OSE data science

Prof. Dr. Philipp Eisenhauer

Dec 08, 2021

CONTENTS

1 Lectures 3

2 Problem sets 135

3 Handouts 159

4 Projects 165

5 Partners 167

6 Organization 169

7 Textbooks 171

8 Reviews 173

9 Powered by 175

This course introduces students to basic microeconometric methods. The objective is to learn how to make and evaluate causal claims. By the end of the course, students should be able to apply each of the methods discussed and critically evaluate research based on them. Throughout the course we will make heavy use of Python and its SciPy ecosystem as well as Jupyter Notebooks.

CHAPTER ONE

LECTURES

We provide a set of lectures that are all provided as Jupyter Notebooks.

1.1 Introduction

We briefly introduce the course and discuss some basic ideas about counterfactuals and causal inference. We touch on the two pillars of the counterfactual approach to casusal analysis. We first explore the basic ideas of the potential outcome model and then preview the use of causal graphs.

1.1.1 Introduction

> This course introduces students to basic microeconmetric methods. The objective is to learn how to make and evaluate causal claims. By the end of the course, students should to able to apply each of the methods discussed and critically evaluate research based on them.

I just want to discuss some basic features of the course. We discuss the core references, the tooling for the course, student projects, and illustrate the basics of the potential outcomes model and causal graphs.

Causal questions

What is the causal effect of …

- neighborhood of residence on educational performance, deviance, and youth development
- school vouchers on learning?
- of charter schools on learning?
- worker training on earnings?
- …

What causal question brought you here?

Core reference Test

The whole course is built on the following textbook:

- **Winship, C., & Morgan, S. L. (2007)**. Counterfactuals and causal inference: Methods and principles for social research. Cambridge, England: *Cambridge University Press*.

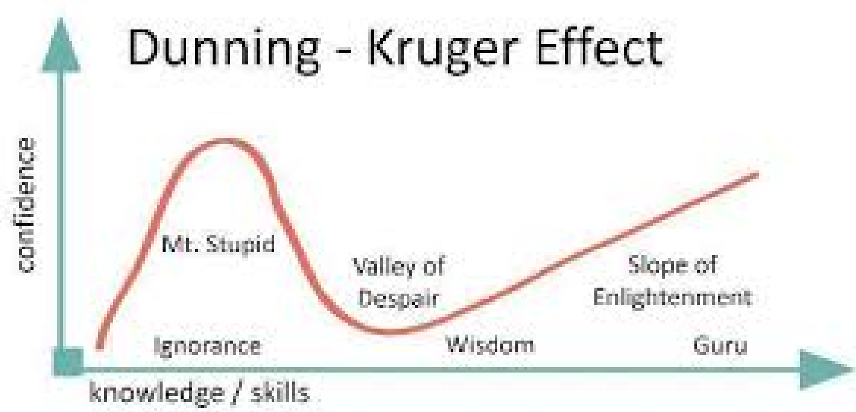
This is a rather non-standard textbook in economics. However, I very much enjoy working with it as it provides a coherent conceptual framework for a host of different methods for causal analysis. It then clearly delineates the special cases that allow the application of particular methods. We will follow their lead and structure our thinking around the **counterfactual approach to causal analysis** and its two key ingredients **potential outcome model** and **directed graphs**.

It also is one of the few textbooks that includes extensive simulation studies to convey the economic assumptions required to apply certain estimation strategies.

It is not very technical at all, so will also need to draw on more conventional resources to fill this gap.

- Wooldridge, J. M. (2001). *Econometric analysis of cross section and panel data*. Cambridge, MA: The MIT Press.
- Angrist, J. D., & Pischke, J. (2009). *Mostly harmless econometrics: An empiricists companion*. Princeton, NJ: Princeton University Press.
- Frölich, M., and Sperlich, S. (2019). *Impact evaluation: Treatment effects and causal analysis*. Cambridge, England: Cambridge University Press.

Focusing on the conceptual framework as much as we do in the class has its cost. We might not get to discuss all the approaches you might be particularly interested in. However, my goal is that all of you can draw on this framework later on to think about your econometric problem in a structured way. This then enables you to choose the right approach for the analysis and study it in more detail on your own.



Combining this counterfactual approach to causal analysis with sufficient domain-expertise will allow you to leave the valley of despair.

Lectures

We follow the general structure of Winship & Morgan (2007).

- Counterfactuals, potential outcomes and causal graphs
- Estimating causal effects by conditioning on observables
  - regression, matching, …
- Estimating causal effects by other means
  - instrumental variables, mechanism-based estimation, regression discontinuity design, …

Tooling

We will use open-source software and some of the tools building on it extensively throughout the course.

- Course website
- GitHub
- Zulip
- Python
- SciPy and statsmodels
- Jupyterlan
- GitHub Actions

We will briefly discuss each of these components over the next week. By then end of the term, you hopefully have a good sense on how we combine all of them to produce sound empirical research. Transparency and reproducibility are a the absolute minimum of sound data science and all then can be very achieved using the kind of tools of our class.

Compared to other classes on the topic, we will do quite some programming in class. I think I have a good reason to do so. From my own experience in learning and teaching the material, there is nothing better to understand the potential and limitations of the approaches we discuss than to implemented them in a simulation setup where we have full control of the underlying data generating process.

To cite Richard Feynman: What I cannot create, I cannot understand.

However, it is often problematic that students have a very, very heterogeneous background regarding their prior programming experience and some feel intimidated by the need to not only learn the material we discuss in class but also catch up on the programming. To mitigate this valid concern, we started several accompanying initiatives that will get you up to speed such as additional workshop, help desks, etc. Make sure to join our Q&A channels in Zulip and attend the our Computing Primer.

Problem sets

Thanks to Mila Kiseleva, Tim Mensinger, and Sebastian Gsell we now have four problem sets available on our website.

- Potential outcome model
- Matching
- Regression-discontinuity design
- Generalized Roy model

Just as the whole course, they do not only require you to further digest the material in the course but also require you to do some programming. They are available on our course website and we will discuss them in due course.

Projects

Applying methods from data science and understanding their potential and limitations is only possible when bringing them to bear on one's one research project. So we will work on student projects during the course. More details are available here.

Data sources

Throughout the course, we will use several data sets that commonly serve as teaching examples. We collected them from several textbooks and are available in a central place in our online repository here.

Potential outcome model

The potential outcome model serves us several purposes:

- help stipulate assumptions
- evaluate alternative data analysis techniques
- think carefully about process of causal exposure

Basic setup

There are three simple variables:

- D, treatment
- Y, observed outcome
- Y_1, outcome in the treatment state
- Y_0, outcome in the no-treatment state

Examples

- economics of education
- health economics
- industrial organization
- …

Exploration

We will use our first dataset to illustrate the basic problems of causal analysis. We will use the original data from the article below:

- LaLonde, R. J. (1986). Evaluating the econometric evaluations of training programs with experimental data. *The American Economic Review*, 76(4), 604-620.

He summarizes the basic setup as follows:

> The National Supported Work Demonstration (NSW) was temporary employment program desinged to help disadvantaged workers lacking basic job skills move into the labor market by giving them work experience and counseling in sheltered environment. Unlike other federally sponsored employment programs, the NSW program assigned qualified applications randomly. Those assigned to the treatment group received all the benefits of the NSW program, while those assigned to the control group were left to fend for themselves.

What is the *effect* of the program?

We will have a quick look at a subset of the data to illustrate the **fundamental problem of evaluation**, i.e. we only observe one of the potential outcomes depending on the treatment status but never both.

[1]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np

# We collected a host of data from two other influential textbooks.
df = pd.read_csv("../../datasets/processed/dehejia_waba/nsw_lalonde.csv")
df.index.set_names("Individual", inplace=True)
```

[2]:
```
df.describe()
```

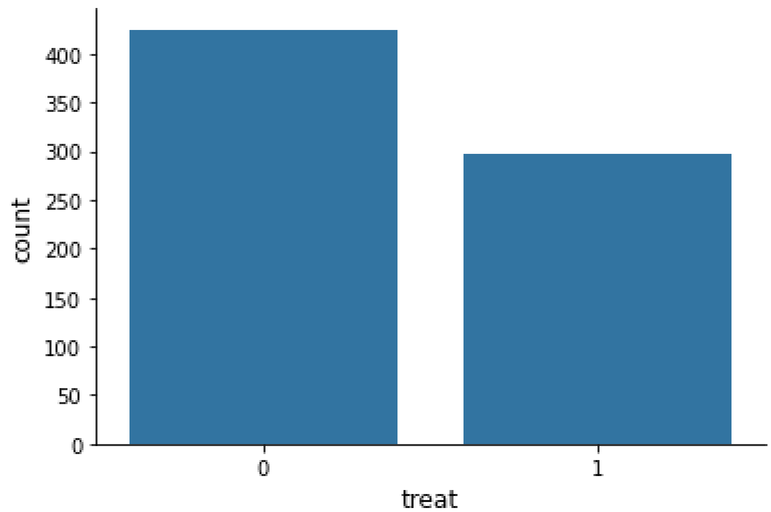
[2]:
```
            treat         age   education       black    hispanic     married  \
count  722.000000  722.000000  722.000000  722.000000  722.000000  722.000000
mean     0.411357   24.520776   10.267313    0.800554    0.105263    0.162050
std      0.492421    6.625947    1.704774    0.399861    0.307105    0.368752
min      0.000000   17.000000    3.000000    0.000000    0.000000    0.000000
25%      0.000000   19.000000    9.000000    1.000000    0.000000    0.000000
50%      0.000000   23.000000   10.000000    1.000000    0.000000    0.000000
75%      1.000000   27.000000   11.000000    1.000000    0.000000    0.000000
max      1.000000   55.000000   16.000000    1.000000    1.000000    1.000000

         nodegree          re75          re78
count  722.000000    722.000000    722.000000
mean     0.779778   3042.896575   5454.635848
std      0.414683   5066.143366   6252.943422
min      0.000000      0.000000      0.000000
25%      1.000000      0.000000      0.000000
50%      1.000000    936.307950   3951.889000
75%      1.000000   3993.207000   8772.004250
max      1.000000  37431.660000  60307.930000
```

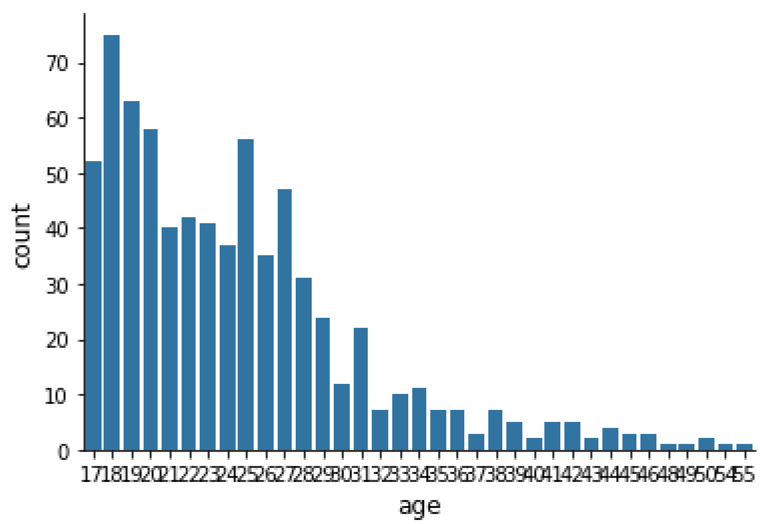
[3]:
```
# It is important to check for missing values first.
for column in df.columns:
    assert not df[column].isna().any()
```

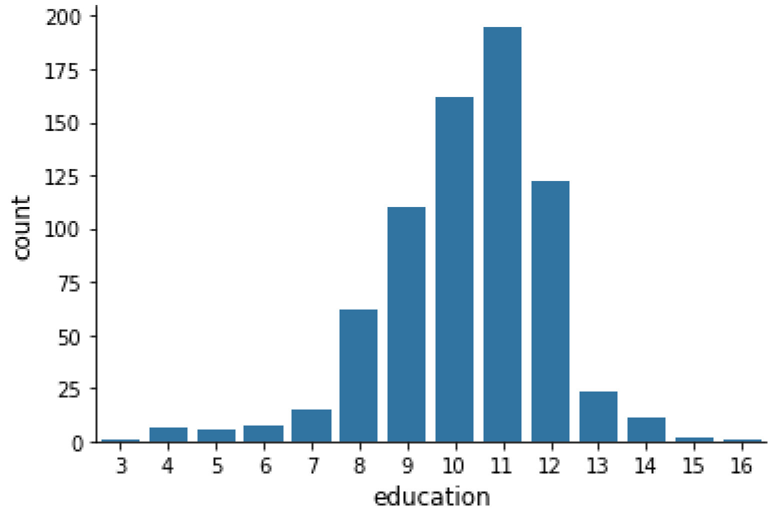
Note that this lecture, just as all other lectures, is available on so you can easily continue working on it and take your exploration to another direction.

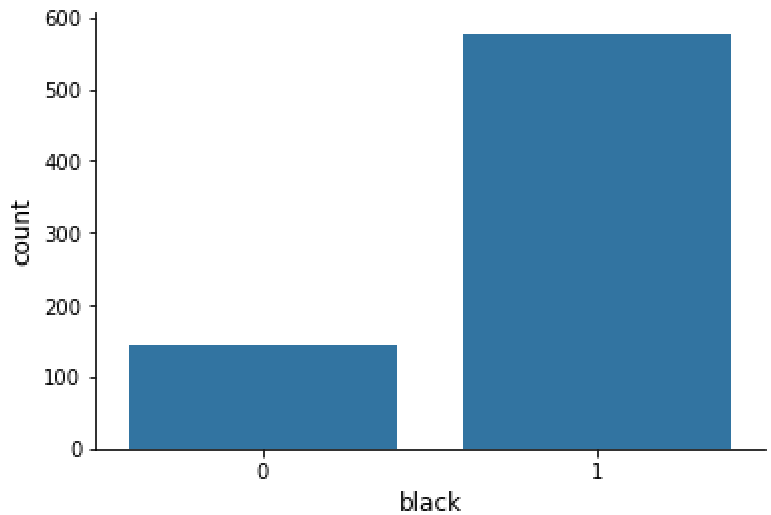
There are numerous discrete variables in this dataset describing the individual's background. How does their distribution look like?

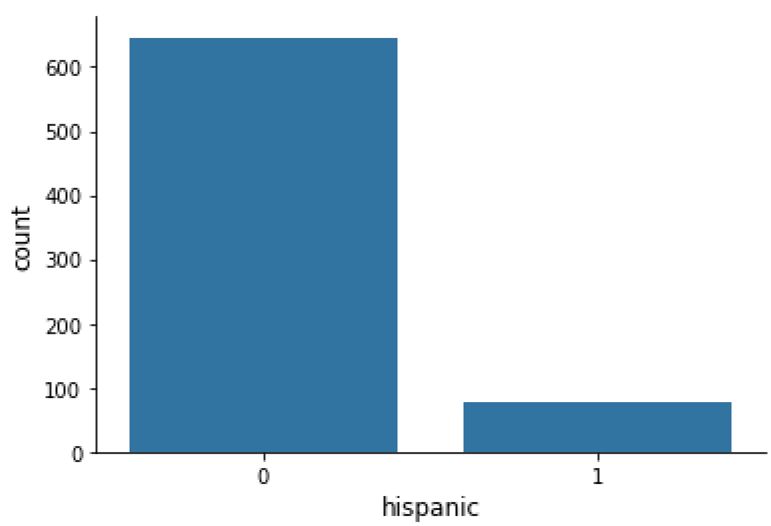
[4]:
```python
columns_background = [
    "treat",
    "age",
    "education",
    "black",
    "hispanic",
    "married",
    "nodegree",
]
for column in columns_background:
    sns.countplot(x=df[column], color="#1f77b4")
    plt.show()
```

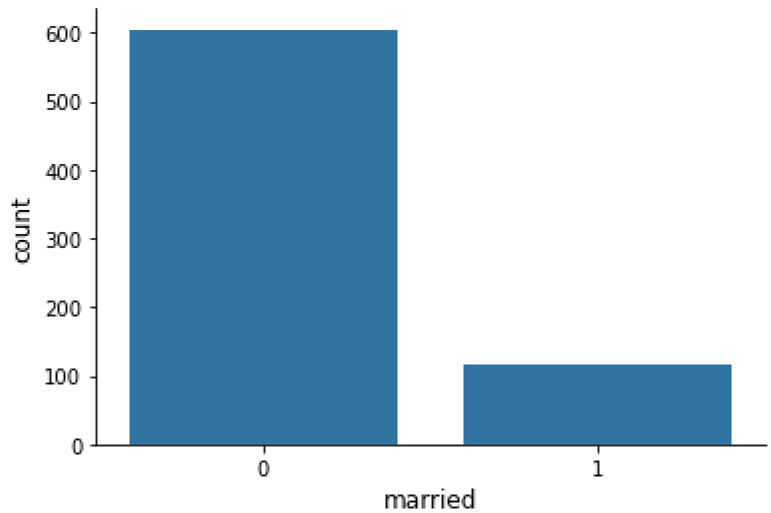


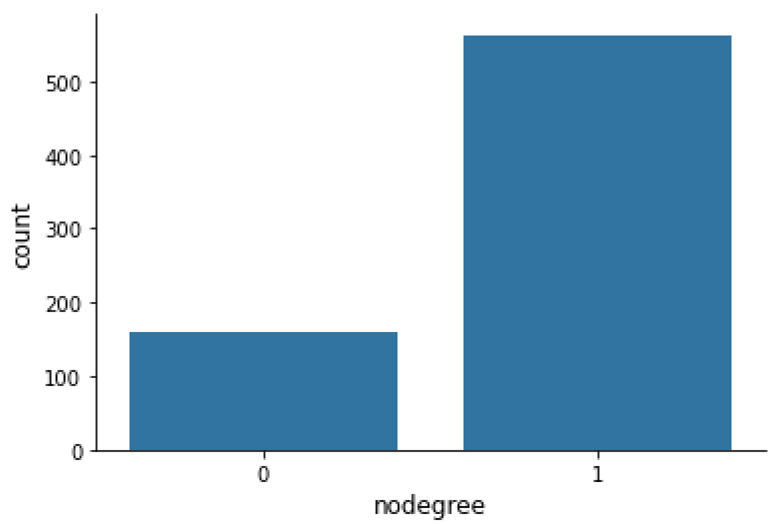



count
education
200
175
150
125
100
75
50
25
0
3
4
5
6
7
8
9
10
11
12
13
14
15
16


count
black
600
500
400
300
200
100
0
0
1

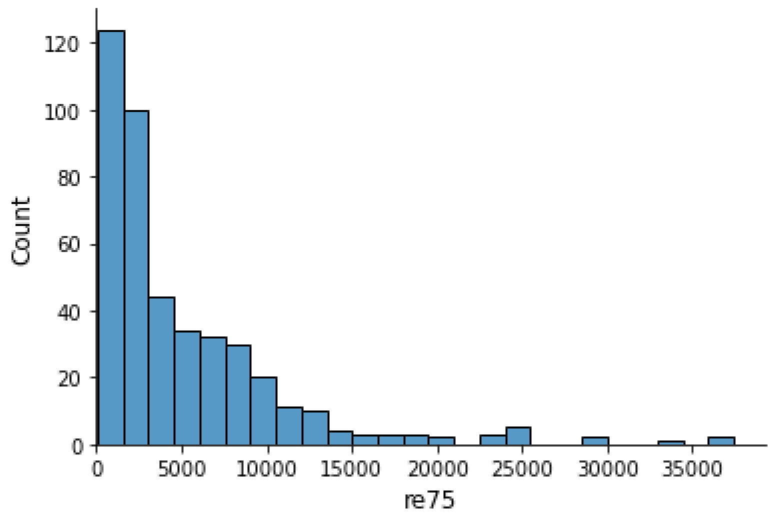

count
hispanic
600
500
400
300
200
100
0
0
1

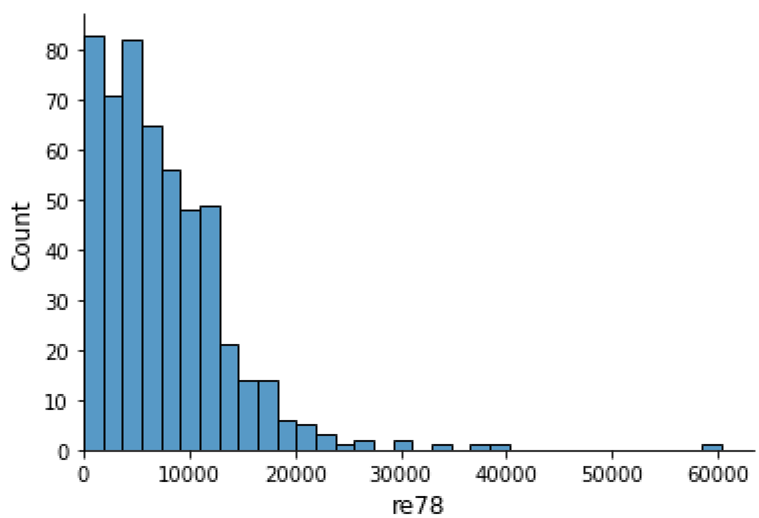




How about the continous earnings variable?

[5]:
```
columns_outcome = ["re75", "re78"]
for column in columns_outcome:

    earnings = df[column]

    # We drop all earnings at zero.
    earnings = earnings.loc[earnings > 0]

    ax = sns.histplot(earnings)
    ax.set_xlim([0, None])

    plt.show()
```





We work under the assumption that the data is generated by an experiment. Let's make sure by checking the distribution of the background variables by treatment status.

[6]:
```
info = ["count", "mean", "std"]
for column in columns_background:
    print("\n\n", column.capitalize())
    print(df.groupby("treat")[column].describe()[info])
```

```


 Treat
       count  mean  std
treat
0      425.0   0.0  0.0
1      297.0   1.0  0.0


 Age
       count       mean       std
```

(continues on next page)

(continued from previous page)

```
treat
0       425.0  24.447059  6.590276
1       297.0  24.626263  6.686391


 Education
       count       mean       std
treat
0       425.0  10.188235  1.618686
1       297.0  10.380471  1.817712


 Black
       count      mean       std
treat
0       425.0  0.800000  0.400471
1       297.0  0.801347  0.399660


 Hispanic
       count      mean       std
treat
0       425.0  0.112941  0.316894
1       297.0  0.094276  0.292706


 Married
       count      mean       std
treat
0       425.0  0.157647  0.364839
1       297.0  0.168350  0.374808


 Nodegree
       count      mean       std
treat
0       425.0  0.814118  0.389470
1       297.0  0.730640  0.444376
```

What is the data that corresponds to (Y, Y_1, Y_0, D)?

[7]:
```
# We first create True / False
is_treated = df["treat"] == 1

df["Y"] = df["re78"]
df["Y_0"] = df.loc[~is_treated, "re78"]
df["Y_1"] = df.loc[is_treated, "re78"]

df["D"] = np.nan
df.loc[~is_treated, "D"] = 0
df.loc[is_treated, "D"] = 1
```

(continues on next page)

(continued from previous page)

```
df[["Y", "Y_1", "Y_0", "D"]].sample(10)
```

[7]:
```
                    Y        Y_1        Y_0    D
Individual
479          6930.336        NaN   6930.336  0.0
94           3881.284   3881.284        NaN  1.0
146          3075.862   3075.862        NaN  1.0
407         20893.110        NaN  20893.110  0.0
269         12590.710  12590.710        NaN  1.0
8            2164.022   2164.022        NaN  1.0
592             0.000        NaN      0.000  0.0
260             0.000      0.000        NaN  1.0
421          3931.238        NaN   3931.238  0.0
35              0.000      0.000        NaN  1.0
```

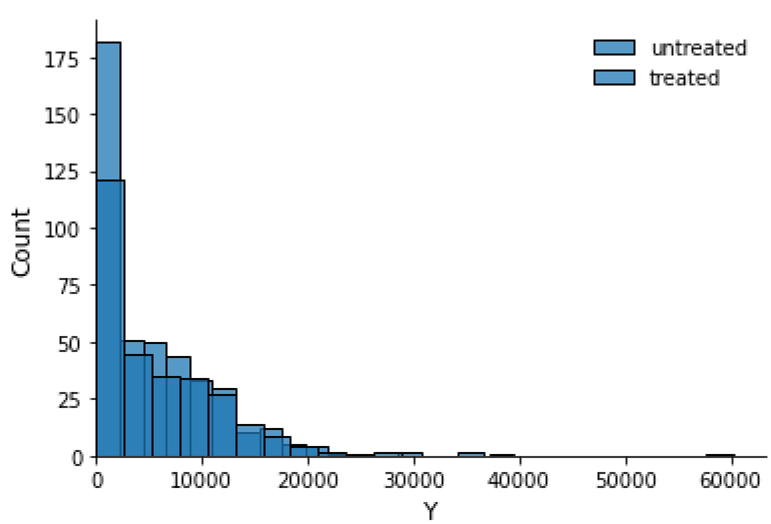
Let us get a basic impression on how the distribution of earnings looks like by treatment status.

[8]:
```
df.groupby("D")["re78"].describe()
```

[8]:
```
     count         mean          std  min        25%       50%       75%  \
D
0.0  425.0  5090.048302  5718.088763  0.0    0.0000  3746.701  8329.823
1.0  297.0  5976.352033  6923.796427  0.0  549.2984  4232.309  9381.295

          max
D
0.0  39483.53
1.0  60307.93
```

[9]:
```
ax = sns.histplot(df.loc[~is_treated, "Y"], label="untreated")
ax = sns.histplot(df.loc[is_treated, "Y"], label="treated")
ax.set_xlim(0, None)
ax.legend()
```

[9]:
```
<matplotlib.legend.Legend at 0x7fec7859b0d0>
```



We are now ready to reproduce one of the key findings from this article. What is the difference in earnings in 1978 between those that did participate in the program and those that did not?

[10]:
```
stat = df.loc[is_treated, "Y"].mean() - df.loc[~is_treated, "Y"].mean()
f"{stat:.2f}"
```

[10]:
```
'886.30'
```

Earnings are $886.30 higher among those that participate in the treatment compared to those that do not. Can we say even more?

References

Here are some further references for the potential outcome model.

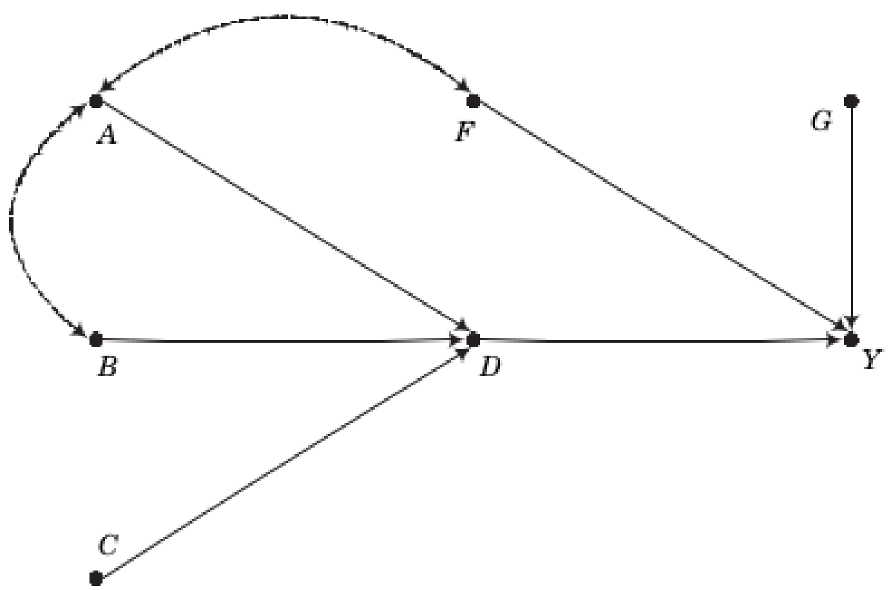
- Heckman, J. J., and Vytlacil, E. J. (2007a). *Econometric evaluation of social programs, part I: Causal effects, structural models and econometric policy evaluation*. In J. J. Heckman, and E. E. Leamer (Eds.), *Handbook of Econometrics* (Vol. 6B, pp. 4779–4874). Amsterdam, Netherlands: Elsevier Science.
- Imbens G. W., and Rubin D. B. (2015). *Causal inference for statistics, social, and biomedical sciences: An introduction*. Cambridge, England: Cambridge University Press.
- Rosenbaum, P. R. (2017). *Observation and experiment: An introduction to causal inference*. Cambridge, MA: Harvard University Press.

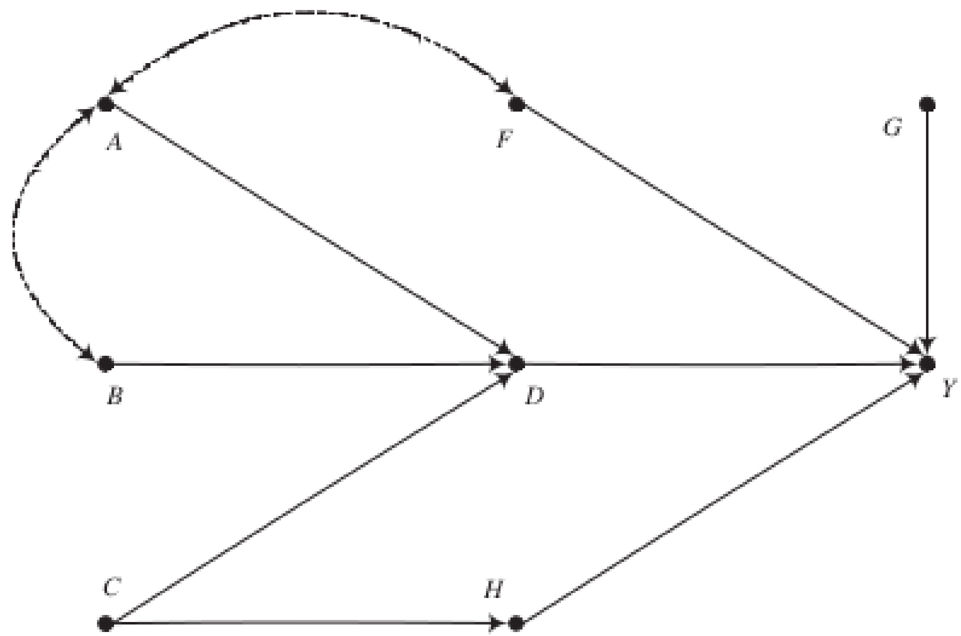
Causal graphs

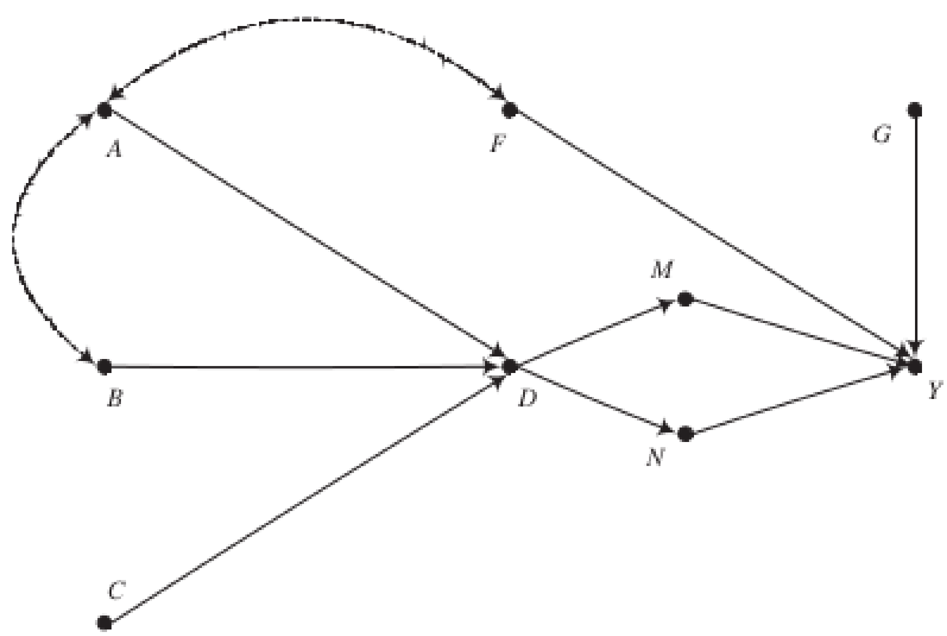
One unique feature of our core textbook is the heavy use of causal graphs to investigate and assess the validity of different estimation strategies. There are three general strategies to estimate causal effects and their applicability depends on the exact structure of the causal graph.

- condition on variables, i.e. matching and regression-based estimation
- exogenous variation, i.e. instrumental variables estimation
- establish an exhaustive and isolated mechanism, i.e. structural estimation

Here are some examples of what to expect.







The key message for now:

- There is often more than one way to estimate a causal effect with differing demands about knowledge and observability

Pearl (2009) is the seminal reference on the use of graphs to represent general causal representations.

References

- **Huntington-Klein, N., Arenas, A., Beam, E., Bertoni, M., Bloem, J., Burli, P., Chen, N., Grieco, P., Ekpe, G., Pugatch, T., Saavedra, M., Stopnitzky, Y. (2021)**. The influence of hidden researcher decisions in applied microeconomics, *Economic Impuiry*, 59, 944–960.

- **Pearl, J. (2014)**. Causality. Cambridge, England: *Cambridge University Press*.
- **Pearl, J., and Mackenzie, D. (2018)**. The book of why: The new science of cause and effect. New York, NY: *Basic Books*.
- **Pearl J., Glymour M., and Jewell N. P. (2016)**. Causal inference in statistics: A primer. Chichester, UK: *Wiley*.
- **Spiegelhalter, D. (2021)**. The Art of Statistics: Learning from Data. New York: *Hachette Book Group*.

Resources

- **LaLonde, R. J. (1986)**. Evaluating the econometric evaluations of training programs with experimental data. *The American Economic Review*, 76(4), 604-620.

1.2 Potential outcome model

We discuss the core conceptual model of the course. We initially discuss the individual-level treatment effect but then quickly scale back our ambitions to learn about population-level parameters instead. Then we turn to the stable-unit treatment assumption and address the challenges to the naive estimation of average causal effects in observational studies. We conclude with some examples that illustrate the flexibility of the potential outcome model to more than a simple binary treatment.

1.2.1 Potential outcome model

Introduction

Given what we know from the introduction about the potential outcome model, we will initially prepare the Lalonde Dataset to fit the framework and use it as a running example going forward.

What are this example's …

- potential outcomes
- counterfactual state
- treatment

[2]:
```
df = get_lalonde_data()
df.head()
```

[2]:

	treat	re78	Y	Y_0	Y_1	D
101	1	9970.681	9970.681	NaN	9970.681	1
611	0	7094.920	7094.920	7094.920	NaN	0
396	0	11223.720	11223.720	11223.720	NaN	0
681	0	4687.937	4687.937	4687.937	NaN	0
397	0	5088.760	5088.760	5088.760	NaN	0

We are dealing with a binary treatment here: $D = 1$ if the individual did participate in the training program and $D = 0$ if it did not. However, in practice assigning **treatment** is never that easy. We lump a lot of heterogeneity together (e.g. different sites, content of curriculum) that might affect the success of program participation. Maybe we should stratify the analysis by site?

Individual-specific effect of treatment

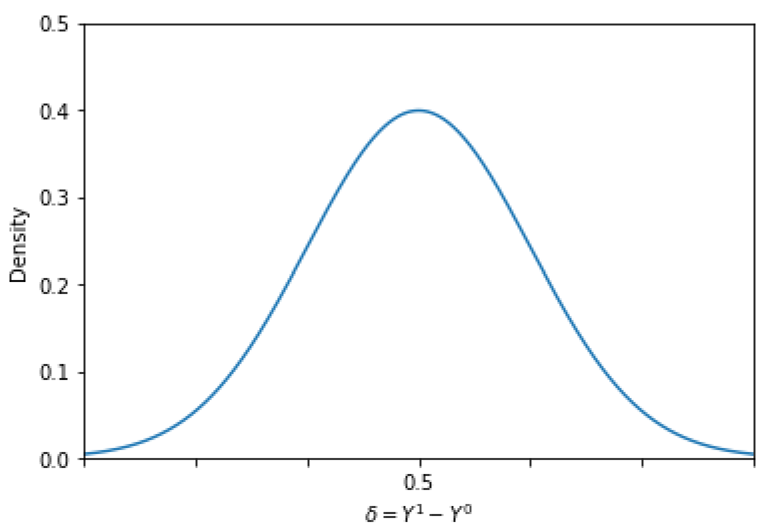
It would be great if we could get our hands on the individual-specific effect of treatment.

$$\delta_i = y_i^1 - y_i^0$$

- Why do individuals have potentially different effects of treatment?

[3]:
```python
fig, ax = plt.subplots()

x = np.linspace(-5, 5, 5000)
pdf = ss.norm.pdf(x, 0, 1)

ax.plot(x, pdf)

ax.set_xlabel(r"$\delta = Y^1 - Y^0$")
ax.set_ylabel("Density")
x_formatter = FixedFormatter(["", "", "", 0.5, "", "", ""])
x_locator = FixedLocator([-3, -2, -1, 0, 1, 2, 3])
ax.xaxis.set_major_locator(x_locator)
ax.xaxis.set_major_formatter(x_formatter)
ax.set_xlim([-3, 3])
ax.set_ylim([0, 0.5])
```

[3]:
```
(0.0, 0.5)
```



There might be considerable heterogeneity in the benefits of treatment among the population. And summarizing the distribution of benefits with a single number, for example $E[\delta]$, might result in a loss of information.

Examples

- medical treatment
-

Give our definitions of (Y^1, Y^0, D) and their individual realizations (y_i^1, y_i^0, d_i) we can now define the observed outcome Y in terms of them.

$$Y = \begin{cases} Y^1 & \text{if} \quad D = 1 \\ Y^0 & \text{if} \quad D = 0 \end{cases}$$

or more compactly in switching-regime notation

$$Y = DY^1 + (1 - D)Y^0.$$

This leads Holland (1986) to describe the fundamental problem of causal inference:

Group	Y^1	Y^0
Treatment group ($D = 1$)	Observable as Y	Counterfactual
Control group ($D = 0$)	Counterfactual	Observable as Y

$\rightarrow$ as only the diagonal of the table is observable we cannot simply compute δ_i by taking the difference in potential outcomes (y_i^1, y_i^0).

[4]:
```
df.head()
```

[4]:

	treat	re78	Y	Y_0	Y_1	D
101	1	9970.681	9970.681	NaN	9970.681	1
611	0	7094.920	7094.920	7094.920	NaN	0
396	0	11223.720	11223.720	11223.720	NaN	0
681	0	4687.937	4687.937	4687.937	NaN	0
397	0	5088.760	5088.760	5088.760	NaN	0

Population-level parameters

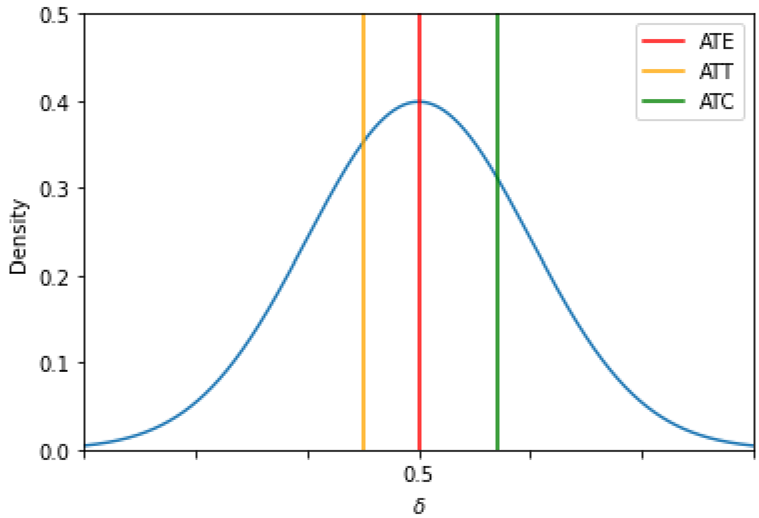
It looks like we need to give up any hope of obtaining the individual-specific effect of treatment. But what can we still hope for?

$\rightarrow$ population-level parameters

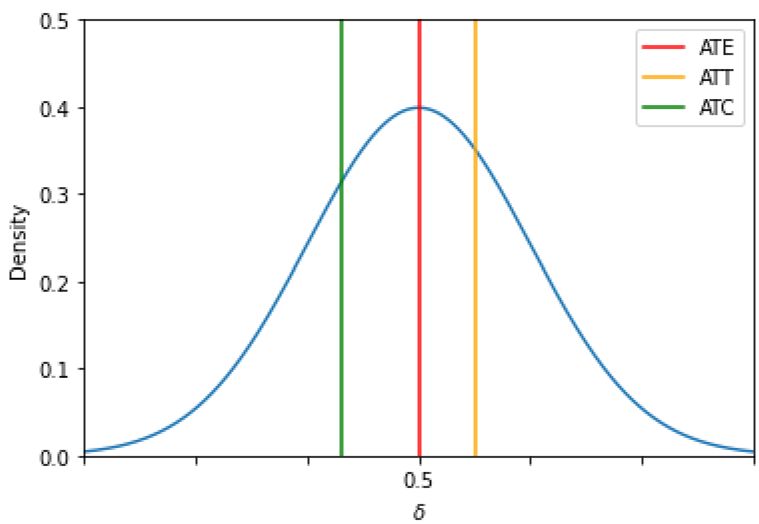
- What are common examples?
- What are the policy questions they address?
- What is their relationship to each other?

$$\begin{array}{lll} E[Y^1 - Y^0] & ATE & \text{average effect of treatment} \\ E[Y^1 - Y^0 \mid D = 1] & ATT & \text{average effect on treated} \\ E[Y^1 - Y^0 \mid D = 0] & ATC & \text{average effect on control} \end{array}$$

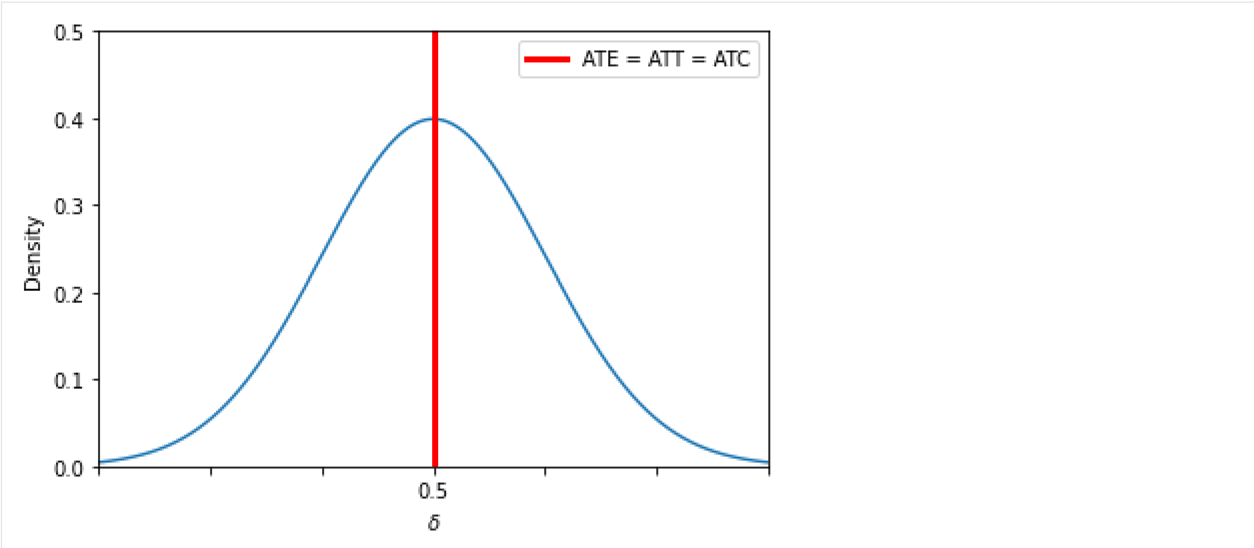
[5]:
```
plot_individual_specific_effects(with_parameters=[0, 0.7, -0.5])
```



[6]:
```
plot_individual_specific_effects(with_parameters=[0, -0.7, 0.5])
```



[6]:
```
plot_individual_specific_effects(with_parameters=[0, 0, 0])
```



Stable unit treatment value assumption

The potential outcome model gets its empirical tractability when combined with the **Stable Unit Treatment Value Assumption (SUTVA)** of which there exist many formulations. We will go with the one from Imbens and Rubin (2015):

> The potential outcomes for any unit do not vary with the treatments assigned to other units, and, for each unit there are no different forms or versions of each treatment level, which lead to different potential outcomes.

The table below shows all possible assignment patterns for a hypothetical treatment where the only constraint is that at least one individual remains in the treatment and control group. As we increase participation from one to two individuals, the potential outcome Y_1 declines.

Treatment assignment patterns	Potential outcomes	
$\begin{bmatrix} d_1 = 1 \\ d_2 = 0 \\ d_3 = 0 \end{bmatrix}$ or $\begin{bmatrix} d_1 = 0 \\ d_2 = 1 \\ d_3 = 0 \end{bmatrix}$ or $\begin{bmatrix} d_1 = 0 \\ d_2 = 0 \\ d_3 = 1 \end{bmatrix}$	$y_1^1 = 3$ $y_2^1 = 3$ $y_3^1 = 3$	$y_1^0 = 1$ $y_2^0 = 1$ $y_3^0 = 1$
$\begin{bmatrix} d_1 = 1 \\ d_2 = 1 \\ d_3 = 0 \end{bmatrix}$ or $\begin{bmatrix} d_1 = 0 \\ d_2 = 1 \\ d_3 = 1 \end{bmatrix}$ or $\begin{bmatrix} d_1 = 1 \\ d_2 = 0 \\ d_3 = 1 \end{bmatrix}$	$y_1^1 = 2$ $y_2^1 = 2$ $y_3^1 = 2$	$y_1^0 = 1$ $y_2^0 = 1$ $y_3^0 = 1$

- When do we need to expect this is violated?
  - **influence patterns** that result from contact across individuals in social or physical space
  - **dilution / concentration patterns** that one can assume would result from changes in the prevalence of treatment

Treatment assignment and observational studies

- randomized experiment

$$(Y^0, Y^1) \perp\!\!\!\perp D$$

- observational study

> A *observational study* is an empirical investigation of treatments, policies, or exposures and the effects they cause, but it differs from an experiment in that the investigator cannot control the assignment of treatments to subjects. (Rosenbaum (2002))

Naive estimation of average causal effects

We will now first outline the problem with the naive estimation of average causal effects. Then we take a closer look at the different sources of biases involved and finally discuss the set of assumptions used to ***solve*** these issues.

$$\hat{\delta}_{NAIVE} \equiv E_N[y_i \mid d_i = 1] - E_N[y_i \mid d_i = 0]$$

We can further decompose the average treatment effect by treatment status as the individual assignment is mutually exclusive.

$$\begin{aligned} E[Y^1 - Y^0] = E[\delta] \qquad & = \{\pi E[Y^1 \mid D = 1] + (1-\pi)E[Y^1 \mid D = 0]\} \\ & -\{\pi E[Y^0 \mid D = 1] + (1-\pi)E[Y^0 \mid D = 0]\} \end{aligned}$$

The average treatment effect is a function of five unknowns. Which components can be easily computed from data?

$$\begin{aligned} E_N[y_i \mid d_i = 1] \xrightarrow{p} E[Y^1 \mid D = 1] \neq E[Y^1] \\ E_N[y_i \mid d_i = 0] \xrightarrow{p} E[Y^0 \mid D = 0] \neq E[Y^0] \end{aligned}$$

Biases

$$\begin{aligned} E[Y^1 \mid D = 1] - E[Y^0 \mid D = 0] = E[\delta] + & \underbrace{\{E[Y^0 \mid D = 1] - E[Y^0 \mid D = 0]\}}_{\text{Baseline bias}} \\ + (1-\pi) & \underbrace{\{E[\delta \mid D = 1] - E[\delta \mid D = 0]\}}_{\text{Differential treatment effect bias}} \end{aligned}$$

Group	$E[Y^1 \vert .]$	$E[Y^0 \vert .]$
Treatment group ($D = 1$)	10	6
Control group ($D = 0$)	8	5

The additional information provided in the text states that $\pi = 0.3$ meaning that 30% of the sample participate in the treatment.

$$\begin{aligned} ATT &= E[Y_1 - Y_0 \mid D = 1] = 10 - 6 = 4 \\ ATC &= E[Y_1 - Y_0 \mid D = 0] = 8 - 5 = 3 \\ \delta^{NAIVE} &= E[Y_1 \mid D = 1] - E[Y_0 \mid D = 0] = 10 - 5 = 5 \end{aligned}$$

Now we are ready to calculate the average treatment effect:

$$\begin{aligned} ATE = E[Y_1 - Y_0] &= \pi\, E[Y_1 - Y_0 \mid D = 1] + (1-\pi)\, E[Y_1 - Y_0 \mid D = 0] \\ &= 0.3 \times 4 + 0.7 \times 3 = 3.3 \end{aligned}$$

Next, we can determine the different components of the bias.

$$\begin{aligned} \Delta^{\text{base}} &= E[Y^0 \mid D = 1] - E[Y^0 \mid D = 0] = 6 - 5 = 1 \\ \Delta^{\text{diff}} &= (1-\pi)\,(E[\delta \mid D = 1] - E[\delta \mid D = 0]) = 0.7\,((10 - 6) - (8 - 5)) = 0.7 \end{aligned}$$

There are several different representation of the bias when using the naive estimator of mean difference in observed outcomes by treatment status as an estimate for the effect of treatment. We continue with the exposition in Frölich & Sperlich (2019) and Heckman, Urzua, & Vytlacil (2006).

$$\begin{aligned} E[Y \mid D = 1] - E[Y \mid D = 0] &= E[Y^1 \mid D = 1] - E[Y^0 \mid D = 0] \\ &= E[Y^1 \mid D = 1] - E[Y^0 \mid D = 1] \\ &+ E[Y^0 \mid D = 1] - E[Y^0 \mid D = 0] \\ &= \underbrace{E[Y^1 - Y^0 \mid D = 1]}_{TT} + \underbrace{E[Y^0 \mid D = 1] - E[Y^0 \mid D = 0]}_{\text{Selection bias}} \end{aligned}$$

Now we can simply add and subtract $E[Y_1 - Y_0]$ to get the more economic version.

$$\begin{aligned} E[Y \mid D = 1] - E[Y \mid D = 0] &= \underbrace{E[Y^1 - Y^0]}_{ATE} \\ &+ \underbrace{E[Y^1 - Y^0 \mid D = 1] - E[Y^1 - Y^0]}_{\text{Sorting on gains}} \\ &+ \underbrace{E[Y^0 \mid D = 1] - E[Y^0 \mid D = 0]}_{\text{Sorting on levels}} \end{aligned}$$

Sorting on levels is simply a different phrase for selection bias.

The exposition in our core textbook is slightly different. Here the term **bias** has two separate components which are **baseline bias** and **differential treatment effect bias**. See the discussion in the book in the subsection on the typical inconsistency and bias of the naive estimator. The term baseline bias refers to the concept of sorting and levels and selection bias.

Differential treatment bias is defined as:

$$\begin{aligned} E[Y \mid D = 1] - E[Y \mid D = 0] &= \underbrace{E[\delta]}_{ATE} + \underbrace{\{E[Y^0 \mid D = 1] - E[Y^0 \mid D = 0]\}}_{\text{Baseline bias}} \\ &+ \underbrace{(1-\pi)\{E[\delta \mid D = 1] - E[\delta \mid D = 0]\}}_{\text{Differential treatment effect bias}} \end{aligned}$$

The last term is derived derived from the term describing selection on gains by the following decomposition.

$$E[Y^1 - Y^0] = \pi E[Y^1 - Y^0 \mid D = 1] + (1-\pi) E[Y^1 - Y^0 \mid D = 0]$$

It is interpreted as the difference in effects between treated and control weighted by the share of control individuals. It is probably best thought of as an increment to the first term describing the average effect of treatment.

Assumptions

So, the SUTVA assumption is only necessary but not sufficient to learn about the effect of treatment in light of the biases discussed above. We are still stuck with several unknowns that we need to compute the average effect of treatment.

Consider the following two assumptions:

$$\begin{aligned} E[Y^1 \mid D = 1] &= E[Y^1 \mid D = 0] \\ E[Y^0 \mid D = 1] &= E[Y^0 \mid D = 0] \end{aligned}$$

and recall our naive estimate

$$\begin{aligned} \hat{\delta}_{NAIVE} &= E_N[y_i \mid d_i = 1] - E_N[y_i \mid d_i = 0] \\ &\xrightarrow{p} E[Y^1 \mid D = 1] - E[Y^0 \mid D = 0] \end{aligned}$$

- What assumptions suffice to estimate the ATE with the naive estimator?
  - about potential outcomes for subsets of the population
  - about the treatment selection / assignment process

Missing data and imputation

This is an adopted example from Imbens & Rubin (2015).

[84]:
```
df = get_lalonde_data()
df.head()
```

[84]:

	treat	re78	Y	Y_0	Y_1	D
100	1	0.000	0.000	NaN	0.0	1
561	0	5670.820	5670.820	5670.820	NaN	0
130	1	0.000	0.000	NaN	0.0	1
318	0	0.000	0.000	0.000	NaN	0
687	0	7659.218	7659.218	7659.218	NaN	0

We can impute the missing values simply by their average counterpart.

[71]:
```
is_treated = df["D"] == 1

df.loc[~is_treated, "Y_1"] = df.loc[is_treated, "Y"].mean()
df.loc[is_treated, "Y_0"] = df.loc[~is_treated, "Y"].mean()
```

[50]:
```
df.head()
```

[50]:

	treat	re78	Y	Y_0	Y_1	D
479	0	6930.336	6930.336	6930.336	NaN	0
480	0	3795.799	3795.799	3795.799	NaN	0
343	0	0.000	0.000	0.000	NaN	0
690	0	2652.625	2652.625	2652.625	NaN	0
70	1	0.000	0.000	NaN	0.0	1

[72]:
```
initial_stat = (df["Y_1"] - df["Y_0"]).mean()
print(f"Our estimated treatment effect is {initial_stat:10.2f}")
```

```
Our estimated treatment effect is     886.30
```

However, this does not really account for any uncertainty in our estimate. Can we do better? We now switch to the imputation of the counterfactual outcome on the individual level.

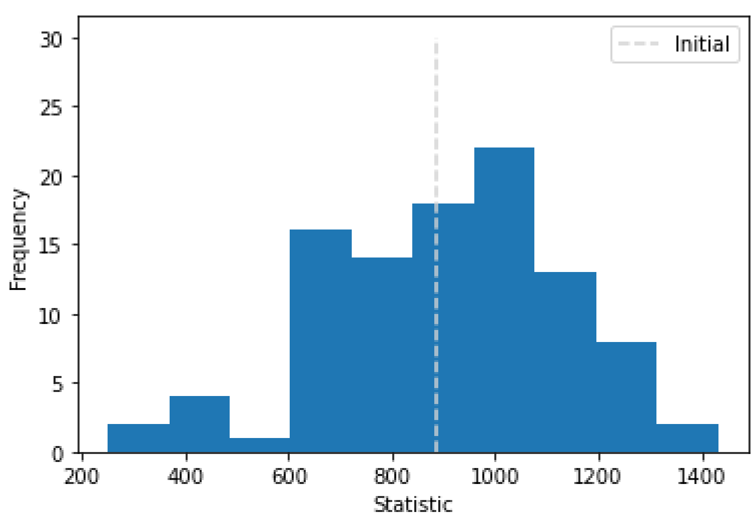
[80]:
```
np.random.seed(123)  # set seed to ensure reproducibility
df = get_lalonde_data()  # get the original data

status_counts = df["D"].value_counts().to_dict()

stats = list()
for _ in range(100):
    y_1_sampled = df["Y_1"].dropna().sample(n=status_counts[0], replace=True).values
    y_0_sampled = df["Y_0"].dropna().sample(n=status_counts[1], replace=True).values

    df_boot = df.copy()

    is_treated = df_boot["D"] == 1
    df_boot.loc[is_treated, "Y_0"] = y_0_sampled
    df_boot.loc[~is_treated, "Y_1"] = y_1_sampled

    stat = (df_boot["Y_1"] - df_boot["Y_0"]).mean()
    stats.append(stat)
print(f"Our estimated treatment effect is {np.mean(stats):10.2f}")
```

```
Our estimated treatment effect is     907.86
```

How does the full distribution of estimates look like?

[74]:
```
fig, ax = plt.subplots()
ax.hist(stats)
ax.set_xlabel("Statistic")
ax.set_ylabel("Frequency")
ax.vlines(initial_stat, 0, 30, linestyles="--", label="Initial", color="lightgrey")
ax.legend()
```

[74]:
```
<matplotlib.legend.Legend at 0x1e0ed15f7c0>
```



Still some limitations remains. For example, we do sample from the empirical distribution of the observed outcomes and not the actual distribution. Phrased differently, we treat the distribution of potential outcomes as known and abstract from any uncertainty in our knowledge about it.

Extensions of the binary potential outcome model

- over-time potential outcomes and causal effects
  - a single unit over time (time series data)
  - many units over time (panel data)
- many-valued treatments

Over-time potential outcomes

We explore the case of a single unit over time.

Ingredients

- discrete time periods, $t \in \{1, ..., T\}$
- sequence of observed values, $\{y_1, y_2, ..., y_T\}$
- treatment initiated in t^*
- duration of treatment k

Setting up the potential outcome model to explore the basic features of before-and-after designs for a single unit of analsysis.

- before the treatment is introduced (for $t < t^*$):

$$\begin{aligned} D_t &= 0 \\ Y_t &= Y_t^0 \end{aligned}$$

- while the treatment is in place (from t^* through $t^* + k$):

$$\begin{aligned} D_t &= 1 \\ Y_t &= Y_t^1 \\ Y_t^0 &\text{exists but is counterfactual} \end{aligned}$$

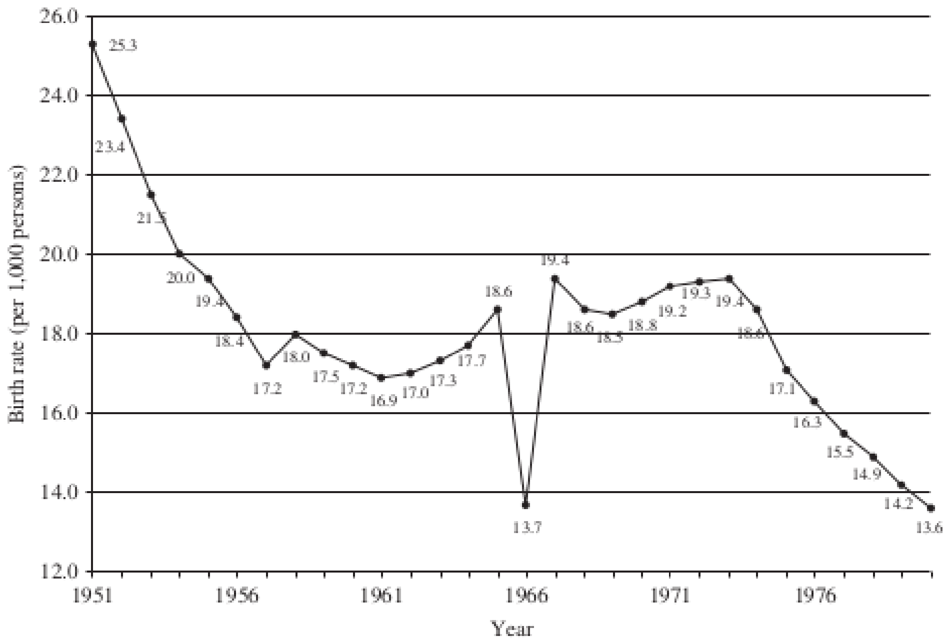
- after the treatment ends (for time periods $t > (t^* + k)$):

$$\begin{aligned} D_t &= 0 \\ Y_t &= Y_t^0 \\ Y_t^1 &\text{exists but is counterfactual} \end{aligned}$$

The following is an adapted example from our textbook.

Year of the fire horse

We study the effect of Japanese folk belief that families who give birth to babies will suffer untold miseries. This example does not only illustrative the versatility of the potential outcome framework but also serves as an example that different approaches (informed by domain-expertise) can result in different reasonable imputations for the counterfactual outcome.



The treatment indicator is as follows: $D_{1966} = 1$ and $D_{\neq 1966} = 0$ and we are interested in its effect on the birth rate in Japan

$$\delta_{1966} = y^1_{1966} - y^0_{1966}.$$

A reasonable approach is to estimate it by:

$$\hat{\delta}_{1966} = y_{1966} - y^0_{1966}$$

[85]:
```
df = pd.read_csv("material/world_bank.csv", skiprows=4)
df.set_index("Country Code", inplace=True)
df.drop(["Indicator Name", "Indicator Code"], axis=1, inplace=True)

df = df.loc["JPN", "1960":"2017"]
df = df.to_frame()
df.index.name = "Year"
df.columns = ["Birth rate"]

df.sort_index(inplace=True)
df.index = df.index.astype(int)
df.head()
```

[85]:
```
      Birth rate
Year
```

(continues on next page)

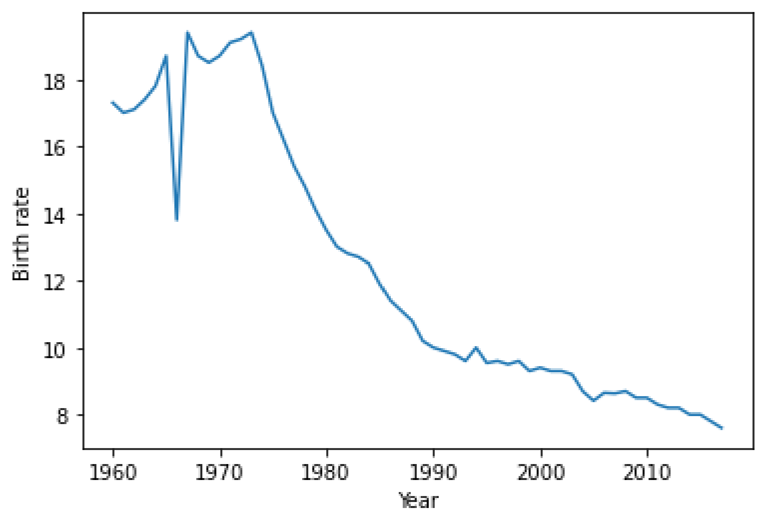
(continued from previous page)

```
1960        17.3
1961          17
1962        17.1
1963        17.4
1964        17.8
```

Let's get to work.

[86]:
```
fig, ax = plt.subplots()
ax.plot(df["Birth rate"].index, df["Birth rate"])
ax.set_ylabel("Birth rate")
ax.set_xlabel("Year")
```

[86]:
```
Text(0.5, 0, 'Year')
```



[87]:
```
df.loc[slice(1960, 1970), "Birth rate"]
```

[87]:
```
Year
1960    17.3
1961      17
1962    17.1
1963    17.4
1964    17.8
1965    18.7
1966    13.8
1967    19.4
1968    18.7
1969    18.5
1970    18.7
Name: Birth rate, dtype: object
```

We can just take the year before or after treatment?

[88]:
```
estimates = list()
for label, year in [("before", 1965), ("after", 1967)]:
    y_0 = df.loc[year, "Birth rate"]
```

(continues on next page)

(continued from previous page)

```
    y_1 = df.loc[1966, "Birth rate"]
    print(f" Using the year {label}, the treatment effect is {y_1 - y_0:10.5f}")
    estimates.append(y_1 - y_0)
```

```
Using the year before, the treatment effect is   -4.90000
Using the year after, the treatment effect is   -5.60000
```

Among demographers, there is the consensus that taking the average of 1963 and 1969 the way to go instead.

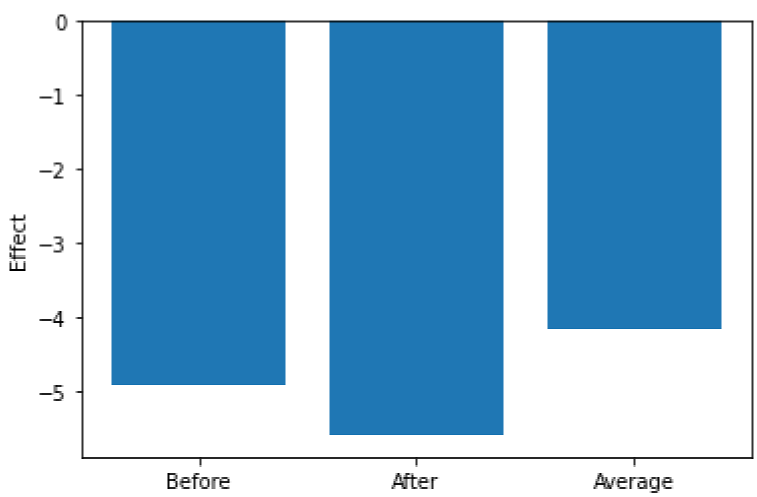
[89]:
```
y_0 = df.loc[[1963, 1969], "Birth rate"].mean()
y_1 = df.loc[1966, "Birth rate"]
print(" Another treatment effect is {:10.5f}".format(y_1 - y_0))
estimates.append(y_1 - y_0)
```

```
Another treatment effect is   -4.15000
```

Now we have multiple effects of treatment. Which is it?

[90]:
```
labels = ["Before", "After", "Average"]
fig, ax = plt.subplots()
ax.bar(labels, estimates)
ax.set_ylabel("Effect")
```

[90]:
```
Text(0, 0.5, 'Effect')
```



Additional resources

- **Imbens, G. W. (2020)**. Potential outcome and directed acyclic graph approaches to causality: Relevance for empirical practice in economics, *Journal of Economic Literature*, 58(4), 1129-79.

Resources

- **Frölich, M., and Sperlich, S. (2019)** . Impact evaluation: Treatment effects and causal analysis. Cambridge, England: *Cambridge University Press*.
- **Heckman, J. J., Urzua, S. and Vytlacil, E. (2006)**. Understanding instrumental variables in models with essential heterogeneity. *Review of Economics and Statistics*, 88(3), 389–432.
- **Holland, P. W. (1986)**. Statistics and causal inference. *Journal of the American Statistical Association*, 81(396), 945–960.
- **Imbens, G. W., and Rubin, D. B. (2015)**. Causal inference in statistics, social, and biomedical sciences. New York, NY: *Cambridge University Press*.
- **Rosenbaum, P. R. (2002)**. Overt bias in observational studies. *Observational studies*, 71–104.

1.3 Causal graphs

We explore the usefulness of causal graphs for the visualization of complex causal systems and the clarification of alternative identification strategies for causal effects. After establishing their basic notation and some key concepts, we link them to structural equations and the potential outcome model.

1.3.1 Causal graphs

Introduction

Graph notation less general than potential outcome framework, but

- thinking about causal systems
- uncover identification strategies

> It is useful to separate the inferential problem into statistical and identification components. Studies of identification seek to characterize the conclusions that could be drawn if one could use the sampling process to obtain an unlimited number of observations. (Manski, 1995)

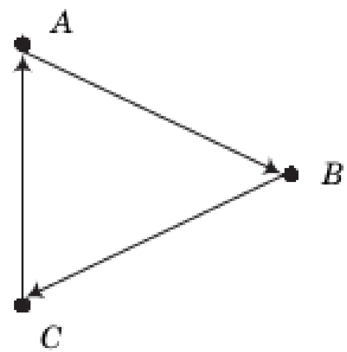
The two most crucial ingredients for an identification analysis are:

- The set of assumptions about causal relationships that the analysis is willing to assert based on theory and past research, including assumptions about relationships between variables that have not been observed but that are related both to the cause and outcome of interest.
- The pattern of information one can assume would be contained in the joint distribution of the variables (**associations**) in the observed dataset if all members of the population had been included in the sample that generated the dataset.

$\rightarrow$ causal graphs offer an effective and efficient representation for both

Basic elements of causal graphs

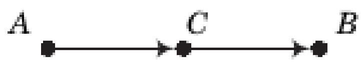
- nodes
- edges
- directed paths
  - parent and child
  - descendant



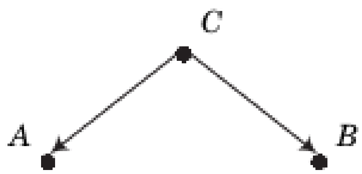
Two representations of the joint dependence of A and B on an unobserved common cause.

Let's look at some basic patterns that will turn out to appear frequently.

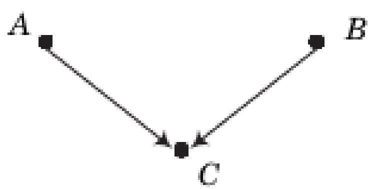
- chain of mediation
- fork of mutual causation
- inverted fork of mutual dependence



(a) Mediation



(b) Mutual dependence



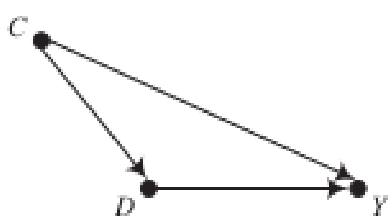
(c) Mutual causation

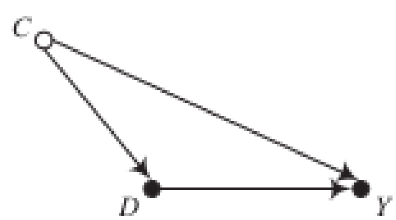
What about the unconditional and conditional association of A and B in each of these cases?

- While there is unconditional dependence between them in the first two cases, there is not in the third.

The **collider variable** C in the third setting does not generate an unconditional association between A and B. However, as we will revisit in more detail later, it can create a conditional association that needs to be handled with care.

Conditioning and confounding



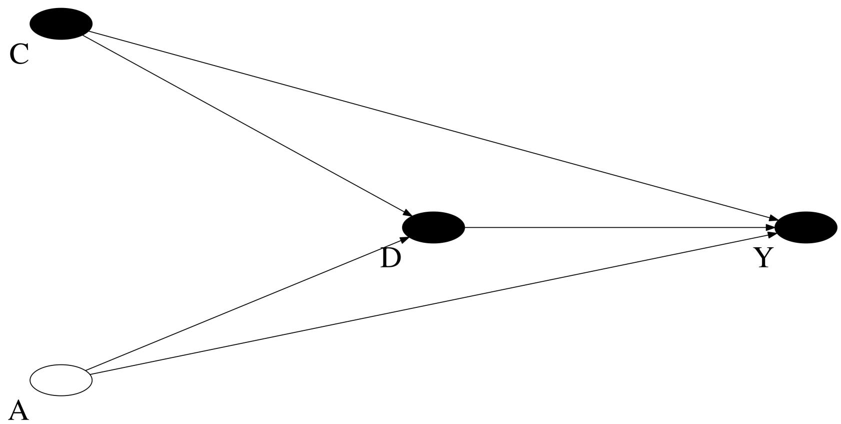


The causal effects $C \rightarrow D$ and $C \rightarrow Y$ render the total association between D and Y unequal to the causal effect $D \rightarrow Y$.

- C is a **confounding variable** that affects both the dependent and independent variable.
- Conditioning is a modelig strategy that allows to determine causal effects in the presence of observed confounders.

$\rightarrow$ What happens if C is unobserved?

How about an example from educational choice where we have observed and unobserved confounders?



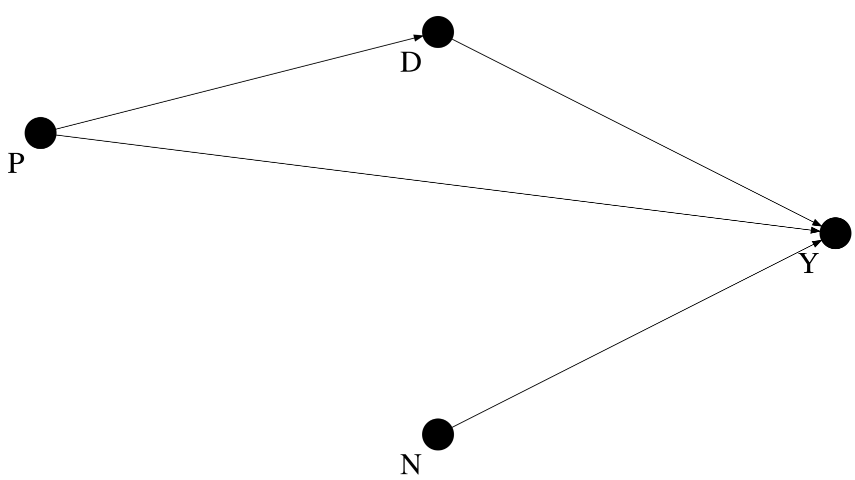
What identification strategies come to mind?

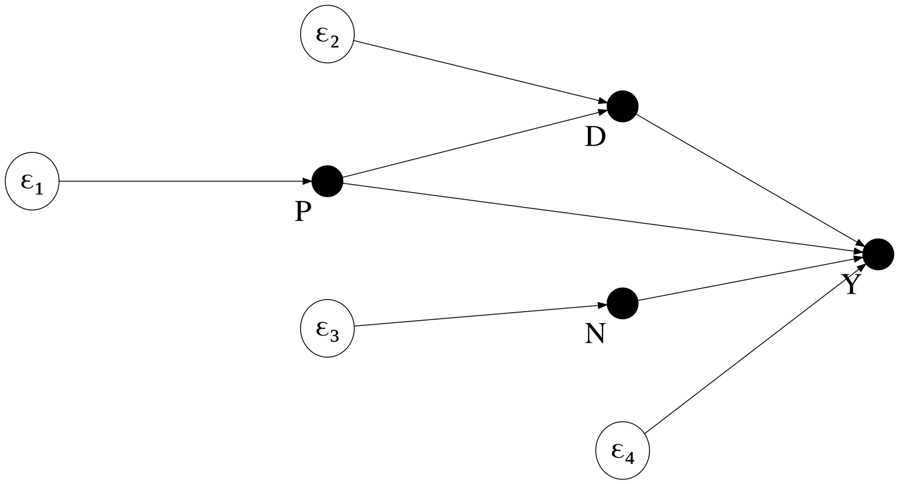
Link to structural equations

Let's look at another example and assume we are interested in the effect of parental background (P), charter schools (D), and neighborhoods (N) on test scores (Y).

We could set up the following **linear** regression equations:

$$D = \alpha_D + b_P P + \epsilon_2$$
$$Y = \alpha_Y + b_D D + b_P P + + b_N N + \epsilon_4$$





We can set up the same *nonparametric* structural equations for both representations:

$$
\begin{aligned}
P &= f_P(\epsilon_1)\\
N &= f_N(\epsilon_3)\\
D &= f_D(P, \epsilon_2)\\
Y &= f_Y(P, D, N, \epsilon_4)
\end{aligned}
$$

How to simulate a sample from a set of structural equations?

[2]:
```
indices = list()
[indices.append(label) for label in product([("alpha")], ("D", "Y"))]
[indices.append(label) for label in product([("beta")], ("P", "N", "D"))]
index = pd.MultiIndex.from_tuples(indices, names=["group", "element"])

values = [1, 1, 0.8, 0.7, -0.3]
params = pd.Series(values, index=index)

# distributional assumptions
get_unobservable = np.random.normal
get_observable = np.random.uniform


num_agents = 10000

df = pd.DataFrame(columns=["Y", "D", "P", "N"])

for i in range(num_agents):
    P, N = get_observable(size=2)

    D = params.loc["alpha", "D"] + params.loc["beta", "P"] * P + get_unobservable()

    Y = (
        params.loc["alpha", "Y"]
        + params.loc["beta", "D"] * D
        + params.loc["beta", "P"] * P
```

(continues on next page)

(continued from previous page)

```
        + params.loc["beta", "N"] * N
        + get_unobservable()
    )

    df.loc[i] = [Y, D, P, N]


df.head()
```

[2]:
```
          Y         D         P         N
0  1.248681  3.289100  0.928133  0.297718
1  1.791849  1.665443  0.221006  0.472631
2  1.181537  0.400000  0.366633  0.706849
3  1.875180  1.226666  0.189232  0.916127
4  3.442131  1.271353  0.025105  0.976486
```

Now lets see if we can uncover the structural parameters by a simple ordinary-least-squares regression and thus go full circle from a parametric structural equation model to a causal graph.

[3]:
```
params
```

[3]:
```
group  element
alpha  D          1.0
       Y          1.0
beta   P          0.8
       N          0.7
       D         -0.3
dtype: float64
```

[4]:
```
smf.ols(formula="Y ~ D + P + N", data=df).fit().summary()
```

[4]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                      Y   R-squared:                       0.143
Model:                            OLS   Adj. R-squared:                  0.143
Method:                 Least Squares   F-statistic:                     556.2
Date:                Tue, 04 May 2021   Prob (F-statistic):               0.00
Time:                        21:05:08   Log-Likelihood:                -14101.
No. Observations:               10000   AIC:                         2.821e+04
Df Residuals:                    9996   BIC:                         2.824e+04
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.9686      0.028     34.853      0.000       0.914       1.023
D             -0.2997      0.010    -30.384      0.000      -0.319      -0.280
P              0.8276      0.035     23.439      0.000       0.758       0.897
N              0.7401      0.034     21.574      0.000       0.673       0.807
==============================================================================
Omnibus:                        2.466   Durbin-Watson:                   1.998
```

(continues on next page)

(continued from previous page)

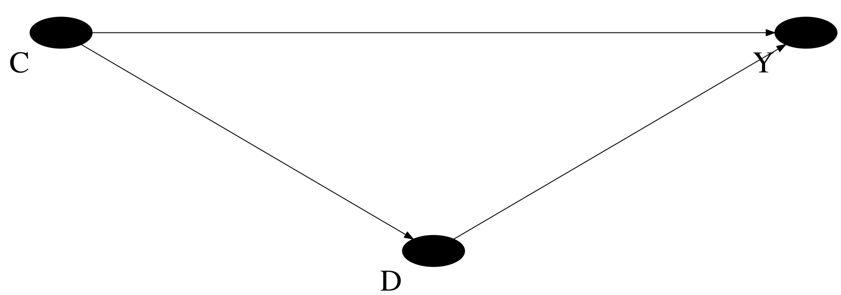
```
Prob(Omnibus):                  0.291   Jarque-Bera (JB):               2.400
Skew:                          -0.010   Prob(JB):                       0.301
Kurtosis:                       2.927   Cond. No.                        8.68
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly␣
↪specified.
"""
```

Link to potential outcome model

Advantages of the potential outcome model

- definition of causal effects
- individual effects as first principle
- decomposition of sources of inconsistency
- ...

However, it is hard to manage the notion for larger causal systems with many confounding variables and treatments.



Based on our previous discussion, unfortunately, $E[Y_1 - Y_0] \neq E[Y \mid D = 1] - E[Y \mid D = 0]$.

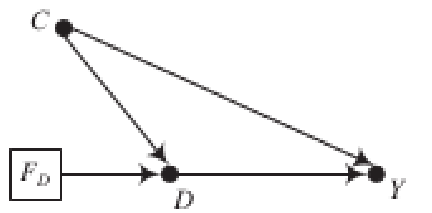
How can we define the treatment effects from the potential outcome model in here?

> Interventions and counterfactuals are defined through a mathematical operator called $do(\cdot)$, which simulates physical interventions by deleting certain functions from the model, replacing them with a constant. (Pearl, 2012)

$$E[Y_1 - Y_0] \quad \text{corresponds to} \quad E[Y \mid do(D = 1)] - E[Y \mid do(D = 0)]$$

The $do(\cdot)$ operator is the exact analog to the superscripts given to potential outcomes in order to designate the underlying causal states that define them.

Graphical presentation of $do(\cdot)$ operator



(a) Augmented casual graph with a "forcing" variable that represents an intervention

The $do(\cdot)$ operator induces a key distinction between the **conditional distribution** of the endogenous variable and its **interventional distribution**.

Let's simulate a sample from a parametrized version of the graph above.

[5]:
```python
np.random.seed(123)

num_agents = 1000
df = pd.DataFrame(columns=["Y", "D", "C"])


def calculate_outcome(C, D):
    """We compute the observed outcome."""
    # If you would like to have it in potential
    # outcome notation.
    Y_1 = 1 + C
    Y_0 = 0 + C
    Y = D * Y_1 + (1 - D) * Y_0

    # So what is the individual treatment effect?:

    return Y


for i in range(num_agents):
    C = np.random.uniform()
    D = np.random.choice([0, 1], p=[C, 1 - C])
    Y = calculate_outcome(C, D)
    df.loc[i] = [Y, D, C]

df.head()
```

[5]:

	Y	D	C

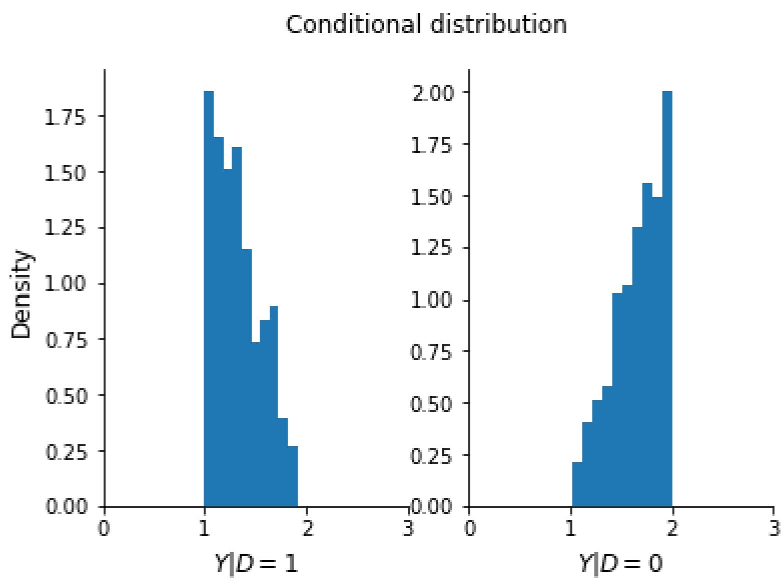
(continues on next page)

(continued from previous page)

```
0  1.696469  0.0  0.696469
1  1.226851  1.0  0.226851
2  1.719469  0.0  0.719469
3  1.980764  0.0  0.980764
4  1.480932  0.0  0.480932
```

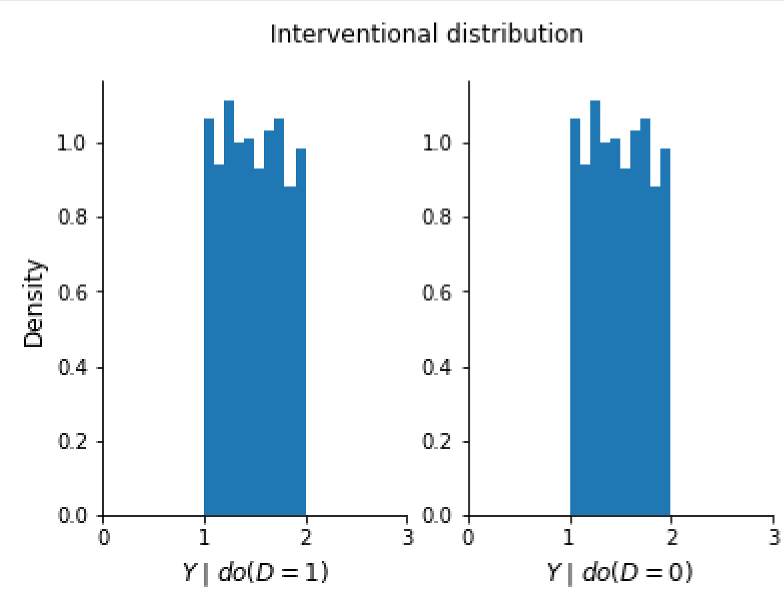
We know how to compute and plot a **conditional distribution**.

[6]:
```
plot_conditional_distribution(df)
```



How can we compute the **interventional distribution**? What do we need to know to do that?

[7]:
```
Y_do_1, Y_do_0 = list(), list()
for i, row in df.iterrows():
    # Note that we calculate the outcome using the
    # individual"s actual C put simply set D to
    # its value unter the intervetion.
    C, D = row["C"], 1
    Y_do_1 += [calculate_outcome(C, D)]

    C, D = row["C"], 0
    Y_do_0 += [calculate_outcome(C, D)]

plot_interventional_distribution(Y_do_1, Y_do_0)
```



Resources

- **Manski, C. F. (1995)**. Identification problems in the social sciences. Cambridge, UK: *Harvard University Press*.
- **Pearl, J. (2012)**. The do-calculus revisited.
- **Peters, J., Janzig, D., and Schölkopf, B. (2018)**. Elements of causal inference: Foundations and learning algorithms. Cambridge, MA: *The MIT Press*.
- **Imbens, G. W. (2020)**. Potential outcome and directed acyclic graph approaches to causality: Relevance for empirical practice in economics. *Journal of Economic Literature*, 58(4).
- **Hünermund, P. and Bareinboim, E. (2019)**. Causal inference and data-fusion in econometrics. *arXiv preprint arXiv:1912.09104*.
- **Pearl, J. (2009)**. Causal inference in statistics: An overview. *Statistics Surveys*, 3, 96-146.

1.4 Randomized Experiments

A lecture on randomized experiments will be part of the next iteration of the OSE data science course, summer semester 2022. Details on this lecture will be realized soon.

1.4.1 Randomized experiments

- **Athey, S., & Imbens, G. (2017)**. Chapter 3 - The econometrics of randomized experiments, in *Handbook of Economic Field Experiments*, 73-140.
- **Freedman, D.A. (2008)**. On regression adjustments to experimental data, *Advances in Applied Mathematics*, 40(2), 180-193.

1.5 Conditioning estimators

We study the basic conditioning strategy for the estimation of causal effects. We first link the concept of conditioning to direct graphs and start discussing the concept of a back-door path. Then we illustrate in a simulated example how collider variables induce a conditional association between two independent variables. Finally, we discuss the back-door criterion and work through some examples.

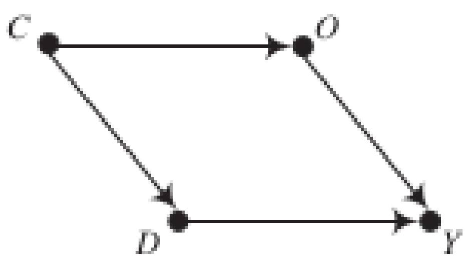
1.5.1 Conditioning estimators

Introduction

Approaches to the estimation of causal effects

- conditioning on variable that block all back-door paths from the causal variable to the outcome variable
- using exogenous variation in an appropriate instrumental variable to isolate covariation in the causal variable and the outcome variable
- establishing the exhaustive and isolated mechanism that intercepts the effect of the causal variable on the outcome variable and then calculating the causal effect as it propagates through the mechanisms

Conditioning and directed graphs



This graph is an example where a simple mean-comparison between the treated and untreated is not informative on the effect of the treatment.

- The total association between D and Y is an unknown composite of the true causal effect $D \rightarrow Y$ and the noncausal association between D and Y.

Conditioning strategies

- balancing the determinants of treatment assignment (e.g. matching estimators)
- adjusting for all other causes of the outcome (e.g. regression estimators)

Back-door path

A back-door path is a path between any causally ordered sequence of two variables that begins with a directed edge that points to the first variable. In the example above, we have two paths: (1) $D \rightarrow Y$, and (2) $D \leftarrow C \rightarrow O \rightarrow Y$. The former is a **causal path**, while the latter is a **back-door path**.

LaLonde dataset

What was the graph behind our analysis of the Lalonde dataset?

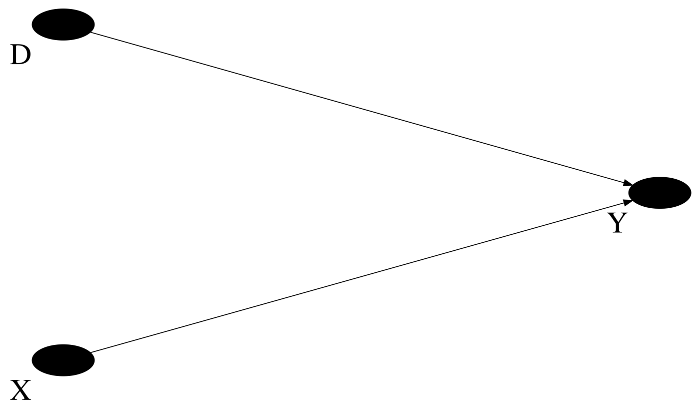
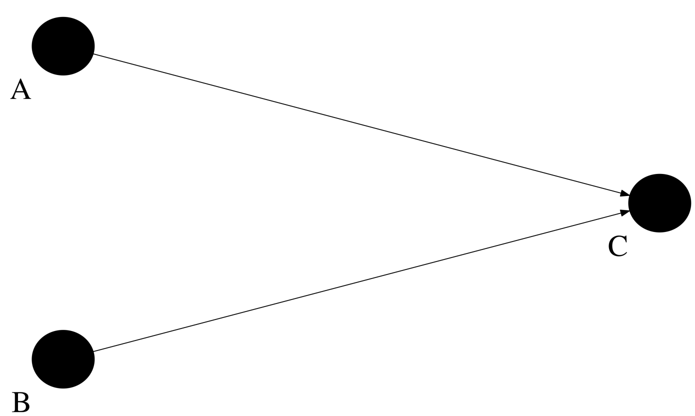


Illustration of collider variables

We introduced **collider variables** earlier. However, they will play a very important role going forward as conditioning on a collider variable that lies along an back-door path does not help to block that path, but instead creates new associations. Thus, we initially study in an illustration how conditioning on a collider induces a conditional association between two variables without an unconditional association.



[2]:
```
num_individuals = 250

# Initialize empty data frame
columns = ["SAT", "motivation", "admission"]
df = pd.DataFrame(columns=columns, index=range(num_individuals))

df["motivation"] = np.random.normal(size=num_individuals)
df["SAT"] = np.random.normal(size=num_individuals)
```

(continues on next page)

(continued from previous page)

```
# Both toghether determine college admission
score = df["motivation"] + df["SAT"]
cutoff = np.percentile(df["motivation"] + df["SAT"], 85)
df["admission"] = score > cutoff
df.head()
```

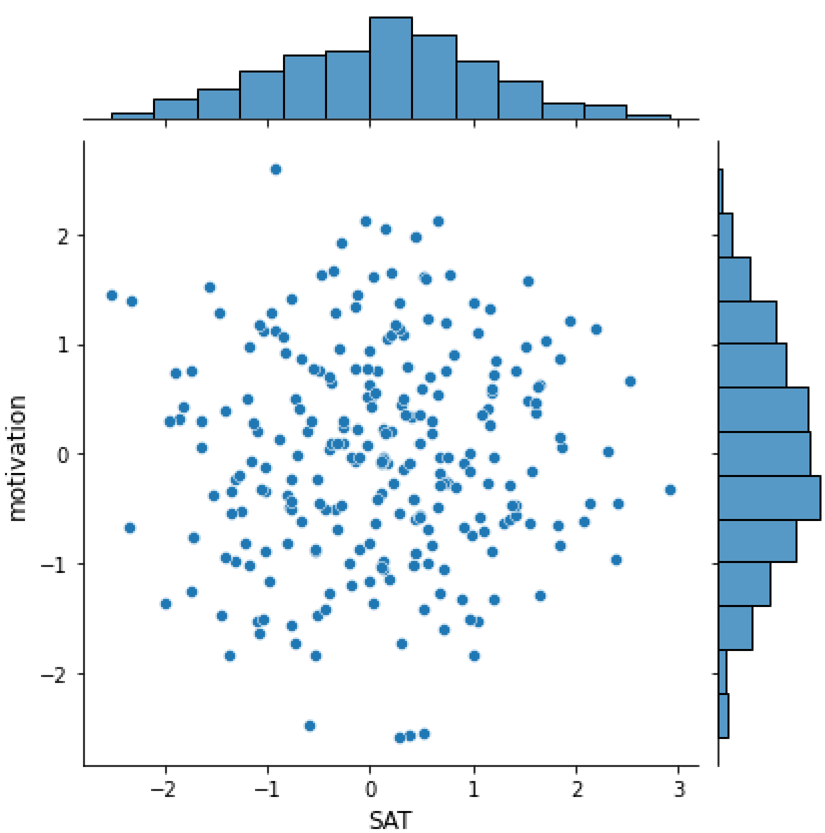
[2]:

```
        SAT  motivation  admission
0 -1.030933    1.127653      False
1  0.163968    1.051246      False
2  0.446524   -0.906215      False
3  0.441111    1.977908       True
4  1.190139    0.551189       True
```

[3]:

```
def get_joint_distribution(df):
    sns.jointplot(x="SAT", y="motivation", data=df)

    stat = stats.pearsonr(df["SAT"], df["motivation"])[0]
    print(f"The Pearson correlation coefficient is {stat:7.3f}")


get_joint_distribution(df)
```

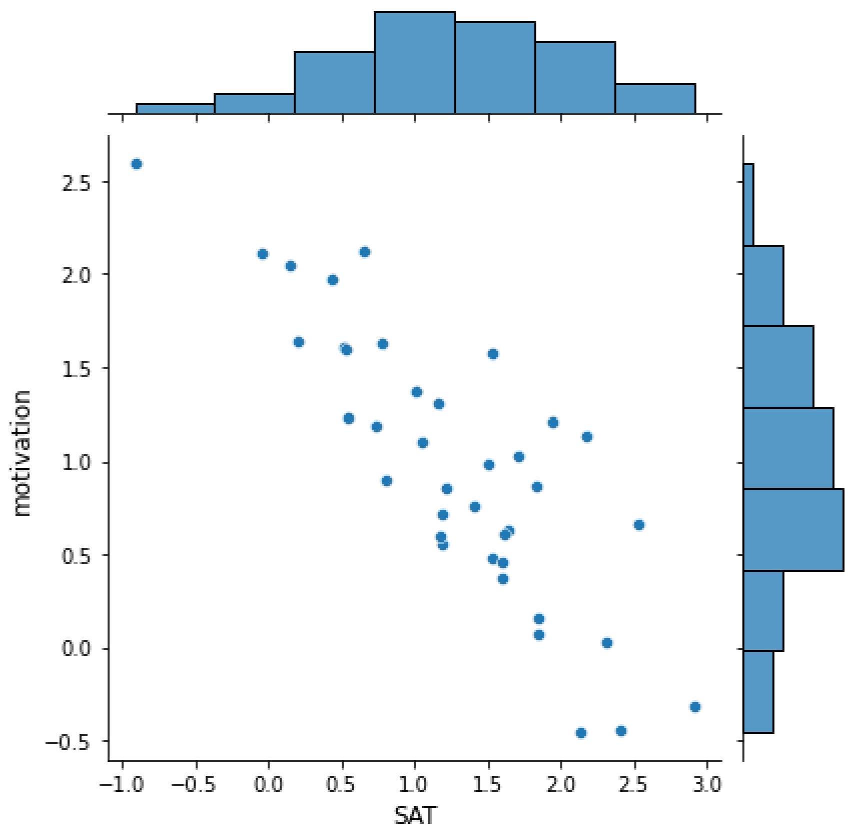
```
The Pearson correlation coefficient is   0.023
```



What happens if we condition on college admittance C, i.e. on a collider variable?

```
[4]: get_joint_distribution(df.query("admission == True"))
```

```
The Pearson correlation coefficient is  -0.836
```



Conditioning on a collider variable that lies along a back-door path does not help to block the back-door path but instead creates new associations.

The back-door criterion

The **back-door** criterion allows to determine the whether or not conditioning on a given set of observed variables will identify the causal effect of interest.

- **Step 1** Write down the back-door paths from the causal variable to the outcome variable, determine which ones are unblocked, and then search for a candidate conditioning set of observed variables that will block all unblocked back-door paths.
- **Step 2** If a candidate conditioning set is found that blocks all back-door paths, inspect the patterns of decent in the graph in order to verify that the variables in the candidate conditioning set do not block or otherwise adjust away any portion of the causal effect of interest.

If one or more back-door paths connect the causal variable to the outcome variable, the causal effect is identified by conditioning on a set of variables Z if

Condition 1 All back-door paths between the causal variable and the outcome variable are blocked after conditioning on Z, which will always be the case if each back-door path

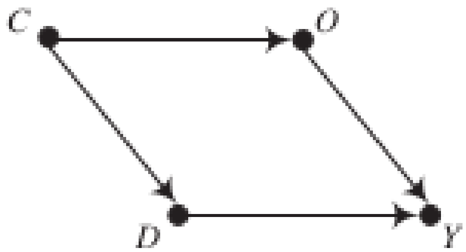
- contains a chain of mediation $A \rightarrow C \rightarrow B$ where the middle variable C is in Z
- contains a fork of mutual dependence $A \leftarrow C \rightarrow B$, where the middle variable C is in Z

- contains an inverted fork of mutual causation $A \rightarrow C \leftarrow B$, where the middle variable C and all of C's decendents are **not** in Z

and …

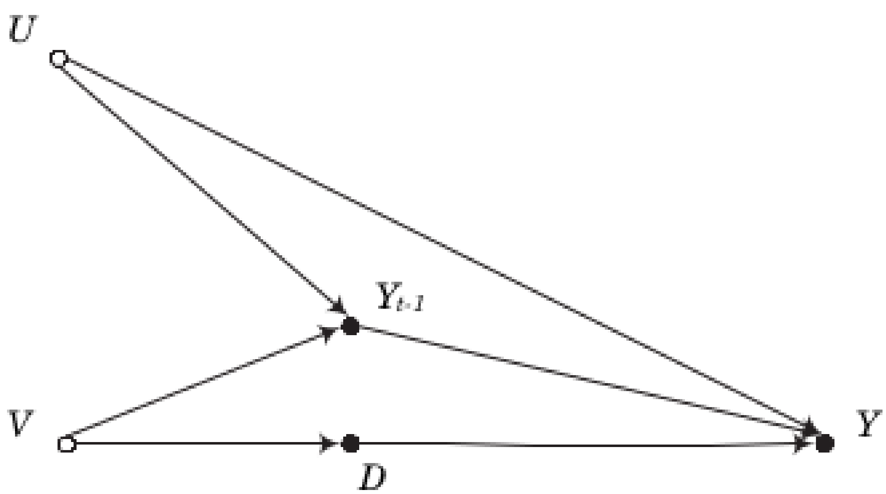
Condition 2 No variables in Z are decendents of the causal variable that lie on (or decend from other variables that lie on) any of the directed paths that begin at the causal variable and reach the outcome variable.

Let's revisit our example earlier and test our vocabulary.



We have a **chain of mediation** from $C \rightarrow O \rightarrow Y$ and a **fork of mutual dependence** with $D \leftarrow C \rightarrow O$.

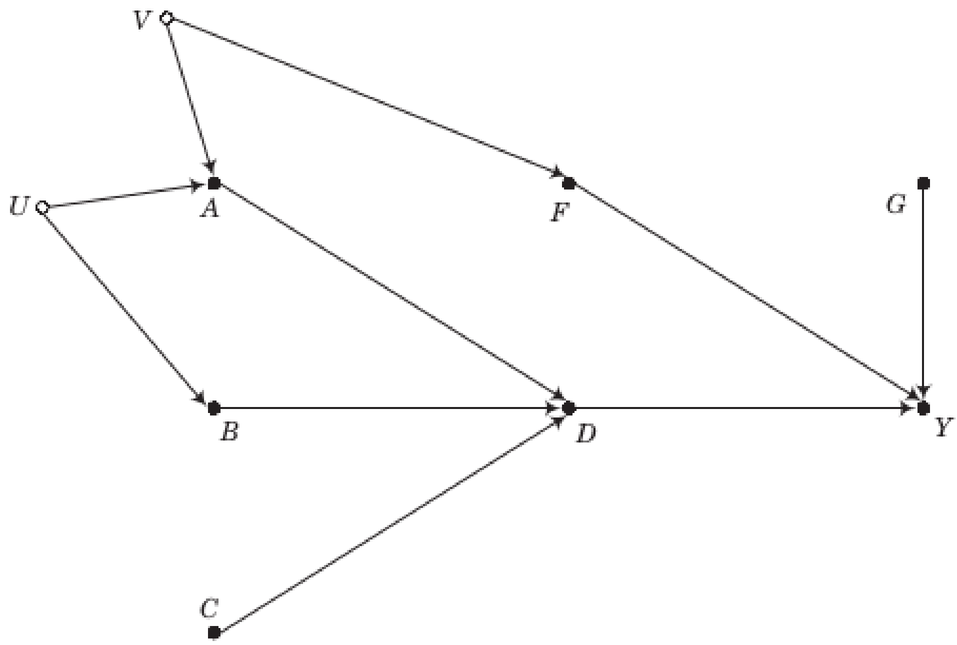
We will now work through two more advanced examples where we focus on only the first conditions of the back-door criterion. Let's start with a simple example and apply the idea of back-door identification to a graph where we consider conditioning on a lagged outcome variable Y_{t-1}.



There exist two back-door paths and Y_{t-1} lies on both of them. However, conditioning on it does not satisfy the back-door criterion. It blocks one path. Y_{t-1} is a collider variable on one of the paths.

Let us practice our understanding for some interesting graph structures. The backdoor algorithm is also available here for your reference.

Let's study the following causal graph:



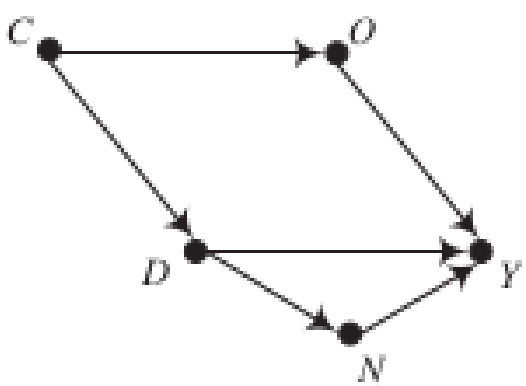
Consider the following three candidate conditioning sets. Any thoughts?

- $\{F\}$
- $\{A\}$
- $\{A, B\}$

Finally, let's focus on the second condition.

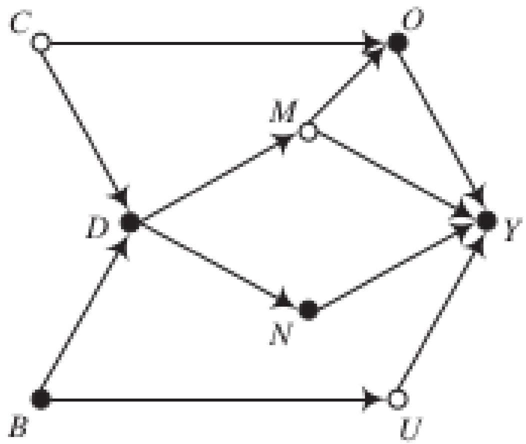
- **Condition 2** No variables in Z are decendents of the causal variable that lie on (or decend from other variables that lie on) any of the directed paths that begin at the causal variable and reach the outcome variable.

We first look at a graph that illustrates what a descendent is and remind ourselves of the difference between a direct and an indirect effect.



Conditioning on N (in addition to either C or O) does not satisfy the back-door criterion due to its violation of the second condition.

How about this causal structure:



Let's evaluate the candidate conditioning set $\{O, B\}$ together.

By now you probably recognized the mechanical nature of checking the back-door criterion **for a given causal graph**. Here are some automated tools to make your life easier in the future, but also allow you to practice your own understanding.

- DAGitty — draw and analyze causal diagrams

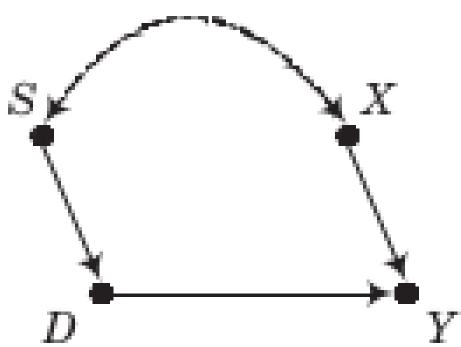
1.6 Matching estimators

We review the fundamental concepts of matching such as stratification of data, weighting to achieve balance, and propensity scores. We explore several alternative implementations as we consider matching as conditioning via stratification, matching as a weighing approach, and matching as a data analysis algorithm. Throughout we heavily rely on simulated examples to explore some practical issues such as sparsity of data.

1.6.1 Matching estimators of causal effects

Introduction

There exists only one back-door path $D \leftarrow S \leftrightarrow X \rightarrow Y$ and both S nor X are observable. Thus, we have a choice to condition on either one of them.



- X, regression estimator, adjustment-for-other-causes conditioning strategy
- S, matching estimator, balancing conditioning strategy

Agenda

- matching as conditioning via stratification
- matching as weighting
- matching as data analysis algorithm

Fundamental concepts

- stratification of data
- weighting to achieve balance
- propensity scores

Views on matching

- method to form quasi-experimental contrasts by sampling comparable treatment and control cases
- nonparametric method of adjustment for treatment assignment patterns

Simulation data

The simulated data is inspired by real-world applications and thus rather complex. Nevertheless, the will serve as examples for several of the upcoming lectures. That is why we will invest some time initially to set up one of them in details.

Matching as conditioning via stratification

Individuals within groups determined by S are entirely indistinguishable from each other in all ways except

- observed treatment status
- differences in potential outcomes that are independent of treatment status

More formally, we are able to assert the following **conditional independence assumptions**.

$$E[Y^1 \mid D = 1, S] = E[Y^1 \mid D = 0, S] \tag{1.1}$$
$$E[Y^0 \mid D = 1, S] = E[Y^0 \mid D = 0, S] \tag{1.2}$$

implied by ...

- treatment assignment is ignorable
- selection on observables

ATC

$$\begin{aligned}
E[\delta \mid D = 0, S] &= E[Y^1 - Y^0 \mid D = 0, S] \\
&= E[Y^1 \mid D = 0, S] - E[Y^0 \mid D = 0, S] \\
&= E[Y^1 \mid D = 1, S] - E[Y^0 \mid D = 0, S] \\
&= E[Y \mid D = 1, S] - E[Y \mid D = 0, S]
\end{aligned}$$

ATT

$$\begin{aligned}
E[\delta \mid D = 1, S] &= E[Y^1 - Y^0 \mid D = 1, S] \\
&= E[Y^1 \mid D = 1, S] - E[Y^0 \mid D = 1, S] \\
&= E[Y \mid D = 1, S] - E[Y \mid D = 0, S]
\end{aligned}$$

Note that each of the two derivations above, requires only one of the two conditional independence assumptions.

Let's turn to our first simulation exercise:

Table 5.1 The Joint Probability Distribution and Conditional Population Expectations for Matching Demonstration 1

	Joint probability distribution of S and D: $D=0$	$D=1$	
$S=1$	$\Pr[S=1,D=0]=.36$	$\Pr[S=1,D=1]=.08$	$\Pr[S=1]=.44$
$S=2$	$\Pr[S=2,D=0]=.12$	$\Pr[S=2,D=1]=.12$	$\Pr[S=2]=.24$
$S=3$	$\Pr[S=3,D=0]=.12$	$\Pr[S=3,D=1]=.2$	$\Pr[S=3]=.32$
	$\Pr[D=0]=.6$	$\Pr[D=1]=.4$	

	Potential outcomes: Under the control state	Under the treatment state	
$S=1$	$E[Y^0 \mid S=1]=2$	$E[Y^1 \mid S=1]=4$	$E[Y^1-Y^0 \mid S=1]=2$
$S=2$	$E[Y^0 \mid S=2]=6$	$E[Y^1 \mid S=2]=8$	$E[Y^1-Y^0 \mid S=2]=2$
$S=3$	$E[Y^0 \mid S=3]=10$	$E[Y^1 \mid S=3]=14$	$E[Y^1-Y^0 \mid S=3]=4$

All the things we can learn:

- naive estimate
- average effect of treatment
- average effect of treatment on controls
- average effect of treatment on treated

Notable features

- The gains from treatment participation differ in each stratum and those that have the most to gain are more likely to participate. So unconditional independence between D and (Y^1, Y^2) does not hold.

Let's study these idealized conditions for a simulated dataset.

[2]:
```
def get_sample_matching_demonstration_1(num_agents):
    """Simulate sample

    Simulates a sample based for mathcing demonstration one using the information␣
↪provided
    in Table 6.1.

    Args:
        num_agents: An integer that specifies the number of individuals
            to sample.

    Returns:
        Returns a dataframe with the observables (Y, S, D) as well as
        the unobservables (Y_1, Y_0).
    """

    def get_potential_outcomes(s):
        """Get potential outcomes.
```

(continues on next page)

(continued from previous page)

```
        Assigns the potential outcomes based on the observable S and
        the information in Table 6.1.

        Notes:
            The two potential outcomes are solely a function of the
            observable and are not associated with the treatment
            variable D.

        Args:
            s: an integer for the value of the stratification variable

        Returns:
            A tuple with the two potential outcomes.
        """
        if s == 1:
            y_1, y_0 = 4, 2
        elif s == 2:
            y_1, y_0 = 8, 6
        elif s == 3:
            y_1, y_0 = 14, 10
        else:
            raise AssertionError

        # We want some randomness.
        y_1 += np.random.normal()
        y_0 += np.random.normal()

        return y_1, y_0

    # Store some information about the sample variables
    # and initialize an empty dataframe.
    info = OrderedDict()
    info["Y"] = float
    info["D"] = int
    info["S"] = int
    info["Y_1"] = float
    info["Y_0"] = float

    df = pd.DataFrame(columns=info.keys())

    for i in range(num_agents):
        # Simulate from the joint distribution of the
        # observables.
        deviates = list(product(range(1, 4), range(2)))
        probs = [0.36, 0.08, 0.12, 0.12, 0.12, 0.20]
        idx = np.random.choice(range(6), p=probs)
        s, d = deviates[idx]

        # Get potential outcomes and determine observed
        # outcome.
        y_1, y_0 = get_potential_outcomes(s)
        y = d * y_1 + (1 - d) * y_0
```

(continues on next page)

(continued from previous page)

```
        # Collect information
        df.loc[i] = y, d, s, y_1, y_0

    # We want to enforce suitable types for each column.
    # Unfortunately, this cannot be done at the time of
    # initialization.
    df = df.astype(info)

    return df
```

Let us see our simulation in action.

[3]:
```
df = get_sample_matching_demonstration_1(num_agents=1000)
df[["Y", "D", "S"]].head()
```

[3]:
```
           Y  D  S
0   9.254559  0  3
1   7.651437  0  2
2  13.795799  1  3
3   5.265936  1  1
4   3.856628  1  1
```

We are in the comfortable position to not only compute the naive estimate but also the true average treatment effect.

[4]:
```
ate_naive = df.query("D == 1")["Y"].mean() - df.query("D == 0")["Y"].mean()
ate_true = df["Y_1"].sub(df["Y_0"]).mean()

f"The true ATE is {ate_true:4.2f} while its naive estimate is {ate_naive:4.2f}. Why?"
```

[4]:
```
'The true ATE is 2.73 while its naive estimate is 5.96. Why?'
```

What to do?

[5]:
```
df.groupby(["S", "D"])["Y"].mean()
```

[5]:
```
S  D
1  0     2.086518
   1     4.187777
2  0     6.032950
   1     8.058900
3  0     9.902624
   1    14.136920
Name: Y, dtype: float64
```

Note that the observed outcomes within each stratum correspond to the average potential outcome within the stratum. We can compute the average treatment effect by looking at the difference within each strata.

[6]:
```
rslt_outc = df.groupby(["S", "D"])["Y"].mean()
rslt_strat = df["S"].value_counts(normalize=True)

ate_est = 0.0
for s in [1, 2, 3]:
    ate_est += (rslt_outc.loc[s, 1] - rslt_outc.loc[s, 0]) * rslt_strat[s]
```

(continues on next page)

(continued from previous page)

```
f"The stratified estimate for the ATE is {ate_est:4.2f}"
```

[6]:
```
'The stratified estimate for the ATE is 2.80'
```

The ATT and ATC can be computed analogously just by applying the appropriate weights to the strata-specific effect of treatment.

More generally.

$$\{E_N[y_i \mid d_i = 1, s = s_i] - E_N[y_i \mid d_i = 0, s = s_i]\}$$
$$\xrightarrow{p} E[Y^1 - Y^0 \mid S = s] = E[\delta \mid S = s].$$

Weighted sums of these stratified estimates can then be taken such as for the unconditional ATE:

$$\sum_s \{E_N[y_i \mid d_i = 1, s_i = s] - E_N[y_i \mid d_i = 0, s_i = s]\}$$
$$* \Pr_N[s_i = s] \xrightarrow{p} E[\delta]$$

This examples shows all of the basic principles in matching estimators that we will discuss in greater detail in this lecture.

- Treatment and control subjects are matched together in the sense that they are grouped together into strata.
- An average difference between the outcomes of the treatment and control subjects is estimated, based on a weighting of the strata by common distribution.

Overlap conditions

Let's introduce our first complication:

Table 5.3 The Joint Probability Distribution and Conditional Population Expectations for Matching Demonstration 2

	Joint probability distribution of S and D: $D=0$	$D=1$	
$S=1$	$\Pr[S=1, D=0] = .4$	$\Pr[S=1, D=1] = 0$	$\Pr[S=1] = .4$
$S=2$	$\Pr[S=2, D=0] = .1$	$\Pr[S=2, D=1] = .13$	$\Pr[S=2] = .23$
$S=3$	$\Pr[S=3, D=0] = .1$	$\Pr[S=3, D=1] = .27$	$\Pr[S=3] = .37$
	$\Pr[D=0] = .6$	$\Pr[D=1] = .4$	

	Potential outcomes: Under the control state	Under the treatment state	
$S=1$	$E[Y^0 \mid S=1] = 2$		
$S=2$	$E[Y^0 \mid S=2] = 6$	$E[Y^1 \mid S=2] = 8$	$E[Y^1 - Y^0 \mid S=2] = 2$
$S=3$	$E[Y^0 \mid S=3] = 10$	$E[Y^1 \mid S=3] = 14$	$E[Y^1 - Y^0 \mid S=3] = 4$
	$E[Y^0 \mid D=0] = \frac{.4}{.6}(2) + \frac{.1}{.6}(6) + \frac{.1}{.6}(10) = 4$	$E[Y^1 \mid D=1] = \frac{.13}{.4}(8) + \frac{.27}{.4}(14) = 12.05$	

[7]:
```
df = get_sample_matching_demonstration_2(num_agents=1000)
df[["Y", "D", "S"]].head()
```

[7]:
```
           Y  D  S
0   0.495446  0  1
1   7.495097  1  2
2   7.614248  1  2
3   5.407392  0  2
4  12.524174  1  3
```

[8]:
```
df.groupby(["S", "D"])["Y"].mean()
```

[8]:
```
S  D
1  0     2.097111
2  0     6.028036
   1     8.058439
3  0     9.981168
   1    14.023508
Name: Y, dtype: float64
```

Can we at least learn about the treatment on the treated? What else can we do?

Matching as weighting

As indicated by the stylized example, there are often many strata where we do not have treated and control individuals available at the same time.

$\rightarrow$ combine information from different strata with the same propensity score p

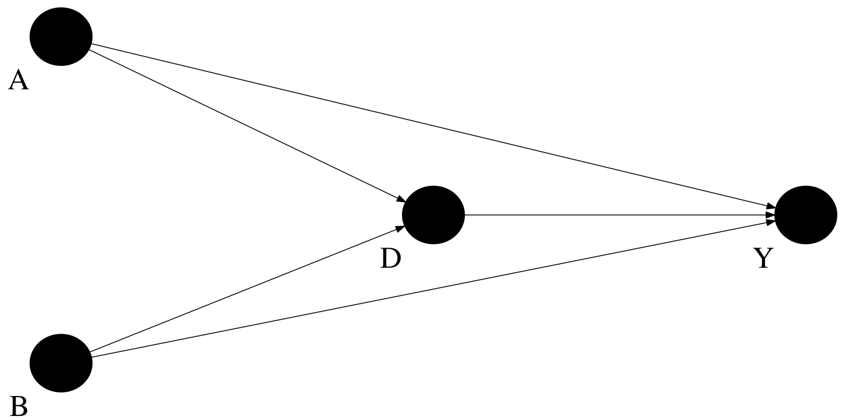
Definition The estimated propensity score is the estimated probability of taking the treatment as a function of variables that predict treatment assignment, i.e. $\Pr[D = 1 \mid S]$.

$\rightarrow$ stratifying on the propensity score itself ameliorates the sparseness problem because the propensity score can be treated as a single stratifying variable (Rosenbaum & Rubin (1983)).

[9]:
```
# We create a grid for two observable characteristics that drive treatment selection and␣
↪o
a_grid = np.linspace(0.01, 1.00, 100)
b_grid = np.linspace(0.01, 1.00, 100)

# We need to study some features of this function to
# to get a sense on the underlying economics.
df, counts = get_sample_matching_demonstration_3(a_grid, b_grid)
df.head()
```

[9]:
```
      a     b  d           y         y_1         y_0         p
0  0.01  0.03  0   94.788634  102.807448   94.788634  0.332700
1  0.01  0.04  1  107.735609  107.735609   93.808018  0.334033
2  0.01  0.05  0  104.010898   97.937608  104.010898  0.335369
3  0.01  0.05  0  107.356485   98.919732  107.356485  0.335369
4  0.01  0.06  1  109.372171  109.372171   95.717052  0.336708
```

underlying causal graphs



We will now look at different ways to construct estimates for the usual causal parameters. So, we first compute their true counterparts.

[10]:
```
stat = df["y_1"].sub(df["y_0"]).mean()
print(f"ATE true:  {stat:5.3f}")

df_treated = df.query("d == 1")
df_control = df.query("d == 0")

stat = df_treated["y"].mean() - df_control["y"].mean()
print(f"ATE naive: {stat:5.3f}")
```

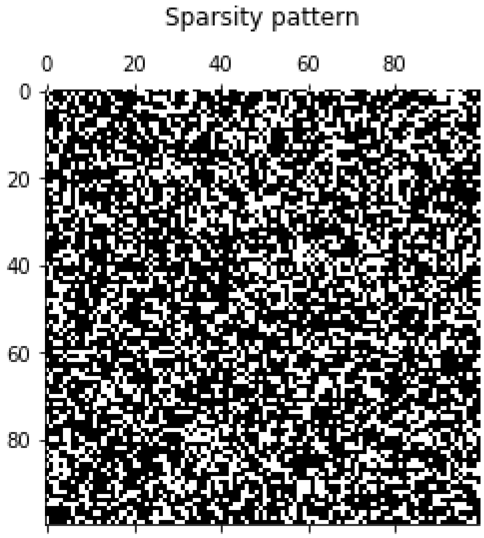
```
ATE true:  4.397
ATE naive: 4.846
```

Let's collect all effects in a dictionary for use further downstream.

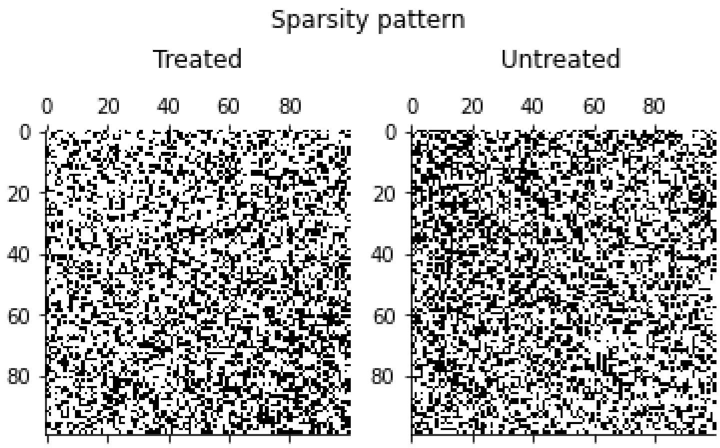
[11]:
```
true_effects = list()
true_effects += [df_treated["y_1"].sub(df_treated["y_0"]).mean()]
true_effects += [df_control["y_1"].sub(df_control["y_0"]).mean()]
true_effects += [(df["y_1"] - df["y_0"]).mean()]
```

How about the issue of sparsity on the data?

[12]:
```
get_sparsity_pattern_overall(counts)
```

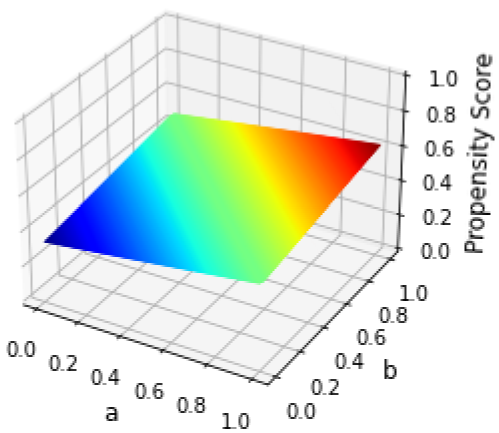


[13]:
```
get_sparsity_pattern_by_treatment(counts)
```



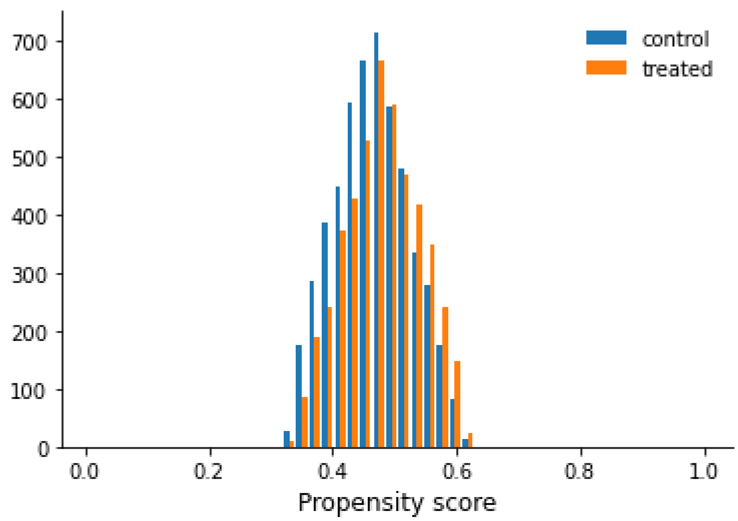
How does the propensity score $P(D = 1 \mid S)$ as a function of the observables (a, b) look like?

[14]:
```
plot_propensity_score(a_grid, b_grid)
```



We still must be worried about common support.

[15]:
```
get_common_support(df)
```



$$\hat{\delta}_{\text{ATT, weight}} \equiv \left(\frac{1}{n^1}\sum_{i:d_i=1} y_i\right) - \left(\frac{\sum_{i:d_i=0}\hat{r}_i y_i}{\sum_{i:d_i=0}\hat{r}_i}\right)$$

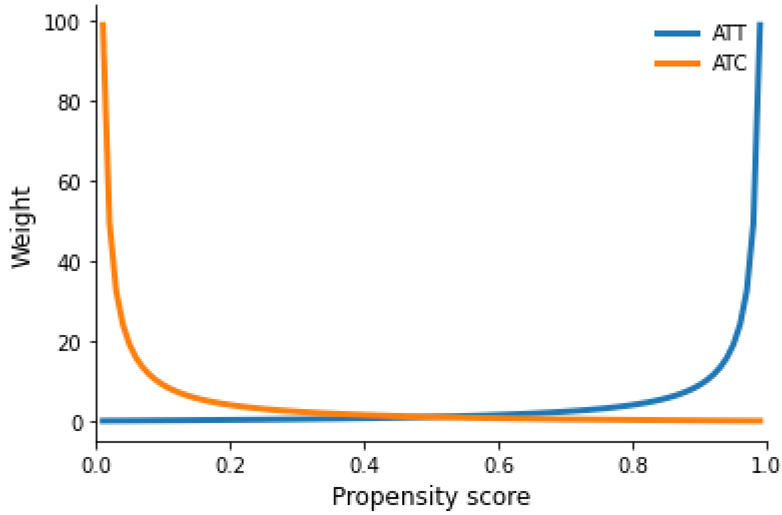
$$\hat{\delta}_{\text{ATC, weight}} \equiv \left(\frac{\sum_{i:d_i=1}\frac{y_i}{\hat{r}_i}}{\sum_{i:d_i=1}\frac{1}{\hat{r}_i}}\right) - \left(\frac{1}{n^0}\sum_{i:d_i=0} y_i\right)$$

$$\hat{\delta}_{\text{ATE, weight}} \equiv \left(\frac{1}{n}\sum_i d_i\right)\hat{\delta}_{\text{ATT, weight}} + \left(1 - \frac{1}{n}\sum_i d_i\right)\hat{\delta}_{\text{ATC, weight}}$$

Weights

$$r_i = \frac{p_i}{1-p_i}$$

[16]:
```
plot_weights()
```



We will now turn to some programming as it introduces you to the actual setup for the propensity score estimation and points towards the issues of potential model misspecification.

[17]:
```
def get_att_weight(df, p):
    """Get weighted ATT.

    Calculates the weighted ATT basd on a provided
    dataset and the propensity score.

    Args:
        df: A dataframe with the observed data.
        p: A numpy array with the weights.

    Returns:
        A float which corresponds to the ATT.
    """
    df_int = df.copy()
    df_int["weights"] = get_odds(p)

    value, weights = df_int.query("d == 0")[["y", "weights"]].values.T
    att = df_int.query("d == 1")["y"].mean() - np.average(value, weights=weights)

    return att


def get_atc_weight(df, p):
    """Get weighted ATC.

    Calculates the weighted ATC basd on a provided
    dataset and the propensity score.

    Args:
        df: A dataframe with the observed data.
        p: A numpy array with the weights.
```

(continues on next page)

(continued from previous page)

```
    Returns:
        A float which corresponds to the ATC.
    """
    df_int = df.copy()
    df_int["weights"] = get_inv_odds(p)

    value, weights = df_int.query("d == 1")[["y", "weights"]].values.T
    atc = np.average(value, weights=weights) - df.query("d == 0")["y"].mean()

    return atc


def get_ate_weight(df, p):
    """Get weighted ATE.

    Calculates the weighted ATE basd on a provided
    dataset and the propensity score.

    Args:
        df: A dataframe with the observed data.
        p: A numpy array with the weights.

    Returns:
        A float which corresponds to the ATE.
    """
    share_treated = df["d"].value_counts(normalize=True)[1]

    atc = get_atc_weight(df, p)
    att = get_att_weight(df, p)

    return share_treated * att + (1.0 - share_treated) * atc


rslt = dict()
for model in ["true", "correct", "misspecified"]:

    print("")
    print(model.capitalize())

    p = get_propensity_score_3(df, model)

    rslt[model] = list()
    rslt[model] += [get_att_weight(df, p)]
    rslt[model] += [get_atc_weight(df, p)]
    rslt[model] += [get_ate_weight(df, p)]

    print("estimated: ATT {:5.3f} ATC {:5.3f} ATE {:5.3f}".format(*rslt[model]))
    print("true:      ATT {:5.3f} ATC {:5.3f} ATE {:5.3f}".format(*true_effects))
```

```
True
```

(continues on next page)

(continued from previous page)

```
estimated: ATT 4.567 ATC 4.356 ATE 4.456
true:      ATT 4.549 ATC 4.259 ATE 4.397

Correct
Optimization terminated successfully.
         Current function value: 0.683753
         Iterations 4
estimated: ATT 4.557 ATC 4.349 ATE 4.448
true:      ATT 4.549 ATC 4.259 ATE 4.397

Misspecified
Optimization terminated successfully.
         Current function value: 0.683792
         Iterations 4
estimated: ATT 4.560 ATC 4.344 ATE 4.447
true:      ATT 4.549 ATC 4.259 ATE 4.397
```

If the treatment assignment can be modeled perfectly, one can solve the sparseness problem that afflict finite datasets.

Requirements

- perfect stratification of the propensity-score-estimating equation
  - capture all back-door paths
  - misspecification of propensity score equation

Matching as data analysis algorithm

$$\hat{\delta}_{\text{ATT, match}} = \frac{1}{n^1}\sum_i \left[(y_i \mid d_i = 1) - \sum_j \omega_{i,j}(y_j \mid d_j = 0)\right]$$

$$\hat{\delta}_{\text{ATC, match}} = \frac{1}{n^0}\sum_j \left[\sum_i \omega_{j,i}(y_i \mid d_i = 1) - (y_j \mid d_j = 0)\right]$$

Alternative matching estimators can be represented as different procedures for deriving the weights $\omega_{i,j}$ and $\omega_{j,i}$ in the two expressions above.

Design choices

- How many matched cases designated for each to-be-matched target?
- How to weigh multiple matched cases if more than one is utilized for each target case?

Basic variants

- exact matching
  - construct counterfactual based on individuals with identical S
- nearest-neighbor and caliper
  - construct counterfactual based on individuals closest on a unidimensional measure (e.g. propensity score), caliper ensures reasonable maximum distance to neighbor
- interval matching
  - construct counterfactual by sorting individuals into segments based on unidimensional metric
- kernel matching
  - constructs counterfactual based on **all** individuals but weights them based on the distance

Benchmarking tutorial

We revisit a simulated version of the data used in Morgan (2001). He contributes to the debate over the size of the causal of effect of Catholic schooling on test scores. The dataset is provided by the textbook and also available in our online repository.

Issues

- What is the relative performance of alternative matching estimators?
- What is the consequence of conditioning only on a subset of the variables in the set of perfect stratification variables S.

[18]:
```
df = get_sample_matching_demonstration_4()
df.describe()
```

[18]:
```
                   y          treat          asian       hispanic          black  \
count   10000.000000   10000.000000   10000.000000   10000.000000   10000.000000
mean      100.459241       0.105200       0.089800       0.143300       0.096100
std        13.493304       0.306826       0.285909       0.350396       0.294743
min        50.065298       0.000000       0.000000       0.000000       0.000000
25%        91.519660       0.000000       0.000000       0.000000       0.000000
50%       100.303288       0.000000       0.000000       0.000000       0.000000
75%       109.459337       0.000000       0.000000       0.000000       0.000000
max       159.426572       1.000000       1.000000       1.000000       1.000000

             natamer       urban          neast       ncentral          south  \
count   10000.000000  10000.00000  10000.000000   10000.000000   10000.000000
mean        0.007500      0.38910      0.208100       0.264000       0.302200
std         0.086281      0.48757      0.405969       0.440821       0.459234
min         0.000000      0.00000      0.000000       0.000000       0.000000
25%         0.000000      0.00000      0.000000       0.000000       0.000000
50%         0.000000      0.00000      0.000000       0.000000       0.000000
75%         0.000000      1.00000      0.000000       1.000000       1.000000
max         1.000000      1.00000      1.000000       1.000000       1.000000

        ...  ncentralblack    southblack       twohisp     neasthisp  \
count   ...   10000.000000  10000.000000  10000.000000  10000.000000
```

(continues on next page)

(continued from previous page)

```
mean   ...        0.018200      0.050500      0.102100      0.016000
std    ...        0.133681      0.218985      0.302795      0.125481
min    ...        0.000000      0.000000      0.000000      0.000000
25%    ...        0.000000      0.000000      0.000000      0.000000
50%    ...        0.000000      0.000000      0.000000      0.000000
75%    ...        0.000000      0.000000      0.000000      0.000000
max    ...        1.000000      1.000000      1.000000      1.000000

       ncentralhisp   southhisp            yt            yc        dshock  \
count  10000.000000  10000.00000  10000.000000  10000.000000  1.000000e+04
mean       0.014900      0.05590    105.729319     99.727395  7.105427e-18
std        0.121159      0.22974     13.525777     13.202781  2.088954e+00
min        0.000000      0.00000     54.788152     50.065298 -7.765338e+00
25%        0.000000      0.00000     96.848440     91.025084 -1.386373e+00
50%        0.000000      0.00000    105.712292     99.685598  1.763749e-02
75%        0.000000      0.00000    114.828609    108.607459  1.391807e+00
max        1.000000      1.00000    159.790235    151.442150  8.432203e+00

                  d
count  10000.000000
mean       6.001924
std        2.146356
min       -2.142958
25%        4.587783
50%        6.010071
75%        7.438161
max       14.653340

[8 rows x 30 columns]
```

Let us look at some example covariates.

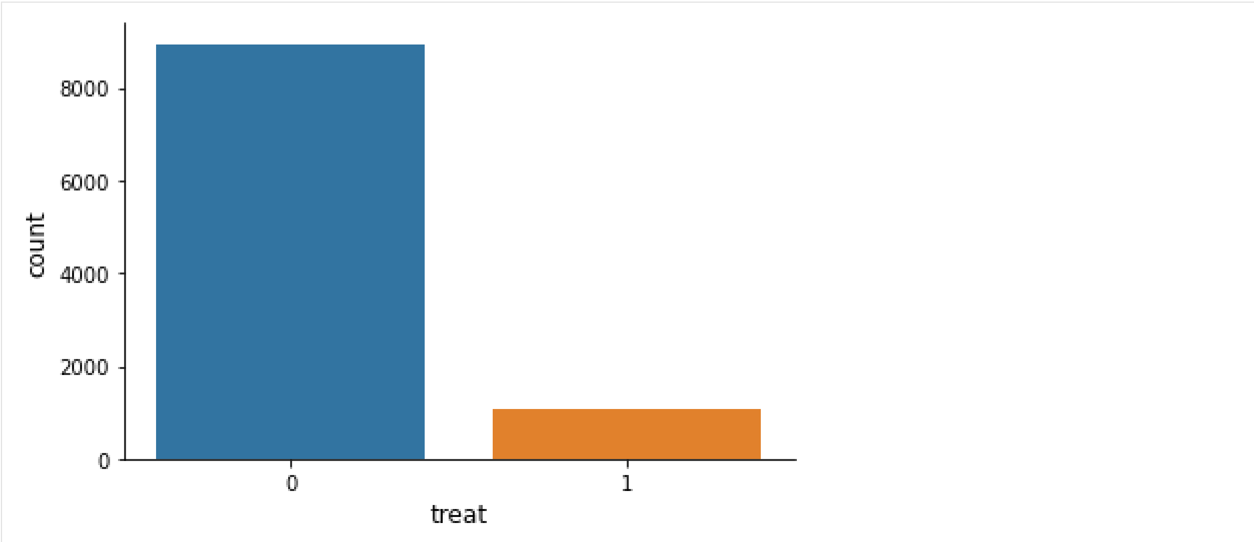
[19]:
```
example_covariates = ["black", "urban", "test"]
df[example_covariates].describe()
```

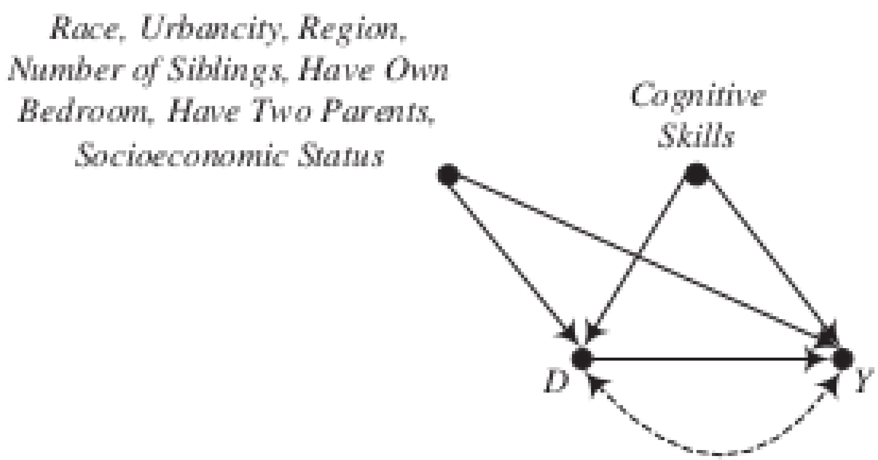
[19]:

	black	urban	test
count	10000.000000	10000.00000	10000.000000
mean	0.096100	0.38910	-0.002229
std	0.294743	0.48757	0.991747
min	0.000000	0.00000	-3.862709
25%	0.000000	0.00000	-0.676463
50%	0.000000	0.00000	-0.006683
75%	0.000000	1.00000	0.660488
max	1.000000	1.00000	3.722783

[20]:
```
sns.countplot(x="treat", data=df)
```

[20]:
```
<AxesSubplot:xlabel='treat', ylabel='count'>
```





Is there any hope in identifying the ATE?

$$
\begin{aligned}
y_i^0 = 100 &+ 2(Asian_i) - 3(Hispanic_i) - 4(Black_i) \\
&- 5(Native\,American_i) - 1(Urban_i) + .5(Northeast_i) \\
&+ .5(North\,Central_i) - .5(South_i) + .02(Number\,of\,Sibling \\
&+ .05(Own\,Bedroom_i) + 1(Two\,Parent\,Household_i) \\
&+ 2(Socioeconomic\,Status_i) + 4(Cognitive\,Skills_i) + \upsilon_i^0,
\end{aligned}
$$

$$
\begin{aligned}
S_i\phi = & -4.6 - .69(Asian_i) + .23(Hispanic_i) - .76(Black_i) \\
& - .46(Native\,American_i) + 2.7(Urban_i) + 1.5(Northeast_i) \\
& + 1.3(North\,Central_i) + .35(South_i) - .02(Number\,of\,Siblings_i) \\
& - .018(Own\,Bedroom_i) + .31(Two\,Parent\,Household_i) \\
& + .39(Socioeconomic\,Status_i) + .33(Cognitive\,Skills_i) \\
& - .032(Socioeconomic\,Status_i^2) - .32(Cognitive\,Skills_i^2) \\
& - .084(Socioeconomic\,Status_i \times Cognitive\,Skills_i) \\
& - .37(Two\,Parent\,Household_i \times Black_i) \\
& + 1.6(Northeast_i \times Black_i) - .38(North\,Central_i \times Black_i) \\
& + .72(South_i \times Black_i) + .23(Two\,Parent\,Household_i \times Hispanic) \\
& - .74(Northeast_i \times Hispanic_i) - 1.3(North\,Central_i \times Hispanic_i) \\
& - 1.3(South_i \times Hispanic_i) + .25\delta_i''.
\end{aligned}
\quad (5.
$$

There exists systematic treatment effect heterogeneity:

$$
\begin{aligned}
0 + 1(& Hispanic_i \times Northeast_i) + .5(Hispanic_i \times North\,Central_i) \\
& + 1.5(Black_i \times Northeast_i) + .75(Black_i \times North\,Central_i) \\
& + .5(Cognitive\,Skills_i)
\end{aligned}
$$

Here comes the key feature that generates the dependence between D and Y based on an unobservable.

$$y_i^1 = y_i^0 + \delta_i' + \delta_i''$$

$\rightarrow \delta_i''$ is a associated with one of the potential outcomes and also affects the probability to select treatment. Individuals that have the most to gain from treatment are more likely to select into treatment.

However, we can still identify the ATT. Why?

$$
\begin{aligned}
E[\delta \mid D = 1, S] &= E[Y^1 - Y^0 \mid D = 1, S] \\
&= E[Y^1 \mid D = 1, S] - E[Y^0 \mid D = 1, S] \\
&= E[Y^1 \mid D = 1, S] - E[Y^0 \mid D = 0, S] \\
&= E[Y \mid D = 1, S] - E[Y \mid D = 0, S]
\end{aligned}
$$

We establish a clear benchmark by looking at the true treatment effect.

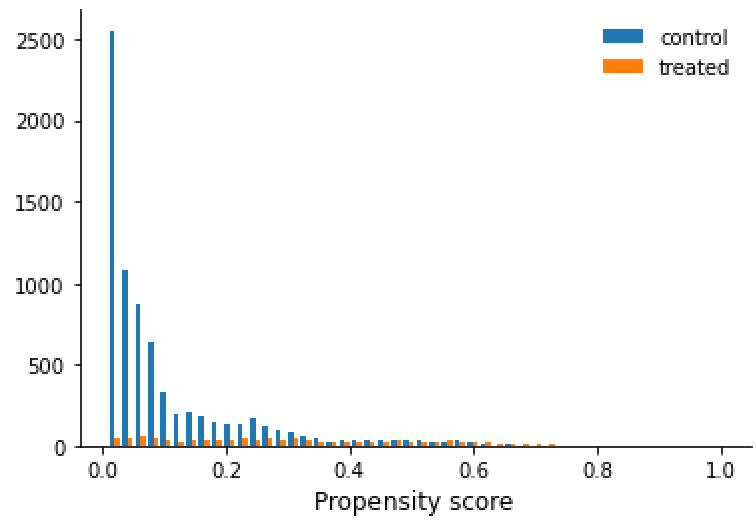
[21]:
```
stat = (df["yt"] - df["yc"])[df["treat"] == 1].mean()
print(f"The true ATT is {stat:5.3f}")
```

```
The true ATT is 6.957
```

How are doing with respect to common support for the propensity score?

[22]:
```
df = get_sample_matching_demonstration_4()
df["p"] = get_propensity_scores_matching_demonstration_4(df)
get_common_support(df, "treat")
```

```
Optimization terminated successfully.
         Current function value: 0.252643
         Iterations 8
```



Now we implement our own nearest neighbor matching routine.

[23]:
```
def nearest_neighbor_algorithm_for_att(df):

    # We select all treated individuals
    df_control = df.query("treat == 0").reset_index()
    df_treated = df.query("treat == 1").reset_index()

    # We create a new dataframe wth the nearest neighbor.
    df_neighbour = pd.DataFrame(columns=df.columns)

    # We want to store the information about the nearest neighbor.
    idx_list = list()

    for i, (index, row) in enumerate(df_treated.iterrows()):

        df_control["distance"] = np.abs(df_control["p"] - row["p"])
        idx_ngbr = df_control["distance"].idxmin()

        df_neighbour.loc[i, :] = df_control.loc[idx_ngbr, :]

        # We want to record the index of the neighbor.
        idx_list.append(idx_ngbr)

    df_neighbour = df_neighbour.add_suffix("_ngbr")
    df_matched = pd.concat([df_treated, df_neighbour], axis=1)

    return df_matched, pd.Series(idx_list)
```

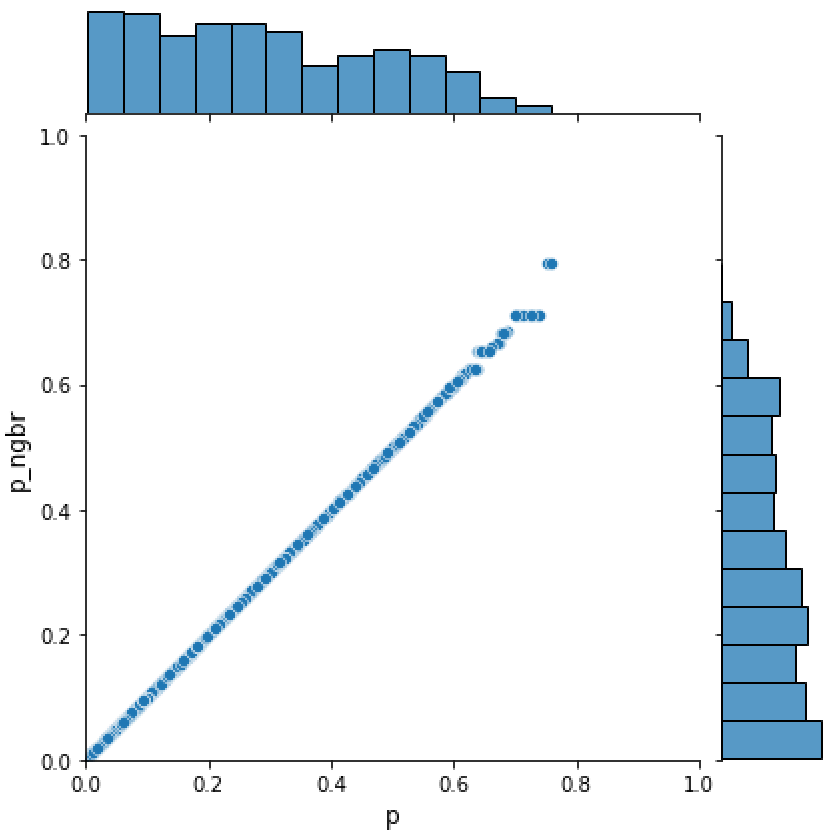
Let’s put our algorithm to work!

[24]:
```
df_matched, idx_series = nearest_neighbor_algorithm_for_att(df)
```

How well are we able to match individuals based on their propensity score? How does earnings compare between matched individuals?

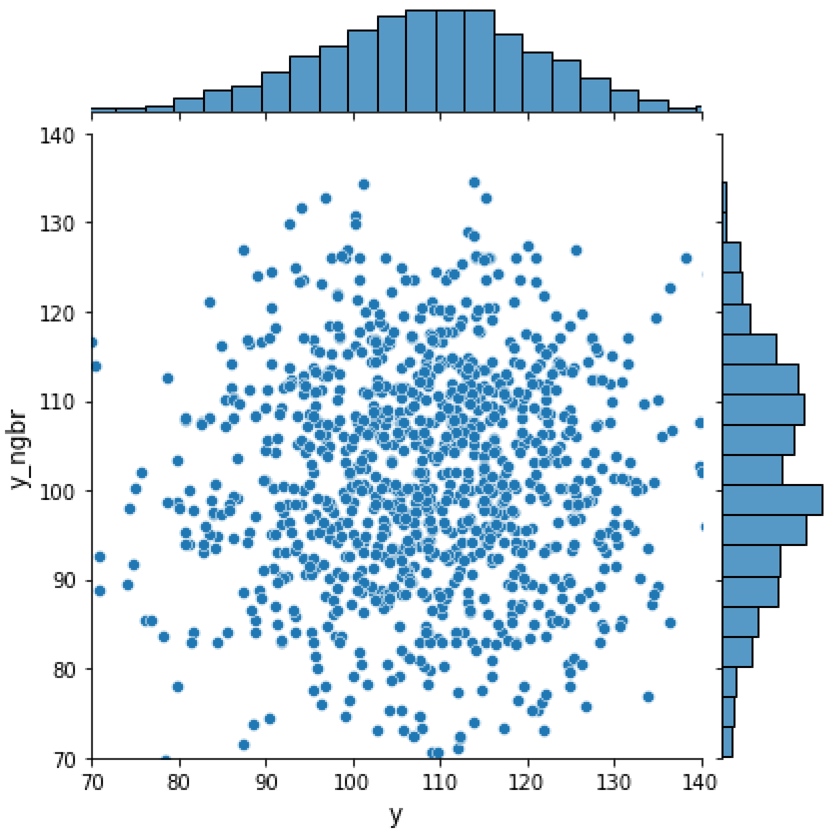
[25]:
```
sns.jointplot(x="p", y="p_ngbr", data=df_matched, xlim=[0, 1], ylim=[0, 1])
```

[25]:
```
<seaborn.axisgrid.JointGrid at 0x7f1d276227d0>
```



[26]:
```
sns.jointplot(x="y", y="y_ngbr", data=df_matched, xlim=[70, 140], ylim=[70, 140])
```

[26]:
```
<seaborn.axisgrid.JointGrid at 0x7f1d276221d0>
```



After all this effort, what is our treatment effect estimate?

[27]:
```
stat = (df_matched["y"] - df_matched["y_ngbr"]).mean()
print(f"Here is our estimate for the treatment effect: {stat:5.3f}")
```

```
Here is our estimate for the treatment effect: 7.236
```

How often do we match the same individual?

[28]:
```
idx_series.value_counts()
```

[28]:
```
1236    13
1701    11
1129    11
1715    10
3424     6
        ..
948      1
2998     1
2999     1
3830     1
2047     1
Length: 790, dtype: int64
```

How do our covariantes balance across treatment status?

[29]:
```
df.groupby("treat")[example_covariates].mean().T
```

[29]:
```
treat         0         1
black  0.092646  0.125475
urban  0.337953  0.824144
test  -0.039018  0.310690
```

We now want to revisit the balancing of covariates.

[30]:
```
for col in example_covariates:
    print("\n", col)
    print(f"treated: {df_matched[col].mean():5.3f}")
    print(f"matched: {df_matched[col + '_ngbr'].mean():5.3f}")
```

```

 black
treated: 0.125
matched: 0.142

 urban
treated: 0.824
matched: 0.827

 test
treated: 0.311
matched: 0.309
```

Let's take a little detour and look at the balancing of observables in the Lalonde dataset.

[31]:
```
df = get_lalonde_data()
df.head()
```

[31]:
```
          data_id  treat  age  education  black  hispanic  married  nodegree  \
0  Lalonde Sample      1   37         11      1         0        1         1
1  Lalonde Sample      1   22          9      0         1        0         1
2  Lalonde Sample      1   30         12      1         0        0         0
3  Lalonde Sample      1   27         11      1         0        0         1
4  Lalonde Sample      1   33          8      1         0        0         1

   re75        re78           Y  Y_0         Y_1  D
0   0.0   9930.0460   9930.0460  NaN   9930.0460  1
1   0.0   3595.8940   3595.8940  NaN   3595.8940  1
2   0.0  24909.4500  24909.4500  NaN  24909.4500  1
3   0.0   7506.1460   7506.1460  NaN   7506.1460  1
4   0.0    289.7899    289.7899  NaN    289.7899  1
```

[32]:
```
example_covariates = ["black", "married", "hispanic", "re75"]
df.groupby("treat")[example_covariates].mean().T
```

[32]:
```
treat               0            1
black        0.800000     0.801347
married      0.157647     0.168350
hispanic     0.112941     0.094276
re75      3026.682743  3066.098187
```

The covariates are balanced before any reweighting thanks to the assignment mechanisms.

Returning to our simulated example. Which matching algorihtm is the best?

Incomplete specification

- missing higher-order interactions in propensity score and omission of cognitive variable

Table 5.6 Matching Estimates of the ATT, Catholic Schooling on Achievement for One Simulated Dataset

Method	Specification of treatment assignment variables: Incomplete	Complete	Number of treatment cases retained for the estimate of the ATT
Nearest-neighbor match:			
1 without replacement (ps2)	7.37	7.03	1052
1 without replacement (MI)	7.55	7.09	1052
1 with replacement (ps2/psc)	7.77	7.17	1052
1 with replacement (MI)	7.96	7.38	1052
1 with replacement and caliper = .05 SD (ps2)	7.77	7.15	1051
1 with replacement and caliper = .05 SD (MI)	7.27	6.43	1051
5 with replacement (ps2)	7.51	6.42	1052
5 with replacement (MI)	8.06	7.29	1052
5 with replacement and caliper = .05 SD (ps2)	7.55	6.37	1051
5 with replacement and caliper = .05 SD (MI)	8.00	7.12	1051
Radius match:			
Caliper = .05 SD (ps2)	7.61	6.36	1051
Caliper = .05 SD (psc)	8.23	7.84	1051
Interval match:			
10 fixed blocks (MI)	8.71	8.71	1052
Variable blocks (ps2)	7.50	6.60	1052
Kernel match:			
Epanechnikov (ps2/psc)	7.57	6.58	1052
Gaussian (ps2/psc)	7.70	6.82	1052
Optimal match (MI-opt)	6.84	6.78	1052
Genetic match (MI-gen)	7.80	6.46	1052
Coarsened exact match (cem)	7.54	6.59	1015/973

Notes: The software utilized is denoted "ps2" for Leuven and Sianesi (2012), "MI" for Ho et al. (2011), "psc" for Becker and Ichino (2005), "opt" for Hansen, Fredrickson, and Buckner (2013), "gen" for Sekhon (2013), and "cem" for Iacus et al. (2012b).

Notes

- The estimates for the incomplete specification are usually much larger.
- Software programs that used the same routine yield very different estimates.

Resources

- **Buso, M., DiNardo, J., & McCrary, J. (2014)**. New Evidence on the finite sample properties of propensity score reweighting and matching estimators, *The Review of Economics and Statistics*, 96(5), 885-897.
- **Heckman, J. J., Ichimura, H., Smith, J. A. and Todd, P. (1998)**. Characterizing selection bias using experimental data. *Econometrica*, 66(5), 1017–98.
- **Heckman, J. J., Ichimura, H. and Todd, P. (1998)**. Matching as an econometric evaluation estimator. *Review of Economic Studies*, 65(2), 261–94.
- **Heckman, J. J. and Hotz, V. J. (1989)**. Choosing among alternative non-experimental methods for estimating the impact of social programs: The case of manpower training. *Journal of the American Statistical Association*, 84(408), 862–74.
- **Heckman, J. and Navarro-Lozano, S. (2004)**. Using matching, instrumental variables, and control functions to estimate economic choice models. *The Review of Economics and Statistics*, 86(1), 30–57.
- **Morgan, S. (2001)**. Counterfactuals, causal effect heterogeneity, and the catholic school effect on learning. *Sociology of Education*, 74(4), 341–374.
- **Rosenbaum, P. R. and Rubin, D. (1983)**. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1), 41–55.
- **Rosenbaum, P. (2020)**. Modern algorithms for matching in observational studies, 7, 143-176.
- **Rubin, D. (2006)**. The design versus the analysis of observational studies for causal effects: parallels with the design of randomized trials, *Statistics in Medicine*, 26(1), 20-36.
- **Smith, J. A. and Todd, P. (2005)**. Does matching evercome Lalonde's eritique of nonexperimental estimators?. *Journal of Econometrics*, 125(1–2), 305–53.
- **Stuart, E. (2010)**. Matching methods for causal inference: a review and a look forward, *Statistical Science*, 25(1), 1-21.

1.7 Regression estimators

We study the most common form of data analysis by looking at simple regression estimators. We first study them as a basic descriptive tool that provides the best linear approximation to the conditional expectation function. Then we turn to the more demanding interpretation that it allows to determine causal effects. We contrast the issues of omitted-variable bias and selection bias. Finally, we conclude with an illustration of Freedman's paradox to showcase some of the challenges in applied empirical work.

1.7.1 Regression estimators of causal effects

Overview

- Regression as a descriptive tool
- Regression adjustment as a strategy to to estimate causal effects
- Regression as conditional-variance-weighted matching
- Regression as an implementation of a perfect stratification
- Regression as supplemental adjustment when matching
- Extensions and other perspectives
- Conclusion

We start with different ways of using regression

- descriptive tools
  - Anscombe quartet
- estimating causal effects
- Freedman's paradox

Regression as a descriptive tool

Goldberger (1991) motivates least squares regression as a technique to estimate a **best-fitting** linear approximation to a conditional expectation function that may be nonlinear in the population.

Best is defined as minimizing the average squared differences between the fitted values and the true values of the conditional expectations functions.

Table 6.1 The Joint Probability Distribution and Conditional Population Expectations for Regression Demonstration 1

	Joint probability distribution of S and D	
	Control group: $D=0$	Treatment group: $D=1$
$S=1$	$\Pr[S=1, D=0] = .36$	$\Pr[S=1, D=1] = .08$
$S=2$	$\Pr[S=2, D=0] = .12$	$\Pr[S=2, D=1] = .12$
$S=3$	$\Pr[S=3, D=0] = .12$	$\Pr[S=3, D=1] = .2$
	Potential outcomes under the control state	
$S=1$	$E[Y^0 \mid S=1, D=0] = 2$	$E[Y^0 \mid S=1, D=1] = 2$
$S=2$	$E[Y^0 \mid S=2, D=0] = 6$	$E[Y^0 \mid S=2, D=1] = 6$
$S=3$	$E[Y^0 \mid S=3, D=0] = 10$	$E[Y^0 \mid S=3, D=1] = 10$
	Potential outcomes under the treatment state	
$S=1$	$E[Y^1 \mid S=1, D=0] = 4$	$E[Y^1 \mid S=1, D=1] = 4$
$S=2$	$E[Y^1 \mid S=2, D=0] = 8$	$E[Y^1 \mid S=2, D=1] = 8$
$S=3$	$E[Y^1 \mid S=3, D=0] = 14$	$E[Y^1 \mid S=3, D=1] = 14$
	Observed outcomes	
$S=1$	$E[Y \mid S=1, D=0] = 2$	$E[Y \mid S=1, D=1] = 4$
$S=2$	$E[Y \mid S=2, D=0] = 6$	$E[Y \mid S=2, D=1] = 8$
$S=3$	$E[Y \mid S=3, D=0] = 10$	$E[Y \mid S=3, D=1] = 14$

```
[18]: df = get_sample_demonstration_1(num_agents=10000)
      df.head()
```

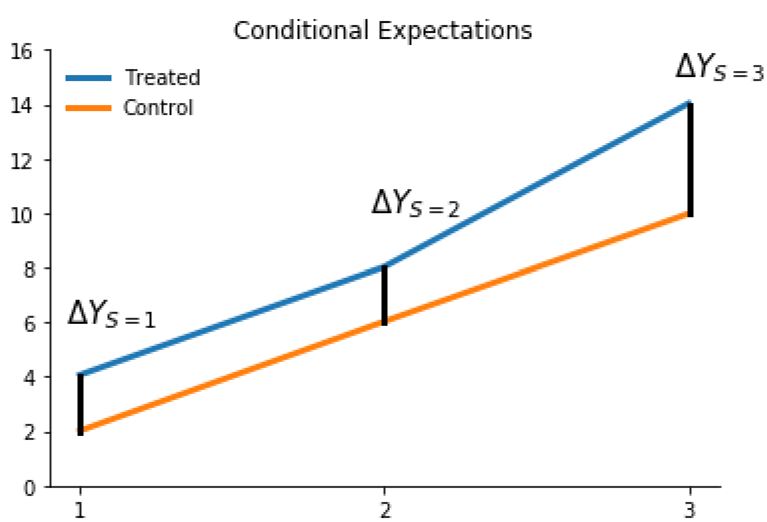
```
[18]:          Y  D  S          Y_1       Y_0
      0  0.113157  0  1   4.055376  0.113157
      1  9.479227  0  3  14.146062  9.479227
      2  0.409400  0  1   2.081023  0.409400
      3  7.087262  1  2   7.087262  5.145585
      4  3.338352  0  1   2.825938  3.338352
```

```
[19]: df.groupby(["D", "S"])["Y"].mean()
```

```
[19]: D  S
      0  1     2.014537
         2     6.032069
         3     9.976885
      1  1     4.067050
         2     8.028103
         3    14.025534
      Name: Y, dtype: float64
```

How does the functional form of the conditional expectation look like?

```
[20]: plot_conditional_expectation_demonstration_1(df)
```



What does the difference between the two lines tell us about treatment effect heterogeneity?

We will fit four different prediction models using ordinary least squares.

$$\hat{Y} = \beta_0 + \beta_1 D + \beta_2 S$$
$$\hat{Y} = \beta_0 + \beta_1 D + \beta_2 S_1 + \beta_3 S_2$$
$$\hat{Y} = \beta_0 + \beta_1 D + \beta_2 S_1 + \beta_3 S_2 + \beta_4 S_1 * D + \beta_5 S_2 * D$$

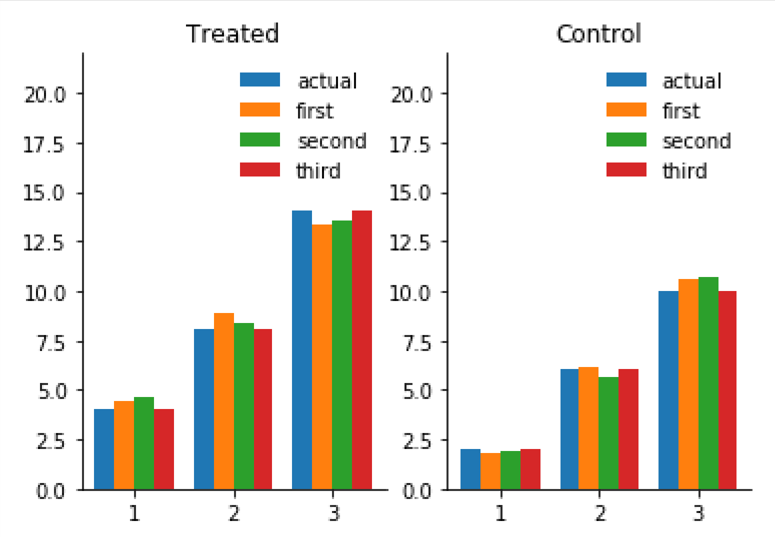
```
[21]: rslt = smf.ols(formula="Y ~ D + S", data=df).fit()
      rslt.summary()
```

[21]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                      Y   R-squared:                       0.941
Model:                            OLS   Adj. R-squared:                  0.941
Method:                 Least Squares   F-statistic:                 8.018e+04
Date:                Tue, 26 May 2020   Prob (F-statistic):               0.00
Time:                        07:30:39   Log-Likelihood:                -15339.
No. Observations:               10000   AIC:                         3.068e+04
Df Residuals:                    9997   BIC:                         3.071e+04
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -2.6594      0.027    -98.877      0.000      -2.712      -2.607
D              2.7202      0.025    108.756      0.000       2.671       2.769
S              4.4181      0.014    311.459      0.000       4.390       4.446
==============================================================================
Omnibus:                        1.627   Durbin-Watson:                   2.023
Prob(Omnibus):                  0.443   Jarque-Bera (JB):                1.637
Skew:                           0.016   Prob(JB):                        0.441
Kurtosis:                       2.946   Cond. No.                         6.15
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
↪specified.
"""
```

[22]:
```python
df["predict"] = rslt.predict()
df.groupby(["D", "S"])[["Y", "predict"]].mean()
```

[22]:
```
             Y    predict
D S
0 1   2.014537   1.758657
  2   6.032069   6.176733
  3   9.976885  10.594808
1 1   4.067050   4.478865
  2   8.028103   8.896941
  3  14.025534  13.315017
```

[23]:
```python
plot_predictions_demonstration_1(df)
```



Anscombe quartet

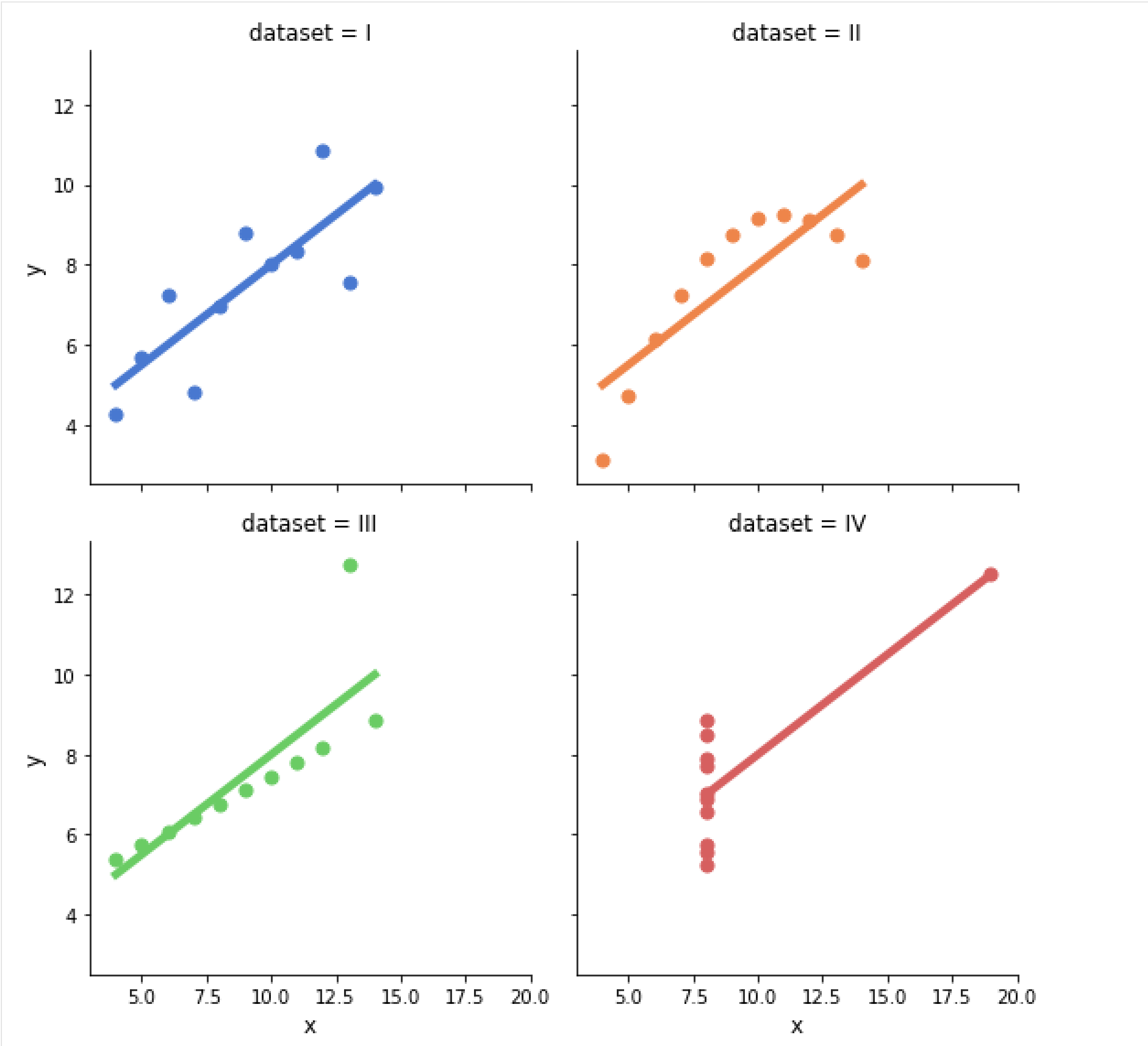
The best linear approximation can be the same for very different functions. The **Anscombe quartet** (Anscombe, 1973) and many other useful datasets are available in `statsmodels` as part of the Datasets Package.

[30]:
```
df1, df2, df3, df4 = get_anscombe_datasets()
for i, df in enumerate([df1, df2, df3, df4]):
    rslt = smf.ols(formula="y ~ x", data=df).fit()
    print(f"\n Dataset {i}")
    print(" Intercept: {:5.3f} x: {:5.3f}".format(*rslt.params))
```

```
Dataset 0
Intercept: 3.000 x: 0.500

Dataset 1
Intercept: 3.001 x: 0.500

Dataset 2
Intercept: 3.002 x: 0.500

Dataset 3
Intercept: 3.002 x: 0.500
```

So what does the data behind these regressions look like?

[31]:
```
plot_anscombe_dataset()
```



Regression adjustment as a strategy to estimate causal effects

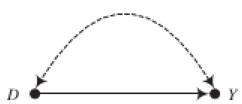
Regression models and omitted-variable bias

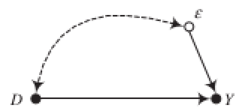
$$Y = \alpha + \delta D + \epsilon$$

- δ is interpreted as an invariant, structural causal effect that applies to all members of the population.
- ϵ is a summary random variable that represents all other causes of Y.

$$\hat{\delta}_{OLS,\text{bivariate}} = \frac{Cov_N(y_i, d_i)}{Var_N(d_i)}$$

It now depends on the correlation between ϵ and D whether $\hat{\delta}$ provides an unbiased and consistent estimate of the true causal effect





We now move to the potential outcomes model to clarify the connection between **omitted-variable bias** and **self-selection bias**.

Potential outcomes and omitted-variable bias

$$Y = \underbrace{\mu^0}_{\alpha} + \underbrace{(\mu^1 - \mu^0)}_{\delta} D + \underbrace{\{\nu^0 + D(\nu^1 - \nu^0)\}}_{\epsilon},$$

where $\mu^0 \equiv E[Y^0]$, $\mu^1 \equiv E[Y^1]$, $\nu^0 \equiv Y^0 - E[Y^0]$, and $\nu^1 \equiv Y^1 - E[Y^1]$.

What induces a correlation between D an $\{\nu^0 + D(\nu^1 - \nu^0)\}$?

- **baseline bias**, there is a net baseline difference in the hypothetical no-treatment state that is correlated with treatment uptake $\rightarrow$ D is correlated with ν_0
- **differential treatment bias**, there is a net treatment effect difference that is correlated with treatment uptake $\rightarrow$ D is correlated with $D(\nu^1 - \nu^0)$

Table 6.2 Examples of the Two Basic Forms of Bias for Least Squares Regression

	y_i^1	y_i^0	υ_i^1	υ_i^0	y_i	d_i	$\upsilon_i^0+d_i(\upsilon_i^1-\upsilon_i^0)$
Differential baseline bias only							
In treatment group	20	10	0	5	20	1	0
In control group	20	0	0	−5	0	0	−5
Differential treatment effect bias only							
In treatment group	20	10	2.5	0	20	1	2.5
In control group	15	10	−2.5	0	10	0	0
Both types of bias							
In treatment group	25	5	5	−2.5	25	1	5
In control group	15	10	−5	2.5	10	0	2.5

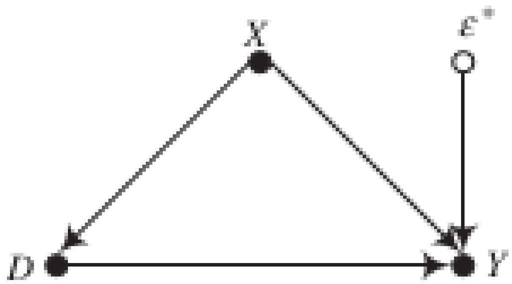
Errata

Please note that there is a relevant correction on the author's website:

- page 198, Table 6.2, first panel: In order to restrict the bias to differential baseline bias only, as required by the label on the first panel of the table, replace 20 with 10 in the first cell of the second row.Then, carry the changes across columns so that (a) the values for ν_i^1 are 5 for the individual in the treatment group and -5 for the individual in the control group and (b) the value for $\nu_i^0 + D(\nu_i^1 \nu_i^0)$ is 5 for the individual in the treatment group

Group	y_i^1	y_i^0	ν_i^1	ν_i^0	y_i	d_i	$\nu_i^1 + d_i(\nu_i^1 - \nu_i^0)$	$ d_i (ν_i^1 − ν_i^0)$
Treated	20	10	5	5	20	1	5.000000000000000000000000	0.000000000
Control	10	0	-5	-5	0	0	-5	0.000000000

Regression as adjustment for otherwise omitted variables



We first want to illustrate how we can *subtract out* the dependence between D and Y induced by their common determinant X. Let's quickly simulate a parameterized example:

$$D = I[X + \eta > 0]$$
$$Y = D + X + \epsilon,$$

where (η, ϵ) follow a standard normal distribution.

[118]:
```
df = get_quick_sample(num_samples=1000)
```

We now first run a complete regression.

[122]:
```
stat = smf.ols(formula="Y ~ D + X", data=df).fit().params[1]
print(f"Estimated effect: {stat:5.3f}")
```

```
Estimated effect: 0.924
```

However, as it turns out, we can also get the identical estimate by first partialling out the effect of X on D as well as Y.

[127]:
```
df_resid = pd.DataFrame(columns=["Y_resid", "D_resid"])
for label in ["Y", "D"]:
    column, formula = f"{label}_resid", f"{label} ~ X"
    df_resid.loc[:, column] = smf.ols(formula=formula, data=df).fit().resid

smf.ols(formula="Y_resid ~ D_resid", data=df_resid).fit().params[1]
print(f"Estimated effect: {stat:5.3f}")
```

```
Estimated effect: 0.924
```

We will now look at two datasets that are observationally equivalent but regression adjustment for observable X does only work in one of them.

Table 6.4 Two Six-Person Examples in Which Regression Adjustment Is Differentially Effective

	Regression adjustment with X generates an unbiased estimate for D							
	y_i^1	y_i^0	υ_i^1	υ_i^0	y_i	d_i	x_i	$\upsilon_i^0+d_i(\upsilon_i^1-\upsilon_i^0)$
In treatment group	20	10	2.5	2.5	20	1	1	2.5
In treatment group	20	10	2.5	2.5	20	1	1	2.5
In treatment group	15	5	-2.5	-2.5	15	1	0	-2.5
In control group	20	10	2.5	2.5	10	0	1	2.5
In control group	15	5	-2.5	-2.5	5	0	0	-2.5
In control group	15	5	-2.5	-2.5	5	0	0	-2.5
	Regression adjustment with X does not generate an unbiased estimate for D							
	y_i^1	y_i^0	υ_i^1	υ_i^0	y_i	d_i	x_i	$\upsilon_i^0+d_i(\upsilon_i^1-\upsilon_i^0)$
In treatment group	20	10	2.83	2.5	20	1	1	2.83
In treatment group	20	10	2.83	2.5	20	1	1	2.83
In treatment group	15	5	-2.17	-2.5	15	1	0	-2.17
In control group	18	10	.83	2.5	10	0	1	2.5
In control group	15	5	-2.17	-2.5	5	0	0	-2.5
In control group	15	5	-2.17	-2.5	5	0	0	-2.5

Note

- The naive estimates will be identical as the observed values y_i and d_i are the same.

[40]:
```
for sample in range(2):

    df = get_sample_regression_adjustment(sample)
    print("Sample {:}\n".format(sample))

    stat = (df["Y_1"] - df["Y_0"]).mean()
    print("True effect:    {:5.4f}".format(stat))

    stat = df.query("D == 1")["Y"].mean() - df.query("D == 0")["Y"].mean()
    print("Naive estimate: {:5.4f}".format(stat))

    rslt = smf.ols(formula="Y ~ D", data=df).fit()
    print(rslt.summary())
```

```
Sample 0

True effect:    10.0000
Naive estimate: 11.6540
                            OLS Regression Results
==============================================================================
Dep. Variable:                      Y   R-squared:                       0.860
Model:                            OLS   Adj. R-squared:                  0.860
```

(continues on next page)

(continued from previous page)

```
Method:                 Least Squares   F-statistic:                     6113.
Date:                Tue, 26 May 2020   Prob (F-statistic):               0.00
Time:                        11:44:40   Log-Likelihood:                -2275.1
No. Observations:                1000   AIC:                              4554.
Df Residuals:                     998   BIC:                              4564.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      6.8379      0.105     65.272      0.000       6.632       7.044
D             11.6540      0.149     78.187      0.000      11.361      11.946
==============================================================================
Omnibus:                     7278.240   Durbin-Watson:                   1.966
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               94.873
Skew:                          -0.097   Prob(JB):                     2.50e-21
Kurtosis:                       1.504   Cond. No.                         2.60
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly␣
↪specified.
Sample 1

True effect:    9.6280
Naive estimate: 11.6540
                            OLS Regression Results
==============================================================================
Dep. Variable:                      Y   R-squared:                       0.860
Model:                            OLS   Adj. R-squared:                  0.860
Method:                 Least Squares   F-statistic:                     6113.
Date:                Tue, 26 May 2020   Prob (F-statistic):               0.00
Time:                        11:44:42   Log-Likelihood:                -2275.1
No. Observations:                1000   AIC:                              4554.
Df Residuals:                     998   BIC:                              4564.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      6.8379      0.105     65.272      0.000       6.632       7.044
D             11.6540      0.149     78.187      0.000      11.361      11.946
==============================================================================
Omnibus:                     7278.240   Durbin-Watson:                   1.966
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               94.873
Skew:                          -0.097   Prob(JB):                     2.50e-21
Kurtosis:                       1.504   Cond. No.                         2.60
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly␣
↪specified.
```

Now we condition on X to see where conditioning might help in obtaining an unbiased estimate of the true effect. Note that the treatment effect $(y_i^1 - y_i^0)$ is uncorrelated with d_i within each strata of X in the first example. That is not true in the second example.

Table 6.5 A Rearrangement of the Example in Table 6.4 That Shows How Regression Adjustment Is Differentially Effective

	y_i^1	y_i^0	υ_i^1	υ_i^0	y_i	d_i	x_i	$\upsilon_i^0+d_i(\upsilon_i^1-\upsilon_i^0)$
Regression adjustment with X generates an unbiased estimate for D								
For those with $X=1$								
In treatment group	20	10	2.5	2.5	20	1	1	2.5
In treatment group	20	10	2.5	2.5	20	1	1	2.5
In control group	20	10	2.5	2.5	10	0	1	2.5
For those with $X=0$								
In treatment group	15	5	−2.5	−2.5	15	1	0	−2.5
In control group	15	5	−2.5	−2.5	5	0	0	−2.5
In control group	15	5	−2.5	−2.5	5	0	0	−2.5
Regression adjustment with X does not generate an unbiased estimate for D								
For those with $X=1$								
In treatment group	20	10	2.83	2.5	20	1	1	2.83
In treatment group	20	10	2.83	2.5	20	1	1	2.83
In control group	18	10	.83	2.5	10	0	1	2.5
For those with $X=0$								
In treatment group	15	5	−2.17	−2.5	15	1	0	−2.17
In control group	15	5	−2.17	−2.5	5	0	0	−2.5
In control group	15	5	−2.17	−2.5	5	0	0	−2.5

[39]:
```python
for sample in range(2):

    df = get_sample_regression_adjustment(sample)
    print("Sample {:}\n".format(sample))

    stat = (df["Y_1"] - df["Y_0"]).mean()
    print(f"True effect:{stat:24.4f}")

    stat = df.query("D == 1")["Y"].mean() - df.query("D == 0")["Y"].mean()
    print(f"Naive estimate:{stat:21.4f}")

    rslt = smf.ols(formula="Y ~ D + X", data=df).fit()
    print(f"Conditional estimate:{rslt.params[1]:15.4f}\n")
```

```
Sample 0

True effect:                10.0000
Naive estimate:             11.6540
Conditional estimate:       10.0000
```

(continues on next page)

(continued from previous page)

```
Sample 1

True effect:                 9.6280
Naive estimate:             11.6540
Conditional estimate:       10.0000
```

To summarize: Regression adjustment by X will yield a consistent and unbiased estimate of the ATE when:

- D is mean independent of (and therefore uncorrelated with) $v^0 + D(v^1 - v^0)$ for each subset of respondent identified by distinct values on the variables in X
- the causal effect of D does not vary with X
- a fully flexible parameterization of X is used

Freedman's paradox

Let's explore some of the challenges of finding the right regression specification.

> In statistical analysis, Freedman's paradox (Freedman, 1983), named after David Freedman, is a problem in model selection whereby predictor variables with no relationship to the dependent variable can pass tests of significance – both individually via a t-test, and jointly via an F-test for the significance of the regression. (Wikipedia)

We fill a dataframe with random numbers. Thus there is no causal relationship between the dependent and independent variables.

[33]:
```
columns = ["Y"]
[columns.append("X{:}".format(i)) for i in range(50)]
df = pd.DataFrame(np.random.normal(size=(100, 51)), columns=columns)
```

Now we run a simple regression of the random independent variables on the dependent variable.

[34]:
```
formula = "Y ~ " + " + ".join(columns[1:])
rslt = smf.ols(formula=formula, data=df).fit()
rslt.summary()
```

[34]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                      Y   R-squared:                       0.545
Model:                            OLS   Adj. R-squared:                  0.081
Method:                 Least Squares   F-statistic:                     1.176
Date:                Tue, 26 May 2020   Prob (F-statistic):              0.286
Time:                        11:32:24   Log-Likelihood:                -108.43
No. Observations:                 100   AIC:                             318.9
Df Residuals:                      49   BIC:                             451.7
Df Model:                          50
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
```

(continues on next page)

(continued from previous page)

```
Intercept       0.1106      0.136      0.811      0.421      -0.163       0.384
X0             -0.0922      0.222     -0.416      0.679      -0.538       0.353
X1             -0.1713      0.141     -1.217      0.229      -0.454       0.111
X2             -0.0741      0.155     -0.478      0.635      -0.386       0.237
X3              0.1575      0.134      1.178      0.244      -0.111       0.426
X4              0.2736      0.172      1.595      0.117      -0.071       0.618
X5             -0.0649      0.172     -0.378      0.707      -0.410       0.280
X6              0.0063      0.141      0.045      0.964      -0.276       0.289
X7              0.1089      0.141      0.774      0.443      -0.174       0.392
X8              0.0694      0.139      0.500      0.619      -0.210       0.348
X9              0.3075      0.149      2.059      0.045       0.007       0.608
X10             0.0189      0.168      0.113      0.911      -0.318       0.356
X11            -0.2898      0.178     -1.626      0.110      -0.648       0.068
X12             0.0207      0.129      0.160      0.873      -0.239       0.280
X13             0.0901      0.119      0.757      0.453      -0.149       0.329
X14            -0.1296      0.154     -0.843      0.403      -0.439       0.179
X15             0.0618      0.153      0.405      0.687      -0.245       0.368
X16            -0.1151      0.122     -0.944      0.350      -0.360       0.130
X17             0.1265      0.170      0.744      0.460      -0.215       0.468
X18             0.1578      0.140      1.129      0.264      -0.123       0.439
X19            -0.0752      0.157     -0.477      0.635      -0.392       0.241
X20            -0.1454      0.133     -1.097      0.278      -0.412       0.121
X21             0.1507      0.150      1.005      0.320      -0.151       0.452
X22             0.4160      0.133      3.123      0.003       0.148       0.684
X23            -0.1169      0.156     -0.750      0.457      -0.430       0.196
X24            -0.2271      0.165     -1.376      0.175      -0.559       0.104
X25             0.1651      0.172      0.962      0.341      -0.180       0.510
X26             0.1461      0.123      1.192      0.239      -0.100       0.392
X27            -0.2343      0.139     -1.685      0.098      -0.514       0.045
X28             0.0508      0.138      0.368      0.715      -0.227       0.329
X29            -0.0187      0.191     -0.098      0.922      -0.403       0.365
X30             0.2245      0.149      1.511      0.137      -0.074       0.523
X31             0.0353      0.146      0.242      0.810      -0.258       0.329
X32             0.0666      0.152      0.438      0.663      -0.239       0.372
X33            -0.0281      0.151     -0.186      0.853      -0.331       0.275
X34             0.0204      0.141      0.144      0.886      -0.263       0.304
X35            -0.1940      0.138     -1.409      0.165      -0.471       0.083
X36             0.1215      0.144      0.843      0.403      -0.168       0.411
X37             0.3450      0.171      2.023      0.049       0.002       0.688
X38             0.2652      0.148      1.787      0.080      -0.033       0.563
X39             0.0370      0.167      0.221      0.826      -0.299       0.373
X40            -0.0072      0.147     -0.049      0.961      -0.302       0.288
X41             0.2258      0.154      1.469      0.148      -0.083       0.535
X42            -0.1910      0.154     -1.242      0.220      -0.500       0.118
X43             0.1973      0.139      1.423      0.161      -0.081       0.476
X44            -0.1017      0.136     -0.750      0.457      -0.374       0.171
X45            -0.1656      0.137     -1.210      0.232      -0.441       0.109
X46            -0.0255      0.152     -0.168      0.867      -0.330       0.279
X47             0.1621      0.148      1.094      0.279      -0.136       0.460
X48             0.1768      0.144      1.228      0.225      -0.113       0.466
X49             0.1961      0.146      1.345      0.185      -0.097       0.489
==============================================================================
```

(continues on next page)

(continued from previous page)

```
Omnibus:                        0.122   Durbin-Watson:                   2.167
Prob(Omnibus):                  0.941   Jarque-Bera (JB):                0.302
Skew:                          -0.007   Prob(JB):                        0.860
Kurtosis:                       2.731   Cond. No.                         6.59
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly␣
↪specified.
"""
```

We use this to inform a second regression where we only keep the variables that were significant at the 25% level.

[14]:
```python
final_covariates = list()
for label in rslt.params.keys():
    if rslt.pvalues[label] > 0.25:
        continue
    final_covariates.append(label)

formula = "Y ~ " + " + ".join(final_covariates)
rslt = smf.ols(formula=formula, data=df).fit()
rslt.summary()
```

[14]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                      Y   R-squared:                       0.402
Model:                            OLS   Adj. R-squared:                  0.260
Method:                 Least Squares   F-statistic:                     2.834
Date:                Tue, 26 May 2020   Prob (F-statistic):           0.000627
Time:                        07:19:03   Log-Likelihood:                -122.11
No. Observations:                 100   AIC:                             284.2
Df Residuals:                      80   BIC:                             336.3
Df Model:                          19
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0853      0.103      0.832      0.408      -0.119       0.289
X1            -0.0874      0.099     -0.879      0.382      -0.285       0.111
X3             0.1712      0.101      1.698      0.093      -0.029       0.372
X4             0.1905      0.111      1.710      0.091      -0.031       0.412
X9             0.2888      0.104      2.768      0.007       0.081       0.496
X11           -0.2161      0.119     -1.812      0.074      -0.453       0.021
X22            0.3742      0.100      3.735      0.000       0.175       0.574
X24           -0.2796      0.103     -2.716      0.008      -0.484      -0.075
X26            0.1391      0.094      1.484      0.142      -0.047       0.326
X27           -0.2348      0.103     -2.283      0.025      -0.439      -0.030
X30            0.1220      0.110      1.109      0.271      -0.097       0.341
X35           -0.1628      0.093     -1.742      0.085      -0.349       0.023
X37            0.2214      0.100      2.206      0.030       0.022       0.421
X38            0.2400      0.106      2.267      0.026       0.029       0.451
```

(continues on next page)

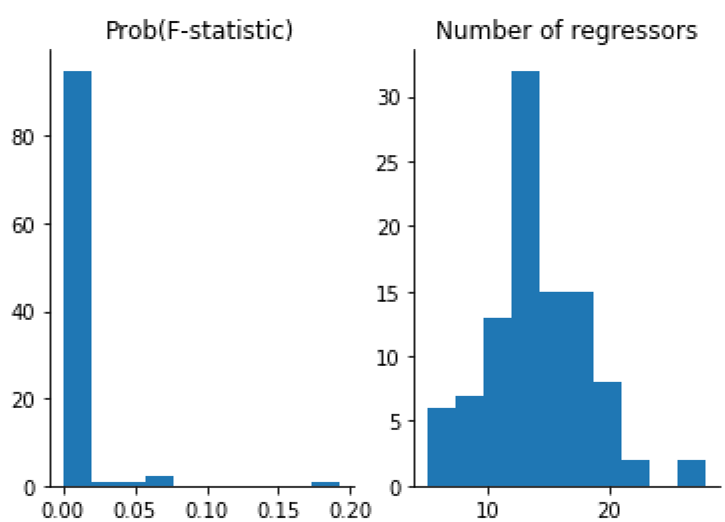
(continued from previous page)

```
X41           0.0482      0.102      0.471      0.639      -0.155       0.252
X42          -0.1669      0.113     -1.478      0.143      -0.392       0.058
X43           0.1723      0.096      1.786      0.078      -0.020       0.364
X45          -0.1853      0.101     -1.838      0.070      -0.386       0.015
X48           0.1667      0.092      1.811      0.074      -0.016       0.350
X49           0.1491      0.100      1.492      0.140      -0.050       0.348
==============================================================================
Omnibus:                        2.282   Durbin-Watson:                   2.259
Prob(Omnibus):                  0.319   Jarque-Bera (JB):                1.718
Skew:                           0.137   Prob(JB):                        0.424
Kurtosis:                       2.420   Cond. No.                         2.43
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly␣
↪specified.
"""
```

What to make of this exercise?

[15]:
```
np.random.seed(213)

df = pd.DataFrame(columns=["F-statistic", "Regressors"])
for i in range(100):
    model = run_freedman_exercise()
    df["Regressors"].loc[i] = len(model.params)
    df["F-statistic"].loc[i] = model.f_pvalue
```

[16]:
```
plot_freedman_exercise(df)
```



Resources

- **Goldberger, A. S. (1991)**. A course in econometrics. Cambridge, MA: *Harvard University Press*.
- **F. J. Anscombe (1973)**. Graphs in Statistical Analysis. *The American Statistician*, 27, 17–21.
- **Freedman, David A.; Freedman, David A. (1983)**. A Note on Screening Regression Equations. *The American Statistician*. 37 (2), 152–155.

1.8 Heterogeneity, selection, and causal graphs

We revisit the issues of treatment effect heterogeneity and individuals' selecting their treatment status based on gains unobserved by the econometrician. We lay the groundwork to estimate causal effects using instrumental variables, front-door identification with causal mechanisms, and conditioning estimators using pretreatment variables. We work through an elaborate panel data demonstration that illustrates the shortcoming of conditioning estimators in the presence of self-selection.

1.8.1 Self-selection, heterogeneity, and causal graphs

Introduction

Alternatives to back-door identification

The next chapters deal with:

- instrumental variables
- front-door identification with causal mechanisms
- conditioning estimators using pretreatment variables

Why do we need to consider alternatives?

$\rightarrow$selection on unobservables / nonignorability of treatment

What makes an unobservable?

- simple confounding, stable unobserved common cause of treatment and outcome variable
- subtle confounding, direct self-selection into the treatment based on accurate perceptions of the individual level treatment effect

Selection on unobservables as a combination of two features:

- treatment effect heterogeneity
- self-selection

Nonignorability and selection on the unobservables

Selection on observables

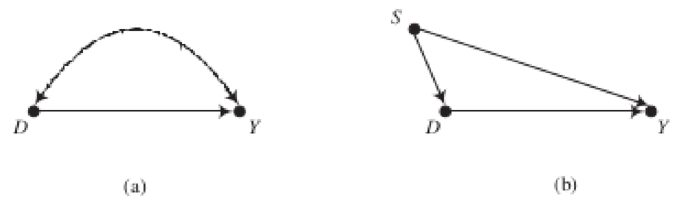


Figure 4.8 Causal diagrams in which treatment assignment is (a) nonignorable and (b) ignorable.

Selection on unobservables

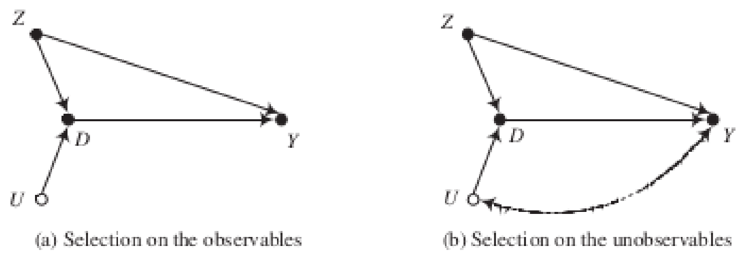


Figure 4.9 Causal diagrams for the terminology from econometric modeling of treatment selection.

Selection on the unobservables and the utility of additional posttreatment measures of the outcome

Catholic school example

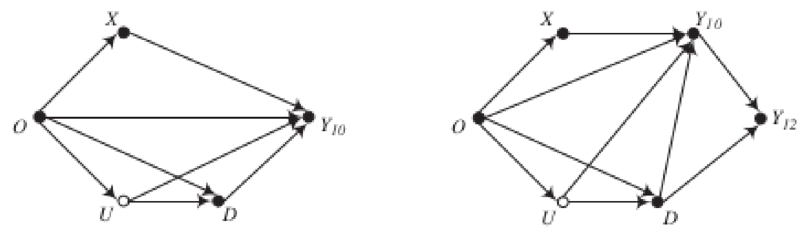
- claim that Catholic schools are more effective than public schools in teaching mathematics and reading to equivalent High School students.
- conditioning on family background and motivation to learn
- those enrolling into Catholic school have the most to gain from doing so

Notation

- Y_{10}, observed score on standardized achievement test given in tenth grade
- D causal variable taking value one if student attends Catholic school
- U unobserved motivation to learn, differences in home environment, anticipation of causal effect itself
- X determinants of achievement tests that have no direct causal effect on school sector or selection
- O ultimate background variables that affect all other variables in graph

We proceed in two steps:

- assess identification for given directed graphs
- examine structure of directed graph itself



(a) The identification puzzle (b) Coleman's approach

Figure 8.1 Coleman's strategy for the identification of the causal effect of Catholic schooling on achievement.

We cannot identify the causal effect of D on Y_{10} in subfigure (a) but in subfigure (b). However, at what cost?

Y_{10} blocks all back-door paths, however it does not satisfy the Condition 2 of the back-door criterion. As such, it adjusts away some of the total causal effect of D on Y_{12}.

Let E denote an student's ability for test taking and allow for the direct effect of U on bow both achievement scores. Then maybe this is a more complete picture?

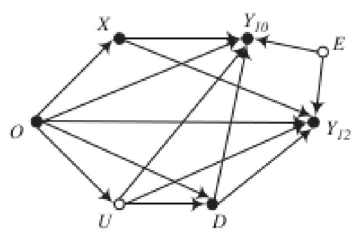


Figure 8.2 Criticism of Coleman's estimates of the effect of Catholic schooling on learning.

Back-door adjustment by Y_{10} ineffective again after revisiting economic implications of the imposed graph. In fact, Y_{10} is now a collider variable that induces a noncausal dependence.

Panel Data Demonstration

The motivation behind this example is simply to show that we cannot learn anything about the underlying causal effect with the conventional strategies and how we model self-selection in the data generating process.

[2]:
```python
def get_propensity_score(o, u):
    """Get the propensity score."""
    level = -3.8 + o + u
    return np.exp(level) / (1 + np.exp(level))


def get_treatment_status(o, u):
    """Sampling treatment status"""
    # Following the causal graph, the treatment indicator is only a function
    # of the background characteristics O and the unobservable U.
    p = get_propensity_score(o, u)
    return np.random.choice([1, 0], p=[p, 1 - p])
```

(continues on next page)

(continued from previous page)

```
def get_covariates():
    """Get covariates."""
    o, e = np.random.normal(size=2)
    x, u = o + np.random.normal(size=2)
    return o, u, x, e
```

[3]:
```
def get_potential_outcomes(grade, o, x, e, u, scenario=0, selection=False):
    """Get potential outcomes.

    Sampling of potential outcome of an individual for the panel data demonstration.

    Args:
        grade: an integer for the grade the individual is in.
        o, x, e : floats of observable characteristics.
        u: a float of unobservable characteristic.
        scenario: an integer for the scenario: (0) no role for E, (1) role for E.
        selection: a boolean indicating whether there is selection on unobservables.

    Returns:
        A tuple of potential outcomes (Y_0, Y_1).

    """
    # We want to make sure we only pass in valid input.
    assert scenario in range(2)
    assert selection in [True, False]
    assert grade in [10, 11, 12]

    # There is a natural progression in test scores.
    level = dict()
    level[10] = 100
    level[11] = 101
    level[12] = 102

    if scenario == 0:
        y_0 = level[grade] + o + u + x + np.random.normal()
    elif scenario == 1:
        y_0 = level[grade] + o + u + x + e + np.random.normal()
    else:
        raise NotImplementedError

    # Sampling of treatment effects. The key difference for selection on unobservables␣
↪is in how
    # the overall treatment effect depends on the  unobservable U that also affects the␣
↪choice
    # probability. This was the major criticism of Coleman's work.
    delta_1 = np.random.normal(loc=10, scale=1)
    if selection:
        delta_2 = np.random.normal(loc=u)
    else:
        delta_2 = np.random.normal()
```

(continues on next page)

(continued from previous page)

```
    if grade == 10:
        y_1 = y_0 + delta_1 + delta_2
    elif grade == 11:
        y_1 = y_0 + (1 + delta_1) + delta_2
    elif grade == 12:
        y_1 = y_0 + (2 + delta_1) + delta_2

    return y_0, y_1
```

```
[4]: def get_sample_panel_demonstration(num_agents=1000, scenario=0, selection=False,␣
↪seed=123):
    """Get sample for demonstration.

    Create a random sample for the demonstration of the usefulness of (or lack thereof)␣
↪of having
    additional posttreatment measures of the outcome.

    Args:
        num_agents: an integer for the number of agents in the sample
        scenario: an integer that indicates whether to include E as a determinant of␣
↪test scores
        selection: a boolean variable indicating whether selection on unobservables is␣
↪an issue
        seed: an integer setting the random seed

    Returns:
        A dataframe with the simulated sample.

    """
    # We first initialize an empty DataFrame that holds the information for each␣
↪individual
    # and each time period.
    columns = ["Y", "D", "O", "X", "E", "U", "Y_1", "Y_0"]
    index = product(range(num_agents), [10, 11, 12])
    index = pd.MultiIndex.from_tuples(index, names=("Identifier", "Grade"))
    df = pd.DataFrame(columns=columns, index=index)

    # Now we are ready to simulate the sample with the desired characteristics.
    np.random.seed(seed)
    for i in range(num_agents):

        o, u, x, e = get_covariates()
        d = get_treatment_status(o, u)
        for grade in [10, 11, 12]:
            y_0, y_1 = get_potential_outcomes(grade, o, x, e, u, scenario, selection)
            y = d * y_1 + (1 - d) * y_0
            df.loc[(i, grade), :] = [y, d, o, x, e, u, y_1, y_0]

    # Finally some type definitions for pretty output.
    df = df.astype(np.float)
    df = df.astype({"D": np.int})
```

(continues on next page)

(continued from previous page)

```
    return df
```

```
[5]: num_agents, scenario, selection = 1000, 0, False
df = get_sample_panel_demonstration(num_agents, scenario, selection)
df.head()
```

```
/home/sebastian/anaconda3/envs/grmpy/lib/python3.7/site-packages/ipykernel_launcher.py:
↪36: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To␣
↪silence this warning, use `float` by itself. Doing this will not modify any behavior␣
↪and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/
↪release/1.20.0-notes.html#deprecations
/home/sebastian/anaconda3/envs/grmpy/lib/python3.7/site-packages/ipykernel_launcher.py:
↪37: DeprecationWarning: `np.int` is a deprecated alias for the builtin `int`. To␣
↪silence this warning, use `int` by itself. Doing this will not modify any behavior and␣
↪is safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `np.int32` to␣
↪specify the precision. If you wish to review your current use, check the release note␣
↪link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/
↪release/1.20.0-notes.html#deprecations
```

```
[5]:                           Y  D         O         X         E         U  \
Identifier Grade
0          10      95.841898  0 -1.085631 -0.802652  0.997345 -2.591925
           11      98.499140  0 -1.085631 -0.802652  0.997345 -2.591925
           12      97.072351  0 -1.085631 -0.802652  0.997345 -2.591925
1          10      97.544210  0 -0.619191 -0.042445 -0.769433 -0.492665
           11     100.583068  0 -0.619191 -0.042445 -0.769433 -0.492665

                         Y_1         Y_0
Identifier Grade
0          10     105.586179   95.841898
           11     106.765876   98.499140
           12     110.597380   97.072351
1          10     108.934451   97.544210
           11     112.137966  100.583068
```

What is the average treatment effect and how does it depend on the presence of selection?

```
[6]: num_agents, scenario = 1000, 0

# This setup allows to freeze some arguments of the function
# that do not change during the analysis.
from functools import partial  # noqa: E402

simulate_sample = partial(get_sample_panel_demonstration, num_agents, scenario)

for selection in [False, True]:
    print(f" Selection {selection}")
    df = simulate_sample(selection)
    for grade in [10, 12]:
        df_grade = df.loc[(slice(None), grade), :]
        stat = (df_grade["Y_1"] - df_grade["Y_0"]).mean()
```

(continues on next page)

(continued from previous page)

```
        print(f" Grade {grade}:  ATE {stat:5.3f}")
    print("\n")
```

```
Selection False
```

```
/home/sebastian/anaconda3/envs/grmpy/lib/python3.7/site-packages/ipykernel_launcher.py:
↪36: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To␣
↪silence this warning, use `float` by itself. Doing this will not modify any behavior␣
↪and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/
↪release/1.20.0-notes.html#deprecations
/home/sebastian/anaconda3/envs/grmpy/lib/python3.7/site-packages/ipykernel_launcher.py:
↪37: DeprecationWarning: `np.int` is a deprecated alias for the builtin `int`. To␣
↪silence this warning, use `int` by itself. Doing this will not modify any behavior and␣
↪is safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `np.int32` to␣
↪specify the precision. If you wish to review your current use, check the release note␣
↪link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/
↪release/1.20.0-notes.html#deprecations
```

```
Grade 10:  ATE 10.090
Grade 12:  ATE 12.014


Selection True
Grade 10:  ATE 10.116
Grade 12:  ATE 12.039
```

The average treatment effects are unaffected by selection. But how does the picture change when we look at subsets of the population?

[7]:
```
for selection in [False, True]:
    print(f" Selection {selection}")
    df = simulate_sample(selection)
    for grade in [10, 12]:
        subset = df.loc[(slice(None), grade), :]

        stat = list()
        for status in range(2):
            df_status = subset.query(f"D == {status}")
            stat.append((df_status["Y_1"] - df_status["Y_0"]).mean())

        print(" Grade {:}:  ATC {:7.3f}   ATT {:7.3f}".format(grade, *stat))
    print("\n")
```

```
Selection False
```

```
/home/sebastian/anaconda3/envs/grmpy/lib/python3.7/site-packages/ipykernel_launcher.py:
↪36: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To␣
↪silence this warning, use `float` by itself. Doing this will not modify any behavior␣
↪and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/
↪release/1.20.0-notes.html#deprecations
```

(continues on next page)

(continued from previous page)

```
/home/sebastian/anaconda3/envs/grmpy/lib/python3.7/site-packages/ipykernel_launcher.py:
↪37: DeprecationWarning: `np.int` is a deprecated alias for the builtin `int`. To␣
↪silence this warning, use `int` by itself. Doing this will not modify any behavior and␣
↪is safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `np.int32` to␣
↪specify the precision. If you wish to review your current use, check the release note␣
↪link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/
↪release/1.20.0-notes.html#deprecations
```

```
 Grade 10:  ATC  10.098   ATT  10.012
 Grade 12:  ATC  12.001   ATT  12.134



 Selection True
 Grade 10:  ATC   9.958   ATT  11.655
 Grade 12:  ATC  11.861   ATT  13.776
```

```
/home/sebastian/anaconda3/envs/grmpy/lib/python3.7/site-packages/ipykernel_launcher.py:
↪36: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To␣
↪silence this warning, use `float` by itself. Doing this will not modify any behavior␣
↪and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/
↪release/1.20.0-notes.html#deprecations
/home/sebastian/anaconda3/envs/grmpy/lib/python3.7/site-packages/ipykernel_launcher.py:
↪37: DeprecationWarning: `np.int` is a deprecated alias for the builtin `int`. To␣
↪silence this warning, use `int` by itself. Doing this will not modify any behavior and␣
↪is safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `np.int32` to␣
↪specify the precision. If you wish to review your current use, check the release note␣
↪link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/
↪release/1.20.0-notes.html#deprecations
```

[8]:
```
for grade in [10, 12]:
    for model in ["Y ~ D", "Y ~ D + X + O"]:
        df_grade = df.loc[(slice(None), grade), :]
        rslt = smf.ols(formula=model, data=df_grade).fit()
        stat = rslt.params["D"]
        print("Grade: {}  Model: {:}".format(*[grade, model]))
        print("   Estimated Treatment Effect: {:5.3f}\n".format(stat))
```

```
Grade: 10  Model: Y ~ D
   Estimated Treatment Effect: 15.767

Grade: 10  Model: Y ~ D + X + O
   Estimated Treatment Effect: 12.181

Grade: 12  Model: Y ~ D
   Estimated Treatment Effect: 17.849

Grade: 12  Model: Y ~ D + X + O
   Estimated Treatment Effect: 14.331
```

(continues on next page)

(continued from previous page)

None of the estimates come even close to our parameters of interest.

Causal graphs for complex patterns of self-selection

We want to make sure that complex patterns of self-selection can be represented by directed graphs.

Separate graphs for separate latent classes

Groups

- $G = 1$, selection of schools mainly for lifestyle reasons, proximity to home and taste for school cultures
- $G = 2$, selection of schools to maximize expected achievement

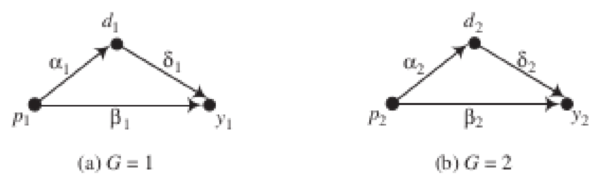


Figure 8.3 Separate causal graphs for two groups of individuals ($G = 1$ and $G = 2$) where the effects of parental background (P) and charter schools (D) on test scores (Y) may differ for the two groups.

What are the economic mechanisms are represented by each of the arrows? Why would we expect them to differ across the two groups?

- families of the second group are more likely to send their children to charter schools $d_2 > d_1$
- parents with higher levels of education are more likely to send their children to charter schools as they value distinctive forms of education $\alpha_1, \alpha_2 > 0$ and are able to support their children with their homework $\beta_1, \beta_2 > 0$.
- existing research suggests $\delta_1, \delta_2 > 0$ and $\delta_2 > \delta_1$ as families in second group put more effort in matching their children to schools

What happens if we block the back-door path by conditioning in P but ignore the existence of two latent classes? If P is associated with latent class membership, then we do not properly weigh the stratum-specific treatment effects as there is heterogeneity within strata.

A single graph that represents all latent classes

We now let G capture effect heterogeneity.

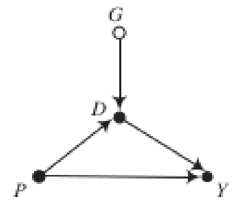


Figure 8.4 A graph where groups are represented by an unobserved latent class variable (G) in a single graph.

We outline more and more elaborate ideas about the economic mechanisms that determine class membership and how they modify the structure of the causal graph.

- **no arrow from :math:\`P\` to :math:\`G\`** implies that students with students who have higher levels of education are no more likely to know of the educational policy dialogue hat claims that charter schools have advantages.
- **no arrow from :math:\`G\` to :math:\`Y\`** implies that there is in fact (on average) no treatment effect heterogeneity.

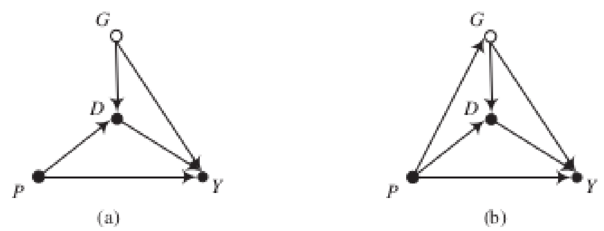


Figure 8.5 Two graphs where selection into charter schools (D) is determined by group (G) and where selection renders the effect of D on Y unidentified as long as G remains unobserved.

Self-selection into the latent class

We now elaborate on the mechanism that determines class membership. We assume that G is at least in part determined by a variable that measures a family's subjective expectations of their child's likely benefit from attending a charter school instead of a regular school.

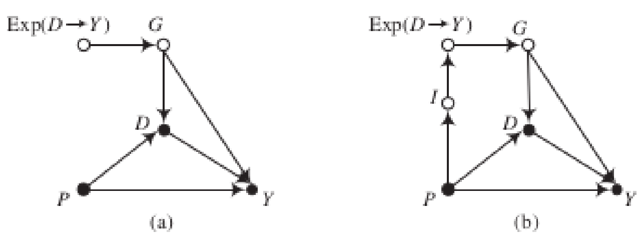


Figure 8.6 Two graphs where selection on the unobservables is given an explicit representation as self-selection on subjective expectations of variation in the causal effect of D on Y. For panel (b), these expectations are determined by information (I) that is differentially available to families with particular parental backgrounds (P).

However, these expectations are potentially based on access to information that is often related to parental background.

1.9 Instrumental variables

We review basic instrumental variables estimation using a simulated example inspired by random assignment of school vouchers. We look at the Wald and 2SLS estimator and discuss its interpretation as a Local Average Treatment Effect in the presence of treatment effect heterogeneity. We conclude with a discussion of seminal papers in the literature and also elevate a more critical assessment to discussion.

1.9.1 Instrumental variable estimators of causal effects

Overview

- Causal effect estimation with a binary IV
- Traditional IV estimators
- Instrumental variable estimators in the presence of individual-level heterogeneity
- Conclusions

Causal effect estimation with a binary IV

We consider the standard relationship

$$Y = \alpha + \delta D + \epsilon,$$

where δ is the true causal effect that (for now) is assumed to be **constant**.

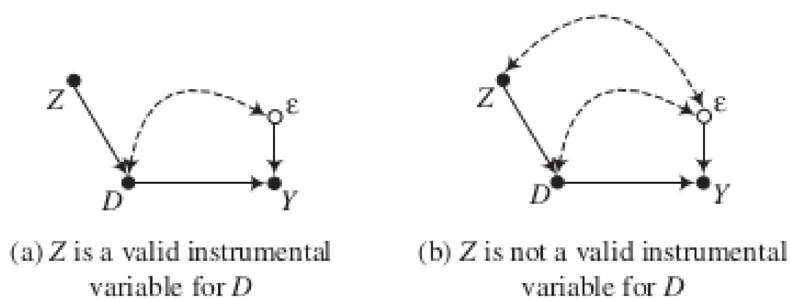


Figure 9.1 Two graphs in which Z is a potential instrumental variable.

- No conditioning estimator would effectively estimate the causal effect of D on Y because no observed variable satisfy the back-door criterion.
- If perfect stratification cannot be be enacted with the available data, one possible solution is to find an exogenous source of variation that determines Y only by way of the causal variable D. The causal effect is then estimated by measuring how much Y varies with the proportion of the total variation in D that is attributable to the exogenous variation.

$$E[Y] = E[\alpha + \delta D + \epsilon] = \alpha + \delta E[D] + E[\epsilon]$$

We can rewrite this as a difference equation in Z:

$$E[Y \mid Z = 1] - E[Y \mid Z = 0] = \delta(E[D \mid Z = 1] - E[D \mid Z = 0]) + (E[\epsilon \mid Z = 1] - E[\epsilon \mid Z = 0])$$

Then we divide both sides by $E[D \mid Z=1] - E[D \mid Z=0]$.

$$\frac{E[Y \mid Z=1] - E[Y \mid Z=0]}{E[D \mid Z=1] - E[D \mid Z=0]} = \frac{\delta(E[D \mid Z=1] - E[D \mid Z=0]) + (E[\epsilon \mid Z=1] - E[\epsilon \mid Z=0])}{E[D \mid Z=1] - E[D \mid Z=0]}$$

If Figure 9.1 (a) is an accurate description of the causal structure, then $E[\epsilon \mid Z=1] = E[\epsilon \mid Z=0] = 0$.

$$\frac{E[Y \mid Z=1] - E[Y \mid Z=0]}{E[D \mid Z=1] - E[D \mid Z=0]} = \delta$$

$$\hat{\delta}_{IV,WALD} = \frac{E[Y \mid Z=1] - E[Y \mid Z=0]}{E[D \mid Z=1] - E[D \mid Z=0]}$$

- The assumption that δ is an invariant structural effect is crucial for this result.

Demonstration dataset

We wish to determine whether private high school outperform public high schools as measured by 9^{th} grade achievement tests. There exists a school voucher program in the city that covers tuition in case one attends private school. However, there are budgetary limits and so the vouchers are available only to 10% of students and allocated by a lottery.

Table 9.1 The Distribution of Voucher Winners by School Sector for IV Demonstration 1

		Public school $d_i = 0$	Private school $d_i = 1$
Voucher loser	$z_i = 0$	8000	1000
Voucher winner	$z_i = 1$	800	200

- Winning the lottery increases private school attendance.

[2]:
```
def get_sample_iv_demonstration():
    """Simulates sample.

    Simulates a sample of 10,000 individuals for the IV demonstration
    based on the information provided in our textbook.

    Notes:

        The school administration distributed 1,000 vouchers for
        private school attendance in order to shift students
        from public into private school. The goals is to increase
        educational achievement.

    Args:
        None
```

(continues on next page)

(continued from previous page)

```
    Returns:
        A pandas Dataframe with the observable characteristics (Y, D, Z)
        for all individuals.

        Y: standardized test for 9th graders
        D: private school attendance
        Z: voucher available

    """
    # We first initialize an empty Dataframe with 10,000 rowns and three
    # columns.
    columns = ["Y", "D", "Z"]
    index = pd.Index(range(10000), name="Identifier")
    df = pd.DataFrame(columns=columns, index=index)

    # We sample the exact number of individuals following the description
    # in Table 9.2.
    for i in range(10000):
        if i < 8000:
            y, d, z = np.random.normal(50), 0, 0
        elif i < 9000:
            y, d, z = np.random.normal(60), 1, 0
        elif i < 9800:
            y, d, z = np.random.normal(50), 0, 1
        else:
            # The lower mean for the observed outcome does indicate
            # that those drawn into treatment due to the instrument
            # only do have smaller gains compared to those that
            # take the treatment regardless.
            y, d, z = np.random.normal(58), 1, 1

        df.loc[i, :] = [y, d, z]

    # We shuffle all rows so we do not have the different subsamples
    # grouped together.
    df = df.sample(frac=1).reset_index(drop=True)

    # We set the types of our columns for prettier formatting later.
    df = df.astype(np.float)
    df = df.astype({"D": np.int, "Z": np.int})

    return df
```

Let's have a look at the structure of the data.

[3]:
```
df = get_sample_iv_demonstration()
df.head()
```

[3]:
```
           Y  D  Z
0  48.606920  0  0
1  50.240003  0  0
2  49.377337  0  0
3  60.885880  1  0
```

(continues on next page)

(continued from previous page)

```
4  50.160785  0  0
```

How about the conditional distribution of observed outcomes?

[4]:
```
df.groupby(["D", "Z"])["Y"].mean()
```

[4]:
```
D  Z
0  0    50.009760
   1    49.962199
1  0    60.034692
   1    58.072959
Name: Y, dtype: float64
```

We can always run an OLS regression first to get a rough sense of the data.

[5]:
```
rslt = smf.ols(formula="Y ~ D", data=df).fit()
rslt.summary()
```

[5]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                      Y   R-squared:                       0.904
Model:                            OLS   Adj. R-squared:                  0.904
Method:                 Least Squares   F-statistic:                 9.374e+04
Date:                Wed, 16 Jun 2021   Prob (F-statistic):               0.00
Time:                        08:40:03   Log-Likelihood:                -14482.
No. Observations:               10000   AIC:                         2.897e+04
Df Residuals:                    9998   BIC:                         2.898e+04
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     50.0054      0.011   4555.317      0.000      49.984      50.027
D              9.7023      0.032    306.173      0.000       9.640       9.764
==============================================================================
Omnibus:                       12.957   Durbin-Watson:                   2.022
Prob(Omnibus):                  0.002   Jarque-Bera (JB):               13.196
Skew:                          -0.074   Prob(JB):                      0.00136
Kurtosis:                       3.100   Cond. No.                         3.13
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly␣
↪specified.
"""
```

However, to exploiting the structure of the dataset, we rather want to compute the IV estimate.

[6]:
```
def get_wald_estimate(df):
    """Calculate Wald estimate.

    Calculates the Wald estimate for the causal effect of treatment
```

(continues on next page)

(continued from previous page)

```
    on an observed outcome using a binary instrument.

    Args:
        df: A pandas DataFrame

    Returns:
        A float with the estimated causal effect.

    """
    # We compute the average difference in observed outcomes.
    average_outcome = df.groupby("Z")["Y"].mean().to_dict()
    numerator = average_outcome[1] - average_outcome[0]

    # We compute the average difference in treatment uptake.
    average_treatment = df.groupby("Z")["D"].mean().to_dict()
    denominator = average_treatment[1] - average_treatment[0]

    rslt = numerator / denominator

    return rslt
```

So, let's see.

[7]:
```
rslt = get_wald_estimate(df)
print(" Wald estimate: {:5.3f}".format(rslt))
```

```
 Wald estimate: 5.183
```

Traditional IV estimators

We now move beyond a binary instrument.

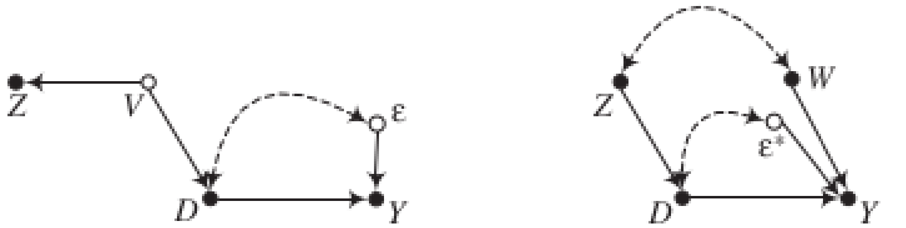
$$\hat{\delta}_{IV} \equiv \frac{Cov_N(y_i, z_i)}{Cov_N(d_i, z_i)}$$

Moving towards the population-level relationships:

$$\begin{aligned}\frac{Cov(Y,Z)}{Cov(D,Z)} &= \frac{\delta Cov(D,Z) + Cov[\epsilon, Z]}{Cov(D,Z)} \\ &= \delta\end{aligned}$$

So, this suggests that:

$$\frac{Cov_N(y_i, z_i)}{Cov_N(d_i, z_i)} \xrightarrow{p} \delta$$



(a) Z is a valid surrogate instrumental variable

(b) Z is a valid conditional instrumental variable

Figure 9.2 Two graphs in which Z is a valid IV.

Returning to our simulated example, we can now apply the two-stage least squares (2SLS) estimator you are familiar with.

[8]:
```
df["D_pred"] = smf.ols(formula="D ~ Z", data=df).fit().predict()
smf.ols(formula="Y ~ D_pred", data=df).fit().summary()
```

[8]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                      Y   R-squared:                       0.002
Model:                            OLS   Adj. R-squared:                  0.002
Method:                 Least Squares   F-statistic:                     17.39
Date:                Wed, 16 Jun 2021   Prob (F-statistic):           3.07e-05
Time:                        08:40:03   Log-Likelihood:                -26171.
No. Observations:               10000   AIC:                         5.235e+04
Df Residuals:                    9998   BIC:                         5.236e+04
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     50.5478      0.153    330.860      0.000      50.248      50.847
D_pred         5.1830      1.243      4.170      0.000       2.747       7.619
==============================================================================
Omnibus:                     3794.296   Durbin-Watson:                   1.993
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            11127.197
Skew:                           2.065   Prob(JB):                         0.00
Kurtosis:                       6.107   Cond. No.                         38.0
==============================================================================


Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly␣
↪specified.
"""
```

Given the structure of our example, both estimators are equivalent. As of now, statsmodels does not provide good support for the instrumental variables estimation. That is true for a host of methods often used by economists. Often linearmodels provides a viable alternative.

[11]:
```python
from linearmodels import IV2SLS  # noqa: E402

df["const"] = 1
IV2SLS(df["Y"], df["const"], df["D"], df["Z"]).fit()
```

[11]:
```
                          IV-2SLS Estimation Summary
==============================================================================
Dep. Variable:                      Y   R-squared:                      0.7076
Estimator:                    IV-2SLS   Adj. R-squared:                 0.7075
No. Observations:               10000   F-statistic:                    77.109
Date:                Wed, Jun 16 2021   P-value (F-stat)                0.0000
Time:                        08:43:00   Distribution:                  chi2(1)
Cov. Estimator:                robust

                             Parameter Estimates
==============================================================================
            Parameter  Std. Err.     T-stat    P-value    Lower CI    Upper CI
------------------------------------------------------------------------------
const          50.548     0.0748     676.20     0.0000      50.401      50.694
D              5.1830     0.5902     8.7812     0.0000      4.0261      6.3398
==============================================================================

Endogenous: D
Instruments: Z
Robust Covariance (Heteroskedastic)
Debiased: False
IVResults, id: 0x7f19084a4d60
```

Instrumental variable estimators in the presence of individual-level heterogeneity

$$\begin{aligned} Y &= Y^0 + D(Y^1 - Y^0) \\ &= Y^0 + \delta D \\ &= \mu^0 + \delta D + \nu^0, \end{aligned}$$

where $\mu^0 \equiv E[Y^0]$ and $\nu^0 \equiv Y^0 - E[Y^0]$. Here, δ now has a clear interpretation.

We need to add a four-category latent variable C:

$$\begin{aligned} \text{Compliers (C = c)} : D^{Z=0} &= 0 \text{ and } D^{Z=1} = 1 \\ \text{Defiers (C = d)} : D^{Z=0} &= 1 \text{ and } D^{Z=1} = 0 \\ \text{Always takers (C = a)} : D^{Z=0} &= 1 \text{ and } D^{Z=1} = 1 \\ \text{Never takers (C = n)} : D^{Z=0} &= 0 \text{ and } D^{Z=1} = 0 \end{aligned}$$

Analogously to the definition of the observed outcome, Y, the observed treatment indicator variable D can then be defined as

$$\begin{aligned} D &= D^{Z=0} + (D^{Z=1} - D^{Z=0})Z \\ &= D^{Z=0} + \kappa Z \end{aligned}$$

What is the value of κ for the different latent groups?

Identifying assumptions for the Local Average Treatment Effect

- Independence, $(Y^1, Y^0, D^{Z=1}, D^{Z=0}) \perp\!\!\!\perp Z$
- Nonzero effect of instrument, $\kappa \neq 0$ for at least some i
- Monotonicty assumption, either $\kappa \geq 0$ for all i or $\kappa \leq 0$ for all i

If these assumptions are valid, then an instrument Z identifies the $LATE$: the average treatment effect for the subset of the population whose treatment selection is induced by the treatment.

$$\hat{\delta}_{IV,WALD} \xrightarrow{p} E[\delta \mid C = c]$$

Table 9.2 The Joint Probability Distribution and Conditional Expectations of the Test Score for Voucher Winner by School Sector for IV Demonstrations 1 and 2

		Public school $d_i = 0$	Private school $d_i = 1$
Voucher loser	$z_i = 0$	$N = 8000$	$N = 1000$
		$\Pr_N[.,.] = .8$	$\Pr_N[.,.] = .1$
		$E_N[y_i \vert .,.] = 50$	$E_N[y_i \vert .,.] = 60$
Voucher winner	$z_i = 1$	$N = 800$	$N = 200$
		$\Pr_N[.,.] = .08$	$\Pr_N[.,.] = .02$
		$E_N[y_i \vert .,.] = 50$	$E_N[y_i \vert .,.] = 58$

What can we learn about the different latent groups?

- Monotonicity, there are no defiers
- Independence, the same distribution of never takes, always takers, and compliers is present among voucher groups

$$\frac{Pr_N[d_i = 1, z_i = 0]}{Pr_N[z_i = 0]} \xrightarrow{p} Pr[C = a]$$
$$\frac{Pr_N[d_i = 0, z_i = 1]}{Pr_N[z_i = 1]} \xrightarrow{p} Pr[C = n]$$

We also know $Pr[C = d] = 0$ and thus

$$1 - \frac{Pr_N[d_i = 1, z_i = 0]}{Pr_N[z_i = 0]} - \frac{Pr_N[d_i = 0, z_i = 1]}{Pr_N[z_i = 1]} \xrightarrow{p} Pr[C = c]$$

Table 9.3 The Distribution of Never Takers, Compliers, and Always Takers for IV Demonstration 2

		Public school $d_i = 0$	Private school $d_i = 1$
Voucher loser	$z_i = 0$	7200 Never takers 800 Compliers	1000 Always takers
Voucher winner	$z_i = 1$	800 Never takers	111 Always takers 89 Compliers

How can we learn about the LATE from the information analyzed so far?

$$E[\delta \mid C = c] = E[Y^1 - Y^0 \mid C = c]$$

Let's start with the following:

$$\begin{aligned} E[Y \mid D = 1, Z = 1] = & \frac{Pr[C = c]}{Pr[C = c] + Pr[C = a]} E[Y^1 \mid C = c] \\ & + \frac{Pr[C = a]}{Pr[C = c] + Pr[C = a]} E[Y^1 \mid C = a] \end{aligned}$$

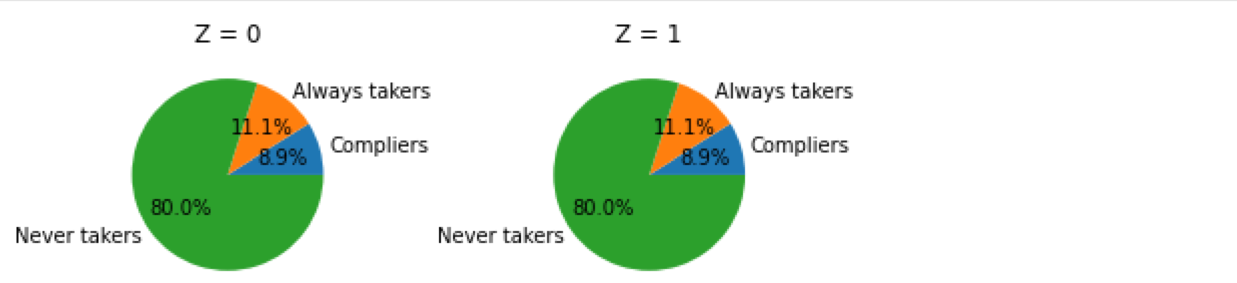
$$\begin{aligned} E[Y \mid D = 0, Z = 0] = & \frac{Pr[C = c]}{Pr[C = c] + Pr[C = n]} E[Y^0 \mid C = c] \\ & + \frac{Pr[C = n]}{Pr[C = c] + Pr[C = n]} E[Y^0 \mid C = n] \end{aligned}$$

Note that we can consistent estimates for $E[Y^0 \mid C = n]$ and $E[Y^1 \mid C = a]$ are provided in the table directly.

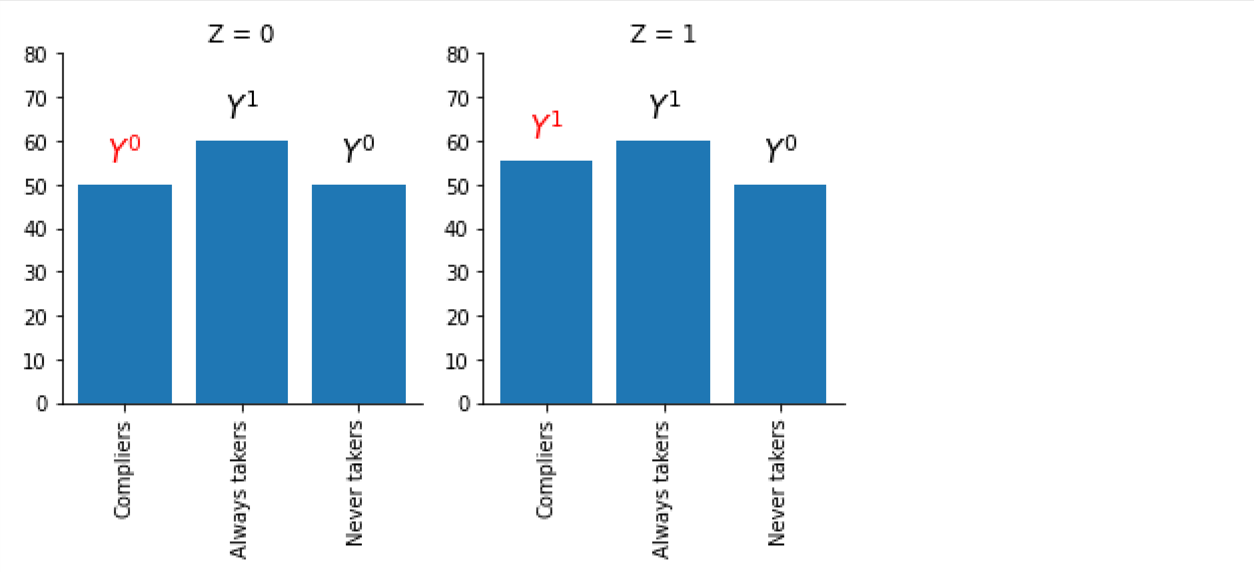
Now lets tie this back to the Wald estimator:

$$\hat{\delta}_{IV,WALD} = \frac{E[Y \mid Z = 1] - E[Y \mid Z = 0]}{E[D \mid Z = 1] - E[D \mid Z = 0]}$$

[12]:
```
get_shares_latent_groups()
```



[13]:
```
get_outcome_latent_groups()
```



Criticism

- instrument-dependent parameter
- limited policy-relevance

Discussion

We revisit and discuss the discussion of the LATE's usefulness.

Lifetime Earnings and the Vietnam Era Draft Lottery: Evidence from Social Security Administrative Records

By JOSHUA D. ANGRIST*

The randomly assigned risk of induction generated by the draft lottery is used to construct estimates of the effect of veteran status on civilian earnings. These estimates are not biased by the fact that certain types of men are more likely than others to service in the military. Social Security administrative records indicate that in the early 1980s, long after their service in Vietnam was ended, the earnings of white veterans were approximately 15 percent less than the earnings of comparable nonveterans. (JEL 824)

A central question in the debate over military manpower policy is whether veterans are adequately compensated for their service. The political process clearly reflects the desire to compensate veterans: since World War II, millions of veterans have enjoyed benefits for medical care, education and training, housing, insurance, and job placement. Recent legislation provides additional benefits for veterans of the Vietnam era. Yet, academic research has not shown conclusively that Vietnam (or other) veterans are worse off economically than nonveterans. Many studies find that Vietnam veterans earn less than nonveterans, but others find positive effects, or effects that vary with age and schooling. Regarding the general position of veterans, a member of the Twentieth Century Fund's Task Force on Policies Toward Veterans concludes that "Within any age group, veterans have higher incomes, more education, and lower unemployment rates than their nonveteran counterparts."[1]

The goal of this paper is to measure the long-term labor market consequences of military service during the Vietnam era. Previous research comparing civilian earnings by veteran status may be biased by the fact that certain types of men are more likely to serve in the armed forces than others. For example, men with relatively few civilian opportunities are probably more likely to enlist. Estimation strategies that do not control for differences in civilian earnings potential will incorrectly attribute lower civilian earnings of veterans to military service. The research reported here overcomes such statistical problems by using the Vietnam era draft

*Department of Economics, Harvard University, Cambridge, MA 02138. Grateful thanks go to Warren Buckler, Cresston Smith, Ada Enis, and Bea Matsui for their assistance in producing the Social Security data; to Chester Bowie for his help in producing the SIPP data; and to Mike Dove for providing DMDC administrative records. Special thanks also go to David Card and Whitney Newey, from whose instruction and comments I have benefited greatly, and to Alan Krueger and an anonymous referee, whose careful reviews of an earlier draft led to substantial improvement. Data collection for this project was funded by the Princeton Industrial Relations Section. Funds for computation and financial support of the author were provided by the Industrial Relations Section, the Princeton Department of Economics, the Sloan Foundation, and the Olin Foundation.

[1]The quote is from Michael Taussig (1974, p. 51). Legislation pertaining to veterans benefits is outlined in Veterans Administration (1984) and in other annual reports of the Veterans Administration. Studies by Sherwin Rosen and Paul Taubman (1982), Saul Schwartz (1986), and Jon Crane and David Wise (1987) find that Vietnam veterans earn less than nonveterans. Dennis DeTray (1982) and Mark Berger and Barry Hirsch (1983) find some positive effects for different age and schooling classes, and Veterans Administration (1981a) researchers find an overall positive effect.

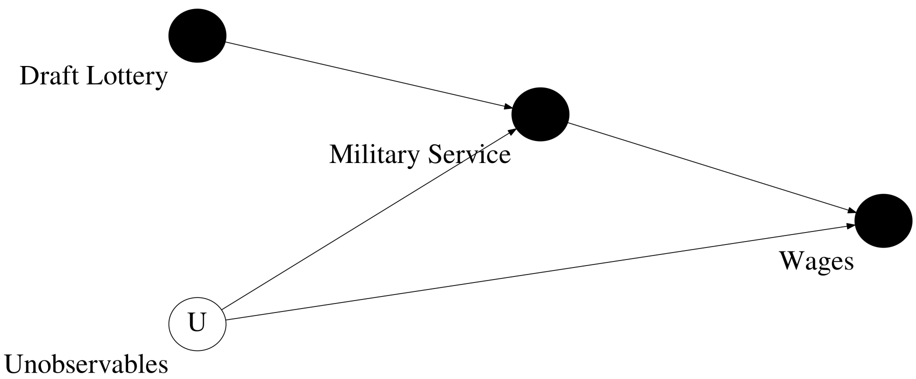


Table of contents

- Introduction
- Background and Data
  - National Random Selection
  - Social Security Earnings Data
- The Effect of Draft Eligibility and Earnings
- The Effect of Military Service on Earnings
  - Estimates Using Draft Eligibility
  - Efficient Instrumental Variables Estimates
- Military Service of Labor Market Experience
- Caveats
  - Treatment Effect Heterogeneity
  - The Absence of Covariates
  - Earnings-Modifying Draft Avoidance Behavior
- Conclusions

THE
QUARTERLY JOURNAL
OF ECONOMICS

Vol. CVI November 1991 Issue 4

DOES COMPULSORY SCHOOL ATTENDANCE AFFECT SCHOOLING AND EARNINGS?*

JOSHUA D. ANGRIST AND ALAN B. KRUEGER

We establish that season of birth is related to educational attainment because of school start age policy and compulsory school attendance laws. Individuals born in the beginning of the year start school at an older age, and can therefore drop out after completing less schooling than individuals born near the end of the year. Roughly 25 percent of potential dropouts remain in school because of compulsory schooling laws. We estimate the impact of compulsory schooling on earnings by using quarter of birth as an instrument for education. The instrumental variables estimate of the return to education is close to the ordinary least squares estimate, suggesting that there is little bias in conventional estimates.

Every developed country in the world has a compulsory schooling requirement, yet little is known about the effect these laws have on educational attainment and earnings.[1] This paper exploits an unusual natural experiment to estimate the impact of compulsory schooling laws in the United States. The experiment stems from the fact that children born in different months of the year start school at different ages, while compulsory schooling laws generally require students to remain in school until their sixteenth or seventeenth birthday. In effect, the interaction of school-entry requirements and compulsory schooling laws compel students born

*We thank Michael Boozer and Lisa Krueger for outstanding research assistance. Financial support was provided by the Princeton Industrial Relations Section, an NBER Olin Fellowship in Economics, and the National Science Foundation (SES-9012149). We are also grateful to Lawrence Katz, John Pencavel, an anonymous referee, and many seminar participants for helpful comments. The data and computer programs used in the preparation of this paper are available on request.

1. See OECD [1983] for a comparison of compulsory schooling laws in different countries.

The Quarterly Journal of Economics, November 1991

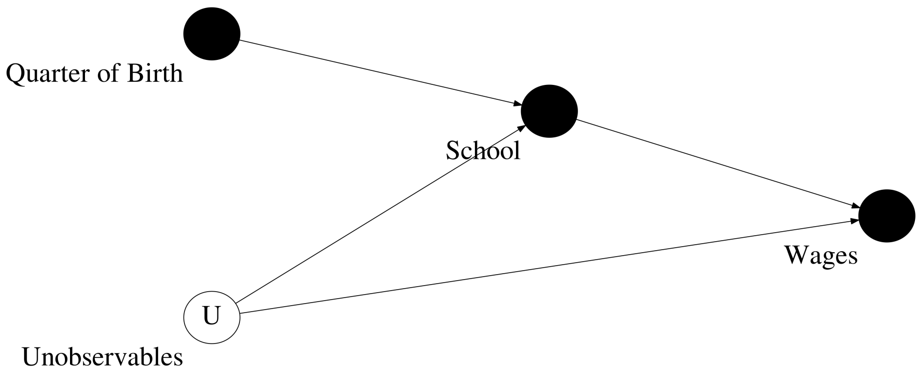


Table of contents

- Introduction
- Season of Birth, Compulsory Schooling, and Years of Education
  - Direct Evidence and teh Effect of Compulsory Schooling Laws
  - Why do Compulsory Schooling Laws Work?
- Estimating the Returns to Education
  - TSLS Estimation
  - Allowing the Seasonal Pattern in Education to Vary by State of Birth
  - Estimates for Black Men
- Other Possible Effects of Season of Birth
- Conclusions

Journal of Economic Literature
Vol. XXXVIII (December 2000) pp. 827–874

Natural "Natural Experiments" in Economics

MARK R. ROSENZWEIG *and* KENNETH I. WOLPIN[1]

1. *Introduction*

THE COSTLINESS OF and limitations on experiments involving human subjects have long been identified as major constraints on the progress of economic science. Indeed, it has been increasingly recognized that identification of many interesting parameters, such as the effects of schooling or work experience on earnings or of income on savings, requires attention to the fact that the variation in many of the variables whose effects are of interest may not be orthogonal to unobservable factors that jointly affect the outcomes studied. Such unmeasured or unmeasurable factors may include pre-existing or endowed skills ("ability"), preferences, or technologies that vary across individuals or firms in the economy. The possible existence of heterogeneity in these attributes means that almost all estimates are open to alternative interpretations in terms of self-selection by such traits. In determining the returns to schooling, for example, individuals cannot be considered to be randomly sorted among schooling levels. Thus, that more-schooled individuals have higher earnings may reflect the fact that more able individuals prefer schooling or face lower schooling costs. Similarly, that fertility and female labor supply are negatively correlated may reflect variation in preferences for children and work in the population.

Economists have used experiments that purposively randomize treatments to assess their effects in the presence of heterogeneity. Among the issues that some of the most prominent experiments have addressed are the impact of a negative income tax on labor supply, the effects of class size on test outcomes, and the effects of job training programs on earnings. However, these "man-made" experiments are subject to the criticisms that they lack generalizability and, most importantly, often do not adhere in implementation to the requirements of treatment randomness. The most widely applied approach to identifying causal or treatment effects, which has a long history in economics, employs instrumental variable techniques. This approach essentially assumes that some components of non-experimental data are random. That is, it is assumed that some variable or event satisfies the criterion of "randomness"—the event or variable is orthogonal to the unobservable and unmalleable factors that could affect the outcomes under study. This assumption, along with a set of additional assumptions

[1] University of Pennsylvania. We are grateful to two anonymous referees and the editor for helpful comments on an earlier draft of this paper. Partial support for the research was provided by NIH grants HD30907 and AG11725 and NSF grants SBR95–11955 and SBR93–08405.

827

We discuss Rosenzweig & Wolpin (2000) in more detail because it provides a small structural economic model of schooling choice that allows to interpret the instrumental variable estimates of Angrist (1990) and Angrist & Krueger (1991).

a	age
y_a	earnings at age a
S	level of schooling attainment
X_a	work experience at age a
μ	ability
a_e	school entry age
a_κ	minimum age to leave school
$S_0 = a_\kappa - a_e$	minimum schooling
c	direct cost of education

Wages are determined as follows:

$$\ln y_a = f(S, \mu) + g(X_a, \mu)$$

The authors assume that individuals work full-time after school and there is no uncertainty about wages. Individuals decide whether to pursue one additional year of schooling after the mandatory minimum. If they do so s_1 takes value one and zero otherwise. So, the final level of schooling is $S_1 = S_0 + s_1$. All individuals work A periods in the labor market. Spending one additional year in school does not reduce total time in the labor market. However, it results in entering the labor market one year later as schooling precludes working. Ability is the only source of heterogeneity and distributed at random in the population.

The individual's objective is to choose their final level of schooling such as to maximize their discounted lifetime earnings under the two scenarios (V_1, V_0).

$$\begin{aligned} V_1(S_1 = 1|S_0) &= -c + \textstyle\sum_{a=0}^{A-1} \beta^{a+1} y_a \\ &= -c + \textstyle\sum_{a=0}^{A-1} \beta^{a+1} \exp(f(S_0+1,\mu) + g(a,\mu)) \\ &= -c + \textstyle\sum_{a=0}^{A-1} \beta^{a+1} \exp(f(S_0+1,\mu)) \exp(g(a,\mu)) \\ &= -c + \exp(f(S_0+1,\mu)) \textstyle\sum_{a=0}^{A-1} \beta^{a+1} \exp(g(a,\mu)) \end{aligned}$$

$$\begin{aligned} V_1(S_1 = 0|S_0) = \sum_{a=0}^{A-1} \beta^a y_a \\ = \exp(f(S_0,\mu)) \sum_{a=0}^{A-1} \beta^a \exp(g(a,\mu)) \end{aligned}$$

We now turn attention to the decision rule $V_1 > V_0$ implies further pursuit of education.

$$\begin{aligned} &-c + \exp(f(S_0+1,\mu) \textstyle\sum_{a=0}^{A-1} \beta^{a+1} \exp(g(a,\mu)) \\ &> \exp(f(S_0+1,\mu)) \textstyle\sum_{a=0}^{A-1} \beta^{a} \exp(g(a,\mu) \end{aligned}$$

$$\begin{aligned} &-c + \exp(f(S_0+1,\mu) \textstyle\sum_{a=0}^{A-1} \beta^{a+1} \exp(g(a,\mu)) \\ &> \underbrace{\exp(f(S_0+1,\mu)) \sum_{a=0}^{A-1} \beta^{a} \exp(g(a,\mu)}_{V_1(S_1=0|S_0)} \end{aligned}$$

now divide by $V_1(S_1 = 0|S_0)$

$$\begin{aligned} \tfrac{\exp(f(S_0+1,\mu))}{\exp(f(S_0,\mu))}\beta &> 1 + \tfrac{c}{V_1(S_1=0|S_0)} \\ &> (1 + \tfrac{c}{V_1(S_1=0|S_0)})(1+r) \\ f(S_0+1,\mu) - f(S_0,\mu) &> r + \tfrac{c}{V_1(S_1=0|S_0)} \end{aligned}$$

using $\ln(1+x) \approx x$ for small x.

$$s_1 = \begin{cases} 1 & \text{if} \quad f(S_0+1,\mu) - f(S_0,\mu) \geq r + \ln\left(\frac{c}{V_1(s_1=0|S_0)} + 1\right) \\ 0 & \text{otherwise} \end{cases}$$

If ability increases the marginal schooling return, then there exists a unique cutoff value for ability μ^* such that individuals with ability above the cutoff continue schooling while those below do not.

$$\frac{\partial f(S_0+1,\mu) - f(S_0,\mu)}{\partial \mu} > 0$$

Even if randomly assigned, optimizing behavior induces an association between schooling and ability. This generates the ability bias. %

$$E[f(S_0+1,\mu) \mid \mu > \mu^*] - E[f(S_0,\mu) \mid \mu < \mu^*] > E[f(S_0+1,\mu)] - E[f(S_0,\mu)]$$

We now turn to the development of the Wald estimator Wald (1940). So, we first derive expected earnings equation for each age a.

$$E[\ln y_a] = \pi_1[f(S_0+1,\mu_1) + g(a-a_\kappa-1,\mu_1)] + (1-\pi_1)[f(S_0,\mu_2) + g(a-a_\kappa,\mu_2)]$$

We now consider the following scenario, where we reduce the school entry age by one year but keep the minimum school leaving age unchanged. Type 1 achieve their optimal level of schooling exactly at the school leaving age. Type 2's will be forced to attend school a year longer. %

$$E[\ln y_a] = \pi_1[f(S_0+1,\mu_1) + g(a-a_\kappa,\mu_1)] + (1-\pi_1)[f(S_0+1,\mu_2) + g(a-a_\kappa,\mu_2)]$$

The difference in expected (ln) earnings divided by the difference in expected schooling $0 \cdot \pi_1 + 1 \cdot (1-\pi_1)$, the Wald estimator, is thus

$$\begin{aligned}
&E[\ln y_a | \underbrace{Z=1}_{\text{reduced entry age}}] - E[\ln y_a | Z=0] \\
&= \pi_1(f(S_0+1,\mu_1) + g(a-a_\kappa,\mu_1)) \\
&\quad +(1-\pi_1)(f(S_0+1,\mu_2) + g(a-o_\kappa,\mu_2)) \\
&\quad -\pi_1(f(S_0+1,\mu_1) + g(a-a_\kappa-1,\mu_1)) \\
&\quad -(1-\pi_1)(f(S_0,\mu_2) + g(a-o_\kappa,\mu_2)) \\
\\
&= \pi_1(g(a-a_\kappa,\mu_1) - g(a-a_\kappa-1,\mu_1)) \\
&\quad +(1-\pi_1)(f(S_0+1,\mu_2) - f(S_0,\mu_2))
\end{aligned}$$

divide by difference in schooling attainment

$$\pi_1 * 0 + (1-\pi_1) * 1$$

$$\frac{\Delta E(\ln y_a)}{\Delta S} = \underbrace{\frac{\pi_1}{1-\pi_1}[g(a-a_\kappa,\mu_1) - g(a-a_\kappa-1,\mu_1)]}_{\text{type 1's additional experience}} + \underbrace{[f(S_0+1,\mu_2) - f(S_0,\mu_2)]}_{\text{effect of interest (compliers only)}},$$

where $\frac{\Delta E(\ln y_a)}{\Delta S}$ corresponds to $E(\ln y_a \mid Z=1) - E(\ln y_a \mid Z=0)$ and Z takes value one under the reduced school entry age and zero otherwise. Thus the estimate does not correspond directly to the effect of interest. However, Angrist & Krueger (1991) make the point in Figure V that for the cohort they are looking at $(a = 40, ..., 49)$ the effect of age on earnings is negligible.

Resources

- **Angrist, J. D. (1990)**. Lifetime earnings and the vietnam era draft lottery: Evidence from social security records. *American Economic Review*, 80(3), 313–336.
- **Angrist, J. D., & Krueger, A. B. (1991)**. Does compulsory school attendance affect schooling and earnings?. *The Quarterly Journal of Economics*, 106(4), 979-1014.
- **Angrist, J. D., Imbens, G. W., & Rubin, D.B. (1996)**. Identification of causal effects using instrumental variables. *Journal of the American Statistical Association*, 91(434), 444-455.
- **Angrist, J. D., & Imbens, G. W. (1999)**. Comment on James Heckman, "Instrumental Variables: A Study of Implicit Behavioral Assumption Used in Making Program Evaluations. *Journal of Human Resources*, 34, 823–827.
- **Deaton, A. S. (2009)**. Instruments of development: randomization in the tropics, and the search for the elusive keys to economic development. *National Bureau of Economic Research*, Working Paper 14690.
- **Heckman, J. J. (1997)**. Instrumental variables: A study of implicit behavioral assumptions used in making program evaluations. *The Journal of Human Resources*, 32(3), 441–462.
- **Heckman, J. J. (1999)**. Instrumental Variables: Response to Angrist and Imbens. *The Journal of Human Resources*, 34(4), 828-837.
- **Heckman, J. J., & Urzúa, S. (2010)**. Comparing IV with structural models: What simple IV can and cannot identify, *Journal of Econometrics*, 156(1), 27-37.
- **Imbens, G. W. (2010)**. Better LATE Than Nothing: Some Comments on Deaton (2009) and Heckman and Urzua (2009). *Journal of Economic Literature*, 48(2), 399-423.
- **Imbens, G. W. (2014)**. Instrumental variables: An econometrician's perspective. *Statistical Science*, 29(3), 323-358.
- **Rosenzweig, M. R., & Wolpin, K. I. (2000)**. Natural "natural" experiments in economics. *Journal of Economic Literature*, 38(4), 827–874.
- **Stock, J. H. & Trebbi, F. (2003)**. Retrospectives: Who Invented Instrumental Variable Regression?, *Journal of Economic Perspectives*, 17(3), 177-194.

1.10 Repeated observations

We now explore models in which we have multiple observations at different points in time. We start with the interrupted time series model and then explore difference-in-difference estimation using Card & Krueger (1994). We then return to the earlier example of school choice to benchmark the performance of alternative estimators as we vary the economics of individual decision-making.

1.10.1 Repeated observations and the estimation of causal effects

Overview

- Interrupted time series models
- Regression discontinuity design
- Panel data
  - Traditional adjustment strategies
  - Model-based approaches

Interrupted time series models (ITS)

$$Y_t = f(t) + D_t b + e_t$$

1. before the treatment is introduced (for $t \leq t^*$), $D_t = 0$ and $Y_t = Y_t^0$
2. after the treatment is in place (from t^* through T), $D_t = 1$ and $Y_t = Y_t^1$

The causal effect of the treatment is then $\delta_t = Y_t^1 - Y_t^0$ for time periods t^* through T. This is equal to $\delta_t = Y_t - Y_t^0$. The crucial assumption is that the obseved values of y_t before t^* can be used to speciy $f(t)$ for all time periods, including those after treatment.

Operation Ceasefire involved meetings with gang-involved youth who were engaged in gang conflict. Gang members were offered educational, employment, and other social services if they committed to refraining from gang-related deviance.

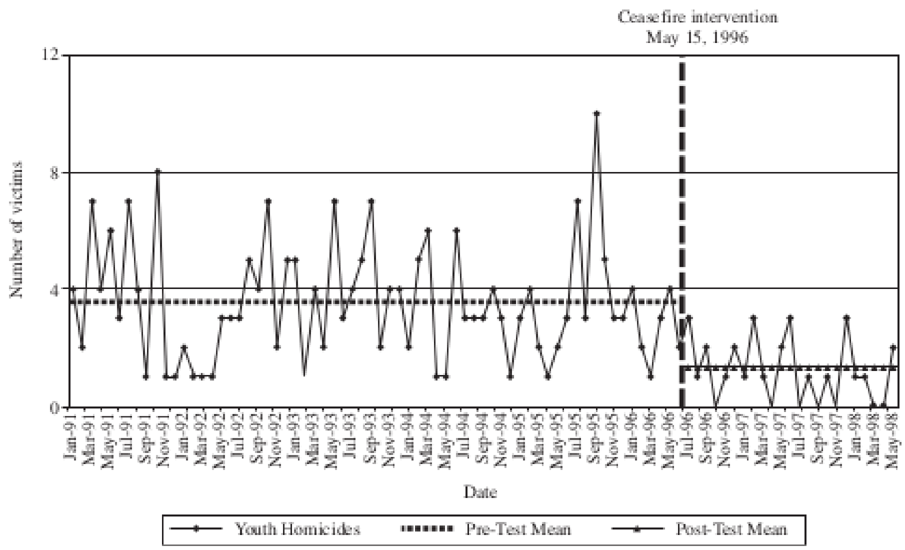


Figure 11.2 Monthly youth homicide rates in Boston, 1991–1999.
Source: Braga et al. (2001), figure 2.

Strategies to strengthen ITS analysis

- Assess the effect of the cause on multiple outcomes that should be be affected by the cause.
- Assess the effect of the cause on outcomes that should not be affected by the cause.
- Assess the effect of the cause withing subgroups across which the causal effect should vary in predictable ways.
- Adjust for trends in other variables that may affect or be related to the underlying time series of interest.
- Assess the impact of the termination of th cause in addition to its initiation.

Panel data

We now need to add a time dimension to our effect analysis, i.e. Y_t^d for $d = 0, 1$.

Seminal paper

- Card and Krueger (1995, 2000)

We briefly discuss the exposition from Angrist & Pischke (2008).

Table 5.2.1: Average employment per store before and after the New Jersey minimum wage increase

Variable		PA (i)	NJ (ii)	Difference, NJ-PA (iii)
1.	FTE employment before,	23.33	20.44	-2.89
	all available observations	(1.35)	(0.51)	(1.44)
2.	FTE employment after,	21.17	21.03	-0.14
	all available observations	(0.94)	(0.52)	(1.07)
3.	Change in mean FTE	-2.16	0.59	2.76
	employment	(1.25)	(0.54)	(1.36)

We are interested in

$$E[{Y_1}^1 - {Y_1}^0 | D = 1] = E[{Y_1}^1 | D = 1] - \underbrace{E[{Y_1}^0 | D = 1]}_{\text{counterfactual}}$$

assuming common trend

$$\begin{aligned} E[{Y_1}^0 - {Y_0}^0 | D = 1] &= E[{Y_1}^0 - {Y_0}^0 | D = 0] \\ E[{Y_1}^0 | D = 1] &= E[{Y_1}^0 - {Y_0}^0 | D = 0] + E[{Y_0}^0 | D = 1] \end{aligned}$$

$$E[{Y_1}^1 - {Y_1}^0 | D = 1] = E[{Y_1}^1 | D = 1] - E[{Y_1}^0 | D = 0] + E[{Y_0}^0 | D = 0] - E[{Y_0}^0 | D = 1]$$

moving to observed outcomes where T indicates period in conditioning set.

$$\begin{aligned} E[{Y_1}^1 - {Y_1}^0 | D = 1] = {} & E[Y | D = 1, T = 1] - E[Y | D = 1, T = 0] \\ & - (E[Y | D = 0, T = 1] - E[Y | D = 0, T = 0])() \end{aligned}$$

We can now map these observed objects to Table 5.2.

$$\begin{aligned} E[Y | D = 1, T = 1] = 21.03 \\ E[Y | D = 1, T = 0] = 20.44 \\ E[Y | D = 0, T = 1] = 21.17 \\ E[Y | D = 0, T = 0] = 23.33 \end{aligned}$$

Demonstration

We consider how alterantive estimators perform assuming a world where:

- no catholic elementary schools or middle schools exist
- all students consider entering either public or Cathlic high schools after end of eight grade
- pretretment achievement test score is available for the eights grade

$$\text{difference-in-difference} \qquad Y_{i10} - Y_{i8} = a + D_i^* c + e_i$$

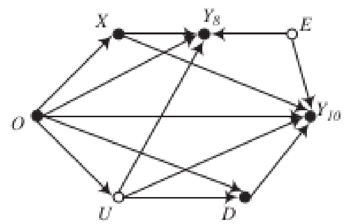


Figure 11.5 A directed graph for the effect of Catholic schooling on tenth grade achievement when a measure of eighth grade achievement is also available.

Control outcomes

$$
\begin{aligned}
Y_{i8}^0 &= 98 + O_i + U_i + X_i + E_i + \nu_{i8}^0 \\
Y_{i9}^0 &= 99 + O_i + U_i + X_i + E_i + \nu_{i9}^0 \\
Y_{i10}^0 &= 100 + O_i + U_i + X_i + E_i + \nu_{i10}^0
\end{aligned}
$$

There is a linear time trend for Y_{it}^0 but we will also consider a diverging trend scenario.

Treated outcomes

$$
\begin{aligned}
Y_{i9}^1 &= Y_{i9}^0 + \delta_i' + \delta_i'' \\
Y_{i10}^1 &= Y_{i10}^0 + (1 + \delta_i') + \delta_i''
\end{aligned}
$$

The treatment effect increases in time.

Treatment selection

$$
\begin{aligned}
\text{baseline} \qquad & Pr[D_i^* = 1 \mid O_i, U_i] = \frac{exp(-3.8 + O_i + U_i)}{1 + exp(-3.8 + O_i + U_i)} \\
\text{self-selection on gains} \qquad & Pr[D_i^* = 1 \mid O_i, U_i] = \frac{exp(-7.3 + O_i + U_i + 5\delta'')}{1 + exp(-7.3 + O_i + U_i + 5\delta'')} \\
\text{self-selection on pretest} \qquad & Pr[D_i^* = 1 \mid O_i, U_i] = \frac{exp(-7.3 + O_i + U_i + k(Y_{i8} - E[Y_{i8}]))}{1 + exp(-7.3 + O_i + U_i + k(Y_{i8} - E[Y_{i8}]))}
\end{aligned}
$$

Why is the average control outcome higher among the (eventually) treated?

[8]:
```
num_agents, selection, trajectory = 10, "baseline", "parallel"
df = get_sample_panel_demonstration(num_agents, selection, trajectory)
df.groupby(["D_ever", "Grade"])["Y"].mean()
```

[8]:
```
D_ever  Grade
0       8              NaN
        9        97.858309
        10       98.398170
Name: Y, dtype: float64
```

How do our standard estimators perform in these setting?

[10]:
```
for selection in [
    "baseline",
    "self-selection on gains",
```

(continues on next page)

(continued from previous page)

```
    "self-selection on pretest",
]:
    for trajectory in ["parallel", "divergent"]:
        print("\n Selection: {:}, Trajectory: {:}".format(selection, trajectory))
        num_agents, selection, trajectory = 1000, selection, trajectory
        df = get_sample_panel_demonstration(num_agents, selection, trajectory)
        for estimator in ["naive", "diff"]:
            rslt = get_panel_estimates(estimator, df)
            print("{:10}: {:5.3f}".format(estimator, rslt.params["D"]))
```

```
 Selection: baseline, Trajectory: parallel
naive     : 15.278
diff      : 9.416

 Selection: baseline, Trajectory: divergent
naive     : 15.363
diff      : 9.774

 Selection: self-selection on gains, Trajectory: parallel
naive     : 14.151
diff      : 11.358

 Selection: self-selection on gains, Trajectory: divergent
naive     : 15.986
diff      : 12.460

 Selection: self-selection on pretest, Trajectory: parallel
naive     : 14.082
diff      : 8.971

 Selection: self-selection on pretest, Trajectory: divergent
naive     : 16.011
diff      : 10.543
```

Table 11.1 Change Score and Analysis of Covariance Estimates of the Catholic School Effect in the Tenth Grade

Setup conditions:				
Self-selection on the causal effect	No	Yes	No	No
Positive self-selection on the pretest	No	No	Yes	No
Negative self-selection on the pretest	No	No	No	Yes
	Parallel Trajectories			
True average treatment effects:				
ATE	10.00	10.00	10.00	10.00
ATT	10.00	11.51	10.00	10.00
ATC	10.00	9.83	10.00	10.00
Estimated coefficients for D^*:				
Naive estimator:				
Regression of Y_{10} on D^*	14.75	13.86	15.92	13.25
Change score estimator:				
Regression of $(Y_{10} - Y_8)$ on D^*	10.00	11.51	7.96	12.26
Analysis of covariance estimator:				
Regression of Y_{10} on D^*, Y_8, O, and X	10.51	11.75	10.49	10.52
	Divergent Trajectories			
True average treatment effects:				
ATE	10.00	10.00	10.00	10.00
ATT	10.00	11.51	10.00	10.00
ATC	10.00	9.83	10.00	10.00
Estimated coefficients for D^*:				
Naive estimator:				
Regression of Y_{10} on D^*	15.75	14.86	16.88	14.30
Change score estimator:				
Regression of $(Y_{10} - Y_8)$ on D^*	11.00	12.51	8.92	13.31
Analysis of covariance estimator:				
Regression of Y_{10} on D^*, Y_8, O, and X	11.52	12.75	11.50	11.53

Resources

- **Angrist, J. D. and Pischke, J.-S. (2008)**. Mostly harmless econometrics: An empiricist's companion. Princeton, NJ: *Princeton University Press*.
- **Bertrand, M., Duflo E., and Mullainathan, S. (2004)**. How much should we trust differences-in-differences estimates?. *The Quarterly Journal of Economics*, 119(1), 249-275.
- **Braga, A. A., Kennedy, D. M., Waring, E. J., and Piehl, A. M. (2001)**. Problem-oriented policing, deterrence, and youth violence: An evaluation of boston's operation ceasefire. *Journal of research in crime and delinquency*, 38(3), 195–225.
- **Card, D., and Krueger, A. B. (1995)**. Time-series minimum-wage studies: A meta-analysis. *The American Economic Review*, 85(2), 238–243.

- **Card, D. and Krueger, A. B. (2000)**. Minimum wages and employment: A case study of the fast-food industry in new jersey and pennsylvania. *American Economic Review*, 90(5), 1397–1420.
- **Frölich, M. and Sperlich, S. (2019)**. Impact evaluation. Cambridge, England: *Cambridge University Press*.
- **Lechner, M. (2010)**. The estimation of causal effects by difference-in-difference methods, 4(3), 165–224.

1.11 Regression discontinuity design

We study regression discontinuity design in more detail. We discuss identification, issues in interpretation, and challenges to its application based on the seminal review by Lee & Lemieux (2010). We look at different conditional mean functions and the issue of bandwidth choice. We reproduce and check the robustness of some of the results in Lee (2008).

1.11.1 Regression discontinuity design

This following material is mostly based on the following review:

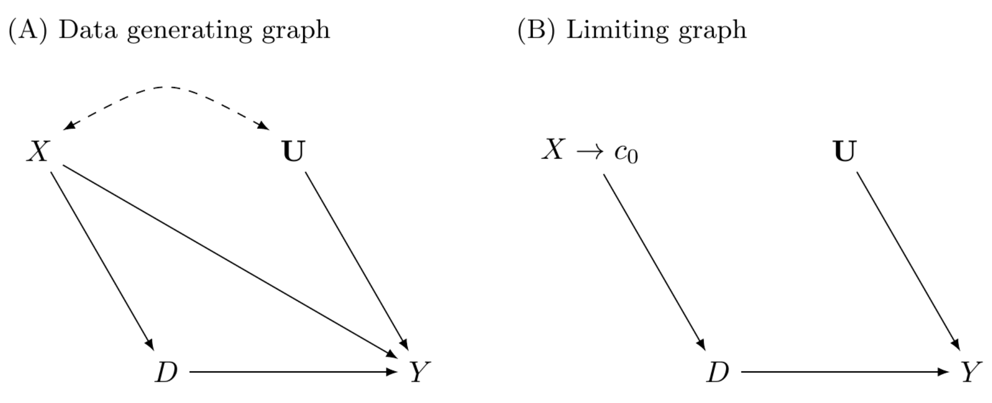
- Lee, D. S., and Lemieux, T. (2010). Regression discontinuity designs in economics. *Journal of Economic Literature, 48*(2), 281–355.

The idea of the authors is to throughout contrast RDD to its alternatives. They initially just mention selected features throughout the introduction but then also devote a whole section to it. This clearly is a core strength of the article. I hope to maintain this focus in my lecture. Also, their main selling point for RDD as the close cousin to standard randomized controlled trial is that the behavioral assumption of imprecise control about the assignment variable translates into the statistical assumptions of a randomized experiment.

Original application

In the initial application of RD designs, Thistlethwaite & Campell (1960) analyzed the impact of merit rewards on future academic outcomes. The awards were allocated based on the observed test score. The main idea behind the research design was that individuals with scores just below the cutoff (who did not get the award) were good comparisons to those just above the cutoff (who did receive the award).

Causal graph



Intuition

Key points of RD design

- RD designs can be invalid if individuals can precisely manipulate the assignment variable - discontinuity rules might generate incentives
- If individuals - even while having some influence - are unable to precisely manipulate the assignment variable, a consequence of this is that the variation in treatment near the threshold is randomized as though from a randomized experiment - contrast to IV assumption
- RD designs can be analyzed - and tested - like randomized experiments.
- Graphical representation of an RD design is helpful and informative, but the visual presentation should not be tilted toward either finding an effect or finding no effect.
- Nonparametric estimation does not represent a "solution" to functional form issues raised by RD designs. It is therefore helpful to view it as a complement to - rather than a substitute for - parametric estimation.
- Goodness-of-fit and other statistical tests can help rule out overly restrictive specifications.

Baseline

A simple way to estimating the treatment effect τ is to run the following linear regression.

$$Y = \alpha + D\tau + X\beta + \epsilon,$$

where $D \in [0, 1]$ and we have $D = 1$ if $X \geq c$ and $D = 0$ otherwise.

Baseline setup

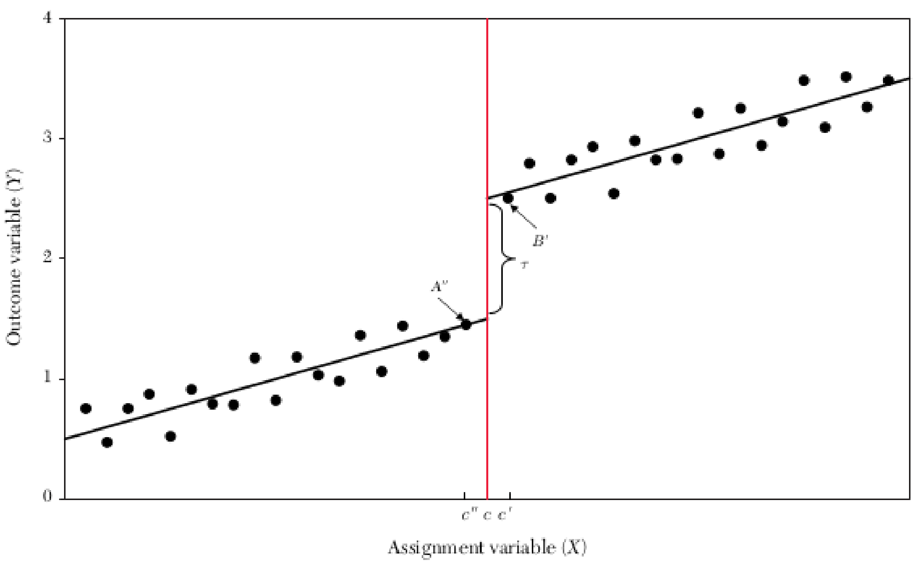


Figure 1. Simple Linear RD Setup

- "all other factors" determining Y must be evolving "smoothly" (continously) with respect to X.
- the estimate will depend on the functional form

Potential outcome framework

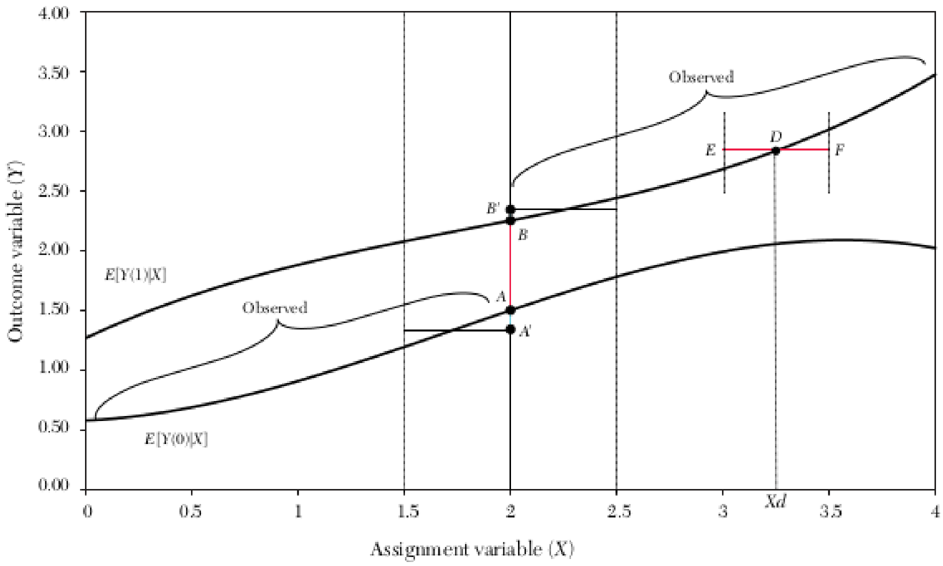


Figure 2. Nonlinear RD

Potential outcome framework

Suppose $D = 1$ if $X \geq c$, and $D = 0$ otherwise

$$\Rightarrow \begin{cases} E(Y \mid X = c) = E(Y_0 \mid X = c) & \text{for} \quad X < c \\ E(Y \mid X = c) = E(Y_1 \mid X = c) & \text{for} \quad X \geq c \end{cases}$$

Suppose $E(Y_1 \mid X = c), E(Y_0 \mid X = c)$ are continuous in x.

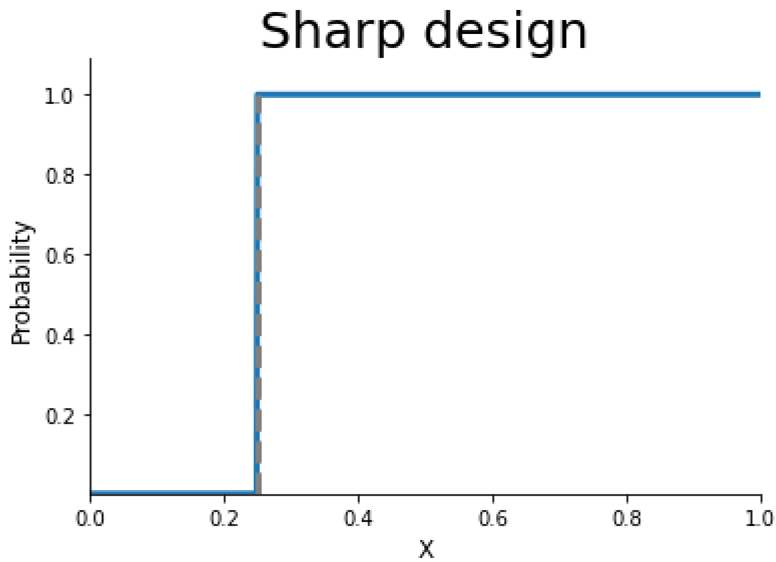
$$\Rightarrow \begin{cases} \lim_{\epsilon \searrow 0} E(Y_0 \mid X = c - \epsilon) = E(Y_0 \mid X = c) \\ \lim_{\epsilon \searrow 0} E(Y_1 \mid X = c + \epsilon) = E(Y_1 \mid X = c) \end{cases}$$

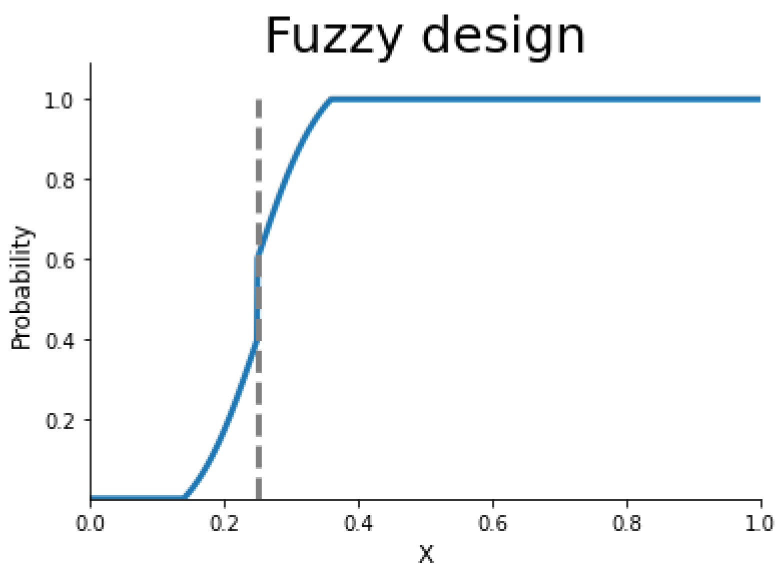
$$\begin{aligned} &\lim_{\epsilon \searrow 0} E(Y \mid X = c + \epsilon) - \lim_{\epsilon \searrow 0} E(Y \mid X = c - \epsilon) \\ &\quad = \lim_{\epsilon \searrow 0} E(Y_1 \mid X = c + \epsilon) - \lim_{\epsilon \searrow 0} E(Y_0 \mid X = c - \epsilon) \\ &\quad = E(Y_1 \mid X = c) - E(Y_0 \mid X = c) \\ &\quad = E(Y_1 - Y_0 \mid X = c) \end{aligned}$$

$\Rightarrow$ average treatment effect at the cutoff

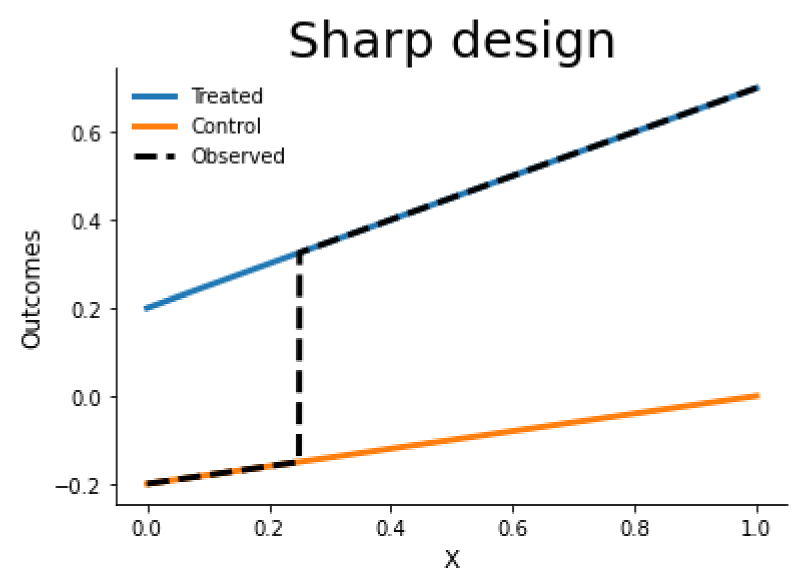
Sharp and Fuzzy design

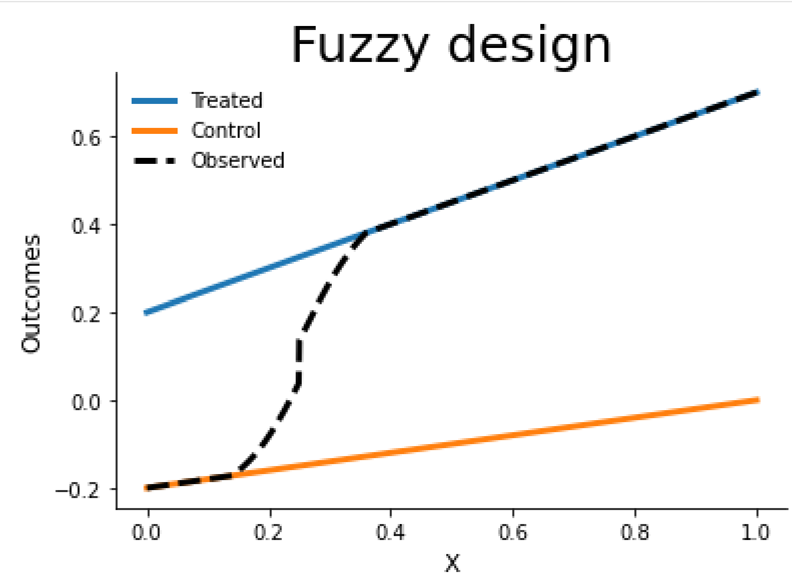
[4]:
```
grid = np.linspace(0, 1.0, num=1000)
for version in ["sharp", "fuzzy"]:
    probs = get_treatment_probability(version, grid)
    get_plot_probability(version, grid, probs)
```





[5]:
```
for version in ["sharp", "fuzzy"]:
    plot_outcomes(version, grid)
```





Alternatives

Consider the standard assumptions for matching:

- ignorability - trivially satisfied by research design as there is no variation left in D conditional on X
- common support - cannot be satisfied and replaced by continuity

Lee and Lemieux (2010) emphasize the close connection of RDD to randomized experiments. - How does the graph in the potential outcome framework change?

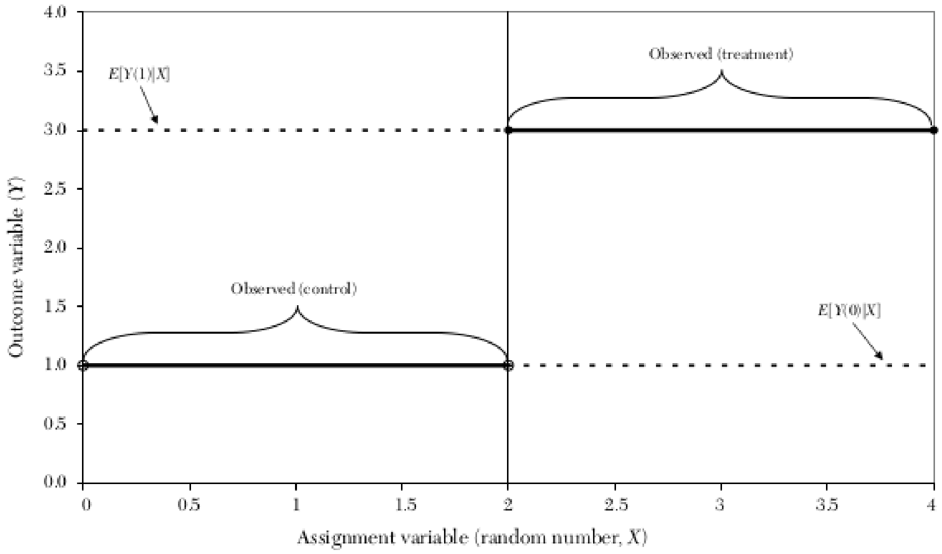


Figure 3. Randomized Experiment as a RD Design

Continuity, the key assumption of RDD, is a consequence of the research design (e.g. randomization) and not simply imposed.

Identification

Ad-hoc × vs. thoughtful answers ✓. Both are true, but only thoughtful consideration clarifies the strength of the regression discontinuity design as opposed to, for example, an instrumental variables approach.

Question

How do I know whether an RD design is appropriate for my context? When are the identification assumptions plausable or implausable?

Answers

× An RD design will be appropriate if it is plausible that all other unobservable factors are "continuously" related to the assignment variable.

✓ When there is a continuously distributed stochastic error component to the assignment variable - which can occur when optimizing agents do not have *precise* control over the assignment variable - then the variation in the treatment will be as good as randomized in a neighborhood around the discontinuity threshold.

Question

Is there any way I can test those assumptions?

Answers

× No, the continuity assumption is necessary so there are no tests for the validity of the design.

✓ Yes. As in randomized experiment, the distribution of observed baseline covariates should not change discontinuously around the threshold.

Simplified setup

$$
\begin{aligned}
Y &= D\tau + W\delta_1 + U \\
D &= I[X \geq c] \\
X &= W\delta_2 + V
\end{aligned}
$$

- W is the vector of all predetermined and observable characteristics.

What are the source of heterogeneity in the outcome and assignment variable?

The setup for an RD design is more flexible than other estimation strategies. - We allow for W to be endogenously determined as long as it is determined prior to V. This ensures some random variation around the threshold. - We take no stance as to whether some elements δ_1 and δ_2 are zero (exclusion restrictions) - We make no assumptions about the correlations between W, U, and V.

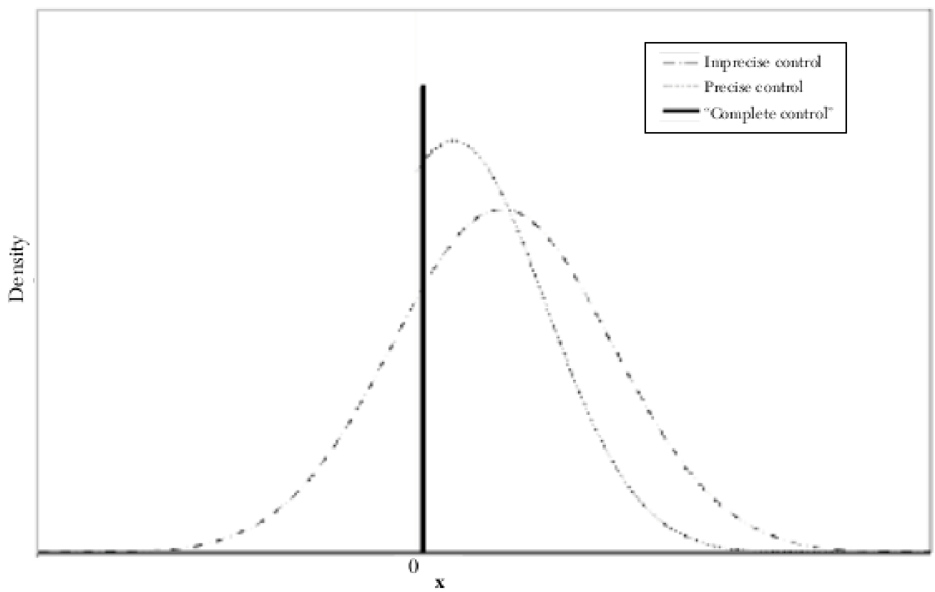


Figure 4. Density of Assignment Variable Conditional on $W = w$, $U = u$

Local randomization

We say individuals have imprecise control over X when conditional on $W = w$ and $U = u$ the density of V (and hence X) is continuous.

Applying Baye's rule

$$
\begin{aligned}
&\Pr[W = w, U = u \mid X = x] \\
&= f(x \mid W = w, U = u) \quad \frac{\Pr[W = w, U = u]}{f(x)}
\end{aligned}
$$

Local randomization: If individuals have imprecise control over X as defined above, then $\Pr[W = w, U = u \mid X = x]$ is continuous in x: the treatment is "as good as" randomly assigned around the cutoff.

⇒ the behavioral assumption of imprecise control of X around the threshold has the prediction that treatment is locally randmized.

Consequences

- testing prediction that $\Pr[W = w, U = u \mid X = x]$ is continuous in X by at least looking at $\Pr[W = w \mid X = x]$
- irrelevance of including baseline covariates

Interpretation

Questions

To what extent are results from RD designs generalizable?

Answers

× The RD estimate of the treatment effect is only applicable to the subpopulation of individuals at the discontinuity threshold and uninformative about the effect everywhere else.

✓ The RD estimand can be interpreted as a weighted average treatment effect, where the weights are relative ex ante probability that the value of an individual's assignment variable will be in the neighborhood of the threshold.

Alternative evaluation strategies

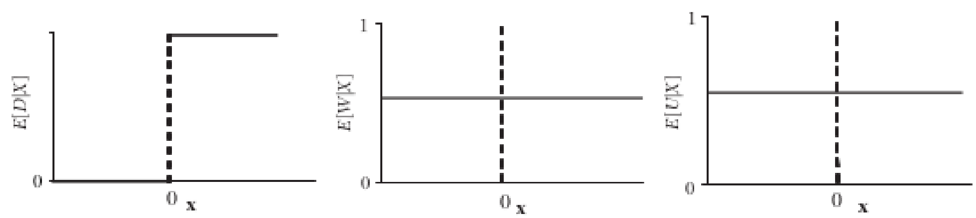
- randomized experiment
- regression discontinuity design
- matching on observables
- instrumental variables

How do the (assumed) relationships between treatment, observables, and unobservable differ across research designs?

Endogenous dummy variable

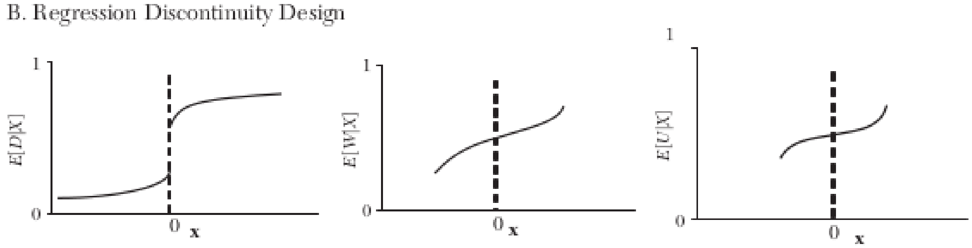
$$
\begin{aligned}
Y &= D\tau + W\delta_1 + U \\
D &= I[X \geq c] \\
X &= W\delta_2 + V
\end{aligned}
$$

A. Randomized Experiment



- By construction X is not related to any other observable or unoservable characteristic.

B. Regression Discontinuity Design

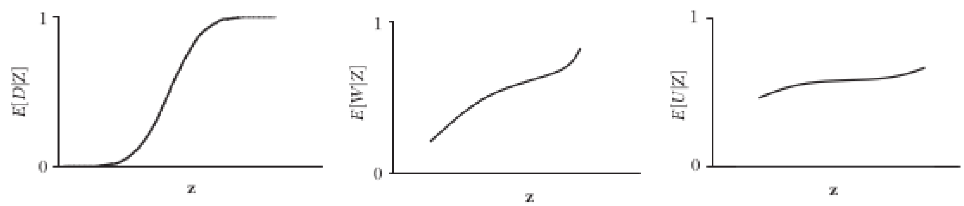


- W and D might be systematically related to X

C. Matching on Observables

- The crucial assumptions is that the two lines in the left graph are actually superimposed of each other.
- The plot in the middle is missing as all variables are used for estimation are not available to test the validity of identifying assumptions.

D. Instrumental Variables



- The instrument must affect treatment probablity.
- A proper instructment requires the line in the right graph to be flat.

Nonlinear expectation

A nonlinear conditional expectation can easily lead to misleading result if the estimated model is based on an local linear regression. The example below, including the simulation code, is adopted from Cunningham (2021). This example is set up closely aligned with the potential outcome framework.

[6]:
```
df = pd.DataFrame(columns=["Y", "Y1", "Y0", "X", "X2"], dtype=float)

# We simulate a running variable, truncate it at
# zero and restrict it below 240.
df["X"] = np.random.normal(100, 50, 1000)
df.loc[df["X"] < 0, "X"] = 0
df = df[df["X"] < 280]

df["X2"] = df["X"] ** 2

df["D"] = 0
df.loc[df["X"] > 140, "D"] = 1
```

We now simulate the potential outcomes and record the observed outcome. Note that there is no effect of treatment.

[7]:
```
def get_outcomes(x, d):

    level = 10000 - 100 * x + x ** 2
    eps = np.random.normal(0, 1000, 2)
```

(continues on next page)

(continued from previous page)

```
    y1, y0 = level + eps
    y = d * y1 + (1 - d) * y0

    return y, y1, y0


for idx, row in df.iterrows():
    df.loc[idx, ["Y", "Y1", "Y0"]] = get_outcomes(row["X"], row["D"])

df = df.astype(float)
```

What about the difference in average outcomes by treatment status. Where does the difference come from?

[8]:
```
df.groupby("D")["Y"].mean()
```

[8]:
```
D
0.0     9836.848389
1.0    21643.244614
Name: Y, dtype: float64
```

Now we are ready for a proper RDD setup.

[9]:
```
for ext_ in ["X", "X + X2 "]:
    rslt = smf.ols(formula=f"Y ~ D + {ext_}", data=df).fit()
    print(rslt.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      Y   R-squared:                       0.783
Model:                            OLS   Adj. R-squared:                  0.782
Method:                 Least Squares   F-statistic:                     1795.
Date:                Wed, 07 Jul 2021   Prob (F-statistic):               0.00
Time:                        08:59:13   Log-Likelihood:                -9407.4
No. Observations:                1000   AIC:                         1.882e+04
Df Residuals:                     997   BIC:                         1.884e+04
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   4377.3226    241.423     18.131      0.000    3903.568    4851.077
D           5889.2320    319.611     18.426      0.000    5262.044    6516.420
X             68.1328      2.699     25.247      0.000      62.837      73.429
==============================================================================
Omnibus:                      799.728   Durbin-Watson:                   2.046
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            27622.480
Skew:                           3.372   Prob(JB):                         0.00
Kurtosis:                      27.849   Cond. No.                         430.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly␣
↪specified.
```

(continues on next page)

(continued from previous page)

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      Y   R-squared:                       0.973
Model:                            OLS   Adj. R-squared:                  0.973
Method:                 Least Squares   F-statistic:                 1.215e+04
Date:                Wed, 07 Jul 2021   Prob (F-statistic):               0.00
Time:                        08:59:13   Log-Likelihood:                -8357.0
No. Observations:                1000   AIC:                         1.672e+04
Df Residuals:                     996   BIC:                         1.674e+04
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   1.005e+04    107.870     93.125      0.000    9833.768    1.03e+04
D              2.2077    131.768      0.017      0.987    -256.368     260.783
X            -99.9678      2.202    -45.405      0.000    -104.288     -95.647
X2             0.9959      0.012     84.522      0.000       0.973       1.019
==============================================================================
Omnibus:                        0.261   Durbin-Watson:                   2.002
Prob(Omnibus):                  0.878   Jarque-Bera (JB):                0.164
Skew:                          -0.004   Prob(JB):                        0.921
Kurtosis:                       3.062   Cond. No.                     6.81e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly␣
↪specified.
[2] The condition number is large, 6.81e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In a nutsheel, the misspecification of the model for the conditional mean functions results in flawed inference.

Estimation

Lee (2008)

The author studies the "incumbency advantage", i.e. the overall causal impact of being the current incumbent party in a district on the votes obtained in the district's election.

- Lee, David S. (2008). Randomized experiments from non-random selection in U.S. House elections. Journal of Econometrics.

[13]:
```
df_base = pd.read_csv("../../datasets/processed/msc/house.csv")
df_base.head()
```

[13]:
```
   vote_last  vote_next
0     0.1049     0.5810
1     0.1393     0.4611
2    -0.0736     0.5434
3     0.0868     0.5846
4     0.3994     0.5803
```

Let's put in some effort to ease the flow of our coming analysis.

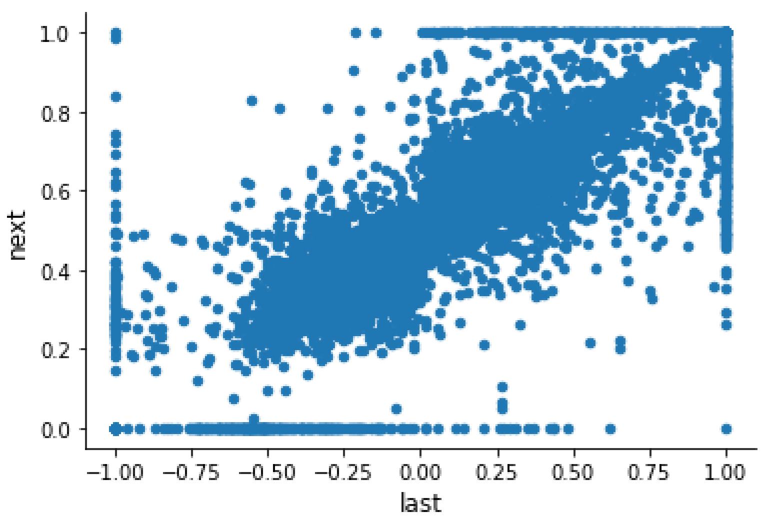
[9]:
```
df_base.rename(columns={"vote_last": "last", "vote_next": "next"}, inplace=True)

df_base["incumbent_last"] = np.where(df_base["last"] > 0.0, "democratic", "republican")
df_base["incumbent_next"] = np.where(df_base["next"] > 0.5, "democratic", "republican")

df_base["D"] = df_base["last"] > 0

for level in range(2, 5):
    label = "last_{:}".format(level)
    df_base.loc[:, label] = df_base["last"] ** level
```

The column `vote_last` refers to the Democrat's winning margin and is thus bounded between -1 and 1. So a positive number indicates a Democrat as the incumbent.

What are the basic characteristics of the dataset?

[10]:
```
df_base.plot.scatter(x="last", y="next")
```

[10]:
```
<AxesSubplot:xlabel='last', ylabel='next'>
```



What is going on at the boundary? What is the re-election rate?

[11]:
```
info = pd.crosstab(df_base["incumbent_last"], df_base["incumbent_next"], normalize=True)
stat = info.to_numpy().diagonal().sum() * 100
print(f"Re-election rate: {stat:5.2f}%")
```

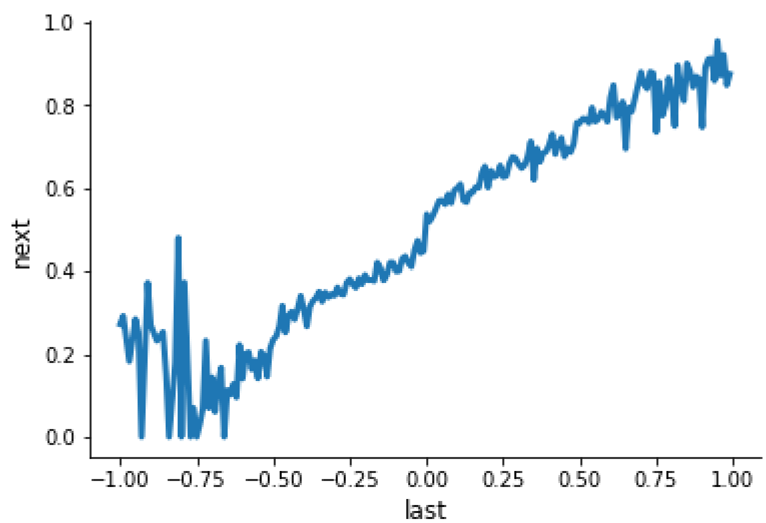
```
Re-election rate: 90.93%
```

Regression discontinuity design

How does the average vote in the next election look like as we move along last year's election.

```
[12]: df_base["bin"] = pd.cut(df_base["last"], 200, labels=False) / 100 - 1
      df_base.groupby("bin")["next"].mean().plot(xlabel="last", ylabel="next")
```

```
[12]: <AxesSubplot:xlabel='last', ylabel='next'>
```



We can now compute the difference at the cutoffs to get an estimate for the treatment effect.

```
[13]: h = 0.05
      df_subset = df_base[df_base["last"].between(-h, h)]
      stat = np.abs(df_subset.groupby("incumbent_last")["next"].mean().diff()[1])
      print(f"Treatment Effect: {stat:5.3f}")
```

```
Treatment Effect: 0.096
```

How does the effect depend on the size subset under consideration?

Regression approach

Now we turn to an explicit model of the conditional mean. We first set up explicit models on both sides of the cutoff and then aggreagte the model into single regression estimations.

```
[14]: def fit_regression(incumbent, df, level=4):

          df_incumbent = df[df["incumbent_last"] == incumbent].copy()

          formula = "next ~ last"
          for level in range(2, level + 1):
              label = "last_{:}".format(level)
              formula += f" + {label}"

          rslt = smf.ols(formula=formula, data=df_incumbent).fit()
          return rslt
```

(continues on next page)

(continued from previous page)

```
rslt = dict()
for incumbent in ["republican", "democratic"]:
    rslt = fit_regression(incumbent, df_base, level=3)
    title = "\n\n {:}\n".format(incumbent.capitalize())
    print(title, rslt.summary())
```

```
 Republican
                            OLS Regression Results
==============================================================================
Dep. Variable:                   next   R-squared:                       0.271
Model:                            OLS   Adj. R-squared:                  0.270
Method:                 Least Squares   F-statistic:                     339.2
Date:                Wed, 30 Jun 2021   Prob (F-statistic):          3.05e-187
Time:                        12:05:34   Log-Likelihood:                 1749.4
No. Observations:                2740   AIC:                            -3491.
Df Residuals:                    2736   BIC:                            -3467.
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.4278      0.007     57.880      0.000       0.413       0.442
last          -0.0971      0.077     -1.264      0.206      -0.248       0.054
last_2        -1.7177      0.205     -8.359      0.000      -2.121      -1.315
last_3        -1.4636      0.142    -10.338      0.000      -1.741      -1.186
==============================================================================
Omnibus:                      203.681   Durbin-Watson:                   1.866
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1087.416
Skew:                          -0.022   Prob(JB):                    7.42e-237
Kurtosis:                       6.086   Cond. No.                         113.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly␣
↪specified.


 Democratic
                            OLS Regression Results
==============================================================================
Dep. Variable:                   next   R-squared:                       0.379
Model:                            OLS   Adj. R-squared:                  0.379
Method:                 Least Squares   F-statistic:                     776.5
Date:                Wed, 30 Jun 2021   Prob (F-statistic):               0.00
Time:                        12:05:34   Log-Likelihood:                 2055.2
No. Observations:                3818   AIC:                            -4102.
Df Residuals:                    3814   BIC:                            -4077.
Df Model:                           3
```

(continues on next page)

(continued from previous page)

```
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.5393      0.007     71.995      0.000       0.525       0.554
last           0.3553      0.071      4.998      0.000       0.216       0.495
last_2         0.1932      0.174      1.107      0.268      -0.149       0.535
last_3        -0.2111      0.114     -1.856      0.064      -0.434       0.012
==============================================================================
Omnibus:                      439.976   Durbin-Watson:                   2.136
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1993.314
Skew:                          -0.477   Prob(JB):                         0.00
Kurtosis:                       6.409   Cond. No.                         114.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly␣
↪specified.
```

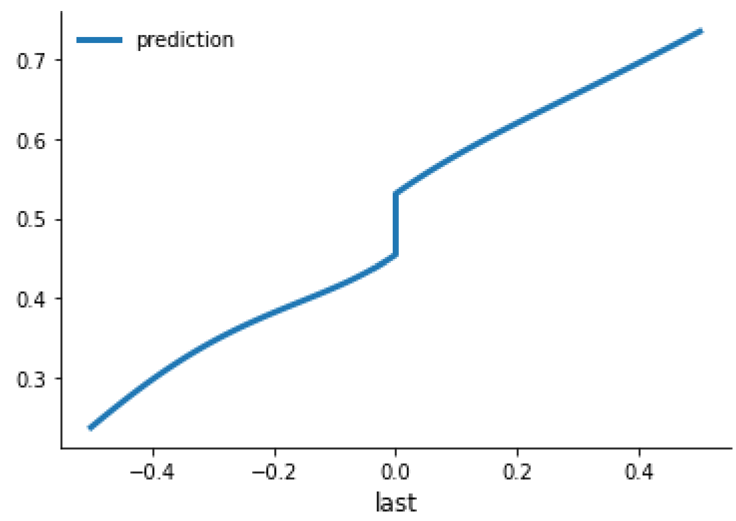
How does the predictions look like?

[15]:
```
dfs = list()

for incumbent in ["republican", "democratic"]:
    rslt = fit_regression(incumbent, df_base, level=4)

    # For our predictions, we need to set up a grid for the evaluation.
    if incumbent == "republican":
        grid = np.linspace(-0.5, 0.0, 51)
    else:
        grid = np.linspace(+0.0, 0.5, 51)

    df_grid = pd.DataFrame(grid, columns=["last"])

    for level in range(2, 5):
        label = "last_{:}".format(level)
        df_grid.loc[:, label] = df_grid["last"] ** level

    tmp = pd.DataFrame(rslt.predict(df_grid), columns=["prediction"])
    tmp.index = df_grid["last"]
    dfs.append(tmp)

rslts = pd.concat(dfs)
```

Let’s have a look at the estimated conditional mean fuctions.

[16]:
```
rslts.plot()
```

[16]:
```
<AxesSubplot:xlabel='last'>
```



Regression

There are several alternatives to estimate the conditional mean functions.

- pooled regressions
- local linear regressions

Pooled regression

We estimate the conditinal mean using the whole function.

$$Y = \alpha + \tau D + \beta X + \epsilon$$

This allows for a difference in levels but not slope.

[17]:
```
smf.ols(formula="next ~ last + D", data=df_base).fit().summary()
```

[17]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:                   next   R-squared:                       0.670
Model:                            OLS   Adj. R-squared:                  0.670
Method:                 Least Squares   F-statistic:                     6658.
Date:                Wed, 30 Jun 2021   Prob (F-statistic):               0.00
Time:                        12:05:34   Log-Likelihood:                 3661.9
No. Observations:                6558   AIC:                            -7318.
Df Residuals:                    6555   BIC:                            -7298.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.4427      0.003    139.745      0.000       0.437       0.449
```

(continues on next page)

(continued from previous page)

```
D[T.True]       0.1137      0.006      20.572      0.000       0.103       0.125
last            0.3305      0.006      55.186      0.000       0.319       0.342
==============================================================================
Omnibus:                      595.910   Durbin-Watson:                   2.143
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             3444.243
Skew:                          -0.225   Prob(JB):                         0.00
Kurtosis:                       6.522   Cond. No.                         5.69
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly␣
↪specified.
"""
```

Local linear regression

We now turn to local regressions by restricting the estimation to observations close to the cutoff.

$$Y = \alpha + \tau D + \beta X + \gamma XD + \epsilon,$$

where $-h \geq X \geq h$. This allows for a difference in levels and slope.

[18]:
```python
for h in [0.3, 0.2, 0.1, 0.05, 0.01]:
    # We restrict the sample to observations close
    # to the cutoff.
    df = df_base[df_base["last"].between(-h, h)]

    formula = "next ~ D + last + D * last"
    rslt = smf.ols(formula=formula, data=df).fit()
    info = [h, rslt.params[1] * 100, rslt.pvalues[1]]
    print(" Bandwidth: {:>4}   Effect {:5.3f}%    pvalue {:5.3f}".format(*info))
```

```
 Bandwidth:  0.3   Effect 8.318%    pvalue 0.000
 Bandwidth:  0.2   Effect 7.818%    pvalue 0.000
 Bandwidth:  0.1   Effect 6.058%    pvalue 0.000
 Bandwidth: 0.05   Effect 4.870%    pvalue 0.010
 Bandwidth: 0.01   Effect 9.585%    pvalue 0.001
```

There exists some work that can guide the choice of the bandwidth. Now, let's summarize the key issues and some review best practices.

Checklist

Recommendations: - To assess the possibility of manipulations of the assignment variable, show its distribution. - Present the main RD graph using binned local averages. - Graph a benchmark polynomial specification - Explore the sensitivity of the results to a range of bandwidth, and a range of orders to the polynomial. - Conduct a parallel RD analysis on the baseline covariates. - Explore the sensitivity of the results to the inclusion of baseline covariates.

References

- **Cattaneo, M.D., Idrobo, N., & Titiunik, R. (2019)**. A practical Introduction to Regression Discontinuity Designs: Foundations, Cambridge University Press.
- **Cunningham, S. (2021)**. Causal Inference: The Mixtape. *Yale University Press*
- **Hahn, J., Todd, P. E., and van der Klaauw, W. (2001)**. Identification and estimation of treatment effects with a regression-discontinuity design. *Econometrica*, 69(1), 201–209.
- **Imbens, G., & Lemieux, G. (2007)**. Regression discontinuity designs: A guide to practice. *Journal of Econometrics*, 142 (2) :615-635.
- **Lee, D. S. (2008)**. Randomized experiments from nonrandom selection in US House elections. *Journal of Econometrics*, 142(2), 675–697.
- **Lee, D. S., and Lemieux, T. (2010)**. Regression discontinuity designs in economics. *Journal of Economic Literature*, 48(2), 281–355.
- **Thistlethwaite, D. L., and Campbell, D. T. (1960)**. Regression-discontinuity analysis: An alternative to the ex-post facto experiment. *Journal of Educational Psychology*, 51(6), 309–317.

1.12 Difference in difference

A lecture on difference-in-difference method will be part of the next iteration of the OSE data science course, summer semester 2022. Details on this lecture will be realized soon.

1.12.1 Difference in Difference

References

- **Athey, S., & Imbens, G. (2021)**. Design-based analysis in difference-in-differences settings with staggered adoption, *Journal of Econometrics*.
- **Bertrand, M., Dufflo, E., & Mullainathan, S. (2004)**. How much should we trust differences-in-differences estimates?, *The Quarterly Journal of Economics*, 119(1), 249-275.
- **Goodman-Bacon, A. (2021)**. Difference-in-differences with variation in treatment timing, *Journal of Econometrics*, 255(2), 254-277.

1.13 Synthetic Control

A lecture on synthetic control method will be part of the next iteration of the OSE data science course, summer semester 2022. Details on this lecture will be realized soon.

1.13.1 Synthetic Control

> The model extends the traditional linear panel data (difference-in-differences) framework, allowing that **the effects of unobserved variables on the outcome vary with time**. (Abadie & Diamond & Hainmueller (2010))

-> This is the key differnce to the difference-in-difference design. However, it is important to clarify that this statement refers to **!time-constant!** unobserved confounders. Now, the intuition that reproducing well a long time-series of pre-treatment outcomes of the eventually treated unit with a weighted average of the donor pool also picks up the effect of **unobserved confounders**. Then, because these are time-constant, their time-varying effect after treatment is also incporporated.

Consider the following factor model:

$$Y_{it}^N = \delta_t + \theta_t Z_i + \lambda_t \mu_i + \epsilon_{it}$$

If $\lambda_t = \lambda$, i.e. λ_t is constant over time, then we are back in the standard setting.

References

- **Abadie, A. (2021)**. Using synthetic controls: feasibility, data requirements, and methodological aspects, *Journal of Economic Literature*, 59(2), 391-425.
- **Abadie, A., Diamond, A., & Hainmueller, J. (2010)**. Synthetic control methods for comparative casecStudies: cstimating the effect of california's tobacco control program, *Journal of the American Statistical Association*, 105(490), 493-505.
- **Firpo, S., & Possebom, V. (2018)**. Synthetic control method: inference, sensitivity analysis and confidence sets, Journal of Causal Inference, 6(2), 2-26.

CHAPTER

TWO

PROBLEM SETS

We provide a set of problem sets to revisit selected issues we discussed during class.

2.1 Potential outcome model

We explore the potential outcome model using observed and simulated data inspired by the National Health Interview Survey. The accompanying data sets are available here.

2.1.1 Potential outcome model

[15]:
```python
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

pd.options.display.float_format = "{:,.2f}".format
```

The National Health Interview Survey (NHIS) collects data on U.S. households since 1957. It covers a broad range of health-related topics, from medical conditions, health insurance, and the number of doctor visits to measures of physical activity. Here we focus on indicators relevant to the Potential outcome model (POM) framework. In particular, we will compare the health status of hospitalized and non-hospitalized individuals in 2018. For this purpose, we use answers to the survey question **During the past 12 months, has the respondent been hospitalized overnight?** with potential answers **Yes** and **No**, which we code as one and zero. Further, we consider answers to the questions **Would you say your health, in general, is excellent, very good, good, fair, poor?** where responses are coded as one for poor health up to five for excellent health. The survey also collects data on relevant characteristics as sex, age, level of education, hours worked last week, and total earnings.

Import the data set **nhis-initial.xslx** (raw file available in our course repository). Try to think of ways to answer the following questions: Are there more females or males? Are there more individuals who hold a degree or not?. Now try to relate individual characteristics to the hospitalization status. Are high or low earners/old or young people more often hospitalized?

[16]:
```python
df = pd.read_excel("data/nhis-initial.xls", index_col=0)
df.index.set_names("Individual", inplace=True)
df.head()
```

[16]:
```
                sex  age education  hours earnings  hospitalized  health
Individual
0              male   49  bachelor     32      low             0       3
1              male   37       PhD     40     high             0       3
2            female   36  bachelor     40     high             0       4
```

(continues on next page)

(continued from previous page)

```
3              male    29   bachelor      25    middle              0        4
4            female    34   bachelor      40    middle              0        5
```

We will have to do so repeatedly, so let's streamline this process and set up a proper function.

[17]:
```python
def get_dataset(fname="initial"):
    df = pd.read_excel(f"data/nhis-{fname}.xls", index_col=0)
    df.index.set_names("Individual", inplace=True)
    return df
```

Let us get a basic feel for the data in front of us.

[18]:
```python
for column in df.columns:
    print("\n", column.capitalize())
    print(df.groupby("hospitalized")[column].describe())
```

```
 Sex
             count unique     top   freq
hospitalized
0            25320      2    male  13450
1             1496      2  female    938

 Age
                count  mean   std   min   25%   50%   75%   max
hospitalized
0           25,320.00 43.45 13.87 13.00 32.00 43.00 54.00 85.00
1            1,496.00 45.95 15.17 18.00 33.00 45.00 58.25 85.00

 Education
             count unique       top   freq
hospitalized
0            25320      5  bachelor  14006
1             1496      5  bachelor    845

 Hours
                count  mean   std  min   25%   50%   75%   max
hospitalized
0           25,320.00 40.60 13.49 1.00 38.00 40.00 45.00 99.00
1            1,496.00 38.78 14.00 1.00 33.75 40.00 43.00 99.00

 Earnings
             count unique  top   freq
hospitalized
0            25320      3  low  12930
1             1496      3  low    852

 Hospitalized
                count  mean  std  min  25%  50%  75%  max
hospitalized
0           25,320.00  0.00 0.00 0.00 0.00 0.00 0.00 0.00
1            1,496.00  1.00 0.00 1.00 1.00 1.00 1.00 1.00
```

(continues on next page)

(continued from previous page)

```
 Health
                  count  mean  std  min  25%  50%  75%  max
hospitalized
0             25,320.00  3.97 0.89 1.00 3.00 4.00 5.00 5.00
1              1,496.00  3.59 1.05 1.00 3.00 4.00 4.00 5.00
```

We want to study average age and working hours in more detail. What are their averages in our data?

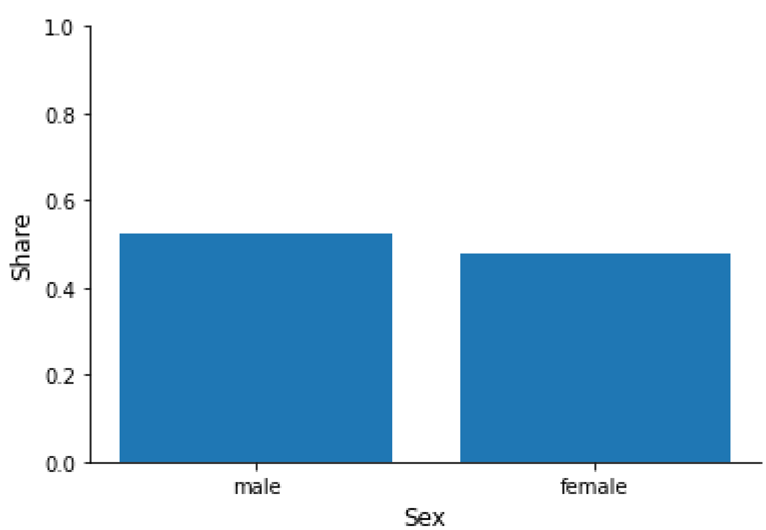
[19]:
```
stat = df["age"].mean()
print(f"Average age in the sample is {stat:.2f}")
```

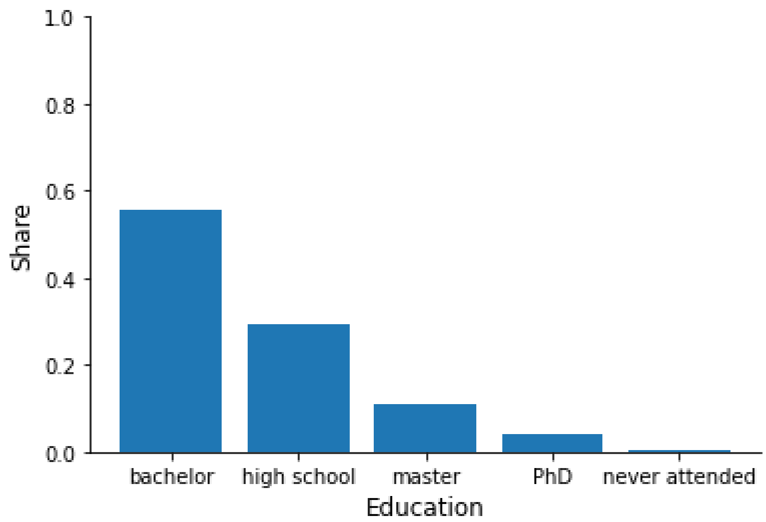
```
Average age in the sample is 43.59
```

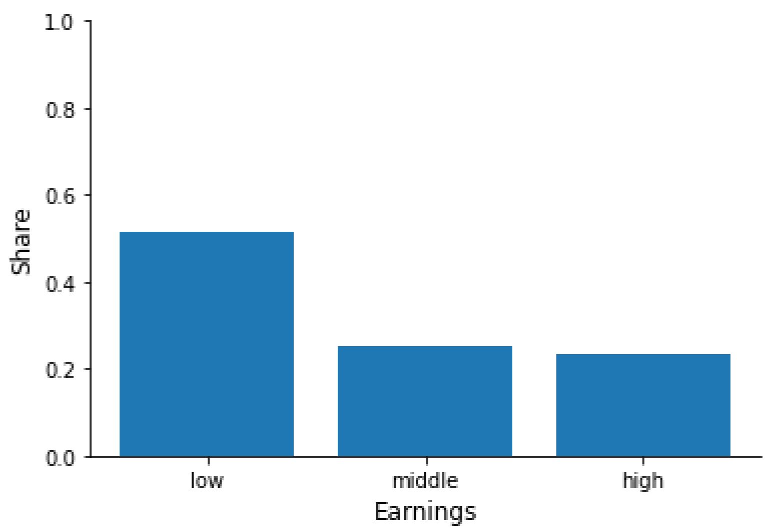
[20]:
```
stat = df["hours"].mean()
print(f"Average of working hours per week in the sample is {stat:.0f}")
```

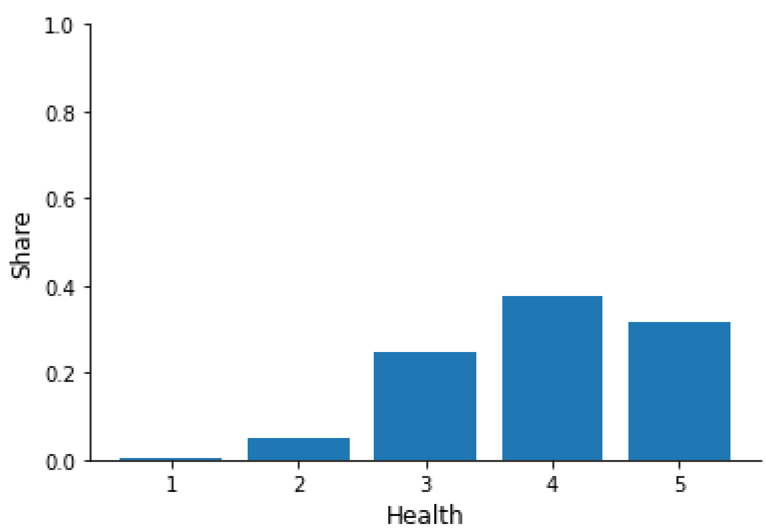
```
Average of working hours per week in the sample is 40
```

[21]:
```
for column in ["sex", "education", "earnings", "health"]:

    fig, ax = plt.subplots()

    info = df[column].value_counts(normalize=True)
    x, y = info.index, info.to_numpy()

    ax.bar(x, y)

    ax.set_xlabel(column.capitalize())

    ax.set_ylim(None, 1)
    ax.set_ylabel("Share")
```









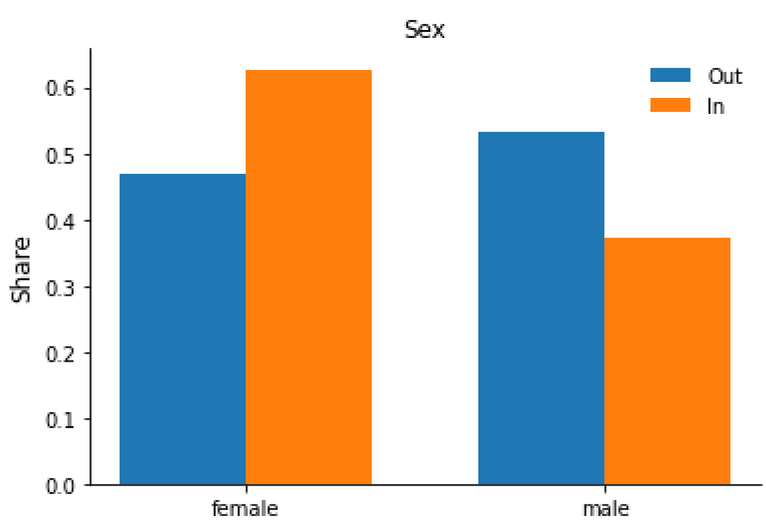
Now try to relate individual characteristics to the hospitalization status.

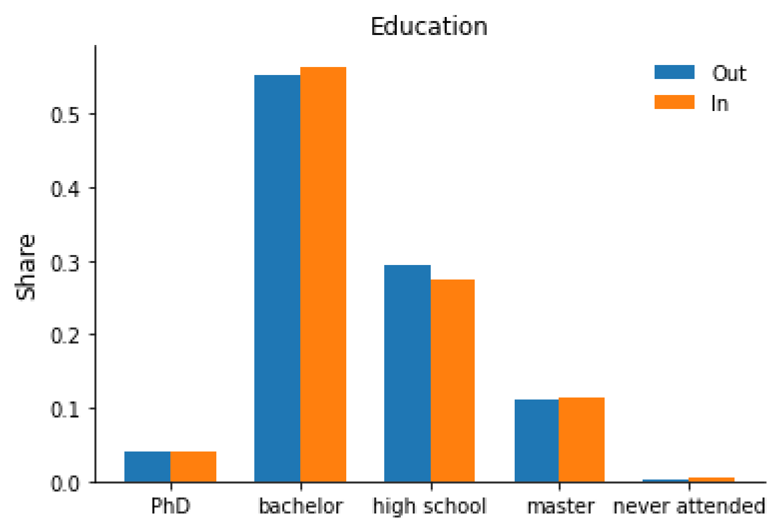
[22]:
```
df.groupby("hospitalized")[["age", "hours"]].mean()
```

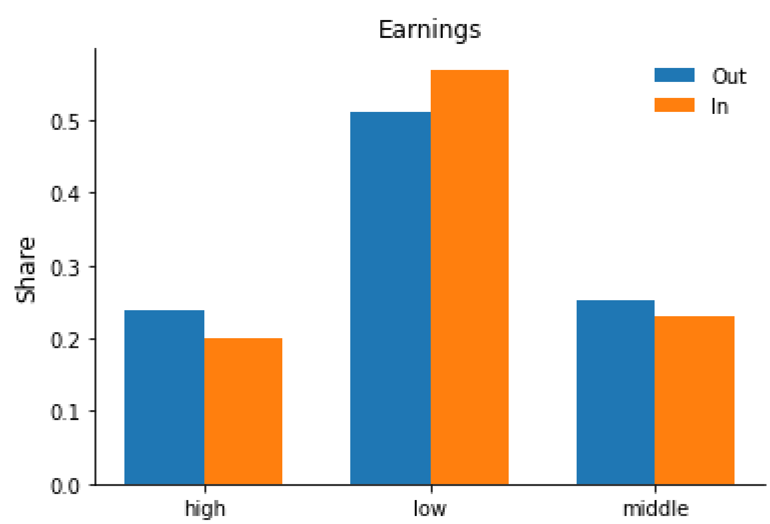
[22]:
```
                age  hours
hospitalized
0             43.45  40.60
1             45.95  38.78
```

Let's practice some plotting and set up a grouped bar chart to explore differences in the observables by hospitalization status. Some additional explanations are available as part of the `matplotlib` gallery here.

[49]:
```
width = 0.35
for column in ["sex", "education", "earnings"]:

    fig, ax = plt.subplots()

    rslt = df.groupby("hospitalized")[column].value_counts(normalize=True).sort_index()
    y_out, y_in = rslt[0].to_numpy(), rslt[1].to_numpy()
    labels = rslt.index.get_level_values(1).unique().sort_values()

    x = np.array(range(len(y_out)))

    ax.bar(x - width / 2, y_out, width, label="Out")
    ax.bar(x + width / 2, y_in, width, label="In")

    ax.set_xticks(x)
    ax.set_xticklabels(labels)

    ax.legend()
    ax.set_title(column.capitalize())
    ax.set_ylabel("Share")
```







Task A.2

Compute the average health status of hospitalized and non-hospitalized individuals. Who is healthier on average? What could be a reason for this difference?

[10]:
```
df.groupby("hospitalized")["health"].mean().to_frame()
```

[10]:

	health
hospitalized	
0	3.97
1	3.59

Task A.3

Adjust the data set for the POM framework, with health status as the outcome and hospitalization as the treatment status.

[11]:
```
df = get_dataset()

df.rename(columns={"health": "Y", "hospitalized": "D"}, inplace=True)

df["Y_1"] = np.where(df["D"] == 1, df["Y"], np.nan)
df["Y_0"] = np.where(df["D"] == 0, df["Y"], np.nan)

df.head()
```

[11]:
```
               sex  age education  hours earnings  D  Y  Y_1  Y_0
Individual
0             male   49  bachelor     32      low  0  3  NaN 3.00
1             male   37       PhD     40     high  0  3  NaN 3.00
2           female   36  bachelor     40     high  0  4  NaN 4.00
3             male   29  bachelor     25   middle  0  4  NaN 4.00
4           female   34  bachelor     40   middle  0  5  NaN 5.00
```

Task A.4

Compute the naive estimate for the average treatment effect (ATE)

[12]:
```
stat = df["Y_1"].mean() - df["Y_0"].mean()
print(f"Our naive estimate is {stat:.1f}")
```

```
Our naive estimate is -0.4
```

Task B.1

As we've seen in the lecture, in reality, we can only ever observe one counterfactual; however, when simulating data, we can bypass this problem. The (simulated) data set nhis-simulated.xslx (raw file available in our course repository) contains counterfactual outcomes, i.e., outcomes under control for individuals assigned to the treatment group and vice versa. Derive and compute the average outcomes in the two observable and two unobservables states. Design them similar to Table 2.3 in Morgan & Winship (2014).

[102]:
```
df = get_dataset("simulated")
```

[103]:
```
rslt = df.groupby("D")[["Y_1", "Y_0"]].mean()

rslt.columns = ["E[Y_1|D]", "E[Y_0|D]"]
rslt.index = ["Untreated", "Treated"]

rslt
```

[103]:
```
           E[Y_1|D]  E[Y_0|D]
Untreated      4.87      3.97
Treated        3.59      3.90
```

Task B.2

From here on we assume that 5% of the population take the treatment. Derive and explain Equation (2.10) from Morgan & Winship (2014) for the naive estimator as a decomposition of true ATE, baseline bias, and differential treatment effect bias.

This derivation is straightforward.

Task B.3

Compute the naive estimate and true value of the ATE for the simulated data. Is the naive estimator upwardly or downwardly biased? Calculate the baseline bias and differential treatment effect bias. How could we interpret these biases in our framework of health status of hospitalized and non-hospitalized respondents?

[104]:
```
pi = 0.05

# naive estimate
naive = rslt.loc["Treated", "E[Y_1|D]"] - rslt.loc["Untreated", "E[Y_0|D]"]

# baseline bias
base = rslt.loc["Treated", "E[Y_0|D]"] - rslt.loc["Untreated", "E[Y_0|D]"]

# differential effect
diff = 0
diff += rslt.loc["Treated", "E[Y_1|D]"] - rslt.loc["Treated", "E[Y_0|D]"]
diff -= rslt.loc["Untreated", "E[Y_1|D]"] - rslt.loc["Untreated", "E[Y_0|D]"]
diff *= 1 - pi

# true average treatment effect
true = 0
true += pi * (rslt.loc["Treated", "E[Y_1|D]"] - rslt.loc["Treated", "E[Y_0|D]"])
true += (1 - pi) * (rslt.loc["Untreated", "E[Y_1|D]"] - rslt.loc["Untreated", "E[Y_0|D]
↪"])
print(f"naive: {naive:.2f}, base: {base:.2f}, diff: {diff:.2f}, true: {true:.2f}")

# We can also test the relationships just to be sure.
np.testing.assert_almost_equal(true, naive - (base + diff), decimal=10)
```

```
naive: -0.38, base: -0.07, diff: -1.14, true: 0.84
```

Task B.4

Under which assumptions does the naive estimator provide the ATE?

We need the *stable unit treatment value assumption* and independence between potential outcomes and the treatment.

References

- **Winship, C., and Morgan, S. L. (2014)**. Counterfactuals and causal inference: Methods and principles for social research. Cambridge, England: *Cambridge University Press*.
- **Angrist, J. D., and Pischke, J. (2009)**. Mostly harmless econometrics: An empiricists companion. Princeton, NJ: *Princeton University Press*.
- **National Center for Health Statistics (2018)**. National Health Interview Survey..

2.2 Matching estimators

We compare the consistency of regression and matching estimators using LaLonde (1986) framework and the Current Population Survey data. The accompanying data sets are available here.

[37]:
```
from sklearn.neighbors import NearestNeighbors
import statsmodels.formula.api as smf
from scipy.stats import ttest_ind
import matplotlib.pyplot as plt
import pandas as pd

pd.options.display.float_format = "{:,.2f}".format
```

2.2.1 Regression and matching estimators in causal effects

In this problem set we are going to compare the consistency of regression and matching estimators of causal effects based on Dehejia & Wahba (1999). For that we employ the experimental study from LaLonde (1986), which provides an opportunity to estimate true treatment effects. We then use these results to evaluate the performance of (treatment effect) estimators one can usuallly obtain in observational studies.

LaLonde (1986) implements the data from the National Supported Work program (NSW) – temporary employment program designed to help disadvantaged workers lacking basic job skills move into the labor market by giving them work experience and counseling in sheltered environment. Unlike other federally sponsored employment programs, the NSW program assigned qualiffed applications randomly. Those assigned to the treatment group received all the bene ts of the NSW program, while those assigned to the control group were left to fend for themselves.

To produce the observational study, we select the sample from the Current Population Survey (CPS) as the comparison group and merge it with the treatment group. We do this to obtain a data set which resembles the data which is commonly used in scientific practice. The two data sets are explained below:*

- **nsw_dehejia.csv** is field-experiment data from the NSW. It contains variables as education, age, ethnicity, marital status, preintervention (1975) and postintervention (1978) earnings of the eligible male applicants. Dehejia & Wahba (1999) also transform the LaLonde (1986) data set to have observations on preintervention 1974 earnings; motivation is explained in their paper.
- **cps.csv** is a non-experimental sample from the CPS which selects all males under age 55 and contains the same range of variables.

Task A

Create the table with the sample means of characteristics by age, education, preintervention earnings, etc. for treated and control groups of NSW sample (you can use the Table 1 from Dehejia and Wahba (1999) as a benchmark). Is the distribution of preintervention variables similar across the treatment and control groups? Check the differences on significance. Add to the table the CPS sample means. Is the comparison group different from the treatment group in terms of age, marital status, ethnicity, and preintervention earnings?

[87]:
```python
demographics = ["age", "ed", "black", "hisp", "married", "nodeg", "age2"]

dtypes = dict()
for column in ["treat"] + demographics:
    dtypes[column] = int

df_nsw = pd.read_csv("data/nsw_dehejia.csv", dtype=dtypes)
df_nsw.index.name = "individual"
df_nsw.head()
```

[87]:
```
            treat  age  ed  black  hisp  married  nodeg  re74  re75      re78  \
individual
0               1   37  11      1     0        1      1  0.00  0.00  9,930.05
1               1   22   9      0     1        0      1  0.00  0.00  3,595.89
2               1   30  12      1     0        0      0  0.00  0.00 24,909.45
3               1   27  11      1     0        0      1  0.00  0.00  7,506.15
4               1   33   8      1     0        0      1  0.00  0.00    289.79

            age2
individual
0           1369
1            484
2            900
3            729
4           1089
```

How does a summary of the data look like?

[88]:
```python
df_nsw.describe()
```

[88]:
```
       treat    age     ed  black   hisp  married  nodeg      re74      re75  \
count 445.00 445.00 445.00 445.00 445.00   445.00 445.00    445.00    445.00
mean    0.42  25.37  10.20   0.83   0.09     0.17   0.78  2,102.27  1,377.14
std     0.49   7.10   1.79   0.37   0.28     0.37   0.41  5,363.58  3,150.96
min     0.00  17.00   3.00   0.00   0.00     0.00   0.00      0.00      0.00
25%     0.00  20.00   9.00   1.00   0.00     0.00   1.00      0.00      0.00
50%     0.00  24.00  10.00   1.00   0.00     0.00   1.00      0.00      0.00
75%     1.00  28.00  11.00   1.00   0.00     0.00   1.00    824.39  1,220.84
max     1.00  55.00  16.00   1.00   1.00     1.00   1.00 39,570.68 25,142.24

          re78   age2
count   445.00 445.00
mean  5,300.76 693.98
std   6,631.49 429.78
min       0.00 289.00
25%       0.00 400.00
50%   3,701.81 576.00
```

(continues on next page)

(continued from previous page)

```
75%    8,124.72   784.00
max   60,307.93 3,025.00
```

Let's look at the mean differences by treatment status.

```
[68]: df_nsw.groupby("treat").mean()
```

```
[68]:        age    ed  black  hisp  married  nodeg     re74     re75     re78  \
treat
0      25.05 10.09   0.83  0.11     0.15   0.83 2,107.03 1,266.91 4,554.80
1      25.82 10.35   0.84  0.06     0.19   0.71 2,095.57 1,532.06 6,349.14

         age2
treat
0      677.32
1      717.39
```

```
[69]: df_nsw.groupby("treat").mean().diff()
```

```
[69]:       age   ed  black  hisp  married  nodeg   re74   re75     re78  age2
treat
0       NaN  NaN    NaN   NaN      NaN    NaN    NaN    NaN      NaN   NaN
1      0.76 0.26   0.02 -0.05     0.04  -0.13 -11.45 265.15 1,794.34 40.08
```

Are these differences statistically significant?

```
[89]: for column in demographics:

    treated = df_nsw.query("treat == 1")[column]
    control = df_nsw.query("treat == 0")[column]

    stat = ttest_ind(treated, control)[1]

    print(f"{column:<7}      {stat:7.3f}")
```

```
age           0.265
ed            0.135
black         0.649
hisp          0.076
married       0.327
nodeg         0.001
age2          0.333
```

```
[90]: df_cps = pd.read_csv("data/cps.csv", dtype=dtypes)
df_cps.index.name = "individual"
df_cps.head()
```

```
[90]:            treat  age  ed  black  hisp  married  nodeg      re74      re75  \
individual
0               0   45  11      0     0        1      1 21,516.67 25,243.55
1               0   21  14      0     0        0      0  3,175.97  5,852.56
2               0   38  12      0     0        1      0 23,039.02 25,130.76
3               0   48   6      0     0        1      1 24,994.37 25,243.55
4               0   18   8      0     0        1      1  1,669.30 10,727.61
```

(continues on next page)

(continued from previous page)

```
                 re78  age2
individual
0           25,564.67  2025
1           13,496.08   441
2           25,564.67  1444
3           25,564.67  2304
4            9,860.87   324
```

How does a summary of the data look like?

[91]:
```
df_cps.describe()
```

[91]:
```
           treat       age        ed     black      hisp   married     nodeg  \
count 15,992.00 15,992.00 15,992.00 15,992.00 15,992.00 15,992.00 15,992.00
mean       0.00     33.23     12.03      0.07      0.07      0.71      0.30
std        0.00     11.05      2.87      0.26      0.26      0.45      0.46
min        0.00     16.00      0.00      0.00      0.00      0.00      0.00
25%        0.00     24.00     11.00      0.00      0.00      0.00      0.00
50%        0.00     31.00     12.00      0.00      0.00      1.00      0.00
75%        0.00     42.00     13.00      0.00      0.00      1.00      1.00
max        0.00     55.00     18.00      1.00      1.00      1.00      1.00

            re74      re75      re78      age2
count 15,992.00 15,992.00 15,992.00 15,992.00
mean  14,016.80 13,650.80 14,846.66  1,225.91
std    9,569.80  9,270.40  9,647.39    784.74
min        0.00      0.00      0.00    256.00
25%    4,403.45  4,398.82  5,669.30    576.00
50%   15,123.58 14,557.11 16,421.97    961.00
75%   23,584.18 22,923.74 25,564.67  1,764.00
max   25,862.32 25,243.55 25,564.67  3,025.00
```

Let's compare mean differences between the synthetic control group and the treatment group.

[92]:
```
for column in demographics:

    treated = df_nsw.query("treat == 1")[column]
    control = df_cps[column]

    stat = ttest_ind(treated, control)[1]

    print(f"{column:<7}     {stat:7.3f}")
```

```
age            0.000
ed             0.000
black          0.000
hisp           0.510
married        0.000
nodeg          0.000
age2           0.000
```

Task B. Regression Adjustment

In this section we compare the results of regression estimates with selection on observables as discussed in the lecture 6.

Task B.1

Merge the treatment group data from the NSW sample with the comparison group data from the CPS sample to imitate an observational study.

[93]:
```python
df_nsw["sample"] = "NSW"
df_cps["sample"] = "CPS"

df_obs = pd.concat([df_nsw.query("treat == 1"), df_cps])
df_obs.set_index(["sample"], append=True, inplace=True)
df_obs.sort_index(inplace=True)

df_obs.loc[(slice(1, 5), "NSW"), :]
```

[93]:
```
                   treat  age  ed  black  hisp  married  nodeg  re74  re75  \
individual sample
1          NSW         1   22   9      0     1        0      1  0.00  0.00
2          NSW         1   30  12      1     0        0      0  0.00  0.00
3          NSW         1   27  11      1     0        0      1  0.00  0.00
4          NSW         1   33   8      1     0        0      1  0.00  0.00
5          NSW         1   22   9      1     0        0      1  0.00  0.00

                       re78  age2
individual sample
1          NSW     3,595.89   484
2          NSW    24,909.45   900
3          NSW     7,506.15   729
4          NSW       289.79  1089
5          NSW     4,056.49   484
```

Task B.2

Which assumption need to hold such that conditioning on observables can help in obtaining an unbiased estimate of the true treatment effect?

$$E[Y^1|D=1,S]=E[Y^1|D=0,S]$$

$$E[Y^0|D=1,S]=E[Y^0|D=0,S]$$

Task B.3

Run a regression on both experimental and non-experimental data using the specification: RE78 on a constant, a treatment indicator, age, age2, education, marital status, no degree, black, hispanic, RE74, and RE75. We recommend using statsmodels, but you are free to use any other software. Is the treatment effect estimate of the observational study consistent with the true estimate?

We first construct the regression equation.

[94]:
```
indep_vars = df_obs.columns.tolist()
indep_vars.remove("re78")

formula = "re78 ~ " + " " " + ".join(indep_vars)
formula
```

[94]:
```
're78 ~ treat  + age  + ed  + black  + hisp  + married  + nodeg  + re74  + re75  + age2'
```

Now we can run the model on both datasets.

[95]:
```
for label, data in [("observational", df_obs), ("experimental", df_nsw)]:
    stat = smf.ols(formula=formula, data=data).fit().params["treat"]
    print(f"Estimate based on {label} data: {stat:7.3f}")
```

```
Estimate based on observational data: 793.587
Estimate based on experimental data: 1675.862
```

Task C. Matching on Propensity Score

Recall that the propensity score p(Si) is the probability of unit i having been assigned to treatment. Most commonly this function is modeled to be dependent on various covariates. We write $p(S_i) := Pr(D_i = 1|S_i) = E(D_i|S_i)$. One assumption that makes estimation strategies feasible is $S_i \perp D_i|p(S_i)$ which means that, conditional on the propensity score, the covariates are independent of assignment to treatment. Therefore, conditioning on the propensity score, each individual has the same probability of assignment to treatment, as in a randomized experiment.*

Estimation is done in two steps. First, we estimate the propensity score using a logistic regression model. Secondly, we match the observations on propensity score employing nearest-neighbor algorithm discussed in the lecture 5. That is, each treatment unit is matched to the comparison unit with the closest propensity score – the unmatched comparison units are discarded.

Task C.1

Before we start with matching on propensity score, let's come back to another matching strategy which was discussed in Lecture 5 - matching on stratification. Looking at the data could you name at least two potential reasons why matching on stratification might be impossible to use here?

Data contains continuous variables; formed stratas might not have treated and control units available at the same time.

Task C.2

Employing our imitated observational data run a logistic regression on the following specification: treatment indicator on age, education, marital status, no degree, black, hispanic, RE74, and RE75. Use, for example, `statsmodels <https://www.statsmodels.org/stable/index.html>`__for this task. Then extract a propensity score for every individual as a probability to be assigned into treatment.

[96]:
```
formula = "treat ~ age + ed + black + hisp + married + nodeg + re74 + re75"
df_obs["pscore"] = smf.logit(formula=formula, data=df_obs).fit().predict()
```

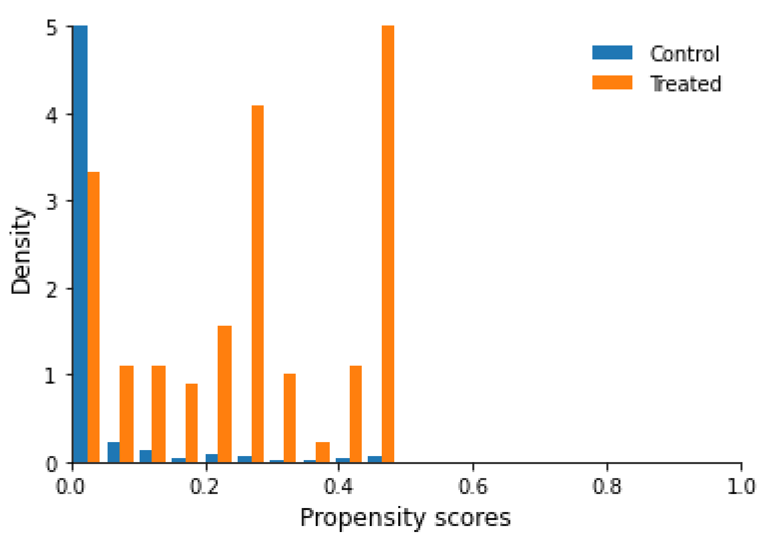
```
Optimization terminated successfully.
         Current function value: 0.031035
         Iterations 12
```

Task C.3

Before proceeding further we have to be sure that propensity scores of treatment units overlap with the propensity scores of control units. Draw a figure showing the distribution of propensity score across treatment and control units (we use the packages matplotlib and seaborn). Do we observe common support?

[97]:
```
fig, ax = plt.subplots()

df_control = df_obs.query("treat == 0")["pscore"]
df_treated = df_obs.query("treat == 1")["pscore"]

ax.hist([df_control, df_treated], density=True, label=["Control", "Treated"])

ax.set_ylim(0, 5)
ax.set_xlim(0, 1)
ax.set_ylabel("Density")
ax.set_xlabel("Propensity scores")
ax.legend()
```

[97]:
```
<matplotlib.legend.Legend at 0x7fa33de94590>
```



Task C.4

Match each treatment unit with control unit one-to-one with replacement. We use the package sklearn.neighbors: apply the algorithm NearestNeighbors to the propensity score of treated and control units and extract the indices of matched control units.

[98]:
```
def get_matched_dataset(df):
    training_data = df.query("treat == 0")["pscore"].to_numpy().reshape(-1, 1)
    eval_point = df.query("treat == 1")["pscore"].to_numpy().reshape(-1, 1)

    neigh = NearestNeighbors(n_neighbors=1)

    neigh.fit(training_data)
    matched = neigh.kneighbors(eval_point, return_distance=False)[:, 0]

    df_treated = df.query("treat == 1")
    df_matched = df.query("treat == 0").iloc[matched]

    df_sample = pd.concat([df_treated, df_matched])

    return df_sample
```

Task C.5

Construct new data set with matched observations. Run the regression to obtain matching on propensity score estimate. Is it more or less consistent estimate of the true effect comparing to the regression estimate with selection on observables? How could you explain this result?

[99]:
```
df_sample = get_matched_dataset(df_obs)
stat = smf.ols(formula="re78 ~ treat", data=df_sample).fit().params["treat"]
print(f"Estimate based on matched for re78 data: {stat:7.3f}")
```

```
Estimate based on matched for re78 data: 1551.477
```

Regression model neglects important nonlinear terms and interactions (Rubin 1973). The benefit of matching over regression is that it is non-parametric (but you do have to assume that you have the right propensity score specification in case of matching).

Let's further explore two selected issues in matching, i.e. the use of placebo testing and trimming.

[100]:
```
stat = smf.ols(formula="re75 ~ treat", data=df_sample).fit().params["treat"]
print(f"Estimate based on matched for re75 data: {stat:7.3f}")
```

```
Estimate based on matched for re75 data: 221.917
```

What happens if we trim our dataset?

[84]:
```
for value in [0.025, 0.05, 0.1, 0.15]:

    lower, upper = value, 1 - value
    df_trimmed = df_obs.loc[df_obs["pscore"].between(lower, upper), :]

    df_sample = get_matched_dataset(df_trimmed)

```

(continues on next page)

(continued from previous page)

```
    stat = smf.ols(formula="re78 ~ treat", data=df_sample).fit().params["treat"]
    print(f"{value:5.3f}: {stat:7.3f}")
```

```
0.025: 1563.983
0.050: 1665.306
0.100: 1744.330
0.150: 2138.977
```

References

- **Bureau of Labor Statistics. (1974, 1975, 1978)**. Current Population Survey.
- **Dehejia, R., and Wahba, S. (1999)**. Causal effects in nonexperimental studies: Reevaluating the evaluation of training programs. *Journal of the American Statistical Association*, 94(448), 1053-1062.
- **LaLonde, R. J. (1986)**. Evaluating the econometric evaluation of training programs with experimental data. *American Economic Review*, 76(4), 604-620.

2.3 Regression discontinuity design

We practice RDD with Lee (2008) framework. In particular, we illustrate a discontinuity at the cutoff point with local averages graph, estimate treatment effect by local linear regression and choose an optimal bandwidth by cross-validation procedure. The accompanying data sets are available here.

[1]:
```
from functools import partial

import statsmodels.formula.api as smf
import pandas as pd
import numpy as np

from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import LeaveOneOut

from auxiliary import plot_bandwidth
from auxiliary import plot_logistic
```

2.3.1 Regression Discontinuity Design (RDD)

In the problem set we are going to practice RDD in the Lee (2008) framework. We employ the original simplified data set on the individual candidates for the US House of Representatives from 1946 to 1998. If a candidate obtains more votes than his or her competitors, he or she takes the office. Each elected candidate represents one of 435 congressional districts. The elections are held every two years. We seek the answer to the question whether winning the election has a causal influence on the probability that the candidate will win the next election.

The observations of the data set **individ_final.dta** are clustered by district and election year. It consists of the following variables:*

- **outcome** is a treatment variable; it is coded as 1 if a candidate won the election in the corresponding year and 0 – otherwise.

- **outcomenext** is an outcome variable. It is coded as 1 if a candidate won the next election; as 0 if he or she did not win the next election; and as -1 if he or she did not participate in the next election.
- **difshare** is an assignment variable; it is the winning candidate's vote share minus the vote share of the highest performing competitor. Therefore, 0 is the cutoff point: a candidate whose vote share is more than 0 is automatically assigned to treatment.

[2]:
```
df = pd.read_stata("data/individ_final.dta")
df.index.set_names("Identifier", inplace=True)
df.head()

# Better handling of missing values
df.replace({"outcomenext": {-1: np.nan}}, inplace=True)
```

Task A

What is the main assumption that makes RDD possible? Define the local randomization condition in the simplified setup presented in the lecture.

Main assumption: agents are unable to precisely control the assignment variable near the known cutoff what leads to the randomized variation in treatment near the threshold.

The framework:

$$Y = D\tau + W\delta_1 + U$$

$$D = I(X \geq c)$$

$$X = W\delta_2 + V,$$

where: - Y is the outcome of interest, - D is the binary treatment indicator, - W is the vector of all predetermined and observable characteristics of the individual that might impact Y and/or X, - X is the assignment variable, - c is the cutoff value

Individuals have imprecise control over X when conditional on W = w and U = u, the density of V (and hence X) is continuous.

Definition of Local Randomization: If individuals have imprecise control over X, then Pr[W = w,U = u|X = x] is continuous in x: the treatment is "as good as" randomly assigned around the cutoff.

Task B

A major advantage of the RD design over competing methods is its transparency, which can be illustrated using graphical methods. A standard way of graphing the data is to divide the assignment variable into a number of bins, making sure there are two separate bins on each side of the cutoff point. Then, the average value of the outcome variable can be computed for each bin and graphed against the mid-points of the bins.

Task B.1

Create a new variable that groups the assignment variable values into 400 bins with a size of 0.005.

```
[3]: df["bin"] = pd.cut(df["difshare"], 400, labels=False) / 200 - 1
     df.sort_values(by="bin", inplace=True)
     df.head()
```

```
[3]:             year  outcome  outcomenext  difshare  bin
     Identifier
     18052       1960        0          NaN -0.997953 -1.0
     16649       1946        0          NaN -0.999208 -1.0
     16651       1950        0          NaN -0.999915 -1.0
     16653       1952        0          NaN -0.997832 -1.0
     16655       1954        0          NaN -0.999943 -1.0
```

Task B.2

Since we are interested in a causal influence on the probability that the candidate will win the next election based on winning the current election, drop the rows that do not have a comparable next election.

How many missing values do we have?

```
[4]: df["outcomenext"].isna().sum()
```

```
[4]: 16403
```

Now get rid of all those observations.

```
[5]: df.dropna(inplace=True)
     np.testing.assert_equal(df["outcomenext"].isna().sum(), 0)
```

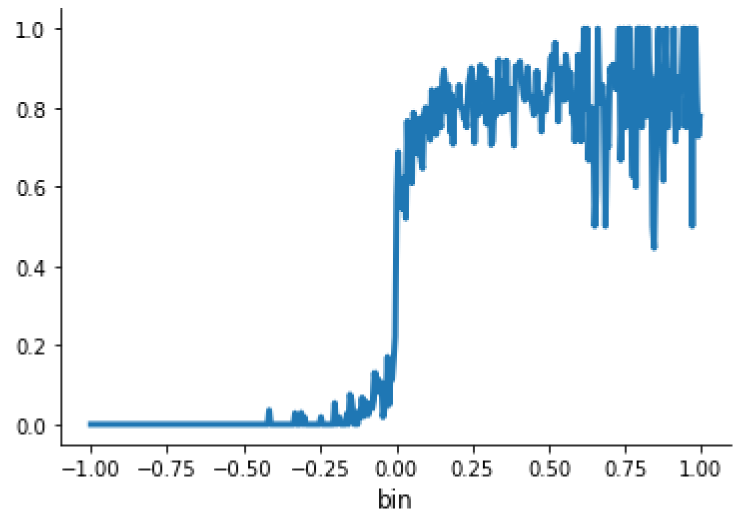
We can now build our remaining pipeline under the assumption that there are no missing values included. However, we might introduce them later again by some accidental operatoin. That is where data validation packages such as pandera come in handy. **pandera** allows to specify and check properties on your data easily .. and thus frequently.

Task B.3

Find the mean of the outcome variable for each bin or, in other words, local average. Draw this relationship on the scatterplot.

```
[6]: df.groupby("bin").mean()["outcomenext"].sort_index().plot()
```

```
[6]: <AxesSubplot:xlabel='bin'>
```



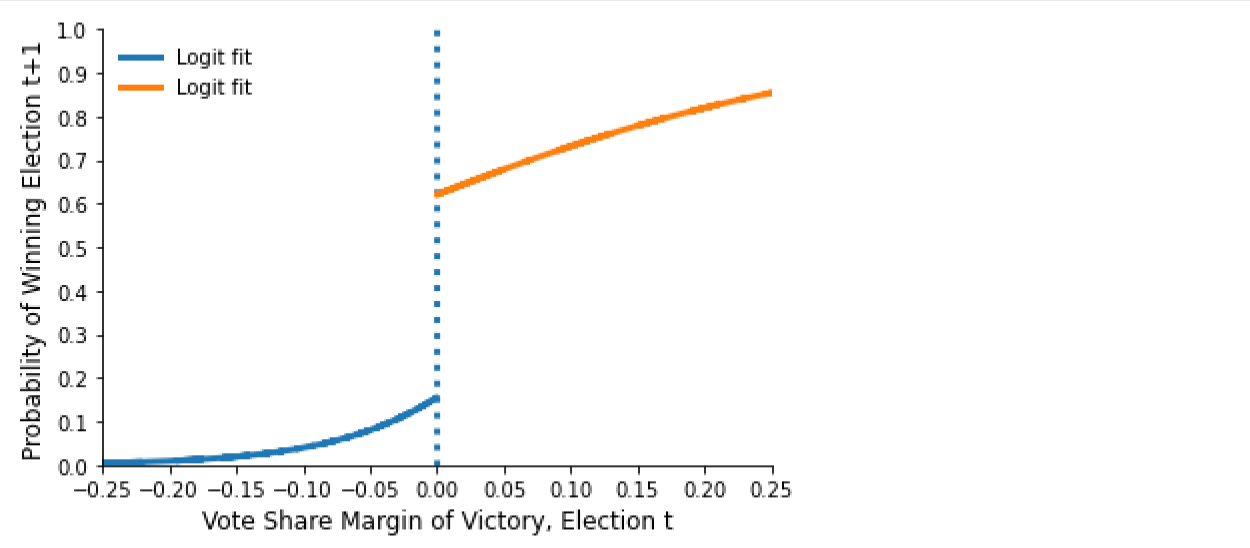
We will now repeatedly split the data at the cutoff.

[7]:
```
df["status"] = None
df.loc[df["difshare"].between(-0.25, +0.00), "status"] = "below"
df.loc[df["difshare"].between(+0.00, +0.25), "status"] = "above"
```

Task B.4

For better visuality we also add to the graph the fitted values of logistic regression around the cutoff. For this apply logistic regression separately on either side of the threshold (we take the bins with the share values from -0.25 to 0.25 and use the package LogisticRegression from sklearn.linear model). Extract probability estimates. Add them to the scatterplot in the proximity of cutoff. Do you observe a discontinuity at the cutoff point?

[8]:
```
probs = dict()

lr = LogisticRegression(C=1e20)
for label in ["below", "above"]:

    df_subset = df.query(f"status == '{label}'")

    y = df_subset["outcomenext"]
    x = df_subset[["difshare"]]

    lr.fit(x, y)
    probs[label] = lr.predict_proba(x)

plot_logistic(df, probs)
```



Task C

LLR as a method restricts the estimation to observations close to the cutoff. It is based on the assumption that regression lines within the bins around the cutoff point are close to linear. That helps to avoid some of the drawbacks of other parametric/non-parametrics approaches (Lee & Lemieux (2010))

Run the LLR with a specification $Y = $\alpha_r + \tau`D+ : nbsphinx - math : betaX + :nbsphinx-math:gammaX D + :nbsphinx-math:epsilon$, where :math: `Xisrectrictedbyabandwidth : $-h < X < h$. Interpret the result. Experiment with few bandwidths on your choice.

```
[9]: for h in [0.25, 0.2, 0.1, 0.05, 0.01]:
         df_subset = df[df["difshare"].between(-h, h)]
         formula = "outcomenext ~ outcome + difshare + difshare*outcome"
         rslt = smf.ols(formula=formula, data=df_subset).fit()
         info = [h, rslt.params[1] * 100, rslt.pvalues[1]]
         print(" Bandwidth: {:>4}    Effect {:5.3f}%    pvalue {:5.3f}".format(*info))
```

```
Bandwidth: 0.25   Effect 52.439%    pvalue 0.000
Bandwidth:  0.2   Effect 49.521%    pvalue 0.000
Bandwidth:  0.1   Effect 43.861%    pvalue 0.000
Bandwidth: 0.05   Effect 38.910%    pvalue 0.000
Bandwidth: 0.01   Effect 25.700%    pvalue 0.069
```

Task D

As you might find, the treatment effect result is sensitive to the bandwidth choice. In general, choosing a bandwidth in estimation involves finding an optimal balance between precision and bias. One the one hand, using a larger bandwidth yields more precise estimates as more observations are available to estimate the regression. On the other hand, the linear specification is less likely to be accurate (Lee & Lemieux (2010)).

We are going to review one of the approaches for choosing a bandwidth – cross-validation "leave one out" procedure. The main idea is to take an observation i in the data, leave it out, run LLR, and use the estimates to predict the value of Y at $X = X_i$. Proceeding with each observation separately on each side of the cutoff, we obtain the predicted values of Y that can be compared to the actual values. The optimal bandwidth is then a value of h that minimizes the mean square of the difference between the predicted and actual values of Y. And overall mean square error is simply the average of the squares of the prediction errors on each side of the cutoff.

Draw the graph showing the relationship between the bandwidth and the mean square error. What is the optimal bandwidth for LLR in our framework?*

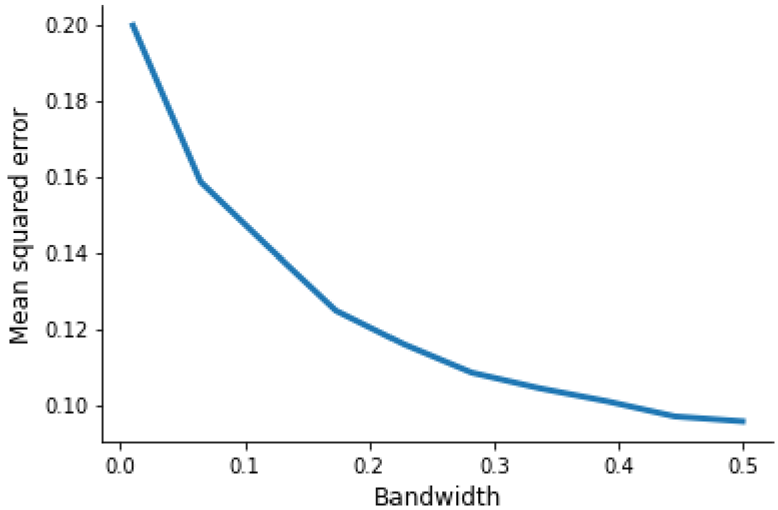
[10]:
```
num_points = 10
bandwidth = np.linspace(0.01, 0.50, num_points)

scoring = "neg_mean_squared_error"
model = LinearRegression()
cv = LeaveOneOut()

cross_val_score_p = partial(cross_val_score, scoring=scoring, cv=cv)
```

We are ready to now run the actual computations.

[11]:
```
rslts = pd.DataFrame(columns=["below", "above", "joint"])
rslts.index.set_names("Bandwidth", inplace=True)

for label in ["below", "above"]:
    for h in bandwidth:

        if label == "below":
            df_subset = df.loc[df["difshare"].between(-h, +0.00)]
        else:
            df_subset = df.loc[df["difshare"].between(+0.00, +h)]

        y = df_subset[["outcomenext"]]
        x = df_subset[["difshare"]]

        rslts.loc[h, label] = -cross_val_score_p(model, x, y).mean()

rslts["joint"] = rslts[["below", "above"]].mean(axis=1)
```

It is time for a visual inspection.

[12]:
```
plot_bandwidth(bandwidth, rslts["joint"])
```



What is the optimal bandwith in this setting?

[13]:
```
print(f" Optimal bandwidth: {rslts['joint'].idxmin():5.3f}")
```

```
 Optimal bandwidth: 0.500
```

References

- **Lee, D. S. (2008)**. Randomized experiments from non-random selection in US house elections. *Journal of Econometrics*, 142(2), 675–697.
- **Lee, D. S., & Lemieux, T. (2010)**. Regression discontinuity designs in economics. *Journal of Economic Literature*, 48, 281-355.

CHAPTER

THREE

HANDOUTS

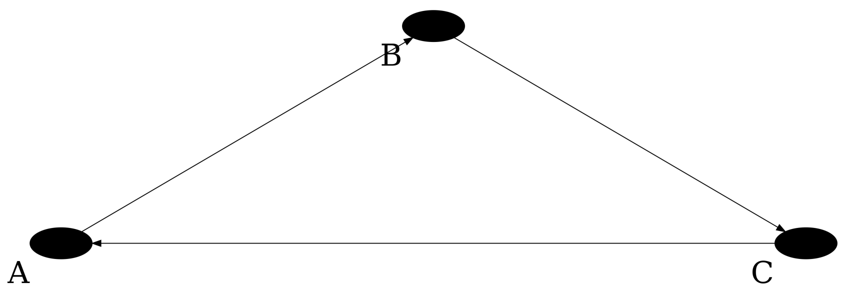
We curate a list of handouts that summarize selected issues.

3.1 Causal Graphs

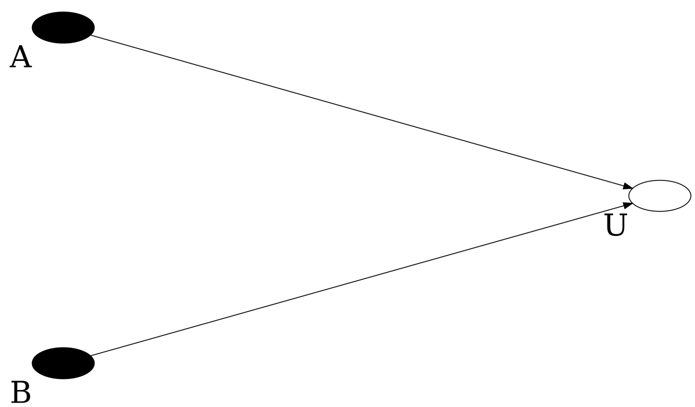
3.1.1 Definitions, patterns, and strategies

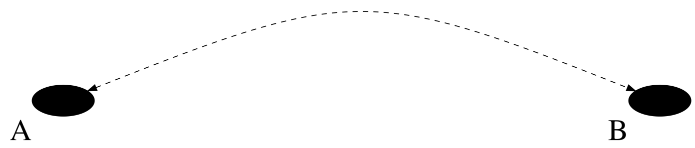
3.1.2 Definitions

- A **node** represents a random variable labeled by letter. Observed random variables are marked by solid circle • and unobserved - by hollow circle ∘.
- An **edge** shows dependence between joining variables.
- Adjacent variables are connected by an edge.
- **Adjacent edges** meet at a variable.
- A **directed edge** represents the cause by a single-headed arrow.
- A **parent/child** is the starting(tail)/ending(head) variable. Therefore, a directed edge represents a direct effect of a parent on a child.
- A **root** is a variable that has no parent. In other words, it is an exogenous variable determined only by forces outside of the graph.
- A **sink** is a variable with no children.
- A **path** is a sequence of adjacent edges.
- A **directed path** is a path traced out entirely along arrows tail-to-head. If there is a directed path from A to B, A is an **ancestor** of B; B is a **descendant** of A.
- A **directed acyclic graph (DAG)** is a graph with only arrows for edges and no feedback loops (i.e. no variable is its own ancestor or its own descendant):



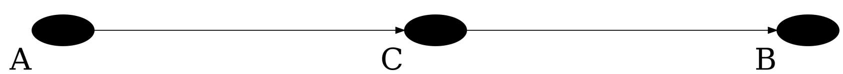
- **Joint dependence** of two variables on one or more common causes is shown either with unobservable variable or with bidirected dashed curved edge:



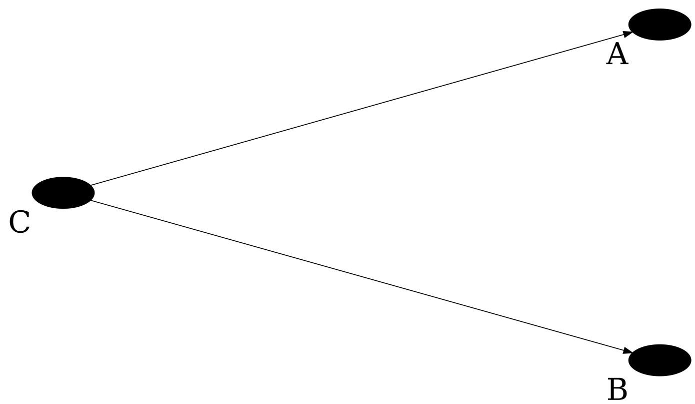


3.1.3 Patterns

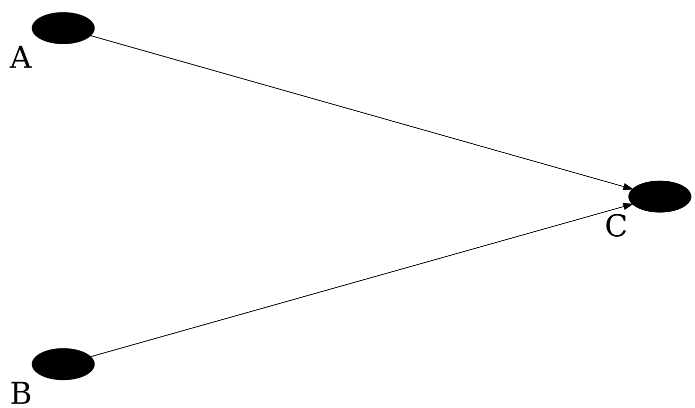
- **Chain of mediation** is a relationship when A affects B through A's causal effect on C and C's causal effect on B:



- **Mutual dependence** is a relationship when A and B are both caused by C (a variable C that affects both the dependent and independent variable is called a **confounding variable**):



- **Mutual causation** is a relationship when A and B are both causes of C (a variable C that has two arrows running into it is called a **collider**):

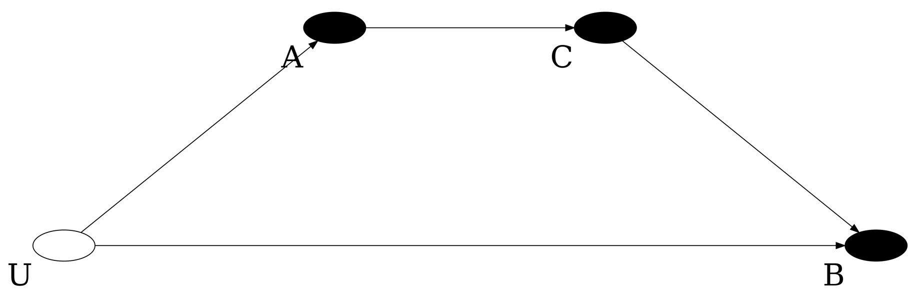


- A **back-door path** is a path between any causally ordered sequence of two variables that include a directed edge that points to the first variable.
- **Conditioning** as a modeling strategy means transforming one graph into a simpler set of component graphs where fewer causes are represented.

3.1.4 Strategies

A **back-door criterion** is a set of conditions used to determine whether or not conditioning on a given set of observed variable will identify the causal effect. The causal effect is identified by conditioning on a set of variables Z if and only if all back-door paths between the causal variable and the outcome variable are blocked after conditioning on Z. All back-door paths are blocked by Z if and only if each back-door path: - contains a chain of mediation AßCßB where the middle variable C is in Z, or - contains a fork of mutual dependence A¸CßB, where the middle variable C is in Z, or - contains an inverted fork of mutual causation AßC¸B, where the middle variable C and all of C's decendents are not in Z.

A **front-door criterion** is an empirical strategy used to identify the causal relationship flowing from A to B if one can find a mechanism C which: - lies on the causal path between A and B, and - it is the only such mechanism, and - it is not affected by the unobserved confounder U:



You can find more on front-door criterion application in the Bellemare & Bloem (2020) paper.

3.1.5 References

- **Bellemare, M., & Bloem, J. (2020)**. The paper of how: Estimating treatment effects using the front-door criterion. Working Paper.
- **Morgan, S. L., & Winship, C. (2014)**. Counterfactuals and causal inference. Cambridge, England: *Cambridge University Press*.
- **Pearl, J. (2009)**. Causality. Cambridge, England: *Cambridge University Press*.

3.2 Back-door identification

If one or more back-door paths connect the causal variable to the outcome variable, the causal effect is identified by conditioning on a set of variables Z if:

Condition 1 All back-door paths between the causal variable and the outcome variable are blocked after conditioning on Z, which will always be the case if each back-door path

- contains a chain of mediation $A \rightarrow C \rightarrow B$ where the middle variable C is in Z
- contains a fork of mutual dependence $A \leftarrow C \rightarrow B$, where the middle variable C is in Z
- contains an inverted fork of mutual causation $A \rightarrow C \leftarrow B$, where the middle variable C and all of C's decendents are not in Z

and:

Condition 2 No variables in Z are decendents of the causal variable that lies on (or decend from other variables that lie on) any of the directed paths that begin at the causal variable and reach the outcome variable.

3.3 Front-door identification

If one or more unblocked bach-door paths connect a causal variable to an outcome variable, the causal effect is identified by conditioning on a set of observed variables $\{M\}$, that make up an identifying mechanism if

- **Condition 1 (exhaustiveness)** The variables in the set $\{M\}$ intercept all directed paths from the causal variable to the outcome variable.
- **Condition 2 (isolation)** No unblocked back-door paths connect the causal variable to the variables in the set $\{M\}$, and all back-door paths from the variables in the set $\{M\}$ to the outcome variable can be blocked by conditioning on the causal variable.

CHAPTER FOUR

PROJECTS

All information regarding your course project is collected in the OSE course projects documentation.

CHAPTER

FIVE

PARTNERS

Our course equips students with the required skills in statistics, technology, and communication to use data for decision-making. Our partnerships with the private and public sector connect students directly with employment opportunities that match their interests and skill set. All information regarding your partners is collected in the OSE course projects documentation.

CHAPTER

SIX

ORGANIZATION

We start on April 13th 2021 and meet on Tuesdays (14:15-15:45pm) and Wednesdays (10:15-11:45pm).

Lecturer Philipp Eisenhauer

Assistant Carolina Alvarez

We will conduct all course communications using the bonn-econ-teaching Zulip chat, so please be sure to join us there. To join the Zulip organization, please click on the button below.

The student projects are due on the 23rd of July.

6.1 Lecture plan

Date	Topic
13/04/2021	Kickoff, Introduction
14/04/2021	Tools for data science
20/04/2021	Counterfactuals and the potential outcome model
21/04/2021	Counterfactuals and the potential outcome model
27/04/2021	Problem set: Potential outcome model
28/04/2021	Causal graphs
04/05/2021	Causal graphs
05/05/2021	Identification criteria for conditioning estimators
11/05/2021	Matching estimators for causal effects
12/05/2021	Matching estimators for causal effects
18/05/2021	Matching estimators for causal effects
19/05/2021	Dies Academicus, office hours
25/05/2021	Pentecost holidays
26/05/2021	Pentecost holidays
01/06/2021	Problem set: Matching estimators
02/06/2021	Guest lecture: Alexander Sommer (Ernst & Young)
08/06/2021	Regression estimators for causal effects
09/06/2021	Self-selection, heterogeneity, and causal graphs
15/06/2021	Instrumental variable estimators of causal effects
16/06/2021	Instrumental variable estimators of causal effects
22/06/2021	Mechanisms and causal explanations
23/06/2021	Guest lecture: Dr. Nils Wittman (McKinsey & Company)
29/06/2021	Regression discontinuity design

continues on next page

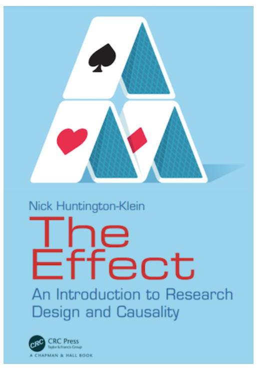
Table 1 – continued from previous page

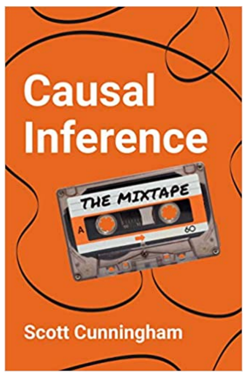
Date	Topic
30/06/2021	Regression discontinuity design
06/07/2021	Guest lecture: Dr. Sebastian Garmann (Bundesrechnungshof)
07/07/2021	Problem set: Regression discontinuity design
13/07/2021	Introduction to structural econometrics
14/07/2021	Guest Lecture: Susane Scholten and Martin Slowik (Deutsche Bank)
20/07/2021	Maximum Likelihood Estimation
21/07/2021	Simulated Methods Moments

CHAPTER

SEVEN

TEXTBOOKS





We use the book The effect: an introduction to research design and causality by Nick Huntington-Klein and Causal inference: the mixtape by Scott Cunningham throughout the course.

CHAPTER

EIGHT

REVIEWS

- **Athey, S., Imbens, G. (2017)**. The state of applied econometrics: causality and policy evaluation , *Journal of Economics Perspectives*, 31(2), 3-32.
- **Abadie, A., Cattaneo, M.D. (2018)**. Econometric methods for program evaluation , *Annual Review of Economics*, 10, 465-503.

CHAPTER NINE

POWERED BY

We gratefully acknowledge funding by the Federal Ministry of Education and Research (BMBF) and the Ministry of Culture and Science of the State of North Rhine-Westphalia (MKW) as part of the Excellence Strategy of the federal and state governments.