T-Cell Network

Example for GeneNet 1.2.7 (June 2013) or later

This note reproduces the "T-Cell" network example from R. Opgen-Rhein and K. Strimmer. 2006a. *Using regularized dynamic correlation to infer gene dependency networks from time-series microarray data.* Proceedings of WCSB 2006 (June 12-13, 2006, Tampere, Finland) and R. Opgen-Rhein and K. Strimmer. 2006b. *Inferring gene dependency networks from genomic longitudinal data: a functional data approach.* REVSTAT **4**:53-65. (http://www.ine.pt/revstat/pdf/rs060103.pdf)

Load GeneNet library

```
library("GeneNet")
```

```
## Loading required package: corpcor
## Loading required package: longitudinal
## Loading required package: fdrtool
## Loading required package: igraph
```

get T cell data

```
data(tcell)
tc44 = combine.longitudinal(tcell.10, tcell.34)
```

Estimate partial correlations

```
pc1 = ggm.estimate.pcor(tc44, lambda=0)                    # static, no shrinkage
```

```
## Specified shrinkage intensity lambda (correlation matrix): 0
```

```
pc2 = ggm.estimate.pcor(tc44, method="dynamic", lambda=0) # dynamic, no shrinkage
```

```
## Specified shrinkage intensity lambda (correlation matrix): 0
```

```
pc3 = ggm.estimate.pcor(tc44)                              # static, with shrinkage
```

```
## Estimating optimal shrinkage intensity lambda (correlation matrix): 0.0209
```

```
pc4 = ggm.estimate.pcor(tc44, method="dynamic")            # dynamic, with shrinkage
```

```
## Estimating optimal shrinkage intensity lambda (correlation matrix): 0.032
```

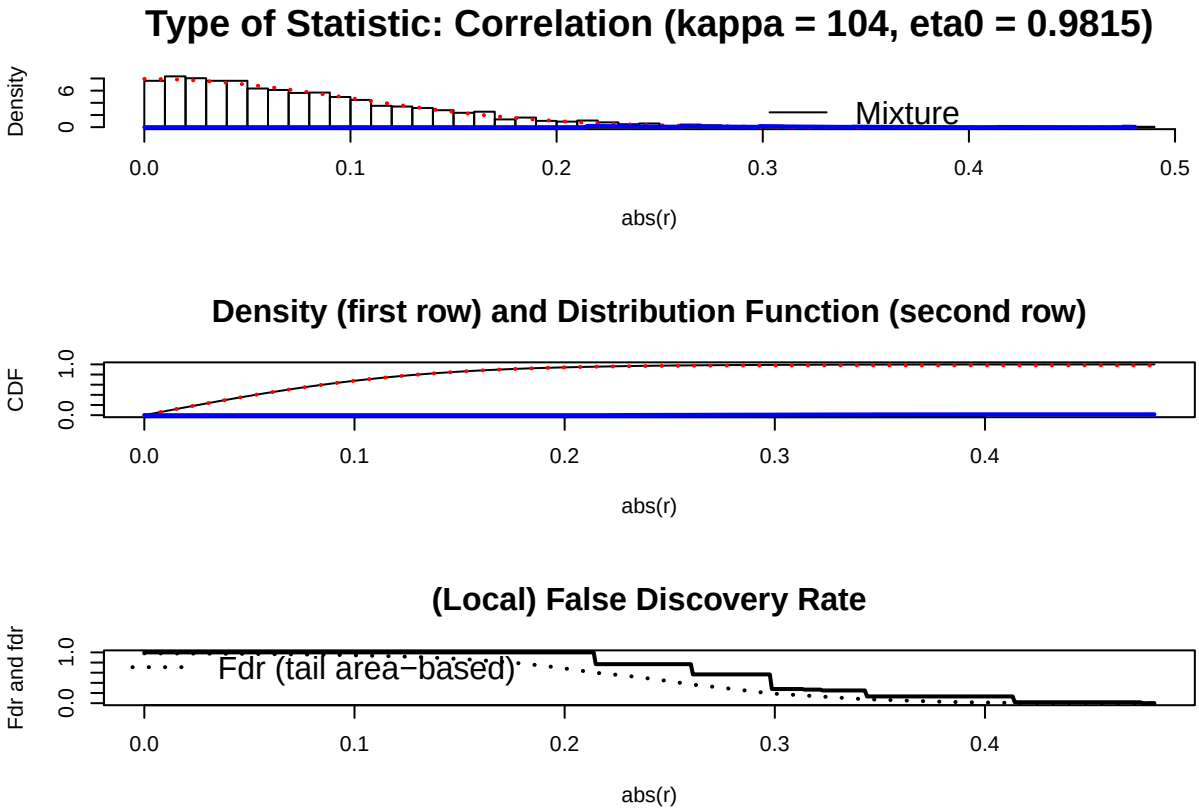
Find significant edges

We use as selection criterion local fdr <= 0.2

static, no shrinkage

```
t1.edges = ggm.test.edges(pc1)
```

```
## Estimate (local) false discovery rates (partial correlations):
## Step 1... determine cutoff point
## Step 2... estimate parameters of null distribution and eta0
## Step 3... compute p-values and estimate empirical PDF/CDF
## Step 4... compute q-values and local fdr
## Step 5... prepare for plotting
```




```
t1.net = extract.network(t1.edges) # prob > 0.8
```

```
##
## Significant edges:  6
##     Corresponding to  0.36 %  of possible edges
```

```
t1.net
```

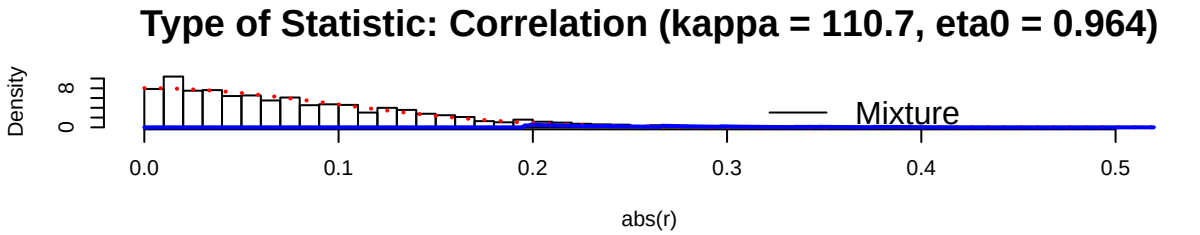
```
##         pcor node1 node2         pval         qval      prob
## 1  0.4806257    11    28 2.124826e-07 0.0002588653 0.9997411
## 2  0.4744295    40    56 3.191076e-07 0.0002588653 0.9821251
```

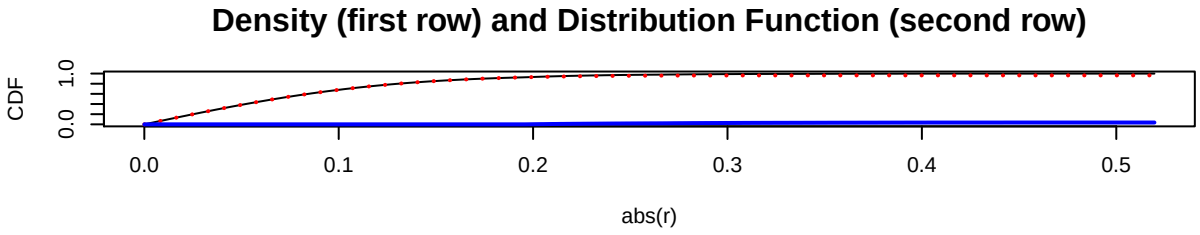
```
## 3 -0.4140609    31    45 1.133647e-05 0.0061308857 0.8673551
## 4  0.3635057    33    48 1.385462e-04 0.0493380622 0.8673551
## 5  0.3538857    23    28 2.132978e-04 0.0632622165 0.8673551
## 6  0.3444279    46    49 3.217399e-04 0.0768025473 0.8673551
```

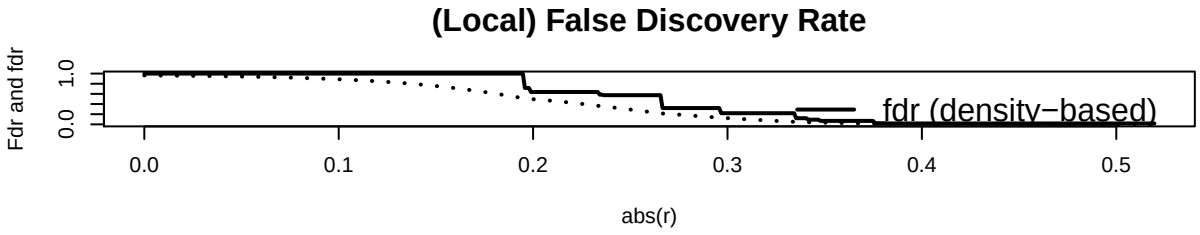
dynamic, no shrinkage

```
t2.edges = ggm.test.edges(pc2)
```

```
## Estimate (local) false discovery rates (partial correlations):
## Step 1... determine cutoff point
## Step 2... estimate parameters of null distribution and eta0
## Step 3... compute p-values and estimate empirical PDF/CDF
## Step 4... compute q-values and local fdr
## Step 5... prepare for plotting
```








```
t2.net = extract.network(t2.edges) # prob > 0.8
```

```
##
## Significant edges:  9
##     Corresponding to  0.54 %  of possible edges
```

```
t2.net
```

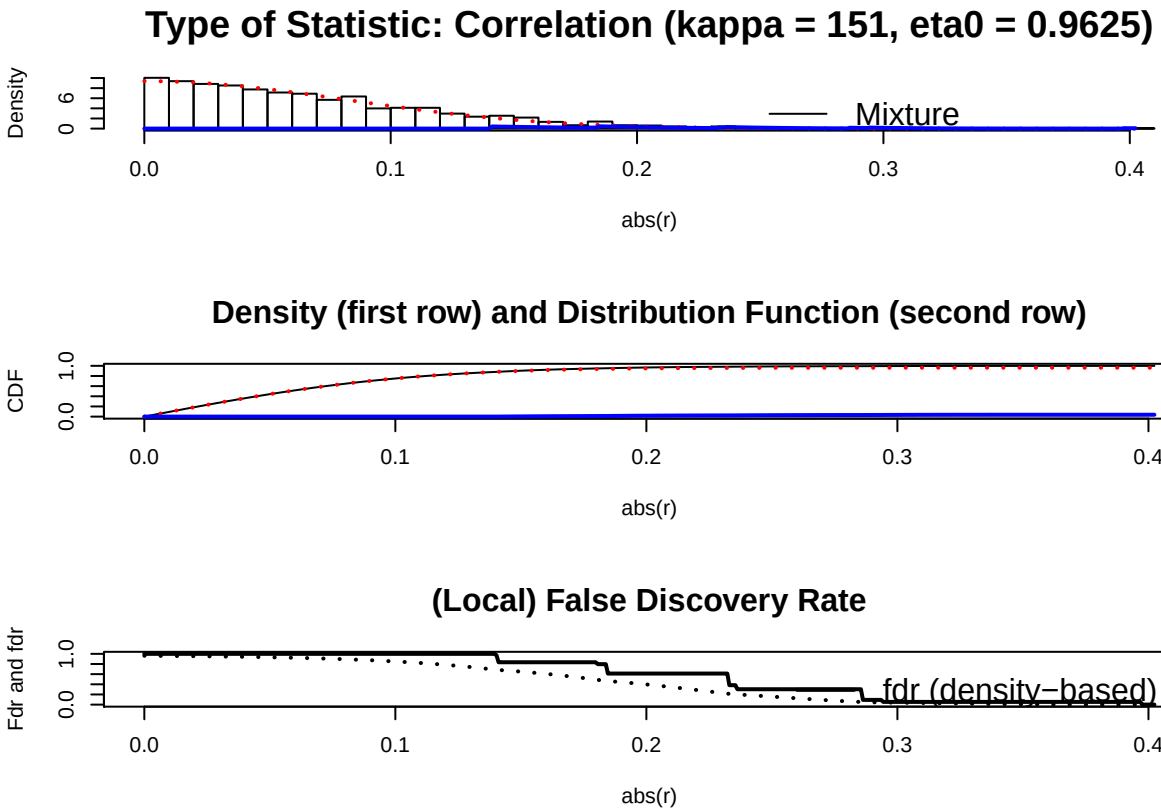
```
##         pcor node1 node2         pval         qval      prob
```

```
## 1  0.5196235    11    28 4.547772e-09 7.246755e-06 0.9821314
## 2  0.3971802    31    33 1.490293e-05 1.019816e-02 0.9821314
## 3  0.3888354    40    56 2.324970e-05 1.205612e-02 0.9821314
## 4  0.3817250    18    44 3.364533e-05 1.340324e-02 0.9778560
## 5  0.3749210    23    28 4.754204e-05 1.515140e-02 0.9318103
## 6 -0.3543569     5    26 1.291406e-04 2.979406e-02 0.9318103
## 7  0.3503019    23    32 1.560326e-04 3.299615e-02 0.9318103
## 8  0.3477015    46    49 1.759213e-04 3.504075e-02 0.9079267
## 9  0.3414539    30    40 2.337028e-04 4.137770e-02 0.8789744
```

static, with shrinkage

```
t3.edges = ggm.test.edges(pc3)
```

```
## Estimate (local) false discovery rates (partial correlations):
## Step 1... determine cutoff point
## Step 2... estimate parameters of null distribution and eta0
## Step 3... compute p-values and estimate empirical PDF/CDF
## Step 4... compute q-values and local fdr
## Step 5... prepare for plotting
```




```
t3.net = extract.network(t3.edges) # prob > 0.8
```

```
##
## Significant edges:  10
##     Corresponding to  0.6 %  of possible edges
```

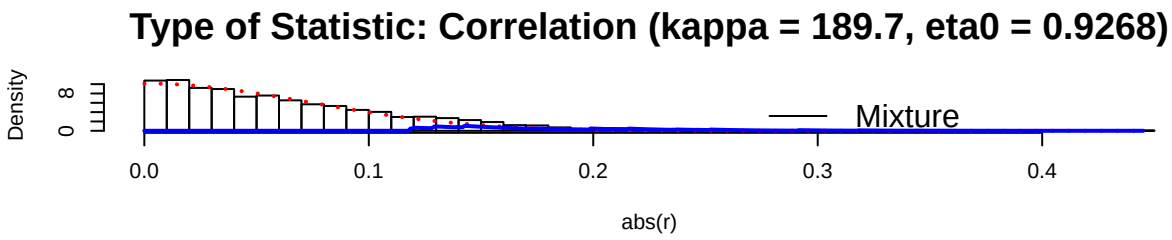
```
t3.net
```

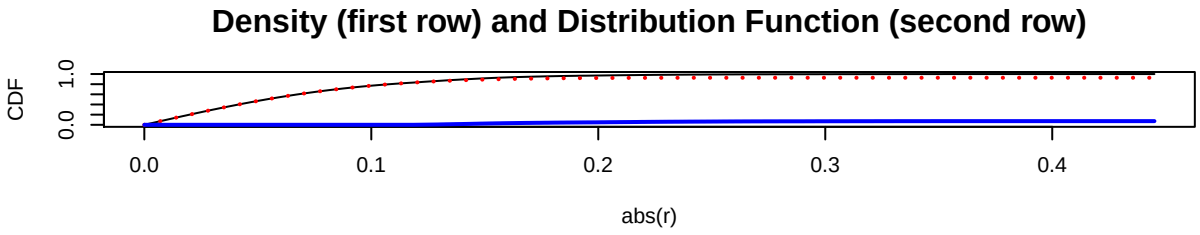
```
##          pcor node1 node2         pval         qval      prob
## 1   0.4023842    40    56 2.762444e-07 0.0003307888 0.9996692
## 2   0.3968325    11    28 4.158062e-07 0.0003307888 0.9463881
## 3   0.3164111    46    49 7.146148e-05 0.0276746523 0.9463881
## 4   0.3128258    23    28 8.710848e-05 0.0303085469 0.9463881
## 5   0.3093519    33    48 1.052786e-04 0.0327666910 0.9463881
## 6  -0.2961398    31    45 2.118707e-04 0.0407350046 0.9463881
## 7   0.2959314    31    33 2.141644e-04 0.0408400640 0.9463881
## 8   0.2954847     6    45 2.191583e-04 0.0410629683 0.9463881
## 9   0.2940215    29    36 2.362839e-04 0.0417716291 0.9070067
## 10  0.2892489     5    30 3.011663e-04 0.0473958579 0.9070067
```

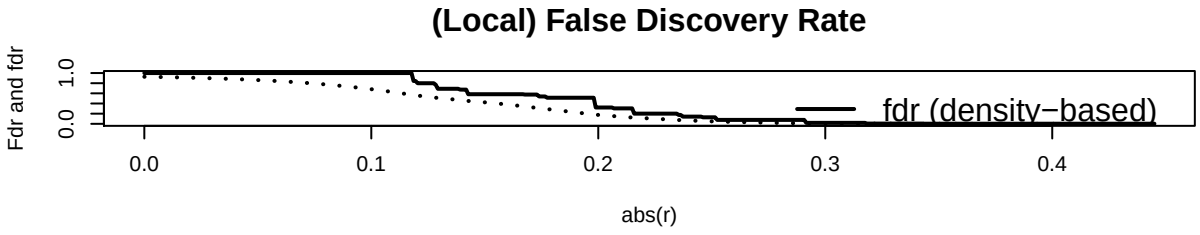
dynamic, with shrinkage

```
t4.edges = ggm.test.edges(pc4)
```

```
## Estimate (local) false discovery rates (partial correlations):
## Step 1... determine cutoff point
## Step 2... estimate parameters of null distribution and eta0
## Step 3... compute p-values and estimate empirical PDF/CDF
## Step 4... compute q-values and local fdr
## Step 5... prepare for plotting
```







```
t4.net = extract.network(t4.edges) # prob > 0.8
```

```
##
## Significant edges:  21
##     Corresponding to  1.27 %  of possible edges
```

```
t4.net
```

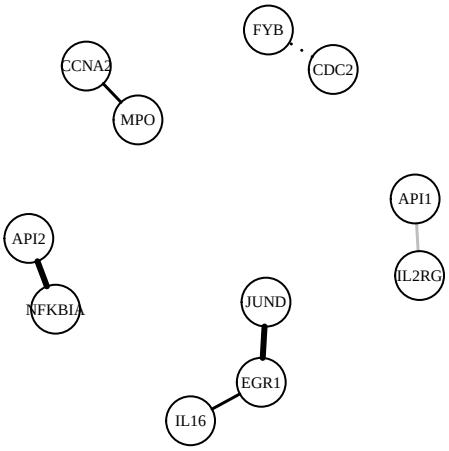
```
##          pcor node1 node2         pval         qval      prob
## 1   0.4450243    11    28 1.151954e-10 1.764880e-07 0.9980860
## 2   0.3425852    18    44 1.249379e-06 9.570710e-04 0.9951212
## 3   0.3217422    23    28 5.750408e-06 2.580987e-03 0.9951212
## 4   0.3177284    40    56 7.618262e-06 2.917936e-03 0.9811075
## 5   0.2987142    23    32 2.735755e-05 7.483611e-03 0.9811075
## 6   0.2923531    46    49 4.114877e-05 9.382601e-03 0.9811075
## 7   0.2910758    31    33 4.461217e-05 9.764167e-03 0.9231219
## 8   0.2784745    37    51 9.700874e-05 1.847602e-02 0.9231219
## 9   0.2670698     5    30 1.898769e-04 2.939947e-02 0.9231219
## 10 -0.2660534    32    41 2.013014e-04 3.046735e-02 0.9231219
## 11  0.2625146     4    55 2.462677e-04 3.424178e-02 0.9231219
## 12  0.2555715    46    57 3.628202e-04 4.166472e-02 0.9231219
## 13 -0.2553619     5    26 3.670299e-04 4.188476e-02 0.9231219
## 14  0.2528018    30    40 4.222203e-04 4.453450e-02 0.9231219
## 15  0.2517914    25    42 4.460449e-04 4.555827e-02 0.8718166
## 16  0.2462722     3    16 5.996532e-04 5.456852e-02 0.8718166
## 17  0.2458450    10    16 6.133780e-04 5.527887e-02 0.8581086
## 18  0.2386917    38    58 8.907959e-04 6.825390e-02 0.8581086
## 19  0.2382936     6    45 9.091928e-04 6.897824e-02 0.8581086
## 20  0.2381243    49    56 9.171238e-04 6.928613e-02 0.8581086
## 21 -0.2367511    40    45 9.838335e-04 7.177649e-02 0.8249718
```

Produce plots using igraph

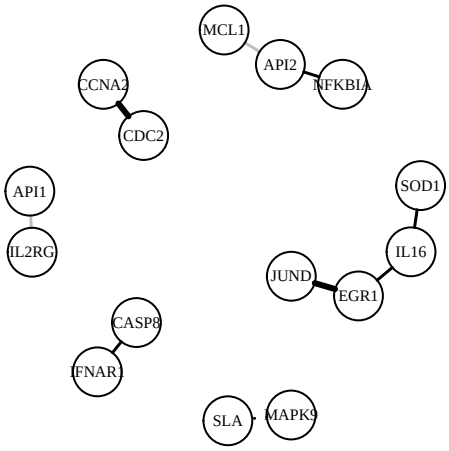
```
node.labels = colnames(tc44)
igr1 = ggm.make.igraph( t1.net, node.labels)
igr2 = ggm.make.igraph( t2.net, node.labels)
igr3 = ggm.make.igraph( t3.net, node.labels)
igr4 = ggm.make.igraph( t4.net, node.labels)
```

```
par(mfrow=c(2,2))
plot(igr1, main="Static, no shrinkage", vertex.label.cex=.6, vertex.size=25)
plot(igr2, main="Dynamic, no shrinkage", vertex.label.cex=.6, vertex.size=25)
plot(igr3, main="Static, with shrinkage", vertex.label.cex=.6, vertex.size=25)
plot(igr4, main="Dynamic, with shrinkage", vertex.label.cex=.6, vertex.size=25)
```

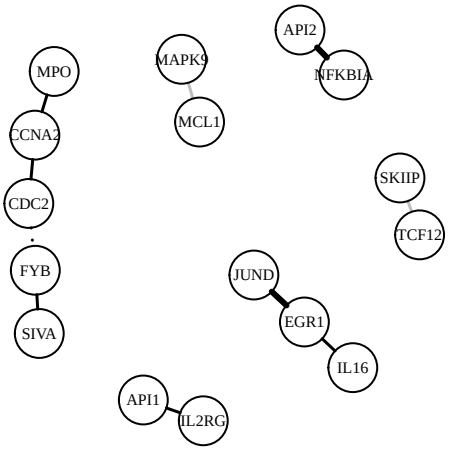
Static, no shrinkage



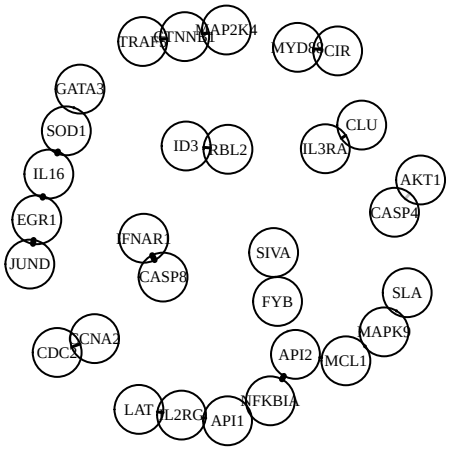
Dynamic, no shrinkage



Static, with shrinkage



Dynamic, with shrinkage




```
par(mfrow=c(1,1))
```