

# 582481 Causal analysis

## Project work

Oct 24, 2008

### General information about the project

This project work is a voluntary part of the course, raising the total number of credits awarded for the course to 6.

The maximum amount of points available for this project work is 15. A minimum of 7 points is required to pass. Please keep in mind that it is probably not enough only to just do half of the project and expect to pass, since it is highly likely that some points will be lost to mistakes, lack of clarity, etc. In other words, complete the whole project as well as you can. If you interrupt the project or your combined grade from the exam and project work is lower than your grade from the exam alone, you always have the option of taking only the exam grade (and only 4 credit points), so there is no risk involved in attempting the project.

Note that this project is *individual*, meaning each student performs the project independently and returns his/her own code, figures, and description. Discussion among students is certainly allowed (and encouraged), but each student is required to write all the code and produce all the figures and the description himself/herself.

If you get stuck with the project, *please don't hesitate to ask for help* from the lecturer. Note that I will mainly answer specific questions, and give hints for how to solve the various subproblems; don't expect me to describe every detail of how to implement the project. Nevertheless, I am here to teach you and help you, so please ask for help when you need it. The best way is to send an email to set up an appointment. Note that after November 11 I will be travelling, so after that time I can only help through email.

Please keep in mind that I have aimed to keep this project work challenging. Nevertheless, a *serious* effort (including asking for help when needed), by someone who has followed the course and taken the exam, is likely to result in at least the required 7 points. If the project turns out to be challenging for the majority of students, I will of course take that into consideration in grading.

Finally, and most importantly, remember to read all of the instructions *carefully*. Start by making sure (*now!*) that you have access to a machine that can run all the necessary software (see below under 'technical issues'). Also, please read through this document in its entirety before starting to work on the project. Finally, before submitting it, re-read it again and make sure you conform to all the requirements, so as to avoid unnecessary deductions of points.

### Returning the project

The project must be returned by November 30, 2008, at 23:59 Helsinki time, at the latest, by email to the lecturer.

Your project package should consist of exactly two parts:

- A document providing a thorough description of all the analysis results, along with explanations of the applied methods. This document is the main result of the project, it should contain all your plots and figures, and it must be returned electronically as an Adobe Acrobat (.pdf) file (i.e. when using Microsoft Word or equivalent, you must export/print the document to pdf format! This can, for instance, be done by printing to 'pdfmail' on the department computers.)

- A zip/tar file containing all the code files used in the analysis. Please do not return any data files as they make the package exceedingly large. Just return the code files that you have written, along with explanations of how to run the code to produce the numbers and the plots documented in the pdf file.

These two parts can be joined together into a single zip/tar file, or sent as two separate email attachments.

## Project description

The aim of this project is to give the students practical experience with causal analysis of data. I will generate data in various ways, then give hints or other ‘prior knowledge’ as to the data-generating process, and it is your job to infer as much as possible about the process, and answer any specific causal queries that I specify. This document will describe the required steps in detail.

## Technical issues

You are free to use any widely available ready-made analysis software. In particular, students are encouraged to use the numerical computation environment they are most familiar with (so that this project is truly about learning causal analysis, not learning a programming language). I expect that many will want to use one of these two:

**Matlab** A commercial numerical computing software system available on many CS department Linux machines. Start up using the command `matlab -nojvm`.

**R** A free statistics software package, available on the CS department Linux machines but can also be downloaded and installed for free on most platforms. Start up using the command `R`.

Please see the appendix for information on how to get help on the above. In addition, the following two software packages will be needed at least for the structure-learning task:

**Tetrad IV** Software which implements constraint-based learning of causal models. On the department Linux machines, start up using the command:  
`javaws http://www.phil.cmu.edu/projects/tetrad\_download/launchers/tetrad-4.3.6-0.jnlp`

**LiNGAM** Matlab package for learning linear non-gaussian causal models. Download it from  
`http://www.cs.helsinki.fi/group/neuroinf/lingam/`

The data files are available on the web. These are in text format (to facilitate their importing into your numerical software of choice) in the directory whose name was specified in the email I sent you. Note that some of the files are quite large because I wanted to give you enough data so that the structure learning task is not too difficult. See Appendix B for detailed instructions of loading the data.

## Task 1: Inferring do-probabilities (4p)

The data in the files `task1adata.txt` and `task1bdata.txt` has presumably been produced using causal models with the DAGs in Figure 1a and 1b (respectively). Each row is one independently drawn sample vector of  $W$ ,  $X$ ,  $Y$ , and  $Z$  (left, second, third, and right column of the data matrix, respectively). The hidden variable  $U$  has not been measured.

In each case, start by examining whether the independencies implied by the graph seem to hold. Does there seem to exist any more independencies in the data? Then, go on to estimate the conditional distribution of

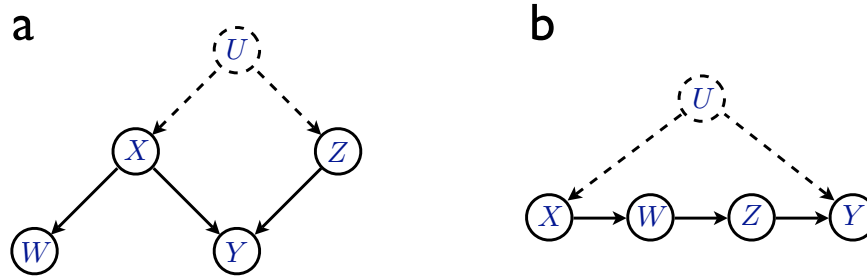


Figure 1: Directed acyclic graphs underlying the data to be used in task 1.

$Y$  given that we *set*  $X := x$ , i.e.  $P(y | \hat{x})$ . Compare this with the conditional distribution of  $Y$  given that we *observe*  $X = x$ , i.e.  $P(y | x)$

Finally, assume that we want to set  $X$  so as to maximize the expected value of  $Y$ . What is the optimal selection of value for  $X$ , and what is the resulting expected value of  $Y$ ? Answer this question both for cases *a* and *b*.

### Task 2: Counterfactuals (3p)

A DAG-based functional causal model is provided in Figure 2. From this model, compute the counterfactual probability  $P(Z_{X:=1} = 0 | X = 0, Z = 0)$ , i.e. the probability that  $Z$  would have attained the value 0 had  $X$  been 1, given that you have observed both  $X$  and  $Z$  to have the value 0.

Next, verify your analytical result numerically: Generate a large number (say ten thousand, or a hundred thousand) data vectors from the model, making sure to store the values of all the disturbance variables as well as the values of the observed variables  $X$ ,  $Y$ , and  $Z$ . Select all data vectors for which  $X = 0$  and  $Z = 0$  holds. Now change the model so that  $X$  is set to 1 in all cases and the values of the disturbance variables are unchanged, and recalculate the values of the other observed variables ( $Y$  and  $Z$ ). In how big a proportion of these cases do we observe that  $Z = 0$ ? This result should (in the limit) equal the analytical result obtained above. If not, re-check your calculations and your code.

Perform the same exercise (both analytical result and numerical check) for  $P(Z_{Y:=0} = 0 | X = 2, Y = 1, Z = 0)$ .

### Task 3: Learning network structure (5p)

For this task, I have generated five datasets (`task3adata.txt`, `task3bdata.txt`, `task3cdata.txt`, `task3ddata.txt`, and `task3edata.txt`) using five different causal DAG models with no hidden variables (note that the same datasets also come in ‘Tetrad-ready’ format, see Appendix B). In each data matrix, each row consists of one independently drawn sample vector, whereas the different columns denote different variables (lets call them  $A$ ,  $B$ ,  $C$ , ... in order of left-to-right). Your job is, using Tetrad IV and/or LiNGAM as appropriate, to try to infer the generating DAG in each case. Please also specify your confidence in your choice, and whether there are several roughly equally likely DAGs. In cases of d-separation-equivalence, specify the pattern and list all DAGs represented by that pattern. Please remember to specify any assumptions inherent in the method(s) used.

Note that it will be important to try to look at the data to judge whether it is discrete, continuous-gaussian, or continuous-non-gaussian. Also remember that you can handle continuous data in at least two different ways: you can assume a linear model, or you can discretize the values as appropriate and then employ a discrete variable framework.

In many cases it might be useful to try a few different approaches and compare the found results before making the final guess.

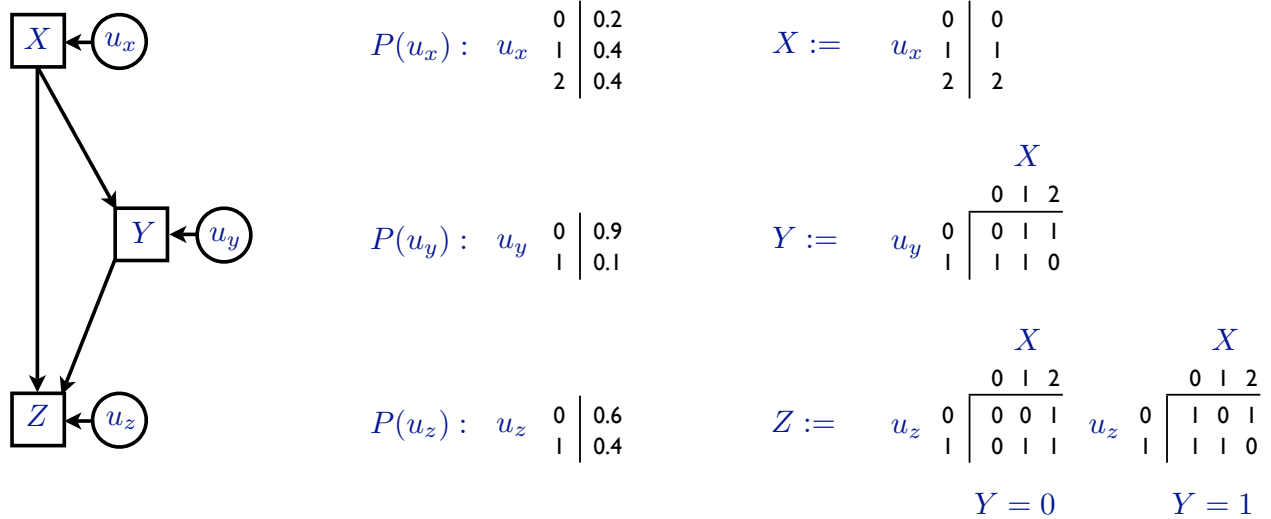


Figure 2: Functional causal model for task 2.

### Task 4: Combining causal information from multiple separate experiments (3p)

(Note that this task is not a straightforward application of the theory we went through in the course, but rather requires you to come up with your own solutions.)

The files `task4dataExp1.txt`, `task4dataExp2.txt`, and `task4dataExp3.txt` contain data from three different experimental settings over the variables A through H (columns in this order in the files, note that there are also Tetrad-ready files), where in each case the values of some variables have been assigned (that is, *set* as in do-conditioning) by randomizing (i.e. we have a randomized controlled experiment) whereas the other variables have been passively observed in this setting. In Experiment 1, variable  $H$  has been randomized; in Experiment 2, variables  $A$  and  $C$  have been randomized, and in Experiment 3 variable  $E$  has been randomized. From the combination of these datasets, try to derive as much information as possible as regards to the structure of the generating model. You may assume that the underlying data-generating structure is a DAG, that there are no hidden variables, and that all relevant distributions are faithful to the structure.

### Appendix A: Getting help on Matlab and R

In Matlab, help on the various functions is obtained by typing `help functionname` at the prompt, where `functionname` is the name of the function in question. The help facilities are quite extensive, type `helpwin` at the prompt to get a help window which you can browse.

Also, the web contains lots of documentation on Matlab. For instance, see

<http://www.owl.net.rice.edu/~elec241/matlab.html>  
[http://people.rit.edu/pnveme/Matlab\\_Course/DEFAULT.HTM](http://people.rit.edu/pnveme/Matlab_Course/DEFAULT.HTM)

As for R, see

<http://www.r-project.org/>  
<http://cran.r-project.org/doc/manuals/R-intro.pdf>

## Appendix B: Loading the data

In Matlab, load the data simply by giving the command

```
load filename.txt
```

which puts the given matrix in a variable of name 'filename', from which it can be further manipulated. Note that LiNGAM requires the data to be transposed, i.e. use

```
X = filename';
```

before feeding  $X$  to LiNGAM.

In R, the data can be loaded using

```
X <- read.table("filename.txt")
```

Finally, Tetrad requires that the file contains labels for the variables. To assist you, for tasks 3 and 4, I have created such versions of the files, they are identified by the ending 'tetrad' after the filename. In Tetrad, start by creating a data node, double-click it and press OK, next choose 'Load Data' from the 'File' menu and select your file. Select the appropriate data type and click import. Next, create a search node, and draw an arrow from the data to the search node. Double-clicking on the search node you are now ready to run the PC algorithm.

Please note that there may be some small bugs in the interface of Tetrad, but it nevertheless seems to work OK. In particular, note that p-values have to be entered with a decimal *point* (.) rather than a comma, even though Tetrad itself may print these as commas.