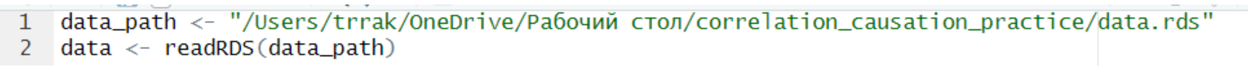
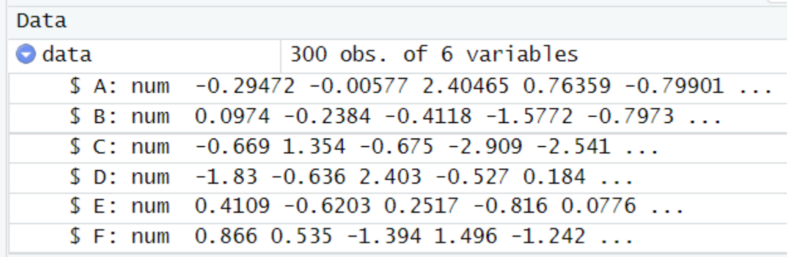
Rakhmanov Timur gr. 72-62

Problem 1: Testing for marginal correlation

Read our data:

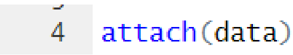


```
data_path <- "/Users/trrak/OneDrive/Рабочий стол/correlation_causation_practice/data.rds"
data <- readRDS(data_path)
```



```
Data
data                  300 obs. of 6 variables
    $ A: num  -0.29472 -0.00577 2.40465 0.76359 -0.79901 ...
    $ B: num  0.0974 -0.2384 -0.4118 -1.5772 -0.7973 ...
    $ C: num  -0.669 1.354 -0.675 -2.909 -2.541 ...
    $ D: num  -1.83 -0.636 2.403 -0.527 0.184 ...
    $ E: num  0.4109 -0.6203 0.2517 -0.816 0.0776 ...
    $ F: num  0.866 0.535 -1.394 1.496 -1.242 ...
```

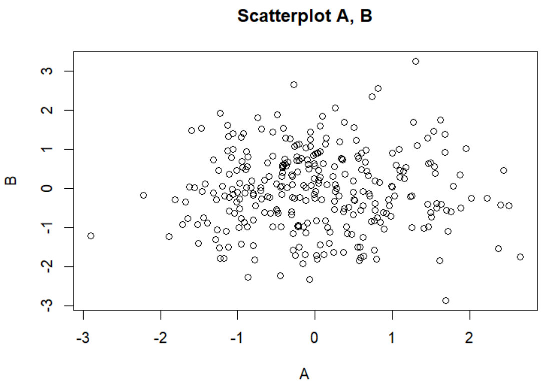
I can simplify the code using code below (A instead of data$A):



```
attach(data)
```

Make a scatterplot of A and B (x and y respectively):

```
plot(A, B, main="Scatterplot A, B", xlab="A", ylab="B")
```



I see this plot doesn’t have any correlation.

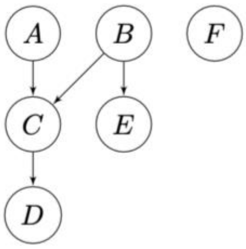
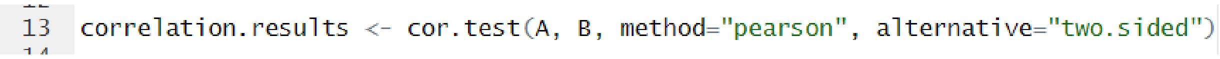


Figure 1

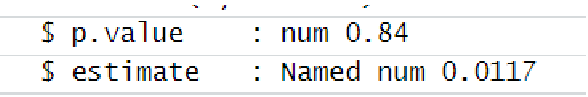
This plot gives a right information, because figure 1 has independent A and B.

Also I can obtain correlation results:



```
correlation.results <- cor.test(A, B, method="pearson", alternative="two.sided")
```

alternative=”two.sided” – it means that value measures between -1 and 1, where 0 – no correlation.



```
 $ p.value    : num 0.84
 $ estimate   : Named num 0.0117
```

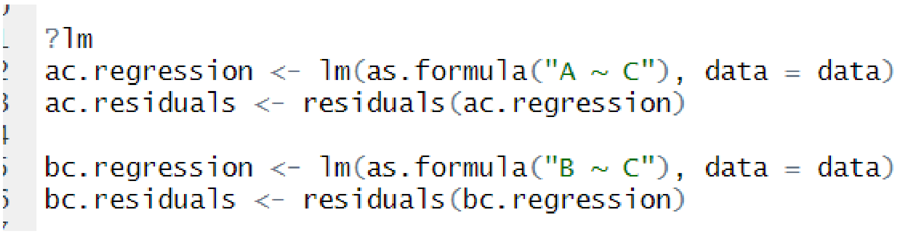
I obtained that correlation really close to zero and high p-value, that’s why correlation no here.

Problem 2: Testing for partial correlation

I tested marginal correlation, but also in figure 1 there is a partial correlation, that makes sense in another case, for instance, A, B and C.

The main idea to test partial correlation – linearly regress A on C, compute and store the residuals, make correlation between residuals (B and C is the same way).

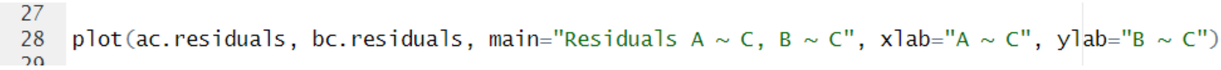
Just use linear model (lm):



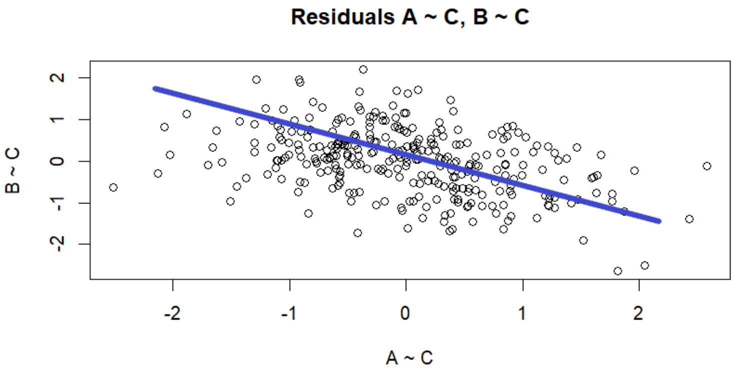
```
?lm
ac.regression <- lm(as.formula("A ~ C"), data = data)
ac.residuals <- residuals(ac.regression)

bc.regression <- lm(as.formula("B ~ C"), data = data)
bc.residuals <- residuals(bc.regression)
```

Then I can make a plot:

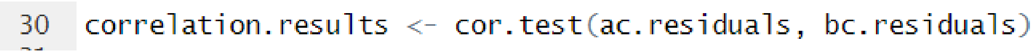


```
plot(ac.residuals, bc.residuals, main="Residuals A ~ C, B ~ C", xlab="A ~ C", ylab="B ~ C")
```

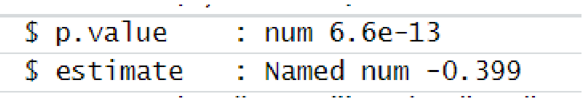


So, there is a negative correlation.

Also I can obtain correlation results:



```
correlation.results <- cor.test(ac.residuals, bc.residuals)
```



```
 $ p.value    : num 6.6e-13
 $ estimate   : Named num -0.399
```

P-value is very low and coefficient of correlation is -0.399, so our plot showed this correlation.

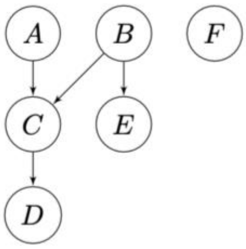


Figure 1

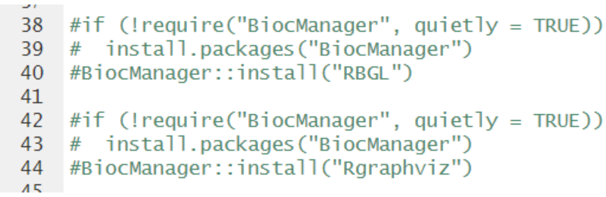
Using Figure 1 I can see that C makes dependence between A and B.

Problem 3: Running the PC algorithm

Next step – try to create a graphs. I tried several times to load library pcalg, the best way to use it:

Update my R version from 4.2.2 to 4.2.3

Install several packages:



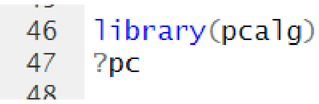
```
#if (!require("BiocManager", quietly = TRUE))
#  install.packages("BiocManager")
#BiocManager::install("RBGL")

#if (!require("BiocManager", quietly = TRUE))
#  install.packages("BiocManager")
#BiocManager::install("Rgraphviz")
```

https://www.bioconductor.org/install/

I commented it, because after run, these codes will disappear (idk why, mb it's bag).

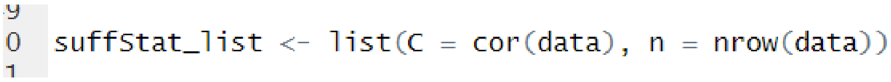
Then we can upload library and check documentation:



```
library(pcalg)
?pc
```

suffStat - a list of sufficient statistics.

Let's create suffStat list:



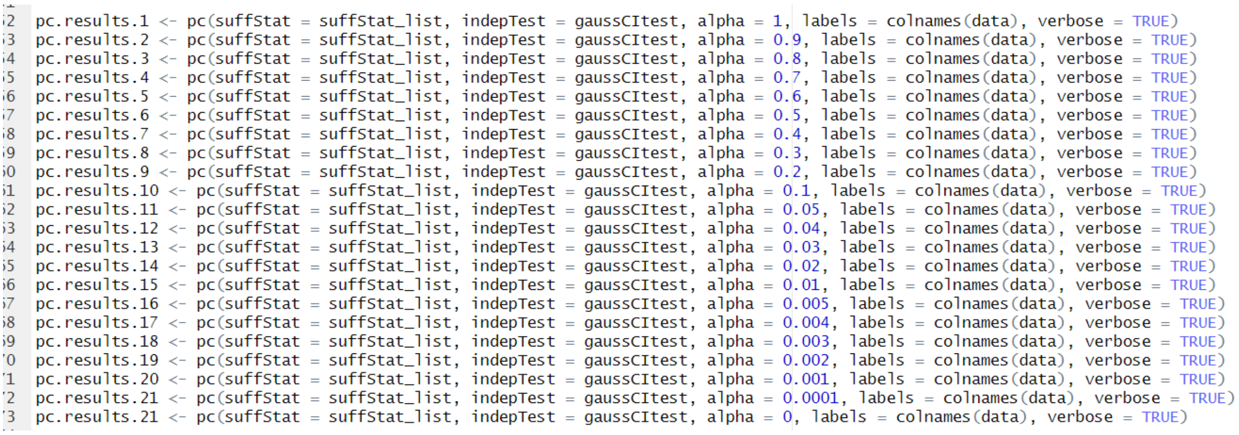
```
suffStat_list <- list(C = cor(data), n = nrow(data))
```

There are two elements – C and n. C – correlation matrix, n – number of rows.

I supplied these as a list for the suffStat argument of the function pc().

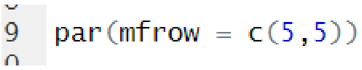
I specified indepTest = gaussCItest, and set a reasonable significance level alpha for the independence tests.

Alpha – is the equivalent of p-value. Alpha - significance level (number in (0,1)) for the individual conditional independence tests.



```
pc.results.1 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 1, labels = colnames(data), verbose = TRUE)
pc.results.2 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.9, labels = colnames(data), verbose = TRUE)
pc.results.3 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.8, labels = colnames(data), verbose = TRUE)
pc.results.4 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.7, labels = colnames(data), verbose = TRUE)
pc.results.5 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.6, labels = colnames(data), verbose = TRUE)
pc.results.6 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.5, labels = colnames(data), verbose = TRUE)
pc.results.7 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.4, labels = colnames(data), verbose = TRUE)
pc.results.8 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.3, labels = colnames(data), verbose = TRUE)
pc.results.9 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.2, labels = colnames(data), verbose = TRUE)
pc.results.10 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.1, labels = colnames(data), verbose = TRUE)
pc.results.11 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.05, labels = colnames(data), verbose = TRUE)
pc.results.12 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.04, labels = colnames(data), verbose = TRUE)
pc.results.13 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.03, labels = colnames(data), verbose = TRUE)
pc.results.14 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.02, labels = colnames(data), verbose = TRUE)
pc.results.15 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.01, labels = colnames(data), verbose = TRUE)
pc.results.16 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.005, labels = colnames(data), verbose = TRUE)
pc.results.17 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.004, labels = colnames(data), verbose = TRUE)
pc.results.18 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.003, labels = colnames(data), verbose = TRUE)
pc.results.19 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.002, labels = colnames(data), verbose = TRUE)
pc.results.20 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.001, labels = colnames(data), verbose = TRUE)
pc.results.21 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0.0001, labels = colnames(data), verbose = TRUE)
pc.results.21 <- pc(suffStat = suffStat_list, indepTest = gaussCItest, alpha = 0, labels = colnames(data), verbose = TRUE)
```

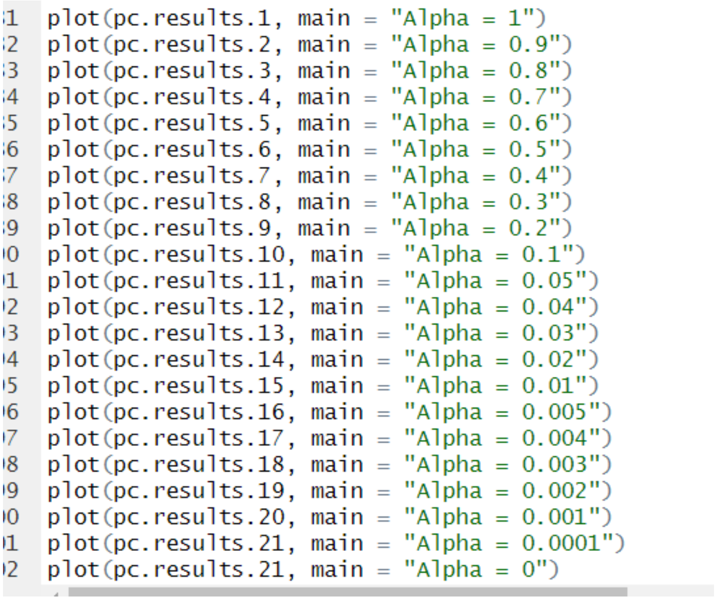
Then we set graphical parameters:



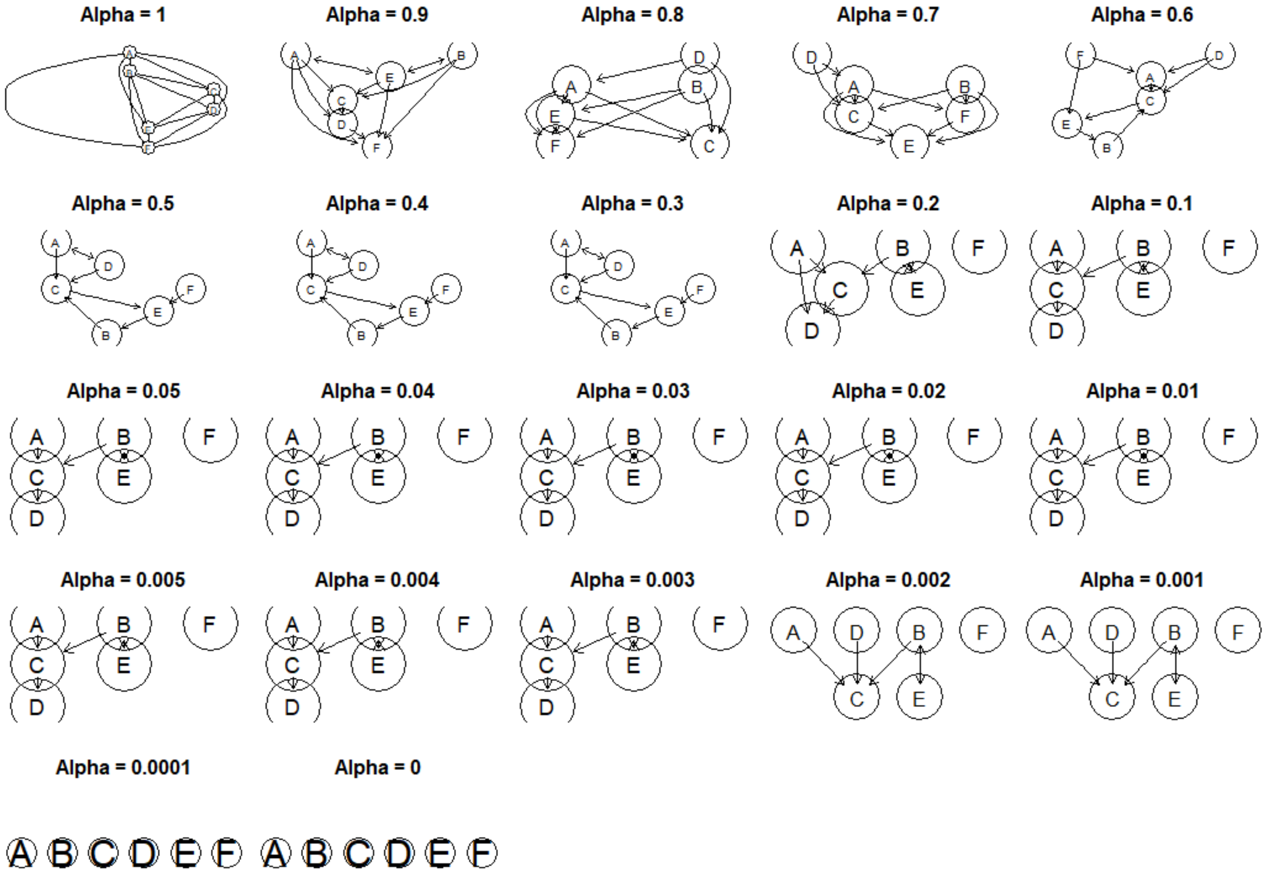
```
par(mfrow = c(5,5))
```

5 x 5 pictures on one plot.

Then we can make all figures using the plot.

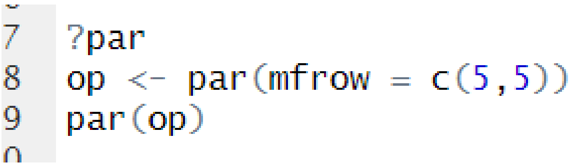


```
plot(pc.results.1, main = "Alpha = 1")
plot(pc.results.2, main = "Alpha = 0.9")
plot(pc.results.3, main = "Alpha = 0.8")
plot(pc.results.4, main = "Alpha = 0.7")
plot(pc.results.5, main = "Alpha = 0.6")
plot(pc.results.6, main = "Alpha = 0.5")
plot(pc.results.7, main = "Alpha = 0.4")
plot(pc.results.8, main = "Alpha = 0.3")
plot(pc.results.9, main = "Alpha = 0.2")
plot(pc.results.10, main = "Alpha = 0.1")
plot(pc.results.11, main = "Alpha = 0.05")
plot(pc.results.12, main = "Alpha = 0.04")
plot(pc.results.13, main = "Alpha = 0.03")
plot(pc.results.14, main = "Alpha = 0.02")
plot(pc.results.15, main = "Alpha = 0.01")
plot(pc.results.16, main = "Alpha = 0.005")
plot(pc.results.17, main = "Alpha = 0.004")
plot(pc.results.18, main = "Alpha = 0.003")
plot(pc.results.19, main = "Alpha = 0.002")
plot(pc.results.20, main = "Alpha = 0.001")
plot(pc.results.21, main = "Alpha = 0.0001")
plot(pc.results.21, main = "Alpha = 0")
```



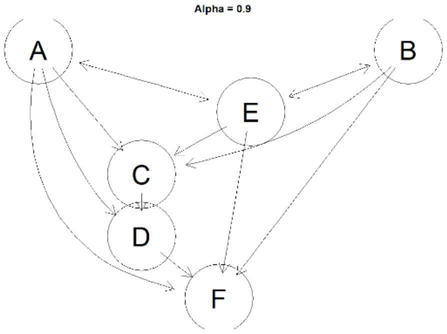
In the description of the pc (?pc) and hints was written that bi-directed edge is the order-dependence issues on the edge orientations.

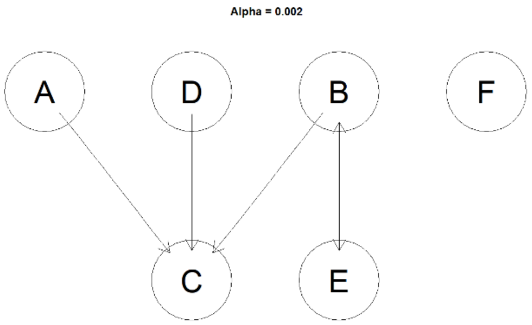
I got each picture separately using code below and again run all plots:

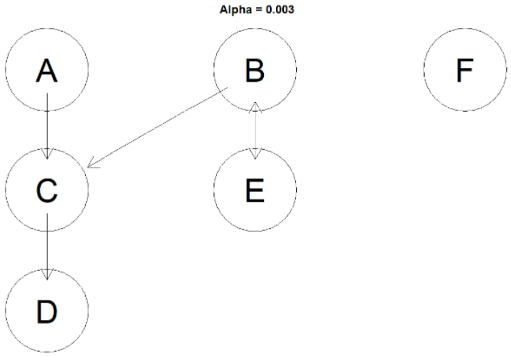


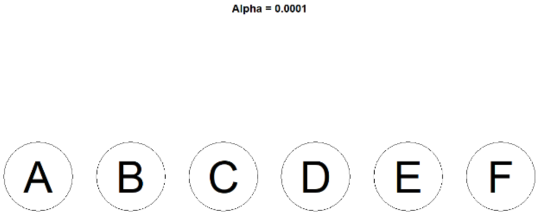
```
?par
op <- par(mfrow = c(5,5))
par(op)
```

Separately figures have better quality.


Alpha = 0.9

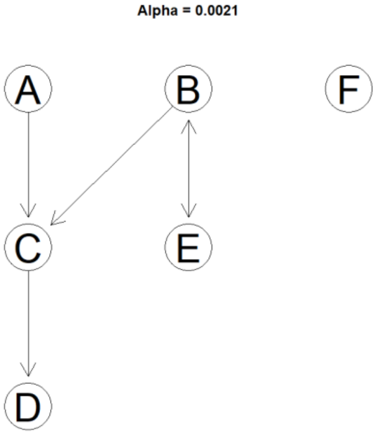

Alpha = 0.002

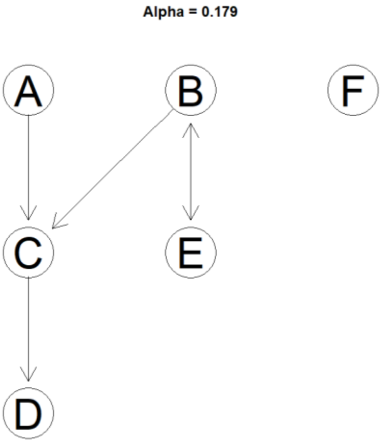

Alpha = 0.003


Alpha = 0.0001

I can see that if alpha becomes smaller, then the number of edges decreases.

I used different values of alpha and I obtained that optimums are alpha ≈ (0.0021, 0.179), it was so close to figure 1 except B-E bi-directed edge.


Alpha = 0.0021


Alpha = 0.179