Network analysis

Lourens Waldorp and Jolanda Kossakowski

Network discovery

You will use several packages in R. The Appendix indicates how to load the packages in one simple download. In the Appendix it is also explained how to load the data set required for this session. Several functions from this package are used to investigate the data and learn which connections are likely. We will first look into the data a bit in the first part and in the second part we will consider discovery using algorithms. In the final part we will consider causal effects.

Part 1.
Given is a large data set called `dat` in R with 8 variables and $n = 5000$ observations. These observations are from a non-experimental setting, i.e., an observational study.

We will first inspect the correlations and partial correlations. Call for the correlations by

```
dat.cor <- cor(dat)
dat.cor
```

This will provide you with the 8×8 correlation matrix of the variables. Find the correlation between variables 2 and 6. This is approximately 0.508. Do you think there could be a connection between these variables? Next consider the partial correlations

```
dat.pcor <- cor2pcor(dat.cor)
dat.pcor
```

This will obtain the partial correlations for each pair of variables, so again an 8×8 matrix. An element ω_{ij} of this matrix tells you what the correlation is between variables i and j when all other variables (there are 6 of them) have been partialled out, i.e., have no influence on the correlation between i and j.

Question 1. Compare the partial correlation between 2 and 6 to the correlation of that same pair. What do you find? Is this surprising? Explain the difference briefly?

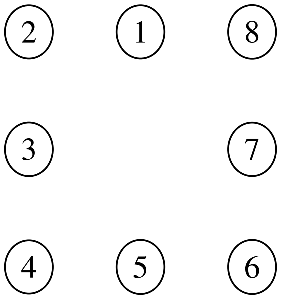
To get a better view at what the interesting partial correlations are, we can use a method called SIN, which stands for (Significant, Intermediate, Non-significant) (Drton and Perlman, 2004). In this algorithm each of the partial correlations is tested against 0. That is, the null hypothesis $H_0 : \omega_{ij} = 0$ is tested the alternative against $H_1 : \omega_{ij} \neq 0$. The P-values have been corrected for multiple comparisons, since you are performing several tests simultaneously (in fact you are doing $p(p-1)/2$) tests). You can get these tests by typing

```
sin.ag(dat)
```

The result is a figure with on the x-axis all possible connections and on the y-axis the P-values. There are two threshold lines in gray, one at 0.1 and one at 0.5. All below 0.1 are significant, all between 0.1 and 0.5 are intermediate (possibly interesting), and finally, all above 0.5 are non-significant. You can obtain exact P-values by typing

```
test <- sin.ag(dat)
test
```

Question 2. Given the results from SIN, which of the connections do you think are likely for these data? You can draw them in the figure below (or make one in your own document)



You can use this figure to later explain what you will find with Pearl's algorithm.

Part 2.
Now that we know a a bit about the data, we can try some algorithms. Recall from the presentation the algorithm Inferred Causation (Pearl and Verma, 1991; Pearl, 2000)

IC-Algorithm Pearl (1988, 1991)

Input $\hat{P}$ a sampled distribution

Output some acyclic graph for $\hat{P}$

1. For each pair a and b, look for $(a \perp\!\!\!\perp b \mid S_{ab})$. If no such S_{ab} exists, then a and b are dependent.
2. For each trio (a, b, c) such that $a - c - b$ check if c belongs to S_{ab}. If so, then nothing. If c is not in S_{ab} then make a collider at c, i.e. $a \rightarrow c \leftarrow b$.
3. Orient as many of the undirected edges as possible, subject to: (i) no new v-structures and (ii) no cycles.

Also recall that the set S_{ab} is a set of nodes (could be empty) that explains 'away' the correlation between a and b.

The first step in the algorithm is to find the sets S_{ab} for each pair of nodes in the graph. This results in an undirected graph and is often called the skeleton. This is obtained by increasing the size (i.e., the number of nodes) in S_{ab}. Zero-order refers to (simple) correlation, first-order to partial correlation with one other variable partialled out (could be any one of them). At each stage (order) there are

$$p(p-1)/2 = 8(8-1)/2 = 28 = \frac{8!}{2!(8-2)!} = \binom{8}{2}$$

possible pairs with 8 nodes. That means that we have to go through all 28 pairs to see if there is a connection. As we saw in class, for each pair we need to inspect whether any of the others or any combination of them can "explain away" the correlation. How many partial correlations do we need to check for each possible pair of nodes?[1]

We begin by defining the data we will use to estimate the connections. This is done by introducing the correlation matrix we made before `dat.cor` in a variable referring to a sufficient statistic (which is what you need to estimate models correctly).

```
suffStat <- list(C=dat.cor,n=5000)
```

Then we define what kind of test we will use to determine whether an edge should be there or not. We use a test for normal (Gaussian) data

[1] Answer: $2^6 = \sum_{i=1}^{8-2} \binom{8-2}{i} = 64$.

```
indepTest <- gaussCItest
```

We will also set a threshold (significance level) for the test

```
alpha <- 0.1
```

and we set the number of variables

```
p <- 8
```

We also need to set the random generator so that the solution will be the same at each run of the algorithm. Type

```
set.seed(34)
```

Each type we run the algorithm we need to set the random generator to 34, otherwise the results could be different.

Now we are ready to begin the algorithm to determine which variables should be connected. For each pair of nodes all possible combinations of other nodes are listed with the P-values of the corresponding partial correlations. For each P-value we need to determine whether it is significant or not. For a given significance level α (e.g., 0.1) these P-values can be tested for significance. You can do this with

```
skel <- skeleton(suffStat, indepTest, alpha, p=p, verbose=TRUE)
```

You will see a list of combinations of nodes (edges) and p-values.

Question 3. What is the p-value for the correlation between nodes 1 and 4? And what does this mean?

Let's make a graph of this result

```
qgraph(skel)
```

Note that the edges are bidirected, indicating undirected edges here.

Question 4. How can you find out in this list why node 4 is isolated? *Hint: Consider simple correlations.*

Question 5. Why is there no edge between nodes 1 and 5 in the `skel` solution? Explain briefly. *Hint: Consider partial correlations.*

Let's briefly compare the result from the first part of the algorithm IC obtained with `skeleton()` and the result from the function `sin.ag()`. Plot both the result from `sin.ag()` (you can use either your drawing from Question 2, or make the adjacency matrix and plot it with qgraph) and from the first part of the algorithm `skeleton()`.

Question 6. What are the similarities and what are the differences? Which edge is likely from `sin.ag()` that is not included in `skeleton()`? How can you explain this?

Now let's continue with finding directed edges. Recall what we need to do from Pearl's algorithm to obtain colliders. We need to to check each triple (a, b, c) whether it has the configuration $a - c - b$. If so, then check whether c is in S_{ab}. Why? If it is *not*, then make $a \rightarrow c \leftarrow b$. This part can be performed by typing

```
set.seed(34)
pc.fit <- pc(suffStat,indepTest,alpha, p=p,verbose=TRUE)
```

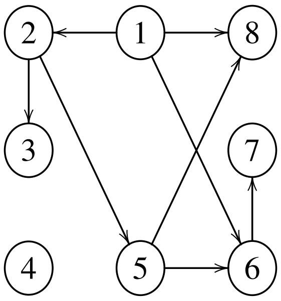
You have obtained a (partial) DAG (why partial?). Plot the graph using

```
qgraph(pc.fit)
```

(If you want both graphs in a different window, then first draw one of them, then type `x11()` on a pc or `quartz()` on a mac and then draw the other one.) This is the end result of Pearl's algorithm IC.

Question 7 Explain briefly why you think that the connection between nodes 2 and 3 remains unresolved (it is still an undirected edge)?

Here's the true graph underlying the data



Compare your result with this graph. Try to explain the differences, recall model equivalence from the lecture.

Part 3: Causal effects.
Given is the following code based on the data from the previous part and new data. In Invariant Causal Prediction (ICP) we are interested in combining (at least two) datasets that are available on the same set of observational units (often subjects) (Peters et al., 2015). We observed behaviour without any intervention (the data we assumed up to now), and we have data based on some intervention. For instance, suppose we were interested in people's opinion on eating meat. We could fist ask them their opiniions (observation), then put to them a scenario in which meat gets high VAT and so is much more expansive (intervention). Then what are their opinions? Or we could have a baseline questionnaire of patients and one after therapy (or several different sessions, etc). With ICP we try to discover if in different circumstances we find evidence for the same causal structures (directed edges).

We run the following code to generate data for two environments `ExpInd==1` means observational data and `ExpInd==2` means intervention data. In the first try we have that node 2 acts on node 3, which is true in both the observational data (same as the graph from the previous part), and in the second try we have that node 7 acts on node 2 which is not true in the observational data (see the graph from the previous part).

```
n1 <- 5000
n2 <- 4000
p  <- 8
datm <- matrix(dat,ncol=p,nrow=n1)
X <- rbind(datm[,-3],matrix(rnorm(n2*(p-1)),nrow=n2,ncol=p-1)) # simulate data
## divide data into observational (ExpInd=1) and interventional (ExpInd=2)
ExpInd <- c(rep(1,n1),rep(2,n2))
## intervention (ExpInd==2), node 2 'acts on' node 3
X[ExpInd==2,] <- sweep(X[ExpInd==2,],2, 2*rnorm(p-1) ,FUN="*")
## second  variable is the causal predictors of Y
beta <- c(0,1,rep(0,p-3))
## response variable Y is node 3 (taken out of dat)
Z <- as.numeric(X[ExpInd==2,]%*%beta + rnorm(n2))
Y <- c(datm[,3],Z) # node 3 is respnse variable
icp <- ICP(X,Y,ExpInd,showCompletion=FALSE,showAcceptedSets=FALSE)
icp
plot(X[ExpInd==1,2],Y[ExpInd==1],bty='n',col='blue',pch=16,
xlab='node 2',ylab='node 3')
points(X[ExpInd==2,2],Y[ExpInd==2],bty='n',col='red',pch=16)

# a second intervention where node 7 'acts on' 3
beta2 <- c(rep(0,p-3),1,0)
Z <- as.numeric(X[ExpInd==2,]%*%beta2 + rnorm(n2))
Y <- c(datm[,3],Z)
```

```
icp <- ICP(X,Y,ExpInd,showCompletion=FALSE,showAcceptedSets=FALSE)
icp
plot(X[ExpInd==1,6],Y[ExpInd==1],bty='n',col='blue',pch=16,
xlab='node 7',ylab='node 3')
points(X[ExpInd==2,6],Y[ExpInd==2],bty='n',col='red',pch=16)
```

Question 8. How would you interpret the result from the ICP list of effects of the nodes on the response variable (node 3) in the first set of data? Consider the first plot in your answer.

Question 9. In the second dataset nothing is found. Why would that be? [Hint: Consider the true graph.

Question 10. Compare the two plots. Can you see from these plots why the conclusion in the first set of data there is an effect of node 2 on node 3, and why in the second dataset the conclusion is that node 7 does not act on node 2?

References

Drton, M., Perlman, M., August 2004. A sinful approach to gaussian graphical model selection. Tech. Rep. 457, University of Washington, Department of Statistics.

Pearl, J., 2000. Causality: Models and prediction. Cambridge University Press.

Pearl, J., Verma, T., 1991. A theory of inferred causation. In: Proceedings of the Second International Conference of Representation and Reasoning. Morgan Kaufmann, San Francisco.

Peters, J., Bühlmann, P., Meinshausen, N., 2015. Causal inference using invariant prediction: identification and confidence intervals. arXiv preprint arXiv:1501.01332 .

APPENDIX

We need several functions to determine the network. Download ”networkAnalysis.zip” into a folder on the computer that you can easily locate. Unpack this zip file so that there are two files. Then in R change the working directory by clicking on ”change dir” in the menu ”File”. Or type

```
setwd("<path name of directory where the files are>")
```

Now the working directory of R is the directory you specified. Then type

```
source("networkAnalysisAssignement.R")
```

It may ask you to choose a repository to download some packages if you do not already have them. To get the data into R you can use the `load` function.

```
load("dat.RData")
```

or double click on the file to have it open in an R session. Once done, you should be ready to go play with the data in the object `dat`.