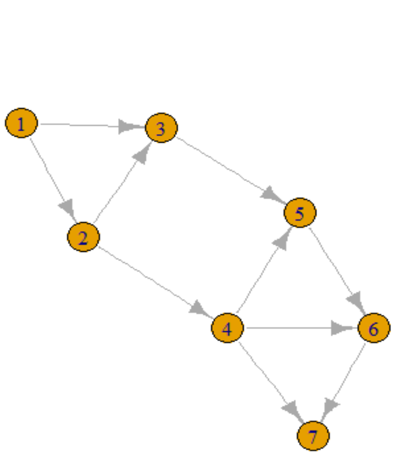
Practical 1

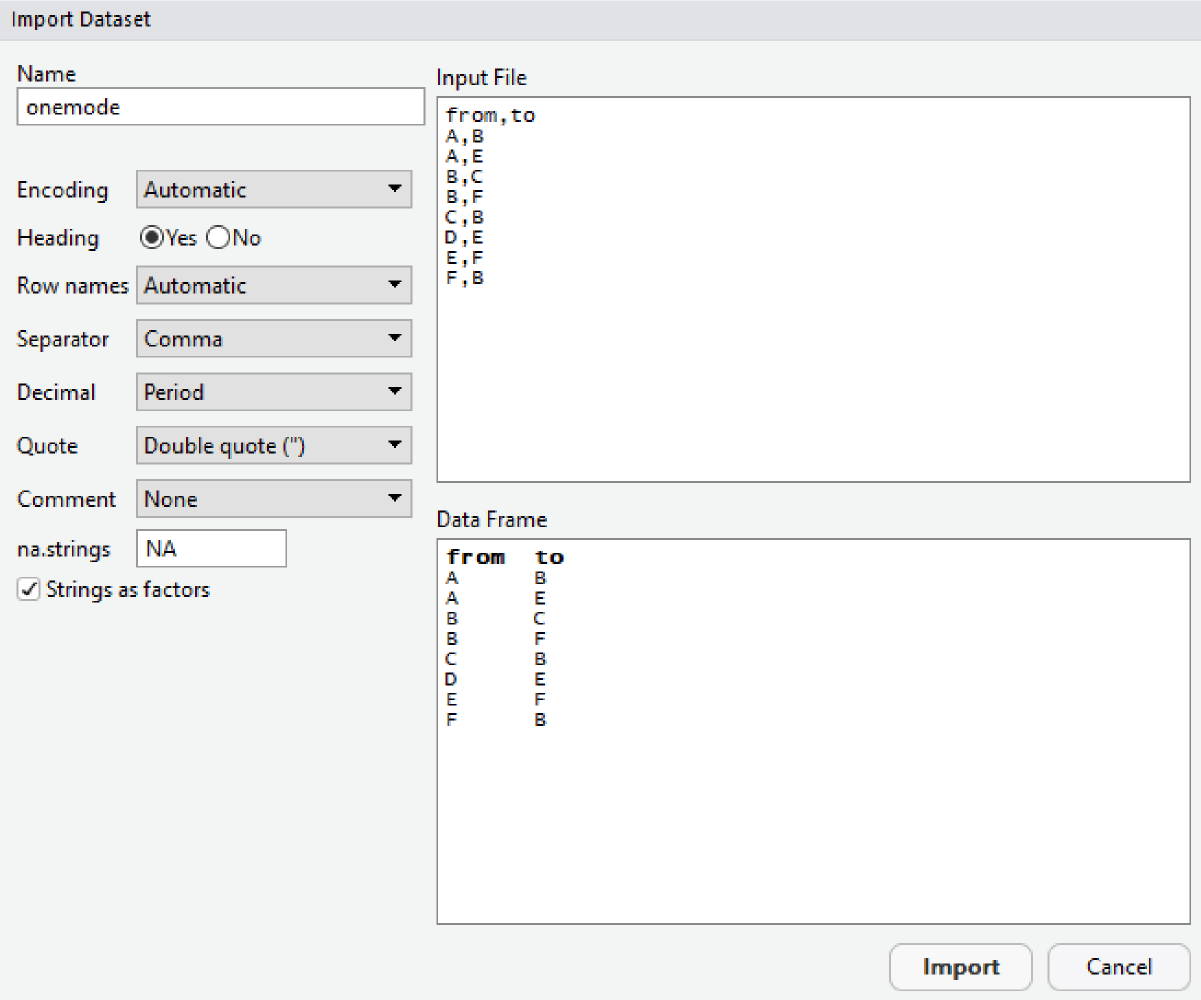
```
> library(igraph)
> library(igraphdata)
> #for directed graph
> dir=graph(edges = c(1,2,1,3,2,3,2,4,3,5,4,5,4,6,4,7,5,6,6,7),n=7,directed = T)
> #for undirected graph
> undir=graph(edges = c(1,2,1,3,2,3,2,4,3,5,4,5,4,6,4,7,5,6,6,7),n=8,directed = F)
> #plotting graph
> plot(dir)
> #return names of vertices
> V(dir)
+ 7/7 vertices:
[1] 1 2 3 4 5 6 7
> #returns edge names
> E(dir)
+ 10/10 edges:
 [1] 1->2 1->3 2->3 2->4 3->5 4->5 4->6 4->7 5->6 6->7
> #counts number of edges
> ecount(undir)
[1] 10
> #counts number of vertices
> vcount(dir)
[1] 7
> #returns names of neighbor node directed edges
> neighbors(dir,4)
+ 3/7 vertices:
[1] 5 6 7
> #counts number of degree for each node
> degree(dir)
[1] 2 3 3 4 3 3 2
> #to find minimum number of degrees in graph
> min(degree(dir))
[1] 2
> #to find names of minimum number of degrees in graph
> V(dir)[degree(dir)==min(degree(dir))]
+ 2/7 vertices:
[1] 1 7
>
> #to get adjacent list
> get.adjlist(dir)
[[1]]
+ 2/7 vertices:
[1] 2 3

[[2]]
+ 3/7 vertices:
[1] 1 3 4
```

```
> #to get edge list
> get.adjedgelist(dir)
[[1]]
+ 2/10 edges:
[1] 1->2 1->3

[[2]]
+ 3/10 edges:
[1] 2->3 2->4 1->2

[[3]]
+ 3/10 edges:
[1] 3->5 1->3 2->3

[[4]]
+ 4/10 edges:
[1] 4->5 4->6 4->7 2->4

[[5]]
+ 3/10 edges:
[1] 5->6 3->5 4->5

[[6]]
+ 3/10 edges:
[1] 6->7 4->6 5->6

[[7]]
+ 2/10 edges:
[1] 4->7 6->7

[[8]]
+ 0/10 edges:

> #to get adjacency matrix
> get.adjacency(dir)
8 x 8 sparse Matrix of class "dgCMatrix"

[1,] . 1 1 . . . . .
[2,] . . 1 1 . . . .
[3,] . . . . 1 . . .
[4,] . . . . 1 1 1 .
[5,] . . . . . 1 . .
[6,] . . . . . . 1 .
[7,] . . . . . . . .
[8,] . . . . . . . .
```

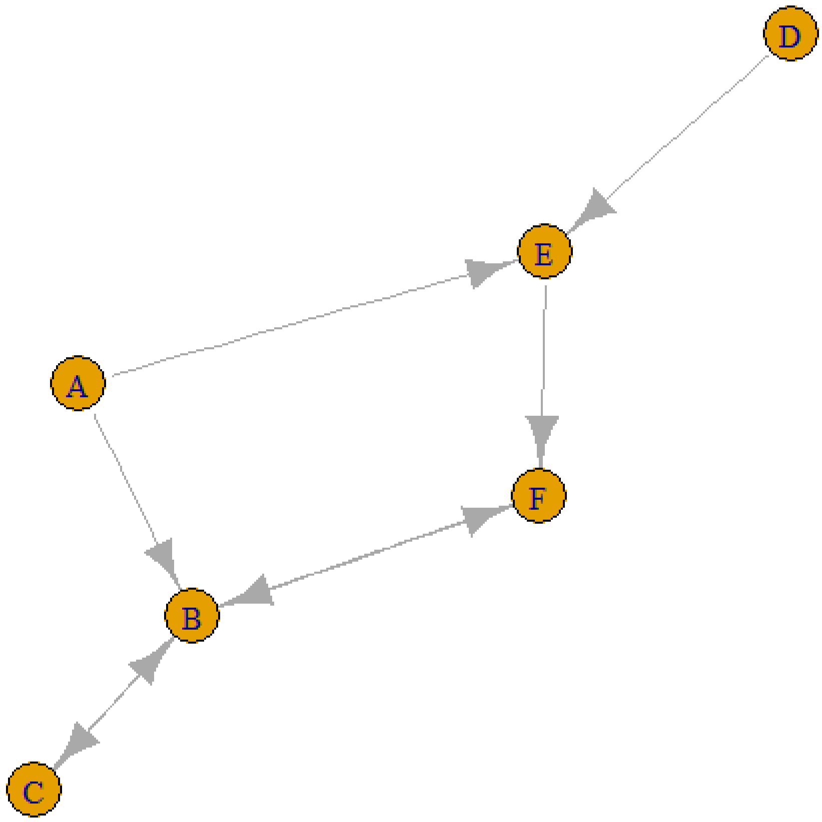


Practical 2

To import dataset, go to environment -> import dataset -> from text (base) -> select the csv and name them accordingly.




```
> library(igraph)
> library(Matrix)
>
> #are automatically executed when we import csv files
> onemode <- read.csv("~/onemode.csv")
> View(onemode)
>
> #to form a directed graph structure
> netgraph=graph.data.frame(onemode,directed = TRUE)
> plot(netgraph,edge.arrow.size=.8)
```



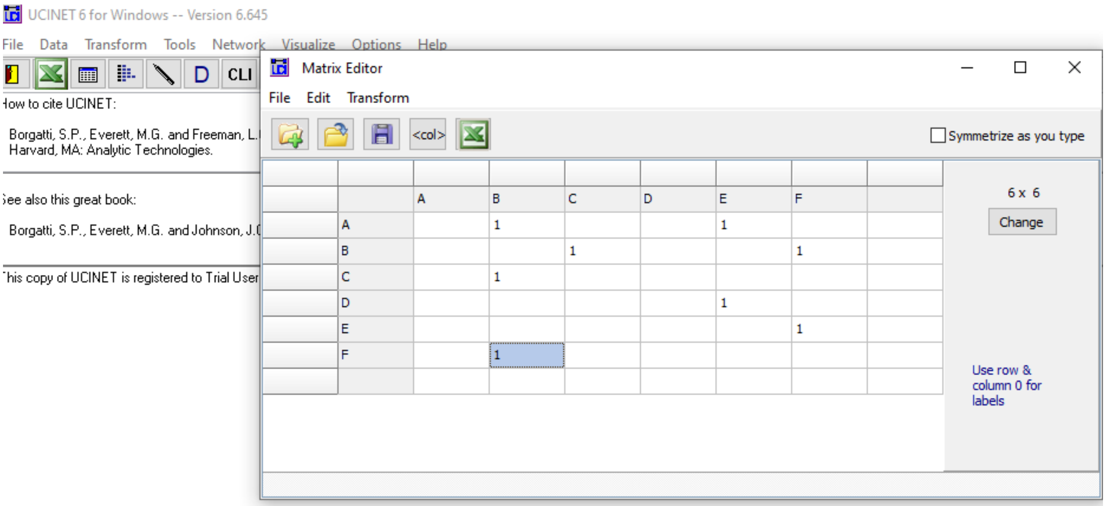

```
> #transforming to adjacency matrix (here we have 6 actors)
> mode_1=matrix(netgraph[],6,6)
> mode_1
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    0    1    0    0    1    0
[2,]    0    0    1    0    0    1
[3,]    0    1    0    0    0    0
[4,]    0    0    0    0    1    0
[5,]    0    0    0    0    0    1
[6,]    0    1    0    0    0    0
>
> #sum to get out degrees
> rowSums(mode_1)
[1] 2 2 1 1 1 1
> #sum to get in degrees
> colSums(mode_1)
[1] 0 3 1 0 2 2
>
> #transposing matrix for alternate view
> t(mode_1)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    0    0    0    0    0    0
[2,]    1    0    1    0    0    1
[3,]    0    1    0    0    0    0
[4,]    0    0    0    0    0    0
[5,]    1    0    0    1    0    0
[6,]    0    1    0    0    1    0
>
```

```
> #multiplying matrix to itself for walk of distance 2
> mode_squared=mode_1%*%mode_1
> mode_squared
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    0    0    1    0    0    2
[2,]    0    2    0    0    0    0
[3,]    0    0    1    0    0    1
[4,]    0    0    0    0    0    1
[5,]    0    1    0    0    0    0
[6,]    0    0    1    0    0    1
> #multiplying previous matrix to original for walk of distance 3
> mode_cubed=mode_squared%*%mode_1
> mode_cubed
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    0    3    0    0    0    0
[2,]    0    0    2    0    0    2
[3,]    0    2    0    0    0    0
[4,]    0    1    0    0    0    0
[5,]    0    0    1    0    0    1
[6,]    0    2    0    0    0    0
>
> #multiplying matrix to itself for knowing if a path of distance 2 exists
> mode_1_boolean_And=mode_1%&%mode_1
> mode_1_boolean_And
6 x 6 sparse Matrix of class "ngCMatrix"

[1,] . . | . . |
[2,] . | . . . .
[3,] . . | . . |
[4,] . . . . . |
[5,] . | . . . .
[6,] . . | . . |
```

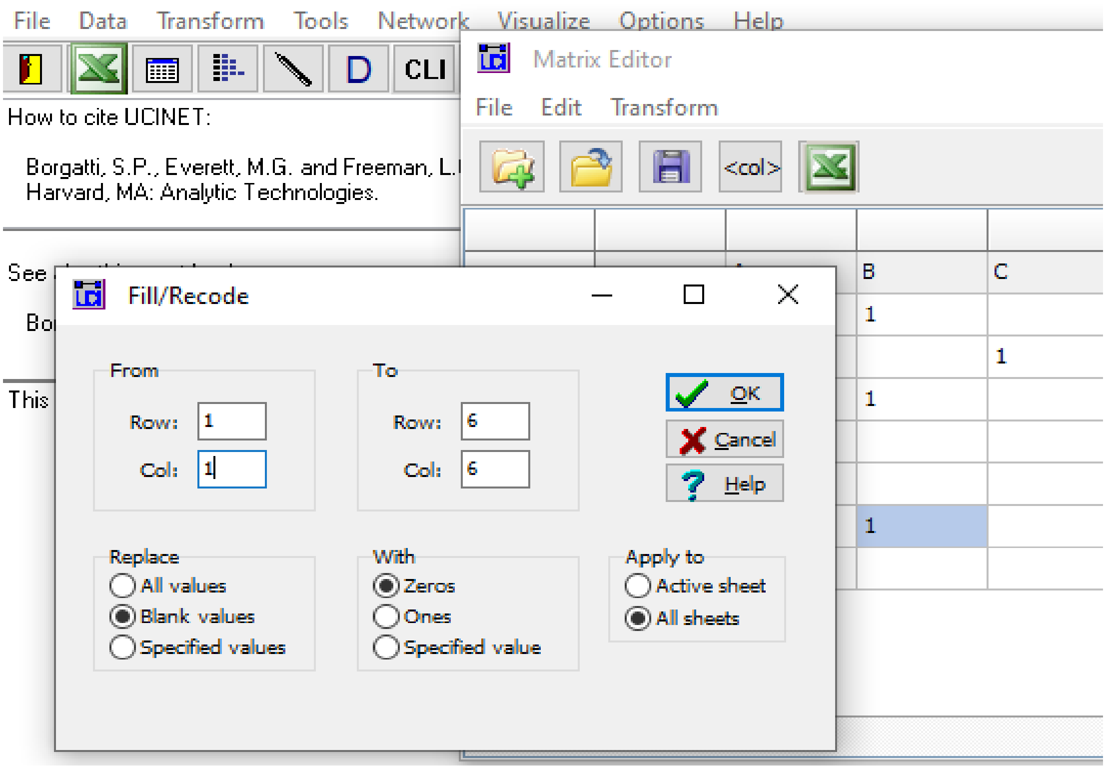
Practical 3

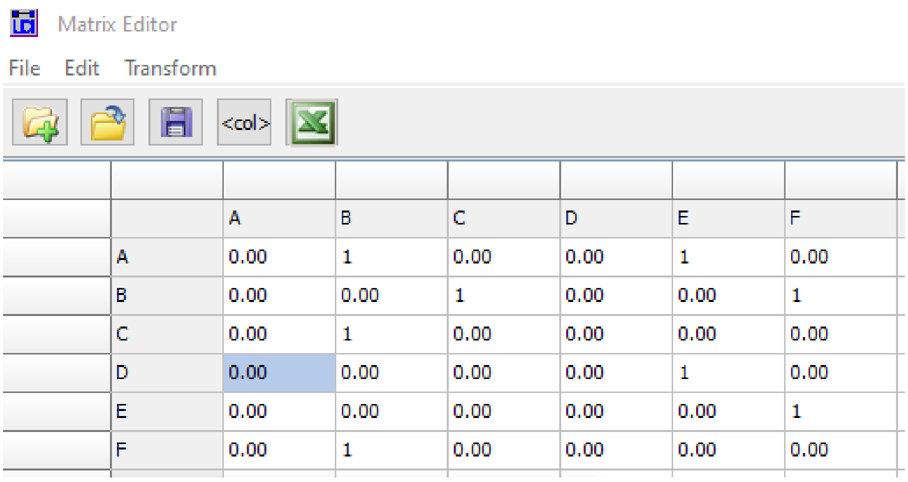
Open excel editor and just name a save a file name onemode in ucinet folder.

Open matrix editor and then open the file saved in excel editor, fill it as follows.



After filling 1's go to transform and fill it with zeros(select apply to active sheet), then save it.





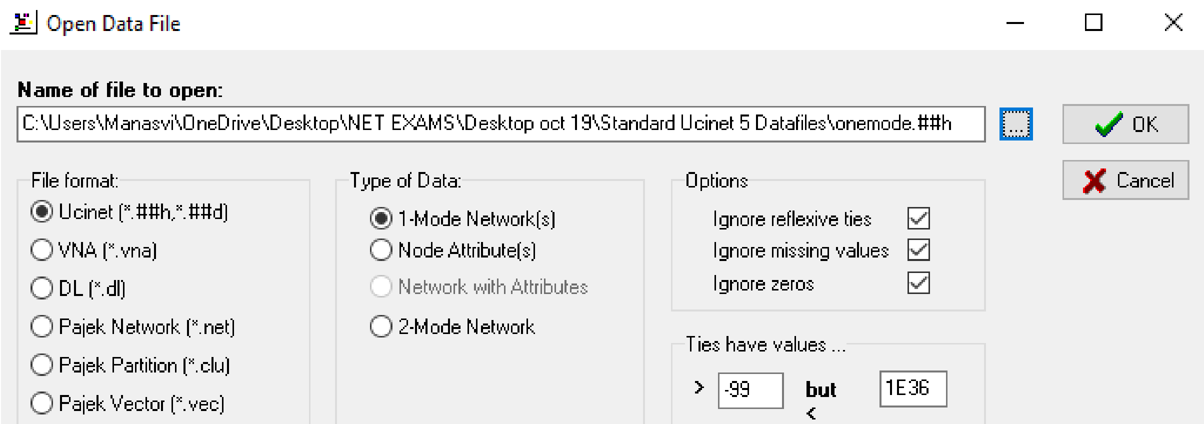
Matrix Editor

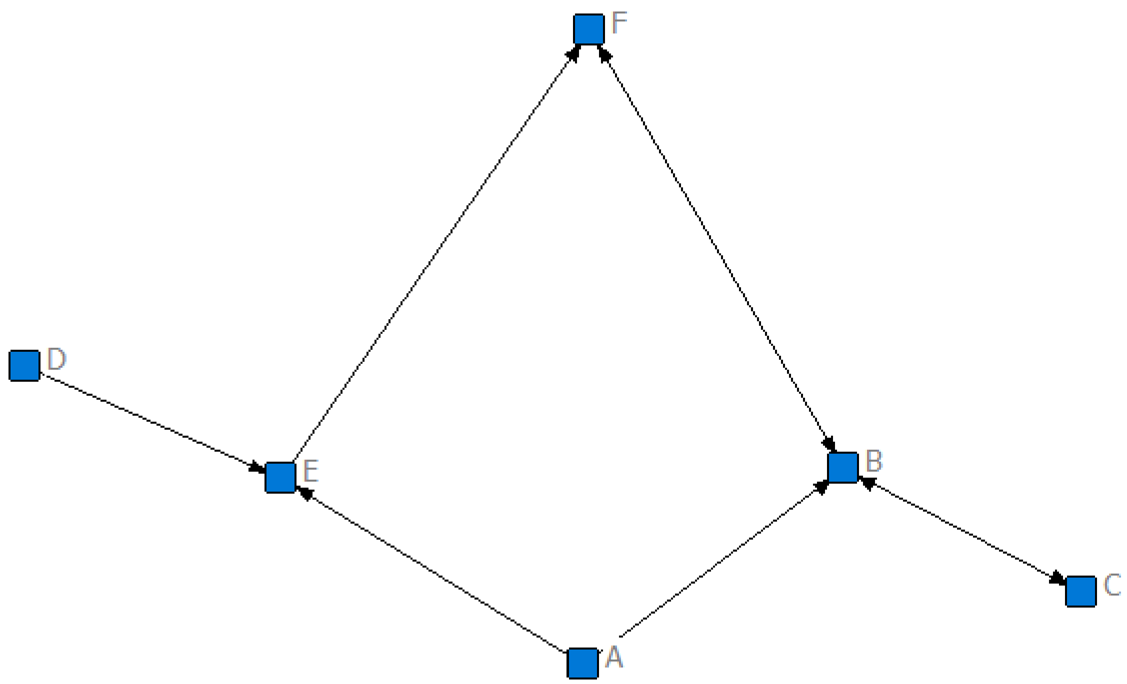
File Edit Transform

	A	B	C	D	E	F
A	0.00	1	0.00	0.00	1	0.00
B	0.00	0.00	1	0.00	0.00	1
C	0.00	1	0.00	0.00	0.00	0.00
D	0.00	0.00	0.00	0.00	1	0.00
E	0.00	0.00	0.00	0.00	0.00	1
F	0.00	1	0.00	0.00	0.00	0.00

Open netdraw then load the data (onemode) to visualize it.

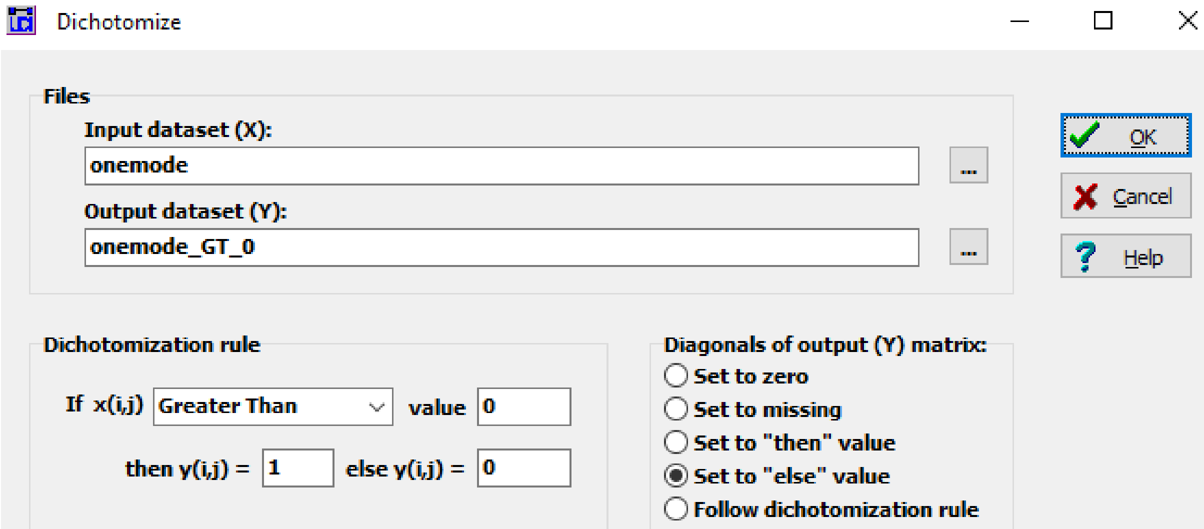
Netdraw => file>open>ucinet dataset>network





For density (ucinet)

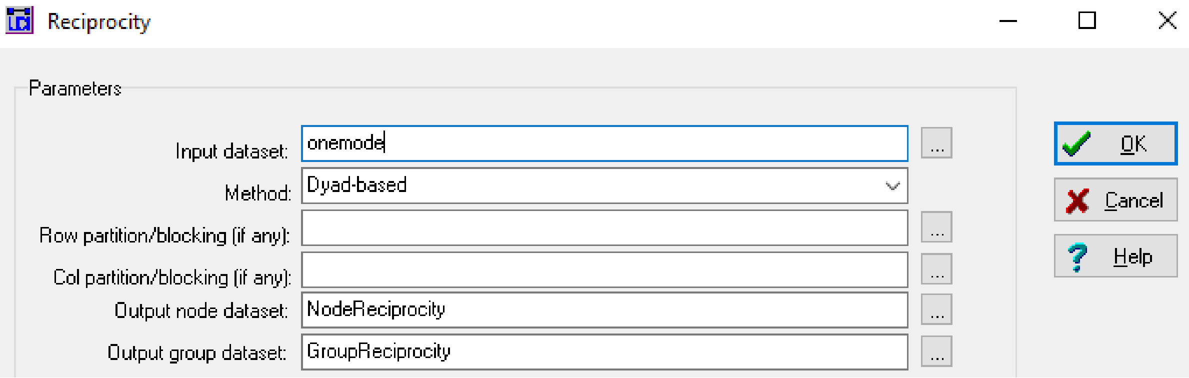
Transform>Dichotomize…




```
        1 2 3 4 5 6
        A B C D E F
        - - - - - -
    1 A 0 1 0 0 1 0
    2 B 0 0 1 0 0 1
    3 C 0 1 0 0 0 0
    4 D 0 0 0 0 1 0
    5 E 0 0 0 0 0 1
    6 F 0 1 0 0 0 0

6 rows, 6 columns, 1 levels.

Number of 1s: 8
Number of cells: 30
Density: 0.266666666666667


--------------------------------------
Running time: 00:00:01 seconds.
Output generated: 30 Oct 19 13:14:18
UCINET 6.645 Copyright (c) 1992-2017 Analytic Technologies
```

For reciprocity (ucinet)

>Network>Cohesion>Reciprocity (select method as Dyad-based)



Recip Arcs (Shows how many arcs are reciprocated)

Unrecip Arcs (Shows opposite of above)

All Arcs (Total number of arcs)

Arc Reciprocity (% of how many are reciprocated)

Sym Dyads(Number of reciprocated dyads)

Asym Dyads (Opposite of above)

All Dyads (Total number of dyads)

Dyad Reciprocity (% of reciprocated dyads)

Overall Reciprocity Measures

		1 Measures
1	Recip Arcs	4
2	Unrecip Arcs	4
3	All Arcs	8
4	Arc Reciprocity	0.500
5	Sym Dyads	2
6	Asym Dyads	4
7	All Dyads	6
8	Dyad Reciprocity	0.333

8 rows, 1 columns, 1 levels.

Arc and dyad measures are explained here:
https://sites.google.com/site/ucinetsoftware/document/faq/reciprocity--arcordyad

Dyad-based Reciprocity: 0.3333

In the dyad-based method, the reciprocity value indicates the prop. of dyads that are reciprocal.
I.e., Num(Xij>0 and Xji>0)/Num(Xij>0 or Xji>0)

Node-level Reciprocity Statistics -- All values are Proportions

	1 Symmetric	2 Non-Symme	3 Out/NonSy	4 In/NonSym	5 Sym/Out	6 Sym/In
1 A	0.000	1.000	1.000	0.000	0.000	
2 B	0.667	0.333	0.000	1.000	1.000	0.667
3 C	1.000	0.000			1.000	1.000
4 D	0.000	1.000	1.000	0.000	0.000	
5 E	0.000	1.000	0.333	0.667	0.000	0.000
6 F	0.500	0.500	0.000	1.000	1.000	0.500

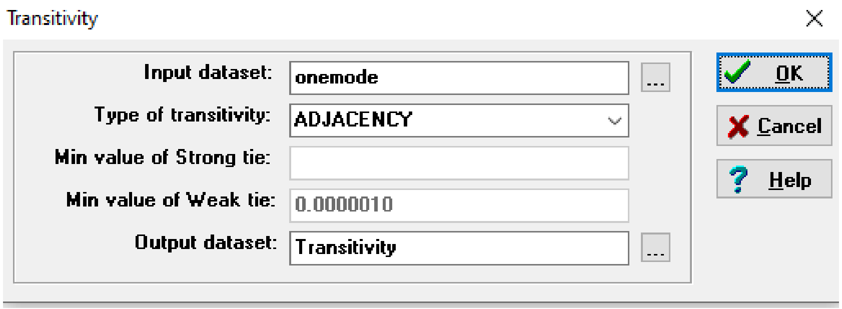
"Symmetric" gives proportion of ego's *undirected* contacts with whom ego has reciprocated ties.
"Non-Symmetric" is 1 - Symmetric
"Out/Non-Sym" gives proportion of ego's non-symmetric ties that are outgoing
"In/Non-Sym" gives proportion of ego's non-symmetric ties that are incoming
"Sym/Out" gives proportion of ego's outgoing ties that are reciprocated
"Sym/In" gives proportion of ego's incoming ties that are reciprocated

Group reciprocity table saved as dataset: GroupReciprocity
Node-level reciprocity saved as dataset: NodeReciprocity

Running time: 00:00:01
Output generated: 30 Oct 19 13:39:45

For Transitivity (Ucinet)

>Network>Cohesion>_Transitivity(legacy)



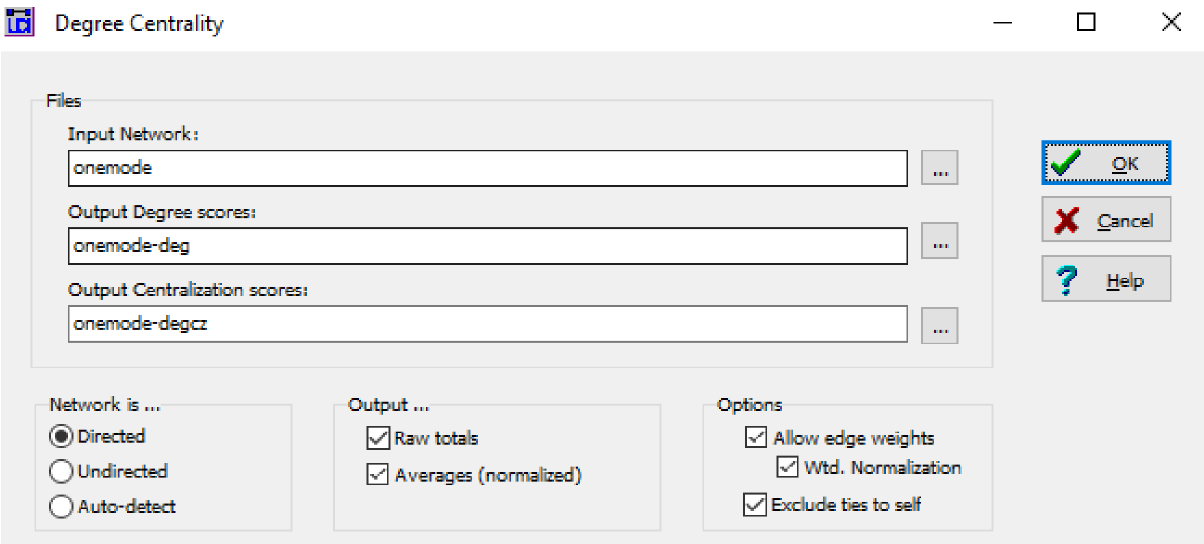
Here we don’t have any transitive tie so it will show 0% transitivity, we simply edit matrix and add a tie FC, CF.

Then output will be,

```
TRANSITIVITY
--------------------------------------------------------------------------------

Type of transitivity:                   ADJACENCY
Input dataset:                          onemode (C:\Users\Manasvi\OneDrive\Desktop\NET I

Number of non-vacuous transitive ordered triples: 6
Number of triples of all kinds: 120
Number of triples in which i-->j and j-->k: 12
Number of triangles with at least 2 legs: 24
Number of triangles with 3 legs: 6

Percentage of all ordered triples: 5.00%
Transitivity: % of ordered triples in which i-->j and j-->k that are transitive: 50.00%
Transitivity: % of triangles with at least 2 legs that have 3 legs: 25.00%


UNDIRECTED GRAPHS:

No. of triples with all 3 legs: 1
No. of triples with at least 2 legs: 4
Transitivity: 25.00%


Network Transitivity

               1
              Tr
              --
    1  Sheet1  25
```

For Degree and centrality (Ucinet)

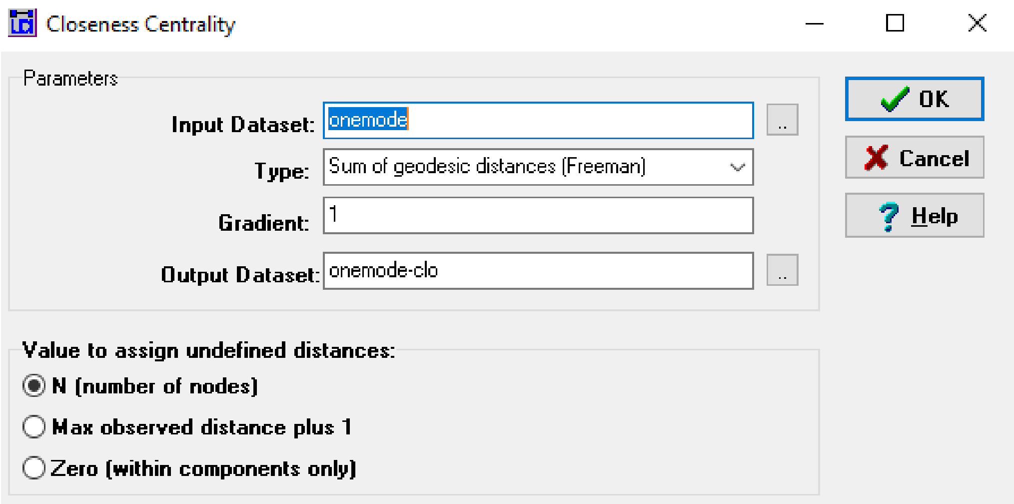
>Network>Centrality and Power>Degree




```
Degree Measures

            1     2     3     4
        Outde Indeg nOutd nInde
            g          eg     g
        ----- ----- ----- -----
   1 A  2.000 0.000 0.400 0.000
   2 B  2.000 3.000 0.400 0.600
   3 C  2.000 2.000 0.400 0.400
   4 D  1.000 0.000 0.200 0.000
   5 E  1.000 2.000 0.200 0.400
   6 F  2.000 3.000 0.400 0.600


6 rows, 4 columns, 1 levels.
```

```
                 1      2
            Out-Ce In-Cen
            ntrali traliz
            zation  ation
            ------ ------
   1 Sheet1 0.0800 0.3200

1 rows, 2 columns, 1 levels.
```

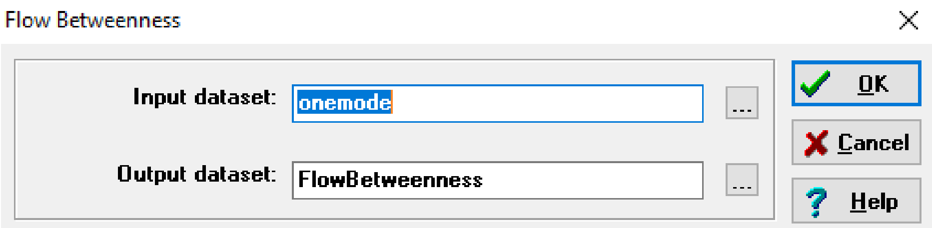
>Network>Centrality and Power>Closeness centrality(old)



Closeness Centrality Measures

		1 inFarness	2 outFarness	3 inCloseness	4 outCloseness
6	F	7.000	20.000	71.429	25.000
2	B	8.000	20.000	62.500	25.000
3	C	9.000	20.000	55.556	25.000
5	E	20.000	17.000	25.000	29.412
1	A	30.000	12.000	16.667	41.667
4	D	30.000	15.000	16.667	33.333

>Network>Centrality and Power>Flow Betweenness



FLOW BETWEENNESS CENTRALITY MEASURES

Input dataset: onemode (C:\Users\

Dataset is not symmetric.

	1	2	3	4	5	6
1	0	2	2	0	1	2
2	0	0	2	0	0	2
3	0	2	0	0	0	2
4	0	1	1	0	1	1
5	0	1	1	0	0	1
6	0	2	2	0	0	0

For Clustering

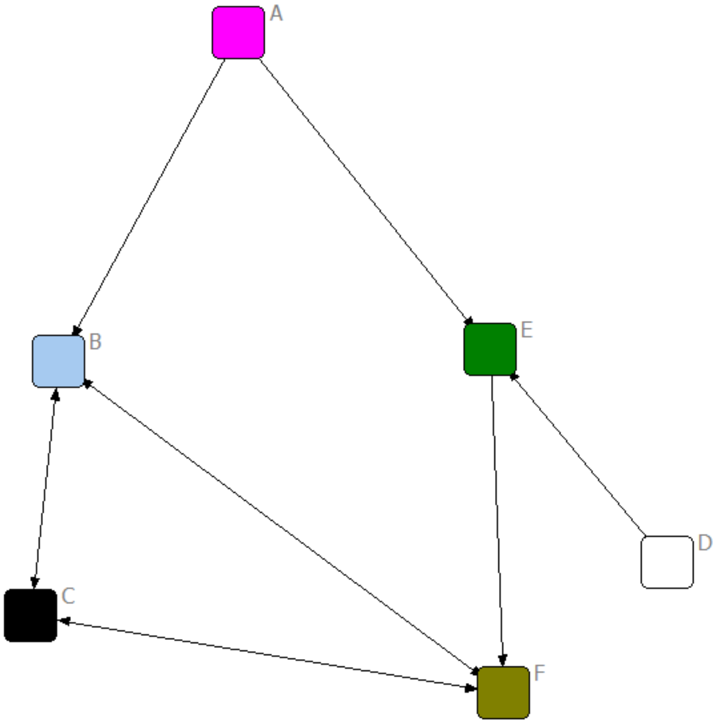
>Network>Cohesion>Clustering Coefficient.

```
Overall graph clustering coefficient: 0.333
Network density: 0.333
(cc - density)/cc: 0.000
Weighted Overall graph clustering coefficient: 0.273
Small world index: 4.233
```

Node Clustering Coefficients

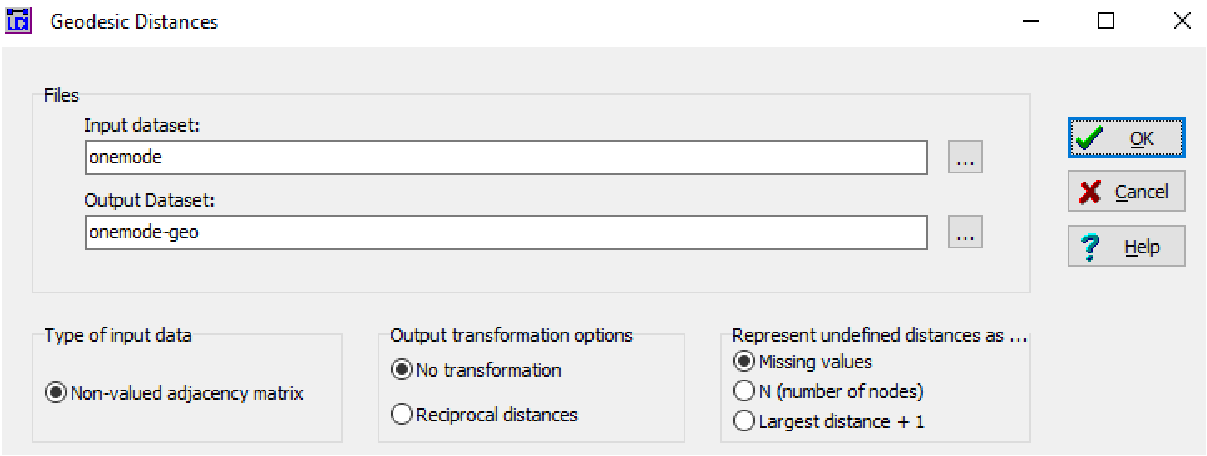
		1 Clus Coef	2 nPairs
1	A	0.000	1.000
2	B	0.333	3.000
3	C	1.000	1.000
4	D		0.000
5	E	0.000	3.000
6	F	0.333	3.000

Practical 4



Finding length of shortest path from all node to other nodes.

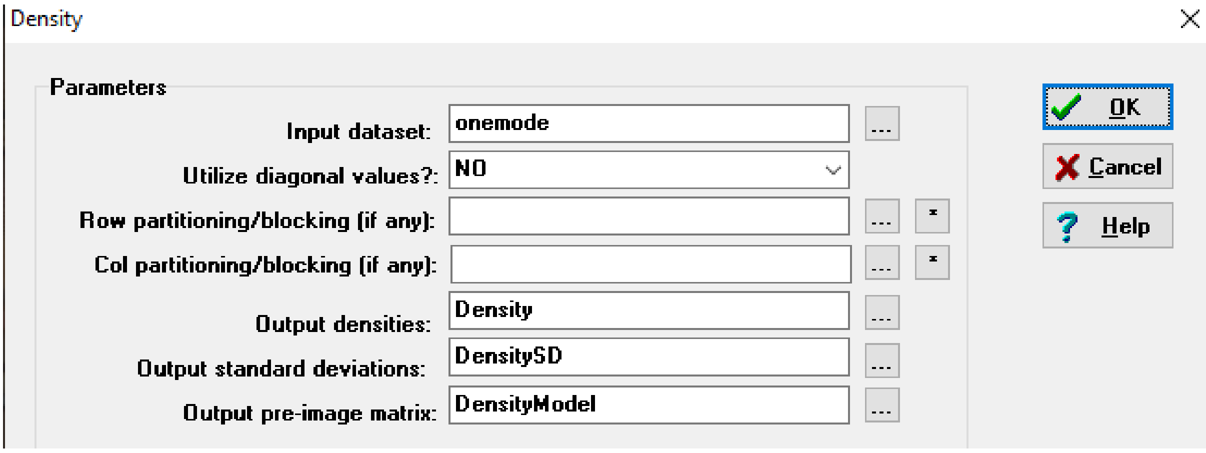
>Network>Cohesion>Geodesic distances



```
Frequencies

              1      2
           Freq   Prop
         ------ ------
    1  1     10  0.333
    2  2      5  0.167
    3  3      2  0.067
    4 NA     13  0.433

4 rows, 2 columns, 1 levels.



Average: :1.5
Std Dev: :0.7



        1 2 3 4 5 6
        A B C D E F
        - - - - - -
    1 A 0 1 2   1 2
    2 B   0 1     1
    3 C   1 0     1
    4 D   3 3 0 1 2
    5 E   2 2   0 1
    6 F   1 1     0

6 rows, 6 columns, 1 levels.
```

Finding density of the graph

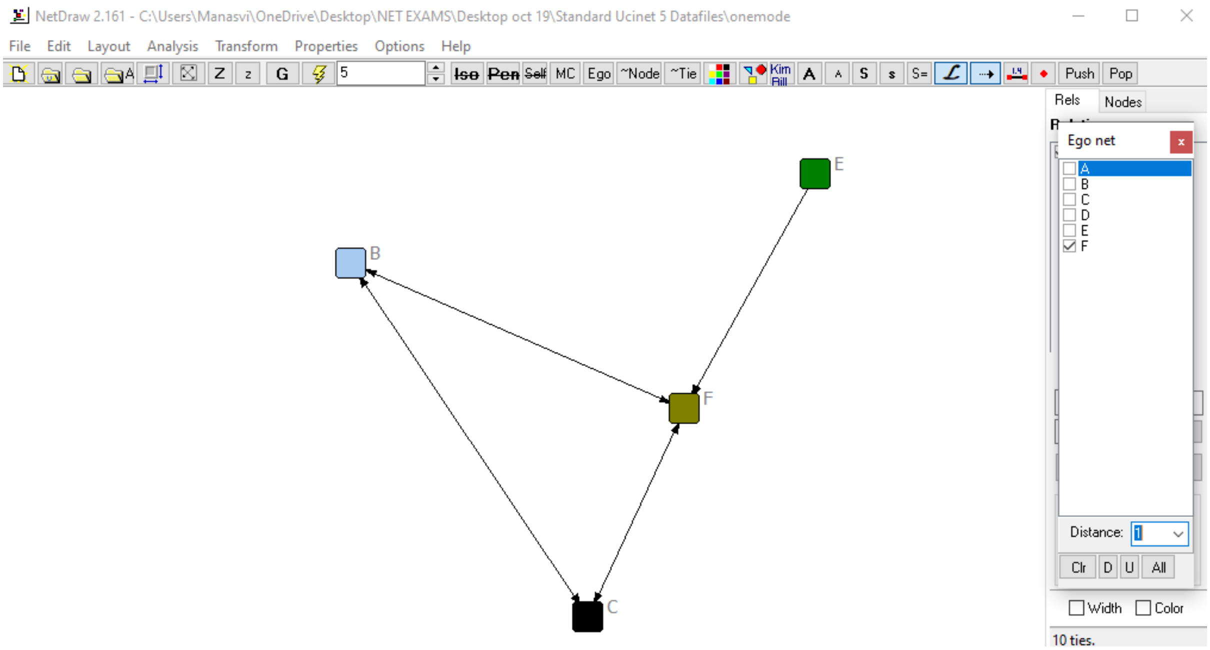
>Network>Cohesion>Density>Old Density Procedure.

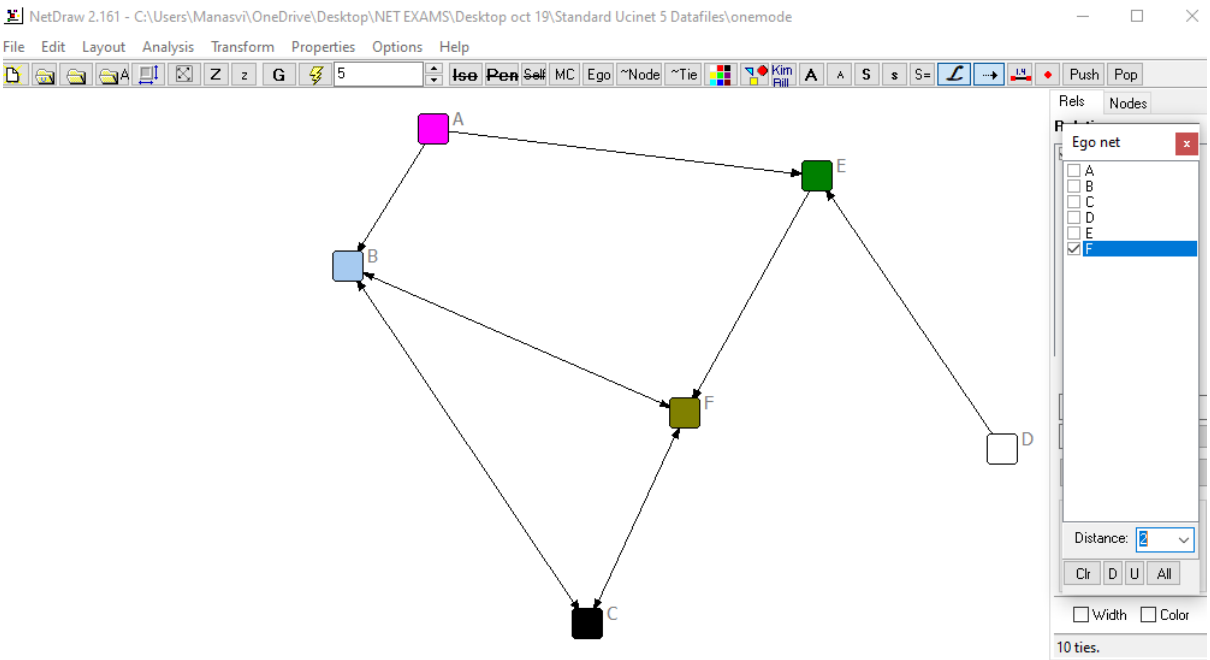



```
Density (matrix average) = 0.3333
Standard deviation = 0.4714
```

Draw egocentric network (NetDraw)

>Layout>Ego network(simple)



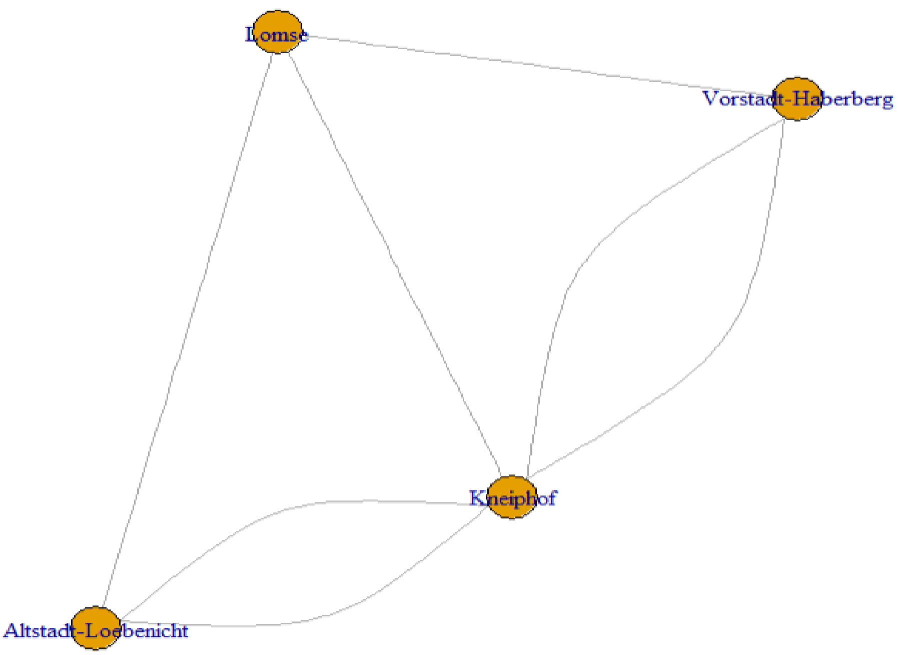


Practical 5

Representing graph using edge list (needs to be smaller to make sense)

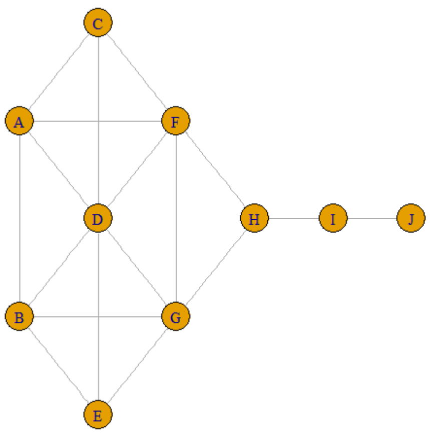
```
> library(igraph)
> library(igraphdata)
>
> #to get list of datasets in igraphdata
> data(package = "igraphdata")
> #load Koenigsberg datasets in environment
> data("Koenigsberg")
> plot(Koenigsberg)
> #representing Koenigsberg network as edgelist
> get.edgelist(Koenigsberg)
     [,1]                   [,2]
[1,] "Altstadt-Loebenicht" "Kneiphof"
[2,] "Altstadt-Loebenicht" "Kneiphof"
[3,] "Altstadt-Loebenicht" "Lomse"
[4,] "Kneiphof"            "Lomse"
[5,] "Vorstadt-Haberberg"  "Lomse"
[6,] "Kneiphof"            "Vorstadt-Haberberg"
[7,] "Kneiphof"            "Vorstadt-Haberberg"
```

```
Data sets in package 'igraphdata':

Koenigsberg             Bridges of Koenigsberg from Euler's times
UKfaculty               Friendship network of a UK university faculty
USairports              US airport network, 2010 December
enron                   Enron Email Network
foodwebs                A collection of food webs
immuno                  Immunoglobulin interaction network
karate                  Zachary's karate club network
kite                    Krackhardt's kite
macaque                 Visuotactile brain areas and connections
rfid                    Hospital encounter network data
yeast                   Yeast protein interaction network
```



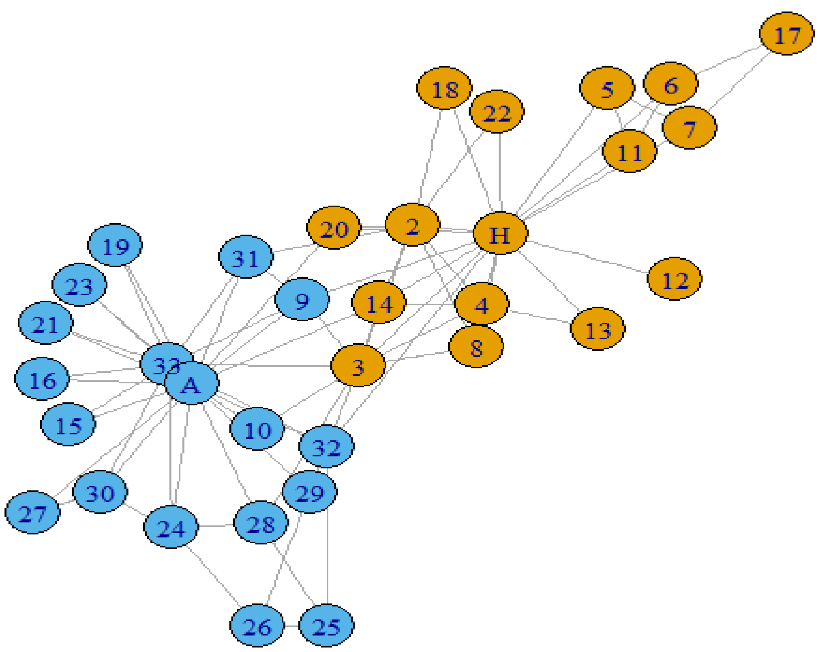
Representing graph as matrix

```
> #load kite datasets in environment
> data(kite)
> plot(kite)
> #using ajacency matrix to represent kite network as edgelist becomes longer
> kite[]
10 x 10 sparse Matrix of class "dgCMatrix"
   [[ suppressing 10 column names ‘A’, ‘B’, ‘C’ ... ]]

A . 1 1 1 . 1 . . . .
B 1 . . 1 1 . 1 . . .
C 1 . . 1 . 1 . . . .
D 1 1 1 . 1 1 1 . . .
E . 1 . 1 . . 1 . . .
F 1 . 1 1 . . 1 1 . .
G . 1 . 1 1 1 . 1 . .
H . . . . . 1 1 . 1 .
I . . . . . . . 1 . 1
J . . . . . . . . 1 .
```



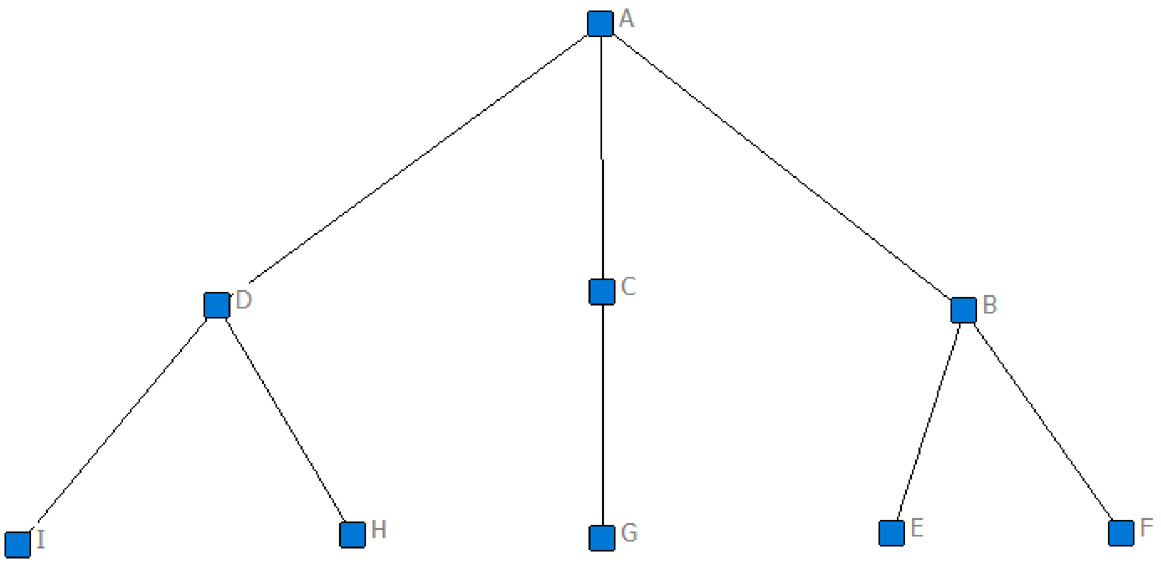
Representing as sociogram(graph)

```
> data("karate")
> #using only sociagram or graph to represnt the network as matrix and list are verylong
> plot(karate)
```



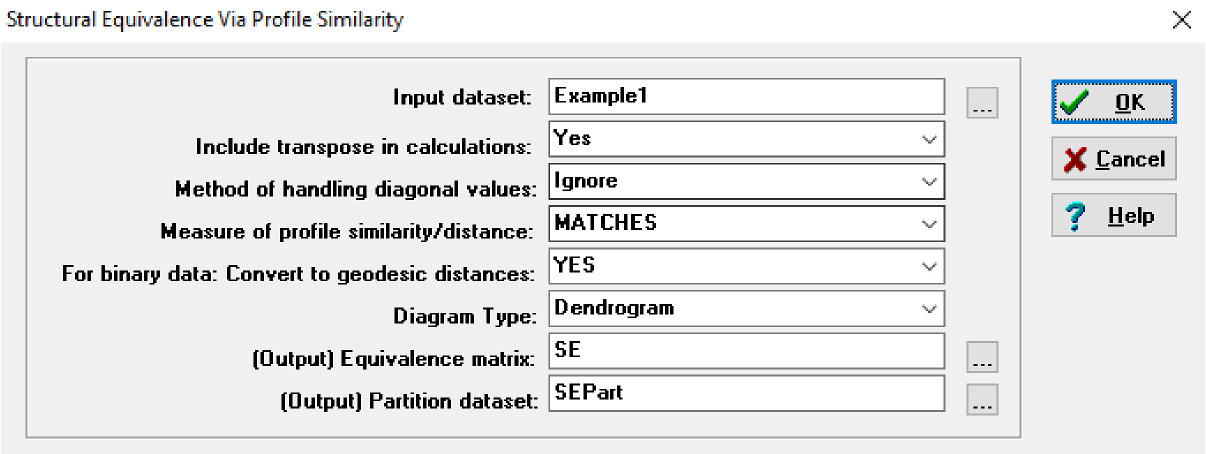
Practical 6

Structural Equivalence



>Network>Roles&Position>Structural>Profile

Use matches as they are good measure for binary relations.



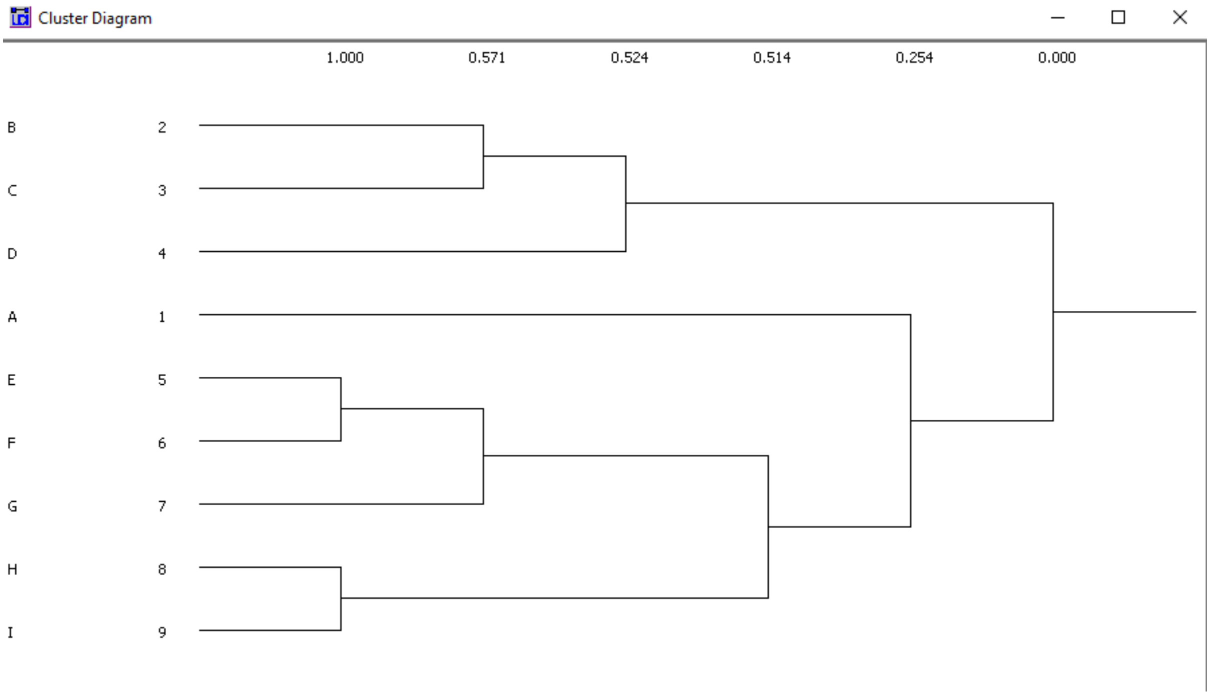
Measure:	Percent of Exact Matches
Include transpose	YES
Diagonal:	Ignore
Use geodesics?	YES
Input dataset:	Example1 (C:\Users\Manasvi

Structural Equivalence Matrix

	1 A	2 B	3 C	4 D	5 E	6 F	7 G	8 H	9 I
1 A	1.00	0.00	0.00	0.00	0.29	0.29	0.14	0.29	0.29
2 B	0.00	1.00	0.57	0.43	0.00	0.00	0.00	0.00	0.00
3 C	0.00	0.57	1.00	0.57	0.00	0.00	0.00	0.00	0.00
4 D	0.00	0.43	0.57	1.00	0.00	0.00	0.00	0.00	0.00
5 E	0.29	0.00	0.00	0.00	1.00	1.00	0.57	0.43	0.43
6 F	0.29	0.00	0.00	0.00	1.00	1.00	0.57	0.43	0.43
7 G	0.14	0.00	0.00	0.00	0.57	0.57	1.00	0.57	0.57
8 H	0.29	0.00	0.00	0.00	0.43	0.43	0.57	1.00	1.00
9 I	0.29	0.00	0.00	0.00	0.43	0.43	0.57	1.00	1.00

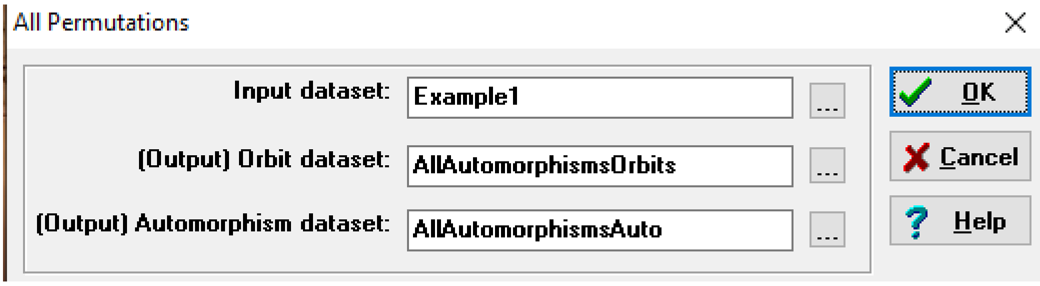
HIERARCHICAL CLUSTERING OF EQUIVALENCE MATRIX

```
        B C D A E F G H I

Level   2 3 4 1 5 6 7 8 9
-----   - - - - - - - - -
1.000   . . . . XXX . XXX
0.571   XXX . . XXXXX XXX
0.524   XXXXX . XXXXX XXX
0.514   XXXXX . XXXXXXXXX
0.254   XXXXX XXXXXXXXXXX
0.000   XXXXXXXXXXXXXXXXX
```



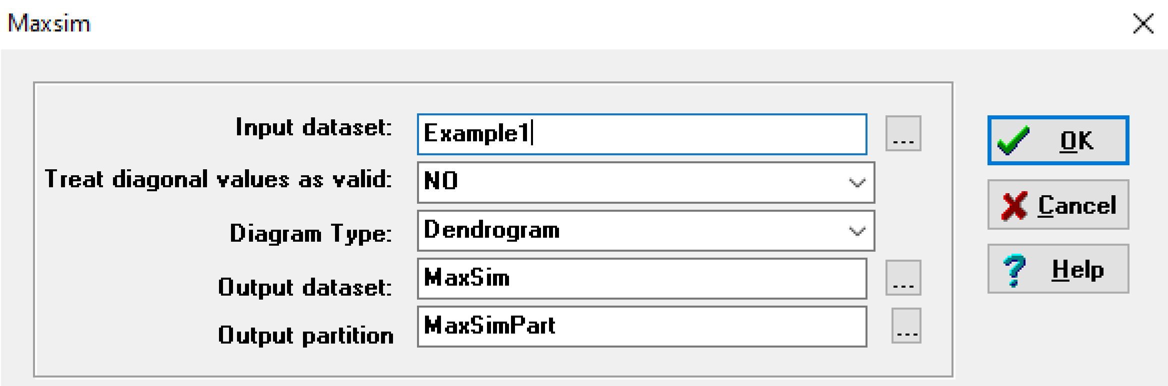
Automorphic equivalence

>Network>Roles&Position>Automorphic>All Permutation




```
AUTOMORPHIC EQUIVALENCE VIA DIRECT SEARCH
--------------------------------------------------------------


Number of permutations examined:   362880
Number of automorphisms found:          8
Hit rate:                            0.002205%

ORBITS:

Orbit #1:   A
Orbit #2:   B D
Orbit #3:   C
Orbit #4:   E F H I
Orbit #5:   G
```

>Network>Roles&Position>Automorphic>Maxsim

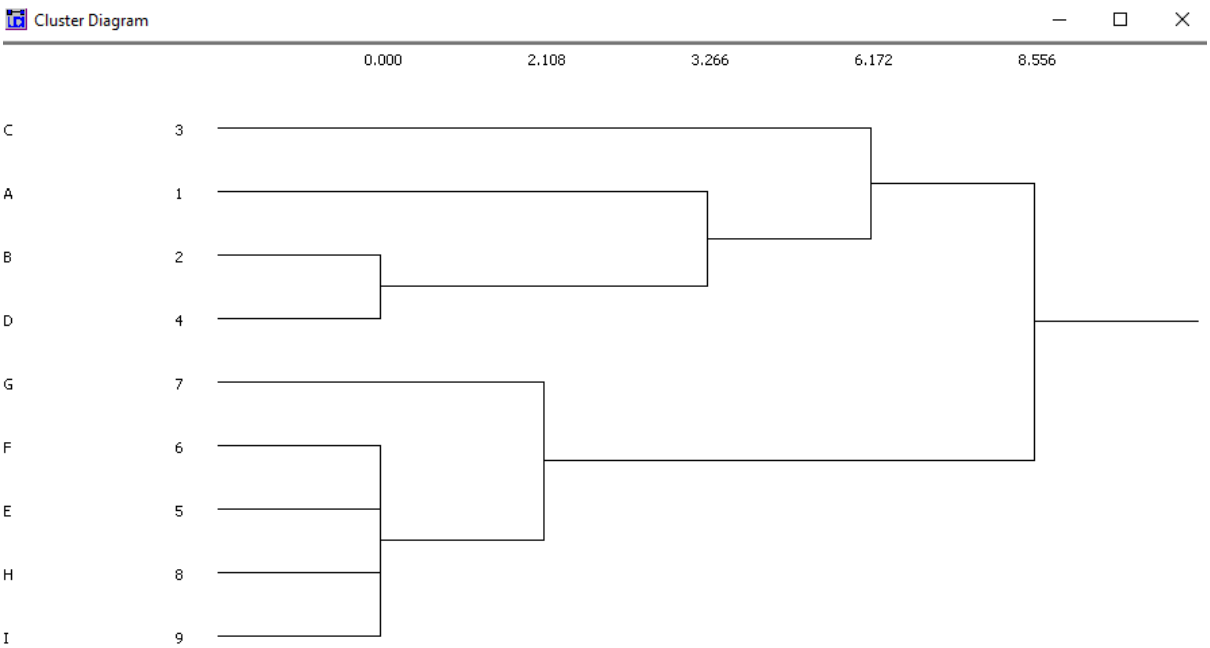


Distances Among Actors

		1 A	2 B	3 C	4 D	5 E	6 F	7 G	8 H	9 I
1	A	0.00	3.27	6.80	3.27	9.75	9.75	11.16	9.75	9.75
2	B	3.27	0.00	5.96	0.00	8.59	8.59	10.15	8.59	8.59
3	C	6.80	5.96	0.00	5.96	6.18	6.18	6.53	6.18	6.18
4	D	3.27	0.00	5.96	0.00	8.59	8.59	10.15	8.59	8.59
5	E	9.75	8.59	6.18	8.59	0.00	0.00	2.11	0.00	0.00
6	F	9.75	8.59	6.18	8.59	0.00	0.00	2.11	0.00	0.00
7	G	11.16	10.15	6.53	10.15	2.11	2.11	0.00	2.11	2.11
8	H	9.75	8.59	6.18	8.59	0.00	0.00	2.11	0.00	0.00
9	I	9.75	8.59	6.18	8.59	0.00	0.00	2.11	0.00	0.00

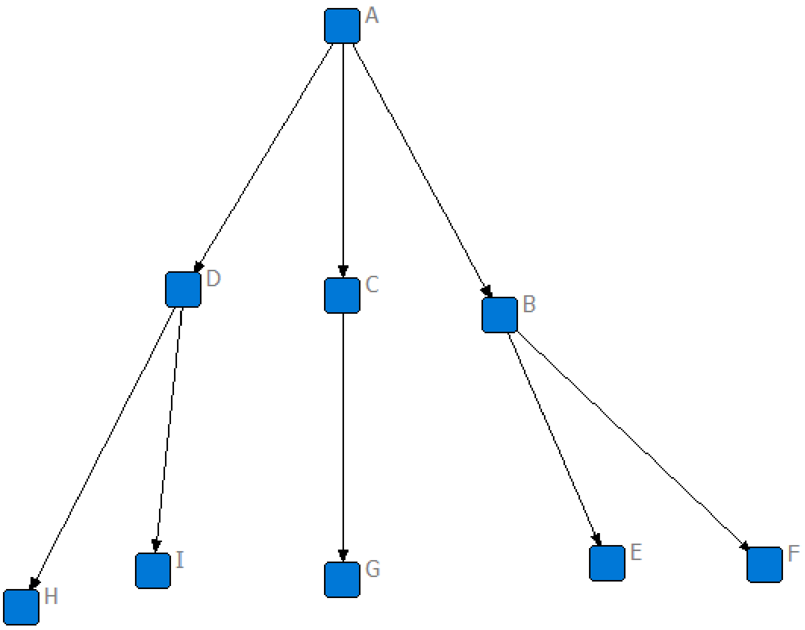
HIERARCHICAL CLUSTERING OF (NON-)EQUIVALENCE MATRIX

```
        C A B D G F E H I

Level   3 1 2 4 7 6 5 8 9
-----   - - - - - - - - -
0.000   . . XXX . XXXXXXX
2.108   . . XXX XXXXXXXXX
3.266   . XXXXX XXXXXXXXX
6.172   XXXXXXX XXXXXXXXX
8.556   XXXXXXXXXXXXXXXXX
```

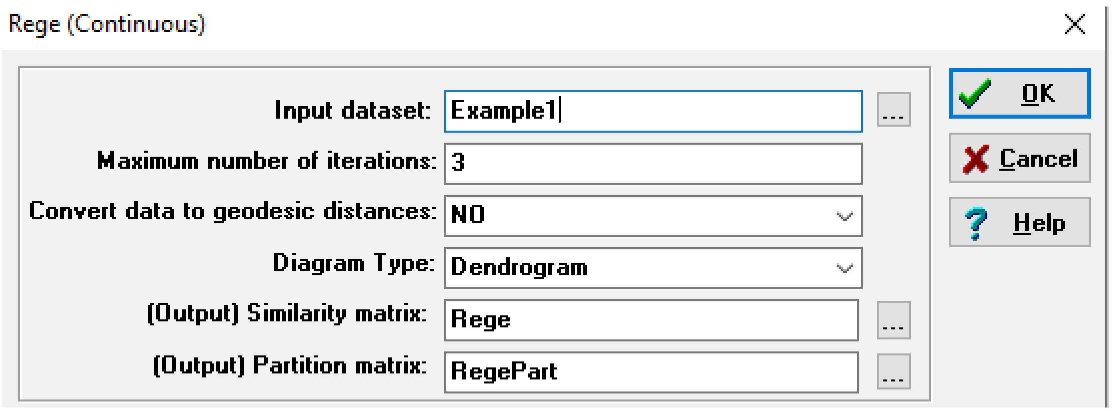


Regular Equivalence

Consider directed graph



>Network>Roles&Position>Maximal Regular>REGE…

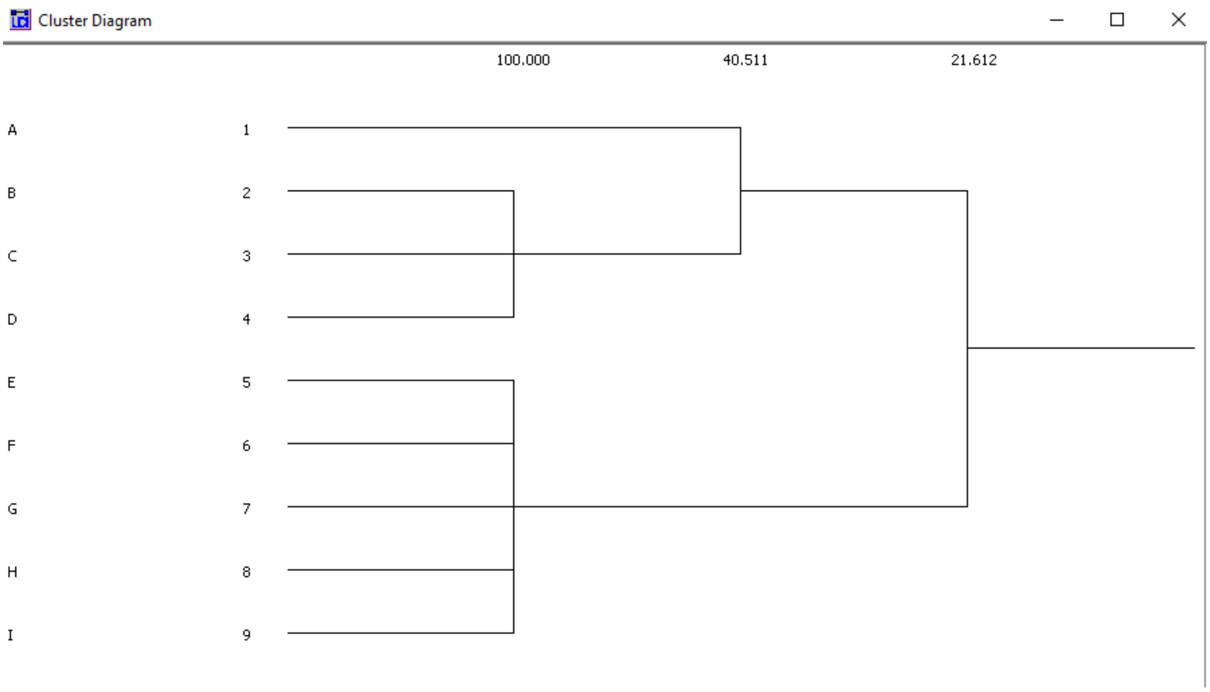


REGE similarities (3 iterations)

		1 A	2 B	3 C	4 D	5 E	6 F	7 G	8 H	9 I
1	A	100	42	37	42	0	0	0	0	0
2	B	42	100	100	100	25	25	23	25	25
3	C	37	100	100	100	33	33	31	33	33
4	D	42	100	100	100	25	25	23	25	25
5	E	0	25	33	25	100	100	100	100	100
6	F	0	25	33	25	100	100	100	100	100
7	G	0	23	31	23	100	100	100	100	100
8	H	0	25	33	25	100	100	100	100	100
9	I	0	25	33	25	100	100	100	100	100

HIERARCHICAL CLUSTERING OF EQUIVALENCE MATRIX

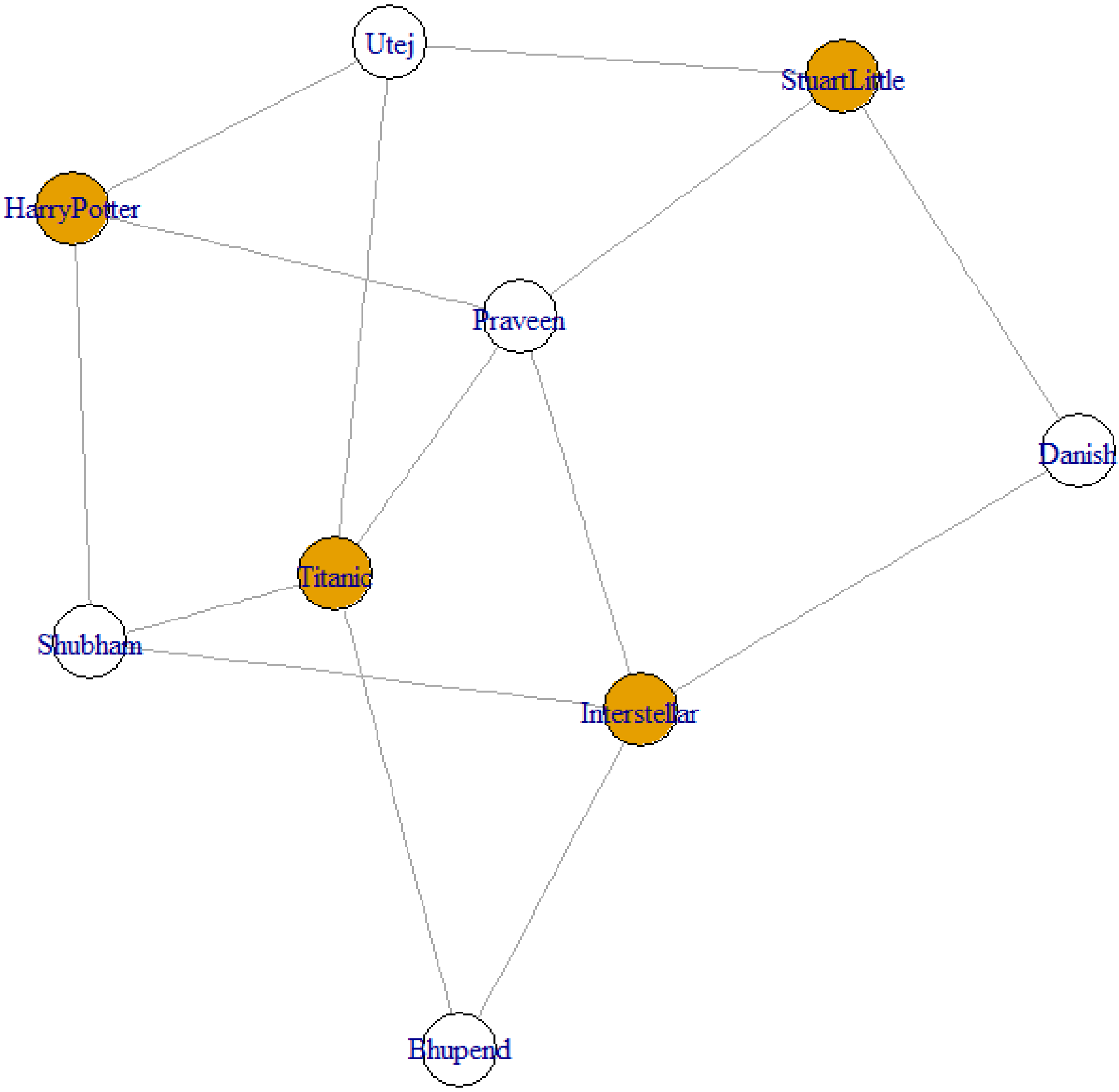
```
          A B C D E F G H I

  Level   1 2 3 4 5 6 7 8 9
-------   - - - - - - - - -
100.000   . XXXXX XXXXXXXXX
 40.511   XXXXXXX XXXXXXXXX
 21.612   XXXXXXXXXXXXXXXXX
```



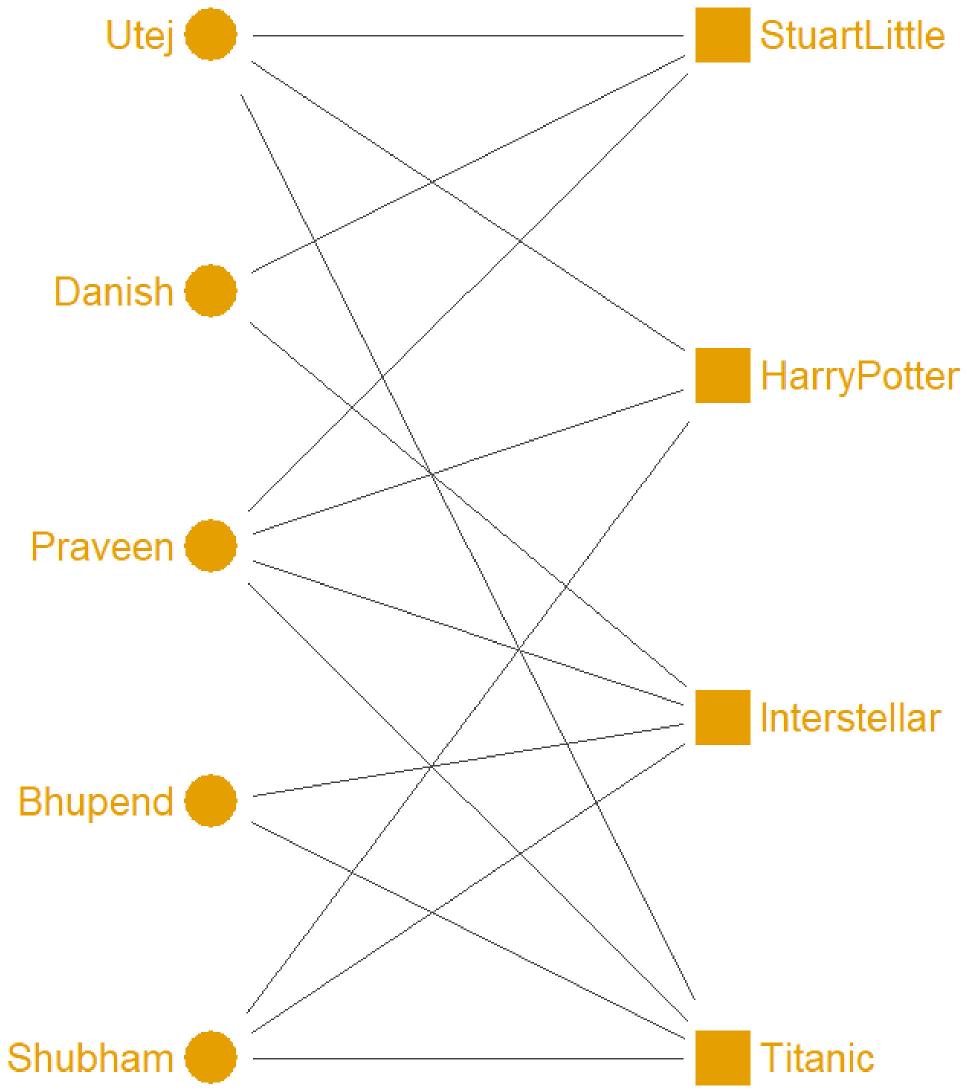
Practical 7

Understanding person to person and committee to committee using one mode and two mode networks.

```
> library(igraph)
> library(multigraph)
>
>
> #creating a two mode network
> affiliation_matrix=matrix(c(
+   1,1,0,1,
+   1,0,1,0,
+   1,1,1,1,
+   0,0,1,1,
+   0,1,1,1),
+   nrow = 5,
+   ncol = 4,
+   byrow = TRUE
+ )
>
> dimnames(affiliation_matrix)=list(
+   c("Utej","Danish","Praveen","Bhupend","Shubham"),
+   c("StuartLittle","HarryPotter","Interstellar","Titanic")
+ )
>
> affiliation_matrix
        StuartLittle HarryPotter Interstellar Titanic
Utej               1           1            0       1
Danish             1           0            1       0
Praveen            1           1            1       1
Bhupend            0           0            1       1
Shubham            0           1            1       1
>
>
> #visualizing two mode network
> two_mode_network=graph.incidence(affiliation_matrix)
> two_mode_network
IGRAPH UN-B 9 14 --
+ attr: type (v/l), name (v/c)
+ edges (vertex names):
 [1] Utej    --StuartLittle Utej    --HarryPotter  Utej    --Titanic      Danish --StuartLittle Danish --Interstellar
 [6] Praveen--StuartLittle Praveen--HarryPotter  Praveen--Interstellar Praveen--Titanic      Bhupend--Interstellar
[11] Bhupend--Titanic      Shubham--HarryPotter  Shubham--Interstellar Shubham--Titanic
>
> plot(two_mode_network,vertex.color=V(two_mode_network)$type)
```

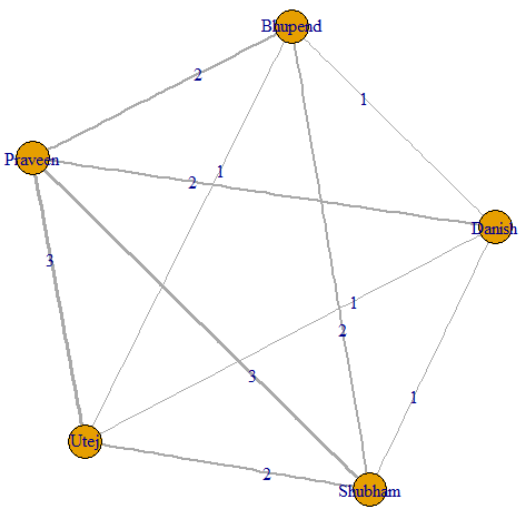


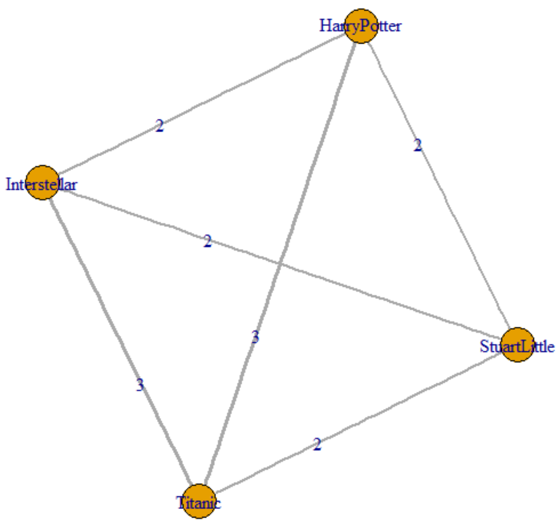

```
> #better way is to use bipartite graph
> bmgraph(affiliation_matrix,pch=16:15)
```




```
> #converting two mode network to one mode networks
> one_mode_networks=bipartite.projection(two_mode_network)
> one_mode_networks
$proj1
IGRAPH UNW- 5 10 --
+ attr: name (v/c), weight (e/n)
+ edges (vertex names):
 [1] Utej    --Danish  Utej    --Praveen Utej    --Shubham Utej    --Bhupend Danish --Praveen Danish --Bhupend Danish --Shubham
 [8] Praveen--Shubham Praveen--Bhupend Bhupend--Shubham

$proj2
IGRAPH UNW- 4 6 --
+ attr: name (v/c), weight (e/n)
+ edges (vertex names):
[1] StuartLittle--HarryPotter  StuartLittle--Titanic      StuartLittle--Interstellar HarryPotter --Titanic
[5] HarryPotter --Interstellar Interstellar--Titanic
```

```
> #visualizing matrix for each one mode network
> get.adjacency(one_mode_networks$proj1,sparse = FALSE,attr = "weight")
        Utej Danish Praveen Bhupend Shubham
Utej       0      1       3       1       2
Danish     1      0       2       1       1
Praveen    3      2       0       2       3
Bhupend    1      1       2       0       2
Shubham    2      1       3       2       0
> get.adjacency(one_mode_networks$proj2,sparse = FALSE,attr = "weight")
             StuartLittle HarryPotter Interstellar Titanic
StuartLittle            0           2            2       2
HarryPotter             2           0            2       3
Interstellar            2           2            0       3
Titanic                 2           3            3       0
>
> #visualizing sociogram for each one mode network
> plot(one_mode_networks$proj1,
+      edge.label=E(one_mode_networks$proj1)$weight,
+      edge.width=E(one_mode_networks$proj1)$weight
+      )
> plot(one_mode_networks$proj2,
+      edge.label=E(one_mode_networks$proj2)$weight,
+      edge.width=E(one_mode_networks$proj2)$weight
+     )
```

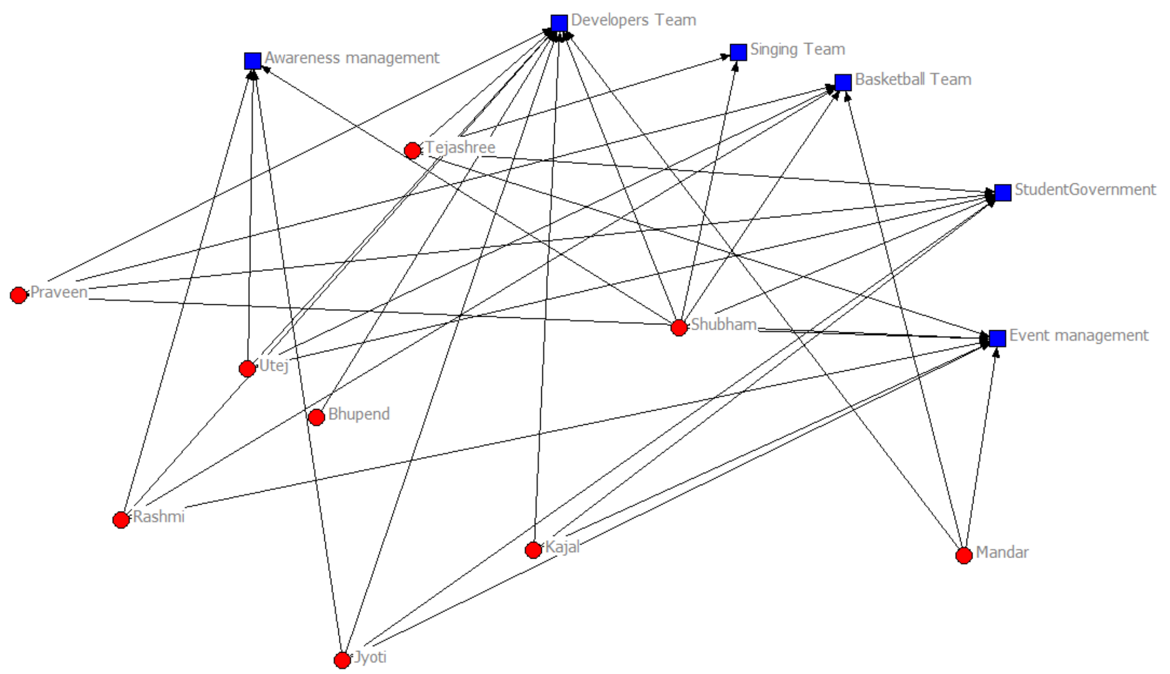




Practical 8

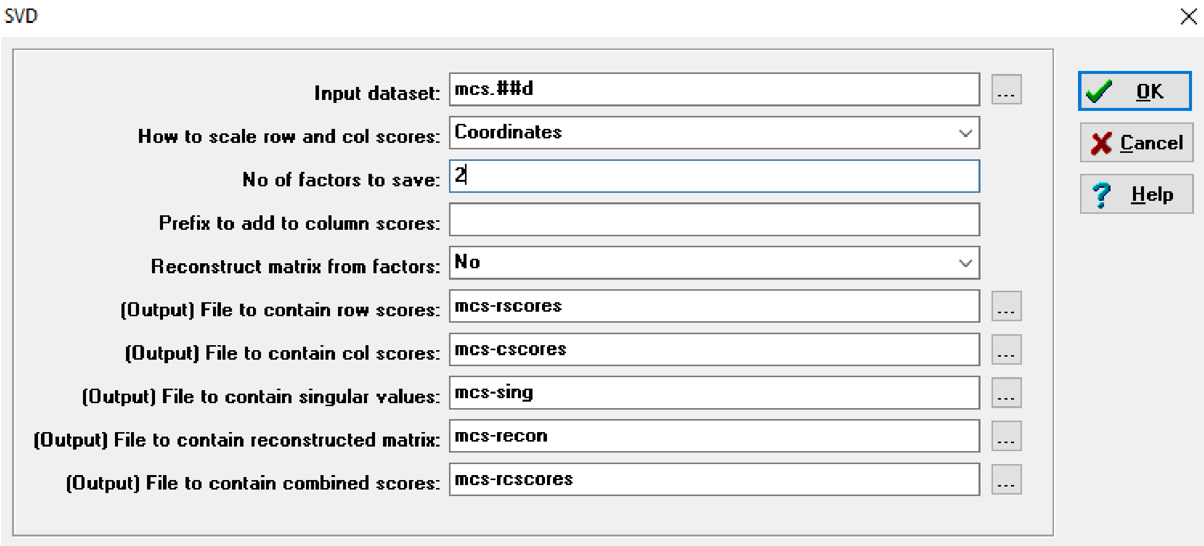
		1	2	3	4	5	6
		StudentGov	Basketball	Event	Awareness	Developers	Singing
1	Jyoti	3	0	3	4	3	0
2	Kajal	4	0	2	0	1	0
3	Rashmi	0	3	2	3	2	0
4	Praveen	3	3	2	0	2	0
5	Mandar	0	2	4	0	3	0
6	Utej	4	2	0	1	4	0
7	Tejashree	2	0	4	0	2	1
8	Bhupend	0	0	0	0	4	0
9	Shubham	3	1	3	2	3	5

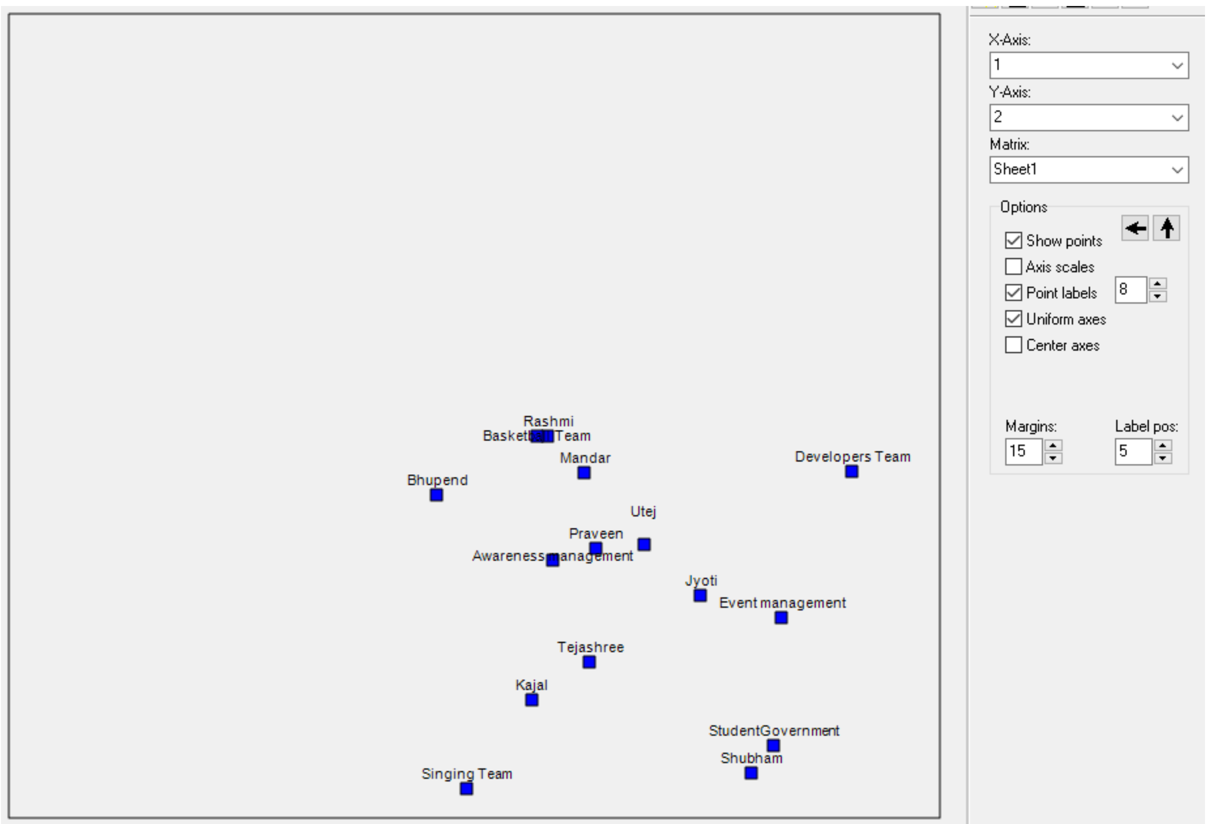
Visualize in netdraw but use 2 mode network.



SVD(Ucinet)

>Tools>Scaling/Decomposition>SVD..





Method: Principal Coordinates (vectors multiplied by singular values)

Matrix rank is 6

SINGULAR VALUES

FACTOR	VALUE	PERCENT	CUM %	RATIO	PRE	CUM PRE
1:	13.73812	39.4	39.4	2.778	0.298	0.298
2:	4.94522	14.2	53.5	1.022	0.188	0.486
3:	4.83689	13.9	67.4	1.162	0.180	0.666
4:	4.16170	11.9	79.3	1.088	0.133	0.799
5:	3.82400	11.0	90.3	1.129	0.112	0.912
6:	3.38685	9.7	100.0		0.088	1.000
	34.89277	100.0				

Row Scores

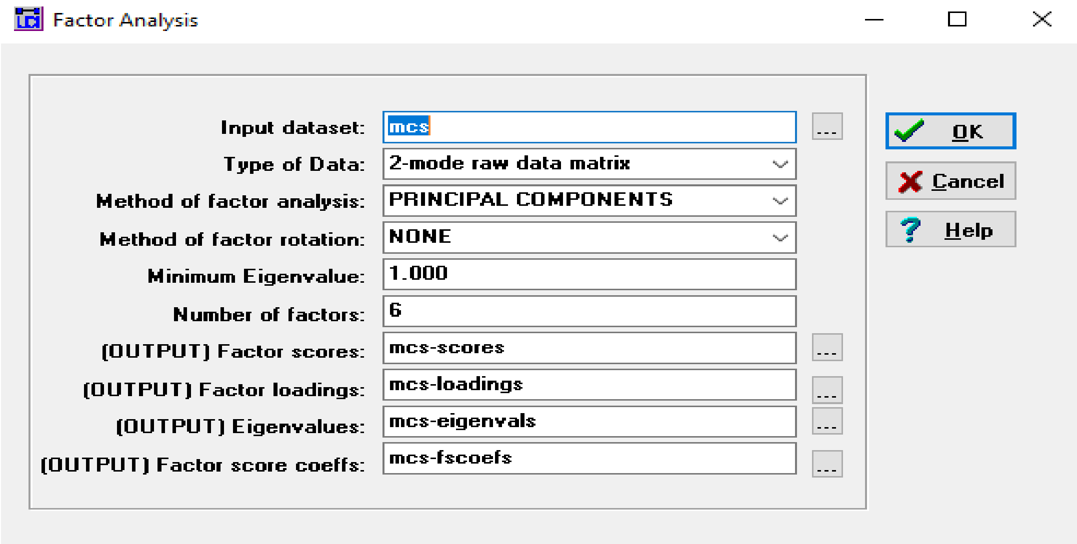
		1	2
1	Jyoti	5.79592	-0.00777
2	Kajal	3.53992	-1.62070
3	Rashmi	3.75807	2.46572
4	Praveen	4.40261	0.73457
5	Mandar	4.23080	1.88863
6	Utej	5.04744	0.79173
7	Tejashree	4.31762	-1.03796
8	Bhupend	2.27375	1.55425
9	Shubham	6.47440	-2.74268

Column Scores

		1	2
1	StudentGovernment	6.76974	-2.31323
2	Basketball Team	3.60405	2.47085
3	Event management	6.87182	-0.34165
4	Awareness management	3.81815	0.54041
5	Developers Team	7.80927	1.92153
6	Singing Team	2.67064	-2.98296

Two mode factor analysis

>Tools> Scaling/Decomposition>Factor Analysis…



EIGENVALUES

FACTOR	VALUE	PERCENT	CUM %	RATIO
1:	1.57246	26.2	26.2	1.230
2:	1.27822	21.3	47.5	1.109
3:	1.15293	19.2	66.7	1.221
4:	0.94407	15.7	82.5	1.248
5:	0.75671	12.6	95.1	2.560
6:	0.29560	4.9	100.0	
	6.00000	100.0		

Unrotated Factor Loadings

		1	2	3
1	StudentGovernment	0.369	0.382	-0.605
2	Basketball Team	-0.329	-0.394	0.450
3	Event management	0.778	-0.336	0.342
4	Awareness management	0.177	0.430	0.602
5	Developers Team	-0.612	0.630	0.221
6	Singing Team	0.564	0.532	0.236

Eigenvalues saved as dataset mcs-eigenvals (C:\Users\Manasvi\One[

Factor scores

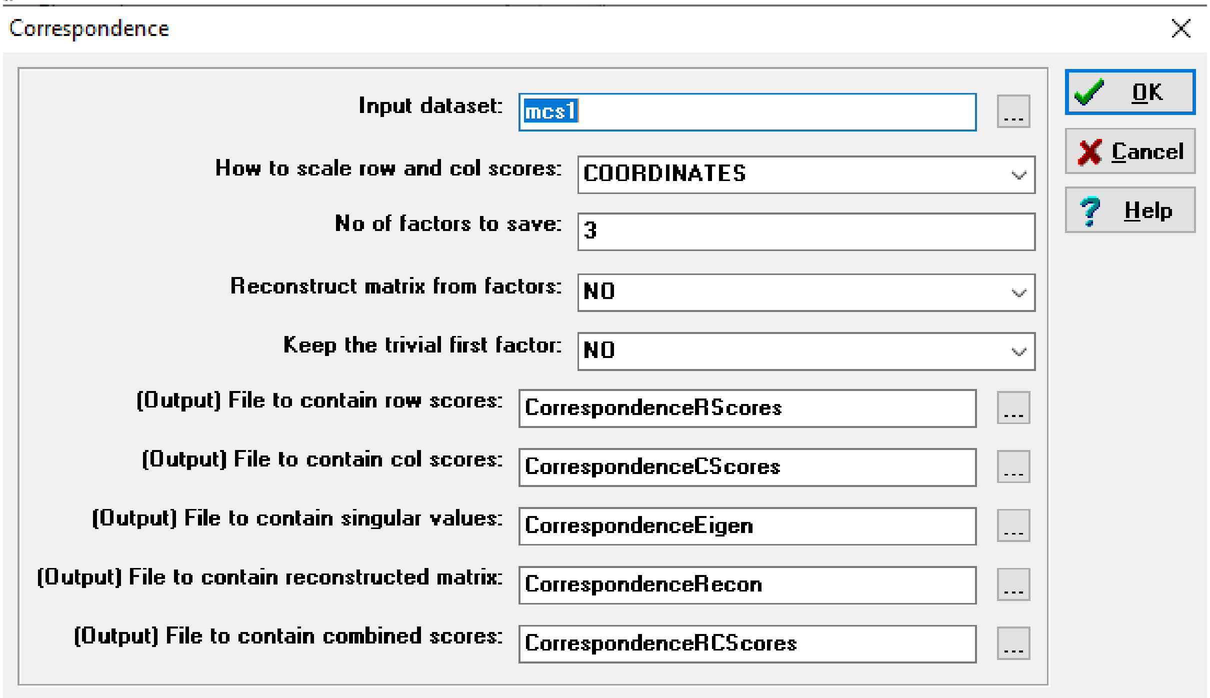
		1	2	3
1	Jyoti	0.549	0.995	0.505
2	Kajal	0.856	-0.603	-1.884
3	Rashmi	-0.422	-0.887	1.671
4	Praveen	-0.215	-1.022	-0.397
5	Mandar	-0.190	-1.187	0.900
6	Utej	-1.354	1.070	-0.703
7	Tejashree	1.087	-0.566	-0.467
8	Bhupend	-1.679	0.590	-0.384
9	Shubham	1.369	1.610	0.758

Two mode correspondence analysis

Best use is with binary values.

> Tools> Scaling/Decomposition>Correspondence..

		1	2	3	4	5	6
		StudentGov	Basketball	Event	Awareness	Developers	Singing
1	Jyoti	1	0	1	1	1	0
2	Kajal	1	0	1	0	1	0
3	Rashmi	0	1	1	1	1	0
4	Praveen	1	1	1	0	1	0
5	Mandar	0	1	1	0	1	0
6	Utej	1	1	0	1	1	0
7	Tejashree	1	0	1	0	1	1
8	Bhupend	0	0	0	0	1	0
9	Shubham	1	1	1	1	1	1



Minimum	Maximum	Sum	# of cells	density
0.000	1.000	33.000	54	0.611

Matrix rank is 5

SINGULAR VALUES

FACTOR	VALUE	PERCENT	CUM %	RATIO	PRE	CUM PRE
1:	0.44536	27.9	27.9	1.235	0.366	0.366
2:	0.36062	22.6	50.5	1.147	0.240	0.606
3:	0.31443	19.7	70.2	1.181	0.182	0.788
4:	0.26631	16.7	86.9	1.272	0.131	0.919
5:	0.20933	13.1	100.0		0.081	1.000
	1.59605	100.0				

Row Scores

		1	2	3
1	Jyoti	0.058	-0.136	-0.537
2	Kajal	-0.252	0.496	-0.371
3	Rashmi	0.545	-0.182	0.156
4	Praveen	0.126	0.300	0.133
5	Mandar	0.398	0.434	0.552
6	Utej	0.428	-0.352	-0.219
7	Tejashree	-0.934	-0.001	0.135
8	Bhupend	0.155	1.013	-0.363
9	Shubham	-0.248	-0.387	0.192

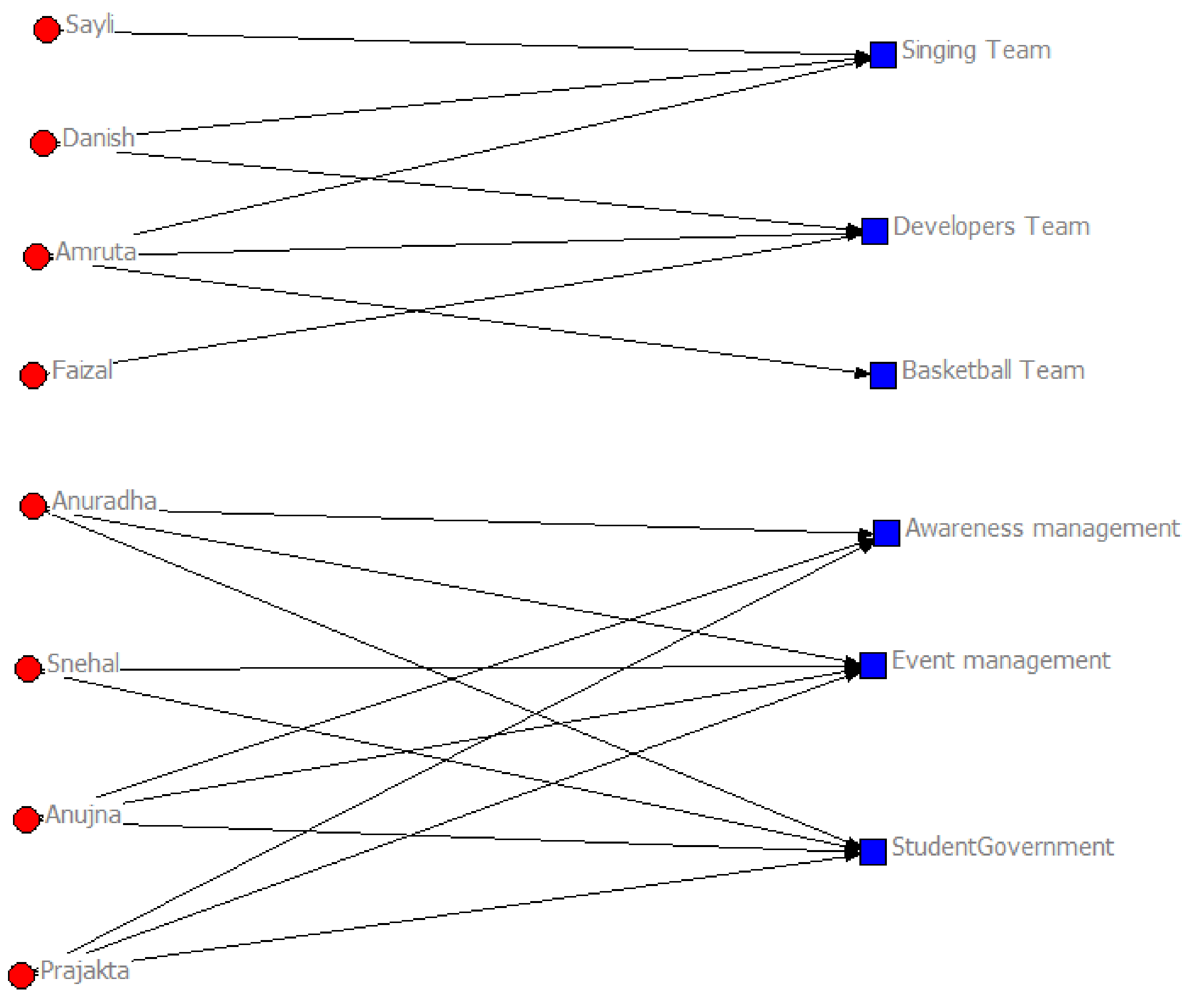
Column Scores

		1	2	3
1	StudentGovernment	-0.308	-0.037	-0.354
2	Basketball Team	0.561	-0.103	0.517
3	Event management	-0.098	0.208	0.118
4	Awareness management	0.440	-0.733	-0.325
5	Developers Team	0.069	0.365	-0.114
6	Singing Team	-1.327	-0.538	0.520

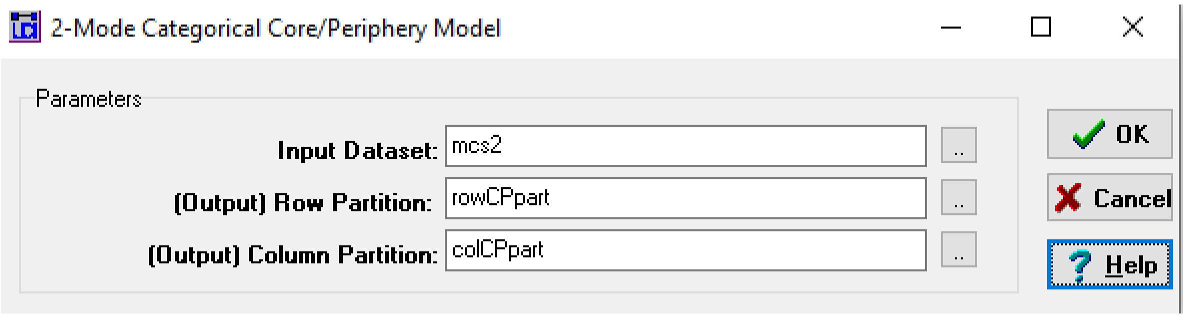
Practical 9

Two mode core-periphery analysis.

		1	2	3	4	5	6
		StudentGov	Basketball	Event	Awareness	Developers	Singing
1	Anujna	1	0	1	1	0	0
2	Amruta	0	1	0	0	1	1
3	Anuradha	1	0	1	1	0	0
4	Snehal	1	0	1	0	0	0
5	Prajakta	1	0	1	1	0	0
6	Danish	0	0	0	0	1	1
7	Faizal	0	0	0	0	1	0
8	Sayli	0	0	0	0	0	1



>Network>2-mode networks>Categorical Core/Periphery

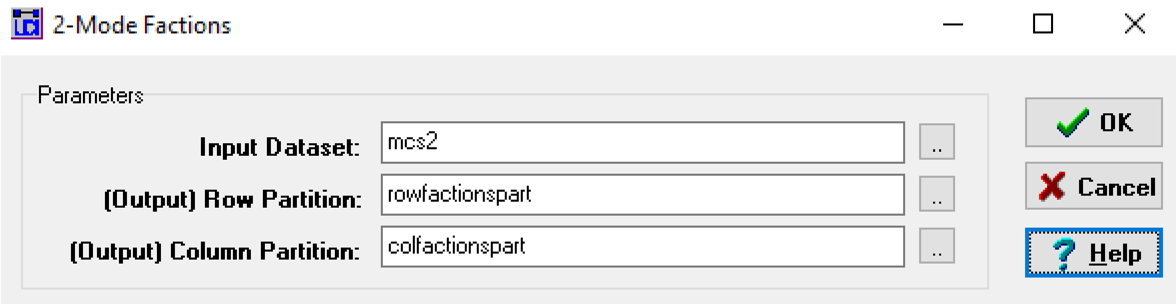



```
Starting fitness: 0.338
Initial partition

               1 5 3   4 2 6
               S D E   A B S
              --------------
  5 Prajakta | 1   1 | 1     |
  2   Amruta |   1   |   1 1 |
  3 Anuradha | 1   1 | 1     |
  4   Snehal | 1   1 |       |
             ----------------
  1   Anujna | 1   1 | 1     |
  6   Danish |   1   |     1 |
  7   Faizal |   1   |       |
  8    Sayli |       |     1 |
              ---------------


Final fitness: 0.548

Blocked Adjacency Matrix -- Final

               1 3 4   2 5 6
               S E A   B D S
              --------------
  1   Anujna | 1 1 1 |       |
  3 Anuradha | 1 1 1 |       |
  5 Prajakta | 1 1 1 |       |
             ----------------
  2   Amruta |       | 1 1 1 |
  4   Snehal | 1 1   |       |
  6   Danish |       |   1 1 |
  7   Faizal |       |   1   |
  8    Sayli |       |     1 |
              ---------------



Density matrix

            1     2
        ----- -----
    1   1.000 0.000
    2   0.143 0.385
```

Practical 10

Two mode faction analysis.

>Network>2-mode networks>2-mode factions




```
Starting fitness: 0.000
Final fitness: 0.775
Correlation to ideal: 0.775

Blocked Adjacency Matrix

               2 6 5   1 4 3
               B S D   S A E
              ---------------
   7   Faizal |     1 |       |
   2   Amruta | 1 1 1 |       |
   8    Sayli |   1   |       |
   6   Danish |   1 1 |       |
             -----------------
   1   Anujna |       | 1 1 1 |
   5 Prajakta |       | 1 1 1 |
   3 Anuradha |       | 1 1 1 |
   4   Snehal |       | 1   1 |
              ----------------



Density matrix

            1     2
        ----- -----
     1  0.583 0.000
     2  0.000 0.917
```