Estimating average treatment effect

Iyar Lin

26 November, 2018

Contents

1 Intro

Assuming we're able to find the DAG underlying the dataset, this script demonstrates average treatment effect (ATE) estimation.

2 Define model and simulate dataset

Below I plot a graph model with the following variable set:

$$V = \{X, Y, Z, W\}$$

and function set (All disturbances U are distributed standard normal unless stated otherwise):

$$F = \{f_1, f_2, f_3, f_4\}$$

such that

$$X = f_1(Z, U_X) = \begin{cases} a, & \text{if } Z + U_X \geq 0.61 \\ b, & \text{if } Z + U_X \geq -0.61 \,\&\, Z + U_X < 0.61 \\ c, & \text{if } Z + U_X < -0.61 \end{cases}$$

$$Z = f_3(U_Z) = U_Z$$

$$W = f_4(X, U_W) \sim \begin{cases} poiss(1) & \text{if } X = a \\ poiss(2) & \text{if } X = b \\ poiss(3) & \text{if } X = c \end{cases}$$
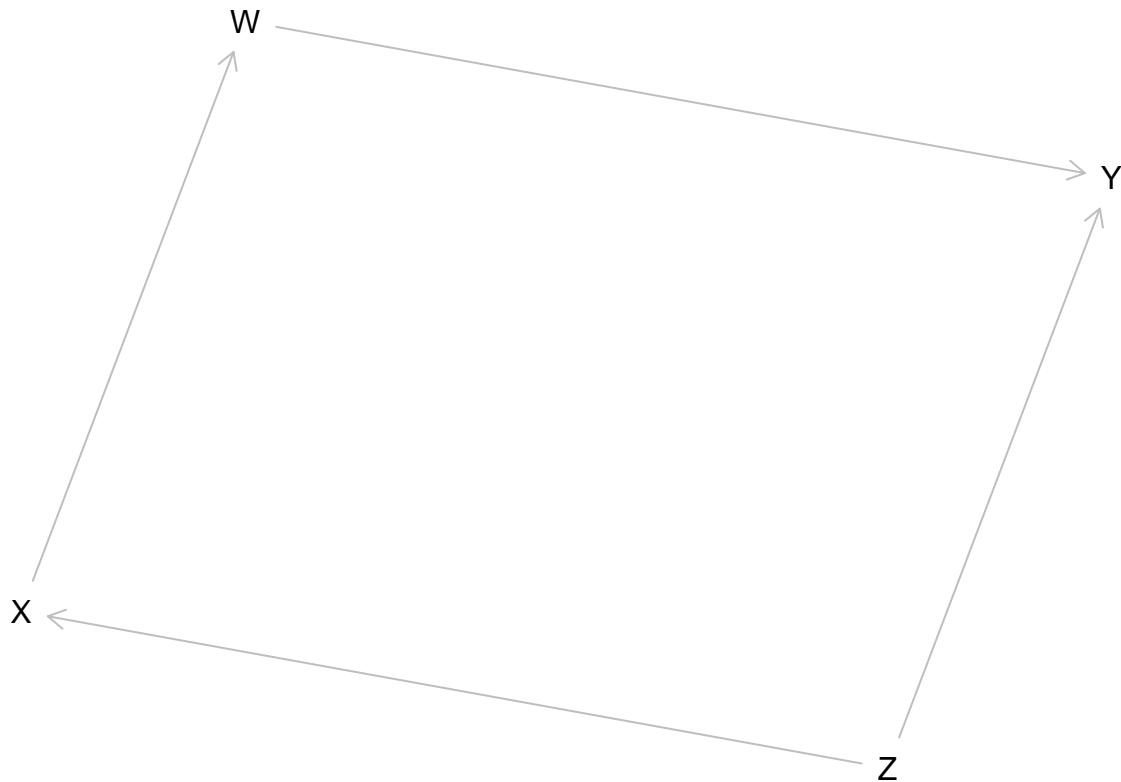
and

$$Y = f_2(W, Z, U_Y) \sim N(Z + W, 1)$$

So finally we have

$$E(Y|do(X=x)) = \begin{cases} 1, & \text{if } X = a \\ 2, & \text{if } X = b \\ 3, & \text{if } X = c \end{cases}$$

Below is a plot of the resulting DAG:

```
g <- dagitty("dag {
Y [outcome]
X [exposure]
X -> W
W -> Y
Z -> X
Z -> Y
}")

plot(graphLayout(g))
```


Below I simulate a dataset according to the above toy model.

```
N <- 1000
Z <- rnorm(N)
X_tag <- Z + rnorm(N)
X <- ifelse(X_tag > qnorm(2/3, 0, sqrt(2)), "a",
            ifelse(X_tag > qnorm(1/3, 0, sqrt(2)), "b", "c"))

W <- sapply(X, function(x){
  if(x == "a") rpois(1, 1)
  else if(x == "b") rpois(1, 2)
  else rpois(1, 3)
})
```

```
Y <- mapply(function(z,w){
  rnorm(1, z + w, 1)
  }, Z, W)

sim_data <- data.frame(X,Y,Z,W)
```

3 Estimate average treatment effect (ATE)

In Pearl eq 3.5 (p. 57) the post intervention of Y given $do(X = x)$ is given by

$$P(Y = y|do(X = x)) = \sum_z P(Y = y|X = x, Z = z)P(Z = z)$$

Where Z is a variable set satisfying the backdoor criteria (A.K.A "adjustment set").

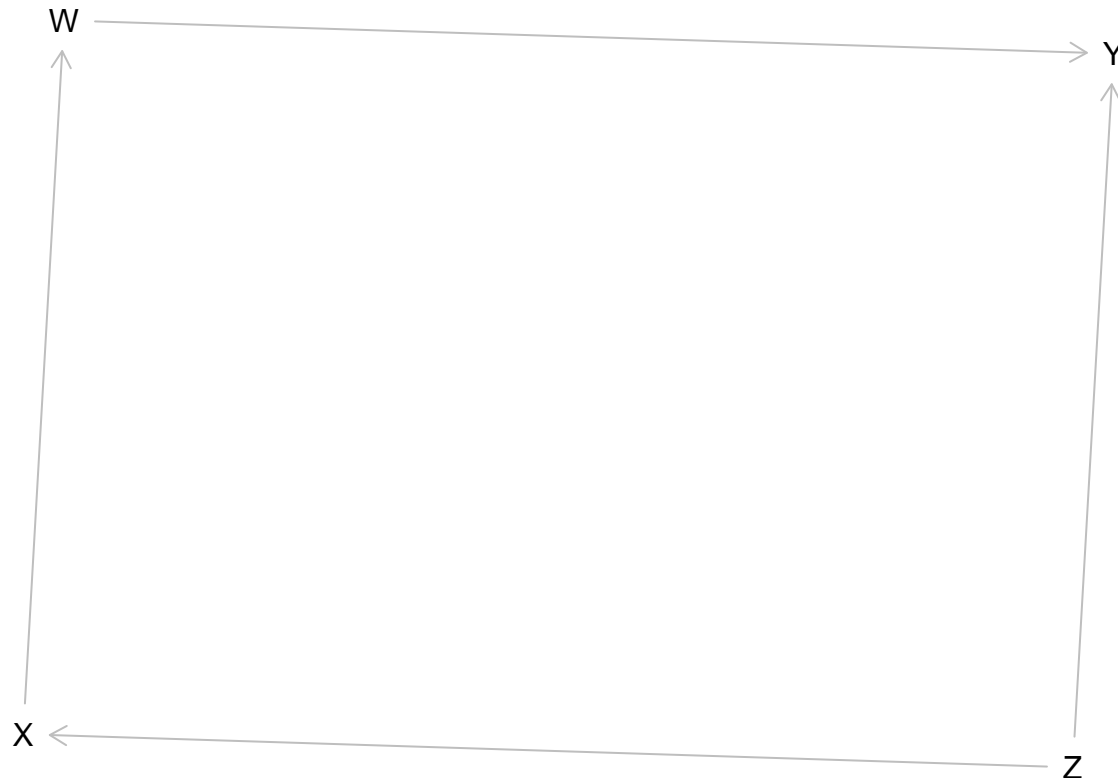The ATE is given by:

$$\mathbb{E}(Y|do(X = x)) = \sum_y \sum_z yP(Y = y|X = x, Z = z)P(Z = z) = \sum_z P(Z = z)\sum_y yP(Y = y|X = x, Z = z) = \sum_z P(Z = z)\mathbb{E}$$

We usually use regular machine learning framworks to estimate $\mathbb{E}(Y|x, z)$. When we predict for the original dataset we have the join probability of Z: $P(Z = z)$ preserved so no need to estimate it. I'll validate that in the results below.

Looking at the graph below:

```
plot(graphLayout(g))
```

we can see that the adjustment set is $Z = \{Z\}$.

We'll use a linear regression to estimate $\mathbb{E}(Y|x, z)$.

Below are the estimated ATE using the correct procedure:

```r
model <- lm(Y ~ Z + X, data = sim_data)
Z_prob <- density(x = sim_data$Z)

interventions <- levels(sim_data$X)
ATE <- vector(length = length(interventions))
for(i in 1:length(interventions)){
  intervention_data <- data.frame(X = interventions[i], Z = Z_prob$x)
  ATE[i] <- sum(Z_prob$y*predict(model, intervention_data)/sum(Z_prob$y))
}

pandoc.table(data.frame(intervention = interventions, ATE = ATE))
```

intervention	ATE
a	0.985
b	2.157
c	3.145

Below are the ATE estimates when regressing Y on X:

```r
model <- lm(Y ~ X, data = sim_data)
ATE <- vector(length = length(interventions))
for(i in 1:length(interventions)){
  intervention_data <- data.frame(X = interventions[i])
  ATE[i] <- predict(model, intervention_data)
}
pandoc.table(data.frame(intervention = interventions, ATE = ATE))
```

intervention	ATE
a	1.811
b	2.204
c	2.28

We can see that the ordering is preserved but the estimates are all bundeled together.

Below are the ATE estimates when regressing Y on all variables:

```r
model <- lm(Y ~ ., data = sim_data)
ATE <- vector(length = length(interventions))
for(i in 1:length(interventions)){
  intervention_data <- sim_data %>%
    mutate(X = interventions[i])
  ATE[i] <- mean(predict(model, intervention_data))
}
pandoc.table(data.frame(intervention = interventions, ATE = ATE))
```

intervention	ATE
a	2.178

intervention	ATE
b	2.152
c	1.966

We can see here X is estimated to have virtually no effect.

Below are the ATE estimates when using the correct adjustment set but without re-weighting:

```r
model <- lm(Y ~ X + Z, data = sim_data)
ATE <- vector(length = length(interventions))
for(i in 1:length(interventions)){
  intervention_data <- sim_data %>%
    mutate(X = interventions[i])
  ATE[i] <- mean(predict(model, intervention_data))
}
pandoc.table(data.frame(intervention = interventions, ATE = ATE))
```

intervention	ATE
a	0.985
b	2.157
c	3.145

Looks like we're getting similar results! So no need to estimate density!