exercise3

Xiang Li

2024/2/20

```
## Install and invoke packages
if (!"dagitty" %in% .packages(all = TRUE)) install.packages("dagitty")
if (!"ggplot2" %in% .packages(all = TRUE)) install.packages("ggplot2")
if (!"GGally" %in% .packages(all = TRUE)) install.packages("GGally")
if (!"gridExtra" %in% .packages(all = TRUE)) install.packages("gridExtra")

## Load packages
library(dagitty)
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

```
library(GGally)
```

```
## Warning: package 'GGally' was built under R version 4.3.2

## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

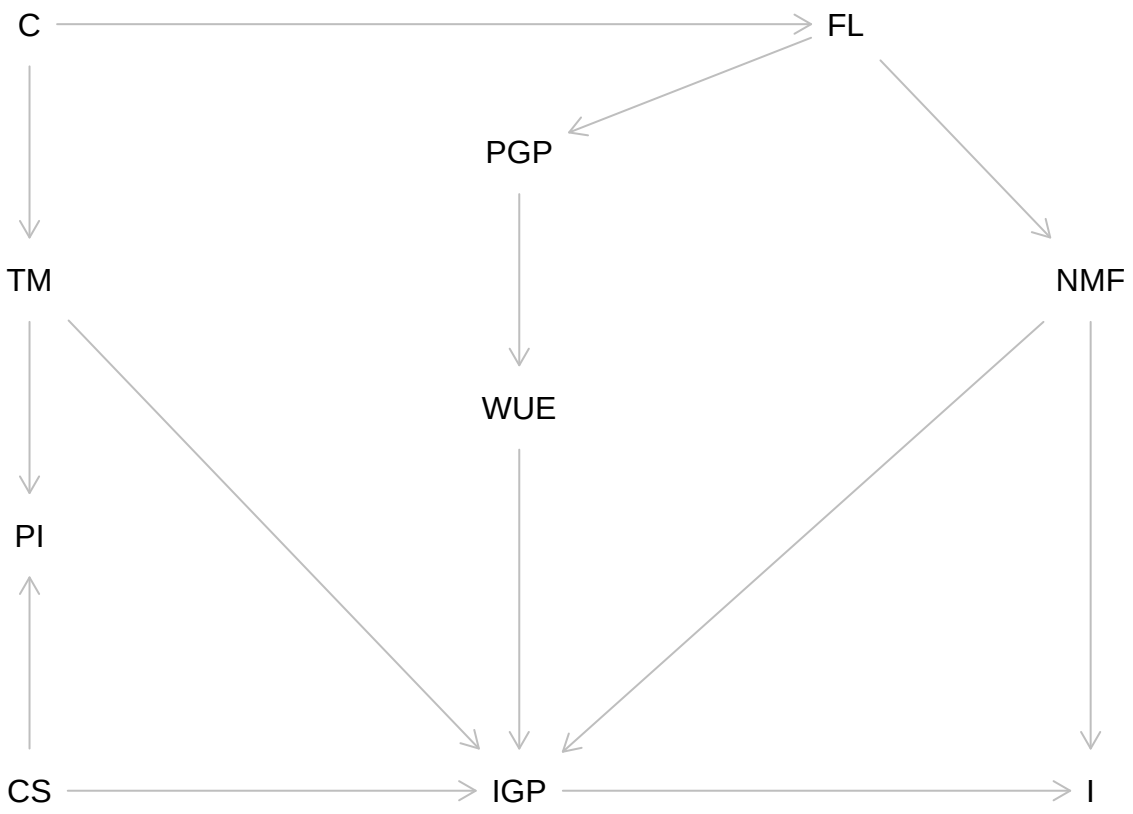
```
library(gridExtra)
```

Exercise 1

```
## Specify the DAG
g.postulated = dagitty("dag {
  C   [pos=\"-4.000,-3.000\"]
  CS  [pos=\"-4.000,3.000\"]
  FL  [pos=\"1.000,-3.000\"]
  I   [outcome,pos=\"2.500,3.000\"]
  IGP [pos=\"-1.000,3.000\"]
  NMF [pos=\"2.500,-1.000\"]
  PGP [pos=\"-1.000,-2.000\"]
  PI  [pos=\"-4.000,1.000\"]
  TM  [pos=\"-4.000,-1.000\"]
  WUE [exposure,pos=\"-1.000,0.000\"]
  C   -> { FL TM }
```

```
  CS  -> { IGP PI }
  FL  -> { NMF PGP }
  IGP -> I
  NMF -> { I IGP }
  PGP -> WUE
  TM  -> { IGP PI }
  WUE -> IGP
  }
")

## Plotting DAG
plot(g.postulated)
```



1

{CS, TM, WUE, NMF}

2

{IGP, CS, TM, C, WUE, PGP, FL, NMF}

3

{IGP, I}

4

{PI, IGP, I}

5

{C->FL->NMF->I, C->FL->NMF->IGP->I, C->FL->PGP->WUE->IGP->I, C->TM->IGP->I}

6

Yes.

Exercise 2

7

{}

8

{IGP, NMF}

9

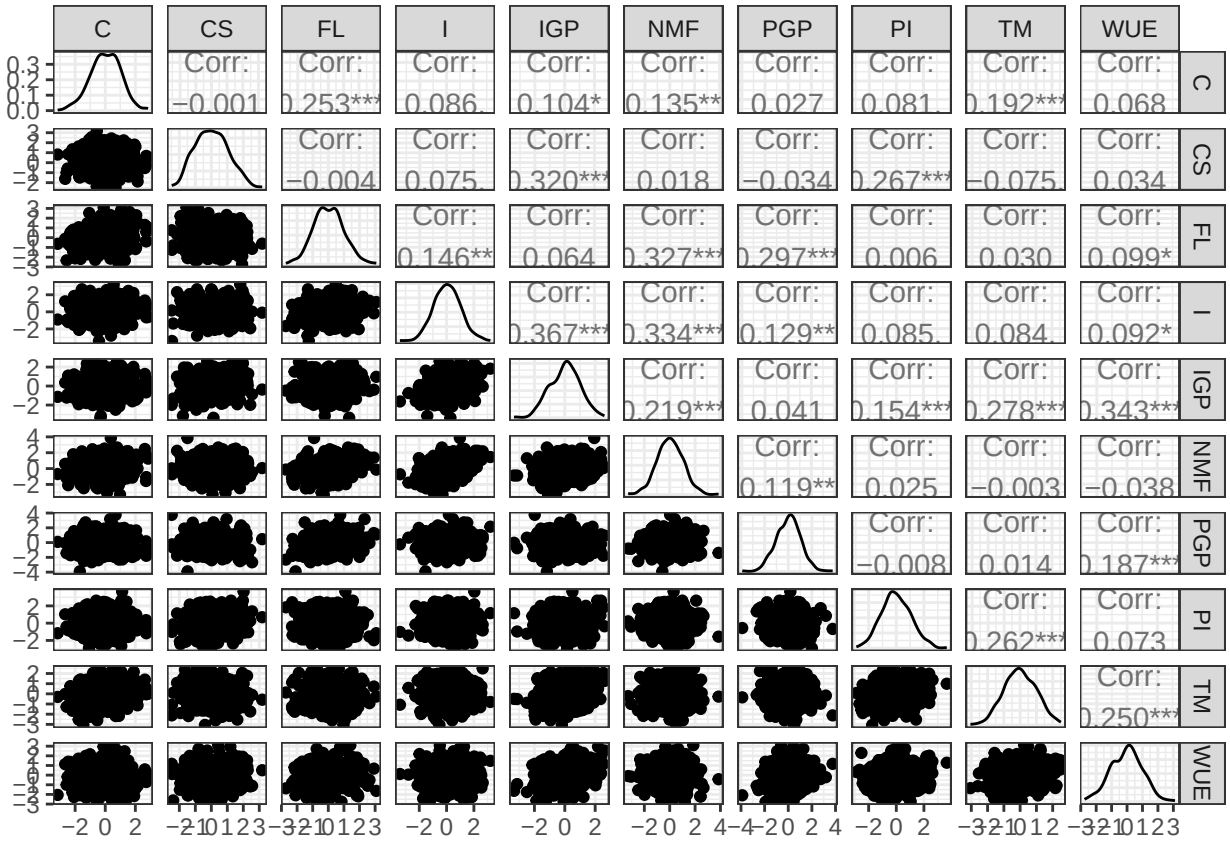
```
impliedConditionalIndependencies(g.postulated)
```

```
## C _||_ CS
## C _||_ I | IGP, NMF
## C _||_ I | NMF, TM, WUE
## C _||_ I | NMF, PGP, TM
## C _||_ I | FL, TM
## C _||_ IGP | NMF, TM, WUE
## C _||_ IGP | NMF, PGP, TM
## C _||_ IGP | FL, TM
## C _||_ NMF | FL
## C _||_ PGP | FL
## C _||_ PI | TM
## C _||_ WUE | PGP
## C _||_ WUE | FL
## CS _||_ FL
## CS _||_ I | IGP, NMF
## CS _||_ NMF
```

```
## CS _||_ PGP
## CS _||_ TM
## CS _||_ WUE
## FL _||_ I | IGP, NMF
## FL _||_ I | NMF, TM, WUE
## FL _||_ I | C, NMF, WUE
## FL _||_ I | NMF, PGP, TM
## FL _||_ I | C, NMF, PGP
## FL _||_ IGP | NMF, TM, WUE
## FL _||_ IGP | C, NMF, WUE
## FL _||_ IGP | NMF, PGP, TM
## FL _||_ IGP | C, NMF, PGP
## FL _||_ PI | TM
## FL _||_ PI | C
## FL _||_ TM | C
## FL _||_ WUE | PGP
## I _||_ PGP | FL, WUE
## I _||_ PGP | C, NMF, WUE
## I _||_ PGP | NMF, TM, WUE
## I _||_ PGP | IGP, NMF
## I _||_ PI | CS, TM
## I _||_ PI | IGP, NMF
## I _||_ TM | IGP, NMF
## I _||_ WUE | IGP, NMF
## IGP _||_ PGP | FL, WUE
## IGP _||_ PGP | C, NMF, WUE
## IGP _||_ PGP | NMF, TM, WUE
## IGP _||_ PI | CS, TM
## NMF _||_ PGP | FL
## NMF _||_ PI | TM
## NMF _||_ PI | C
## NMF _||_ PI | FL
## NMF _||_ TM | C
## NMF _||_ TM | FL
## NMF _||_ WUE | PGP
## NMF _||_ WUE | FL
## PGP _||_ PI | TM
## PGP _||_ PI | C
## PGP _||_ PI | FL
## PGP _||_ TM | C
## PGP _||_ TM | FL
## PI _||_ WUE | PGP
## PI _||_ WUE | FL
## PI _||_ WUE | C
## PI _||_ WUE | TM
## TM _||_ WUE | PGP
## TM _||_ WUE | FL
## TM _||_ WUE | C
```

10

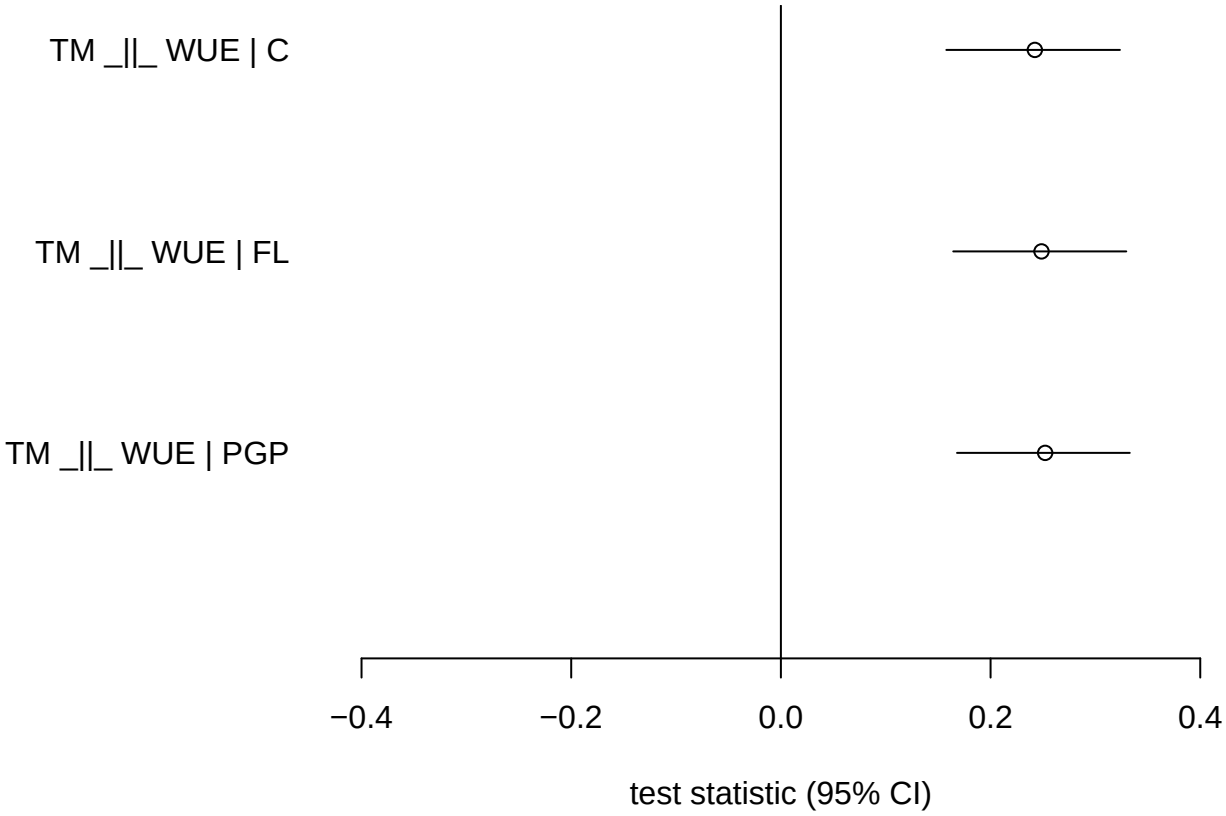
```
## Load and inspect data
load("sportsData.RData")
ggpairs(data) + theme_bw()
```




```
## Perform local tests
r = localTests(g.postulated, data)

## Adjust using Holm-Bonferroni correction Retain only those tests with
## adjusted p-value < .05
r = r[p.adjust(r$p.value) < 0.05, ]

## Plot conditional independencies inconsistent with data
r = r[order(r$p.value), ]
par(mar = c(4, 8.5, 1.5, 0.5))
plotLocalTestResults(r, bty = "n", xlim = c(-0.4, 0.4), ylim = c(0.1, 3.1), axis.pars = list(las = 1,
    lty = 0))
```



Because it makes no assumption of independence between the consecutive tests.

11

Given {C}, TM and WUE are not independent. Given {FL}, TM and WUE are not independent. Given {PGP}, TM and WUE are not independent.

12

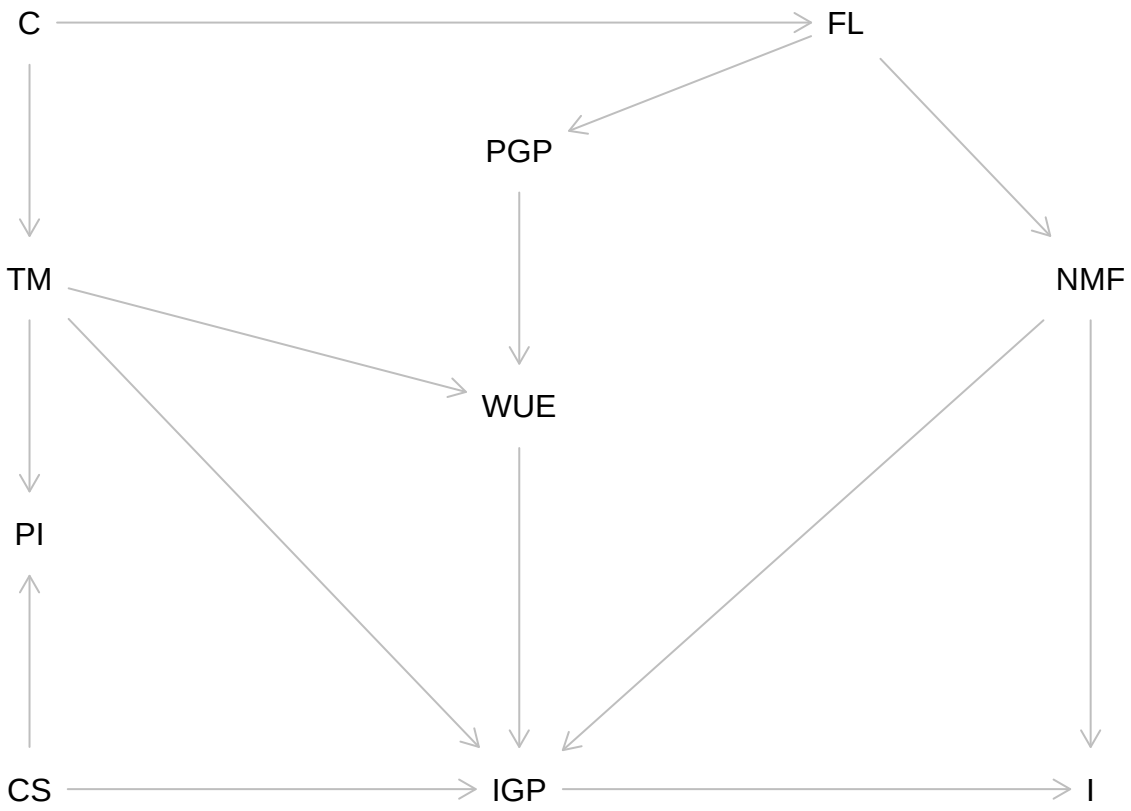
Find a new path between TM and WUE and then find d-seperating sets and check the conditional independencies again.

Exercise 3

```
## Amended DAG
g.amended = dagitty("dag {
  C   [pos=\"-4.000,-3.000\"]
  CS  [pos=\"-4.000,3.000\"]
  FL  [pos=\"1.000,-3.000\"]
  I   [outcome,pos=\"2.500,3.000\"]
  IGP [pos=\"-1.000,3.000\"]
  NMF [pos=\"2.500,-1.000\"]
```

```
  PGP [pos=\"-1.000,-2.000\"]
  PI  [pos=\"-4.000,1.000\"]
  TM  [pos=\"-4.000,-1.000\"]
  WUE [exposure,pos=\"-1.000,0.000\"]
  C   -> { FL TM }
  CS  -> { IGP PI }
  FL  -> { NMF PGP }
  IGP -> I
  NMF -> { I IGP }
  PGP -> WUE
  TM  -> { IGP PI WUE}
  WUE -> IGP
  }
")

## Plotting amended DAG
plot(g.amended)
```



13

{PGP, TM} {FL, TM} {PGP, TM, C}

```
adjustmentSets(g.amended, type = "all")
```

```
## { FL, TM }
## { C, FL, TM }
## { CS, FL, TM }
## { C, CS, FL, TM }
## { NMF, TM }
## { C, NMF, TM }
## { CS, NMF, TM }
## { C, CS, NMF, TM }
## { FL, NMF, TM }
## { C, FL, NMF, TM }
## { CS, FL, NMF, TM }
## { C, CS, FL, NMF, TM }
## { PGP, TM }
## { C, PGP, TM }
## { CS, PGP, TM }
## { C, CS, PGP, TM }
## { FL, PGP, TM }
## { C, FL, PGP, TM }
## { CS, FL, PGP, TM }
## { C, CS, FL, PGP, TM }
## { NMF, PGP, TM }
## { C, NMF, PGP, TM }
## { CS, NMF, PGP, TM }
## { C, CS, NMF, PGP, TM }
## { FL, NMF, PGP, TM }
## { C, FL, NMF, PGP, TM }
## { CS, FL, NMF, PGP, TM }
## { C, CS, FL, NMF, PGP, TM }
## { FL, PI, TM }
## { C, FL, PI, TM }
## { CS, FL, PI, TM }
## { C, CS, FL, PI, TM }
## { NMF, PI, TM }
## { C, NMF, PI, TM }
## { CS, NMF, PI, TM }
## { C, CS, NMF, PI, TM }
## { FL, NMF, PI, TM }
## { C, FL, NMF, PI, TM }
## { CS, FL, NMF, PI, TM }
## { C, CS, FL, NMF, PI, TM }
## { PGP, PI, TM }
## { C, PGP, PI, TM }
## { CS, PGP, PI, TM }
## { C, CS, PGP, PI, TM }
## { FL, PGP, PI, TM }
## { C, FL, PGP, PI, TM }
## { CS, FL, PGP, PI, TM }
## { C, CS, FL, PGP, PI, TM }
## { NMF, PGP, PI, TM }
## { C, NMF, PGP, PI, TM }
## { CS, NMF, PGP, PI, TM }
## { C, CS, NMF, PGP, PI, TM }
## { FL, NMF, PGP, PI, TM }
## { C, FL, NMF, PGP, PI, TM }
```

```
## { CS, FL, NMF, PGP, PI, TM }
## { C, CS, FL, NMF, PGP, PI, TM }
```

14

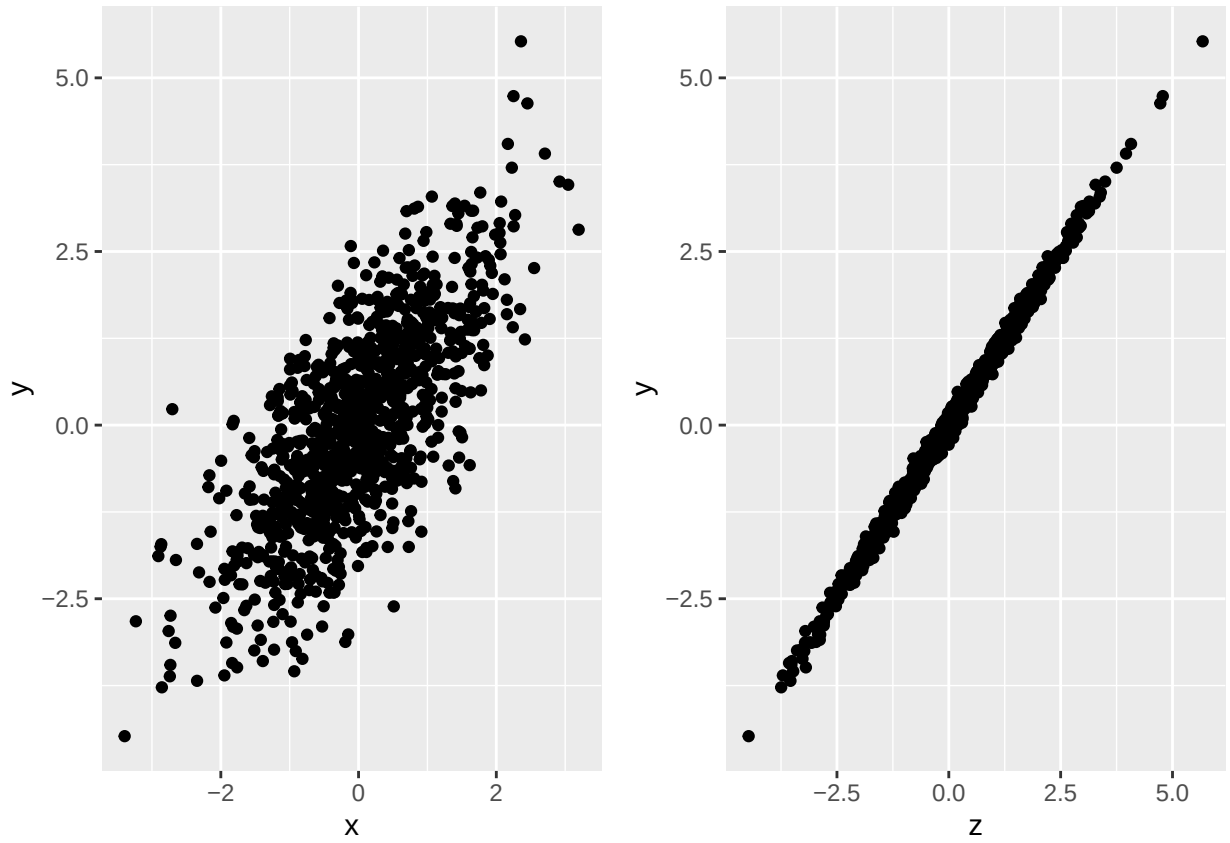
{PGP, TM} {FL, TM} {NMF, TM}

```
adjustmentSets(g.amended, type = "minimal")
```

```
## { NMF, TM }
## { FL, TM }
## { PGP, TM }
```

Additional (take-home) exercise

```
## Generate and plot data
set.seed(1234)
n = 1000
x = rnorm(n, 0, 1)
y = x + rnorm(n, 0, 1)
z = y + rnorm(n, 0, 0.1)
DAT = data.frame(x, y, z)
xyplot = ggplot(DAT, aes(x = x, y = y)) + geom_point()
zyplot = ggplot(DAT, aes(x = z, y = y)) + geom_point()
grid.arrange(xyplot, zyplot, ncol = 2)
```



15

```
lm1 = lm(y ~ x, data = DAT)
summary(lm1)
```

```
##
## Call:
## lm(formula = y ~ x, data = DAT)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.1661 -0.6439  0.0145  0.6537  3.0684
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.01599    0.03100   0.516    0.606
## x            1.05571    0.03109  33.954   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9801 on 998 degrees of freedom
## Multiple R-squared:  0.536,  Adjusted R-squared:  0.5355
## F-statistic:  1153 on 1 and 998 DF,  p-value: < 2.2e-16
```

```r
lm2 = lm(y ~ z, data = DAT)
summary(lm2)
```

```
##
## Call:
## lm(formula = y ~ z, data = DAT)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.304275 -0.068659 -0.002899  0.070714  0.309222
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.002932   0.003198  -0.917    0.359
## z            0.995975   0.002221 448.415   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1011 on 998 degrees of freedom
## Multiple R-squared:  0.9951, Adjusted R-squared:  0.9951
## F-statistic: 2.011e+05 on 1 and 998 DF,  p-value: < 2.2e-16
```

Z would be a better predictor for Y, because the association between Z and Y is stronger.

16

A good predictor in the regression sense doesn't always have a good causal effect on the outcome of interest.