chapter 4

yousri elfattah

2023-12-09

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document.

# Chapter 4

## Required libraries

Running the code in this document requires installing the following R libraries:

install.packages(c(‘dagitty’,‘ggdag’,‘ggplot2’,‘mvtnorm’))

#Library to create and analyze causal graphs  
library(dagitty)  
#library to plot causal graphs   
library(ggdag)

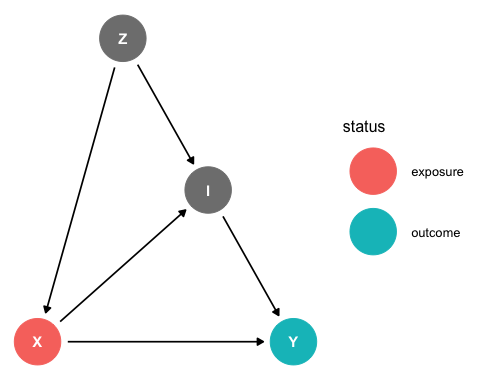
##   
## Attaching package: 'ggdag'

## The following object is masked from 'package:stats':  
##   
## filter

library(ggplot2)  
#library for multivariate normal and t distributions  
library(mvtnorm)  
library(dosearch)

## Example 4.1- Causal Graph

# Example: DAG with one exposure, and one outcome, and two covariates  
g <- dagitty("dag{  
 X -> Y   
 X <- Z -> I   
 X -> I -> Y  
 X [exposure]  
 Y [outcome]  
}")   
# set coordinates for graph layout  
coords<-list(X=c(X=1, Z=2, I=3, Y=4),  
 Y=c(X=1, Z=3, I=2, Y=1))  
coords\_df<-coords2df(coords)  
  
coordinates(g)<-coords2list(coords\_df)  
#Plot causal graph  
ggdag\_status(g) + theme\_dag()



Find all paths in g and their open status

paths(g)

## $paths  
## [1] "X -> I -> Y" "X -> Y" "X <- Z -> I -> Y"  
##   
## $open  
## [1] TRUE TRUE TRUE

Find all causal (directed) between “X” and “Y”

paths(g, "X", "Y", directed= TRUE)$paths

## [1] "X -> I -> Y" "X -> Y"

Find all descendants of “Z”

descendants(g, "Z")

## [1] "Z" "X" "Y" "I"

Find all ancestors of “Y”

ancestors(g, "Y")

## [1] "Y" "X" "Z" "I"

Find all conditional independencies

impliedConditionalIndependencies(g)

## Y \_||\_ Z | I, X

Check if “Y” and “Z” are dconnected given “I”

dconnected(g, "Y", "Z", "I")

## [1] TRUE

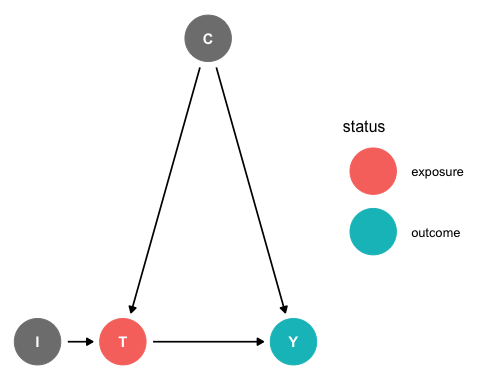
Check if “Y” and “Z” are dseparated given “X” and “I”

dseparated(g, "Y", "Z", c("X","I"))

## [1] TRUE

## Example 4.2 [Instrument Structure]

#create graph  
g <- dagitty("dag{  
 T -> Y   
 I -> T <- C -> Y  
 T [exposure]  
 Y [outcome]  
}")   
#Set position of nodes   
coords<-list(x=c(I = 0, T = 1, C = 2, Y = 3),  
 y=c(I = 0, T = 0, C = 1, Y = 0))   
coords\_df<-coords2df(coords)  
  
coordinates(g)<-coords2list(coords\_df)  
#Plot causal graph  
ggdag\_status(g) + theme\_dag()



Finding paths between “T” and “Y”. There are two open paths: a causal path T->Y, and a biasing path I<-C->Y.

paths(g, "T", "Y")

## $paths  
## [1] "T -> Y" "T <- C -> Y"  
##   
## $open  
## [1] TRUE TRUE

If the confounder “C” is unobserved (i.e., a latent variable in the structured equation model), verify that under linearity assumptions of the structured equation model, “I” can be used as an instrumental variable to infer the total effect of an exposure “T” on an outcome “Y” in the presence of latent confounding.

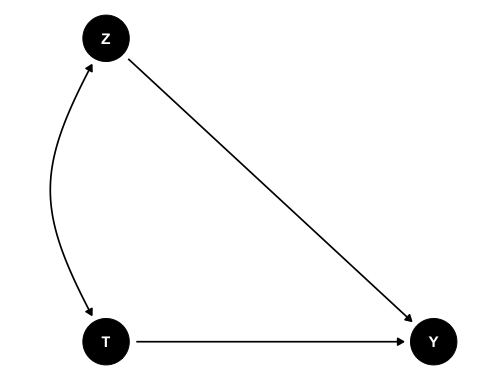
instrumentalVariables(g, "T", "Y" )

## I

## Example 4.3 [Model M1]

### Plot causal graph

# cleans workspace  
rm(list = ls())  
#-----------------------  
# dagify: Create a dagitty DAG using R-like syntax  
# ~ "regressed on"   
# ~~ "covaries with"  
#-----------------------  
g1 <- dagify(  
 Y ~ T + Z,  
 T ~~ Z  
)  
# set coordinates for graph layout  
coords<-list(X=c(Z = 1, T = 1, Y = 2),  
 Y=c(Z = 2, T = 1, Y = 1))  
coords\_df<-coords2df(coords)  
  
coordinates(g1)<-coords2list(coords\_df)  
#Plot causal graph  
ggdag(g1)+theme\_dag()



### Simulate observational data set

# simulate data (linear model)  
n <- 1e4  
# we will generate correlated, normal random variables with the program rmmvnorm in the mvtnorm library  
out <- rmvnorm(n, mean = c(0, 0),   
 sigma = matrix(c(1,.8, .8, 1), 2, 2))  
Z <- out[,1]  
T <- out[,2]  
Y <- 0.5 \* T + 0.7 \* Z + rnorm(n)  
model.data <- data.frame (  
 Z <- Z,  
 T <- T,  
 Y <- Y  
)

Find adjustment sets for path coefficient from “T” to “Y”

print( adjustmentSets(g1, "T", "Y"))

## { Z }

Linear regression without adjustment is biased

coef(lm(Y ~ T))

## (Intercept) T   
## -0.003810666 1.065931619

Regression with adjustment is unbiased

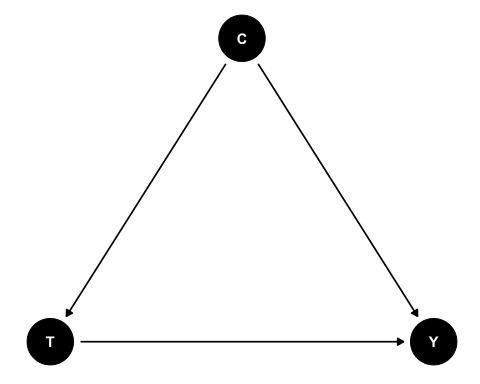
coef(lm(Y ~ T + Z))

## (Intercept) T Z   
## 0.0006793432 0.5005103882 0.7062401324

## Example 4.4 [Model M2]

### Plot causal graph

#create graph  
g <-dagify(T~C,Y~T+C)   
#Set position of nodes   
coords<-list(x=c(T = 0, C = 1, Y = 2),  
 y=c(T = 0, C = 1, Y = 0))   
coords\_df<-coords2df(coords)  
coordinates(g)<-coords2list(coords\_df)  
  
#Plot causal graph   
ggdag(g)+theme\_dag()



### Simulate observational data set

# simulate data (linear model)  
n <- 1e4  
C <- rnorm(n)  
T <- 0.5 \* C + rnorm(n)  
Y <- 0.7 \* T + 1.2 \* C + rnorm(n)  
model.data <- data.frame (  
 Z <- Z,  
 T <- T,  
 Y <- Y  
)

Find adjustment sets for path coefficient from “T” to “Y”

print( adjustmentSets(g, "T", "Y"))

## { C }

Unadjusted estimate is confounded

coef(lm(Y ~ T))

## (Intercept) T   
## 0.005266578 1.162231315

Adjusting for C recovers the causal effect

coef(lm(Y ~ T + C))

## (Intercept) T C   
## -0.006681753 0.693530300 1.189893842

## Example 4.5 [Intervention Conditioning]

data1 <- "P(x,y)"  
query1 <- "P(y|do(x))"  
graph1 <- "  
 x -> y  
 "  
dosearch(data1, query1, graph1)

## p(y|x)

data2 <- "P(x,y)"  
query2 <- "P(y|do(x))"  
graph2 <- "  
 y -> x  
 "  
dosearch(data2, query2, graph2)

## p(y)