Seminar 6 and HW 4

Lesnykh Kirill

```
library(rmarkdown)
library(RColorBrewer)
library(NetData)
library(network)
```

```
## network: Classes for Relational Data
## Version 1.13.0.1 created on 2015-08-31.
## copyright (c) 2005, Carter T. Butts, University of California-Irvine
##                     Mark S. Handcock, University of California -- Los Angeles
##                     David R. Hunter, Penn State University
##                     Martina Morris, University of Washington
##                     Skye Bender-deMoll, University of Washington
##  For citation information, type citation("network").
##  Type help("network-package") to get started.
```

```
library(sna)
```

```
## Loading required package: statnet.common
```

```
## Warning: package 'statnet.common' was built under R version 3.5.2
```

```
##
## Attaching package: 'statnet.common'
```

```
## The following object is masked from 'package:base':
##
##     order
```

```
## sna: Tools for Social Network Analysis
## Version 2.4 created on 2016-07-23.
## copyright (c) 2005, Carter T. Butts, University of California-Irvine
##  For citation information, type citation("sna").
##  Type help(package="sna") to get started.
```

```
data(kracknets, package="NetData")

head(advice_data_frame)
```

```
##   ego alter advice_tie
## 1   1     1          0
## 2   1     2          1
## 3   1     3          0
## 4   1     4          1
```

```
## 5   1     5          0
## 6   1     6          0
```

```
head(friendship_data_frame)
```

```
##   ego alter friendship_tie
## 1   1     1              0
## 2   1     2              1
## 3   1     3              0
## 4   1     4              1
## 5   1     5              0
## 6   1     6              0
```

```
head(reports_to_data_frame)
```

```
##   ego alter reports_to_tie
## 1   1     1              0
## 2   1     2              1
## 3   1     3              0
## 4   1     4              0
## 5   1     5              0
## 6   1     6              0
```

```
krack <- list(advice_data_frame,
              friendship_data_frame,
              reports_to_data_frame)

krack <- list(advice_data_frame,
friendship_data_frame,
reports_to_data_frame)

graphs <- c('advice','friendship','reports')
names(krack) <- graphs

length(krack)
```

```
## [1] 3
```

```
names(krack)
```

```
## [1] "advice"     "friendship" "reports"
```

```
for (i in 1:length(krack)){
krack[[i]] <- as.matrix(krack[[i]])
}

for(i in 1:3){
krack[[i]] <- subset(krack[[i]],
```

```r
(krack[[i]][,3] > 0 ))
}
dim(krack[[1]])
```

```
## [1] 190   3
```

```r
head(krack[[1]])
```

```
##      ego alter advice_tie
## [1,]   1     2          1
## [2,]   1     4          1
## [3,]   1     8          1
## [4,]   1    16          1
## [5,]   1    18          1
## [6,]   1    21          1
```

```r
names(attributes)
```

```
## [1] "AGE"    "TENURE" "LEVEL"  "DEPT"
```

```r
for (i in 1:3){
krack[[i]] <- network(krack[[i]],
matrix.type = 'edgelist',
vertex.attr = list(attributes[,1], attributes[,2],
attributes[,3], attributes[,4]),
vertex.attrnames = list("AGE","TENURE","LEVEL","DEPT"))
}
advice <- krack$advice
friendship <- krack$friendship
reports <- krack$reports

print(advice)
```

```
##  Network attributes:
##   vertices = 21
##   directed = TRUE
##   hyper = FALSE
##   loops = FALSE
##   multiple = FALSE
##   bipartite = FALSE
##   total edges= 190
##     missing edges= 0
##     non-missing edges= 190
##
##  Vertex attribute names:
##     AGE DEPT LEVEL TENURE vertex.names
```

```
##
## No edge attributes
```

```
print(friendship)
```

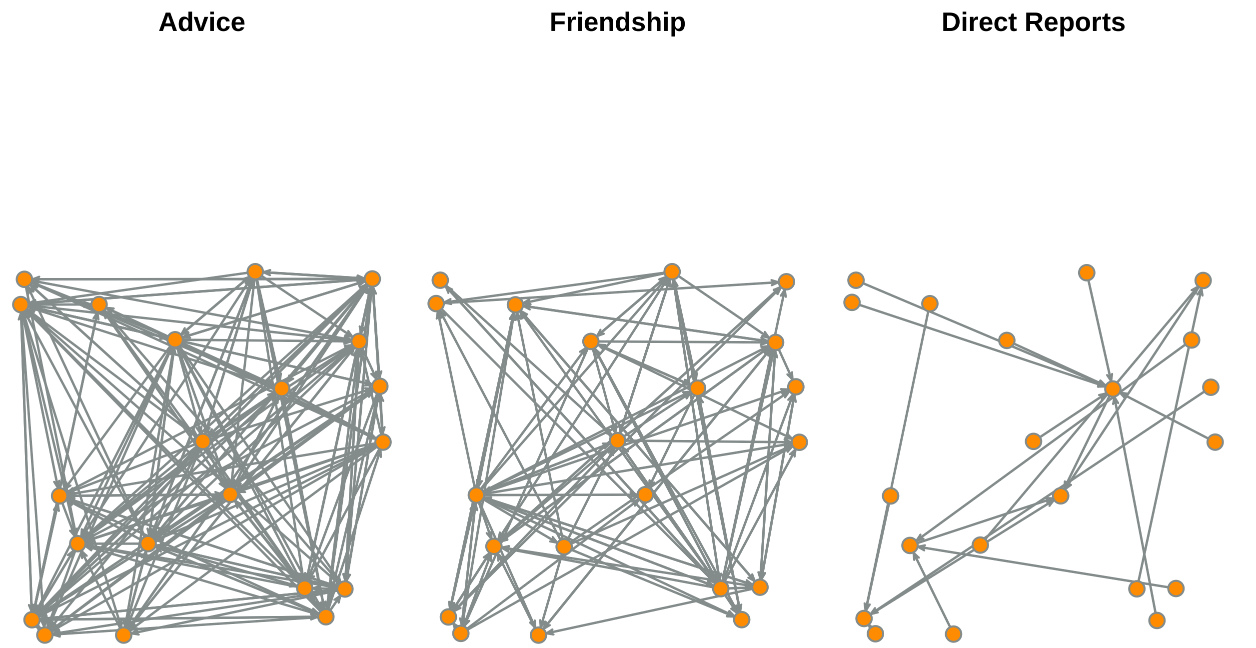
```
##  Network attributes:
##   vertices = 21
##   directed = TRUE
##   hyper = FALSE
##   loops = FALSE
##   multiple = FALSE
##   bipartite = FALSE
##   total edges= 102
##     missing edges= 0
##     non-missing edges= 102
##
##  Vertex attribute names:
##     AGE DEPT LEVEL TENURE vertex.names
##
## No edge attributes
```

```
print(reports)
```

```
##  Network attributes:
##   vertices = 21
##   directed = TRUE
##   hyper = FALSE
##   loops = FALSE
##   multiple = FALSE
##   bipartite = FALSE
##   total edges= 20
##     missing edges= 0
##     non-missing edges= 20
##
##  Vertex attribute names:
##     AGE DEPT LEVEL TENURE vertex.names
##
## No edge attributes
```

```
n<-network.size(advice)
v1<-sample((0:(n-1))/n)
v2<-sample(v1)
x <- n/(2 * pi) * sin(2 * pi * v1)
y <- n/(2 * pi) * cos(2 * pi * v2)
mycoord <- cbind(x,y)
```

```
par(mar=c(0,0,1,0))
par(mfrow=c(1,3))
plot(advice, edge.col='azure4', vertex.col='darkorange',
vertex.border='azure4',vertex.cex=2,coord=mycoord,
main ='Advice')
plot(friendship, edge.col='azure4', vertex.col='darkorange',
vertex.border='azure4',vertex.cex=2, coord=mycoord,
main ='Friendship')
plot(reports, edge.col='azure4', vertex.col='darkorange',
vertex.border='azure4',vertex.cex=2, coord=mycoord,
main='Direct Reports')
```



Assignment task. For the networks we???ve obtained, please calculate the following: 1. Dyad census 2. Different kinds of reciprocity 3. Triad census 4. Transitivity 5. Paths 6. Cycles 7. Cliques

```
dyad.census(advice)
```

```
##      Mut Asym Null
## [1,]  45  100   65
```

```
dyad.census(friendship)
```

```
##      Mut Asym Null
## [1,]  23   56  131
```

```
dyad.census(reports)
```

```
##      Mut Asym Null
```

```
## [1,]    0   20  190
```

```r
grecip(advice)
```

```
##       Mut
## 0.5238095
```

```r
grecip(friendship)
```

```
##       Mut
## 0.7333333
```

```r
grecip(reports)
```

```
##       Mut
## 0.9047619
```

```r
triad.census(advice)
```

```
##      003 012 102 021D 021U 021C 111D 111U 030T 030C 201 120D 120U 120C 210
## [1,]  74 153  90  160   86   49   59  101  190    2  72   62   78   17 107
##      300
## [1,]  30
```

```r
triad.census(friendship)
```

```
##      003 012 102 021D 021U 021C 111D 111U 030T 030C 201 120D 120U 120C 210
## [1,] 376 366 143  114   34   35   39  101   23    0  20   16   25    9  23
##      300
## [1,]   6
```

```r
triad.census(reports)
```

```
##       003 012 102 021D 021U 021C 111D 111U 030T 030C 201 120D 120U 120C
## [1,] 1003 274   0    0   37   16    0    0    0    0   0    0    0    0
##      210 300
## [1,]   0   0
```

```r
gtrans(advice)
```

```
## [1] 0.6639785
```

```r
gtrans(friendship)
```

```
## [1] 0.4610526
```

```r
gtrans(reports)
```

```
## [1] 0
```

```r
kpath.census(advice)
```

```
## $path.count
##     Agg    1    2    3    4    5   6    7    8    9   10   11  12  13   14
## 1   190   19   21   20   20   20  11   21   18   17   23   14   9  10   14
## 2  1488  218  223  232  244  212 108  246  223  163  302  132  84 109  147
## 3 10708 2133 1962 2231 2445 1960 952 2307 2197 1484 3018 1138 744 984 1362
##     15   16   17   18   19   20   21
## 1   24   12   14   32   15   20   26
## 2  274  129  147  495  182  236  358
## 3 2579 1217 1357 5152 1692 2309 3609
```

```r
kpath.census(friendship)
```

```
## $path.count
##    Agg   1   2   3   4   5   6  7   8   9  10  11  12  13  14  15  16   17
## 1  102  13  13   7  11  13   8  3   6   6   8  19  12   3   7  12   6   24
## 2  475  92  82  40  84 105  48 13  34  25  31 154  91  25  41  89  33  203
## 3 1968 501 423 206 480 598 249 60 183 100 189 847 560 134 208 508 167 1150
##    18  19  20  21
## 1   5  14   5   9
## 2  28 106  32  69
## 3 129 611 176 393
```

```r
kpath.census(reports)
```

```
## $path.count
##   Agg 1 2 3 4 5 6  7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
## 1  20 1 4 1 1 1 1  4 1 1  1  1  1  1  8  1  1  1  3  1  1  5
## 2  16 1 3 1 1 1 1 16 1 1  1  1  1  1  7  1  1  1  2  1  1  4
## 3   0 0 0 0 0 0 0  0 0 0  0  0  0  0  0  0  0  0  0  0  0  0
```

```r
kcycle.census(advice)
```

```
## $cycle.count
##   Agg  1  2  3  4  5 6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21
## 2  45  3  2  3  6  4 1  7  4  1  8  1  2  2  3  4  3  3 15  4  3 11
## 3 186 28 14 30 37 22 6 26 30 14 46  5  5  9 17 32 15 12 98 18 33 61
```

```r
kcycle.census(friendship)
```

```
## $cycle.count
##   Agg 1 2 3  4  5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
## 2  23 4 3 1  4  3 1 0 1 0  0  5  4  1  1  3  1  6  1  4  0  3
## 3  44 8 4 3 10 12 3 0 2 0  2 15 11  1  3 11  2 24  1 12  2  6
```

```r
kcycle.census(reports)
```

```
## $cycle.count
##   Agg 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
## 2   0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0
## 3   0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0
```

```r
clique.census(advice)
```

```
## $clique.count
##   Agg 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
## 1   0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0
## 2   3 0 0 0 0 0 1 1 0 1  0  1  0  0  0  0  0  0  1  0  0  1
## 3  12 2 1 2 2 1 0 3 0 0  2  0  1  1  0  1  2  2  8  0  2  6
## 4   5 0 0 0 2 1 0 1 2 0  3  0  0  0  1  1  0  0  5  2  0  2
##
## $cliques
## $cliques[[1]]
## NULL
##
## $cliques[[2]]
## $cliques[[2]][[1]]
## [1]  9 18
##
## $cliques[[2]][[2]]
## [1]  6 21
##
## $cliques[[2]][[3]]
## [1]  7 11
##
##
## $cliques[[3]]
## $cliques[[3]][[1]]
## [1]  4 17 21
##
## $cliques[[3]][[2]]
## [1]  1 16 18
##
## $cliques[[3]][[3]]
## [1]  5 13 18
##
## $cliques[[3]][[4]]
## [1]  3 18 21
##
## $cliques[[3]][[5]]
```

```
## [1] 10 16 18
##
## $cliques[[3]][[6]]
## [1] 18 20 21
##
## $cliques[[3]][[7]]
## [1] 15 18 20
##
## $cliques[[3]][[8]]
## [1]  3 10 18
##
## $cliques[[3]][[9]]
## [1]  7 17 21
##
## $cliques[[3]][[10]]
## [1]  7 12 21
##
## $cliques[[3]][[11]]
## [1]  2  7 21
##
## $cliques[[3]][[12]]
## [1]  1  4 18
##
##
## $cliques[[4]]
## $cliques[[4]][[1]]
## [1]  4  8 18 21
##
## $cliques[[4]][[2]]
## [1]  4  8 10 18
##
## $cliques[[4]][[3]]
## [1]  5 10 18 19
##
## $cliques[[4]][[4]]
## [1] 10 15 18 19
##
## $cliques[[4]][[5]]
## [1]  7 14 18 21
```

```
clique.census(friendship)
```

```
## $clique.count
##   Agg 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
## 1   4 0 0 0 0 0 0 1 0 1  1  0  0  0  0  0  0  0  0  0  1  0
```

```
## 2    9 2 3 1 1 0 1 0 1 0  0  1  0  1  1  1  1  1  1  1  0  1
## 3    6 1 0 0 2 2 0 0 0 0  0  3  3  0  0  1  0  3  0  2  0  1
##
## $cliques
## $cliques[[1]]
## $cliques[[1]][[1]]
## [1] 20
##
## $cliques[[1]][[2]]
## [1] 10
##
## $cliques[[1]][[3]]
## [1] 9
##
## $cliques[[1]][[4]]
## [1] 7
##
##
## $cliques[[2]]
## $cliques[[2]][[1]]
## [1]  2 21
##
## $cliques[[2]][[2]]
## [1]  2 18
##
## $cliques[[2]][[3]]
## [1]  1 16
##
## $cliques[[2]][[4]]
## [1] 14 15
##
## $cliques[[2]][[5]]
## [1]  6 17
##
## $cliques[[2]][[6]]
## [1] 11 13
##
## $cliques[[2]][[7]]
## [1] 4 8
##
## $cliques[[2]][[8]]
## [1]  3 19
##
## $cliques[[2]][[9]]
## [1] 1 2
```

```
##
##
## $cliques[[3]]
## $cliques[[3]][[1]]
## [1]  5 11 19
##
## $cliques[[3]][[2]]
## [1] 11 15 19
##
## $cliques[[3]][[3]]
## [1]  5 11 17
##
## $cliques[[3]][[4]]
## [1] 12 17 21
##
## $cliques[[3]][[5]]
## [1]  4 12 17
##
## $cliques[[3]][[6]]
## [1]  1  4 12
```

```
clique.census(reports)
```

```
## $clique.count
##   Agg 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
## 1  21 1 1 1 1 1 1 1 1 1  1  1  1  1  1  1  1  1  1  1  1  1
##
## $cliques
## $cliques[[1]]
## $cliques[[1]][[1]]
## [1] 21
##
## $cliques[[1]][[2]]
## [1] 20
##
## $cliques[[1]][[3]]
## [1] 19
##
## $cliques[[1]][[4]]
## [1] 18
##
## $cliques[[1]][[5]]
## [1] 17
##
## $cliques[[1]][[6]]
```

```
## [1] 16
##
## $cliques[[1]][[7]]
## [1] 15
##
## $cliques[[1]][[8]]
## [1] 14
##
## $cliques[[1]][[9]]
## [1] 13
##
## $cliques[[1]][[10]]
## [1] 12
##
## $cliques[[1]][[11]]
## [1] 11
##
## $cliques[[1]][[12]]
## [1] 10
##
## $cliques[[1]][[13]]
## [1] 9
##
## $cliques[[1]][[14]]
## [1] 8
##
## $cliques[[1]][[15]]
## [1] 7
##
## $cliques[[1]][[16]]
## [1] 6
##
## $cliques[[1]][[17]]
## [1] 5
##
## $cliques[[1]][[18]]
## [1] 4
##
## $cliques[[1]][[19]]
## [1] 3
##
## $cliques[[1]][[20]]
## [1] 2
##
## $cliques[[1]][[21]]
```

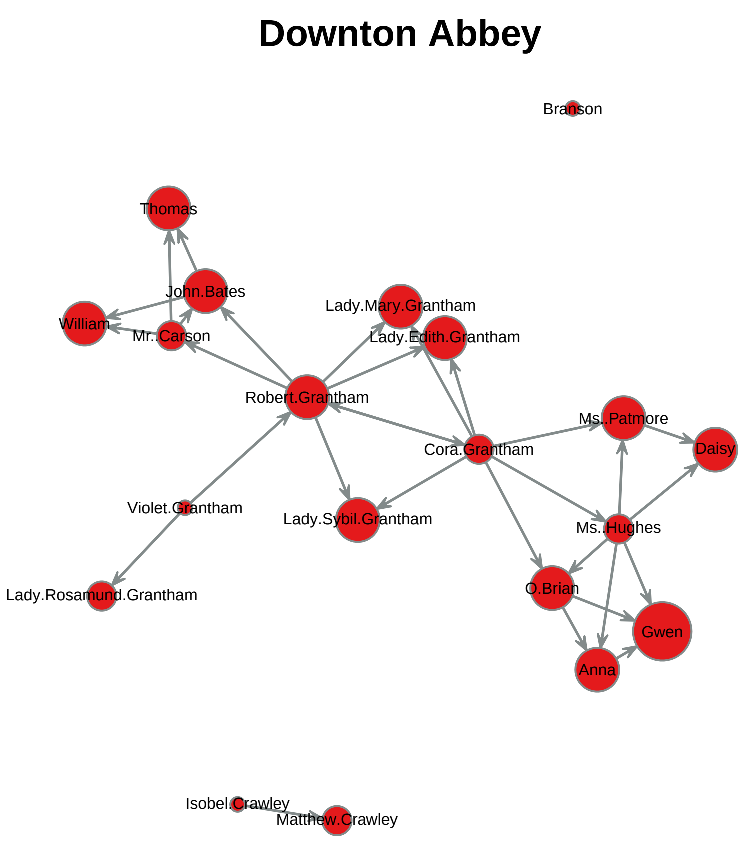
[1] 1

Assignment 1.

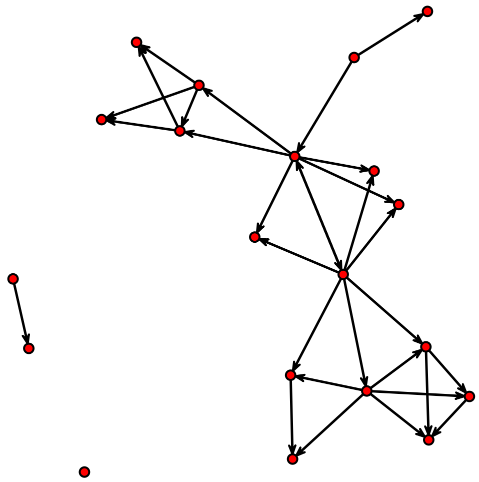
Based on those measures we can make the following statements. The “Reports” network, comparing to the other ones has a much less ties and connections, so its transitivity, its connections among nodes is much weaker, than of the “Friendship” and of the “Advice”. Moreover, the “Reports” has only directed ties, so, its paths and are also very low. In the analogical way, the connections of “Friendship” a less weaker than are of the “Advice”.

From all these three networks we can tell, that a much more common and popular thing is going for the advice, that the friendship and then the reporting. That is the range of oftennes that happens in that sampling working collective.

```
formal<-as.matrix(read.csv("formal.csv", header = TRUE, row.names=1))
roles<-read.csv("roles.csv", header=TRUE, row.names=1)

formalnet <- network(formal)
par(mar=c(0,0,2,0))
indeg <- degree(formalnet, cmode = 'indegree')
mycoord <- plot(formalnet, displaylabels=TRUE, edge.col='azure4',
                vertex.col="#E41A1C", vertex.border='azure4',
                vertex.cex = indeg + 1 , main ='Downton Abbey',
                label.cex=0.5, label.pos = 5)
```



```
plot(formalnet)
```



```
orRule <- symmetrize(formalnet, rule='weak')
class(orRule)
```

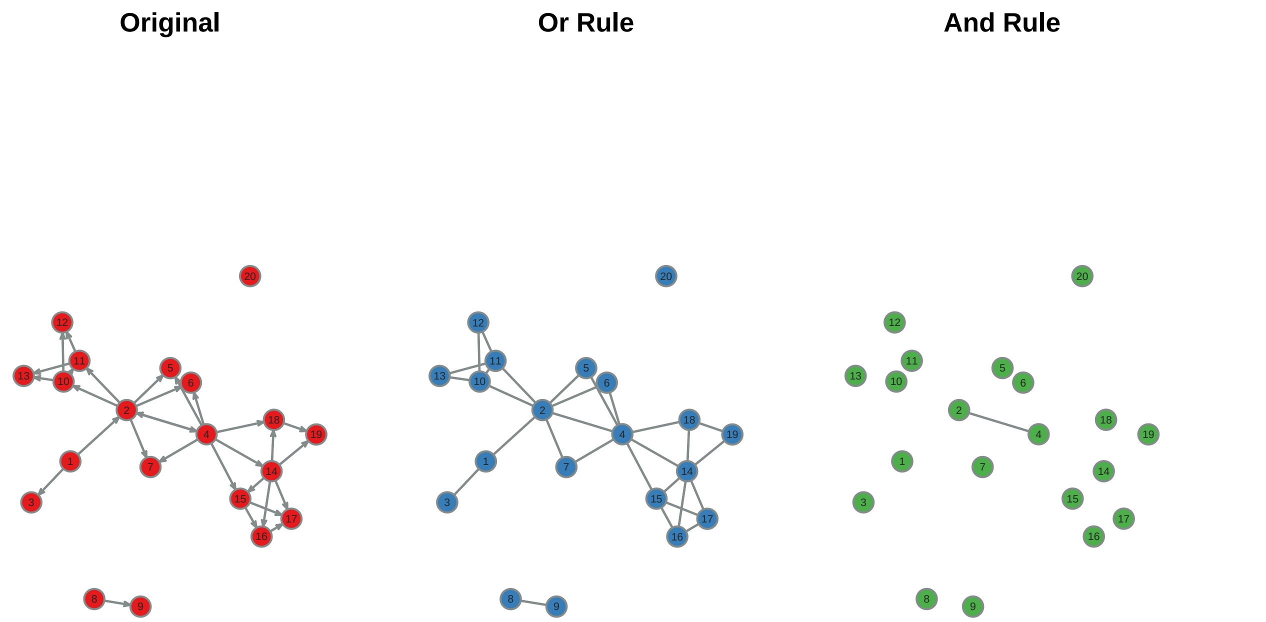
```
## [1] "matrix"
```

```
orRule <- network(symmetrize(formalnet, rule='weak'),
directed = FALSE)
class(orRule)
```

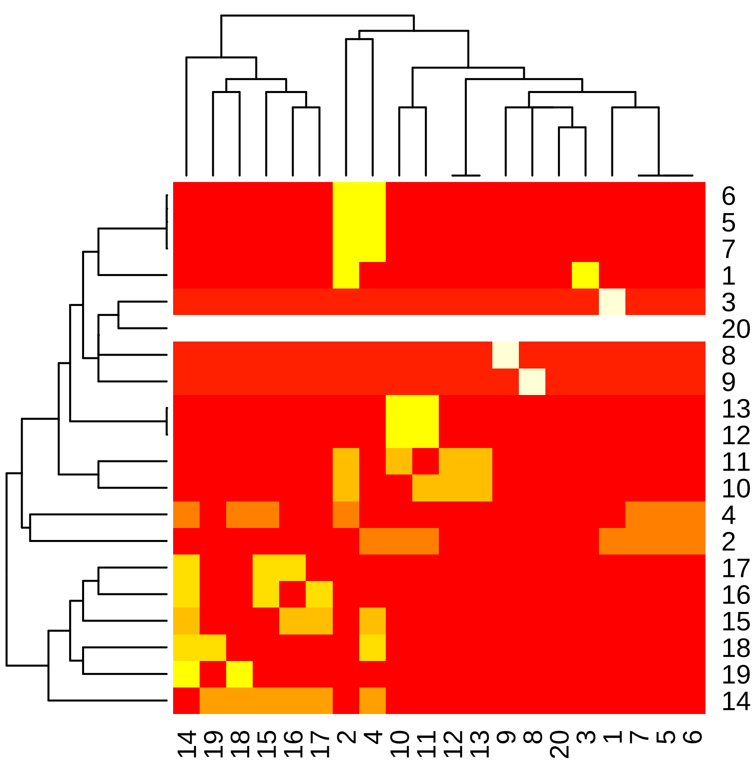
```
## [1] "network"
```

```
andRule <- network(symmetrize(formalnet, rule='strong'),
directed = FALSE)
```

```
par(mar=c(1,1,2,1))
par(mfrow=c(1,3))
plot(formalnet, main = 'Original', coord=mycoord, vertex.cex =3,
     edge.col='azure4', vertex.col="#E41A1C", vertex.border='azure4',
     label=seq(1:20),label.pos=5,label.cex=.5,label.col='gray15')
plot(orRule, main = 'Or Rule', coord=mycoord, vertex.cex =3,
     edge.col='azure4', vertex.col="#377EB8", vertex.border='azure4',
     label=seq(1:20),label.pos=5,label.cex=.5,label.col='gray15')
plot(andRule, main = 'And Rule', coord=mycoord, vertex.cex =3,
     edge.col='azure4', vertex.col="#4DAF4A", vertex.border='azure4',
     label=seq(1:20),label.pos=5,label.cex=.5,label.col='gray15')
```




```
snasymmformal <- orRule
aprioriformal<-blockmodel(snasymmformal, roles$commdetect,
                          block.content="density", mode="graph",
                          diag=FALSE)

heatmap(aprioriformal[[4]])
```



```
roles$commdetect
```

```
##  [1] 1 1 2 2 2 2 2 3 3 4 4 4 4 5 5 5 5 5 5 5
```

```
apriorformal[[1]]
```

```
##  [1] 1 1 2 2 2 2 2 3 3 4 4 4 4 5 5 5 5 5 5 5
```

```
apriorformal[[2]]
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

```
apriorformal[[3]]
```

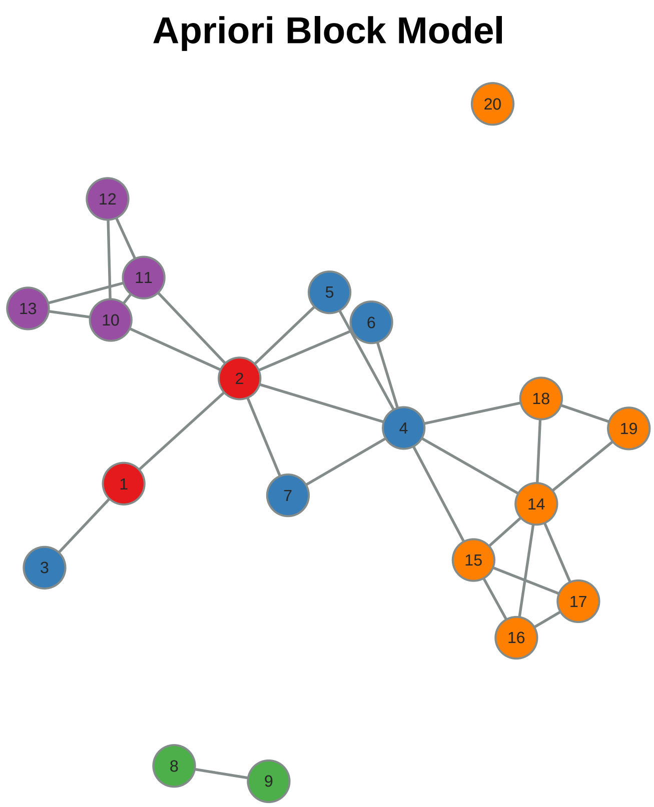
```
## [1] "density"
```

```
apriorformal[[4]]
```

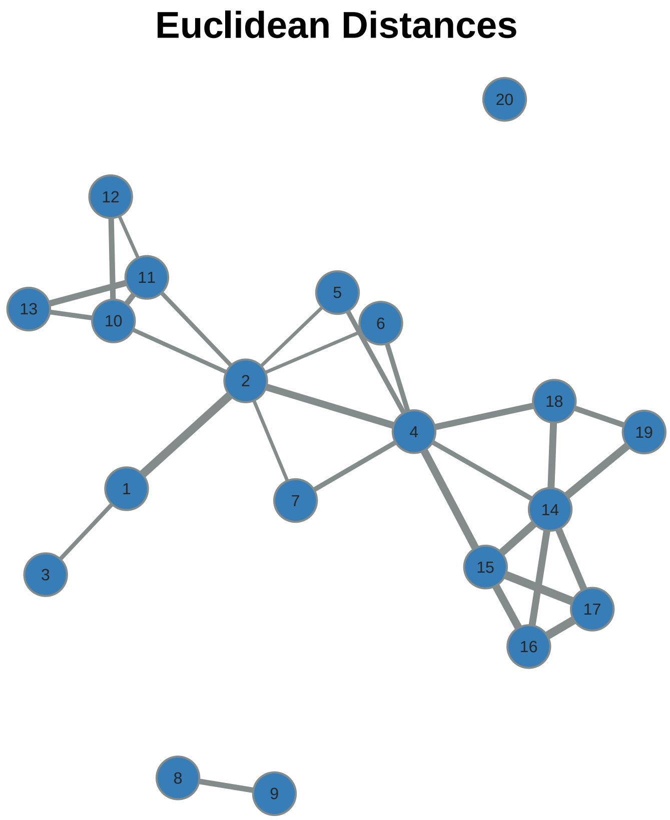
```
##    1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
## 1  0 1 1 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0
## 2  1 0 0 1 1 1 1 0 0  1  1  0  0  0  0  0  0  0  0  0
## 3  1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0
## 4  0 1 0 0 1 1 1 0 0  0  0  0  0  1  1  0  0  1  0  0
## 5  0 1 0 1 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0
## 6  0 1 0 1 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0
## 7  0 1 0 1 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0
## 8  0 0 0 0 0 0 0 0 1  0  0  0  0  0  0  0  0  0  0  0
## 9  0 0 0 0 0 0 0 1 0  0  0  0  0  0  0  0  0  0  0  0
## 10 0 1 0 0 0 0 0 0 0  0  1  1  1  0  0  0  0  0  0  0
## 11 0 1 0 0 0 0 0 0 0  1  0  1  1  0  0  0  0  0  0  0
## 12 0 0 0 0 0 0 0 0 0  1  1  0  0  0  0  0  0  0  0  0
## 13 0 0 0 0 0 0 0 0 0  1  1  0  0  0  0  0  0  0  0  0
## 14 0 0 0 1 0 0 0 0 0  0  0  0  0  0  1  1  1  1  1  0
## 15 0 0 0 1 0 0 0 0 0  0  0  0  0  1  0  1  1  0  0  0
## 16 0 0 0 0 0 0 0 0 0  0  0  0  0  1  1  0  1  0  0  0
## 17 0 0 0 0 0 0 0 0 0  0  0  0  0  1  1  1  0  0  0  0
## 18 0 0 0 1 0 0 0 0 0  0  0  0  0  1  0  0  0  0  1  0
## 19 0 0 0 0 0 0 0 0 0  0  0  0  0  1  0  0  0  1  0  0
## 20 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0
```

```
library(RColorBrewer)
par(mar=c(1,1,1,1),mfrow=c(2,3))
col5 <- brewer.pal(5, 'Set1')
cols <- ifelse(apriorformal[[1]] == 1, col5[1],
          ifelse(apriorformal[[1]] == 2, col5[2],
            ifelse(apriorformal[[1]] == 3, col5[3],
              ifelse(apriorformal[[1]] == 4, col5[4], col5[5]))))
```

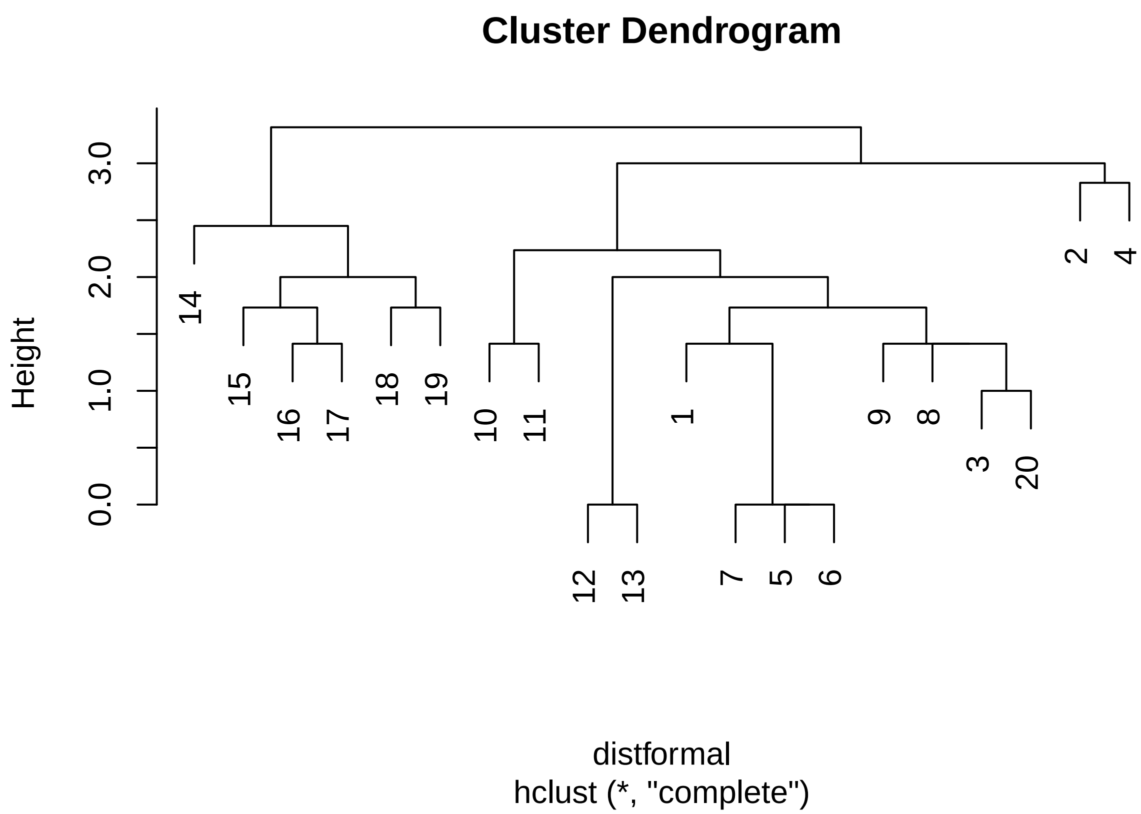
```
par(mar=c(1,1,2,1),mfrow=c(1,1))
plot(snasymmformal, main = 'Apriori Block Model', coord=mycoord,
     vertex.cex =3, edge.col='azure4', vertex.col=cols,
     vertex.border='azure4', label=seq(1:20), label.pos=5,
     label.cex=.5, label.col='gray15')
```



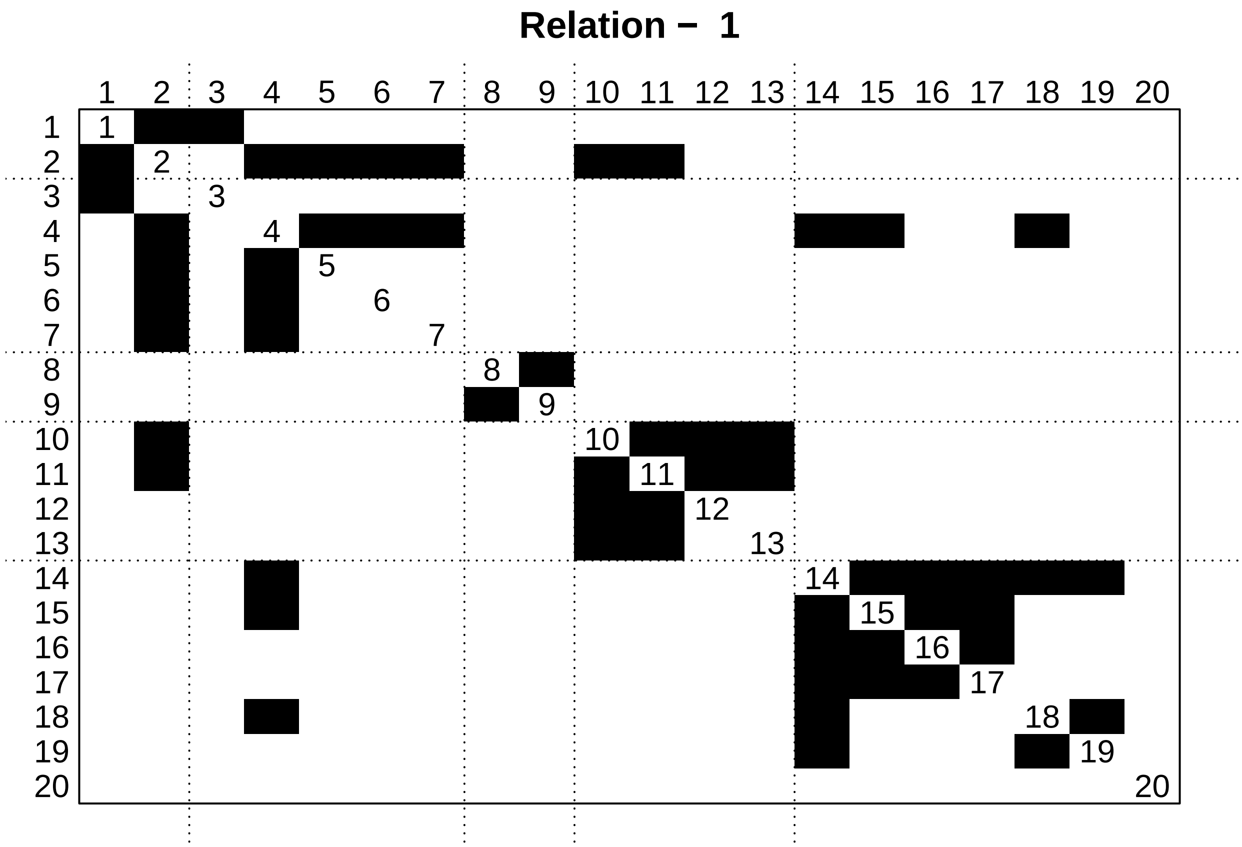

```
distformal <- dist(snasymmformal, method="euclidian", diag=FALSE)
thick <- as.vector(distformal)

par(mar=c(0.5,0,2,0))
plot(snasymmformal, main = 'Euclidean Distances', coord=mycoord,
     vertex.cex =3, edge.col='azure4', vertex.col=col5[2],
     vertex.border='azure4', label=seq(1:20),label.pos=5,
     label.cex=.5,label.col='gray15', edge.lwd = thick^2)
```



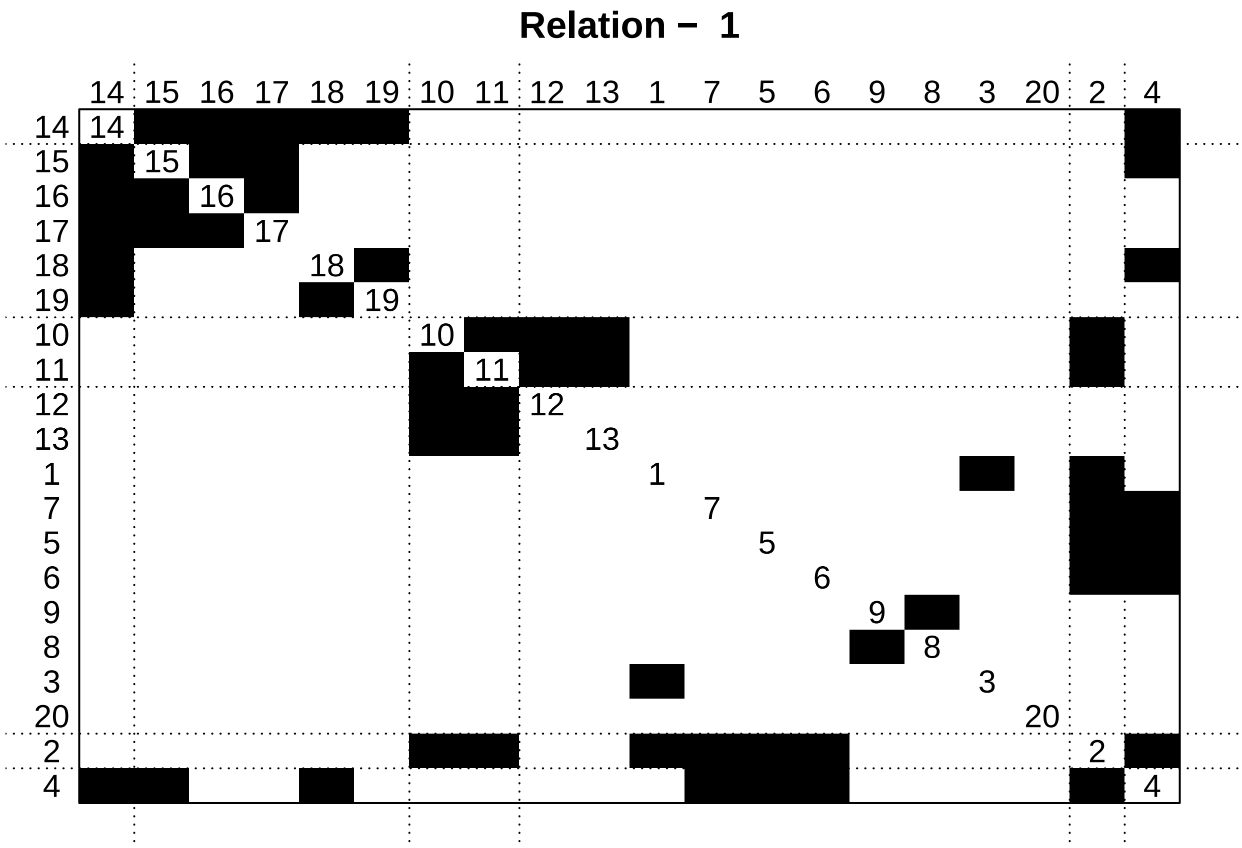

```r
formalclust <- hclust(distformal, method="complete")
plot(formalclust)
```




```
exploratoryformal<-blockmodel(snasymmformal, formalclust, k=6,
                               block.content="density", mode="graph",
                               diag=FALSE)

par(mar=c(0,0,2,0))
plot.blockmodel(aprioriformal)
```

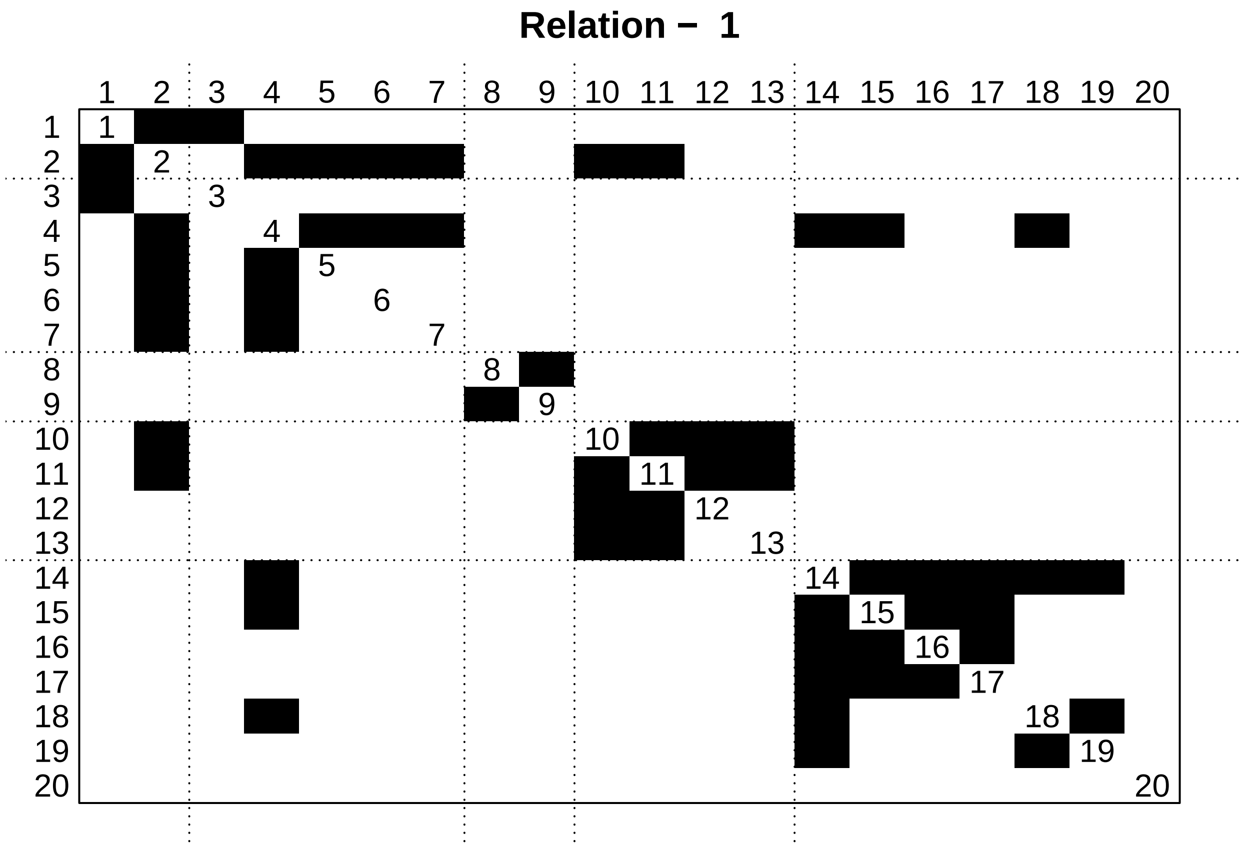



```
plot.blockmodel(exploratoryformal)
```

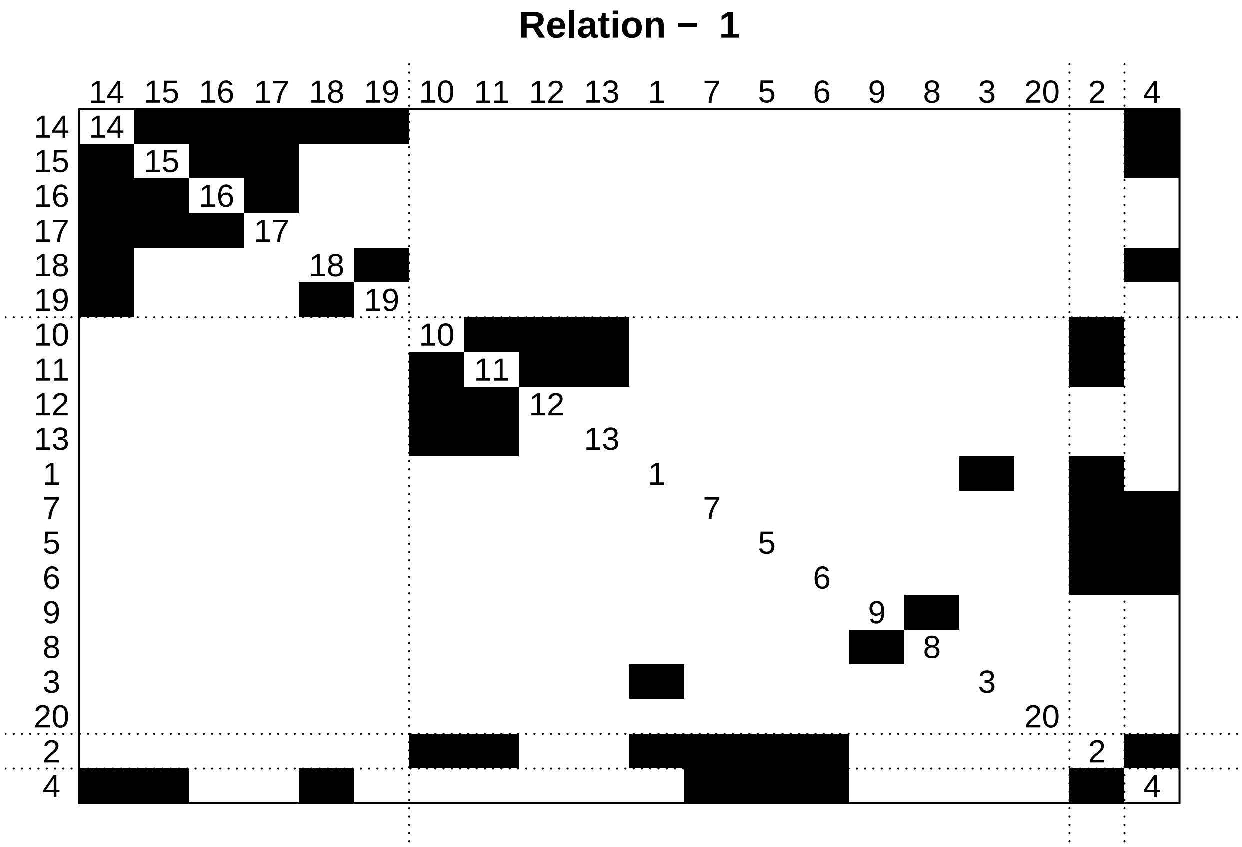


Assignment task. 1. Experiment with k. We???ve set it to 6, but would another number make more sense? 2. Which of the two blockmodels appear to be more accurate to you? Why?

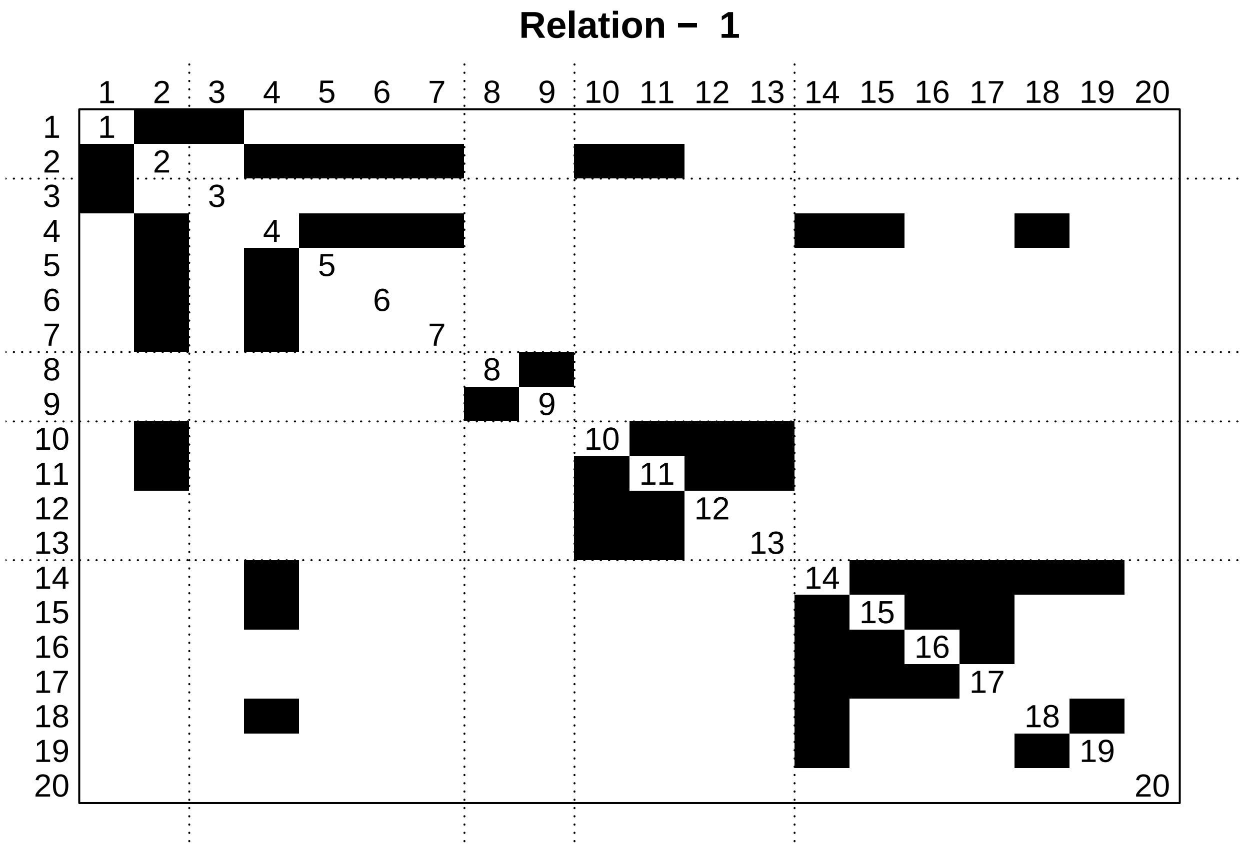
```
exploratoryformal<-blockmodel(snasymmformal, formalclust, k=4,
                              block.content="density", mode="graph",
                              diag=FALSE)

par(mar=c(0,0,2,0))
plot.blockmodel(aprioriformal)
```




```
plot.blockmodel(exploratoryformal)
```

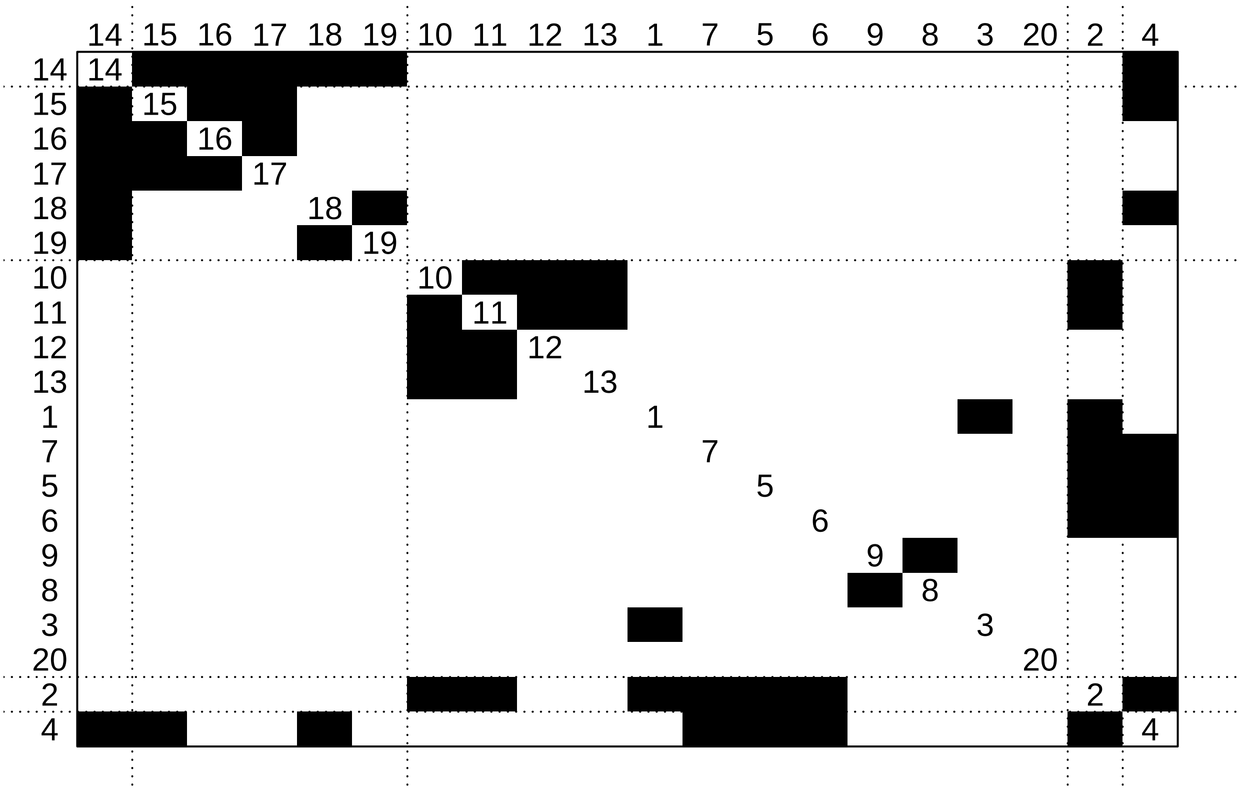



```
exploratoryformal<-blockmodel(snasymmformal, formalclust, k=5,
                              block.content="density", mode="graph",
                              diag=FALSE)

par(mar=c(0,0,2,0))
plot.blockmodel(aprioriformal)
```




```
plot.blockmodel(exploratoryformal)
```

Relation − 1

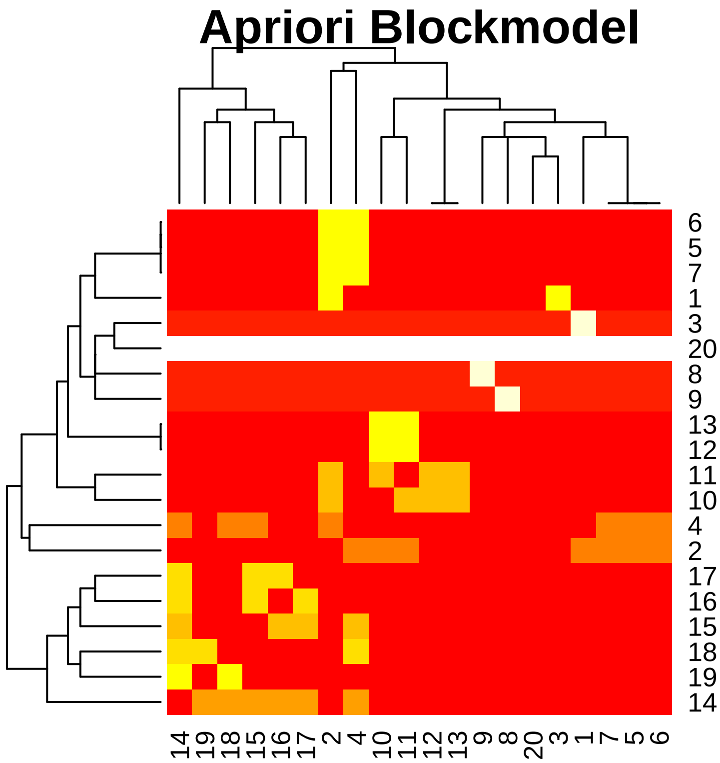


Assignments:

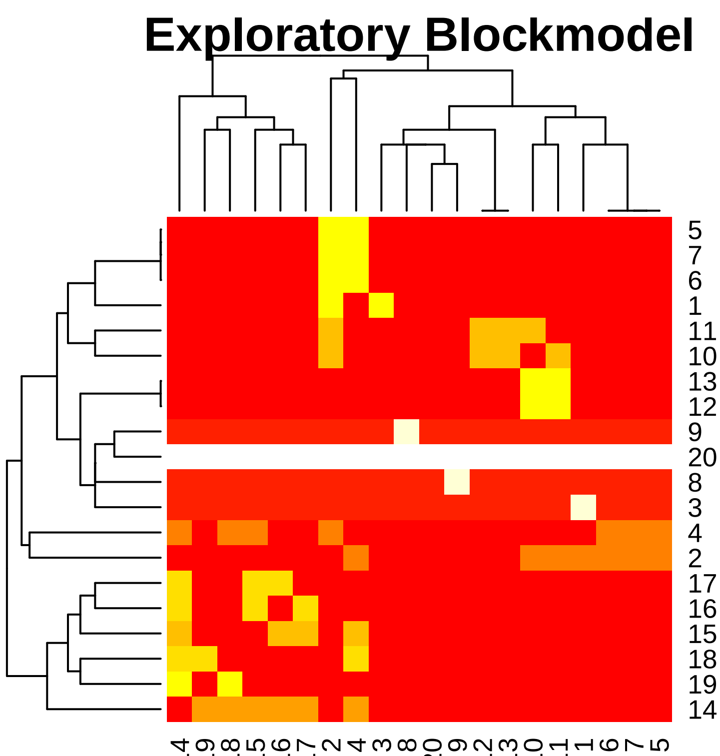
1. I have tried to build a k=4 and k=5 models, and but it doesn't bring a lot of differences. The point is that the other clusters that are made contain only one node, and it doesn't bring us a lot of sense in that way. In that parcitular case, the 4, 5 or 6 k models don't differ very much.

2. The more accurate model is the second one (the 'exploratoryformal' one). It has a more accurate and contrast borders of clusters, moreover, the clusters are more differed and can be interpreted better.

```
par(mar = c(1,1,4,1), mfrow = c(1,2))
heatmap(aprioriformal[[4]], main ='Apriori Blockmodel')
```




```r
heatmap(exploratoryformal[[4]], main ='Exploratory Blockmodel')
```

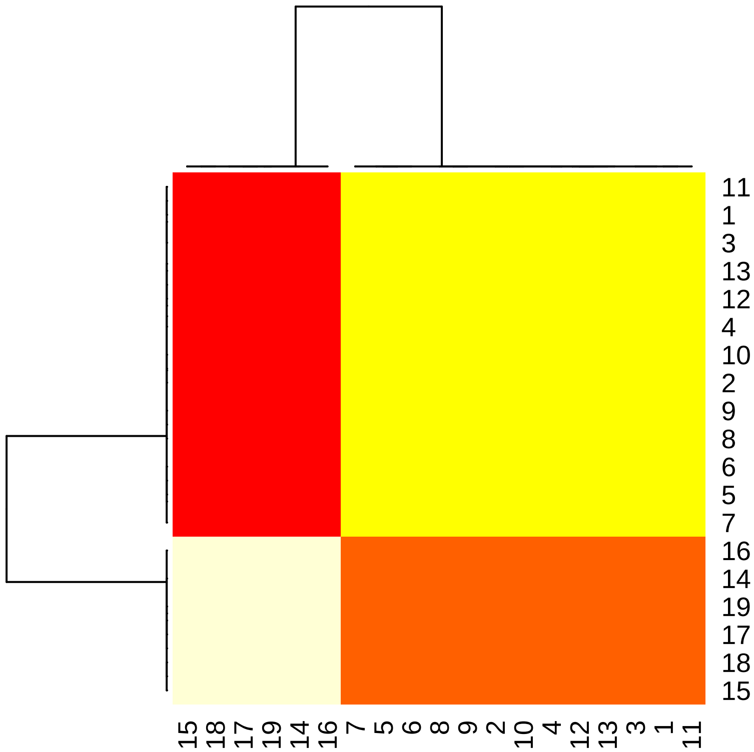



```r
connectedformal<-formal[-20,-20]
class(connectedformal)
```

```
## [1] "matrix"
```

```r
CONCOR <- function(mat, max.iter=1000, epsilon=1e-10){
  mat <- rbind(mat, t(mat)) # stack
  colN <- ncol(mat) # width
  X <- matrix(rep(0, times=colN*colN), nrow=colN, ncol=colN)
  target.abs.value <- colN * colN - epsilon # convergence target
  for (iter in 1:max.iter){
    for(i in 1:colN){
      for(j in i:colN){
        X[i,j]<-cor(mat[,i], mat[,j], method=c("pearson"))
      } # end for j
    } # end for i
    mat <- X+(t(X)-diag(diag((X))))
    if (sum(abs(mat)) > target.abs.value) { # test convergence
      #Finished before max.iter iterations
      return(mat)
      } # end if
  } # end for iterations
  return(mat) # return matrix
} # end function
```

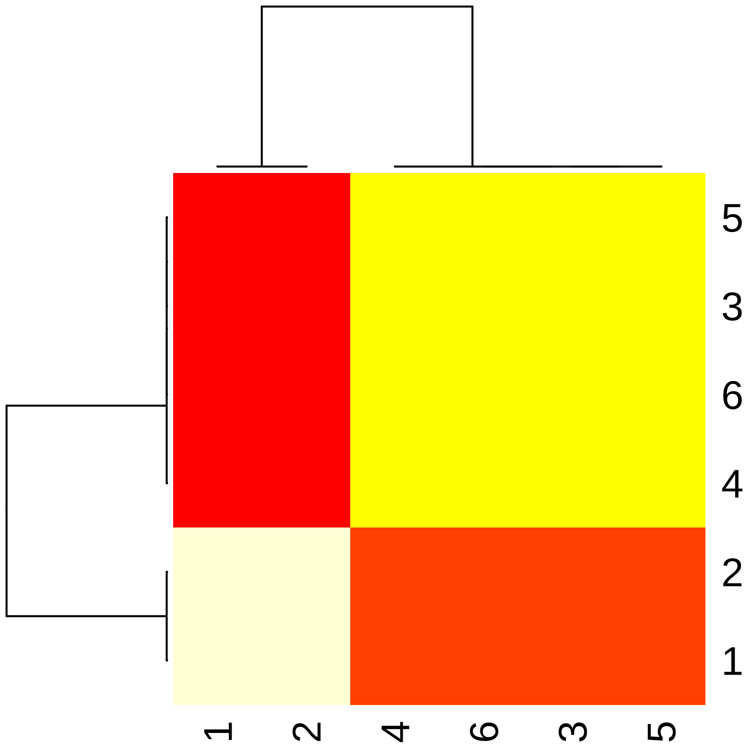
```r
rownames(connectedformal) <- row.names(roles)[1:19]
colnames(connectedformal) <- row.names(roles)[1:19]

CONCORFORMAL<-CONCOR(connectedformal)

heatmap(CONCORFORMAL)
```



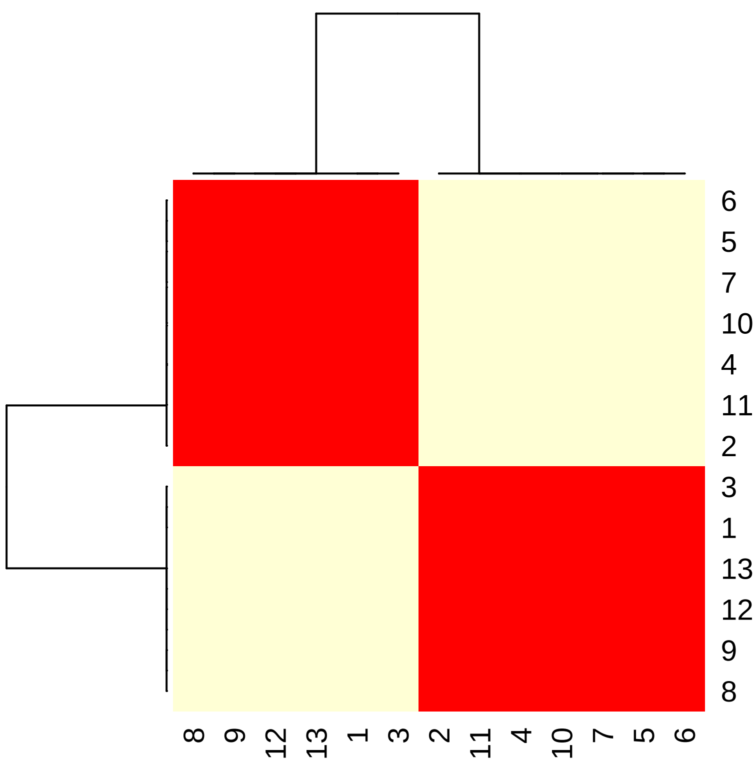

```
part1 <- connectedformal[14:19,14:19]
colnames(part1)
```

```
## [1] "Ms. Hughes"  "O'Brian"     "Anna"        "Gwen"        "Ms. Patmore"
## [6] "Daisy"
```

```
concor1 <- CONCOR(part1)
heatmap(concor1)
```



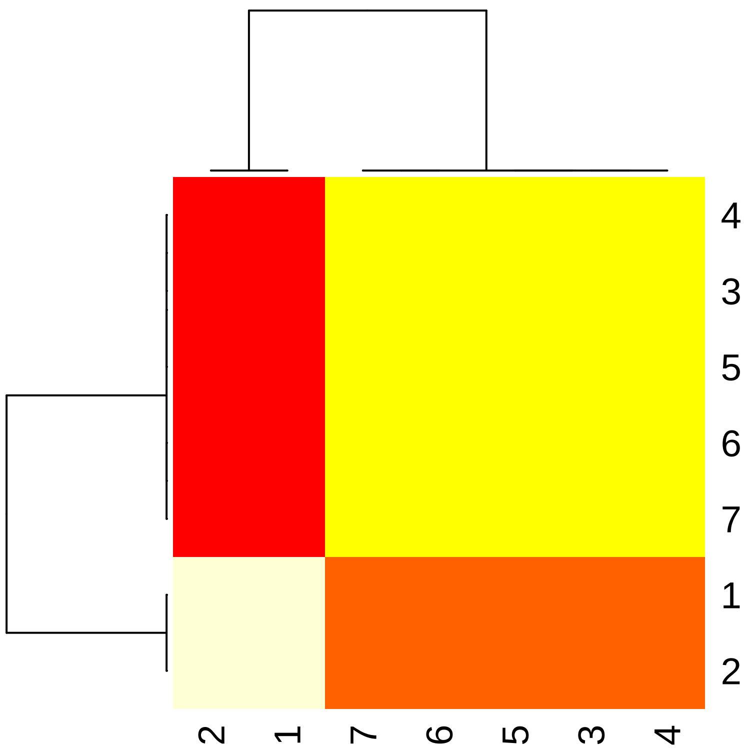

```
part2 <- connectedformal[1:13,1:13]

concor2 <- CONCOR(part2)
heatmap(concor2)
```



```
part3<-c(1,3,8,9,12,13)
part3.1<-part2[part3,part3]
colnames(part3.1)
```

```
## [1] "Violet Grantham"        "Lady Rosamund Grantham"
## [3] "Isobel Crawley"         "Matthew Crawley"
## [5] "Thomas"                 "William"
```

```
part3.2 <- part2[-part3,-part3]
concor3.2 <- CONCOR(part3.2)
heatmap(concor3.2)
```




```
colnames(part3.2[1:2,1:2])
```

```
## [1] "Robert Grantham" "Cora Grantham"
```

```
colnames(part3.2[3:7,3:7])
```

```
## [1] "Lady Mary Grantham"  "Lady Edith Grantham" "Lady Sybil Grantham"
## [4] "Mr. Carson"          "John Bates"
```

```
part3.2.2 <- part3.2[3:7,3:7]

##concor3.2.2<-CONCOR(part3.2.2)
```

Assignment task. Try not to get lost in all the partitions! Please list all the finite block-partitions that we have generated and the names of all people that ended up in every block

```
#part1
colnames(part1)
```

```
## [1] "Ms. Hughes"  "O'Brian"     "Anna"        "Gwen"        "Ms. Patmore"
## [6] "Daisy"
```

```
#part3.1
colnames(part3.1)
```

```
## [1] "Violet Grantham"        "Lady Rosamund Grantham"
## [3] "Isobel Crawley"         "Matthew Crawley"
## [5] "Thomas"                 "William"
```

```
#part3.2
colnames(part3.2[1:2,1:2])
```

```
## [1] "Robert Grantham" "Cora Grantham"
```

```
#part3.2.2
colnames(part3.2.2)
```

```
## [1] "Lady Mary Grantham"  "Lady Edith Grantham" "Lady Sybil Grantham"
## [4] "Mr. Carson"          "John Bates"
```

Homework 3

1. Choose a dataset from one of our previous labs.
2. Apply the same routines we did for the exploratory blockmodel here (it???s just copy/paste and then explore the k option). Make a heatmap for your model, vary the k, make a heatmap again. . . ..etc., until you select a k.
3. Apply the CONCOR function to the dataset you selected and plot its heatmap side by side with the heatmap for your exploratory blockmodel.

```
load('flo.Rdata')
flo.marriage.net <- as.network(as.matrix(flo.marriage), directed=FALSE)
flo.biz.net <- as.network(as.matrix(flo.biz), directed=FALSE)
```

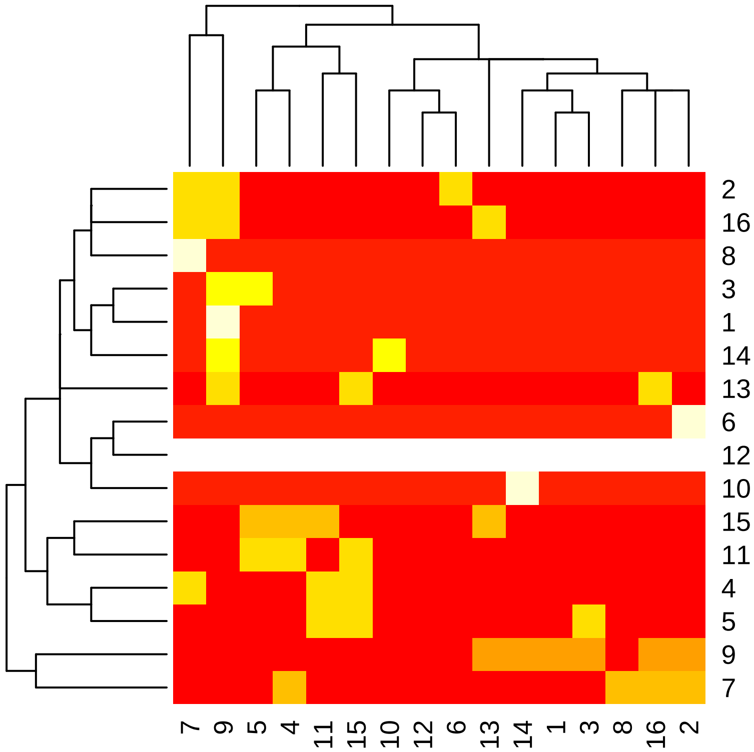
I have decided to run all the codes on two sets of data:

- Florentine marriages
- Florentine bisnesses

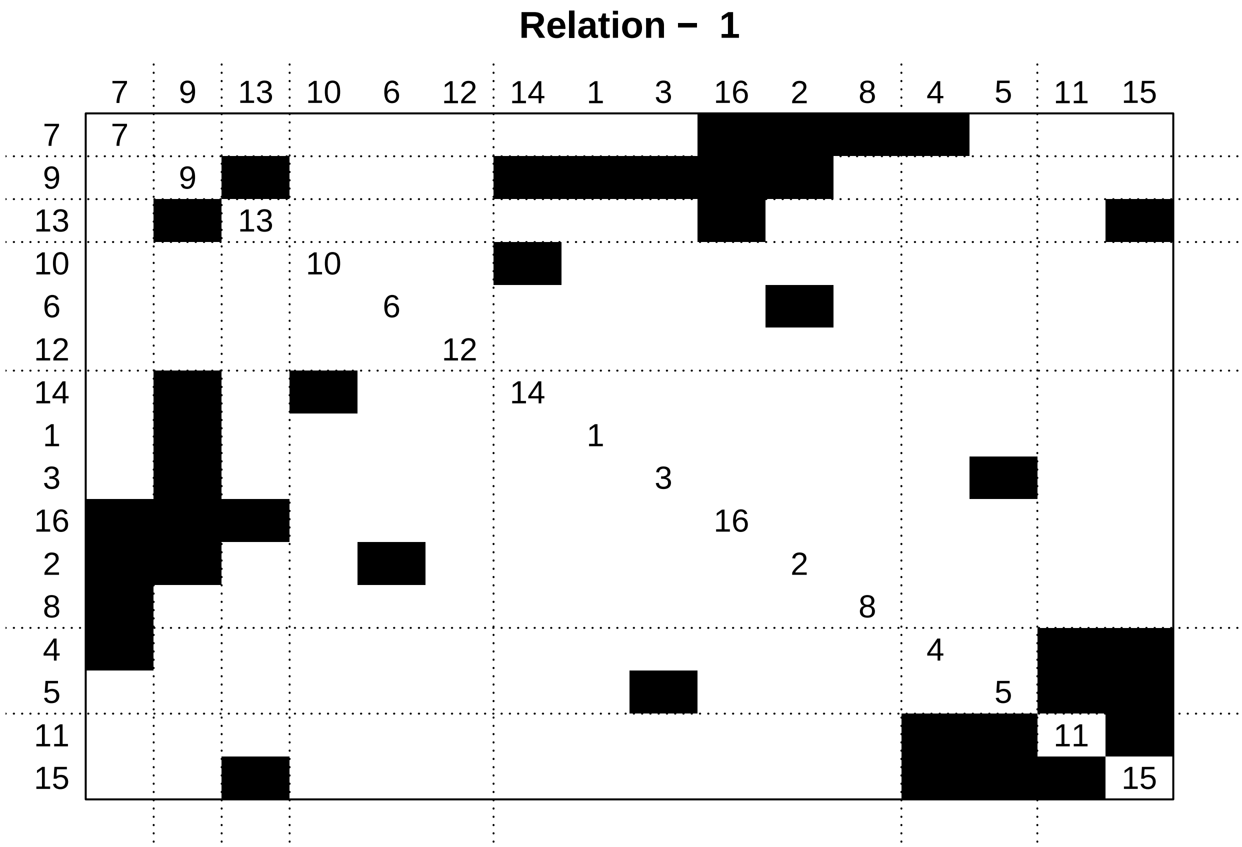
First of all, I will do everything on the set of Florentine marriages

```
snasymm.flomar <- flo.marriage.net
apriori.flomar<-blockmodel(flo.marriage.net, flo.att[[4]],
                           block.content="density", mode="graph",
                           diag=FALSE)
```

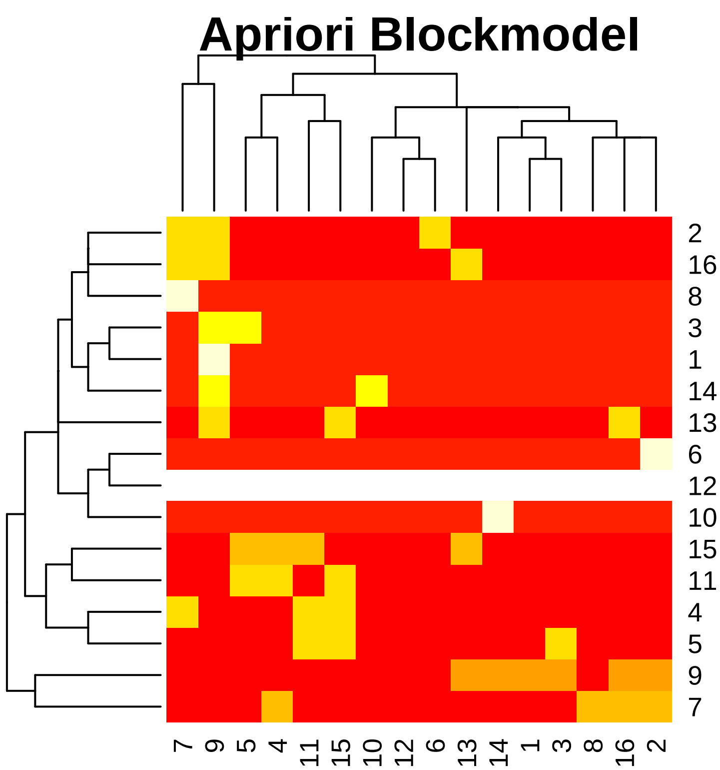
```
heatmap(apriori.flomar[[4]])
```



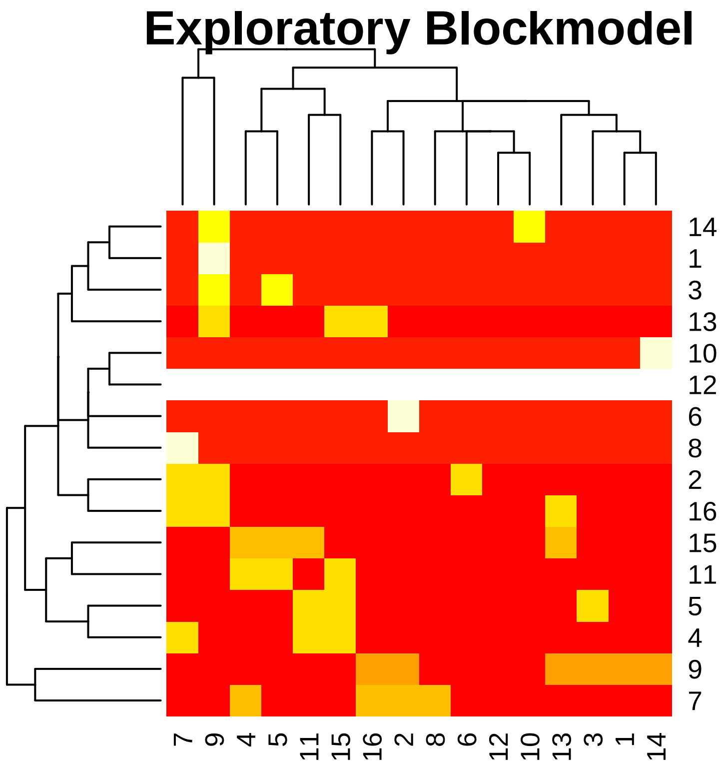

```
dist.flomar <- dist(snasymm.flomar, method="euclidian", diag=FALSE)
thick <- as.vector(dist.flomar)

clust.flomar <- hclust(dist.flomar, method="complete")

exploratory.flomar<-blockmodel(snasymm.flomar, clust.flomar, k=7,
block.content="density", mode="graph",
diag=FALSE)

par(mar=c(0,0,2,0))
plot.blockmodel(exploratory.flomar)
```




```
par(mar = c(1,1,4,1), mfrow = c(1,2))
heatmap(apriori.flomar[[4]], main ='Apriori Blockmodel')
```



```
heatmap(exploratory.flomar[[4]], main ='Exploratory Blockmodel')
```



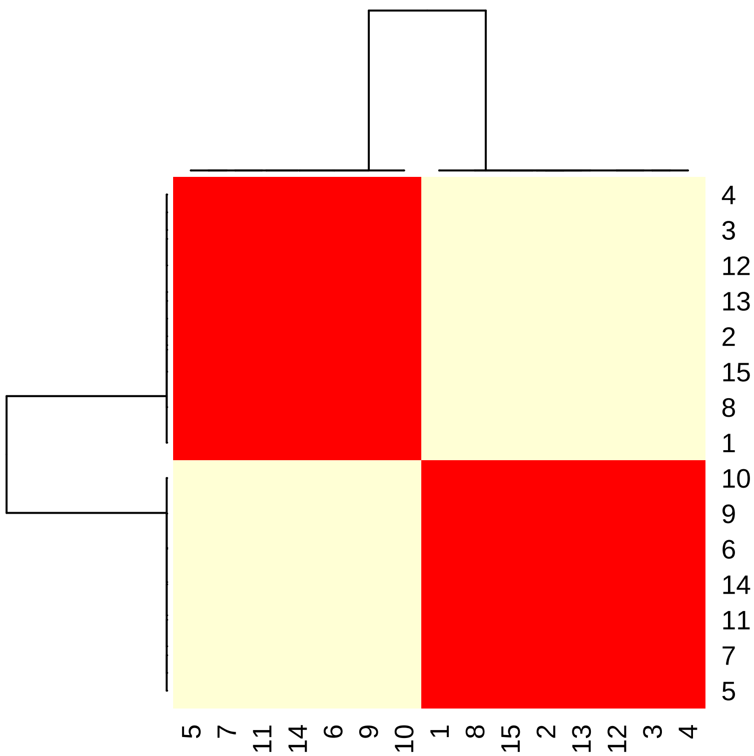
According to the "exploratory blockmodel", that is the way, in which it is possible to cluster the marriages. k=7 model is the most optimal one. Moreover, it will let us to work with the block 14, 1, 3, 16, 2, 8 separetly, in case, there are any other analysises can be made.

```
CONCOR <- function(mat, max.iter=1000, epsilon=1e-10){
  mat <- rbind(mat, t(mat)) # stack
  colN <- ncol(mat) # width
  X <- matrix(rep(0, times=colN*colN), nrow=colN, ncol=colN)
  target.abs.value <- colN * colN - epsilon # convergence target
  for (iter in 1:max.iter){
    for(i in 1:colN){
      for(j in i:colN){
        X[i,j]<-cor(mat[,i], mat[,j], method=c("pearson"))
      } # end for j
    } # end for i
    mat <- X+(t(X)-diag(diag((X))))
    if (sum(abs(mat)) > target.abs.value) { # test convergence
      #Finished before max.iter iterations
      return(mat)
      } # end if
  } # end for iterations
  return(mat) # return matrix
} # end function
```

```
connected.flomar<-flo.marriage.net[-12,-12]
class(connected.flomar)
```

```
## [1] "matrix"
```

```
rownames(connected.flomar) <- row.names(flo.marriage)[c(1,2,3,4,5,6,7,8,9,10,11,13,14,15,16)]
colnames(connected.flomar) <- row.names(flo.marriage)[c(1,2,3,4,5,6,7,8,9,10,11,13,14,15,16)]

CONCOR.flomar <- CONCOR(connected.flomar)

heatmap(CONCOR.flomar)
```




```
part1 <- connectedformal[14:19,14:19]

part1 <- connected.flomar[c(5, 6, 7, 9, 10, 11, 14), c(5, 6, 7, 9, 10, 11, 14)]
part2 <- connected.flomar[c(1, 2, 3, 4, 8, 12, 13, 15), c(1, 2, 3, 4, 8, 12, 13, 15)]
part1
```

```
##           CASTELLAN GINORI GUADAGNI MEDICI PAZZI PERUZZI STROZZI
## CASTELLAN         0      0        0      0     0       1       1
## GINORI            0      0        0      0     0       0       0
## GUADAGNI          0      0        0      0     0       0       0
## MEDICI            0      0        0      0     0       0       0
## PAZZI             0      0        0      0     0       0       0
```

```
## PERUZZI          1      0        0       0     0       0       1
## STROZZI          1      0        0       0     0       1       0
```

```
part2
```

```
##          ACCIAIUOL ALBIZZI BARBADORI BISCHERI LAMBERTES RIDOLFI SALVIATI
## ACCIAIUOL         0       0         0        0         0       0        0
## ALBIZZI           0       0         0        0         0       0        0
## BARBADORI         0       0         0        0         0       0        0
## BISCHERI          0       0         0        0         0       0        0
## LAMBERTES         0       0         0        0         0       0        0
## RIDOLFI           0       0         0        0         0       0        0
## SALVIATI          0       0         0        0         0       0        0
## TORNABUON         0       0         0        0         0       1        0
##          TORNABUON
## ACCIAIUOL         0
## ALBIZZI           0
## BARBADORI         0
## BISCHERI          0
## LAMBERTES         0
## RIDOLFI           1
## SALVIATI          0
## TORNABUON         0
```

```
##concor1 <- CONCOR(part1)
##concor2 <- CONCOR(part2)
```

```
colnames(part1)
```

```
## [1] "CASTELLAN" "GINORI"    "GUADAGNI"  "MEDICI"    "PAZZI"     "PERUZZI"
## [7] "STROZZI"
```

```
colnames(part2)
```

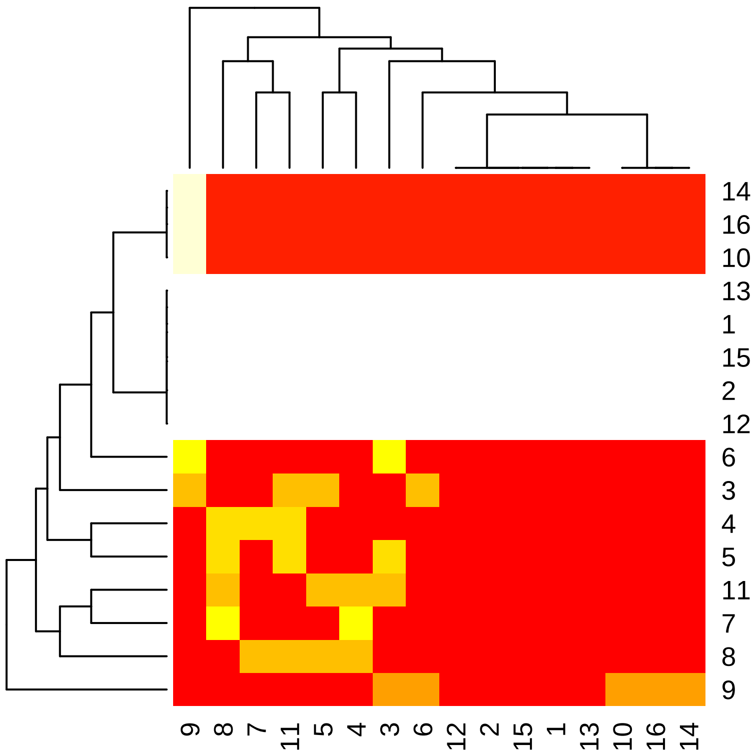
```
## [1] "ACCIAIUOL" "ALBIZZI"   "BARBADORI" "BISCHERI"  "LAMBERTES" "RIDOLFI"
## [7] "SALVIATI"  "TORNABUON"
```

According to the CONCOR analysis, there is anly two blocks that can be separated. The further analysis isn't possible. It can be explaines b ythe fact, that there are not a lot of mariages connections, and based on the characteristics of network it is not possbile to make a more deeper analysis.

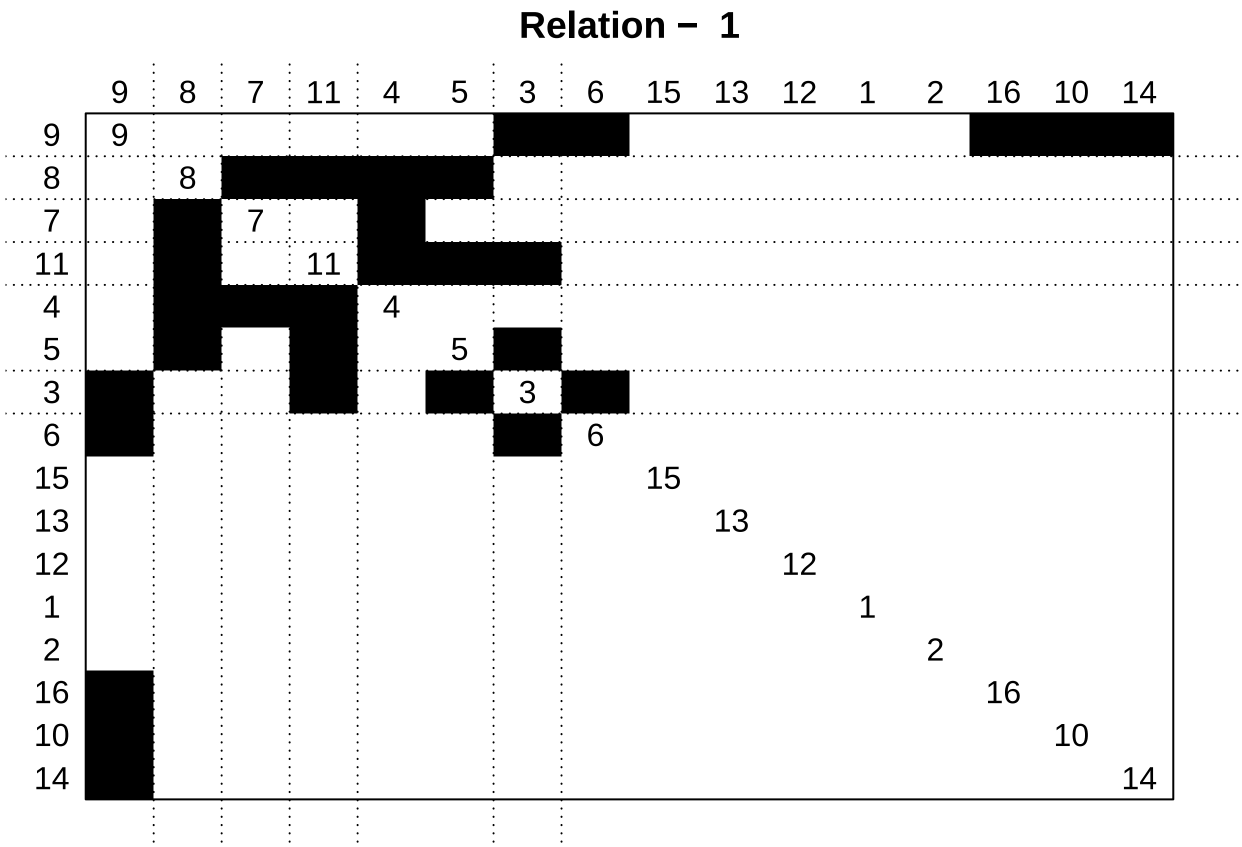
Now, I will explore the data of Florentine bisnesses

```
snasymm.flobiz <- flo.biz.net
apriori.flobiz<-blockmodel(flo.biz.net, flo.att[[2]],
                           block.content="density", mode="graph",
                           diag=FALSE)
```

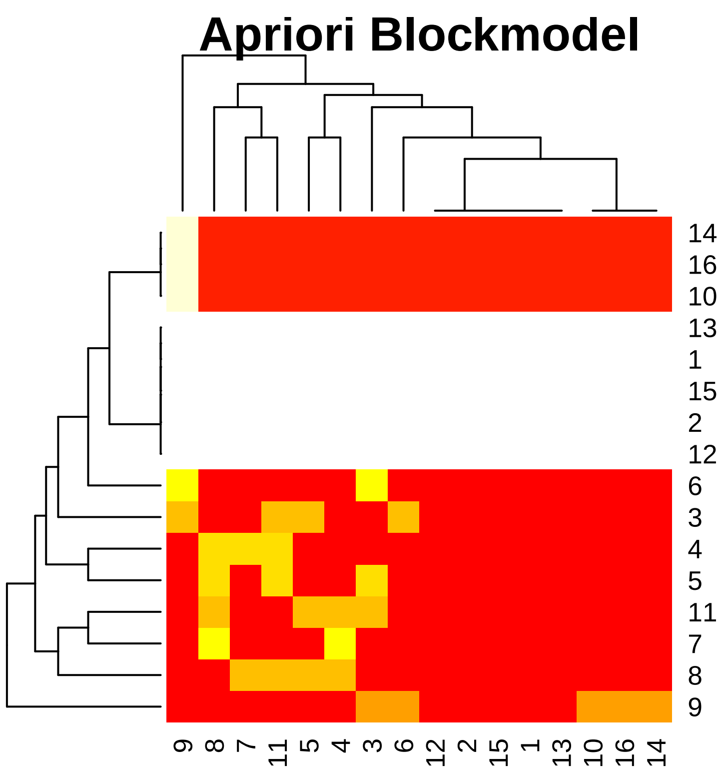
```
heatmap(apriori.flobiz[[4]])
```




```
dist.flobiz <- dist(snasymm.flobiz, method="euclidian", diag=FALSE)
thick <- as.vector(dist.flobiz)

clust.flobiz <- hclust(dist.flobiz, method="complete")

exploratory.flobiz<-blockmodel(snasymm.flobiz, clust.flobiz, k=7,
block.content="density", mode="graph",
diag=FALSE)

par(mar=c(0,0,2,0))
plot.blockmodel(exploratory.flobiz)
```




```r
par(mar = c(1,1,4,1), mfrow = c(1,2))
heatmap(apriori.flobiz[[4]], main ='Apriori Blockmodel')
```

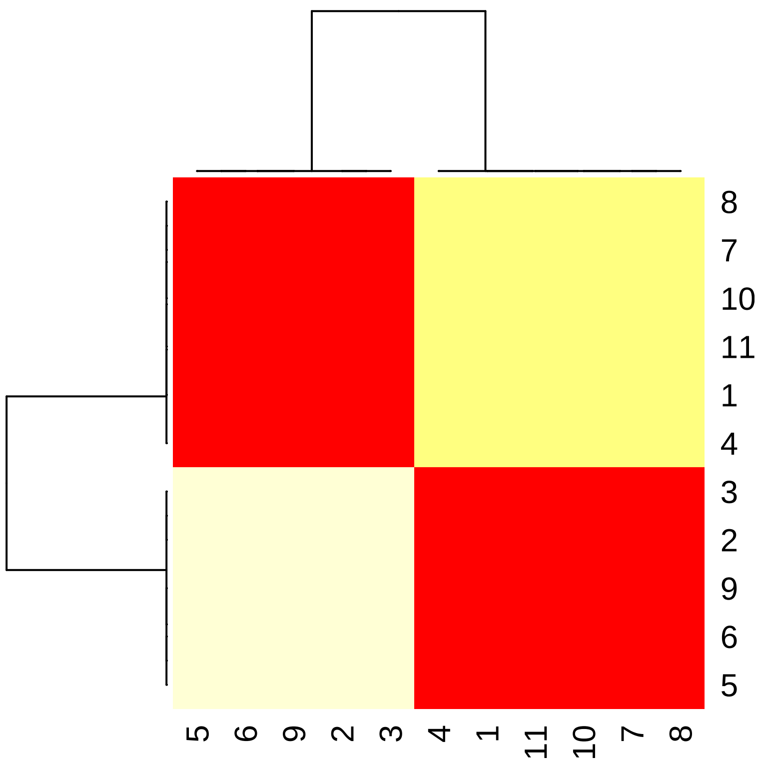


According to the "exploratory blockmodel", that is the way, in which it is possible to cluster the businesses. k=7 model is the most optimal one. Moreover, it will let us to work with the block 1, 2, 12, 13, 15 separetly, in case, there are any other analysises can be made.

```
connected.flobiz<-flo.biz.net[c(3,4,5,6,7,8,9,10,11,14,16), c(3,4,5,6,7,8,9,10,11,14,16)]

class(connected.flomar)
```

```
## [1] "matrix"
```

```
rownames(connected.flobiz) <- row.names(flo.biz)[c(3,4,5,6,7,8,9,10,11,14,16)]
colnames(connected.flobiz) <- row.names(flo.biz)[c(3,4,5,6,7,8,9,10,11,14,16)]

CONCOR.flobiz <- CONCOR(connected.flobiz)

heatmap(CONCOR.flobiz)
```




```
part1 <- connected.flobiz[c(2,3,5,6,9), c(2,3,5,6,9)]
part2 <- connected.flobiz[c(1,4,7,8,10,11), c(1,4,7,8,10,11)]
part1
```

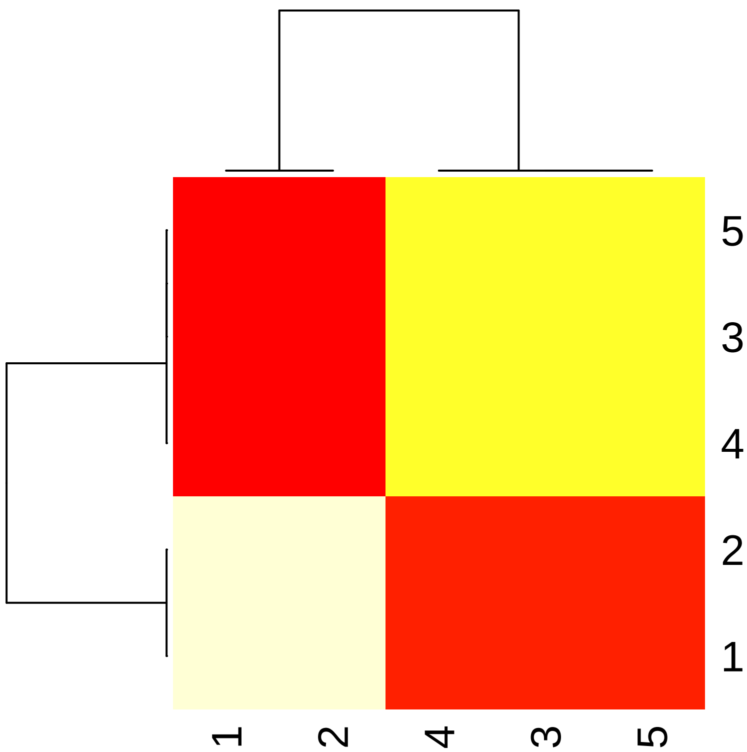
```
##           BISCHERI CASTELLAN GUADAGNI LAMBERTES PERUZZI
## BISCHERI         0         0        1         1       1
## CASTELLAN        0         0        0         1       1
## GUADAGNI         1         0        0         1       0
## LAMBERTES        1         1        1         0       1
```

```
## PERUZZI          1         1        0         1       0
```

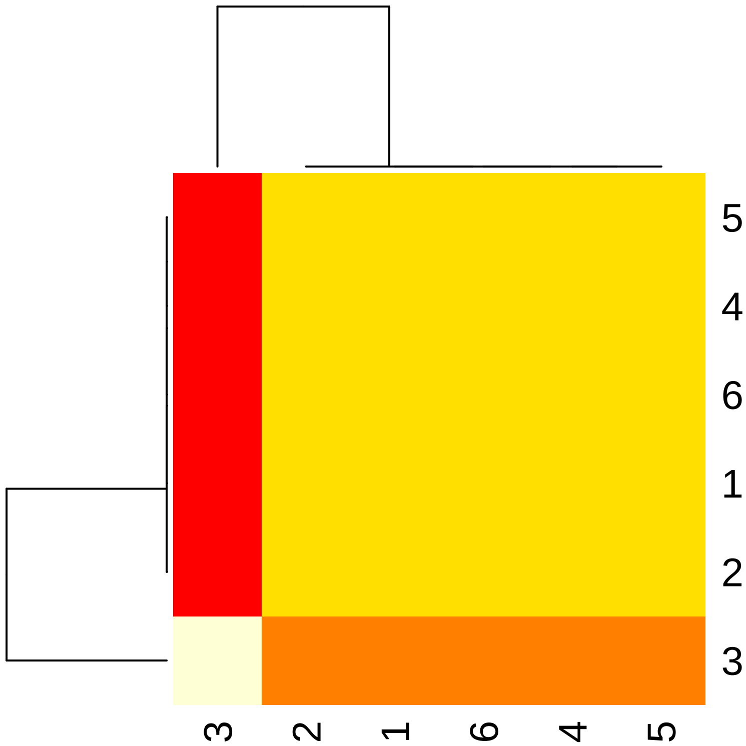
```
part2
```

```
##            BARBADORI GINORI MEDICI PAZZI SALVIATI TORNABUON
## BARBADORI          0      1      1     0        0         0
## GINORI             1      0      1     0        0         0
## MEDICI             1      1      0     1        1         1
## PAZZI              0      0      1     0        0         0
## SALVIATI           0      0      1     0        0         0
## TORNABUON          0      0      1     0        0         0
```

```
concor1 <- CONCOR(part1)
concor2 <- CONCOR(part2)

heatmap(concor1)
```




```
heatmap(concor2)
```




```
part3<-c(1, 2)
part3.1<-part1[part3,part3]
colnames(part3.1)
```

```
## [1] "BISCHERI"  "CASTELLAN"
```

```
##concor3 <- CONCOR(part3.1)

part3.2
```

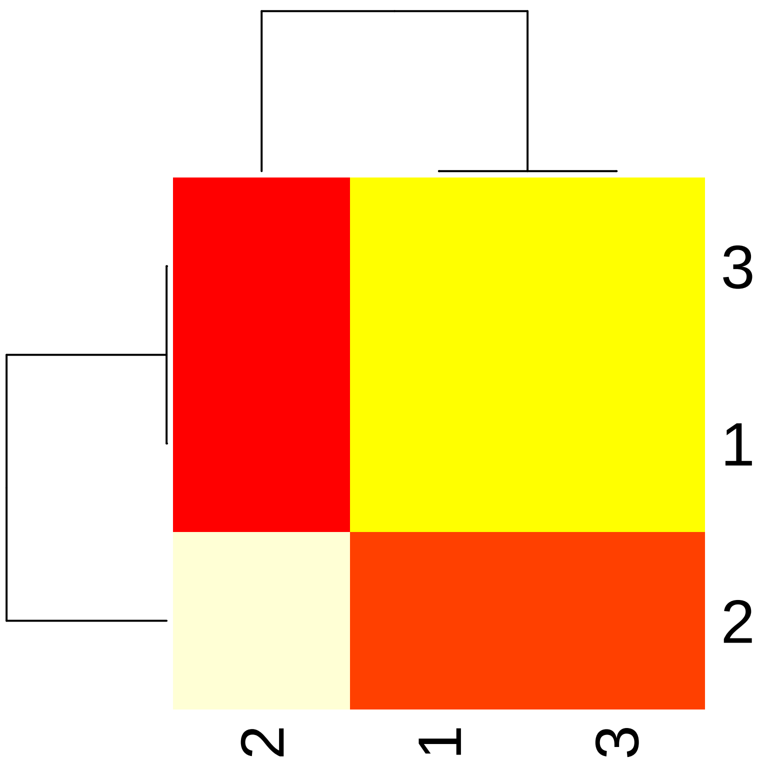
```
##                     Robert Grantham Cora Grantham Lady Mary Grantham
## Robert Grantham                   0             1                  1
## Cora Grantham                     1             0                  1
## Lady Mary Grantham                0             0                  0
## Lady Edith Grantham               0             0                  0
## Lady Sybil Grantham               0             0                  0
## Mr. Carson                        0             0                  0
## John Bates                        0             0                  0
##                     Lady Edith Grantham Lady Sybil Grantham Mr. Carson
## Robert Grantham                       1                   1          1
## Cora Grantham                         1                   1          0
## Lady Mary Grantham                    0                   0          0
## Lady Edith Grantham                   0                   0          0
## Lady Sybil Grantham                   0                   0          0
## Mr. Carson                            0                   0          0
## John Bates                            0                   0          0
```

```
##                         John Bates
## Robert Grantham                  1
## Cora Grantham                    0
## Lady Mary Grantham               0
## Lady Edith Grantham              0
## Lady Sybil Grantham              0
## Mr. Carson                       1
## John Bates                       0
```

```
part3.2<-part1[-part3,-part3]
colnames(part3.2)
```

```
## [1] "GUADAGNI"  "LAMBERTES" "PERUZZI"
```

```
concor3.1 <- CONCOR(part3.2)
heatmap(concor3.1)
```




```
part3.2.1 <- part3.2[-2, -2]
colnames(part3.2.1)
```

```
## [1] "GUADAGNI" "PERUZZI"
```

```
##concor3.1.1 <- CONCOR(part3.2.1)
```

```
part4<-c(1, 2, 4, 5, 6)
part4.1<-part2[part4,part4]
colnames(part4.1)
```

```
## [1] "BARBADORI" "GINORI"    "PAZZI"     "SALVIATI"  "TORNABUON"
```

```
##concor4 <- CONCOR(part4.1)
```

```
colnames(part3.1)
```

```
## [1] "BISCHERI"  "CASTELLAN"
```

```
colnames(part3.2)[2]
```

```
## [1] "LAMBERTES"
```

```
colnames(part2)[3]
```

```
## [1] "MEDICI"
```

```
colnames(part3.2.1)
```

```
## [1] "GUADAGNI" "PERUZZI"
```

```
colnames(part4.1)
```

```
## [1] "BARBADORI" "GINORI"    "PAZZI"     "SALVIATI"  "TORNABUON"
```

The clustreing of businesses turned out to be able to conduct a more deeper anaysis. There are 5 groups I have been able to separate that data. It is not completely possible to tell, on what basis the separations by the CONCOR was made. I can suggest that is was done based on the amount of ties with other families or on the basis of wealth. To explain the results, a deeper thepretical research should be made.