Adjacency List**

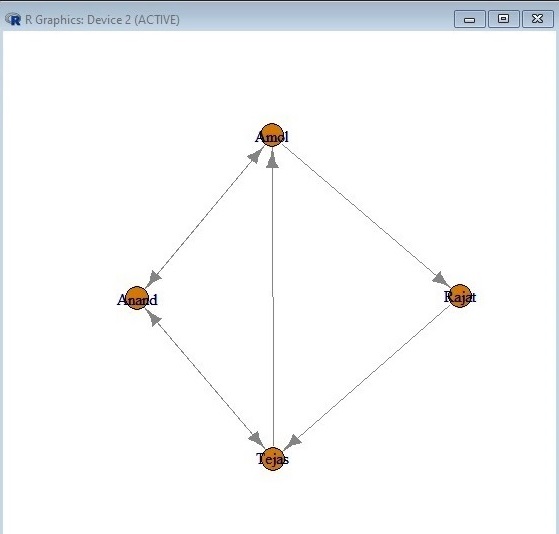


1. To find **Adjacency Matrix**.



1. Graph with **Names**





**PRACTICAL 02**

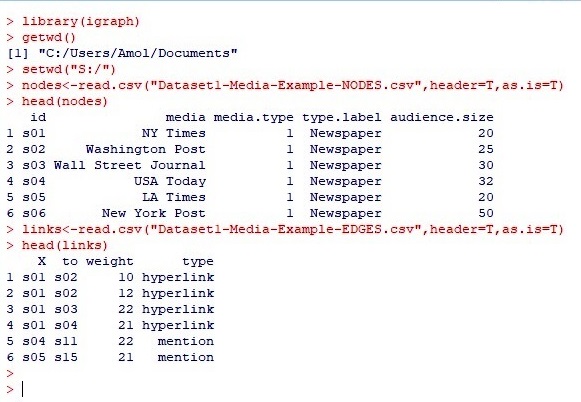
**Aim**: (i) View data collection forms and/or import one-mode/two-mode datasets; (ii) Basic Networks matrices transformations

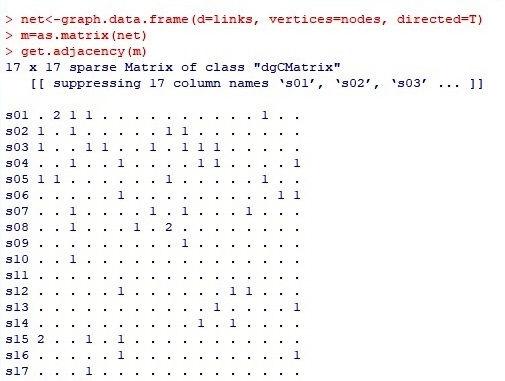
1. Save following files in some Folder or, (as here, in S:\) directly in some Drive:

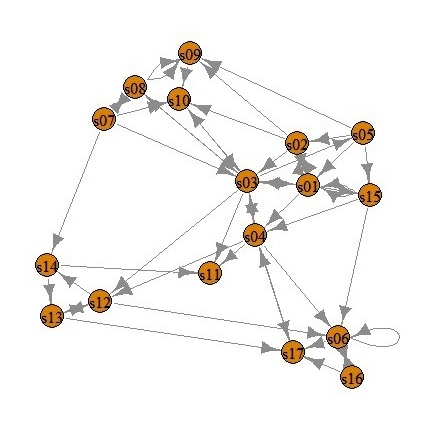
Dataset1-Media-Example-NODES.csv

Dataset1-Media-Example-EDGES.csv

data.txt

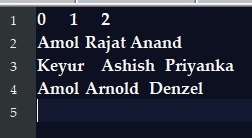


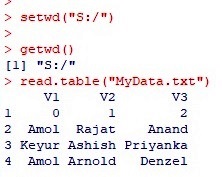




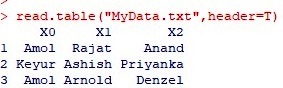
1. Reading Table from a Source File.

MyData.txt:





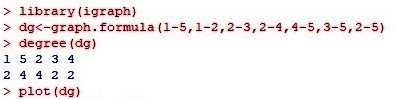
read.table("MyData.txt",header=T)

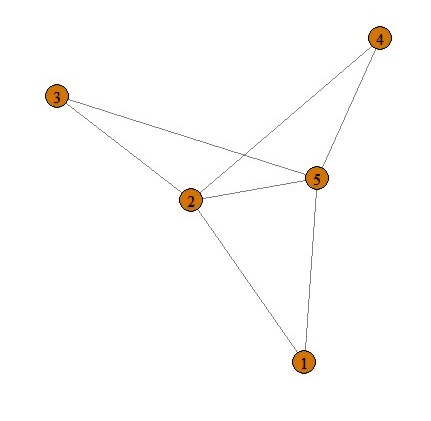


**PRACTICAL 03:**

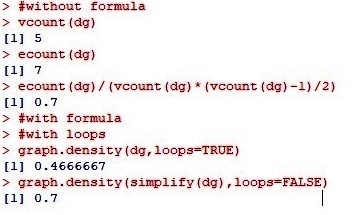
**Aim:** Compute the following node level measures: i) Density; ii) Degree; iii) Reciprocity; iv) Transitivity; v) Centralization; vi) Clustering.

1. **Degree**

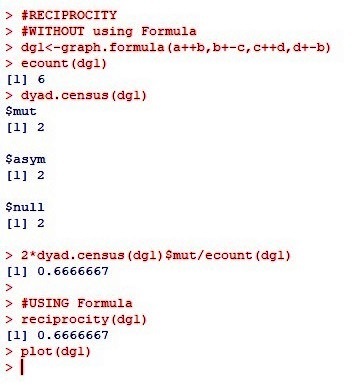


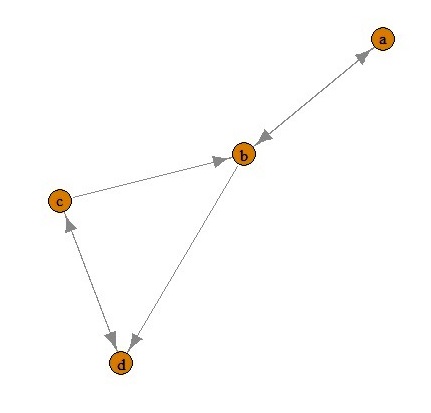


1. **Density**

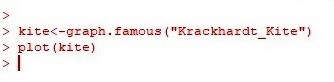
****

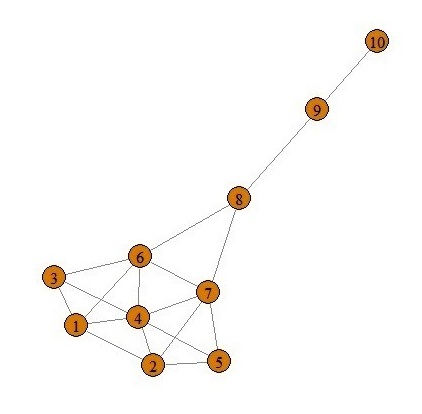
1. **Reciprocity**

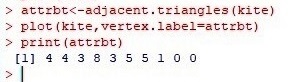
****

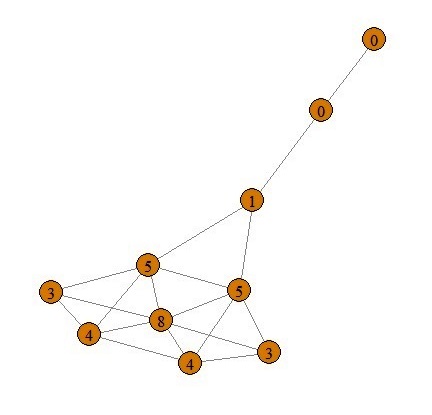


1. **Transitivity**



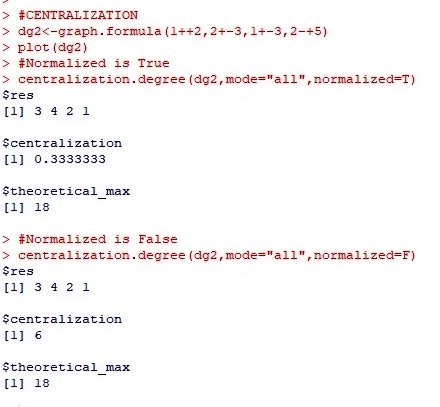


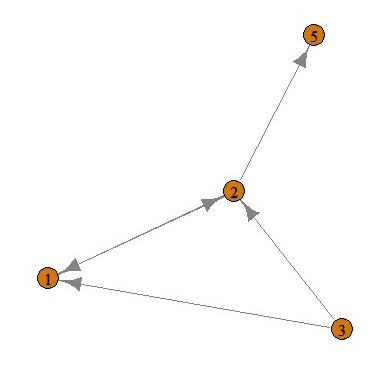


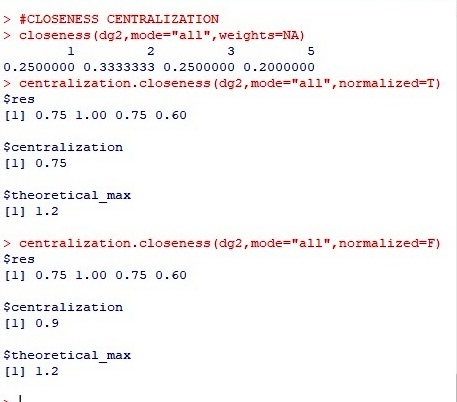


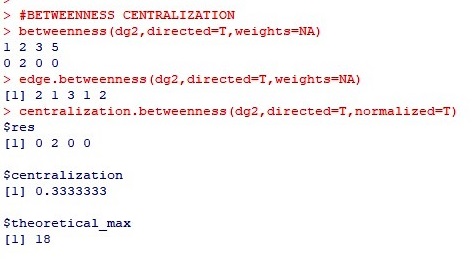


1. **Centralization**

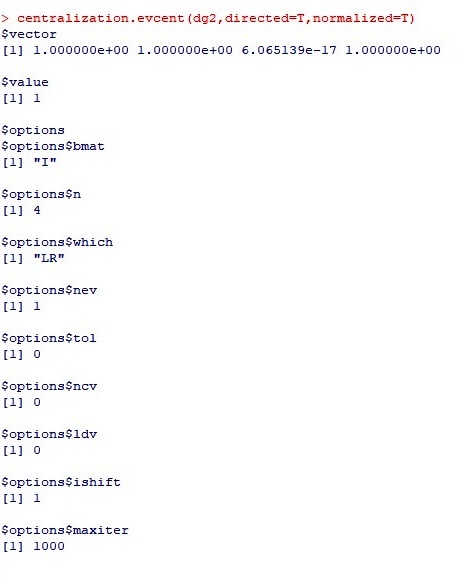








1. **Eigenvector centrality (e-v-cent)**

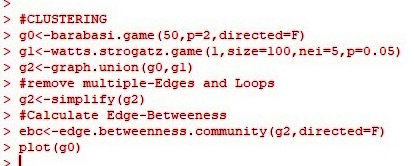
****

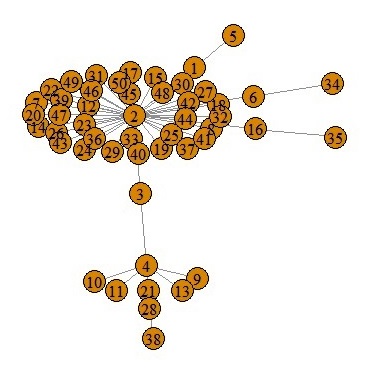
1. **Clustering**

g<-barabasi.game(50,p=2,directed=F)

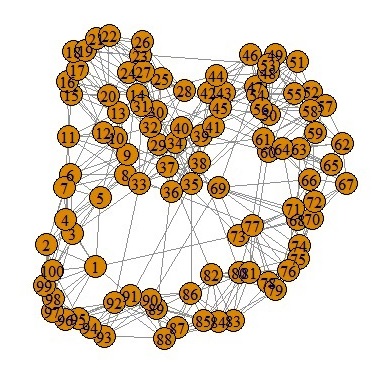
where:

p: Re-wiring Probability

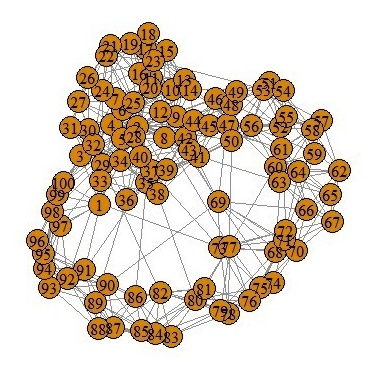


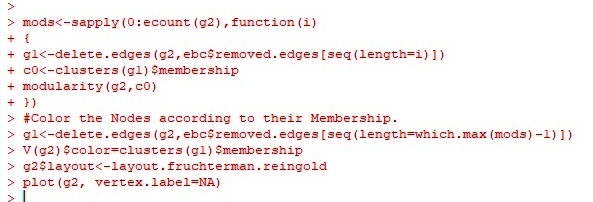


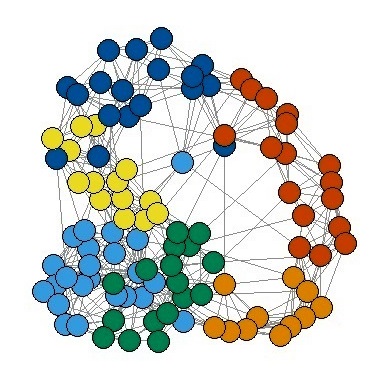
plot(g1)



plot(g2)



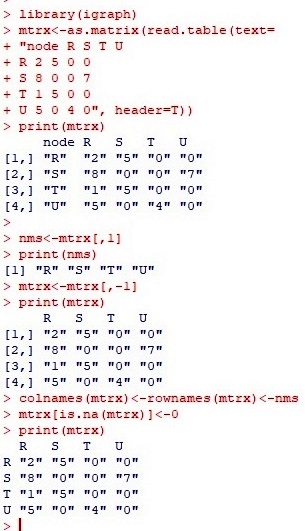
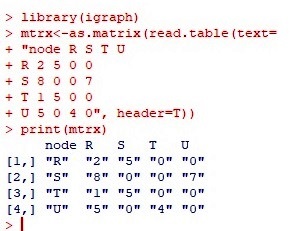


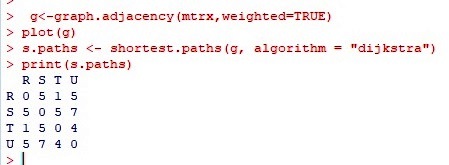


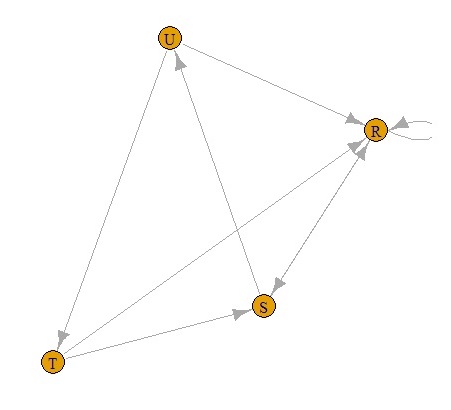
**PRACTICAL 04**

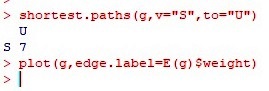
**Aim:** For a given network find the following: i) Length of the shortest path from a given node to another node; ii) the density of the graph; iii) Draw egocentric network of node G with chosen configuration parameters.

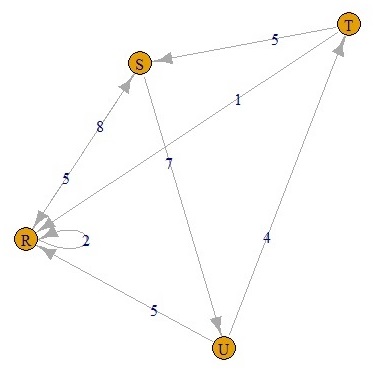
1. **Length of the shortest path from a given node to another node.**



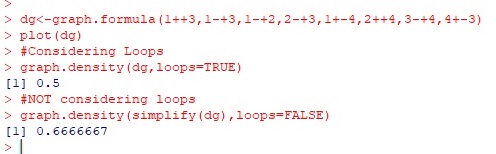


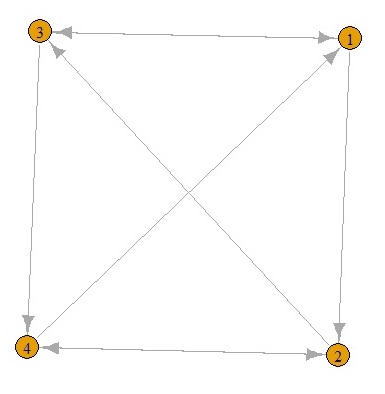






**Density of the graph**

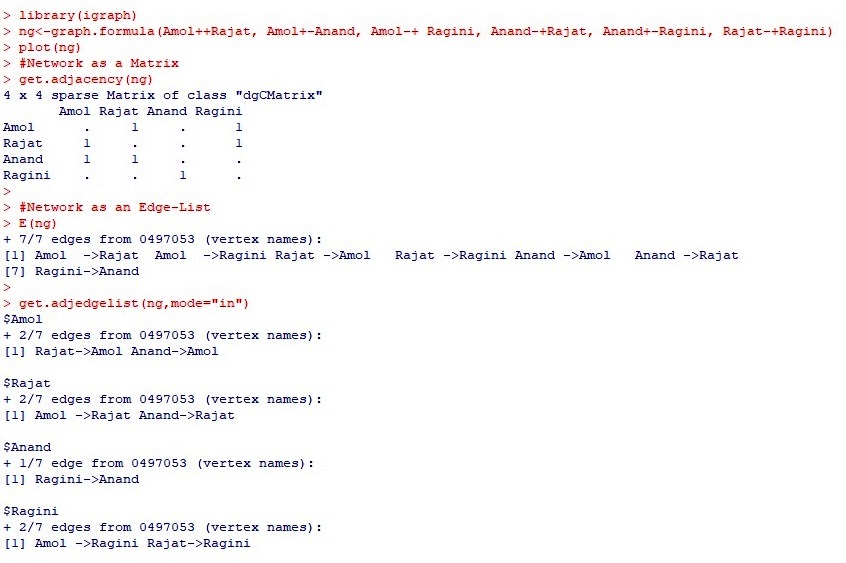


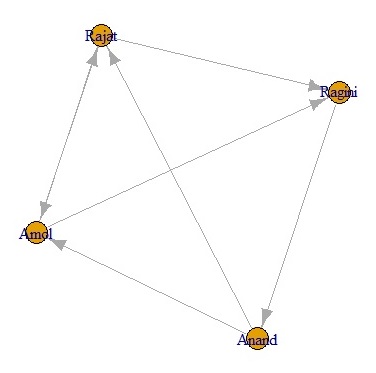


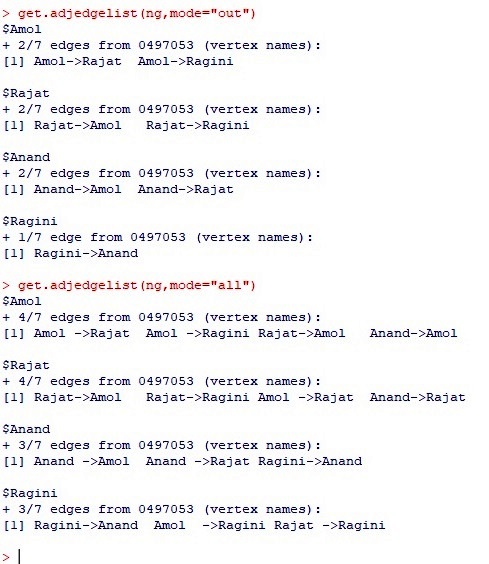
**PRACTICAL 05**

**Aim**: Write a program to distinguish between a network as a matrix, a network as an edge list, and a network as a sociogram (or “network graph”) using 3 distinct networks representatives of each.

**A Network as a Sociogram (or “Network Graph”)**

****

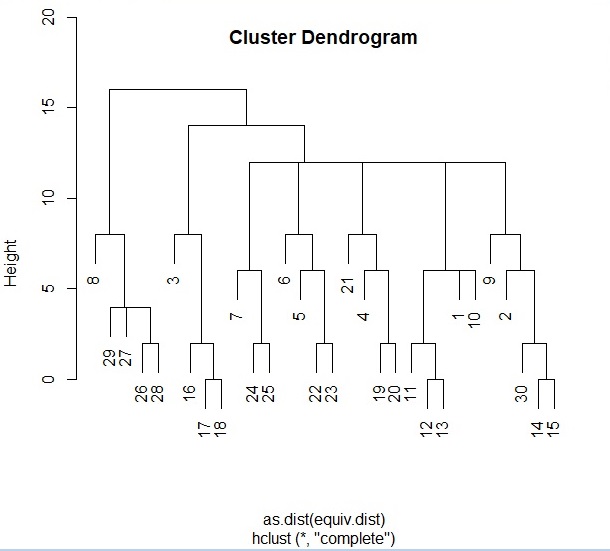
****

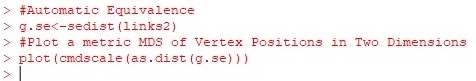
****

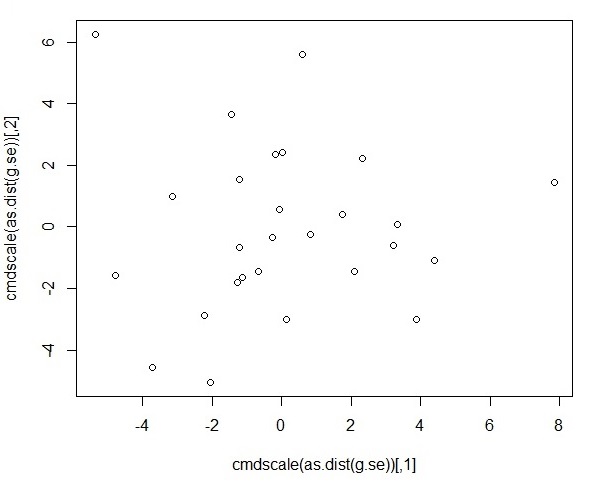
**PRACTICAL 06**

**Aim**: Write a program to exhibit structural equivalence, automatic equivalence, and regular equivalence from a network.

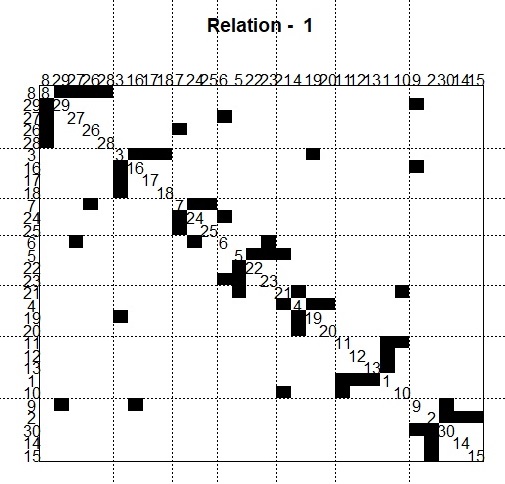




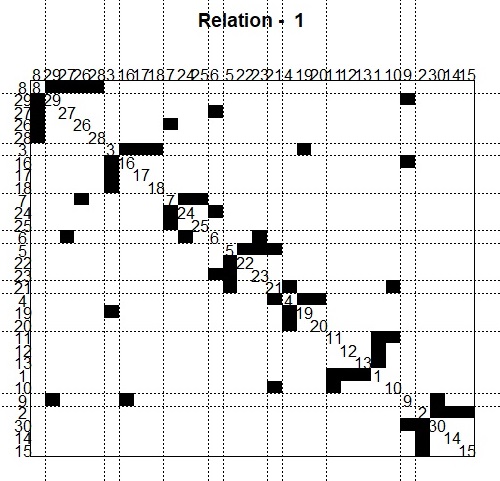




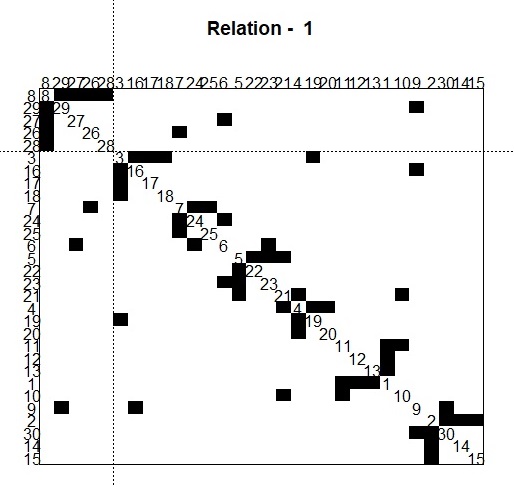




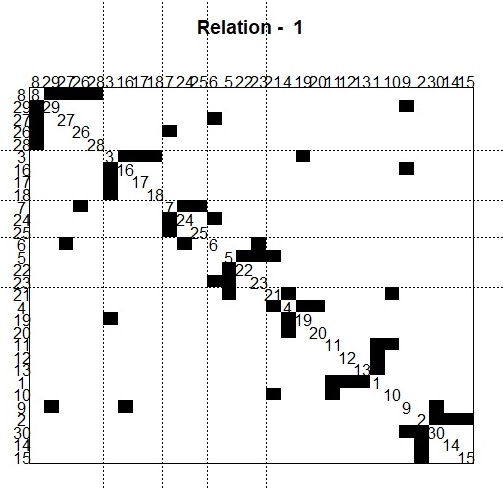




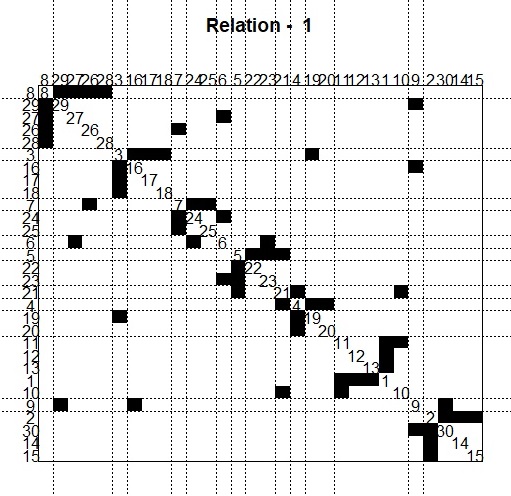








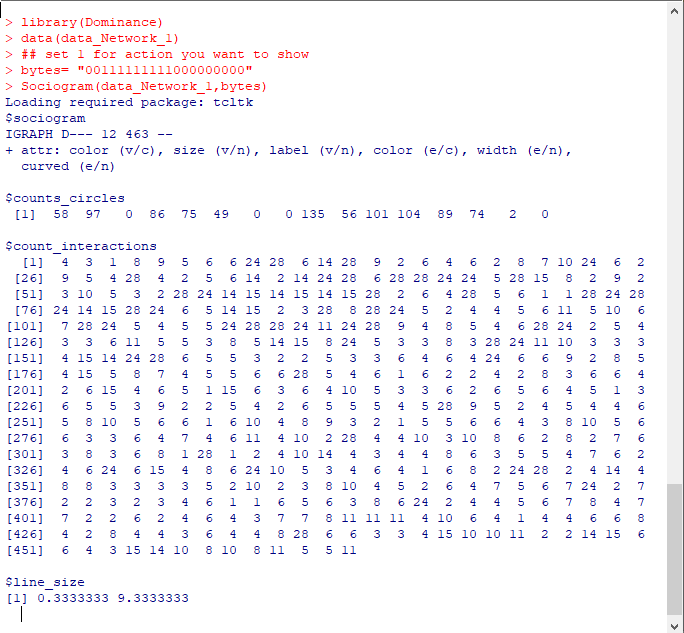


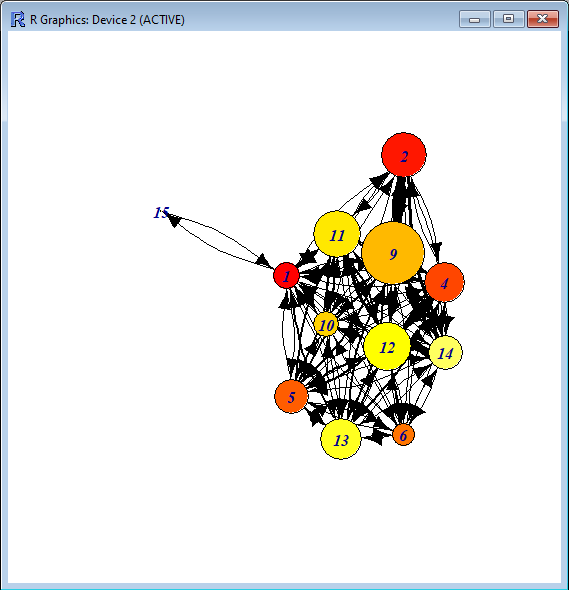


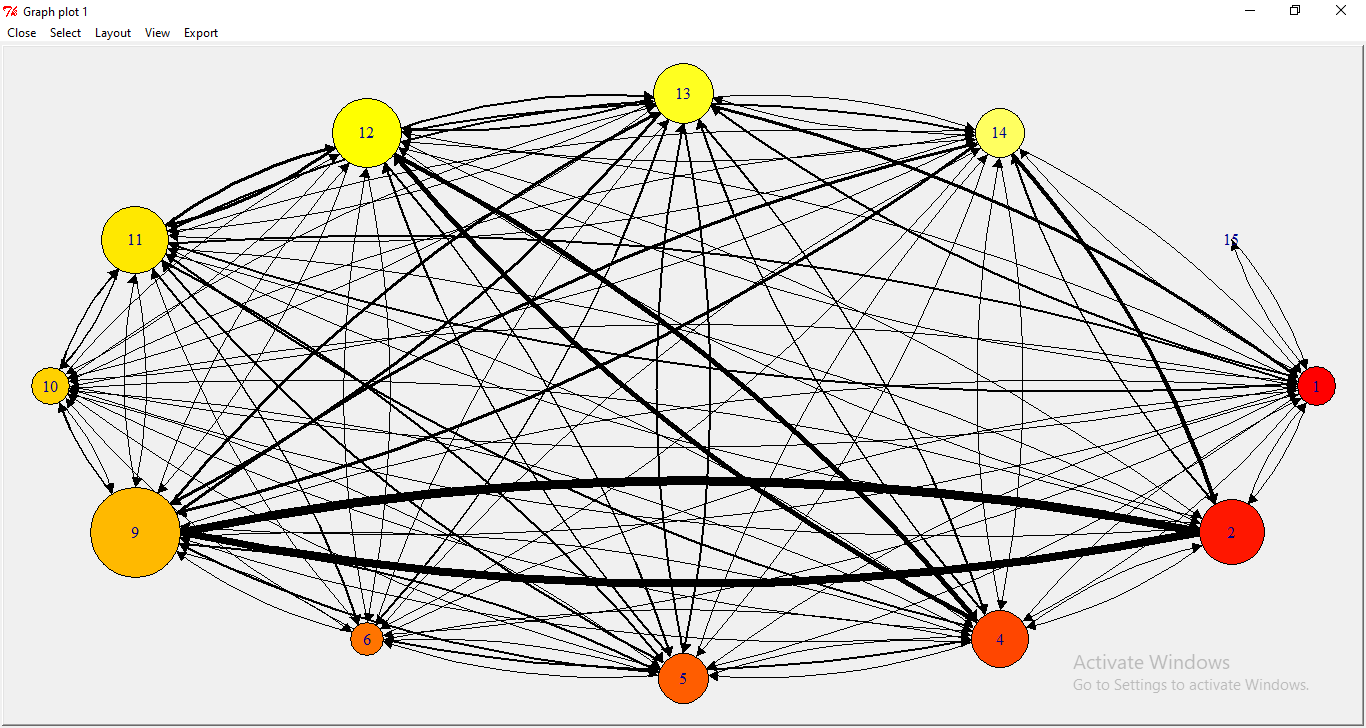


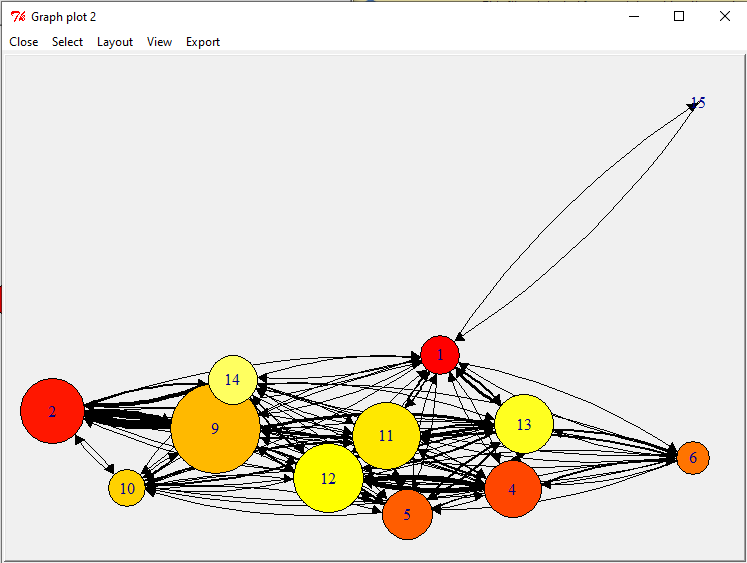
**PRACTICAL 07**

**Aim**: Create sociograms for the persons-by-persons network and the committee-by committee network for a given relevant problem. Create one-mode network and two-node network for the same.

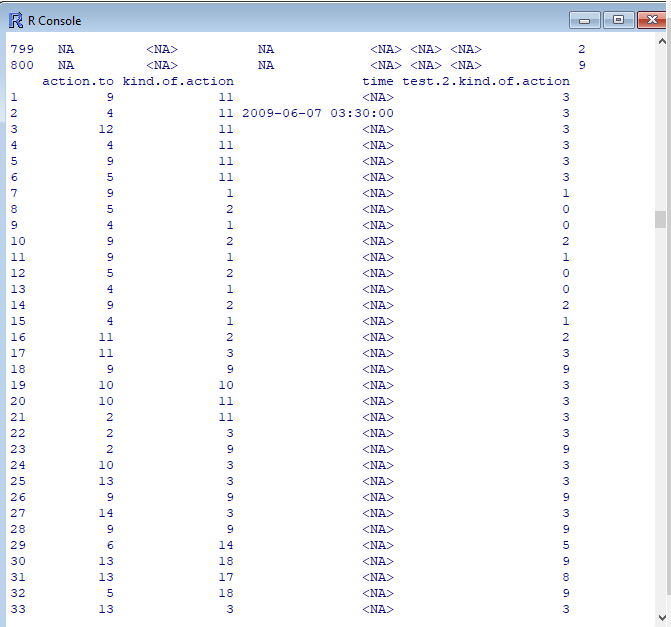






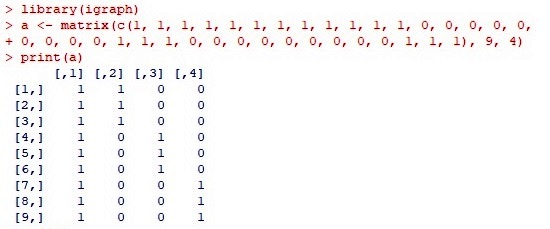


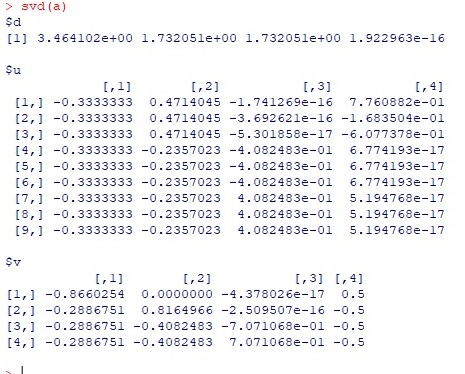
print(data\_Network\_1)



**PRACTICAL 08**

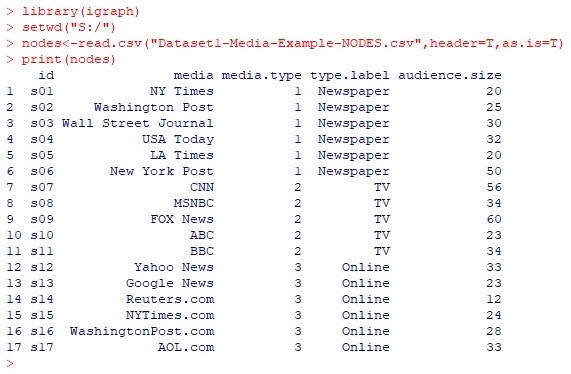
**Aim**: Perform SVD analysis of a network.

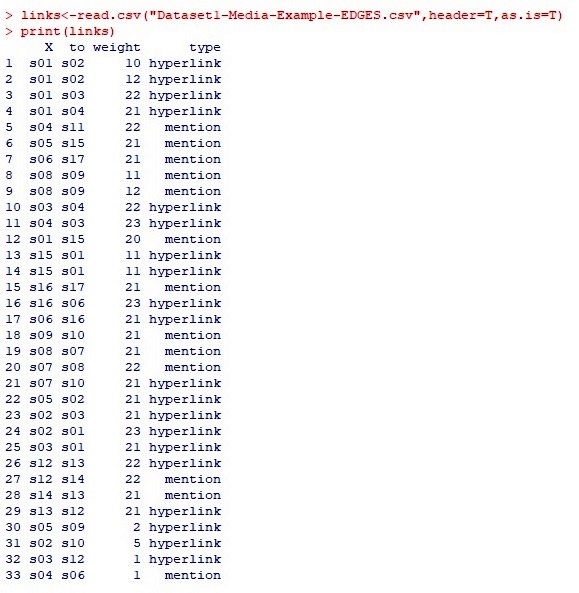
****

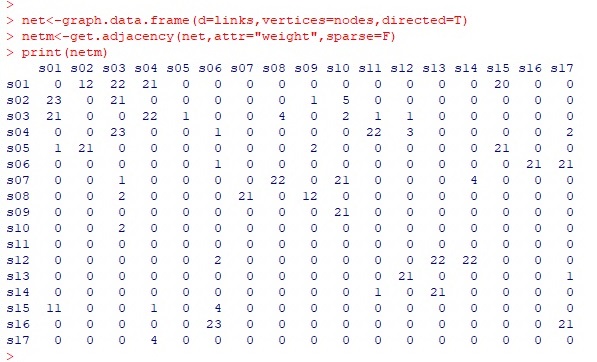
****

**PRACTICAL 09**

**Aim**: Identify ties within the network using two-mode core periphery analysis.

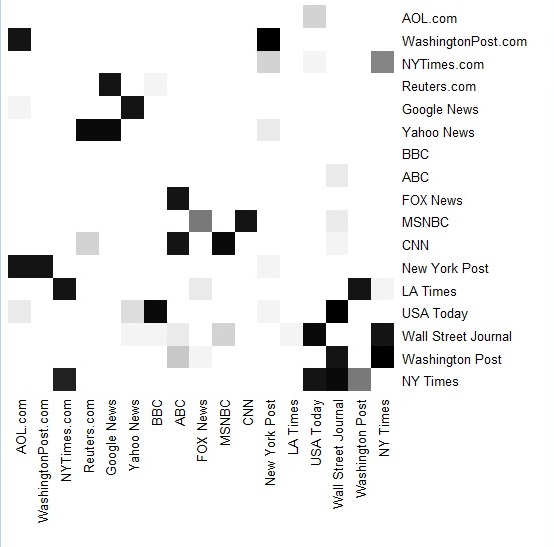
****

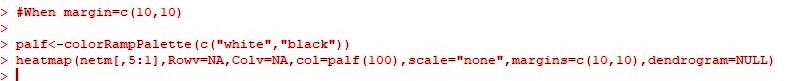
****

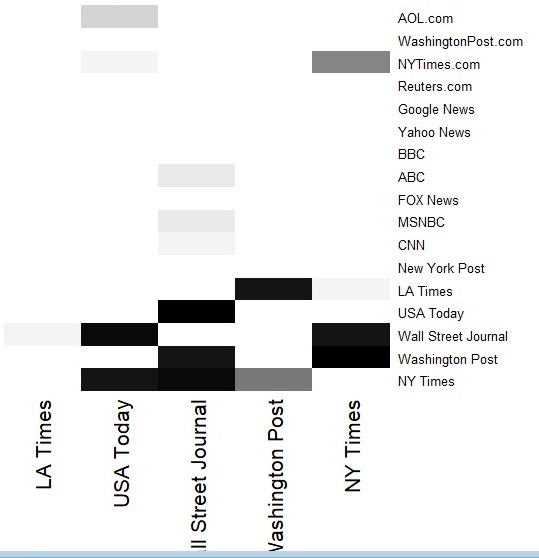
****

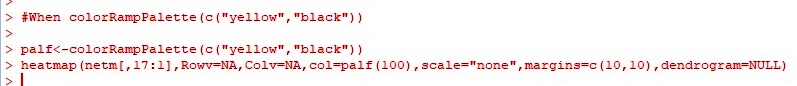
****

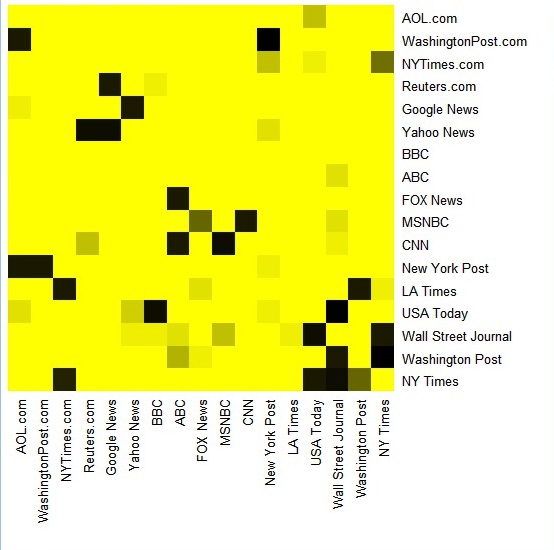
****

****

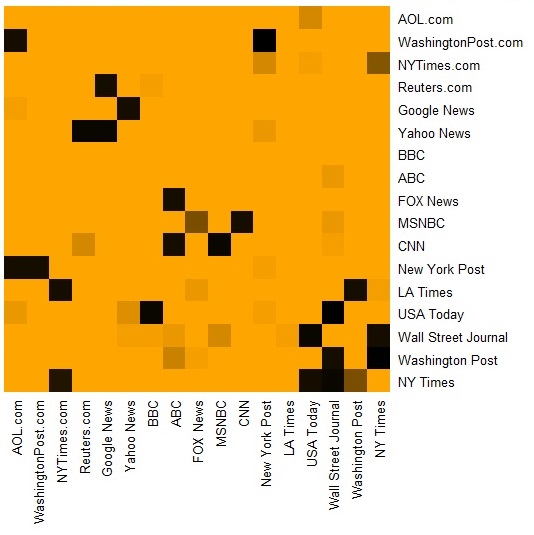
****

****

****

****

****

****

**PRACTICAL 10**

**Aim**: Find “factions” in the network using two-mode faction analysis.

