Computer Lab - Graphical Models

Marta Kałużna

3/23/2020

G-Lasso. Method of Friedman

```
library(readxl)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 3.0-2
```

```
library(glasso)
library(MASS)
```

In our calculations, we will use the covariance matrix Gamma:

```
mat_min <- function(p){
  m <- matrix(0, p, p)
  for(i in 1:p){
    for(j in 1:p){
      m[i,j] <- min(i,j)
    }
  }
  return(m)
}

Gamma <- mat_min(10)
Gamma
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    1    1    1    1    1    1    1    1    1     1
##  [2,]    1    2    2    2    2    2    2    2    2     2
##  [3,]    1    2    3    3    3    3    3    3    3     3
##  [4,]    1    2    3    4    4    4    4    4    4     4
##  [5,]    1    2    3    4    5    5    5    5    5     5
##  [6,]    1    2    3    4    5    6    6    6    6     6
##  [7,]    1    2    3    4    5    6    7    7    7     7
##  [8,]    1    2    3    4    5    6    7    8    8     8
##  [9,]    1    2    3    4    5    6    7    8    9     9
## [10,]    1    2    3    4    5    6    7    8    9    10
```

Precision matrix of Gamma:

```
solve(Gamma)
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    2   -1    0    0    0    0    0    0    0     0
##  [2,]   -1    2   -1    0    0    0    0    0    0     0
##  [3,]    0   -1    2   -1    0    0    0    0    0     0
##  [4,]    0    0   -1    2   -1    0    0    0    0     0
##  [5,]    0    0    0   -1    2   -1    0    0    0     0
##  [6,]    0    0    0    0   -1    2   -1    0    0     0
##  [7,]    0    0    0    0    0   -1    2   -1    0     0
##  [8,]    0    0    0    0    0    0   -1    2   -1     0
##  [9,]    0    0    0    0    0    0    0   -1    2    -1
## [10,]    0    0    0    0    0    0    0    0   -1     1
```

Firstly, we apply the G-Lasso method to the data A for different regularisation parameters $\rho = 0.1$, 1, 5, 10, 100.

```
# e), f)
options(witdh=100)
n <- 5
p <- 10
Gamma <- mat_min(p)
A = mvrnorm(n, rep(0,p), Gamma)
A
```

```
##             [,1]       [,2]       [,3]       [,4]         [,5]       [,6]
## [1,]  0.89276002 -0.4868572 -1.1262868 -0.6491208  1.479635399  1.0715820
## [2,] -0.45140919 -0.5788762  0.9727870  0.1250413 -0.739851679  0.3277257
## [3,] -1.45541374 -2.6570663 -3.4726872 -5.7886358 -5.988582886 -5.2452802
## [4,] -0.02225135 -0.1030363 -0.2138314  0.2805496 -0.001877919 -0.2127585
## [5,] -1.51578010 -1.7824536 -1.3918257  0.6100591 -0.206481477 -0.2452072
##            [,7]        [,8]       [,9]      [,10]
## [1,]  1.6964153  2.50273217  1.5678338  2.5649534
## [2,]  1.1249838  0.66876029  1.4456395  0.5070470
## [3,] -5.0745620 -2.38457061 -1.1112626 -1.5511042
## [4,] -0.4324556 -0.51374938 -0.2945746  0.3626063
## [5,]  0.2794439 -0.09088912 -0.9173093 -0.4033554
```

```
S <- var(A)
gl <- list()
rho <- c(0.1, 1, 5, 10, 100)
for(i in 1:5){
  gl[[i]] <- glasso(S, rho = rho[i])
}

scaled_precision_matrices <- list()
for(j in 1:5){
  scaled_precision_matrices[[j]] <- round(cov2cor(gl[[j]]$wi),3)
}
```

Scaled precision matrices for different parameters:

```
options(width=100)
for(i in 1:5){
  cat('rho = ', rho[i], '\n')
  scaled_precision_matrices[[i]][abs(scaled_precision_matrices[[i]]) < 0.1] = 0
  print(scaled_precision_matrices[[i]])
  cat('\n')
}
```

```
## rho =  0.1
##         [,1]   [,2]   [,3]   [,4]   [,5]   [,6]   [,7]   [,8]   [,9]  [,10]
##  [1,]  1.000 -0.324  0.000  0.147  0.000  0.000  0.000  0.000  0.000 -0.428
##  [2,] -0.324  1.000 -0.360  0.000  0.000  0.000  0.000  0.000  0.000  0.000
##  [3,]  0.000 -0.360  1.000 -0.403  0.223 -0.111  0.000  0.190 -0.397  0.000
##  [4,]  0.147  0.000 -0.403  1.000 -0.378 -0.223 -0.262  0.000  0.316  0.186
##  [5,]  0.000  0.000  0.223 -0.378  1.000 -0.420  0.000  0.000  0.000 -0.280
##  [6,]  0.000  0.000 -0.111 -0.223 -0.420  1.000 -0.464  0.000  0.000  0.000
##  [7,]  0.000  0.000  0.000 -0.262  0.000 -0.464  1.000 -0.326 -0.160  0.000
##  [8,]  0.000  0.000  0.190  0.000  0.000  0.000 -0.326  1.000 -0.276 -0.381
##  [9,]  0.000  0.000 -0.397  0.316  0.000  0.000 -0.160 -0.276  1.000 -0.139
## [10,] -0.428  0.000  0.000  0.187 -0.281  0.000  0.000 -0.381 -0.139  1.000
##
## rho =  1
##         [,1]   [,2]  [,3]   [,4]   [,5]   [,6]   [,7]   [,8]   [,9]  [,10]
##  [1,]  1.000  0.000  0.00  0.000 -0.111  0.000  0.000  0.000  0.000  0.000
##  [2,]  0.000  1.000  0.00  0.000 -0.122  0.000  0.000  0.000  0.000  0.000
##  [3,]  0.000  0.000  1.00 -0.200  0.000  0.000 -0.120  0.000  0.000  0.000
##  [4,]  0.000  0.000 -0.20  1.000 -0.249 -0.216 -0.185  0.000  0.000  0.000
##  [5,] -0.111 -0.122  0.00 -0.249  1.000 -0.289 -0.248 -0.137  0.000 -0.150
##  [6,]  0.000  0.000  0.00 -0.216 -0.289  1.000 -0.284  0.000  0.000  0.000
##  [7,]  0.000  0.000 -0.12 -0.185 -0.248 -0.284  1.000 -0.186 -0.127  0.000
##  [8,]  0.000  0.000  0.00  0.000 -0.137  0.000 -0.186  1.000 -0.100 -0.168
##  [9,]  0.000  0.000  0.00  0.000  0.000  0.000 -0.127 -0.100  1.000  0.000
## [10,]  0.000  0.000  0.00  0.000 -0.150  0.000  0.000 -0.168  0.000  1.000
##
## rho =  5
##       [,1] [,2] [,3]  [,4]   [,5]   [,6]   [,7] [,8] [,9] [,10]
##  [1,]    1    0    0  0.00  0.000  0.000  0.000    0    0     0
##  [2,]    0    1    0  0.00  0.000  0.000  0.000    0    0     0
##  [3,]    0    0    1  0.00  0.000  0.000  0.000    0    0     0
##  [4,]    0    0    0  1.00 -0.120  0.000  0.000    0    0     0
##  [5,]    0    0    0 -0.12  1.000 -0.134 -0.153    0    0     0
##  [6,]    0    0    0  0.00 -0.134  1.000 -0.110    0    0     0
##  [7,]    0    0    0  0.00 -0.153 -0.110  1.000    0    0     0
##  [8,]    0    0    0  0.00  0.000  0.000  0.000    1    0     0
##  [9,]    0    0    0  0.00  0.000  0.000  0.000    0    1     0
## [10,]    0    0    0  0.00  0.000  0.000  0.000    0    0     1
##
## rho =  10
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    1    0    0    0    0    0    0    0    0     0
##  [2,]    0    1    0    0    0    0    0    0    0     0
##  [3,]    0    0    1    0    0    0    0    0    0     0
##  [4,]    0    0    0    1    0    0    0    0    0     0
##  [5,]    0    0    0    0    1    0    0    0    0     0
##  [6,]    0    0    0    0    0    1    0    0    0     0
##  [7,]    0    0    0    0    0    0    1    0    0     0
##  [8,]    0    0    0    0    0    0    0    1    0     0
##  [9,]    0    0    0    0    0    0    0    0    1     0
## [10,]    0    0    0    0    0    0    0    0    0     1
##
## rho =  100
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    1    0    0    0    0    0    0    0    0     0
##  [2,]    0    1    0    0    0    0    0    0    0     0
##  [3,]    0    0    1    0    0    0    0    0    0     0
##  [4,]    0    0    0    1    0    0    0    0    0     0
##  [5,]    0    0    0    0    1    0    0    0    0     0
##  [6,]    0    0    0    0    0    1    0    0    0     0
##  [7,]    0    0    0    0    0    0    1    0    0     0
##  [8,]    0    0    0    0    0    0    0    1    0     0
##  [9,]    0    0    0    0    0    0    0    0    1     0
## [10,]    0    0    0    0    0    0    0    0    0     1
```

We lose information about conditional dependence when $\rho \geq 5$. When $\rho = 1$, non-zero values are close to the diagonal.

Glassopath

```
gl_path <- glassopath(S)
```

```
options(width=100)
for(i in 1:length(gl_path$rholist)){
  cat('rho= ', round(gl_path$rholist[i],3), '\n')
  print(round(cov2cor(gl_path$wi[,,i]),3))
  cat('\n')
}
```

```
## rho=  0.817
##         [,1]   [,2]   [,3]   [,4]   [,5]   [,6]   [,7]   [,8]   [,9]  [,10]
##  [1,]  1.000  0.000  0.000  0.000 -0.107  0.000  0.000 -0.029  0.000 -0.112
##  [2,]  0.000  1.000 -0.041 -0.012 -0.131 -0.079 -0.024  0.000  0.000  0.000
##  [3,]  0.000 -0.041  1.000 -0.215  0.000 -0.089 -0.119  0.000  0.000  0.000
##  [4,]  0.000 -0.012 -0.215  1.000 -0.246 -0.219 -0.179  0.000  0.000  0.000
##  [5,] -0.107 -0.131  0.000 -0.246  1.000 -0.298 -0.247 -0.126  0.000 -0.148
##  [6,]  0.000 -0.079 -0.089 -0.219 -0.298  1.000 -0.302 -0.087 -0.016 -0.043
##  [7,]  0.000 -0.024 -0.119 -0.179 -0.247 -0.302  1.000 -0.193 -0.128 -0.057
##  [8,] -0.029  0.000  0.000  0.000 -0.126 -0.087 -0.193  1.000 -0.128 -0.201
##  [9,]  0.000  0.000  0.000  0.000  0.000 -0.016 -0.128 -0.128  1.000 -0.085
## [10,] -0.112  0.000  0.000  0.000 -0.148 -0.043 -0.057 -0.201 -0.085  1.000
##
## rho=  1.634
##         [,1]   [,2]   [,3]   [,4]   [,5]   [,6]   [,7]   [,8]   [,9]  [,10]
##  [1,]  1.000  0.000  0.000  0.000 -0.046  0.000  0.000  0.000  0.000  0.000
##  [2,]  0.000  1.000  0.000  0.000 -0.089 -0.025 -0.012  0.000  0.000  0.000
##  [3,]  0.000  0.000  1.000 -0.153  0.000 -0.075 -0.103  0.000  0.000  0.000
##  [4,]  0.000  0.000 -0.153  1.000 -0.248 -0.207 -0.195  0.000  0.000  0.000
##  [5,] -0.046 -0.089  0.000 -0.248  1.000 -0.272 -0.250 -0.145  0.000 -0.140
##  [6,]  0.000 -0.025 -0.075 -0.207 -0.272  1.000 -0.257 -0.088 -0.015 -0.041
##  [7,]  0.000 -0.012 -0.103 -0.195 -0.250 -0.257  1.000 -0.170 -0.107 -0.077
##  [8,]  0.000  0.000  0.000  0.000 -0.145 -0.088 -0.170  1.000 -0.005 -0.069
##  [9,]  0.000  0.000  0.000  0.000  0.000 -0.015 -0.107 -0.005  1.000  0.000
## [10,]  0.000  0.000  0.000  0.000 -0.140 -0.041 -0.077 -0.069  0.000  1.000
##
## rho=  2.451
##       [,1]   [,2]   [,3]   [,4]   [,5]   [,6]   [,7]   [,8]  [,9]  [,10]
##  [1,]    1  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.00  0.000
##  [2,]    0  1.000  0.000  0.000 -0.002  0.000  0.000  0.000  0.00  0.000
##  [3,]    0  0.000  1.000 -0.095  0.000 -0.037 -0.068  0.000  0.00  0.000
##  [4,]    0  0.000 -0.095  1.000 -0.229 -0.187 -0.188  0.000  0.00  0.000
##  [5,]    0 -0.002  0.000 -0.229  1.000 -0.253 -0.247 -0.125  0.00 -0.104
##  [6,]    0  0.000 -0.037 -0.187 -0.253  1.000 -0.236 -0.059  0.00 -0.008
##  [7,]    0  0.000 -0.068 -0.188 -0.247 -0.236  1.000 -0.130 -0.01 -0.049
##  [8,]    0  0.000  0.000  0.000 -0.125 -0.059 -0.130  1.000  0.00  0.000
##  [9,]    0  0.000  0.000  0.000  0.000  0.000 -0.010  0.000  1.00  0.000
## [10,]    0  0.000  0.000  0.000 -0.104 -0.008 -0.049  0.000  0.00  1.000
## 
## rho=  3.269 
##       [,1] [,2]   [,3]   [,4]   [,5]   [,6]   [,7]   [,8] [,9]  [,10]
##  [1,]    1    0  0.000  0.000  0.000  0.000  0.000  0.000    0  0.000
##  [2,]    0    1  0.000  0.000  0.000  0.000  0.000  0.000    0  0.000
##  [3,]    0    0  1.000 -0.033  0.000  0.000 -0.016  0.000    0  0.000
##  [4,]    0    0 -0.033  1.000 -0.201 -0.157 -0.167  0.000    0  0.000
##  [5,]    0    0  0.000 -0.201  1.000 -0.223 -0.232 -0.084    0 -0.037
##  [6,]    0    0  0.000 -0.157 -0.223  1.000 -0.205 -0.017    0  0.000
##  [7,]    0    0 -0.016 -0.167 -0.232 -0.205  1.000 -0.080    0  0.000
##  [8,]    0    0  0.000  0.000 -0.084 -0.017 -0.080  1.000    0  0.000
##  [9,]    0    0  0.000  0.000  0.000  0.000  0.000  0.000    1  0.000
## [10,]    0    0  0.000  0.000 -0.037  0.000  0.000  0.000    0  1.000
## 
## rho=  4.086 
##       [,1] [,2] [,3]   [,4]   [,5]   [,6]   [,7]   [,8] [,9] [,10]
##  [1,]    1    0    0  0.000  0.000  0.000  0.000  0.000    0     0
##  [2,]    0    1    0  0.000  0.000  0.000  0.000  0.000    0     0
##  [3,]    0    0    1  0.000  0.000  0.000  0.000  0.000    0     0
##  [4,]    0    0    0  1.000 -0.166 -0.119 -0.133  0.000    0     0
##  [5,]    0    0    0 -0.166  1.000 -0.184 -0.200 -0.028    0     0
##  [6,]    0    0    0 -0.119 -0.184  1.000 -0.162  0.000    0     0
##  [7,]    0    0    0 -0.133 -0.200 -0.162  1.000 -0.019    0     0
##  [8,]    0    0    0  0.000 -0.028  0.000 -0.019  1.000    0     0
##  [9,]    0    0    0  0.000  0.000  0.000  0.000  0.000    1     0
## [10,]    0    0    0  0.000  0.000  0.000  0.000  0.000    0     1
## 
## rho=  4.903 
##       [,1] [,2] [,3]   [,4]   [,5]   [,6]   [,7] [,8] [,9] [,10]
##  [1,]    1    0    0  0.000  0.000  0.000  0.000    0    0     0
##  [2,]    0    1    0  0.000  0.000  0.000  0.000    0    0     0
##  [3,]    0    0    1  0.000  0.000  0.000  0.000    0    0     0
##  [4,]    0    0    0  1.000 -0.125 -0.075 -0.092    0    0     0
##  [5,]    0    0    0 -0.125  1.000 -0.139 -0.158    0    0     0
##  [6,]    0    0    0 -0.075 -0.139  1.000 -0.116    0    0     0
##  [7,]    0    0    0 -0.092 -0.158 -0.116  1.000    0    0     0
##  [8,]    0    0    0  0.000  0.000  0.000  0.000    1    0     0
##  [9,]    0    0    0  0.000  0.000  0.000  0.000    0    1     0
## [10,]    0    0    0  0.000  0.000  0.000  0.000    0    0     1
## 
## rho=  5.72 
##       [,1] [,2] [,3]   [,4]   [,5]   [,6]   [,7] [,8] [,9] [,10]
##  [1,]    1    0    0  0.000  0.000  0.000  0.000    0    0     0
##  [2,]    0    1    0  0.000  0.000  0.000  0.000    0    0     0
##  [3,]    0    0    1  0.000  0.000  0.000  0.000    0    0     0
##  [4,]    0    0    0  1.000 -0.079 -0.028 -0.046    0    0     0
##  [5,]    0    0    0 -0.079  1.000 -0.090 -0.110    0    0     0
##  [6,]    0    0    0 -0.028 -0.090  1.000 -0.065    0    0     0
##  [7,]    0    0    0 -0.046 -0.110 -0.065  1.000    0    0     0
##  [8,]    0    0    0  0.000  0.000  0.000  0.000    1    0     0
##  [9,]    0    0    0  0.000  0.000  0.000  0.000    0    1     0
## [10,]    0    0    0  0.000  0.000  0.000  0.000    0    0     1
```

```
##
## rho=  6.537
##       [,1] [,2] [,3]   [,4]   [,5]   [,6]   [,7] [,8] [,9] [,10]
##  [1,]    1    0    0  0.000  0.000  0.000  0.000    0    0     0
##  [2,]    0    1    0  0.000  0.000  0.000  0.000    0    0     0
##  [3,]    0    0    1  0.000  0.000  0.000  0.000    0    0     0
##  [4,]    0    0    0  1.000 -0.025  0.000  0.000    0    0     0
##  [5,]    0    0    0 -0.025  1.000 -0.035 -0.056    0    0     0
##  [6,]    0    0    0  0.000 -0.035  1.000 -0.009    0    0     0
##  [7,]    0    0    0  0.000 -0.056 -0.009  1.000    0    0     0
##  [8,]    0    0    0  0.000  0.000  0.000  0.000    1    0     0
##  [9,]    0    0    0  0.000  0.000  0.000  0.000    0    1     0
## [10,]    0    0    0  0.000  0.000  0.000  0.000    0    0     1
##
## rho=  7.354
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    1    0    0    0    0    0    0    0    0     0
##  [2,]    0    1    0    0    0    0    0    0    0     0
##  [3,]    0    0    1    0    0    0    0    0    0     0
##  [4,]    0    0    0    1    0    0    0    0    0     0
##  [5,]    0    0    0    0    1    0    0    0    0     0
##  [6,]    0    0    0    0    0    1    0    0    0     0
##  [7,]    0    0    0    0    0    0    1    0    0     0
##  [8,]    0    0    0    0    0    0    0    1    0     0
##  [9,]    0    0    0    0    0    0    0    0    1     0
## [10,]    0    0    0    0    0    0    0    0    0     1
##
## rho=  8.172
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    1    0    0    0    0    0    0    0    0     0
##  [2,]    0    1    0    0    0    0    0    0    0     0
##  [3,]    0    0    1    0    0    0    0    0    0     0
##  [4,]    0    0    0    1    0    0    0    0    0     0
##  [5,]    0    0    0    0    1    0    0    0    0     0
##  [6,]    0    0    0    0    0    1    0    0    0     0
##  [7,]    0    0    0    0    0    0    1    0    0     0
##  [8,]    0    0    0    0    0    0    0    1    0     0
##  [9,]    0    0    0    0    0    0    0    0    1     0
## [10,]    0    0    0    0    0    0    0    0    0     1
```

Glassopath chooses different regularization parameters on its own.
Once again, around $\rho = 5$ we lose some information about conditional depedence.

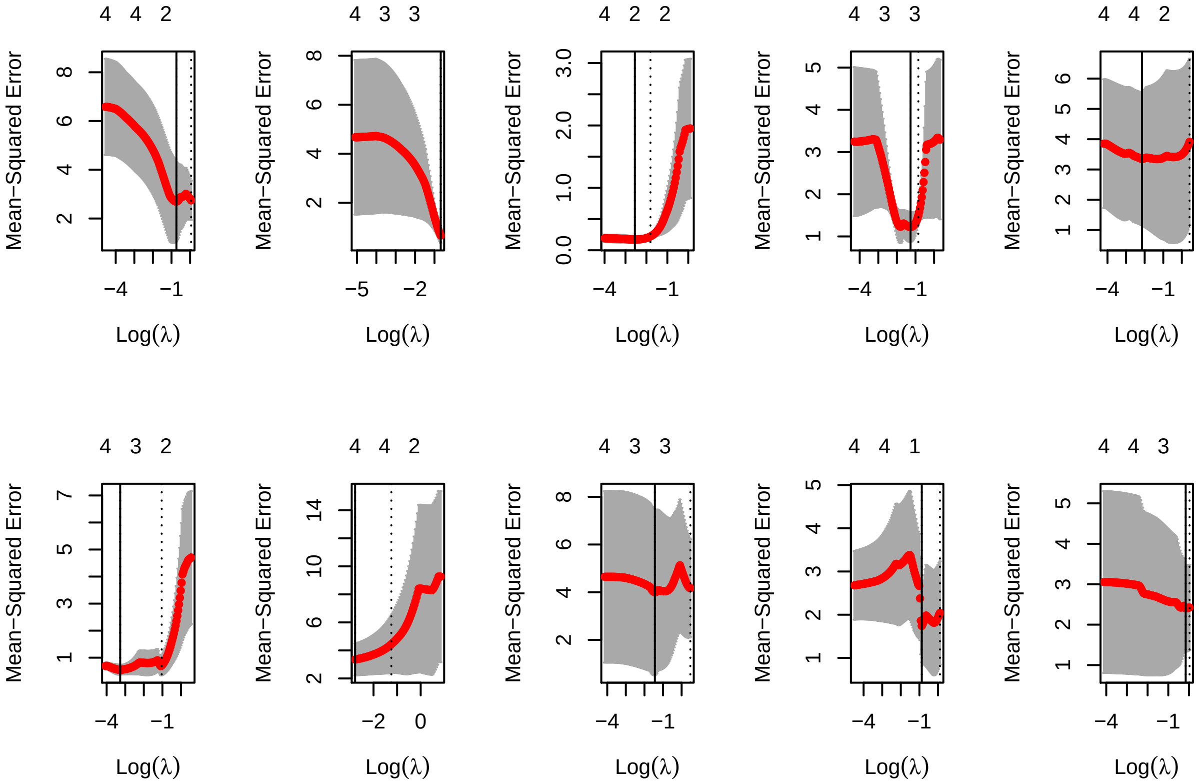
For n = 50, p = 100 I didn't do simulations, because my computer didn't want to cooperate.

```r
# h)
n1 <- 50
p1 <- 100
mat_100 <- mat_min(p1)
A <- mvrnorm(n1, rep(0,p1), mat_100)

S1 <- var(A)
gl1 <- list()
rho <- c(0.1, 1, 5, 10, 100)
```

```r
for(i in 1:5){
  gl1[[i]] <- glasso(S1, rho = rho[i])
}

scaled_precision_matrices1 <- list()

for(j in 1:5){
  scaled_precision_matrices1[[j]] <- cov2cor(gl1[[j]]$wi)
}
```

G-Lasso. Method od Meinshausen-Buhlmann

```r
n = 5
p = 10
Gamma = mat_min(p)
A = mvrnorm(n, rep(0,p), Gamma)
mat_precision <- matrix(0, p, p)

par(mfrow = c(2,5))
for(i in 1:p){
  X = A[,-i]
  Y = A[,i]
  l_min = cv.glmnet(X, Y,alpha=1)$lambda.min
  model = glmnet(X, Y, alpha = 1, lambda = l_min)
  plot(cv.glmnet(X,Y))
  abline(v = log(l_min))
  mat_precision[,i] = as.vector(coef(model))
}
```



Using cross-validation method, I've plotted the best value of λ (vertical line, the one that gives the smallest mean-squared error), which is changing for different p (number of column).

Precision matrix which is build from the coefficients of the models:

```
mat_precision[mat_precision<0.1] = 0
diag(mat_precision) <- 1
round(cov2cor(mat_precision),3)
```

```
##        [,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7]  [,8]  [,9] [,10]
##  [1,] 1.000 0.326 0.000 0.000 0.000 0.254 0.384 0.171 1.229 1.881
##  [2,] 0.919 1.000 0.000 0.000 0.000 0.000 0.186 0.000 0.000 0.000
##  [3,] 0.000 0.000 1.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
##  [4,] 0.000 0.000 0.195 1.000 0.000 1.057 0.000 0.000 0.000 0.000
##  [5,] 0.000 0.000 0.000 0.432 1.000 0.000 0.000 0.000 0.000 0.000
##  [6,] 0.000 0.000 0.450 0.000 0.000 1.000 0.679 0.000 0.000 0.000
##  [7,] 0.000 0.000 0.000 0.000 0.110 0.000 1.000 0.293 0.000 0.000
##  [8,] 0.000 0.000 0.000 0.000 0.000 0.317 0.000 1.000 0.000 0.000
##  [9,] 0.000 0.000 0.000 0.000 0.197 0.000 0.163 0.690 1.000 0.105
## [10,] 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.148 0.000 1.000
```

glasso

```
gl2 <- list()
S <- var(A)
for(i in 1:5){
  gl2[[i]] <- glasso(S, rho = rho[i], approx = TRUE)
}

for(i in 1:5){
  diag(gl2[[i]]$wi) <- 1
  cat('rho=', rho[i], '\n')
  print(round(gl2[[i]]$wi,3))
  cat('\n')
}
```

```
## rho= 0.1
##         [,1]   [,2]  [,3]  [,4]   [,5]  [,6]  [,7]  [,8]   [,9]  [,10]
##  [1,]  1.000  0.265 0.000 0.000  0.000 0.000 0.190 0.000  0.000 -0.592
##  [2,]  0.119  1.000 0.000 0.416  0.000 0.000 0.000 0.000 -0.030  0.000
##  [3,]  0.000  0.000 1.000 0.049  0.000 0.754 0.000 0.000  0.000  0.000
##  [4,]  0.000  0.238 0.154 1.000  0.372 0.000 0.000 0.000  0.000  0.000
##  [5,]  0.000  0.000 0.000 0.601  1.000 0.000 0.717 0.000  0.083  0.000
##  [6,]  0.568  0.000 0.450 0.000 -0.216 1.000 0.553 0.371  0.000  0.748
##  [7,]  0.257  0.000 0.037 0.000  0.550 0.140 1.000 0.000  0.000  0.000
##  [8,]  0.000  0.000 0.000 0.000  0.000 0.325 0.155 1.000  0.580  0.000
##  [9,]  0.000 -0.190 0.000 0.000  0.000 0.000 0.000 0.703  1.000  0.000
## [10,] -1.056  0.000 0.000 0.245  0.000 0.000 0.000 0.193  0.000  1.000
##
## rho= 1
##        [,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7]  [,8]  [,9] [,10]
##  [1,] 1.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
##  [2,] 0.000 1.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
##  [3,] 0.000 0.000 1.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
##  [4,] 0.000 0.000 0.000 1.000 0.000 0.000 0.000 0.000 0.000 0.000
##  [5,] 0.000 0.000 0.000 0.000 1.000 0.000 0.562 0.000 0.000 0.000
##  [6,] 0.000 0.000 0.000 0.000 0.000 1.000 0.642 0.165 0.000 0.000
##  [7,] 0.199 0.014 0.305 0.413 0.486 0.545 1.000 0.354 0.188 0.000
##  [8,] 0.000 0.000 0.000 0.000 0.000 0.000 0.000 1.000 0.137 0.329
##  [9,] 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 1.000 0.000
## [10,] 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 1.000
##
## rho= 5
##       [,1] [,2] [,3] [,4] [,5]  [,6]  [,7] [,8] [,9] [,10]
##  [1,]    1    0    0    0    0 0.000 0.000    0    0     0
##  [2,]    0    1    0    0    0 0.000 0.000    0    0     0
##  [3,]    0    0    1    0    0 0.000 0.000    0    0     0
##  [4,]    0    0    0    1    0 0.000 0.000    0    0     0
##  [5,]    0    0    0    0    1 0.000 0.000    0    0     0
##  [6,]    0    0    0    0    0 1.000 0.014    0    0     0
##  [7,]    0    0    0    0    0 0.007 1.000    0    0     0
##  [8,]    0    0    0    0    0 0.000 0.000    1    0     0
##  [9,]    0    0    0    0    0 0.000 0.000    0    1     0
## [10,]    0    0    0    0    0 0.000 0.000    0    0     1
##
## rho= 10
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    1    0    0    0    0    0    0    0    0     0
##  [2,]    0    1    0    0    0    0    0    0    0     0
##  [3,]    0    0    1    0    0    0    0    0    0     0
##  [4,]    0    0    0    1    0    0    0    0    0     0
##  [5,]    0    0    0    0    1    0    0    0    0     0
##  [6,]    0    0    0    0    0    1    0    0    0     0
##  [7,]    0    0    0    0    0    0    1    0    0     0
##  [8,]    0    0    0    0    0    0    0    1    0     0
##  [9,]    0    0    0    0    0    0    0    0    1     0
## [10,]    0    0    0    0    0    0    0    0    0     1
##
## rho= 100
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    1    0    0    0    0    0    0    0    0     0
##  [2,]    0    1    0    0    0    0    0    0    0     0
##  [3,]    0    0    1    0    0    0    0    0    0     0
##  [4,]    0    0    0    1    0    0    0    0    0     0
##  [5,]    0    0    0    0    1    0    0    0    0     0
##  [6,]    0    0    0    0    0    1    0    0    0     0
##  [7,]    0    0    0    0    0    0    1    0    0     0
##  [8,]    0    0    0    0    0    0    0    1    0     0
##  [9,]    0    0    0    0    0    0    0    0    1     0
## [10,]    0    0    0    0    0    0    0    0    0     1
```

The results are similar: for $\rho \geq 5$ there appear more zeros in the matrix.

Graphical Model Selection for Frets' Heads Data

The frets data frame has 25 rows and 4 columns. It consist of measurements of the length and breadth of the heads of pairs of adult brothers in 25 randomly sampled families. All measurements are expressed in millimetres. l1 - The head length of the eldest son. b1 - The head breadth of the eldest son. l2 - The head length of the second son. b2 - The head breadth of the second son.

```
library(boot)
head(frets)
```

```
##    l1  b1  l2  b2
## 1 191 155 179 145
## 2 195 149 201 152
## 3 181 148 185 149
## 4 183 153 188 149
## 5 176 144 171 142
## 6 208 157 192 152
```

Scaled precision matrix:

```
frets_var <- var(frets)
scaled_prec_mat <- round(cov2cor(solve(frets_var)),3)
scaled_prec_mat
```

```
##        l1     b1     l2     b2
## l1  1.000 -0.425 -0.223 -0.152
## b1 -0.425  1.000 -0.132 -0.225
## l2 -0.223 -0.132  1.000 -0.626
## b2 -0.152 -0.225 -0.626  1.000
```

It seems that every variable is connected with each other.

Method of Friedman

```
gl_frets <- list()
rho <- c(0.1, 1, 10, 25, 50, 60, 75, 100)

for(i in 1:length(rho)){
  gl_frets[[i]] <- glasso(frets_var, rho = rho[i])
}

scaled_precision_matrices_frets <- list()
for(j in 1:length(rho)){
  scaled_precision_matrices_frets[[j]] <- round(cov2cor(gl_frets[[j]]$wi),3)
}
```

Scaled precision matrices:

```
for(i in 1:length(rho)){
  cat('rho = ', rho[i], '\n')
  scaled_precision_matrices_frets[[i]][abs(scaled_precision_matrices_frets[[i]]) < 0.01] <- 0
  print(scaled_precision_matrices_frets[[i]])
  cat('\n')
}
```

```
## rho =  0.1
##        [,1]   [,2]   [,3]   [,4]
## [1,]  1.000 -0.424 -0.224 -0.153
## [2,] -0.424  1.000 -0.135 -0.223
## [3,] -0.224 -0.135  1.000 -0.622
## [4,] -0.153 -0.223 -0.622  1.000
##
## rho =  1
##        [,1]   [,2]   [,3]   [,4]
## [1,]  1.000 -0.411 -0.238 -0.159
## [2,] -0.411  1.000 -0.154 -0.206
## [3,] -0.238 -0.154  1.000 -0.591
## [4,] -0.159 -0.206 -0.591  1.000
##
## rho =  10
##        [,1]   [,2]   [,3]   [,4]
## [1,]  1.000 -0.317 -0.289 -0.166
## [2,] -0.317  1.000 -0.205 -0.120
## [3,] -0.289 -0.205  1.000 -0.418
## [4,] -0.166 -0.120 -0.418  1.000
##
## rho =  25
##        [,1]   [,2]   [,3]   [,4]
## [1,]  1.000 -0.207 -0.267 -0.115
## [2,] -0.207  1.000 -0.165 -0.026
## [3,] -0.267 -0.165  1.000 -0.270
## [4,] -0.115 -0.026 -0.270  1.000
##
## rho =  50
##        [,1]   [,2]   [,3]   [,4]
## [1,]  1.000 -0.022 -0.132  0.000
## [2,] -0.022  1.000  0.000  0.000
## [3,] -0.132  0.000  1.000 -0.054
## [4,]  0.000  0.000 -0.054  1.000
##
## rho =  60
##        [,1] [,2]   [,3] [,4]
## [1,]  1.000    0 -0.061    0
## [2,]  0.000    1  0.000    0
## [3,] -0.061     0  1.000     0
## [4,]  0.000     0  0.000     1
##
## rho =  75
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    1    0    0
## [3,]    0    0    1    0
## [4,]    0    0    0    1
##
## rho =  100
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    1    0    0
## [3,]    0    0    1    0
## [4,]    0    0    0    1
```

```
# g)
gl_path_frets <- glassopath(frets_var)
```

glassopath

```
for(i in 1:length(gl_path_frets$rholist)){
  cat('rho= ', round(gl_path_frets$rholist[i],3), '\n')
  print(round(cov2cor(gl_path_frets$wi[,,i]),3))
  cat('\n')
}
```

```
## rho=  10.081
##        [,1]   [,2]   [,3]   [,4]
## [1,]  1.000 -0.317 -0.289 -0.166
## [2,] -0.317  1.000 -0.205 -0.120
## [3,] -0.289 -0.205  1.000 -0.417
## [4,] -0.166 -0.120 -0.417  1.000
##
## rho=  20.161
##        [,1]   [,2]   [,3]   [,4]
## [1,]  1.000 -0.241 -0.281 -0.136
## [2,] -0.241  1.000 -0.184 -0.056
## [3,] -0.281 -0.184  1.000 -0.312
## [4,] -0.136 -0.056 -0.312  1.000
##
## rho=  30.242
##        [,1]   [,2]   [,3]   [,4]
## [1,]  1.000 -0.169 -0.247 -0.088
## [2,] -0.169  1.000 -0.139  0.000
## [3,] -0.247 -0.139  1.000 -0.226
## [4,] -0.088  0.000 -0.226  1.000
##
## rho=  40.323
##        [,1]   [,2]   [,3]   [,4]
## [1,]  1.000 -0.093 -0.198 -0.023
## [2,] -0.093  1.000 -0.073  0.000
## [3,] -0.198 -0.073  1.000 -0.139
## [4,] -0.023  0.000 -0.139  1.000
```

```
##
## rho=  50.403
##        [,1]   [,2]   [,3]   [,4]
## [1,]  1.000 -0.019 -0.129  0.000
## [2,] -0.019  1.000 -0.005  0.000
## [3,] -0.129 -0.005  1.000 -0.051
## [4,]  0.000  0.000 -0.051  1.000
##
## rho=  60.484
##        [,1] [,2]   [,3] [,4]
## [1,]  1.000    0 -0.058    0
## [2,]  0.000    1  0.000    0
## [3,] -0.058    0  1.000    0
## [4,]  0.000    0  0.000    1
##
## rho=  70.565
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    1    0    0
## [3,]    0    0    1    0
## [4,]    0    0    0    1
##
## rho=  80.645
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    1    0    0
## [3,]    0    0    1    0
## [4,]    0    0    0    1
##
## rho=  90.726
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    1    0    0
## [3,]    0    0    1    0
## [4,]    0    0    0    1
##
## rho=  100.807
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    1    0    0
## [3,]    0    0    1    0
## [4,]    0    0    0    1
```

Both *glasso* and *glassopath* give similar results. When ρ is around 70, all connections between the variables disappear. For example, for $\rho = 50$ we have graph: b2-l2-l1-b1.

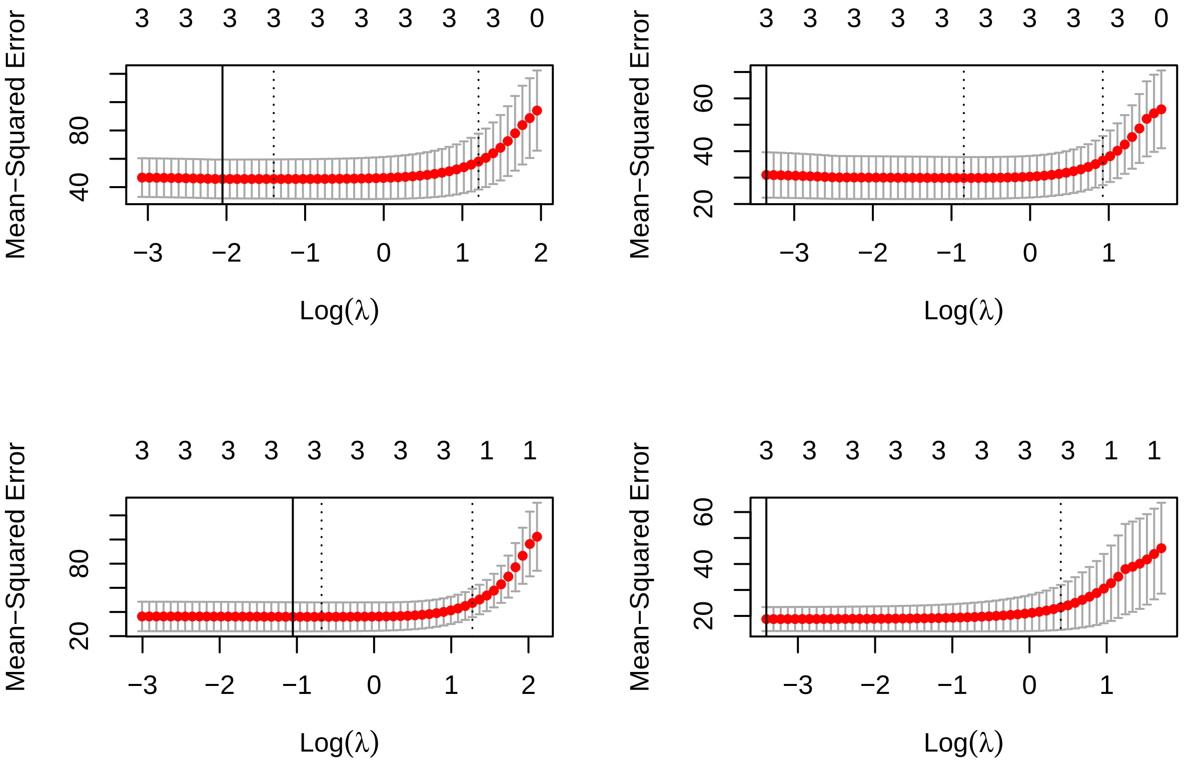
```
rho <- 0
gl_0 <- glasso(frets_var, rho = rho)
scaled_precision_matrix_0 <- round(cov2cor(gl_0$wi),3)
scaled_precision_matrix_0
```

```
##        [,1]   [,2]   [,3]   [,4]
## [1,]  1.000 -0.425 -0.223 -0.152
## [2,] -0.425  1.000 -0.132 -0.225
## [3,] -0.223 -0.132  1.000 -0.626
## [4,] -0.152 -0.225 -0.626  1.000
```

When $\rho = 0$, we have exactly scaled precision matrix of Frets' Head Data.

Meinshausen-Buhlman Method

```
p = 4
mat_precision_frets <- matrix(0, p, p)
par(mfrow = c(2,2))
A <- frets
for(i in 1:p){
  X = as.matrix(A[,-i])
  Y = as.vector(A[,i])
  l_min = cv.glmnet(X, Y,alpha=1)$lambda.min
  model = glmnet(X, Y, alpha = 1, lambda = l_min)
  plot(cv.glmnet(X,Y))
  abline(v = log(l_min))
  mat_precision_frets[,i] = as.vector(coef(model))
}
```



Similar plots, λ parameter is changing (vertical line), depending on the value of p.

```
scaled_mat_precision_frets <- cov2cor(mat_precision_frets)
scaled_mat_precision_frets[scaled_mat_precision_frets < 0.01] = 0
round(scaled_mat_precision_frets,3)
```

```
##       [,1]   [,2]  [,3]   [,4]
## [1,] 1.000 10.732 0.000 11.926
## [2,] 0.230  1.000 0.866  0.235
## [3,] 0.168  0.565 1.000  0.718
## [4,] 0.096  0.815 3.971  1.000
```

This time, connection between l1 and l2 disappears.