POL 212

Causality and Data Wrangling

Amanda Loehrke

University of California, Davis

February 12, 2026

Today's Plan

- Causality
- DAGs
- Data wrangling (Assignment #5 tips)

Experimental design and causal inference

- **Experimental design**: process of setting up studies to test causal relationships
  - Experiments manipulate variables and use randomization
- **Causal inference**: identifying a cause-and-effect relationship from data
  - Goes beyond association/correlation between variables

Potential outcomes framework

- Used to understand the impact of treatments, interventions, or exposures on outcomes
- Considers the potential result that would occur for each individual under different treatment conditions
- Assumes each individual has a potential outcome for both the treated and untreated conditions
- The treatment effect is the difference between these two potential outcomes (averaged over the sample)

The fundamental problem of causal inference

- We only ever observe one outcome for each individual
- Each unit either does or does not receive treatment
- We cannot see the hypothetical untreated outcome for a unit in the treatment condition, and vice versa
- We cannot actually compare both potential outcomes for each individual

The fundamental problem, continued

- The unobservable or unrealized outcome is called the counterfactual
- The goal of causal inference is to identify a causal effect by addressing solutions to this fundamental problem
- We use causal inference to predict what would have happened in the counterfactual
- Take POL 285 to learn much, much more!

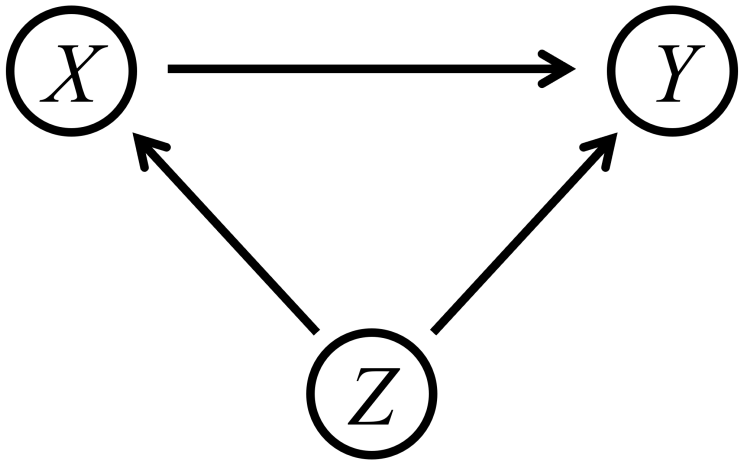
Directed Acyclic Graphs

- A graphical representation of a chain of causal effects
- Visual tools to represent causal relationships
- Helpful to identify confounders, mediators, and colliders
- The "shape" of the DAG depends on the relationship you assume between variables
- The paths (arrows) tell us which variable causes the other

Types of DAGs

- **Confounding** (aka the fork)
  - A third variable is a common cause of X and Y
  - Failing to control for this variable biases the estimate of the causal effect of X on Y
  - Conditioning on this variable blocks the backdoor path
- **Collider**
  - X and Y have a common effect
  - Conditioning on this variable opens a spurious path and induces bias
- **Mediator** (aka the pipe)
  - A variable lies on the causal path from X to Y
  - Conditioning on this variable blocks part (or all) of the causal effect

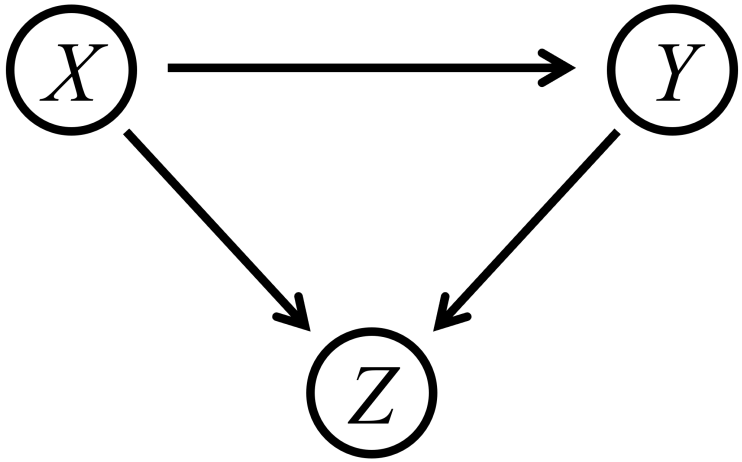
DAG w/ confounding variable (fork)



The Fork

- Z has a causal effect on both X and Y
- Since we want to isolate the causal effect of X on Y, we need to control for Z to demonstrate the unbiased effect of the treatment on the outcome
- This gets rid of the back-door path where $X \leftarrow Z \rightarrow Y$

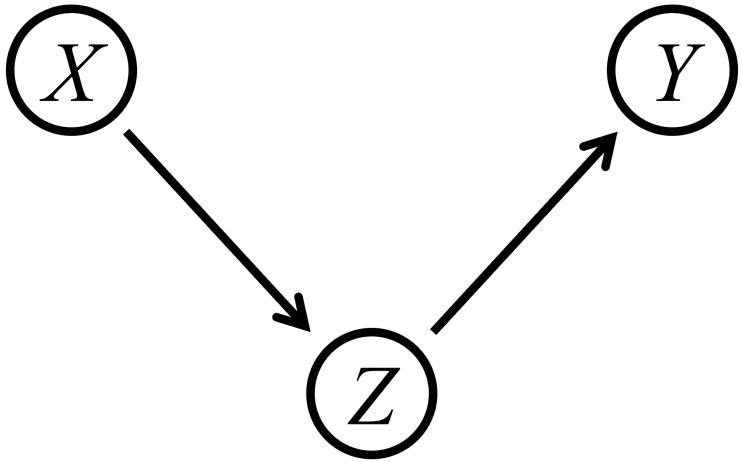
DAG w/ colliding variable



The Collider

- X and Y both have a causal effect on Z
- We don't control for Z since it is a common effect and would bias the estimate of the effect of X on Y

DAG w/ mediating variable (pipe)



The Pipe

- X does not have an independent causal effect on Y
- The causal effect only happens through a mediator, Z

Data wrangling and regression

- Goals:
  - Translate survey instruments into usable variables
  - Diagnose variable types and measurement labels
  - Use `tidyverse` to transform and summarize data
  - Compute group means
  - Run a linear regression model
  - Extract and interpret model output

What does the data look like?

- `head()` view the first six rows (or more)
- `summary()` summary statistics or frequency tables
- `glimpse()` compact overview of variables and data types
- `nrow()` number of observations (sample size)
- `class()` identify the data type of an object
- `levels()` view category levels of a factor
- `table()` frequency counts of values

Useful functions for transformations and descriptive statistics

- `mutate()` create or modify variables
- `case_when()` multiple, ordered conditions
- `if_else()` binary, type-stable conditional assignment
- `select()` choosing variables
- `filter()` subsetting observations
- `group_by()` define groups for comparison
- `summarise()` compute summary statistics by group
- `factor()` convert a variable into a categorical variable
- `as.numeric()` coerce a variable to numeric

Linear regression

- `lm()` estimate a linear regression (OLS)
- `summary()` with a model object, print regression coefficients, statistics, etc.

Writing functions

- `function()` define a user-written function
- `return()` specify the function's output

Logical and numeric operators

- > greater than
- < less than
- >= greater than or equal to
- <= less than or equal to
- == equal to
- | or
- ! not
- & and
- %>% pipe operator

In R example

```r
# Load packages
library(haven)
library(tidyverse)

# Import dataset
anes <- read_dta("anes_pilot_2022_stata_20221214.dta")

# Explore dataset (some options):
head(anes)
View(anes)
summary(anes)
glimpse(anes)
nrow(anes)
```

Example, continued

- Find the variables for race and Trump feeling thermometer in the user guide for the ANES data (note: race is multiple variables like `eth`, `rwh`, `rbl`, `rain`)
- Race: this is a categorical/nominal variable
- Below, the code makes one `race` variable

```
# Let's do four categories | White (rwh), Black (rbl), Hispanic (eth), Other
anes <- anes %>% mutate(race_cat = case_when(
  eth == 1  ~ "Hispanic", # eth: 1 = Yes (Hispanic), 2 = No. Hispanic checked
  ↪  first so all eth==1 are Hispanic (n=242)
  rwh == 1  ~ "White",
  rbl == 1  ~ "Black",
  TRUE      ~ "Other"),
  race = factor(race_cat, levels = c("White", "Black", "Hispanic", "Other")))
  ↪  # Make into a factor variable

table(anes$race) # frequency of each category
class(anes$race) # type of variable
levels(anes$race) # see the factor levels
head(anes$race, 10) # print first ten rows
```

Example, continued

- Feeling thermometer Trump: we need to get rid of missingness (values that are -7 in the `fttrump` variable)

```
# Feeling thermometer (Trump): this is a scale ranging from 0-100
# Note which end of the scale is warmer/colder (check codebook!)
# We need to account for missing data (codebook range [-7,100])
table(anes$fttrump)
class(anes$fttrump)

anes <- anes %>%
  mutate(ft_trump = if_else(
    fttrump >= 0 & fttrump <= 100, # if statement
    as.numeric(fttrump), # return: numeric scores
    NA_real_ # else: numeric NAs
  ))

class(anes$ft_trump) # type of variable
summary(anes$ft_trump) # Useful for summary stats of a numeric variable
```

Example, continued

- Below, the code calculates the average Trump feeling thermometer for the newly created `race` variable

```
## Find the group means of the Trump ft by race using dplyr
# 1. group by racial categories
# 2. calculate mean feeling thermometer for each category
# (optional) 3. Keep the count of each category
# (optional) 4. Store the group means as an object
group_means <- anes %>%
  group_by(race) %>%
  summarise(mean_ft_trump = mean(ft_trump, na.rm = TRUE),
            n = n())

group_means
print(group_means)
```

Example, continued

- Next, we will calculate a linear regression

```
## Estimate a linear regression:
# y ~ XB + E
# y = feeling thermometer (Trump)
# X = race

reg <- lm(ft_trump ~ race, data = anes)
summary(reg)
```

- This gives us the estimated regression equation:
- $\widehat{ft\_trump} = 42.4 - 17.5(\text{Black}) - 6.7(\text{Hispanic}) + 0.3(\text{Other})$

Example, continued

- What are the predicted ft means for each category from the model?
  - White (reference group): 42.4
  - Black: 42.4-17.5 = 24.9
  - Hispanic: 42.4-6.7= 35.7
  - Other: 42.4+0.3 = 42.7
- In words, the intercept represents the mean Trump ft score among White respondents. Black respondents score, on average, about 17.5 points lower than White respondents. Hispanic respondents score about 6.7 points lower than White respondents. Respondents in the "Other" category do not differ meaningfully from White respondents.
- Remember our group means from above? Algebraically, the linear regression model is equivalent to comparing the group means. Each coefficient is the difference between the group's mean and the 'White' mean.

Example, continued

- Next, create a function to return *only* the R squared value from the regression model's output

```r
## Return only the R^2 value from the linear regression
names(summary(reg))

summary(reg)$r.squared

# Function:
get_R2 <- function(model) {
  R2 <- summary(model)$r.squared
  return(R2)
}

get_R2(reg)
```

- Note, this isn't exactly what Assignment #5 is asking you to do, but will help introduce important functions you will use in R for the data wrangling and linear regression tasks!
- As always, send any questions you may have aloehrke@ucdavis.edu