Statistical Tools for Causal Inference

Sylvain Chabé-Ferret

2022-01-18

Contents

Introduction

Tools of causal inference are the basic statistical building block behind most scientific results. It is thus extremely useful to have an open source collectively aggreed upon resource presenting and assessing them, as well as listing the current unresolved issues. The content of this book covers the basic theoretical knowledge and technical skills required for implementing staistical methods of causal inference. This means:

- Understanding of the basic language to encode causality,
- Knowledge of the fundamental problems of inference and the biases of intuitive estimators,
- Understanding of how econometric methods recover treatment effects,
- Ability to compute these estimators along with an estimate of their precision using the statistical software R.

This book is geared for teaching causal inference to graduate students that want to apply statistical tools of causal inference. The demonstration of theoretical results are provided, but the final goal is not to have students reproduce them, but mostly to enable them to grasp a better understanding of the fundations for the tools that they will be using. The focus is on understanding the issues and solutions more than understanding the maths that are behind, even though the maths are there and are used to convey the notions rigorously. All the notions and estimators are introduced using a numerical example and simulations, so that each notion is illustrated and appears more intuitive to the students. The second version of this book will contain examples using real applications. The third version will contain exercises.

This book is written in Rmarkdown using the bookdown package. It is available both as a web-book and as a pdf book.

This book is a collaborative effort that is part of the Social Science Knowledge Accumulation Initiative (SKY). The code behind this book is publically available on GitHub and you can propose corrections and updates. How to make contributions to this book is explained on the SKY website. Do not hesitate to make suggestions, modifications and extensions. This way this book will grow and become the living open source collaborative reference for methodological work that it could be.

Part I

Fundamental Problems of Inference

Introduction: the Two Fundamental Problems of Inference

When trying to estimate the effect of a program on an outcome, we face two very important and difficult problems: the Fundamental Problem of Causal Inference (FPCI) and the Fundamental Problem of Statistical Inference (FPSI).

In its most basic form, the FPCI states that our causal parameter of interest (TT, short for Treatment on the Treated, that we will define shortly) is fundamentally unobservable, even when the sample size is infinite. The main reason for that is that one component of TT, the outcome of the treated had they not received the program, remains unobservable. We call this outcome a counterfactual outcome. The FPCI is a very dispiriting result, and is actually the basis for all of the statistical methods of causal inference. All of these methods try to find ways to estimate the counterfactual by using observable quantities that hopefully approximate it as well as possible. Most people, including us but also policymakers, generally rely on intuitive quantities in order to generate the counterfactual (the individuals without the program or the individuals before the program was implemented). Unfortunately, these approximations are generally very crude, and the resulting estimators of TT are generally biased, sometimes severely.

The Fundamental Problem of Statistical Inference (FPSI) states that, even if we have an estimator E that identifies TT in the population, we cannot observe E because we only have access to a finite sample of the population. The only thing that we can form from the sample is a sample equivalent $\hat{E}$ to the population quantity E, and $\hat{E} \neq E$. Why is $\hat{E} \neq E$? Because a finite sample is never perfectly representative of the population. What can we do to deal with the FPSI? I am going to argue that there are mainly two things that we might want to do: estimating the extent of sampling noise and decreasing sampling noise.

Chapter 1

Fundamental Problem of Causal Inference

In order to state the FPCI, we are going to describe the basic language to encode causality set up by Rubin, and named Rubin Causal Model (RCM). RCM being about partly observed random variables, it is hard to make these notions concrete with real data. That's why we are going to use simulations from a simple model in order to make it clear how these variables are generated. The second virtue of this model is that it is going to make it clear the source of selection into the treatment. This is going to be useful when understanding biases of intuitive comparisons, but also to discuss the methods of causal inference. A third virtue of this approach is that it makes clear the connexion between the treatment effects literature and models. Finally, a fourth reason that it is useful is that it is going to give us a source of sampling variation that we are going to use to visualize and explore the properties of our estimators.

I use X_i to denote random variable X all along the notes. I assume that we have access to a sample of N observations indexed by $i \in \{1, \ldots, N\}$. "i" will denote the basic sampling units when we are in a sample, and a basic element of the probability space when we are in populations. Introducing rigorous measure-theoretic notations for the population is feasible but is not necessary for comprehension.

When the sample size is infinite, we say that we have a population. A population is a very useful fiction for two reasons. First, in a population, there is no sampling noise: we observe an infinite amount of observations, and our estimators are infinitely precise. This is useful to study phenomena independently of sampling noise. For example, it is in general easier to prove that an estimator is equal to TT under some conditions in the population. Second, we are most of the time much more interested in estimating the values of parameters in the population rather than in the sample. The population parameter, independent of sampling

noise, gives a much better idea of the causal parameter for the population of interest than the parameter in the sample. In general, the estimator for both quantities will be the same, but the estimators for the effetc of sampling noise on these estimators will differ. Sampling noise for the population parameter will generally be larger, since it is affected by another source of variability (sample choice).

1.1 Rubin Causal Model

RCM is made of three distinct building blocks: a treatment allocation rule, that decides who receives the treatment; potential outcomes, that measure how each individual reacts to the treatment; the switching equation that relates potential outcomes to observed outcomes through the allocation rule.

1.1.1 Treatment allocation rule

The first building block of RCM is the treatment allocation rule. Throughout this class, we are going to be interested in inferring the causal effect of only one treatment with respect to a control condition. Extensions to multi-valued treatments are in general self-explanatory.

In RCM, treatment allocation is captured by the variable D_i. $D_i = 1$ if unit i receives the treatment and $D_i = 0$ if unit i does not receive the treatment and thus remains in the control condition.

The treatment allocation rule is critical for several reasons. First, because it switches the treatment on or off for each unit, it is going to be at the source of the FPCI. Second, the specific properties of the treatment allocatoin rule are going to matter for the feasibility and bias of the various econometric methods that we are going to study.

Let's take a few examples of allocation rules. These allocation rules are just examples. They do not cover the space of all possible allocation rules. They are especially useful as concrete devices to understand the sources of biases and the nature of the allocation rule. In reality, there exists even more complex allocation rules (awareness, eligibility, application, acceptance, active participation). Awareness seems especially important for program participation and has only been tackled recently by economists.

First, some notation. Let's imagine a treatment that is given to individuals. Whether each individual receives the treatment partly depends on the level of her outcome before receiving the treatment. Let's denote this variable Y_i^B, with B standing for "Before". It can be the health status assessed by a professional before deciding to give a drug to a patient. It can be the poverty level of a household used to assess its eligibilty to a cash transfer program.

1.1.1.1 Sharp cutoff rule

The sharp cutoff rule means that everyone below some threshold $\bar{Y}$ is going to receive the treatment. Everyone whose outcome before the treatment lies above $\bar{Y}$ does not receive the treatment. Such rules can be found in reality in a lot of situations. They might be generated by administrative rules. One very simple way to model this rule is as follows:

$$D_i = \mathbb{1}[Y_i^B \leq \bar{Y}], \tag{1.1}$$

where $\mathbb{1}[A]$ is the indicator function, taking value 1 when A is true and 0 otherwise.

Example 1.1 (Sharp cutoff rule)**.** Imagine that $Y_i^B = \exp(y_i^B)$, with $y_i^B = \mu_i + U_i^B$, $\mu_i \sim \mathcal{N}(\bar{\mu}, \sigma_\mu^2)$ and $U_i^B \sim \mathcal{N}(0, \sigma_U^2)$. Now, let's choose some values for these parameters so that we can generate a sample of individuals and allocate the treatment among them. I'm going to switch to R for that.

```
param <- c(8,.5,.28,1500)
names(param) <- c("barmu","sigma2mu","sigma2U","barY")
param
```

```
##    barmu sigma2mu  sigma2U     barY
##     8.00     0.50     0.28  1500.00
```

Now, I have choosen values for the parameters in my model. For example, $\bar{\mu} = 8$ and $\bar{Y} = 1500$. What remains to be done is to generate Y_i^B and then D_i. For this, I have to choose a sample size ($N = 1000$) and then generate the shocks from a normal.

```
# for reproducibility, I choose a seed that will give me the same random sample each time I run t
set.seed(1234)
N <-1000
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
Ds <- ifelse(YB<=param["barY"],1,0)
```

Let's now build a histogram of the data that we have just generated.

```
# building histogram of yB with cutoff point at ybar
# Number of steps
Nsteps.1 <- 15
#step width
step.1 <- (log(param["barY"])-min(yB[Ds==1]))/Nsteps.1
Nsteps.0 <- (-log(param["barY"])+max(yB[Ds==0]))/step.1
breaks <- cumsum(c(min(yB[Ds==1]),c(rep(step.1,Nsteps.1+Nsteps.0+1))))
```

Table 1.1: Treatment allocation with sharp cutoff rule

0	771
1	229

```
hist(yB,breaks=breaks,main="")
abline(v=log(param["barY"]),col="red")
```

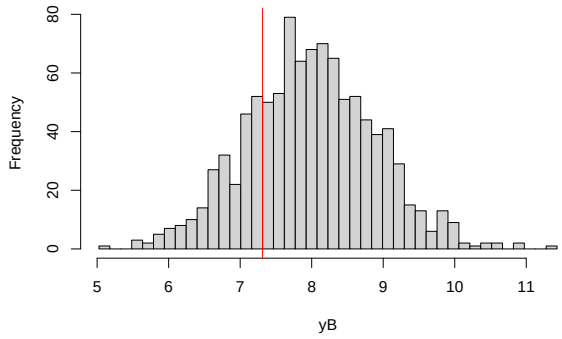


Figure 1.1: Histogram of y_B

You can see on Figure 1.1 a histogram of y_i^B with the red line indicating the cutoff point: $\bar{y} = \ln(\bar{Y}) = 7.3$. All the observations below the red line are treated according to the sharp rule while all the one located above are not. In order to see how many observations eventually receive the treatment with this allocation rule, let's build a contingency table.

```
table.D.sharp <- as.matrix(table(Ds))
knitr::kable(table.D.sharp,caption='Treatment allocation with sharp cutoff rule',bookta
```

We can see on Table 1.1 that there are 229 treated observations.

1.1.1.2 Fuzzy cutoff rule

This rule is less sharp than the sharp cutoff rule. Here, other criteria than Y_i^B enter into the decision to allocate the treatment. The doctor might measure the health status of a patient following official guidelines, but he might also measure other factors that will also influence his decision of giving the drug to the patient. The officials administering a program might measure the official income level of a household, but they might also consider other features of the household situation when deciding to enroll the household into the program or not. If these additional criteria are unobserved to the econometrician, then we have the fuzzy cutoff rule. A very simple way to model this rule is as follows:

$$D_i = \mathbb{1}[Y_i^B + V_i \leq \bar{Y}], \tag{1.2}$$

where V_i is a random variable unobserved to the econometrician and standing for the other influences that might drive the allocation of the treatment. V_i is distributed according to a, for the moment, unspecified cumulative distribution function F_V. When V_i is degenerate (*i.e.* it has only one point of support: it is a constant), the fuzzy cutoff rule becomes the sharp cutoff rule.

1.1.1.3 Eligibility + self-selection rule

It is also possible that households, once they have been made eligible to the treatment, can decide whether they want to receive it or not. A patient might be able to refuse the drug that the doctor suggests she should take. A household might refuse to participate in a cash transfer program to which it has been made eligible. Not all programs have this feature, but most of them have some room for decisions by the agents themselves of whether they want to receive the treatment or not. One simple way to model this rule is as follows:

$$D_i = \mathbb{1}[D_i^* \geq 0]E_i, \tag{1.3}$$

where D_i^* is individual i's valuation of the treatment and E_i is whether or not she is deemed eligible for the treatment. E_i might be choosen according to the sharp cutoff rule of to the fuzzy cutoff rule, or to any other eligibility rule. We will be more explicit about D_i^* in what follows.

SIMULATIONS ARE MISSING FOR THESE LAST TWO RULES

1.1.2 Potential outcomes

The second main building block of RCM are potential outcomes. Let's say that we are interested in the effect of a treatment on an outcome Y. Each unit i can thus be in two potential states: treated or non treated. Before the allocation of the treatment is decided, both of these states are feasible for each unit.

Definition 1.1 (Potential outcomes)**.** For each unit i, we define two potential outcomes:

- Y_i^1: the outcome that unit i is going to have if it receives the treatment,
- Y_i^0: the outcome that unit i is going to have if it **does not** receive the treatment.

Example 1.2. Let's choose functional forms for our potential outcomes. For simplicity, all lower case letters will denote log outcomes. $y_i^0 = \mu_i + \delta + U_i^0$, with δ a time shock common to all the observations and $U_i^0 = \rho U_i^B + \epsilon_i$, with $|\rho| < 1$. In the absence of the treatment, part of the shocks U_i^B that the

individuals experienced in the previous period persist, while some part vanish. $y_i^1 = y_i^0 + \bar{\alpha} + \theta\mu_i + \eta_i$. In order to generate the potential outcomes, one has to define the laws for the shocks and to choose parameter values. Let's assume that $\epsilon_i \sim \mathcal{N}(0, \sigma_\epsilon^2)$ and $\eta_i \sim \mathcal{N}(0, \sigma_\eta^2)$. Now let's choose some parameter values:

```
l <- length(param)
param <- c(param,0.9,0.01,0.05,0.05,0.05,0.1)
names(param)[(l+1):length(param)] <- c("rho","theta","sigma2epsilon","sigma2eta","delta
param
```

```
##         barmu      sigma2mu       sigma2U          barY           rho
##          8.00          0.50          0.28       1500.00          0.90
##         theta sigma2epsilon     sigma2eta         delta      baralpha
##          0.01          0.05          0.05          0.05          0.10
```

We can finally generate the potential outcomes;

```
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
```

Now, I would like to visualize my potential outcomes:

```
plot(y0,y1)
```

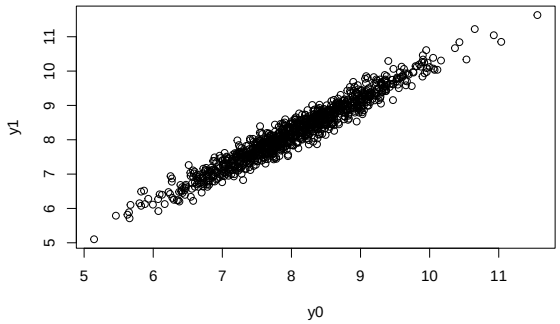


Figure 1.2: Potential outcomes

You can see on the resulting Figure 1.2 that both potential outcomes are positively correlated. Those with a large potential outcome when untreated (*e.g.* in good health without the treatment) also have a positive health with the treatment. It is also true that individuals with bad health in the absence of the treatment also have bad health with the treatment.

1.1.3 Switching equation

The last building block of RCM is the switching equation. It links the observed outcome to the potential outcomes through the allocation rule:

$$\begin{aligned} Y_i &= \begin{cases} Y_i^1 & \text{if } D_i = 1 \\ Y_i^0 & \text{if } D_i = 0 \end{cases} \\ &= Y_i^1 D_i + Y_i^0 (1 - D_i) \end{aligned} \tag{1.4}$$

Example 1.3. In order to generate observed outcomes in our numerical example, we simply have to enforce the switching equation:

```
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
```

What the switching equation (1.4) means is that, for each individual i, we get to observe only one of the two potential outcomes. When individual i belongs to the treatment group (*i.e.* $D_i = 1$), we get to observe Y_i^1. When individual i belongs to the control group (*i.e.* $D_i = 0$), we get to observe Y_i^0. Because the same individual cannot be at the same time in both groups, we can NEVER see both potential outcomes for the same individual at the same time.

For each of the individuals, one of the two potential outcomes is unobserved. We say that it is a **counterfactual**. A counterfactual quantity is a quantity that is, according to Hume's definition, contrary to the observed facts. A counterfactual cannot be observed, but it can be conceived by an effort of reason: it is the consequence of what would have happened had some action not been taken.

Remark. One very nice way of visualising the switching equation has been proposed by Jerzy Neyman in a 1923 prescient paper. Neyman proposes to imagine two urns, each one filled with N balls. One urn is the treatment urn and contains balls with the id of the unit and the value of its potential outcome Y_i^1. The other urn is the control urn, and it contains balls with the value of the potential outcome Y_i^0 for each unit i. Following the allocation rule D_i, we decide whether unit i is in the treatment or control group. When unit i is in the treatment group, we take the corresponding ball from the first urn and observe the potential outcome on it. But, at the same time, the urns are connected so that the corresponding ball with the potential outcome of unit i in the control urn disappears as soon as we draw ball i from the treatment urn.

The switching equation works a lot like Schrodinger's cat paradox. Schrodinger's cat is placed in a sealed box and receives a dose of poison when an atom emits a radiation. As long as the box is sealed, there is no way we can know whether the cat is dead or alive. When we open the box, we observe either a dead cat or a living cat, but we cannot observe the cat both alive and dead at the same time. The switching equation is like opening the box, it collapses the observed outcome into one of the two potential ones.

Example 1.4. One way to visualize the inner workings of the switching equation is to plot the potential outcomes along with the criteria driving the allocation rule. In our simple example, it simply amounts to plotting observed (y_i) and potential outcomes (y_i^1 and y_i^0) along y_i^B.

```
plot(yB[Ds==0],y0[Ds==0],pch=1,xlim=c(5,11),ylim=c(5,11),xlab="yB",ylab="Outcomes")
points(yB[Ds==1],y1[Ds==1],pch=3)
points(yB[Ds==0],y1[Ds==0],pch=3,col='red')
points(yB[Ds==1],y0[Ds==1],pch=1,col='red')
test <- 5.8
i.test <- which(abs(yB-test)==min(abs(yB-test)))
points(yB[abs(yB-test)==min(abs(yB-test))],y1[abs(yB-test)==min(abs(yB-test))],col='gre
points(yB[abs(yB-test)==min(abs(yB-test))],y0[abs(yB-test)==min(abs(yB-test))],col='gre
abline(v=log(param["barY"]),col="red")
legend(5,11,c('y0|D=0','y1|D=1','y0|D=1','y1|D=0',paste('y0',i.test,sep=''),paste('y1'
```

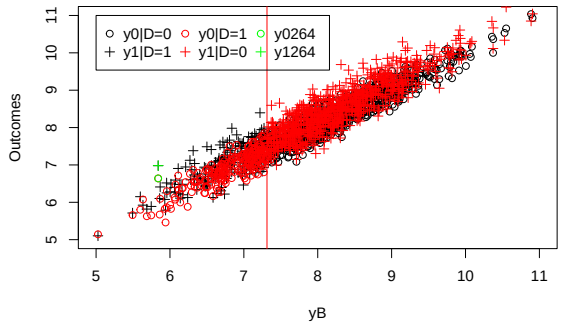


Figure 1.3: Potential outcomes

```
plot(yB[Ds==0],y0[Ds==0],pch=1,xlim=c(5,11),ylim=c(5,11),xlab="yB",ylab="Outcomes")
points(yB[Ds==1],y1[Ds==1],pch=3)
legend(5,11,c('y|D=0','y|D=1'),pch=c(1,3))
abline(v=log(param["barY"]),col="red")
```

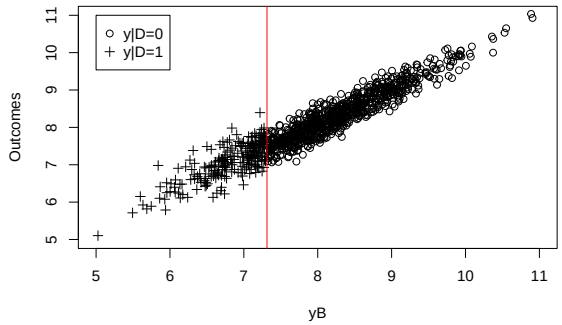


Figure 1.4: Observed outcomes

Figure 1.3 plots the observed outcomes y_i along with the unobserved potential outcomes. Figure 1.3 shows that each individual in the sample is endowed with two potential outcomes, represented by a circle and a cross. Figure 1.4 plots the observed outcomes y_i that results from applying the switching equation. Only one of the two potential outcomes is observed (the cross for the treated group and the circle for the untreated group) and the other is not. The observed sample in Figure 1.4 only shows observed outcomes, and is thus silent on the values of the missing potential outcomes.

1.2 Treatment effects

RCM enables the definition of causal effects at the individual level. In practice though, we generally focus on a summary measure: the effect of the treatment on the treated.

1.2.1 Individual level treatment effects

Potential outcomes enable us to define the central notion of causal inference: the causal effect, also labelled the treatment effect, which is the difference between the two potential outcomes.

Definition 1.2 (Individual level treatment effect)**.** For each unit i, the causal effect of the treatment on outcome Y is: $\Delta_i^Y = Y_i^1 - Y_i^0$.

Example 1.5. The individual level causal effect in log terms is: $\Delta_i^y = \alpha_i = \bar{\alpha} + \theta\mu_i + \eta_i$. The effect is the sum of a part common to all individuals, a part correlated with μ_i: the treatment might have a larger or a smaller effect depending on the unobserved permanent ability or health status of individuals, and a random shock. It is possible to make the effect of the treatment to depend on U_i^B also, but it would complicate the model.

In Figure 1.3, the individual level treatment effects are the differences between each cross and its corresponding circle. For example, for observation 264, the two potential outcomes appear in green in Figure 1.3. The effect of the treatment on unit 264 is equal to:

$$\Delta_{264}^y = y_{264}^1 - y_{264}^0 = 6.98 - 6.64 = 0.34.$$

Since observation 264 belongs to the treatment group, we can only observe the potential outcome in the presence of the treatment, y_{264}^1.

RCM allows for heterogeneity of treatment effects. The treatment has a large effect on some units and a much smaller effect on other units. We can even have some units that benefit from the treatment and some units that are harmed by the treatment. The individual level effect of the treatment is itself a random variable (and not a fixed parameter). It has a distribution, F_{Δ^Y}.

Heterogeneity of treatment effects seems very natural: the treatment might interact with individuals' different backgrounds. The effect of a drug might depend on the genetic background of an individual. An education program might only work for children that already have sufficient non-cognitive skills, and thus might depend in turn on family background. An environmental regulation or a behavioral intervention might only trigger reactions by already environmentally aware individuals. A CCT might have a larger effect when indiviuals are credit-constrained or face shocks.

Example 1.6. In our numerical example, the distribution of $\Delta_i^y = \alpha_i$ is a normal: $\alpha_i \sim \mathcal{N}(\bar{\alpha} + \theta\bar{\mu}, \theta^2\sigma_\mu^2 + \sigma_\eta^2)$. We would like to visualize treatment effect heterogeneity. For that, we can build a histogram of the individual level causal effect.

On top of the histogram, we can also draw the theoretical distribution of the treatment effect: a normal with mean 0.18 and variance 0.05.

```
hist(alpha,main="",prob=TRUE)
curve(dnorm(x, mean=(param["baralpha"]+param["theta"]*param["barmu"]), sd=sqrt(param["
```

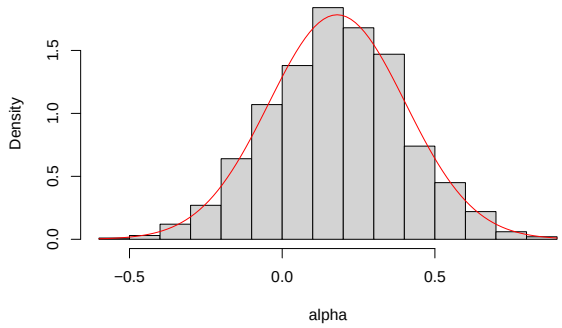


Figure 1.5: Histogram of Δ^y

The first thing that we can see on Figure 1.5 is that the theoretical and the empirical distributions nicely align with each other. We also see that the majority of the observations lies to the right of zero: most people experience a positive effect of the treatment. But there are some individuals that do not benefit from the treatment: the effect of the treatment on them is negative.

1.2.2 Average treatment effect on the treated

We do not generally estimate individual-level treatment effects. We generally look for summary statistics of the effect of the treatment. By far the most widely reported causal parameter is the Treatment on the Treated parameter (TT). It can be defined in the sample at hand or in the population.

Definition 1.3 (Average and expected treatment effects on the treated)**.** The Treatment on the Treated parameters for outcome Y are:

- The average Treatment effect on the Treated in the sample:

$$\Delta^Y_{TT_s} = \frac{1}{\sum_{i=1}^N D_i} \sum_{i=1}^N (Y_i^1 - Y_i^0) D_i,$$

- The expected Treatment effect on the Treated in the population:

$$\Delta^Y_{TT} = \mathbb{E}[Y_i^1 - Y_i^0 | D_i = 1].$$

The TT parameters measure the average effect of the treatment on those who actually take it, either in the sample at hand or in the popluation. It is generally considered to be the most policy-relevant parameter since it measures the effect of the treatment as it has actually been allocated. For example, the expected causal effect on the overall population is only relevant if policymakers are considering implementing the treatment even on those who have not been selected to receive it. For a drug or an anti-poverty program, it would mean giving the treatment to healthy or rich people, which would make little sense.

TT does not say anything about how the effect of the treatment is distributed in the population or in the sample. TT does not account for the heterogneity of treatment effects. In Lecture 7, we will look at other parameters of interest that look more closely into how the effect of the treatment is distributed.

Example 1.7. The value of TT in our sample is:

$$\Delta^y_{TT_s} = 0.168.$$

Computing the population value of TT is slightly more involved: we have to use the formula for the conditional expectation of a censored bivariate normal random variable:

$$\begin{aligned}
\Delta^y_{TT} &= \mathbb{E}[\alpha_i | D_i = 1] \\
&= \bar{\alpha} + \theta \mathbb{E}[\mu_i | \mu_i + U_i^B \leq \bar{y}] \\
&= \bar{\alpha} + \theta \left(\bar{\mu} - \frac{\sigma_\mu^2}{\sqrt{\sigma_\mu^2 + \sigma_U^2}} \frac{\phi\left(\frac{\bar{y} - \bar{\mu}}{\sqrt{\sigma_\mu^2 + \sigma_U^2}} \right)}{\Phi\left(\frac{\bar{y} - \bar{\mu}}{\sqrt{\sigma_\mu^2 + \sigma_U^2}} \right)} \right) \\
&= \bar{\alpha} + \theta \bar{\mu} - \theta \left(\frac{\sigma_\mu^2}{\sqrt{\sigma_\mu^2 + \sigma_U^2}} \frac{\phi\left(\frac{\bar{y} - \bar{\mu}}{\sqrt{\sigma_\mu^2 + \sigma_U^2}} \right)}{\Phi\left(\frac{\bar{y} - \bar{\mu}}{\sqrt{\sigma_\mu^2 + \sigma_U^2}} \right)} \right),
\end{aligned}$$

where ϕ and Φ are respectively the density and the cumulative distribution functions of the standard normal. The second equality follows from the definition of α_i and D_i and from the fact that η_i is independent from μ_i and U_i^B. The third equality comes from the formula for the expectation of a censored bivariate normal random variable. In order to compute the population value of TT easily for different sets of parameter values, let's write a function in R:

```
delta.y.tt <- function(param){return(param["baralpha"]+param["theta"]*param["barmu"]
                                     -param["theta"]*((param["sigma2mu"]*dnorm((log(pa
                                                      /(sqrt(param["sigma2mu"]+param["
                                                        *pnorm((log(param["barY"])-para
```

The population value of TT computed using this function is: $\Delta^y_{TT} = 0.172$. We can see that the values of TT in the sample and in the population differ slightly. This is because of sampling noise: the units in the sample are not perfectly representative of the units in the population.

1.3 Fundamental problem of causal inference

At least in this lecture, causal inference is about trying to infer TT, either in the sample or in the population. The FPCI states that it is impossible to directly observe TT because one part of it remains fundamentally unobserved.

Theorem 1.1 (Fundamental problem of causal inference)**.** *It is impossible to observe TT, either in the population or in the sample.*

Proof. The proof of the FPCI is rather straightforward. Let me start with the sample TT:

$$\begin{aligned}
\Delta^Y_{TT_s} &= \frac{1}{\sum_{i=1}^N D_i} \sum_{i=1}^N (Y_i^1 - Y_i^0) D_i \\
&= \frac{1}{\sum_{i=1}^N D_i} \sum_{i=1}^N Y_i^1 D_i - \frac{1}{\sum_{i=1}^N D_i} \sum_{i=1}^N Y_i^0 D_i \\
&= \frac{1}{\sum_{i=1}^N D_i} \sum_{i=1}^N Y_i D_i - \frac{1}{\sum_{i=1}^N D_i} \sum_{i=1}^N Y_i^0 D_i.
\end{aligned}$$

Since Y_i^0 is unobserved whenever $D_i = 1$, $\frac{1}{\sum_{i=1}^N D_i} \sum_{i=1}^N Y_i^0 D_i$ is unobserved, and so is $\Delta^Y_{TT_s}$. The same is true for the population TT:

$$\begin{aligned}\Delta^Y_{TT} &= \mathbb{E}[Y_i^1 - Y_i^0 | D_i = 1] \\ &= \mathbb{E}[Y_i^1 | D_i = 1] - \mathbb{E}[Y_i^0 | D_i = 1] \\ &= \mathbb{E}[Y_i | D_i = 1] - \mathbb{E}[Y_i^0 | D_i = 1].\end{aligned}$$

$\mathbb{E}[Y_i^0 | D_i = 1]$ is unobserved, and so is Δ^Y_{TT}. □

The key insight in order to understand the FPCI is to see that the outcomes of the treated units had they not been treated are unobservable, and so is their average or expectation. We say that they are counterfactual, contrary to what has happened.

Definition 1.4 (Couterfactual)**.** Both $\frac{1}{\sum_{i=1}^N D_i} \sum_{i=1}^N Y_i^0 D_i$ and $\mathbb{E}[Y_i^0 | D_i = 1]$ are counterfactual quantities that we will never get to observe.

Example 1.8. The average counterfactual outcome of the treated is the mean of the red circles in the y axis on Figure 1.3:

$$\frac{1}{\sum_{i=1}^N D_i} \sum_{i=1}^N y_i^0 D_i = 6.91.$$

Remember that we can estimate this quantity only because we have generated the data ourselves. In real life, this quantity is hopelessly unobserved.

$\mathbb{E}[y_i^0 | D_i = 1]$ can be computed using the formula for the expectation of a censored normal random variable:

$$\begin{aligned}\mathbb{E}[y_i^0 | D_i = 1] &= \mathbb{E}[\mu_i + \delta + U_i^0 | D_i = 1] \\ &= \mathbb{E}[\mu_i + \delta + \rho U_i^B + \epsilon_i | D_i = 1] \\ &= \delta + \mathbb{E}[\mu_i + \rho U_i^B | y_i^B \leq \bar{y}] \\ &= \delta + \bar{\mu} - \frac{\sigma_\mu^2 + \rho\sigma_U^2}{\sqrt{\sigma_\mu^2 + \sigma_U^2}} \frac{\phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}{\Phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}.\end{aligned}$$

We can write a function in R to compute this value:

```
esp.y0.D1 <- function(param){
  return(param["delta"]+param["barmu"]
         -((param["sigma2mu"]+param["rho"]*param["sigma2U"])
           *dnorm((log(param["barY"])-param["barmu"])/(sqrt(param["sigma2mu"]+param["sigma2U"])))
         /(sqrt(param["sigma2mu"]+param["sigma2U"])*pnorm((log(param["barY"])-param["barmu"])
```

```
                                                    /(sqrt(param["sigma2mu"]+para
}
```

The population value of TT computed using this function is: $\mathbb{E}[y_i^0|D_i = 1] = 6.9$.

1.4 Intuitive estimators, confounding factors and selection bias

In this section, we are going to examine the properties of two intuitive comparisons that laypeople, policymakers but also ourselves make in order to estimate causal effects: the with/wihtout comparison (WW) and the before/after comparison (BA). WW compares the average outcomes of the treated individuals with those of the untreated individuals. BA compares the average outcomes of the treated after taking the treatment to their average outcomes before they took the treatment. These comparisons try to proxy for the expected counterfactual outcome in the treated group by using an observed quantity. WW uses the expected outcome of the untreated individuals as a proxy. BA uses the expected outcome of the treated before they take the treatment as a proxy.

Unfortunately, both of these proxies are generally poor and provide biased estimates of TT. The reason that these proxies are poor is that the treatment is not the only factor that differentiates the treated group from the groups used to form the proxy. The intuitive comparisons are biased because factors, other than the treatment, are correlated to its allocation. The factors that bias the intuitive comparisons are generally called confouding factors or confounders.

The treatment effect measures the effect of a ceteris paribus change in treatment status, while the intuitive comparisons capture both the effect of this change and that of other correlated changes that spuriously contaminate the comparison. Intuitive comparisons measure correlations while treatment effects measure causality. The old motto "correlation is not causation" applies vehemently here.

Remark. A funny anecdote about this expression "correlation is not causation". This expression is due to Karl Pearson, the father of modern statistics. He coined the phrase in his famous book "The Grammar of Science." Pearson is famous for inventing the correlation coefficient. He actually thought that correlation was a much superior, much more rigorous term, than causation. In his book, he actually used the sentence to argue in favor of abandoning causation altogether and focusing on the much better-defined and measurable concept of correlation. Interesting turn of events that his sentence is now used to mean that correlation is weaker than causation, totally reverting the original intended meaning.

In this section, we are going to define both comparisons, study their biases and state the conditions under which they identify TT. This will prove to be a very useful introduction to the notion of identification. It is also very important to be able to understand the sources of bias of comparisons that we use every day

and that come very naturally to policy makers and lay people.

Remark. In this section, we state the definitions and formulae in the population. This is for two reasons. First, it is simpler, and lighter in terms of notation. Second, it emphasizes that the problems with intuitive comparisons are independent of sampling noise. Most of the results stated here for the population extend to the sample, replacing the expectation operator by the average operator. I will nevertheless give examples in the sample, since it is so much simpler to compute. I will denote sample equivalents of population estimators with a hat.

1.4.1 With/Without comparison, selection bias and cross-sectional confounders

The with/without comparison (WW) is very intuitive: just compare the outcomes of the treated and untreated individuals in order to estimate the causal effect. This approach is nevertheless generally biased. We call the bias of WW selection bias (SB). Selection bias is due to unobserved confounders that are distributed differently in the treatment and control group and that generate differences in outcomes even in the absence of the treatment. In this section, I define the WW estimator, derives its bias, introduces the confounders and states conditions under which it is unbiased.

1.4.1.1 With/Without comparison

The with/without comparison (WW) is very intuitive: just compare the outcomes of the treated and untreated individuals in order to estimate the causal effect.

Definition 1.5 (With/without comparison)**.** The with/without comparison is the difference between the expected outcomes of the treated and the expected outcomes of the untreated:

$$\Delta^Y_{WW} = \mathbb{E}[Y_i|D_i = 1] - \mathbb{E}[Y_i|D_i = 0].$$

Example 1.9. In the population, WW can be computed using the traditional formula for the expectation of a truncated normal distribution:

$$\begin{aligned}
\Delta^y_{WW} &= \mathbb{E}[y_i|D_i = 1] - \mathbb{E}[y_i|D_i = 0] \\
&= \mathbb{E}[y^1_i|D_i = 1] - \mathbb{E}[y^0_i|D_i = 0] \\
&= \mathbb{E}[\alpha_i|D_i = 1] + \mathbb{E}[\mu_i + \rho U^B_i|\mu_i + U^B_i \leq \bar{y}] - \mathbb{E}[\mu_i + \rho U^B_i|\mu_i + U^B_i > \bar{y}] \\
&= \bar{\alpha} + \theta \left(\bar{\mu} - \frac{\sigma^2_\mu}{\sqrt{\sigma^2_\mu + \sigma^2_U}} \frac{\phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma^2_\mu+\sigma^2_U}}\right)}{\Phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma^2_\mu+\sigma^2_U}}\right)} \right) - \frac{\sigma^2_\mu + \rho\sigma^2_U}{\sqrt{\sigma^2_\mu + \sigma^2_U}} \left(\frac{\phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma^2_\mu+\sigma^2_U}}\right)}{\Phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma^2_\mu+\sigma^2_U}}\right)} + \frac{\phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma^2_\mu+\sigma^2_U}}\right)}{1-\Phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma^2_\mu+\sigma^2_U}}\right)} \right).
\end{aligned}$$

In order to compute this parameter, we are going to set up a R function. For reasons that will become clearer later, we will define two separate functions to compute the first and second part of the formula. In the first part, you should have recognised TT, that we have already computed in Lecture 1. We are going to call the second part SB, for reasons that will become explicit in a bit.

```
delta.y.tt <- function(param){
  return(param["baralpha"]+param["theta"]*param["barmu"]-param["theta"]
         *((param["sigma2mu"]*dnorm((log(param["barY"])-param["barmu"])
                                    /(sqrt(param["sigma2mu"]+param["sigma2U"])))))
           /(sqrt(param["sigma2mu"]+param["sigma2U"])*pnorm((log(param["barY"])-param[
                                                            /(sqrt(param["sigma2mu"]+pa
}
delta.y.sb <- function(param){
  return(-(param["sigma2mu"]+param["rho"]*param["sigma2U"])/sqrt(param["sigma2mu"]+para
         *dnorm((log(param["barY"])-param["barmu"])/(sqrt(param["sigma2mu"]+param["sigm
         *(1/pnorm((log(param["barY"])-param["barmu"])/(sqrt(param["sigma2mu"]+param["s
           +1/(1-pnorm((log(param["barY"])-param["barmu"])/(sqrt(param["sigma2mu"]+para
}
delta.y.ww <- function(param){
  return(delta.y.tt(param)+delta.y.sb(param))
}
```

As a conclusion of all these derivations, WW in the population is equal to -1.298. Remember that the value of TT in the population is 0.172.

In order to compute the WW estimator in a sample, I'm going to generate a brand new sample and I'm going to choose a seed for the pseudo-random number generator so that we obtain the same result each time we run the code. I use `set.seed(1234)` in the code chunk below.

```
param <- c(8,.5,.28,1500)
names(param) <- c("barmu","sigma2mu","sigma2U","barY")
set.seed(1234)
N <-1000
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
Ds <- rep(0,N)
Ds[YB<=param["barY"]] <- 1
l <- length(param)
param <- c(param,0.9,0.01,0.05,0.05,0.05,0.1)
names(param)[(l+1):length(param)] <- c("rho","theta","sigma2epsilon","sigma2eta","delta
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
U0 <- param["rho"]*UB + epsilon
```

```
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
```

In this sample, the average outcome of the treated in the presence of the treatment is

$$\frac{1}{\sum_{i=1}^{N} D_i} \sum_{i=1}^{N} D_i y_i = 7.074.$$

It is materialized by a circle on Figure 1.6. The average outcome of the untreated is

$$\frac{1}{\sum_{i=1}^{N} (1 - D_i)} \sum_{i=1}^{N} (1 - D_i) y_i = 8.383.$$

It is materialized by a plus sign on Figure 1.6.

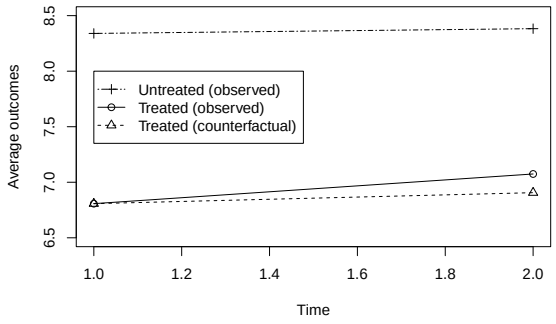


Figure 1.6: Evolution of average outcomes in the treated and control group before (Time =1) and after (Time=2) the treatment

The estimate of the WW comparison in the sample is thus:

$$\hat{\Delta^{Y}_{WW}} = \frac{1}{\sum_{i=1}^{N} D_i} \sum_{i=1}^{N} Y_i D_i - \frac{1}{\sum_{i=1}^{N} (1 - D_i)} \sum_{i=1}^{N} Y_i (1 - D_i).$$

We have $\hat{\Delta^{y}_{WW}}$ = -1.308. Remember that the value of TT in the sample is $\Delta^{y}_{TT_s} = 0.168$.

Overall, WW severely underestimates the effect of the treatment in our example. WW suggests that the treatment has a negative effect on outcomes whereas we know by construction that it has a positive one.

1.4.1.2 Selection bias

When we form the with/without comparison, we do not recover the TT parameter. Instead, we recover TT plus a bias term, called **selection bias**:

$$\Delta^Y_{WW} = \Delta^Y_{TT} + \Delta^Y_{SB}.$$

Definition 1.6 (Selection bias)**.** Selection bias is the difference between the with/without comparison and the treatment on the treated parameter:

$$\Delta^Y_{SB} = \Delta^Y_{WW} - \Delta^Y_{TT}.$$

WW tries to approximate the counterfactual expected outcome in the treated group by using $\mathbb{E}[Y^0_i|D_i = 0]$, the expected outcome in the untreated group . Selection bias appears because this proxy is generally poor. It is very easy to see that selection bias is indeed directly due to this bad proxy problem:

Theorem 1.2 (Selection bias and counterfactual)**.** *Selection bias is the difference between the counterfactual expected potential outcome in the absence of the treatment among the treated and the expected potential outcome in the absence of the treatment among the untreated.*

$$\Delta^Y_{SB} = \mathbb{E}[Y^0_i|D_i = 1] - \mathbb{E}[Y^0_i|D_i = 0].$$

Proof.

$$\begin{aligned}\Delta^Y_{SB} &= \Delta^Y_{WW} - \Delta^Y_{TT} \\ &= \mathbb{E}[Y_i|D_i = 1] - \mathbb{E}[Y_i|D_i = 0] - \mathbb{E}[Y^1_i - Y^0_i|D_i = 1] \\ &= \mathbb{E}[Y^0_i|D_i = 1] - \mathbb{E}[Y^0_i|D_i = 0].\end{aligned}$$

The first and second equalities stem only from the definition of both parameters. The third equality stems from using the switching equation: $Y_i = Y^1_i D_i + Y^0_i(1 - D_i)$, so that $\mathbb{E}[Y_i|D_i = 1] = \mathbb{E}[Y^1_i|D_i = 1]$ and $\mathbb{E}[Y_i|D_i = 0] = \mathbb{E}[Y^0_i|D_i = 0]$. □

Example 1.10. In the population, SB is equal to

$$\begin{aligned}\Delta^y_{SB} &= \Delta^y_{WW} - \Delta^y_{TT} \\ &= -1.298 - 0.172 \\ &= -1.471\end{aligned}$$

We could have computed SB directly using the formula from Theorem 1.2:

$$\begin{aligned}\Delta^y_{SB} &= \mathbb{E}[y_i^0|D_i=1] - \mathbb{E}[y_i^0|D_i=0] \\ &= -\frac{\sigma_\mu^2 + \rho\sigma_U^2}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\left(\frac{\phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}{\Phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)} + \frac{\phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}{1-\Phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}\right).\end{aligned}$$

When using the R function for SB that we have defined earlier, we indeed find: Δ^y_{SB} = -1.471.

In the sample, $\Delta^{\hat{y}}_{SB}$ =-1.308-0.168 = -1.476. Selection bias emerges because we are using a bad proxy for the counterfactual. The average outcome for the untreated is equal to $\frac{1}{\sum_{i=1}^N (1-D_i)}\sum_{i=1}^N (1-D_i)y_i = 8.383$ while the counterfactual average outcome for the treated is $\frac{1}{\sum_{i=1}^N D_i}\sum_{i=1}^N D_i y_i^0 = 6.906$. Their difference is as expected equal to SB: $\Delta^{\hat{y}}_{SB} = 6.906 - 8.383$ = -1.476. The counterfactual average outcome of the treated is much smaller than the average outcome of the untreated. On Figure 1.6, this is materialized by the fact that the plus sign is located much above the triangle.

Remark. The concept of selection bias is related to but different from the concept of sample selection bias. With sample selection bias, we worry that selection into the sample might bias the estimated effect of a treatment on outcomes. With selection bias, we worry that selection into the treatment itself might bias the effect of the treatment on outcomes. Both biases are due to unbserved covariates, but they do not play out in the same way.

For example, estimating the effect of education on women's wages raises both selection bias and sample selection bias issues. Selection bias stems from the fact that more educated women are more likely to be more dynamic and thus to have higher earnings even when less educated. Selection bias would be positive in that case, overestimating the effect of education on earnings.

Sample selection bias stems from the fact that we can only use a sample of working women in order to estimate the effect of education on wages, since we do not observe the wages on non working women. But, selection into the labor force might generate sample selection bias. More educated women participate more in the labor market, while less educated women participate less. As a consequence, less educated women that work are different from the overall sample of less educated women. They might be more dynamic and work-focused. As a consequence, their wages are higher than the average wages of the less educated women. Comparing the wages of less educated women that work to those of more educated women that work might understate the effect of education on earnings. Sample selection bias would generate a negative bias on the education coefficient.

1.4.1.3 Confounding factors

Confounding factors are the factors that generate differences between treated and untreated individuals even in the absence of the treatment. The confounding factors are thus responsible for selection bias. In general, the mere fact of being selected for receiving the treatment means that you have a host of characteristics that would differentiate you from the unselected individuals, even if you were not to receive the treatment eventually.

For example, if a drug is given to initially sicker individuals, then, we expect that they will be sicker that the untreated in the absence of the treatment. Comparing sick individuals to healthy ones is not a sound way to estimate the effect of a treatment. Obviously, even if our treatment performs well, healthier individuals will be healthier after the treatment has been allocated to the sicker patients. The best we can expect is that the treated patients have recovered, and that their health after the treatment is comparable to that of the untreated patients. In that case, the with/without comparison is going to be null, whereas the true effect of the treatment is positive. Selection bias is negative in that case: in the absence of the treatment, the average health status of the treated individuals would have been smaller than that of the untreated individuals. The confounding factor is the health status of individuals when the decision to allocate the drug has been taken. It is correlated to both the allocation of the treatment (negatively) and to health in the absence of the treatment (positively).

Example 1.11. In our example, μ_i and U_i^B are the confounding factors. Because the treatment is only given to individuals with pre-treament outcomes smaller than a threshold ($y_i^B \leq \bar{y}$), participants tend to have smaller μ_i and U_i^B than non participants, as we can see on Figure 1.7.

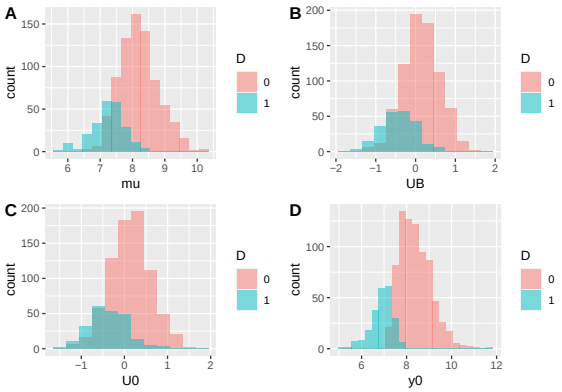


Figure 1.7: Distribution of confounders in the treated and control group

Since confounding factors are persistent, they affect the outcomes of participants and non participants after the treatment date. μ_i persists entirely over time, and U_i^B persists at a rate ρ. As a consequence, even in the absence of the treatment,

participants have lower outcomes than non participants, as we can see on Figure 1.7.

We can derive the contributions of both confouding factors to overall SB:

$$
\begin{aligned}
\mathbb{E}[Y_i^0|D_i=1] &= \mathbb{E}[\mu_i+\delta+U_i^0|\mu_i+U_i^B\leq\bar{y}] \\
&= \delta+\mathbb{E}[\mu_i|\mu_i+U_i^B\leq\bar{y}]+\rho\mathbb{E}[U_i^B|\mu_i+U_i^B\leq\bar{y}] \\
\Delta_{SB}^y &= \mathbb{E}[\mu_i|\mu_i+U_i^B\leq\bar{y}]-\mathbb{E}[\mu_i|\mu_i+U_i^B>\bar{y}] \\
&\quad+\rho\left(\mathbb{E}[U_i^B|\mu_i+U_i^B\leq\bar{y}]-\mathbb{E}[U_i^B|\mu_i+U_i^B>\bar{y}]\right) \\
&= -\frac{\sigma_\mu^2}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\left(\frac{\phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}{\Phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}+\frac{\phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}{1-\Phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}\right) \\
&\quad-\frac{\rho\sigma_U^2}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\left(\frac{\phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}{\Phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}+\frac{\phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}{1-\Phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}\right)
\end{aligned}
$$

In order to evaluate these quantities, let's build two R functions:

```
delta.y.sb.mu <- function(param){
  return(-(param["sigma2mu"])/sqrt(param["sigma2mu"]+param["sigma2U"])
         *dnorm((log(param["barY"])-param["barmu"])/(sqrt(param["sigma2mu"]+param["sigma2U"])))
         *(1/pnorm((log(param["barY"])-param["barmu"])/(sqrt(param["sigma2mu"]+param["sigma2U"])))
           +1/(1-pnorm((log(param["barY"])-param["barmu"])/(sqrt(param["sigma2mu"]+param["sigma2U
}
delta.y.sb.U <- function(param){
  return(-(param["rho"]*param["sigma2U"])/sqrt(param["sigma2mu"]+param["sigma2U"])
         *dnorm((log(param["barY"])-param["barmu"])/(sqrt(param["sigma2mu"]+param["sigma2U"])))
         *(1/pnorm((log(param["barY"])-param["barmu"])/(sqrt(param["sigma2mu"]+param["sigma2U"])))
           +1/(1-pnorm((log(param["barY"])-param["barmu"])/(sqrt(param["sigma2mu"]+param["sigma2U
}
```

The contribution of μ_i to selection bias is -0.978 while that of U_i^0 is of -0.493.

1.4.1.4 When does WW identify TT?

Are there conditions under which WW identify TT? The answer is yes: when there is no selection bias, the proxy used by WW for the counterfactual quantity is actually valid. Formally, WW identifies TT when the following assumption holds:

Definition 1.7 (No selection bias)**.** We assume the following:

$$\mathbb{E}[Y_i^0|D_i=1]=\mathbb{E}[Y_i^0|D_i=0].$$

Under Assumption 1.7, the expected counterfactual outcome of the treated is equal to the expected potential outcome of the untreated in the absence of the treatment. This yields to the following result:

Theorem 1.3. *Under Assumption 1.7, WW identifies the TT parameter:*

$$\Delta^Y_{WW} = \Delta^Y_{TT}.$$

Proof.

$$\begin{aligned}\Delta^Y_{WW} &= \mathbb{E}[Y_i|D_i = 1] - \mathbb{E}[Y_i|D_i = 0] \\ &= \mathbb{E}[Y_i^1|D_i = 1] - \mathbb{E}[Y_i^0|D_i = 0] \\ &= \mathbb{E}[Y_i^1|D_i = 1] - \mathbb{E}[Y_i^0|D_i = 1] \\ &= \Delta^Y_{TT},\end{aligned}$$

where the second equation uses the switching equation and the third uses Assumption 1.7. □

So, under Assumption 1.7, the WW comparison actually identifies the TT parameter. We say that Assumption 1.7 is an **identification assumption**: it serves to identify the parameter of interest using observed data. The intuition for this result is simply that, under Assumption 1.7, there are no confounding factors and thus no selection bias. under Assumption 1.7, the factors that yield individuals to receive or not the treatment are mean-independent of the potential outcomes in the absence of the treatment. In this case, the expected outcome in the untreated group actually is a perfect proxy for the counterfactual expected outcome of the treated group.

Obviously, Assumption 1.7 is extremely unlikely to hold in real life. For Assumption 1.7 to hold, it has to be that **all** the determinants of D_i are actually unrelated to Y_i^0. One way to enforce Assumption 1.7 is to randomize treatment intake. We will see this in the Lecture on RCTs. It might also be possible that Assumption 1.7 holds in the data in the absence of an RCT. But this is not very likely, and should be checked by every mean possible.

One way to test for the validity of Assumption 1.7 is to compare the values of observed covariates in the treated and untreated group. For Assumption 1.7 to be credible, observed covariates should be distributed in the same way.

Another nice way to test for the validity of Assumption 1.7 with observed data is to implement a **placebo test**. A placebo test looks for an effect where there should be none, if we believe the identification assumptions. For example, under Assumption 1.7 it should be (even though it is not rigorously implied) that outcomes before the treatment are also mean-independent of the treatment allocation. And actually, since a future treatment cannot have an effect today (unless people anticipate the treatment, which we assume away here), the WW comparison before the treatment should be null, therefore giving a zero effect of the placebo treatment "will receive the treatment in the future."

Example 1.12. When the allocation rule defining D_i is the eligibility rule that we have used so far, we have already seen that Assumption 1.7 does not hold and the placebo test should not pass either.

One way of generating Assumption 1.7 from the eligibility rule that we are using is to mute the persistence in outcome dynamics. For example, one could set $\rho = 0$ and $\sigma^2_\mu = 0$.

```
param <- c(8,0,.28,1500,0,0.01,0.05,0.05,0.05,0.1)
names(param) <- c("barmu","sigma2mu","sigma2U","barY","rho","theta","sigma2epsilon","sigma2eta","
param
```

```
##         barmu      sigma2mu       sigma2U          barY           rho
##          8.00          0.00          0.28       1500.00          0.00
##         theta sigma2epsilon     sigma2eta         delta      baralpha
##          0.01          0.05          0.05          0.05          0.10
```

In that case, outcomes are not persistent and Assumption 1.7 holds:

$$\begin{aligned}
\mathbb{E}[y_i^0|D_i = 1] &= \mathbb{E}[\mu_i + \delta + U_i^0|y_i^B \leq \bar{y}] \\
&= \mathbb{E}[\bar{\mu} + \delta + \epsilon_i|\bar{\mu} + U_i^B \leq \bar{y}] \\
&= \bar{\mu} + \delta + \mathbb{E}[\epsilon_i|\bar{\mu} + U_i^B \leq \bar{y}] \\
&= \bar{\mu} + \delta + \mathbb{E}[\epsilon_i|\bar{\mu} + U_i^B > \bar{y}] \\
&= \mathbb{E}[\mu_i + \delta + U_i^0|y_i^B > \bar{y}] \\
&= \mathbb{E}[y_i^0|D_i = 0],
\end{aligned}$$

where the second equality follows from $\sigma^2_\mu = 0$ and $\rho = 0$ and the fourth from $\epsilon_i \perp\!\!\!\perp U_i^B$. Another direct way to see this is to use the formula for selection bias that we have derived above. It is easy to see that with $\rho = 0$ and $\sigma^2_\mu = 0$, $\Delta^y_{SB} = 0$. To be sure, we can compute Δ^y_{SB} with the new parameter values: $\Delta^y_{SB} = 0$. As a consequence, $\Delta^y_{TT} = 0.18 = 0.18 = \Delta^y_{WW}$.

Remark. You might have noticed that the value of Δ^y_{TT} is different than before. It is normal, since it depends on the values of parameters, and especially on σ^2_μ and ρ.

Let's see how these quantities behave in the sample.

```
set.seed(1234)
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
Ds <- rep(0,N)
Ds[YB<=param["barY"]] <- 1
```

```
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
```

We can see that $\mathbb{E}[Y_i^0|\hat{D}_i = 1] = 8.038 \approx 8.055 = \mathbb{E}[Y_i^0|\hat{D}_i = 0]$. This means that WW should be close to TT: $\Delta_{TT}^{\hat{y}} = 0.198 \approx 0.182 = \Delta_{WW}^{\hat{y}}$. Note that $\hat{WW}$ in the sample is not exactly, but only approximately, equal to TT in the population and in the sample. This is an instance of the Fundamental Problem of Statistical Inference that we will study in the next chapter.

Under these restrictions, the placebo test would unfortunately conclude against Assumption 1.7 even though it is valid:

$$\begin{aligned}
\mathbb{E}[y_i^B|D_i = 1] &= \mathbb{E}[\mu_i + U_i^B|y_i^B \leq \bar{y}] \\
&= \mathbb{E}[\bar{\mu} + U_i^B|\bar{\mu} + U_i^B \leq \bar{y}] \\
&= \bar{\mu} + \mathbb{E}[U_i^B|\bar{\mu} + U_i^B \leq \bar{y}] \\
&\neq \bar{\mu} + \mathbb{E}[U_i^B|\bar{\mu} + U_i^B > \bar{y}] \\
&= \mathbb{E}[\mu_i + U_i^0|y_i^B > \bar{y}] \\
&= \mathbb{E}[y_i^B|D_i = 0].
\end{aligned}$$

In the sample, we indeed have that $\mathbb{E}[Y_i^B|\hat{D}_i = 1] = 7.004 \neq 8.072 = \mathbb{E}[Y_i^B|\hat{D}_i = 0]$. The reason for the failure of the placebo test to conclude that Ww is actually correct is that the U_i^B shock enters both into the selection equation and the outcome equation for y_i^B, generating a wage at period B between the outcomes of the treated and of the untreated. Since it is not persistent, this wedge does not generate selection bias. This wedge would not be detected if we could perform it further back in time, before the selection period.

Another way to make Assumption 1.7 work is to generate a new allocation rule where all the determinants of treatment intake are indeed orthogonal to potential outcomes and to outcomes before the treatment. Let's assume for example that $D_i = \mathbb{1}[V_i \leq \bar{y}]$, with $V_i \sim \mathcal{N}(\bar{\mu}, \sigma_\mu^2 + \sigma_U^2)$ and $V_i \perp\!\!\!\perp (Y_i^0, Y_i^1, Y_i^B, \mu_i, \eta_i)$. In that case, Assumption 1.7 holds and the placebo test does work. Indeed, we have:

$$\begin{aligned}
\Delta^y_{TT} &= \mathbb{E}[Y_i^1 - Y_i^0 | D_i = 1] \\
&= \mathbb{E}[\alpha_i | D_i = 1] \\
&= \mathbb{E}[\bar{\alpha} + \theta\mu_i + \eta_i | V_i \leq \bar{y}] \\
&= \bar{\alpha} + \theta\bar{\mu} \\
&= \Delta^y_{ATE} \\
\Delta^y_{WW} &= \mathbb{E}[Y_i | D_i = 1] - \mathbb{E}[Y_i | D_i = 0] \\
&= \mathbb{E}[Y_i^1 | D_i = 1] - \mathbb{E}[Y_i^0 | D_i = 0] \\
&= \mathbb{E}[Y_i^1 | V_i \leq \bar{y}] - \mathbb{E}[Y_i^0 | V_i > \bar{y}] \\
&= \mathbb{E}[Y_i^1] - \mathbb{E}[Y_i^0] \\
&= \Delta^y_{ATE}
\end{aligned}$$

ATE is the Average Treatment Effect in the population. It is the expected effect of the treatment on all the members of the population, not only on the treated. When the treatment is randomly allocated, both TT and ATE are equal, since the treated are a random subset of the overall population. I prefer to use ATE for my definition of the R function in order not to erase the definition of the TT function:

```
delta.y.ate <- function(param){
  return(param["baralpha"]+param["theta"]*param["barmu"])
}
```

In the population, WW identifies TT: $\Delta^y_{TT} = 0.18 = \Delta^y_{WW}$. Let's see how these quantities behave in the sample:

```
set.seed(1234)
N <-1000
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
Ds <- rep(0,N)
V <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]+param["sigma2U"]))
Ds[V<=log(param["barY"])] <- 1
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
```

```
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
```

In the sample, the counterfactual is well approximated by the outcomes of the untreated: $\mathbb{E}[Y_i^0|\hat{D}_i = 1] = 8.085 \approx 8.054 = \mathbb{E}[Y_i^0|\hat{D}_i = 0]$. As a consequence, WW should be close to TT: $\Delta_{TT}^{\hat{y}} = 0.168 \approx 0.199 = \Delta_{WW}^{\hat{y}}$. The placebo test is also valid in that case: $\mathbb{E}[Y_i^B|\hat{D}_i = 1] = 7.95 \approx 7.99 = \mathbb{E}[Y_i^B|\hat{D}_i = 0]$.

1.4.2 The before/after comparison, temporal confounders and time trend bias

The before/after comparison (BA) is also very intuitive: it consists in looking at how the outcomes of the treated have changed over time and to attribute this change to the effect of the treatment. The problem is that other changes might have affected outcomes in the absence of the treatment, thereby biasing BA. The bias of BA is called time-trend bias. It is due to confounders that affect the outcomes of the treated over time. This section defines the BA estimator, derives its bias, describes the role of the confounders and states conditions under which BA identifies TT.

Example 1.13. Before computing any estimates, we need to reset all our parameter values and generated sample it their usual values:

```
param <- c(8,.5,.28,1500)
names(param) <- c("barmu","sigma2mu","sigma2U","barY")
set.seed(1234)
N <-1000
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
Ds <- rep(0,N)
Ds[YB<=param["barY"]] <- 1
l <- length(param)
param <- c(param,0.9,0.01,0.05,0.05,0.05,0.1)
names(param)[(l+1):length(param)] <- c("rho","theta","sigma2epsilon","sigma2eta","delta
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
```

1.4.2.1 The before/after comparison

The before/after estimator (BA) compares the outcomes of the treated after taking the treatment to the outcomes of the treated before taking the treatment. It is also sometimes called a "pre-post comparison."

Definition 1.8 (Before/after comparison)**.** The before/after comparison is the difference between the expected outcomes in the treated group after the treatment and the expected outcomes in the same group before the treatment:

$$\Delta^Y_{BA} = \mathbb{E}[Y_i|D_i = 1] - \mathbb{E}[Y_i^B|D_i = 1].$$

Example 1.14. In the population, the BA estimator has the following shape:

$$\begin{aligned}
\Delta^y_{BA} &= \mathbb{E}[y_i|D_i = 1] - \mathbb{E}[y_i^B|D_i = 1] \\
&= \mathbb{E}[y_i^1 - y_i^B|D_i = 1] \\
&= \mathbb{E}[\alpha_i|D_i = 1] + \delta + (\rho - 1)\mathbb{E}[U_i^B|\mu_i + U_i^B \leq \bar{y}] \\
&= \Delta^y_{TT} + \delta + (1 - \rho)\left(\frac{\sigma_U^2}{\sqrt{\sigma_\mu^2 + \sigma_U^2}} \frac{\phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}{\Phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma_\mu^2+\sigma_U^2}}\right)}\right).
\end{aligned}$$

In order to compute BA in the population, we can again use a R function, combining the value of TT and that of the second part of the formula, that we are going to denote TB for reasons that are going to become clear in a bit.

```
delta.y.tb <- function(param){
  return(param["delta"]
          +(1-param["rho"])*((param["sigma2U"])/sqrt(param["sigma2mu"]+param["sigma2U"]))
         *dnorm((log(param["barY"])-param["barmu"])/(sqrt(param["sigma2mu"]+param["sigma2U"])))
         /pnorm((log(param["barY"])-param["barmu"])/(sqrt(param["sigma2mu"]+param["sigma2U"]))))
}
delta.y.ba <- function(param){
  return(delta.y.tt(param)+ delta.y.tb(param))
}
```

The value of BA in the population is thus $\Delta^y_{BA} = 0.265$. Remember that the true value of TT in the population is 0.172. In the sample, the value of BA is $\Delta^{\hat{y}}_{BA} = 0.267$. Remember that the value of TT in the sample is $\Delta^y_{TT_s} = 0.168$.

1.4.2.2 Time trend bias

When we form the before/after comparison, we do not recover the TT parameter. Instead, we recover TT plus a bias term, called **time trend bias**:

$$\Delta^Y_{BA} = \Delta^Y_{TT} + \Delta^Y_{TB}.$$

Definition 1.9 (Time trend bias)**.** Time trend bias is the difference between the before/after comparison and the treatment on the treated parameter:

$$\Delta^Y_{TB} = \Delta^Y_{BA} - \Delta^Y_{TT}.$$

BA uses the expected outcome in the treated group before the treatment as a proxy for the expected counterfactual outcome in the absence of the treatment in the same group. TB is due to the fact that BA uses an imperfect proxy for the counterfactual expected outcome of the treated:

Theorem 1.4. *Time trend bias is the difference between the counterfactual expected potential outcome in the absence of the treatment among the treated and the expected outcome before the treatment in the same group.*

$$\Delta^Y_{TB} = \mathbb{E}[Y^0_i|D_i = 1] - \mathbb{E}[Y^B_i|D_i = 1].$$

Proof.

$$\begin{aligned}\Delta^Y_{TB} &= \Delta^Y_{BA} - \Delta^Y_{TT} \\ &= \mathbb{E}[Y_i|D_i = 1] - \mathbb{E}[Y^B_i|D_i = 1] - \mathbb{E}[Y^1_i - Y^0_i|D_i = 1] \\ &= \mathbb{E}[Y^0_i|D_i = 1] - \mathbb{E}[Y^B_i|D_i = 1].\end{aligned}$$

The first and second equalities stem from the definition of both parameters. The third equality stems from using the switching equation: $Y_i = Y^1_i D_i + Y^0_i(1 - D_i)$, so that $\mathbb{E}[Y_i|D_i = 1] = \mathbb{E}[Y^1_i|D_i = 1]$. □

Example 1.15. In the population, TB is equal to

$\Delta^y_{TB} = \Delta^y_{BA} - \Delta^y_{TT} = 0.265 - 0.172 = 0.093$. We could have computed this result using Theorem1.4:

$$\begin{aligned}\Delta^y_{TB} &= \mathbb{E}[y^0_i|D_i = 1] - \mathbb{E}[y^B_i|D_i = 1] \\ &= \delta + (1 - \rho)\left(\frac{\sigma^2_U}{\sqrt{\sigma^2_\mu + \sigma^2_U}} \frac{\phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma^2_\mu+\sigma^2_U}}\right)}{\Phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{\sigma^2_\mu+\sigma^2_U}}\right)}\right).\end{aligned}$$

Using the R function that we have defined previously, this approach gives $\Delta^y_{TB} = 0.093$.

In the sample $\Delta^{\hat{y}}_{BA} = 0.267$ while $\Delta^{\hat{y}}_{TT} = 0.168$, so that $\Delta^{\hat{y}}_{TB} = 0.099$. Time trend bias emerges because we are using a bad proxy for the counterfactual average outcomes of the treated. The average outcome of the treated before the treatment takes place is $\mathbb{E}[y^B_i|\hat{D}_i = 1] = 6.807$ while the true counterfactual average outcome for the treated after the treatment is $\mathbb{E}[y^0_i|\hat{D}_i = 1] = 6.906$.

Outcomes would have increased in the treatment group even in the absence of the treatment. As a consequence, the BA comparison overestimates the true effect of the treatment. TB estimated using Theorem 1.4 is equal to: $\Delta^{\hat{y}}_{TB} = 6.906 - 6.807 = 0.099$. This can be seen on Figure 1.6: the triangle in period 2 is higher than in period 1.

1.4.2.3 Temporal confounders

Temporal confounders make the outcomes of the treated change at the same time as the treatment status changes, thereby confounding the effect of the treatment. Temporal confounders are responsible for time trend bias.

Over time, there are other things that change than the treatment status. For example, maybe sick individuals naturally recover, and thus their counterfactual health status is better than ther health status before taking the treatment. As a results, BA might overstimate the effect of the treatment. It might also be that the overall level of health in the country has increased, because of increasing GDP, for example.

Example 1.16. In our example, δ and U_i^B are the confounders. δ captures the overall changes in outcomes over time (business cycle, general improvement of health status). U_i^B captures the fact that transitorily sicker individuals tend at the same time to receive the treatment and also to recover naturally. The BA comparison incorrectly attributes both of these changes to the effect of the treatment. We can compute the relative contributions of both sources to the overall time-trend bias in the population.

δ contributes for 0.05 while U_i^B contributes for 0.043.

1.4.2.4 When does BA identify TT?

Are there conditions under which BA actually identifies TT? The answer is yes, when there are no temporal confounders. When that is the case, the variation of outcomes over time is only due to the treatment and it identifies the treatment effect.

More formally, we make the following assumption:

Definition 1.10 (No time trend bias)**.** We assume the following:

$$\mathbb{E}[Y_i^0|D_i = 1] = \mathbb{E}[Y_i^B|D_i = 1].$$

Under Assumption 1.10, the expected counterfactual outcome of the treated is equal to the expected potential outcome of the untreated in the absence of the treatment. This yields to the following result:

Theorem 1.5. *Under Assumption 1.10, BA identifies the TT parameter:*

$$\Delta^Y_{BA} = \Delta^Y_{TT}.$$

Proof.

$$\begin{aligned}\Delta^Y_{BA} &= \mathbb{E}[Y_i|D_i=1] - \mathbb{E}[Y_i^B|D_i=1] \\ &= \mathbb{E}[Y_i^1|D_i=1] - \mathbb{E}[Y_i^B|D_i=1] \\ &= \mathbb{E}[Y_i^1|D_i=1] - \mathbb{E}[Y_i^0|D_i=1] \\ &= \Delta^Y_{TT},\end{aligned}$$

where the second equation uses the switching equation and the third uses Assumption 1.10. □

Under Assumption 1.10 the outcomes of the treated before the treatment takes place are a good proxy for the counterfactual. As a consequence, BA identifies TT. Under Assumption 1.10is very unlikely to hold in real life. Indeed, it requires tha nothing happens to the outcomes of the treated in the absence of the treatment. Assumption 1.10 rules out economy-wide shocks but also mean-reversion, such as when sick people naturally recover from an illness.

A good way to test for the validity of Assumption 1.10 is to perform a placebo test. Two of these tests are possible. One placebo test would be to apply the BA estimator between two pre-treatment periods where nothing should happen since the treatment status does not vary and, by assumption, nothing else should vary. A second placebo test would be to apply the BA estimator to a group that does not receive the treatment. The untreated group is a perfectly suitable candidate for that. Assumption 1.10 does not imply that there should be no change in the untreated outcomes, but detecting such a change would cast a serious doubt on the validity of Assumption 1.10.

Example 1.17. One way to generate a population in which Assumption 1.10 holds is to shut down the two sources of confounders in our original model by setting both $\delta = 0$ and $\rho = 1$.

```
param <- c(8,0.5,.28,1500,1,0.01,0.05,0.05,0,0.1)
names(param) <- c("barmu","sigma2mu","sigma2U","barY","rho","theta","sigma2epsilon","si
param
```

```
##         barmu      sigma2mu       sigma2U          barY           rho
##          8.00          0.50          0.28       1500.00          1.00
##         theta sigma2epsilon     sigma2eta         delta      baralpha
##          0.01          0.05          0.05          0.00          0.10
```

In that case, according to the formula we have derived for TB, we have: $\Delta^y_{TB} = 0$. Let's see how these quantities behave in the sample:

```
set.seed(1234)
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
```

```
Ds <- rep(0,N)
Ds[YB<=param["barY"]] <- 1
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
```

In the sample, the value of BA is $\Delta^{\hat{y}}_{BA} = 0.173$ while the value of TT in the sample is $\Delta^{y}_{TT_s} = 0.168$. We cannot perform a placebo test using two periods of pre-treatment outcomes for the treated since we have generated only one period of pre-treatment outcome. We will be able to perform this test later in the DID lecture. We can perfom the placebo test that applies the BA estimator to the untreated. The value of BA for the untreated is $\hat{\Delta^{y}_{BA|D=0}} = 0.007$, which is reasonably close to zero.

Chapter 2

Fundamental Problem of Statistical Inference

The Fundamental Problem of Statistical Inference (FPSI) states that, even if we have an estimator E that identifies TT in the population, we cannot observe E because we only have access to a finite sample of the population. The only thing that we can form from the sample is a sample equivalent $\hat{E}$ to the population quantity E, and $\hat{E} \neq E$. For example, the sample analog to WW is the difference in means between treated and untreated units $\hat{WW}$. As we saw in the last lecture, $\hat{WW}$ is never exactly equal to WW.

Why is $\hat{E} \neq E$? Because a finite sample is never perfectly representative of the population. In a sample, even in a random sample, the distribution of the observed and unobserved covariates deviates from the true population one. As a consequence, the sample value of the estimator is never precisely equal to the population value, but fluctuates around it with sampling noise. The main problem with the FPSI is that if we find an effect of our treatment, be it small or large, we cannot know whether we should attribute it to the treatment or to the bad or good luck of sampling noise.

What can we do to deal with the FPSI? I am going to argue that there are mainly two things that we might want to do: estimating the extent of sampling noise and decreasing sampling noise.

Estimating sampling noise means measuring how much variability there is in our estimate $\hat{E}$ due to the sampling procedure. This is very useful because it enables us to form a confidence interval that gauges how far from $\hat{E}$ the true value E might be. It is a measure of the precision of our estimation and of the extent to which sampling noise might drive our results. Estimating sampling noise is very hard because we have only access to one sample and we would like to know the behavior of our estimator over repeated samples. We are going to learn four

ways to estimate the extent of sampling noise using data from one sample.

Because sampling noise is such a nuisance and makes our estimates imprecise, we would like to be able to make it as small as possible. We are going to study three ways of decreasing sampling noise, two that take place before collecting the data (increasing sample size, stratifying) and one that takes place after (conditioning).

Maybe you are surprised not to find statistical significance tests as an important answer to the FPSI. I argue in this lecture that statistical tests are misleading tools that make us overestimate the confidence in our results and underestimate the scope of sampling noise. Statistical tests are not meant to be used for scientific research, but were originally designed to make decisions in industrial settings where the concept of successive sampling made actual sense. Statistical tests also generate collective behaviors such as publication bias and specification search that undermine the very foundations of science. A general movement in the social sciences, but also in physics, is starting to ban the reporting of p-values.

2.1 What is sampling noise? Definition and illustration

In this section, I am going to define sampling noise and illustrate it with a numerical example. In Section 2.1.1, I define sampling noise. In section 2.1.2, I illustrate how sampling noise varies when one is interested in the population treatment effect. In section 2.1.3, I illustrate how sampling noise varies when one is interesetd in the sample treatment effect. Finally, in section 2.1.4, I show how confidence intervals can be built from an estimate of sampling noise.

2.1.1 Sampling noise, a definition

Sampling noise measures how much sampling variability moves the sample estimator $\hat{E}$ around. One way to define it more rigorously is to make it equal to the width of a confidence interval:

Definition 2.1 (Sampling noise)**.** Sampling noise is the width of the symmetric interval around TT within which $\delta * 100\%$ of the sample estimators fall, where δ is the confidence level. As a consequence, sampling noise is equal to 2ϵ where ϵ is such that:

$$\Pr(|\hat{E} - TT| \leq \epsilon) = \delta.$$

This definition tries to capture the properties of the distribution of $\hat{E}$ using only one number. As every simplification, it leaves room for dissatisfaction, exactly as a 2D map is a convenient albeit arbitrary betrayal of a 3D phenomenon. For example, there is nothing sacred about the symmetry of the interval. It is just

extremely convenient. One might prefer an interval that is symmetric in tail probabilities instead. Feel free to explore with different concepts if you like.

A related concept to that of sampling noise is that of precision: the smaller the sampling noise, the higher the precision. Precision can be defined for example as the inverse of sampling noise $\frac{1}{2\epsilon}$.

Finally, a very useful concept is that of signal to noise ratio. It is not used in economics, but physicists use this concept all the time. The signal to noise ratio measures the treatment effect in multiple of the sampling noise. If they are of the same order of magnitude, we have a lot of noise and little confidence in our estimates. If the signal is much larger than the noise, we tend to have a lot of confidence in our parameter estimates. The signal to noise ratio can be computed as follows: $\frac{E}{2\epsilon}$ or $\frac{\hat{E}}{2\epsilon}$.

Remark. A very striking result is that the signal to noise ratio of a result that is marginally significant at the 5% level is very small, around one half, meaning that the noise is generally double the signal in these results. We will derive this result after studying how to estimate sampling noise with real data.

There are two distinct ways of understanding sampling noise, depending on whether we are after the population treatment effect (Δ^Y_{TT}) or the sample treatment effect ($\Delta^Y_{TT_s}$). Sampling noise for the population treatment effect stems from the fact that the sample is not perfectly representative of the population. The sample differs from the population and thus the sample estimates differs from the population estimate. Sampling noise for the sample parameter stems from the fact that the control group is not a perfect embodiment of the counterfactual. Discrepancies between treated and control samples are going to generate differences between the WW estimate and the TT effect in the sample.

2.1.2 Sampling noise for the population treatment effect

Sampling noise for the population treatment effect stems from the fact that the sample is not perfectly representative of the population.

Example 2.1. In order to assess the scope of sampling noise for our population treatment effect estimate, let's first draw a sample. In order to be able to do that, I first have to define the parameter values:

```
param <- c(8,.5,.28,1500,0.9,0.01,0.05,0.05,0.05,0.1)
names(param) <- c("barmu","sigma2mu","sigma2U","barY","rho","theta","sigma2epsilon","sigma2eta","
param
```

```
##         barmu      sigma2mu       sigma2U          barY           rho
##          8.00          0.50          0.28       1500.00          0.90
##         theta sigma2epsilon     sigma2eta         delta      baralpha
##          0.01          0.05          0.05          0.05          0.10
```

```
set.seed(1234)
N <-1000
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
Ds <- rep(0,N)
V <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]+param["sigma2U"]))
Ds[V<=log(param["barY"])] <- 1
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)

delta.y.ate <- function(param){
  return(param["baralpha"]+param["theta"]*param["barmu"])
}
```

In this sample, the WW estimator yields an estimate of $\hat{\Delta^{y}_{WW}} = 0.133$. Despite random assignment, we have $\hat{\Delta^{y}_{WW}} \neq \Delta^{y}_{TT} = 0.18$, an instance of the FPSI.

In order to see how sampling noise varies, let's draw another sample. In order to do so, I am going to choose a different seed to initialize the pseudo-random number generator in R.

```
set.seed(12345)
N <-1000
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
Ds <- rep(0,N)
V <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]+param["sigma2U"]))
Ds[V<=log(param["barY"])] <- 1
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta
y1 <- y0+alpha
```

```
Y0 <- exp(y0)
Y1 <- exp(y1)
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
```

In this sample, the WW estimator yields an estimate of $\hat{\Delta^y_{WW}} = 0.179$. Again, despite random assignment, we have $\hat{\Delta^y_{WW}} \neq \Delta^y_{TT} = 0.18$, an instance of the FPSI. Furthermore, the estimate of the population treatment effect in this sample differs from the previous one, a consequence of sampling noise.

Let's now visualize the extent of sampling noise by repeating the procedure multiple times with various sample sizes. This is called Monte Carlo replications: in each replication, I choose a sample size, draw one sample from the population and compute the $\hat{WW}$ estimator. At each replication, the sample I'm using is different, reflecting the actual sampling process and enabling me to gauge the extent of sampling noise. In order to focus on sampling noise alone, I am running the replications in the model in which selection into the treatment is independent on potential outcomes, so that $WW = TT$ in the population. In order to speed up the process, I am using parallelized computing: I send each sample to a different core in my computer so that several samples can be run at the same time. You might want to adapt the program below to the number of cores you actually have using the `ncpus` variable in the beginning of the `.Rmd` file that generates this page.. In order to parallelize computations, I use the Snowfall package in R, that gives very simple and intuitive parallelization commands. In order to save time when generating the graph, I use the wonderful "cache" option of knitr: it stores the estimates from the code chunk and will not rerun it as long as the code inside the chunk has not been altered nor the code of the chunks that it depends on (parameter values, for example).

```
monte.carlo.ww <- function(s,N,param){
  set.seed(s)
  mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
  UB <- rnorm(N,0,sqrt(param["sigma2U"]))
  yB <- mu + UB
  YB <- exp(yB)
  Ds <- rep(0,N)
  V <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]+param["sigma2U"]))
  Ds[V<=log(param["barY"])] <- 1
  epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
  eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
  U0 <- param["rho"]*UB + epsilon
  y0 <- mu +  U0 + param["delta"]
  alpha <- param["baralpha"]+  param["theta"]*mu + eta
  y1 <- y0+alpha
  Y0 <- exp(y0)
  Y1 <- exp(y1)
```

```
  y <- y1*Ds+y0*(1-Ds)
  Y <- Y1*Ds+Y0*(1-Ds)
  return(c((1/sum(Ds))*sum(y*Ds)-(1/sum(1-Ds))*sum(y*(1-Ds)),var(y[Ds==1]),var(y[Ds==0]
}

simuls.ww.N <- function(N,Nsim,param){
  simuls.ww <- as.data.frame(matrix(unlist(lapply(1:Nsim,monte.carlo.ww,N=N,param=param
  colnames(simuls.ww) <- c('WW','V1','V0','p')
  return(simuls.ww)
}

sf.simuls.ww.N <- function(N,Nsim,param){
  sfInit(parallel=TRUE,cpus=ncpus)
  sim <- as.data.frame(matrix(unlist(sfLapply(1:Nsim,monte.carlo.ww,N=N,param=param)),n
  sfStop()
  colnames(sim) <- c('WW','V1','V0','p')
  return(sim)
}

simuls.ww <- lapply(N.sample,sf.simuls.ww.N,Nsim=Nsim,param=param)

par(mfrow=c(2,2))
for (i in 1:4){
  hist(simuls.ww[[i]][,'WW'],main=paste('N=',as.character(N.sample[i])),xlab=expression
  abline(v=delta.y.ate(param),col="red")
}
```

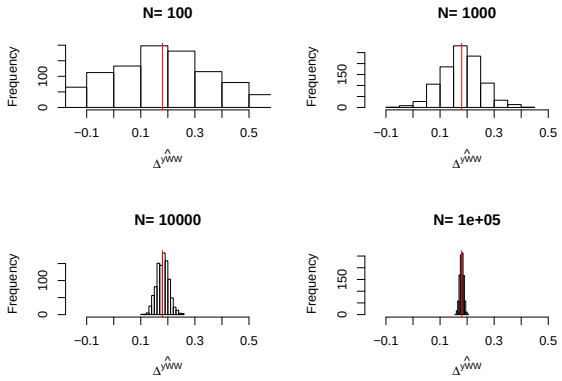


Figure 2.1: Distribution of the WW estimator over replications of samples of different sizes

Figure 2.1 is essential to understanding statistical inference and the properties of our estimators. We can see on Figure 2.1 that the estimates indeed move around at each sample replication. We can also see that the estimates seem to

Table 2.1: Sampling noise of $W\hat{W}$ for the population treatment effect with $\delta = 0.99$ for various sample sizes

	Sampling noise	Precision	Signal to noise ratio
100	1.10	0.91	0.16
1000	0.39	2.56	0.46
10000	0.12	8.63	1.55
1e+05	0.04	28.35	5.10

be concentrated around the truth. We also see that the estimates are more and more concentrated around the truth as sample size grows larger and larger.

How big is sampling noise in all of these examples? We can compute it by using the replications as approximations to the true distribution of the estimator after an infinite number of samples has been drawn. Let's first choose a confidence level and then compute the empirical equivalent to the formula in Definition 2.1.

```
delta<- 0.99
delta.2 <- 0.95
samp.noise <- function(estim,delta){
  return(2*quantile(abs(delta.y.ate(param)-estim),prob=delta))
}
samp.noise.ww <- sapply(lapply(simuls.ww,`[`,,1),samp.noise,delta=delta)
names(samp.noise.ww) <- N.sample
samp.noise.ww
```

```
##        100       1000      10000      1e+05
## 1.09916429 0.39083801 0.11582492 0.03527744
```

Let's also compute precision and the signal to noise ratio and put all of these results together in a nice table.

```
precision <- function(estim,delta){
  return(1/samp.noise(estim,delta))
}
signal.to.noise <- function(estim,delta,param){
  return(delta.y.ate(param)/samp.noise(estim,delta))
}
precision.ww <- sapply(lapply(simuls.ww,`[`,,1),precision,delta=delta)
names(precision.ww) <- N.sample
signal.to.noise.ww <- sapply(lapply(simuls.ww,`[`,,1),signal.to.noise,delta=delta,param=param)
names(signal.to.noise.ww) <- N.sample
table.noise <- cbind(samp.noise.ww,precision.ww,signal.to.noise.ww)
colnames(table.noise) <- c('Sampling noise', 'Precision', 'Signal to noise ratio')
knitr::kable(table.noise,caption=paste('Sampling noise of $\\hat{WW}$ for the population treatmen
```

Finally, a nice way to summarize the extent of sampling noise is to graph how

sampling noise varies around the true treatment effect, as shown on Figure 2.2.

```
colnames(table.noise) <- c('sampling.noise', 'precision', 'signal.to.noise')
table.noise <- as.data.frame(table.noise)
table.noise$N <- as.numeric(rownames(table.noise))
table.noise$TT <- rep(delta.y.ate(param),nrow(table.noise))
ggplot(table.noise, aes(x=as.factor(N), y=TT)) +
  geom_bar(position=position_dodge(), stat="identity", colour='black') +
  geom_errorbar(aes(ymin=TT-sampling.noise/2, ymax=TT+sampling.noise/2), width=.2,posi
  xlab("Sample Size")+
  theme_bw()
```

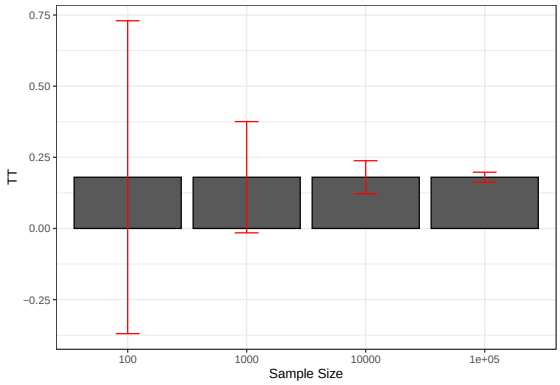


Figure 2.2: Sampling noise of $\hat{WW}$ (99% confidence) around TT for various sample sizes

With $N = 100$, we can definitely see on Figure 2.2 that sampling noise is ridiculously large, especially compared with the treatment effect that we are trying to estimate. The signal to noise ratio is 0.16, which means that sampling noise is an order of magnitude bigger than the signal we are trying to extract. As a consequence, in 22.2% of our samples, we are going to estimate a negative effect of the treatment. There is also a 20.4% chance that we end up estimating an effect that is double the true effect. So how much can we trust our estimate from one sample to be close to the true effect of the treatment when $N = 100$? Not much.

With $N = 1000$, sampling noise is still large: the signal to noise ratio is 0.46, which means that sampling noise is double the signal we are trying to extract. As a consequence, the chance that we end up with a negative treatment effect has decreased to 0.9% and that we end up with an effect double the true one is 1%. But still, the chances that we end up with an effect that is smaller than three quarters of the true effect is 25.6% and the chances that we end up with an estimator that is 25% bigger than the true effect is 26.2%. These are nontrivial differences: compare a program that increases earnings by 13.5% to one that increases them by 18% and another by 22.5%. They would have completely

different cost/benefit ratios. But we at least trust our estimator to give us a correct idea of the sign of the treatment effect and a vague and imprecise idea of its magnitude.

With $N = 10^4$, sampling noise is smaller than the signal, which is encouraging. The signal to noise ratio is 1.55. In only 1% of the samples does the estimated effect of the treatment become smaller than 0.125 or bigger than 0.247. We start gaining a lot of confidence in the relative magnitude of the effect, even if sampling noise is still responsible for economically significant variation.

With $N = 10^5$, sampling noise has become trivial. The signal to noise ratio is 5.1, which means that the signal is now 5 times bigger than the sampling noise. In only 1% of the samples does the estimated effect of the treatment become smaller than 0.163 or bigger than 0.198. Sampling noise is not any more responsible for economically meaningful variation.

2.1.3 Sampling noise for the sample treatment effect

Sampling noise for the sample parameter stems from the fact that the treated and control groups are not perfectly identical. The distribution of observed and unobserved covariates is actually different, because of sampling variation. This makes the actual comparison of means in the sample a noisy estimate of the true comparison that we would obtain by comparing the potential outcomes of the treated directly.

In order to understand this issue well and to be able to illustrate it correctly, I am going to focus on the average treatment effect in the whole sample, not on the treated: $\Delta^Y_{ATE_s} = \frac{1}{N}\sum_{i=1}^{N}(Y_i^1 - Y_i^0)$. This enables me to define a sample parameter that is independent of the allocation of D_i. This is without important consequences since these two parameters are equal in the population when there is no selection bias, as we are assuming since the beginning of this lecture. Furthermore, if we view the treatment allocation generating no selection bias as a true random assignment in a Randomized Controlled Trial (RCT), then it is still possible to use this approach to estimate TT if we view the population over which we randomise as the population selected for receiving the treatment, as we will see in the lecture on RCTs.

Example 2.2. In order to assess the scope of sampling noise for our sample treatment effect estimate, we first have to draw a sample:

```
set.seed(1234)
N <-1000
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
Ds <- rep(0,N)
V <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]+param["sigma2U"]))
```

```
Ds[V<=log(param["barY"])] <- 1
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
```

In this sample, the treatment effect parameter is $\Delta^{y}_{ATE_s} = 0.171$. The WW estimator yields an estimate of $\Delta^{\hat{y}}_{WW} = 0.133$. Despite random assignment, we have $\Delta^{y}_{ATE_s} \neq \Delta^{\hat{y}}_{WW}$, an instance of the FPSI.

In order to see how sampling noise varies, let's draw a new treatment allocation, while retaining the same sample and the same potential outcomes.

```
set.seed(12345)
N <-1000
Ds <- rep(0,N)
V <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]+param["sigma2U"]))
Ds[V<=log(param["barY"])] <- 1
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
```

In this sample, the treatment effect parameter is still $\Delta^{y}_{ATE_s} = 0.171$. The WW estimator yields now an estimate of $\Delta^{\hat{y}}_{WW} = 0.051$. The WW estimate is different from our previous estimate because the treatment was allocated to a different random subset of people.

Why is this second estimate so imprecise? It might because it estimates one of the two components of the average treatment effect badly, or both. The true average potential outcome with the treatment is, in this sample, $\frac{1}{N}\sum_{i=1}^{N} y_i^1 = 8.207$ while the WW estimate of this quantity is $\frac{1}{\sum_{i=1}^{N} D_i}\sum_{i=1}^{N} D_i y_i = 8.113$. The true average potential outcome without the treatment is, in this sample, $\frac{1}{N}\sum_{i=1}^{N} y_i^0 = 8.036$ while the WW estimate of this quantity is $\frac{1}{\sum_{i=1}^{N}(1-D_i)}\sum_{i=1}^{N}(1-D_i)y_i = 8.062$. It thus seems that most of the bias in the estimated effect stems from the fact that the treatment has been allocated to individuals with lower than expected outcomes with the treatment, be it because they did not react strongly to the treatment, or because they were in worse shape without the treatment. We can check which one of these two explanations is more important. The true average effect of the treatment is, in this sample, $\frac{1}{N}\sum_{i=1}^{N}(y_i^1 - y_i^0) = 0.171$ while,

in the treated group, this quantity is $\frac{1}{\sum_{i=1}^{N} D_i} \sum_{i=1}^{N} D_i(y_i^1 - y_i^0) = 0.18$. The true average potential outcome without the treatment is, in this sample, $\frac{1}{N} \sum_{i=1}^{N} y_i^0 = 8.036$ while, in the treated group, this quantity is $\frac{1}{\sum_{i=1}^{N} D_i} \sum_{i=1}^{N} D_i y_i^0 = 7.933$. The reason for the poor performance of the WW estimator in this sample is that individuals with lower counterfactual outcomes were included in the treated group, not that the treatment had lower effects on them. The bad counterfactual outcomes of the treated generates a bias of -0.103, while the bias due to heterogeneous reactions to the treatment is of 0.009. The last part of the bias is the one due to the fact that the individuals in the control group have slightly better counterfactual outcomes than in the sample: -0.026. The sum of these three terms yields the total bias of our WW estimator in this second sample: -0.12.

Let's now assess the overall effect of sampling noise on the estimate of the sample treatment effect for various sample sizes. In order to do this, I am going to use parallelized Monte Carlo simulations again. For the sake of simplicity, I am going to generate the same potential outcomes in each replication, using the same seed, and only choose a different treatment allocation.

```
monte.carlo.ww.sample <- function(s,N,param){
  set.seed(1234)
  mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
  UB <- rnorm(N,0,sqrt(param["sigma2U"]))
  yB <- mu + UB
  YB <- exp(yB)
  epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
  eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
  U0 <- param["rho"]*UB + epsilon
  y0 <- mu +  U0 + param["delta"]
  alpha <- param["baralpha"]+  param["theta"]*mu + eta
  y1 <- y0+alpha
  Y0 <- exp(y0)
  Y1 <- exp(y1)
  set.seed(s)
  Ds <- rep(0,N)
  V <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]+param["sigma2U"]))
  Ds[V<=log(param["barY"])] <- 1
  y <- y1*Ds+y0*(1-Ds)
  Y <- Y1*Ds+Y0*(1-Ds)
  return((1/sum(Ds))*sum(y*Ds)-(1/sum(1-Ds))*sum(y*(1-Ds)))
}

simuls.ww.N.sample <- function(N,Nsim,param){
  return(unlist(lapply(1:Nsim,monte.carlo.ww.sample,N=N,param=param)))
}
```

```
sf.simuls.ww.N.sample <- function(N,Nsim,param){
  sfInit(parallel=TRUE,cpus=ncpus)
  sim <- sfLapply(1:Nsim,monte.carlo.ww.sample,N=N,param=param)
  sfStop()
  return(unlist(sim))
}

simuls.ww.sample <- lapply(N.sample,sf.simuls.ww.N.sample,Nsim=Nsim,param=param)

monte.carlo.ate.sample <- function(N,s,param){
  set.seed(s)
  mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
  UB <- rnorm(N,0,sqrt(param["sigma2U"]))
  yB <- mu + UB
  YB <- exp(yB)
  epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
  eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
  U0 <- param["rho"]*UB + epsilon
  y0 <- mu +  U0 + param["delta"]
  alpha <- param["baralpha"]+  param["theta"]*mu + eta
  y1 <- y0+alpha
  Y0 <- exp(y0)
  Y1 <- exp(y1)
  Ds <- rep(0,N)
  V <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]+param["sigma2U"]))
  Ds[V<=log(param["barY"])] <- 1
  y <- y1*Ds+y0*(1-Ds)
  Y <- Y1*Ds+Y0*(1-Ds)
  return(mean(alpha))
}

par(mfrow=c(2,2))
for (i in 1:4){
  hist(simuls.ww.sample[[i]],main=paste('N=',as.character(N.sample[i])),xlab=expressio
  abline(v=monte.carlo.ate.sample(N.sample[[i]],1234,param),col="red")
}
```

Let's also compute sampling noise, precision and the signal to noise ratio in these examples.

```
samp.noise.sample <- function(i,delta,param){
  return(2*quantile(abs(monte.carlo.ate.sample(1234,N.sample[[i]],param)-simuls.ww.samp
}
samp.noise.ww.sample <- sapply(1:4,samp.noise.sample,delta=delta,param=param)
names(samp.noise.ww.sample) <- N.sample
```

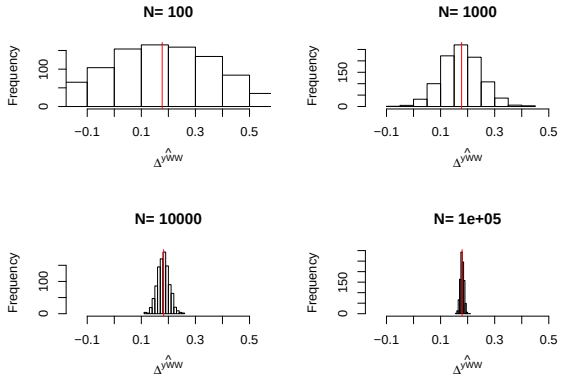


Figure 2.3: Distribution of the WW estimator over replications of treatment allocation for samples of different sizes

Table 2.2: Sampling noise of $\hat{WW}$ for the sample treatment effect with $\delta = 0.99$ and for various sample sizes

	Sampling noise	Precision	Signal to noise ratio
100	1.208	0.828	0.149
1000	0.366	2.729	0.482
10000	0.122	8.218	1.585
1e+05	0.033	30.283	5.453

```
precision.sample <- function(i,delta,param){
  return(1/samp.noise.sample(i,delta,param=param))
}
signal.to.noise.sample <- function(i,delta,param){
  return(monte.carlo.ate.sample(1234,N.sample[[i]],param)/samp.noise.sample(i,delta,param=param))
}
precision.ww.sample <- sapply(1:4,precision.sample,delta=delta,param=param)
names(precision.ww.sample) <- N.sample
signal.to.noise.ww.sample <- sapply(1:4,signal.to.noise.sample,delta=delta,param=param)
names(signal.to.noise.ww.sample) <- N.sample
table.noise.sample <- cbind(samp.noise.ww.sample,precision.ww.sample,signal.to.noise.ww.sample)
colnames(table.noise.sample) <- c('Sampling noise', 'Precision', 'Signal to noise ratio')
knitr::kable(table.noise.sample,caption=paste('Sampling noise of $\\hat{WW}$ for the sample treat
```

Finally, let's compare the extent of sampling noise for the population and the sample treatment effect parameters.

```
colnames(table.noise.sample) <- c('sampling.noise', 'precision', 'signal.to.noise')
table.noise.sample <- as.data.frame(table.noise.sample)
```

```
table.noise.sample$N <- as.numeric(rownames(table.noise.sample))
table.noise.sample$TT <- sapply(N.sample,monte.carlo.ate.sample,s=1234,param=param)
table.noise.sample$Type <- 'TTs'
table.noise$Type <- 'TT'
table.noise.tot <- rbind(table.noise,table.noise.sample)
table.noise.tot$Type <- factor(table.noise.tot$Type)

ggplot(table.noise.tot, aes(x=as.factor(N), y=TT,fill=Type)) +
  geom_bar(position=position_dodge(), stat="identity", colour='black') +
  geom_errorbar(aes(ymin=TT-sampling.noise/2, ymax=TT+sampling.noise/2), width=.2,posi
  xlab("Sample Size")+
  theme_bw()+
  theme(legend.position=c(0.85,0.88))
```

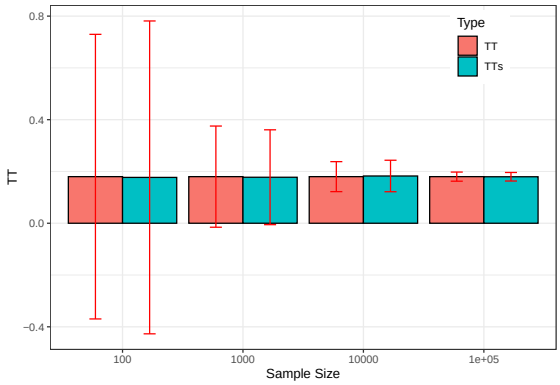


Figure 2.4: Sampling noise of $W\hat{W}$ (99% confidence) around TT and TT_s for various sample sizes

Figure 2.3 and Table 2.2 present the results of the simulations of sampling noise for the sample treatment effect parameter. Figure 2.4 compares sampling noise for the population and sample treatment effects.
For all practical purposes, the estimates of sampling noise for the sample treatment effect are extremely close to the ones we have estimated for the population treatment effect. I am actually surprised by this result, since I expected that keeping the potential outcomes constant over replications would decrease sampling noise. It seems that the variability in potential outcomes over replications of random allocations of the treatment in a given sample mimicks very well the sampling process from a population. I do not know if this result of similarity of sampling noise for the population and sample treatment effect is a general one, but considering them as similar or close seems innocuous in our example.

2.1.4 Building confidence intervals from estimates of sampling noise

In real life, we do not observe TT. We only have access to $\hat{E}$. Let's also assume for now that we have access to an estimate of sampling noise, 2ϵ. How can we use these two quantities to assess the set of values that TT might take? One very useful device that we can use is the confidence interval. Confidence intervals are very useful because they quantify the zone within which we have a chance to find the true effect TT:

Theorem 2.1 (Confidence interval)**.** *For a given level of confidence δ and corresponding level of sampling noise 2ϵ of the estimator $\hat{E}$ of TT, the confidence interval $\left\{\hat{E}-\epsilon, \hat{E}+\epsilon\right\}$ is such that the probability that it contains TT is equal to δ over sample replications:*

$$\Pr(\hat{E}-\epsilon \leq TT \leq \hat{E}+\epsilon) = \delta.$$

Proof. From the definition of sampling noise, we know that:

$$\Pr(|\hat{E}-TT| \leq \epsilon) = \delta.$$

Now:

$$\begin{aligned}\Pr(|\hat{E}-TT| \leq \epsilon) &= \Pr(TT-\epsilon \leq \hat{E} \leq TT+\epsilon) \\ &= \Pr(-\hat{E}-\epsilon \leq -TT \leq -\hat{E}+\epsilon) \\ &= \Pr(\hat{E}-\epsilon \leq TT \leq \hat{E}+\epsilon),\end{aligned}$$

which proves the result. □

It is very important to note that confidence intervals are centered around $\hat{E}$ and not around TT. When estimating sampling noise and building Figure 2.2, we have centered our intervals around TT. The interval was fixed and $\hat{E}$ was moving across replications and 2ϵ was defined as the length of the interval around TT containing a proportion δ of the estimates $\hat{E}$. A confidence interval cannot be centered around TT, which is unknown, but is centered around $\hat{E}$, that we can observe. As a consequence, it is the interval that moves around across replications, and δ is the proportion of samples in which the interval contains TT.

Example 2.3. Let's see how confidence intervals behave in our numerical example.

```
N.plot <- 40
plot.list <- list()

for (k in 1:length(N.sample)){
  set.seed(1234)
```

```
  test <- sample(simuls.ww[[k]][,'WW'],N.plot)
  test <- as.data.frame(cbind(test,rep(samp.noise(simuls.ww[[k]][,'WW'],delta=delta)),r
  colnames(test) <- c('WW','sampling.noise.1','sampling.noise.2')
  test$id <- 1:N.plot
  plot.test <- ggplot(test, aes(x=as.factor(id), y=WW)) +
      geom_bar(position=position_dodge(), stat="identity", colour='black') +
      geom_errorbar(aes(ymin=WW-sampling.noise.1/2, ymax=WW+sampling.noise.1/2), width=
      geom_errorbar(aes(ymin=WW-sampling.noise.2/2, ymax=WW+sampling.noise.2/2), width=
      geom_hline(aes(yintercept=delta.y.ate(param)), colour="#990000", linetype="dashed
      #ylim(-0.5,1.2)+
      xlab("Sample id")+
      theme_bw()+
      ggtitle(paste("N=",N.sample[k]))
  plot.list[[k]] <- plot.test
}
plot.CI <- plot_grid(plot.list[[1]],plot.list[[2]],plot.list[[3]],plot.list[[4]],ncol=1
print(plot.CI)
```

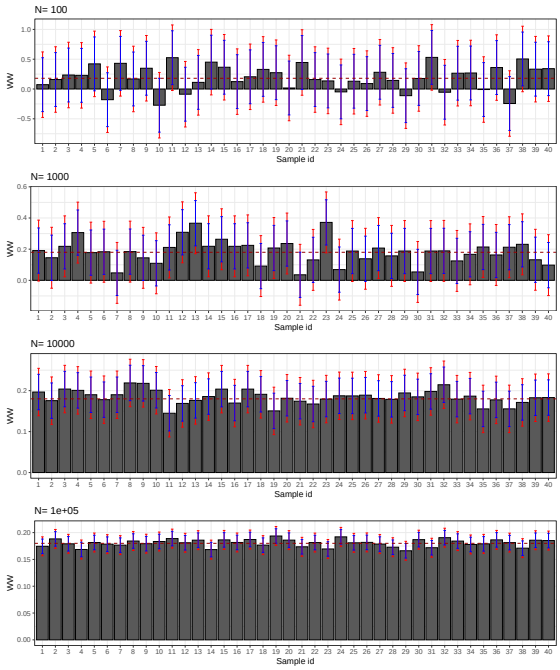


Figure 2.5: Confidence intervals of $\hat{WW}$ for $\delta = 0.99$ (red) and 0.95 (blue) over sample replications for various sample sizes

Figure 2.5 presents the 99% and 95% confidence intervals for 40 samples selected from our simulations. First, confidence intervals do their job: they contain the

true effect most of the time. Second, the 95% confidence interval misses the true effect more often, as expected. For example, with $N = 1000$, the confidence intervals in samples 13 and 23 do not contain the true effect, but it is not far from their lower bound. Third, confidence intervals faithfully reflect what we can learn from our estimates at each sample size. With $N = 100$, the confidence intervals make it clear that the effect might be very large or very small, even strongly negative. With $N = 1000$, the confidence intervals suggest that the effect is either positive or null, but unlikely to be strongly negative. Most of the time, we get the sign right. With $N = 10^4$, we know that the true effect is bigger than 0.1 and smaller than 0.3 and most intervals place the true effect somewhere between 0.11 and 0.25. With $N = 10^5$, we know that the true effect is bigger than 0.15 and smaller than 0.21 and most intervals place the true effect somewhere between 0.16 and 0.20.

2.1.5 Reporting sampling noise: a proposal

Once sampling noise is measured (and we'll see how to get an estimate in the next section), one still has to communicate it to others. There are many ways to report sampling noise:

- Sampling noise as defined in this book ($2 * \epsilon$)
- The corresponding confidence interval
- The signal to noise ratio
- A standard error
- A significance level
- A p-value
- A t-statistic

The main problem with all of these approaches is that they do not express sampling noise in a way that is directly comparable to the magnitude of the TT estimate. Other ways of reporting sampling noise such as p-values and t-stats are nonlinear transforms of sampling noise, making it difficult to really gauge the size of sampling noise as it relates to the magnitude of TT.

My own preference goes to the following format for reporting results: $TT \pm \epsilon$. As such, we can readily compare the size of the noise to the sizee of the TT estimate. We can also form all the other ways of expressing sampling noise directly.

Example 2.4. Let's see how this approach behaves in our numerical example.

```
test.all <- list()
for (k in 1:length(N.sample)){
  set.seed(1234)
  test <- sample(simuls.ww[[k]][,'WW'],N.plot)
  test <- as.data.frame(cbind(test,rep(samp.noise(simuls.ww[[k]][,'WW'],delta=delta)),rep(samp.no
  colnames(test) <- c('WW','sampling.noise.1','sampling.noise.2')
  test$id <- 1:N.plot
  test.all[[k]] <- test
```

```
}
```

With $N = 100$, the reporting of the results for sample 1 would be something like: "we find an effect of 0.07 ± 0.55." Note how the choice of δ does not matter much for the result. The previous result was for $\delta = 0.99$ while the result for $\delta = 0.95$ would have been: "we find an effect of 0.07 ± 0.45." The precise result changes with δ, but the qualitative result stays the same: the magnitude of sampling noise is large and it dwarfs the treatment effect estimate.

With $N = 1000$, the reporting of the results for sample 1 with $\delta = 0.99$ would be something like: "we find an effect of 0.19 ± 0.2." With $\delta = 0.95$: "we find an effect of 0.19 ± 0.15." Again, although the precise quantitative result is affected by the choice of δ, but hte qualitative message stays the same: sampling noise is of the same order of magnitude as the estimated treatment effect.

With $N = 10^4$, the reporting of the results for sample 1 with $\delta = 0.99$ would be something like: "we find an effect of 0.2 ± 0.06." With $\delta = 0.95$: "we find an effect of 0.2 ± 0.04." Again, see how the qualitative result is independent of the precise choice of δ: sampling noise is almost one order of magnitude smaller than the treatment effect estimate.

With $N = 10^5$, the reporting of the results for sample 1 with $\delta = 0.99$ would be something like: "we find an effect of 0.17 ± 0.02." With $\delta = 0.95$: "we find an effect of 0.17 ± 0.01." Again, see how the qualitative result is independent of the precise choice of δ: sampling noise is one order of magnitude smaller than the treatment effect estimate.

Remark. What I hope the example makes clear is that my proposed way of reporting results gives the same importance to sampling noise as it gives to the treatment effect estimate. Also, comparing them is easy, without requiring a huge computational burden on our brain.

Remark. One problem with the approach that I propose is when you have a non-symetric distribution of sampling noise, or when $TT \pm \epsilon$ exceeds natural bounds on TT (such as if the effect cannot be bigger than one, for example). I think these issues are minor and rare and can be dealt with on a case by case basis. The advantage of having one simple and directly readable number comparable to the magnitude of the treatment effect is overwhelming and makes this approach the most natural and adequate, in my opinion.

2.1.6 Using effect sizes to normalize the reporting of treatment effects and their precision

When looking at the effect of a program on an outcome, we depend on the scaling on that outcome to appreciate the relative size of the estimated treatment effect. It is often difficult to appreciate the relative importance of the size of an effect, even if we know the scale of the outcome of interest. One useful device to normalize the treatment effects is called Cohen's d, or effect size. The idea is

to compare the magnitude of the treatment effect to an estimate of the usual amount of variation that the outcome undergoes in the population. The way to build Cohen's d is by dividing the estimated treatment effect by the standard deviation of the outcome. I generally prefer to use the standard devaition of the outcome in the control group, so as not to include the additional amoiunt of variation due to the heterogeneity in treatment effects.

Definition 2.2 (Cohen's d)**.** Cohen's d, or effect size, is the ratio of the estimated treatment effect to the standard deviation of outcomes in the control group:

$$d = \frac{\hat{TT}}{\sqrt{\frac{1}{N^0}\sum_{i=1}^{N^0}(Y_i - \bar{Y^0})^2}}$$

where $\hat{TT}$ is an estimate of the treatment effect, N^0 is the number of individuals in the treatment group and $\bar{Y^0}$ is the average outcome in the treatment group.

Cohen's d can be interpreted in terms of magnitude of effect size:

- It is generally considered that an effect is large when its d is larger than 0.8.
- An effect size around 0.5 is considered medium
- An effect size around 0.2 is considered to be small
- An effect size around 0.02 is considered to be very small.

There probably could be a rescaling of these terms, but that is the actual state of the art.

What I like about effect sizes is that they encourage an interpretation of the order of magnitude of the treatment effect. As such, they enable to include the information on precision by looking at which orders of magnitude are compatible with the estimated effect at the estimated precision level. Effect sizes and orders of magnitude help make us aware that our results might be imprecise, and that the precise value that we have estimated is probably not the truth. What is important is the range of effect sizes compatible with our results (both point estimate and precision).

Example 2.5. Let's see how Cohen's d behaves in our numerical example.

The value of Cohen's d (or effect size) in the population is equal to:

$$ES = \frac{TT}{\sqrt{V^0}} = \frac{\bar{\alpha} + \theta\bar{\mu}}{\sqrt{\sigma_\mu^2 + \rho^2\sigma_U^2 + \sigma_\epsilon^2}}$$

We can write a function to compute this parameter, as well as functions to implement its estimator in the simulated samples:

```
V0 <- function(param){
  return(param["sigma2mu"]+param["rho"]^2*param["sigma2U"]+param["sigma2epsilon"])
}

ES <- function(param){
  return(delta.y.ate(param)/sqrt(V0(param)))
}

samp.noise.ES <- function(estim,delta,param=param){
  return(2*quantile(abs(delta.y.ate(param)/sqrt(V0(param))-estim),prob=delta))
}

for (i in 1:4){
  simuls.ww[[i]][,'ES'] <- simuls.ww[[i]][,'WW']/sqrt(simuls.ww[[i]][,'V0'])
}
```

The true effect size in the population is thus 0.2. It is considered to be small according to the current classification, although I'd say that a treatment able to move the outcomes by 20% of their usual variation is a pretty effective treatment, and this effect should be labelled at least medium. Let's stick with the classification though. In our example, the effect size does not differ much from the treatment effect since the standard deviation of outcomes in the control group is pretty close to one: it is equal to 0.88. Let's now build confidence intervals for the effect size and try to comment on the magnitudes of these effects using the normalized classification.

```
N.plot.ES <- 40
plot.list.ES <- list()

for (k in 1:length(N.sample)){
  set.seed(1234)
  test.ES <- sample(simuls.ww[[k]][,'ES'],N.plot)
  test.ES <- as.data.frame(cbind(test.ES,rep(samp.noise.ES(simuls.ww[[k]][,'ES'],delta=
  colnames(test.ES) <- c('ES','sampling.noise.ES.1','sampling.noise.ES.2')
  test.ES$id <- 1:N.plot.ES
  plot.test.ES <- ggplot(test.ES, aes(x=as.factor(id), y=ES)) +
      geom_bar(position=position_dodge(), stat="identity", colour='black') +
      geom_errorbar(aes(ymin=ES-sampling.noise.ES.1/2, ymax=ES+sampling.noise.ES.1/2),
      geom_errorbar(aes(ymin=ES-sampling.noise.ES.2/2, ymax=ES+sampling.noise.ES.2/2),
      geom_hline(aes(yintercept=ES(param)), colour="#990000", linetype="dashed")+
      #ylim(-0.5,1.2)+
      xlab("Sample id")+
      ylab("Effect Size")+
      theme_bw()+
      ggtitle(paste("N=",N.sample[k]))
  plot.list.ES[[k]] <- plot.test.ES
```

```
}
plot.CI.ES <- plot_grid(plot.list.ES[[1]],plot.list.ES[[2]],plot.list.ES[[3]],plot.list.ES[[4]],r
print(plot.CI.ES)
```

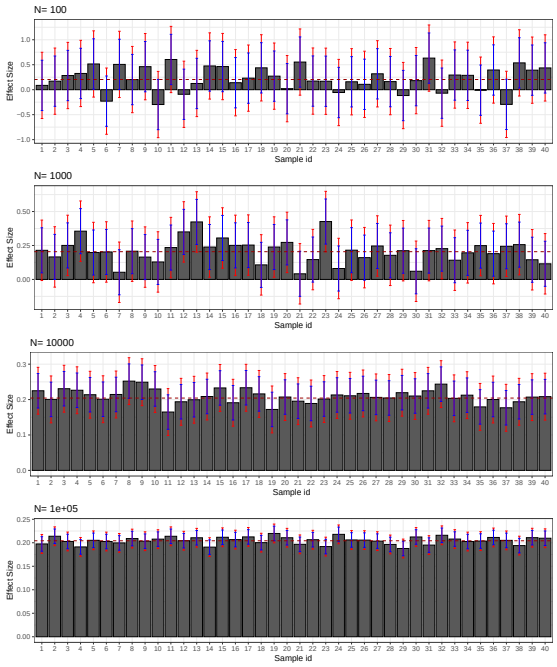


Figure 2.6: Confidence intervals of $\hat{ES}$ for $\delta = 0.99$ (red) and 0.95 (blue) over sample replications for various sample sizes

Figure 2.6 presents the 99% and 95% confidence intervals for the effect size estimated in 40 samples selected from our simulations. Let's regroup our estimate and see how we could present their results.

```
test.all.ES <- list()
for (k in 1:length(N.sample)){
  set.seed(1234)
  test.ES <- sample(simuls.ww[[k]][,'ES'],N.plot)
  test.ES <- as.data.frame(cbind(test.ES,rep(samp.noise.ES(simuls.ww[[k]][,'ES'],delta=delta,para
  colnames(test.ES) <- c('ES','sampling.noise.ES.1','sampling.noise.ES.2')
  test.ES$id <- 1:N.plot.ES
  test.all.ES[[k]] <- test.ES
}
```

With $N = 100$, the reporting of the results for sample 1 would be something like: "we find an effect size of 0.09 ± 0.66" with $\delta = 0,99$. With $\delta = 0.95$ we would say: "we find an effect of 0.09 ± 0.5." All in all, our estimate is compatible with

the treatment having a large positive effect size and a medium negative effect size. Low precision prevents us from saying much else.

With $N = 1000$, the reporting of the results for sample 1 with $\delta = 0.99$ would be something like: "we find an effect size of 0.21 ± 0.22." With $\delta = 0.95$: "we find an effect size of 0.21 ± 0.17." Our estimate is compatible with a medium positive effect or a very small positive or even negative effect (depending on the choice of δ).

With $N = 10^4$, the reporting of the results for sample 1 with $\delta = 0.99$ would be something like: "we find an effect size of 0.22 ± 0.07." With $\delta = 0.95$: "we find an effect size of 0.22 ± 0.05." Our estimate is thus compatible with a small effect of the treatment. We can rule out that the effect of the treatment is medium since the upper bound of the 99% confidence interval is equal to 0.29. We can also rule out that the effect of the treatment is very small since the lower bound of the 99% confidence interval is equal to 0.16. With this sample size, we have been able to reach a precision level sufficient enough to pin down the order of magnitude of the effect size of our treatment. There still remains a considerable amount of uncertainty about the true effect size, though: the upper bound of our confidence interval is almost double the lower bound.

With $N = 10^5$, the reporting of the results for sample 1 with $\delta = 0.99$ would be something like: "we find an effect size of 0.2 ± 0.02." With $\delta = 0.95$: "we find an effect size of 0.2 ± 0.02." Here, the level of precision of our result is such that, first, it does not depend on the choice of δ in any meaningful way, and second, we can do more than pinpoint the order of magnitude of the effect size, we can start to zero in on its precise value. From our estimate, the true value of the effect size is really close to 0.2. It could be equal to 0.18 or 0.22, but not further away from 0.2 than that. Remember that is actually equal to 0.2.

Remark. One issue with Cohen's d is that its magnitude depends on the dispersion of the outcomes in the control group. That means that for the same treatment, and same value of the treatment effect, the effect size is larger in a population where oucomes are more homogeneous. This is not an attractive feature of a normalizing scale that its size depends on the particular application. One solution would be, for each outcome, to provide a standardized scale, using for example the estmated standard deviation in a reference population. This would be similar to the invention of the metric system, where a reference scale was agreed uppon once and for all.

Remark. Cohen's d is well defined for continuous outcomes. For discrete outcomes, the use of Cohen's d poses a series of problems, and alternatives such as relative risk ratios and odds ratios have been proposed. I'll comment on that in the last chapter.

2.2 Estimating sampling noise

Gauging the extent sampling noise is very useful in order to be able to determine how much we should trust our results. Are they precise, so that the true treatment effect lies very close to our estimate? Or are our results imprecise, the true treatment effect maybe lying very far from our estimate?

Estimating sampling noise is hard because we want to infer a property of our estimator over repeated samples using only one sample. In this lecture, I am going to introduce four tools that enable you to gauge sampling noise and to choose sample size. The four tools are Chebyshev's inequality, the Central Limit Theorem, resampling methods and Fisher's permutation method. The idea of all these methods is to use the properties of the sample to infer the properties of our estimator over replications. Chebyshev's inequality gives an upper bound on the sampling noise and a lower bound on sample size, but these bounds are generally too wide to be useful. The Central Limit Theorem (CLT) approximates the distribution of $\hat{E}$ by a normal distribution, and quantifies sampling noise as a multiple of the standard deviation. Resampling methods use the sample as a population and draw new samples from it in order to approximate sampling noise. Fisher's permutation method, also called randomization inference, derives the distribution of $\hat{E}$ under the assumption that all treatment effects are null, by reallocating the treatment indicator among the treatment and control group. Both the CLT and resampling methods are approximation methods, and their approximation of the true extent of sampling noise gets better and better as sample size increases. Fisher's permutation method is exact-it is not an approximation-but it only works for the special case of the WW estimator in a randomized design.

The remaining of this section is structured as follows. Section 2.2.1 introduces the assumptions that we will need in order to implement the methods. Section 2.2.2 presents the Chebyshev approach to gauging sampling noise and choosing sample size. Section 2.2.3 introduces the CLT way of approximating sampling noise and choosing sample size. Section 2.2.4 presents the resampling methods.

Remark. I am going to derive the estimators for the precision only for the WW estimator. In the following lectures, I will show how these methods adapt to other estimators.

2.2.1 Assumptions

In order to be able to use the theorems that power up the methods that we are going to use to gauge sampling noise, we need to make some assumptions on the properties of the data. The main assumptions that we need are that the estimator identifies the true effect of the treatment in the population, that the estimator is well-defined in the sample, that the observations in the sample are independently and identically distributed (i.i.d.), that there is no interaction between units and that the variances of the outcomes in the treated and untreated group are finite.

We know from last lecture that for the WW estimator to identify TT, we need to assume that there is no selection bias, as stated in Assumption 1.7. One way to ensure that this assumption holds is to use a RCT.

In order to be able to form the WW estimator in the sample, we also need that there is at least one treated and one untreated in the sample:

Definition 2.3 (Full rank)**.** We assume that there is at least one observation in the sample that receives the treatment and one observation that does not receive it:

$$\exists i, j \leq N \text{ such that } D_i = 1 \& D_j = 0.$$

One way to ensure that this assumption holds is to sample treated and untreated units.

In order to be able to estimate the variance of the estimator easily, we assume that the observations come from random sampling and are i.i.d.:

Definition 2.4 (i.i.d. sampling)**.** We assume that the observations in the sample are identically and independently distributed:

$$\begin{aligned} \forall i, j \leq N,\, i \neq j,\, (Y_i, D_i) &\perp\!\!\!\perp (Y_j, D_j), \\ (Y_i, D_i) \& (Y_j, D_j) &\sim F_{Y,D}. \end{aligned}$$

We have to assume something on how the observations are related to each other and to the population. Identical sampling is natural in the sense that we are OK to assume that the observations stem from the same population model. Independent sampling is something else altogether. Independence means that the fates of two closely related individuals are assumed to be independent. This rules out two empirically relevant scenarios:

1. The fates of individuals are related because of common influences, as for example the environment, etc,
2. The fates of individuals are related because they directly influence each other, as for example on a market, but also for example because there are diffusion effects, such as contagion of deseases or technological adoption by imitation.

We will address both sources of failure of the independence assumption in future lectures.

Finally, in order for all our derivations to make sense, we need to assume that the outcomes in both groups have finite variances, otherwise sampling noise is going to be too extreme to be able to estimate it using the methods developed in this lecture:

Definition 2.5 (Finite variance of $\Delta^{\hat{Y}}_{WW}$)**.** We assume that $\mathbb{V}[Y^1|D_i = 1]$ and $\mathbb{V}[Y^0|D_i = 0]$ are finite.

2.2.2 Using Chebyshev's inequality

Chebyshev's inequality is a fundamental building block of statistics. It relates the sampling noise of an estimator to its variance. More precisely, it derives an upper bound on the samplig noise of an unbiased estimator:

Theorem 2.2 (Chebyshev's inequality)**.** *For any unbiased estimator $\hat{\theta}$, sampling noise level 2ϵ and confidence level δ, sampling noise is bounded from above:*

$$2\epsilon \leq 2\sqrt{\frac{\mathbb{V}[\hat{\theta}]}{1-\delta}}.$$

Remark. The more general version of Chebyshev's inequality that is generally presented is as follows:

$$\Pr(|\hat{\theta} - \mathbb{E}[\hat{\theta}]| > \epsilon) \leq \frac{\mathbb{V}[\hat{\theta}]}{\epsilon^2}.$$

The version I present in Theorem 2.2 is adapted to the bouding of sampling noise for a given confidence level, while this version is adapted to bounding the confidence level for a given level of sampling noise. In order to go from this general version to Theorem 2.2, simply remember that, for an unbiased estimator, $\mathbb{E}[\hat{\theta}] = \theta$ and that, by definition of sampling noise, $\Pr(|\hat{\theta} - \theta| > \epsilon) = 1 - \delta$. As a result, $1 - \delta \leq \mathbb{V}[\hat{\theta}]/\epsilon^2$, hence the result in Theorem 2.2.

Using Chebyshev's inequality, we can obtain an upper bound on the sampling noise of the WW estimator:

Theorem 2.3 (Upper bound on the sampling noise of $\hat{WW}$)**.** *Under Assumptions 1.7, 2.3 and 2.4, for a given confidence level δ, the sampling noise of the $\hat{WW}$ estimator is bounded from above:*

$$2\epsilon \leq 2\sqrt{\frac{1}{N(1-\delta)}\left(\frac{\mathbb{V}[Y_i^1|D_i=1]}{\Pr(D_i=1)} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{1-\Pr(D_i=1)}\right)} \equiv 2\bar{\epsilon}.$$

Proof. See in Appendix A.1.1 □

Theorem 2.3 is a useful step forward for estimating sampling noise. Theorem 2.3 states that the actual level of sampling noise of the $\hat{WW}$ estimator (2ϵ) is never bigger than a quantity that depends on sample size, confidence level and on the variances of outcomes in the treated and control groups. We either know all the components of the formula for $2\bar{\epsilon}$ or we can estimate them in the sample. For

example, $\Pr(D_i = 1)$, $\mathbb{V}[Y_i^1|D_i = 1]$ and $\mathbb{V}[Y_i^0|D_i = 0]$ by can be approximated by, respectively:

$$\Pr(\hat{D_i} = 1) = \frac{1}{N}\sum_{i=1}^{N} D_i$$

$$\mathbb{V}[Y_i^1|\hat{D}_i = 1] = \frac{1}{\sum_{i=1}^{N} D_i}\sum_{i=1}^{N} D_i(Y_i - \frac{1}{\sum_{i=1}^{N} D_i}\sum_{i=1}^{N} D_i Y_i)^2$$

$$\mathbb{V}[Y_i^0|\hat{D}_i = 0] = \frac{1}{\sum_{i=1}^{N}(1-D_i)}\sum_{i=1}^{N}(1-D_i)(Y_i - \frac{1}{\sum_{i=1}^{N}(1-D_i)}\sum_{i=1}^{N}(1-D_i)Y_i)^2.$$

Using these approximations for the quantities in the formula, we can compute an estimate of the upper bound on sampling noise, $2\hat{\bar{\epsilon}}$.

Example 2.6. Let's write an R function that is going to compute an estimate for the upper bound of sampling noise for any sample:

```
samp.noise.ww.cheb <- function(N,delta,v1,v0,p){
  return(2*sqrt((v1/p+v0/(1-p))/(N*(1-delta))))
}
```

Let's estimate this upper bound in our usual sample:

```
set.seed(1234)
N <-1000
delta <- 0.99
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
Ds <- rep(0,N)
V <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]+param["sigma2U"]))
Ds[V<=log(param["barY"])] <- 1
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
```

In our sample, for $\delta = 0.99$, $2\hat{\bar{\epsilon}} = 1.35$. How does this compare with the true extent of sampling noise when $N = 1000$? Remember that we have computed

an estimate of sampling noise out of our Monte Carlo replications. In Table 2.2, we can read that sampling noise is actually equal to 0.39. The Chebyshev upper bound overestimates the extent of sampling noise by 245%.

How does the Chebyshev upper bound fares overall? In order to know that, let's compute the Chebyshev upper bound for all the simulated samples. You might have noticed that, when running the Monte Carlo simulations for the population parameter, I have not only recovered $\hat{WW}$ for each sample, but also the estimates of the components of the formula for the upper bound on sampling noise. I can thus easily compute the Chebyshev upper bound on sampling noise for each replication.

```
for (k in (1:length(N.sample))){
  simuls.ww[[k]]$cheb.noise <- samp.noise.ww.cheb(N.sample[[k]],delta,simuls.ww[[k]][,'V1'],simul
}
par(mfrow=c(2,2))
for (i in 1:4){
  hist(simuls.ww[[i]][,'cheb.noise'],main=paste('N=',as.character(N.sample[i])),xlab=expression(h
  abline(v=table.noise[i,colnames(table.noise)=='sampling.noise'],col="red")
}
```

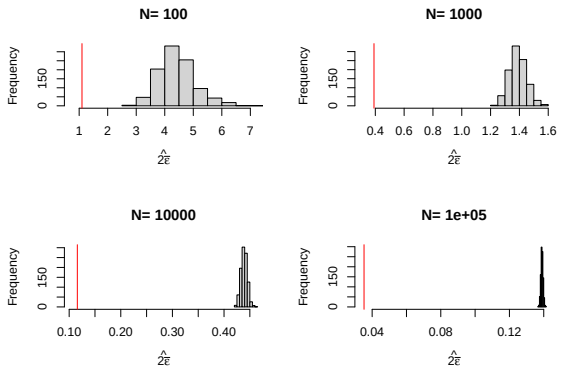


Figure 2.7: Distribution of the Chebyshev upper bound on sampling noise over replications of samples of different sizes (true sampling noise in red)

Figure 2.7 shows that the upper bound works: it is always bigger than the true sampling noise. Figure 2.7 also shows that the upper bound is large: it generally is of an order of magnitude bigger than the true sampling noise, and thus offers a blurry and too pessimistic view of the precision of an estimator. Figure 2.8 shows that the average Chebyshev bound gives an inflated estimate of sampling noise. Figure 2.9 shows that the Chebyshev confidence intervals are clearly less precise than the true unknown ones. With $N = 1000$, the true confidence intervals generally reject large negative effects, whereas the Chebyshev confidence intervals do not rule out this possibility. With $N = 10^4$, the true confidence intervals generally reject effects smaller than 0.1, whereas the Chebyshev confidence intervals cannot rule out small negative effects.

As a conclusion on Chebyshev estimates of sampling noise, their advantage is that they offer an upper bound on the noise: we can never underestimate noise if we use them. A downside of Chebyshev sampling noise estimates is their low precision, which makes it hard to pinpoint the true confidence intervals.

```
for (k in (1:length(N.sample))){
  table.noise$cheb.noise[k] <- mean(simuls.ww[[k]]$cheb.noise)
}
ggplot(table.noise, aes(x=as.factor(N), y=TT)) +
  geom_bar(position=position_dodge(), stat="identity", colour='black') +
  geom_errorbar(aes(ymin=TT-sampling.noise/2, ymax=TT+sampling.noise/2), width=.2,posit
  geom_errorbar(aes(ymin=TT-cheb.noise/2, ymax=TT+cheb.noise/2), width=.2,position=posi
  xlab("Sample Size")+
  theme_bw()
```

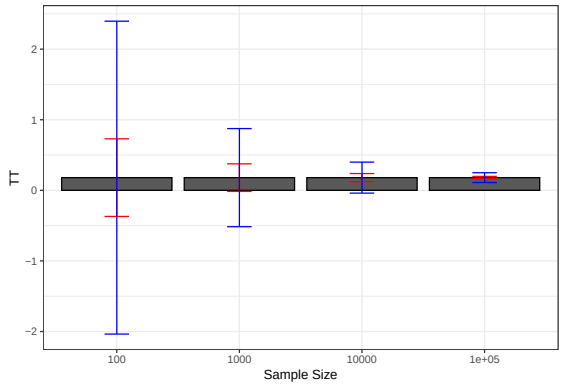


Figure 2.8: Average Chebyshev upper bound on sampling noise over replications of samples of different sizes (true sampling noise in red)

```
N.plot <- 40
plot.list <- list()

for (k in 1:length(N.sample)){
  set.seed(1234)
  test.cheb <- simuls.ww[[k]][sample(N.plot),c('WW','cheb.noise')]
  test.cheb <- as.data.frame(cbind(test.cheb,rep(samp.noise(simuls.ww[[k]][,'WW'],delta
  colnames(test.cheb) <- c('WW','cheb.noise','sampling.noise')
  test.cheb$id <- 1:N.plot
  plot.test.cheb <- ggplot(test.cheb, aes(x=as.factor(id), y=WW)) +
      geom_bar(position=position_dodge(), stat="identity", colour='black') +
      geom_errorbar(aes(ymin=WW-sampling.noise/2, ymax=WW+sampling.noise/2), width=.2,p
      geom_errorbar(aes(ymin=WW-cheb.noise/2, ymax=WW+cheb.noise/2), width=.2,position=
      geom_hline(aes(yintercept=delta.y.ate(param)), colour="#990000", linetype="dashed
      xlab("Sample id")+
```

```
      theme_bw()+
      ggtitle(paste("N=",N.sample[k]))
  plot.list[[k]] <- plot.test.cheb
}
plot.CI <- plot_grid(plot.list[[1]],plot.list[[2]],plot.list[[3]],plot.list[[4]],ncol=1,nrow=leng
print(plot.CI)
```

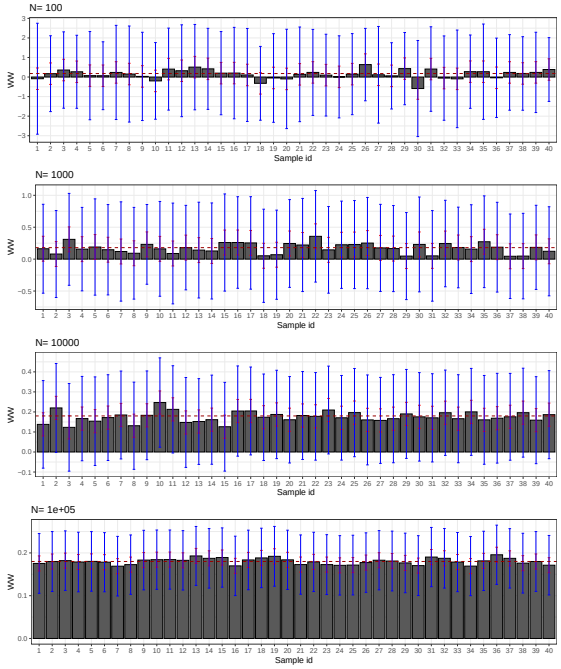


Figure 2.9: Chebyshev confidence intervals of $W\hat{W}$ for $\delta = 0.99$ over sample replications for various sample sizes (true confidence intervals in red)

2.2.3 Using the Central Limit Theorem

The main problem with Chebyshev's upper bound on sampling noise is that it is an upper bound, and thus it overestimates sampling noise and underestimates precision. One alternative to using Chebyshev's upper bound is to use the Central Limit Theorem (CLT). In econometrics and statistics, the CLT is used to derive approximate values for the sampling noise of estimators. Because these approximations become more and more precise as sample size increases, we call them asymptotic approximations.

Taken to its bare bones, the CLT states that the sum of i.i.d. random variables behaves approximately like a normal distribution when the sample size is large:

Theorem 2.4 (Central Limit Theorem)**.** *Let X_i be i.i.d. random variables with*

$\mathbb{E}[X_i] = \mu$ and $\mathbb{V}[X_i] = \sigma^2$, and define $Z_N = \frac{\frac{1}{N}\sum_{i=1}^{N} X_i - \mu}{\frac{\sigma}{\sqrt{N}}}$, then, for all z we have:

$$\lim_{N\to\infty} \Pr(Z_N \leq z) = \Phi(z),$$

where Φ is the cumulative distribution function of the centered standardized normal.

We say that Z_N converges in distribution to a standard normal random variable, and we denote: $Z_N \xrightarrow{d} \mathcal{N}(0,1)$.

The CLT is a beautiful result: the distribution of the average of realisations of any random variable that has finite mean and variance can be approximated by a normal when the sample size is large enough. The CLT is somehow limited though because not all estimators are sums. Estimators are generally more or less complex combinations of sums. In order to derive the asymptotic approximation for a lot of estimators that are combinations of sums, econometricians and statisticians complement the CLT with two other extremely powerful tools: Slutsky's theorem and the Delta method. Slutsky's theorem states that sums, products and ratios of sums that converge to a normal converge to the sum, product or ratio of these normals. The Delta method states that a function of a sum that converges to a normal converges to a normal whose variance is a quadratic form of the variance of the sum and of the first derivative of the function. Both of these tools are stated more rigorously in the appendix, but you do not need to know them for this class. The idea is for you to be aware of how the main approximations that we are going to use throughout this class have been derived.

Let me now state the main result of this section:

Theorem 2.5 (Asymptotic Estimate of Sampling Noise of WW)**.** *Under Assumptions 1.7, 2.3, 2.4 and 2.5, for a given confidence level δ and sample size N, the sampling noise of $\hat{WW}$ can be approximated as follows:*

$$2\epsilon \approx 2\Phi^{-1}\left(\frac{\delta+1}{2}\right)\frac{1}{\sqrt{N}}\sqrt{\frac{\mathbb{V}[Y_i^1|D_i=1]}{\Pr(D_i=1)} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{1-\Pr(D_i=1)}} \equiv 2\tilde{\epsilon}.$$

Proof. See in Appendix A.1.2. □

Let's write an R function that computes this formula:

```
samp.noise.ww.CLT <- function(N,delta,v1,v0,p){
  return(2*qnorm((delta+1)/2)*sqrt((v1/p+v0/(1-p))/N))
}
```

Example 2.7. Let's see how the CLT performs in our example.

In our sample, for $\delta = 0.99$, the CLT estimate of sampling noise is $2\hat{\tilde{\epsilon}} = 0.35$. How does this compare with the true extent of sampling noise when $N = 1000$? Remember that we have computed an estimate of sampling noise out of our Monte Carlo replications. In Table 2.1, we can read that sampling noise is actually equal to 0.39. The CLT approximation is pretty precise: it only underestimates the true extent of sampling noise by 11%.

We can also compute the CLT approximation to sampling noise in all of our samples:

```
for (k in (1:length(N.sample))){
  simuls.ww[[k]]$CLT.noise <- samp.noise.ww.CLT(N.sample[[k]],delta,simuls.ww[[k]][,'V1'],simuls.
}
par(mfrow=c(2,2))
for (i in 1:4){
  hist(simuls.ww[[i]][,'CLT.noise'],main=paste('N=',as.character(N.sample[i])),xlab=expression(ha
  abline(v=table.noise[i,colnames(table.noise)=='sampling.noise'],col="red")
}
```

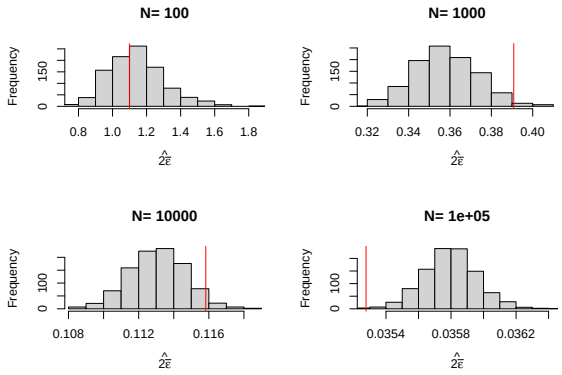


Figure 2.10: Distribution of the CLT approximation of sampling noise over replications of samples of different sizes (true sampling noise in red)

Figure 2.10 shows that the CLT works: CLT-based estimates of sampling noise approximates true sampling noise well. CLT-based approximations of sampling noise are even impressively accurate: they always capture the exact order of magnitude of sampling noise, although there is a slight underestimation when $N = 1000$ and 10^4 and a slight overestimation when $N = 10^5$. This success should not come as a surprise as all shocks in our model are normally distributed, meaning that the CLT results are more than an approximation, they are exact. Results might be less spectacular when estimating the effect of the treatment on the outcomes in levels rather than in logs.

As a consequence, the average CLT-based estimates of sampling noise and of confidence intervals are pretty precise, as Figures 2.11 and 2.12 show. Let's

pause for a second at the beauty of what we have achieved using the CLT: by using only information from one sample, we have been able to gauge extremely precisely how the estimator would behave over sampling repetitions.

```
for (k in (1:length(N.sample))){
  table.noise$CLT.noise[k] <- mean(simuls.ww[[k]]$CLT.noise)
}
ggplot(table.noise, aes(x=as.factor(N), y=TT)) +
  geom_bar(position=position_dodge(), stat="identity", colour='black') +
  geom_errorbar(aes(ymin=TT-sampling.noise/2, ymax=TT+sampling.noise/2), width=.2,posit
  geom_errorbar(aes(ymin=TT-CLT.noise/2, ymax=TT+CLT.noise/2), width=.2,position=positi
  xlab("Sample Size")+
  theme_bw()
```

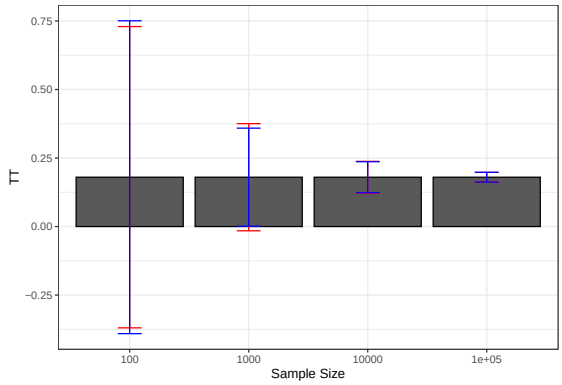


Figure 2.11: Average CLT-based approximations of sampling noise over replications of samples of different sizes (true sampling noise in red)

```
N.plot <- 40
plot.list <- list()

for (k in 1:length(N.sample)){
  set.seed(1234)
  test.CLT <- simuls.ww[[k]][sample(N.plot),c('WW','CLT.noise')]
  test.CLT <- as.data.frame(cbind(test.CLT,rep(samp.noise(simuls.ww[[k]][,'WW'],delta=
  colnames(test.CLT) <- c('WW','CLT.noise','sampling.noise')
  test.CLT$id <- 1:N.plot
  plot.test.CLT <- ggplot(test.CLT, aes(x=as.factor(id), y=WW)) +
      geom_bar(position=position_dodge(), stat="identity", colour='black') +
      geom_errorbar(aes(ymin=WW-sampling.noise/2, ymax=WW+sampling.noise/2), width=.2,
      geom_errorbar(aes(ymin=WW-CLT.noise/2, ymax=WW+CLT.noise/2), width=.2,position=po
      geom_hline(aes(yintercept=delta.y.ate(param)), colour="#990000", linetype="dashe
      xlab("Sample id")+
      theme_bw()+
```

```
      ggtitle(paste("N=",N.sample[k]))
  plot.list[[k]] <- plot.test.CLT
}
plot.CI <- plot_grid(plot.list[[1]],plot.list[[2]],plot.list[[3]],plot.list[[4]],ncol=1,nrow=leng
print(plot.CI)
```

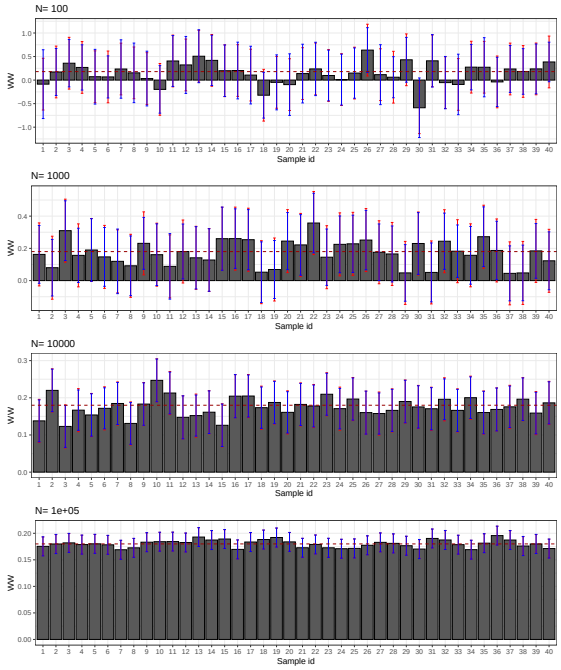


Figure 2.12: CLT-based confidence intervals of $\hat{WW}$ for $\delta = 0.99$ over sample replications for various sample sizes (true confidence intervals in red)

Remark. In proving the main result on the asymptotic distribution of $\hat{WW}$, we have also proved a very useful result: $\hat{WW}$ is the Ordinary Least Squares (OLS) estimator of β in the regression $Y_i = \alpha + \beta D_i + U_i$. This is pretty cool since we now can use our classical OLS estimator in our statistical package to estimate $\hat{WW}$. Let's compute the OLS estimate of WW in our sample:

```
ols.ww <- lm(y~Ds)
ww.ols <- ols.ww$coef[[2]]
```

We have $\hat{WW}_{OLS} = 0.13 = 0.13 = \hat{WW}$.

Remark. Another pretty cool consequence of Theorem 2.5 and of its proof is that the standard error of the OLS estimator of $\hat{WW}$ (σ_β) is related to the sampling noise of $\hat{WW}$ by the following formula: $2\tilde{\epsilon} = 2\Phi^{-1}\left(\frac{\delta+1}{2}\right)\sigma_\beta$.

This implies that sampling noise is equal to 5 σ_β when $\delta = 0.99$ and to 4 σ_β

when $\delta = 0.95$. It is thus very easy to move from estimates of the standard error of the β coefficient to the extent of sampling noise.

Remark. A last important consequence of Theorem 2.5 and of its proof is that the standard error of the OLS estimator of $W\hat{W}$ (σ_β) that we use is the heteroskedasticity-robust one.

Using the RCM, we can indeed show that:

$$\begin{aligned}
\alpha &= \mathbb{E}[Y_i^0|D_i = 0] \\
\beta &= \Delta_{TT}^Y \\
U_i &= Y_i^0 - \mathbb{E}[Y_i^0|D_i = 0] + D_i(\Delta_i^Y - \Delta_{TT}^Y),
\end{aligned}$$

Under Assumption 1.7, we have:

$$U_i = (1 - D_i)(Y_i^0 - \mathbb{E}[Y_i^0|D_i = 0]) + D_i(Y_i^1 - \mathbb{E}[Y_i^1|D_i = 1])$$

There is heteroskedasticity because the outcomes of the treated and of the untreated have different variances:

$$\begin{aligned}
\mathbb{V}[U_i|D_i = d] &= \mathbb{E}[U_i^2|D_i = d] \\
&= \mathbb{E}[(Y_i^d - \mathbb{E}[Y_i^d|D_i = d])^2|D_i = d] \\
&= \mathbb{V}[Y_i^d|D_i = d]
\end{aligned}$$

We do not want to assume homoskedasticity, since it would imply a constant treatment effect. Indeed, $\mathbb{V}[Y_i^1|D_i = 1] = \mathbb{V}[Y_i^0|D_i = 1] + \mathbb{V}[\alpha_i|D_i = 1]$.

Remark. In order to estimate the heteroskedasticity robust standard error from the OLS regression, we can use the sandwich package in R Most available heteroskedasticity robust estimators based on the CLT can be written in the following way:

$$\mathbb{V}[\hat{\Theta}_{OLS}] \approx (X'X)^{-1}X'\hat{\Omega}X(X'X)^{-1},$$

where X is the matrix of regressors and $\hat{\Omega} = \text{diag}(\hat{\sigma}_{U_1}^2, \ldots, \hat{\sigma}_{U_N}^2)$ is an estimate the covariance matrix of the residuals U_i. Here are various classical estimators for $\hat{\Omega}$:

$$
\begin{aligned}
&\text{HC0:} && \hat{\sigma_{U_i}}^2 = \hat{U_i}^2 \\
&\text{HC1:} && \hat{\sigma_{U_i}}^2 = \frac{N}{N-K}\hat{U_i}^2 \\
&\text{HC2:} && \hat{\sigma_{U_i}}^2 = \frac{\hat{U_i}^2}{1-h_i} \\
&\text{HC3:} && \hat{\sigma_{U_i}}^2 = \frac{\hat{U_i}^2}{(1-h_i)^2},
\end{aligned}
$$

where $\hat{U}_i$ is the residual from the OLS regression, K is the number of regressors, h_i is the leverage of observation i, and is the i^{th} diagonal element of $H = X(X'X)^{-1}X'$. HC1 is the one reported by Stata when using the 'robust' option.

Example 2.8. Using the sandwich package, we can estimate the heteroskedasticity-robust variance-covariance matrix and sampling noise as follows:

```
ols.ww.vcov.HC0 <- vcovHC(ols.ww, type = "HC0")
samp.noise.ww.CLT.ols <- function(delta,reg,...){
  return(2*qnorm((delta+1)/2)*sqrt(vcovHC(reg,...)[2,2]))
}
```

For $\delta = 0.99$, sampling noise estimated using the "HC0" option is equal to 0.35. This is exactly the value we have estimated using our CLT-based formula ($2\widehat{\epsilon} = 0.35$). Remember that sampling noise is actually equal to 0.39. Other "HC" options might be better in small samples. For example, with the "HC1" option, we have an estimate for sampling noise of 0.35. What would have happened to our estimate of sampling noise if we had ignored heteroskedasticity? The default OLS standard error estimate yields an estimate for sampling noise of 0.36.

2.2.4 Using resampling methods

The main intuition behind resampling methods is to use the sample as a population, to draw samples from it and compute our estimator on each of these samples in order to gauge its variability over sampling repetitions. There are three main methods of resampling that work that way: bootstrapping, radomization inference and subsampling. Bootstrapping draws samples with replacement, so that each sample has the same size as the original sample. Subsampling draws samples without replacement, thereby the samples are of a smaller size than the original one. Randomization inference keeps the same sample in all repetitions, but changes the allocation of the treatment.

Why would we use resampling methods instead of CLT-based standard errors? There are several possible reasons:

1. Asymptotic refinements: sometimes, resampling methods are more precise in small samples than the CLT-based asymptotic approaches. In that case, we say that resampling methods offer asymptotic refinements.
2. Ease of computation: for some estimators, the CLT-based estimates of sampling noise are complex or cumbersome to compute, whereas resampling methods are only computationally intensive.
3. Inexistence of CLT-based estimates of sampling noise: some estimators do not have any CLT-based estimates of sampling noise yet. That was the case for the Nearest-Neighbour Matching estimator (NNM) for a long time for example. It still is the case for the Synthetic Control Method estimator. Beware though that the bootstrap is not valid for all estimators. For example, it is possible to show that the bootstrap is invalid for NNM. Subsampling is valid for NNM though (see Abadie and Imbens, 2006).

2.2.4.1 Bootstrap

The basic idea of the bootstrap is to use Monte Carlo replications to draw samples from the original sample with replacement. Then, at each replication, we compute the value of our estimator $\hat{E}$ on the new sample. Let's call this new value $\hat{E}_k^*$ for bootstrap replication k. Under certain conditions, the distribution of $\hat{E}_k^*$ approximates the distribution of $\hat{E}$ over sample repetitions very well, and all the more so as the sample size gets large.

What are the conditions under which the bootstrap is going to provide an accurate estimation of the distribution of $\hat{E}$? Horowitz (2001) reports on a very nice result by Mammen that makes these conditions clear:

Theorem 2.6 (Mammen (1992))**.** *Let $\{X_i : i = 1, \dots, N\}$ be a random sample from a population. For a sequence of functions g_N and sequences of numbers t_N and σ_N, define $\bar{g}_N = \frac{1}{N}\sum_{i=1}^{N} g_N(X_i)$ and $T_N = (\bar{g}_N - t_N)/\sigma_N$. For the bootstrap sample $\{X_i^* : i = 1, \dots, N\}$, define $\bar{g}_N^* = \frac{1}{N}\sum_{i=1}^{N} g_N(X_i^*)$ and $T_N^* = (\bar{g}_N^* - \bar{g}_N)/\sigma_N$. Let $G_N(\tau) = \Pr(T_N \leq \tau)$ and $G_N^*(\tau) = \Pr(T_N^* \leq \tau)$, where this last probability distribution is taken over bootstrap sampling replications. Then G_N^* consistently estimates G_N if and only if $T_N \xrightarrow{d} \mathcal{N}(0, 1)$.*

Theorem 2.6 states that the bootstrap will offer a consistent estimation of the distribution of a given estimator if and only if this estimator is asymptotically normally distributed. It means that we could theoretically use the CLT-based asymptotic distribution to compute sampling noise. So, and it demands to be stronlgy emphasized, **the bootstrap is not valid when the CLT fails**.

How do we estimate sampling noise with the bootstrap? There are several ways to do so, but I am going to emphasize the most widespread here, that is known as the percentile method. Let's define $E^*_{\frac{1-\delta}{2}}$ and $E^*_{\frac{1+\delta}{2}}$ as the corresponding quantiles of the bootstrap distribution of $\hat{E}_k^*$ over a large number K of replications. The bootstrapped sampling noise using the percentile method is simply the distance between these two quantities.

Theorem 2.7 (Bootstrapped Estimate of Sampling Noise of WW)**.** *Under Assumptions 1.7, 2.3, 2.4 and 2.5, for a given confidence level δ and sample size N, the sampling noise of $\hat{WW}$ can be approximated as follows:*

$$2\epsilon \approx E^*_{\frac{1+\delta}{2}} - E^*_{\frac{1-\delta}{2}} \equiv 2\hat{\epsilon}^b.$$

Proof. The WW estimator can be written as a sum:

$$\Delta^{\hat{Y}}_{WW} = \frac{1}{N}\sum_{i=1}^{N} \frac{\left(Y_i - \frac{1}{N}\sum_{i=1}^N Y_i\right)\left(D_i - \frac{1}{N}\sum_{i=1}^N D_i\right)}{\frac{1}{N}\sum_{i=1}^N \left(D_i - \frac{1}{N}\sum_{i=1}^N D_i\right)^2}.$$

Using Lemma A.5, we know that the WW estimator is asymptotically normal under Assumptions 1.7, 2.3, 2.4 and 2.5. Using Theorem 2.6 proves the result. □

Remark. With the bootstrap, we are not going to define the confidence interval using Theorem 2.1 but directly using $\left\{E^*_{\frac{1-\delta}{2}}; E^*_{\frac{1+\delta}{2}}\right\}$. Indeed, we have defined the bootstrapped estimator of sampling noise by using the asymetric confidence interval. We could have used the equivalent of Definition 2.1 on the bootstrapped samples to compute sampling noise using the symmetric confidence interval. Both are feasible and similar in large samples, since the asymptotic distribution is symmetric. One advantage of asymetric confidence intervals is that they might capture deviations from the normal distribution in small samples. These advantages are part of what we call asymptotic refinements. Rigorously, though, asymptotic refinements have not been proved to exist for the percentile method but only for the method bootstrapping asymptotically pivotal quantities.

Remark. We say that a method brings asymptotic refinements if it increases the precision when estimating sampling noise and confidence intervals relative to the asymptotic CLT-based approximation. The bootstrap has been shown rigorously to bring asymptotic refinements when used to estimate the distribution of asymptotically pivotal statistic. An asymptotically pivotal statistic is a statistic that can be computed from the sample but that, asymptotically, converges to a quantity that does not depend on the sample, like for example a standard normal. Using Lemma A.5, we know for example that the following statistic is asymptotically normal:

$$T^{WW}_N = \frac{\Delta^{\hat{Y}}_{WW} - \Delta^{Y}_{TT}}{\sqrt{\frac{\frac{\mathbb{V}[Y_i^1|D_i=1]}{\Pr(D_i=1)} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{1-\Pr(D_i=1)}}{N}}} \xrightarrow{d} \mathcal{N}(0,1).$$

To build a confidence interval bootstrapping T_N^{WW}, compute an estimator of T_N^{WW} for each bootstrapped sample, say $\hat{T}_{N,k}^{WW*}$. You can for example use the OLS estimator in the bootstrapped sample, with a heteroskedasticity-robust standard error estimator. Or you can compute the WW estimator by hand in the sample along with an estimator of its variance using the variance of the outcomes in the treated and control groups. You can then estimate the confidence interval as follows: $\left\{\hat{\Delta^Y_{WW}} - \sigma_{\hat{WW}}\hat{T}^{WW*}_{N,\frac{1-\delta}{2}}; \hat{\Delta^Y_{WW}} + \sigma_{\hat{WW}}\hat{T}^{WW*}_{N,\frac{1+\delta}{2}}\right\}$, where $\hat{T}^{WW*}_{N,q}$ iq the q^{th} quantile of the distribution of $\hat{T}^{WW*}_{N,k}$ over sampling replications and $\sigma_{\hat{WW}}$ is an estimate of the variance of $\hat{\Delta^Y_{WW}}$ (either the CLT-based approximation of the bootstrapped one, see below).

Remark. One last possibility to develop an estimator for sampling noise and confidence interval is to use the bootstrap in order to estimate the variance of the estimator $\hat{E}$, $\hat{\sigma^2_E}$, and then use it to compute sampling noise. If $\hat{E}$ is asymptotically normally distributed, we have that sampling noise is equal to $2\Phi^{-1}\left(\frac{\delta+1}{2}\right)\hat{\sigma_E}$. You can use the usual formula from Theore 2.1 to compute the confidence interval. The bootstrapped variance of $\hat{E}$, $\hat{\sigma^2_E}$, is simply the variance of $\hat{E}^*_k$ over bootstrap replications.

Example 2.9. In the numerical example, I am going to derive the bootstrapped confidence intervals and sampling noise for the percentile method. Let's first put the dataset from our example in a nice data frame format so that resampling is made easier. We then define a function taking a number of bootstrapped replications and spitting out sampling noise and confidence intervals.

```
data <- as.data.frame(cbind(y,Ds,yB))
boot.fun.ww.1 <- function(seed,data){
  set.seed(seed,kind="Wichmann-Hill")
  data <- data[sample(nrow(data),replace = TRUE),]
  ols.ww <- lm(y~Ds,data=data)
  ww <- ols.ww$coef[[2]]
  return(ww)
}

boot.fun.ww <- function(Nboot,data){
  #sfInit(parallel=TRUE,cpus=8)
  boot <- lapply(1:Nboot,boot.fun.ww.1,data=data)
  #sfStop()
  return(unlist(boot))
}

boot.CI.ww <- function(boot,delta){
  return(c(quantile(boot,prob=(1-delta)/2),quantile(boot,prob=(1+delta)/2)))
}
```

```
boot.samp.noise.ww <- function(boot,delta){
  return(quantile(boot,prob=(1+delta)/2)-quantile(boot,prob=(1-delta)/2))
}

Nboot <- 500
ww.boot <- boot.fun.ww(Nboot,data)
ww.CI.boot <- boot.CI.ww(ww.boot,delta)
ww.samp.noise.boot <- boot.samp.noise.ww(ww.boot,delta)
```

Over 500 replications, the 99% bootstrapped confidence interval using the percentile method is $\{-0.023; 0.316\}$. As a consequence, the bootstrapped estimate of 99% sampling noise is of 0.339. Remember that, with $N = 1000$, sampling noise is actually equal to 0.39.

In order to assess the global precision of bootstrapping, we are going to resort to Monte Carlo simulations. For each Monte Carlo sample, we are going to estimate sampling noise and confidence intervals using the bootstrap. As you can imagine, this is going to prove rather computationally intensive. I cannot use parallelization twice: I have to choose whether to parallelize the Monte Carlo simulations or the bootstrap simulations. I have choosen to parallelize the outer loop, so that a given job takes longer on each cluster.

```
monte.carlo.ww.boot <- function(s,N,param,Nboot,delta){
  set.seed(s)
  mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
  UB <- rnorm(N,0,sqrt(param["sigma2U"]))
  yB <- mu + UB
  YB <- exp(yB)
  Ds <- rep(0,N)
  V <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]+param["sigma2U"]))
  Ds[V<=log(param["barY"])] <- 1
  epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
  eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
  U0 <- param["rho"]*UB + epsilon
  y0 <- mu +  U0 + param["delta"]
  alpha <- param["baralpha"]+  param["theta"]*mu + eta
  y1 <- y0+alpha
  Y0 <- exp(y0)
  Y1 <- exp(y1)
  y <- y1*Ds+y0*(1-Ds)
  Y <- Y1*Ds+Y0*(1-Ds)
  data <- as.data.frame(cbind(y,Ds,yB))
  ww.boot <- boot.fun.ww(Nboot,data)
  ww.CI.boot <- boot.CI.ww(ww.boot,delta)
  ww.samp.noise.boot <- boot.samp.noise.ww(ww.boot,delta)
  return(c((1/sum(Ds))*sum(y*Ds)-(1/sum(1-Ds))*sum(y*(1-Ds)),var(y[Ds==1]),var(y[Ds==0]),mean(Ds)
```

```
}

sf.simuls.ww.N.boot <- function(N,Nsim,Nboot,delta,param){
  sfInit(parallel=TRUE,cpus=2*ncpus)
  sfExport("boot.fun.ww","boot.CI.ww","boot.samp.noise.ww","boot.fun.ww.1")
  sim <- as.data.frame(matrix(unlist(sfLapply(1:Nsim,monte.carlo.ww.boot,N=N,Nboot=Nboo
  sfStop()
  colnames(sim) <- c('WW','V1','VO','p','boot.lCI','boot.uCI','boot.samp.noise')
  return(sim)
}

simuls.ww.boot <- lapply(N.sample,sf.simuls.ww.N.boot,Nsim=Nsim,param=param,Nboot=Nboot
```

We can now graph our bootstrapped estimate of sampling noise in all of our samples, the average bootstrapped estimates of sampling noise and of confidence intervals, in Figures 2.13, 2.14 and 2.15 show.

```
par(mfrow=c(2,2))
for (i in 1:4){
  hist(simuls.ww.boot[[i]][,'boot.samp.noise'],main=paste('N=',as.character(N.sample[i]
  abline(v=table.noise[i,colnames(table.noise)=='sampling.noise'],col="red")
}
```

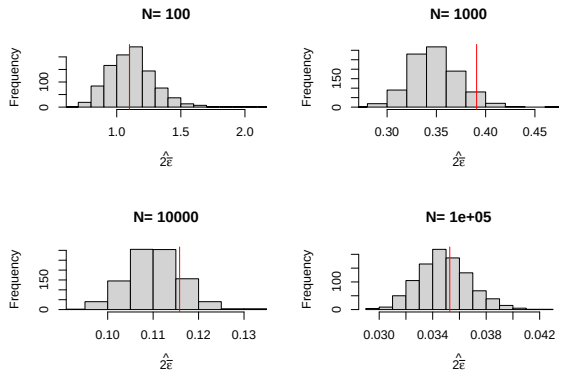


Figure 2.13: Distribution of the bootstrapped approximation of sampling noise over replications of samples of different sizes (true sampling noise in red)

```
for (k in (1:length(N.sample))){
  table.noise$boot.noise[k] <- mean(simuls.ww.boot[[k]]$boot.samp.noise)
}
ggplot(table.noise, aes(x=as.factor(N), y=TT)) +
  geom_bar(position=position_dodge(), stat="identity", colour='black') +
  geom_errorbar(aes(ymin=TT-sampling.noise/2, ymax=TT+sampling.noise/2), width=.2,posit
  geom_errorbar(aes(ymin=TT-boot.noise/2, ymax=TT+boot.noise/2), width=.2,position=posi
  xlab("Sample Size")+
```

```
  theme_bw()
```

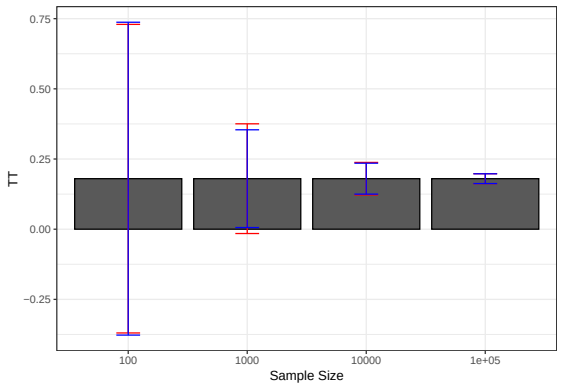


Figure 2.14: Average bootstrapped approximations of sampling noise over replications of samples of different sizes (true sampling noise in red)

```
N.plot <- 40
plot.list <- list()

for (k in 1:length(N.sample)){
 set.seed(1234)
 test.boot <- simuls.ww.boot[[k]][sample(N.plot),c('WW','boot.lCI','boot.uCI')]
 test.boot <- as.data.frame(cbind(test.boot,rep(samp.noise(simuls.ww.boot[[k]][,'WW'],delta=delta
 colnames(test.boot) <- c('WW','boot.lCI','boot.uCI','sampling.noise')
 test.boot$id <- 1:N.plot
 plot.test.boot <- ggplot(test.boot, aes(x=as.factor(id), y=WW)) +
     geom_bar(position=position_dodge(), stat="identity", colour='black') +
     geom_errorbar(aes(ymin=WW-sampling.noise/2, ymax=WW+sampling.noise/2), width=.2,position=pos
     geom_errorbar(aes(ymin=boot.lCI, ymax=boot.uCI), width=.2,position=position_dodge(.9),color=
     geom_hline(aes(yintercept=delta.y.ate(param)), colour="#990000", linetype="dashed")+
     xlab("Sample id")+
     theme_bw()+
     ggtitle(paste("N=",N.sample[k]))
 plot.list[[k]] <- plot.test.boot
}
plot.CI <- plot_grid(plot.list[[1]],plot.list[[2]],plot.list[[3]],plot.list[[4]],ncol=1,nrow=leng
print(plot.CI)
```

TO DO: COMMENT AND USE PIVOTAL TEST STATISTIC

2.2.4.2 Randomization inference

Randomization inference (a.k.a. Fisher's permutation approach) tries to mimick the sampling noise due to the random allocation of the treatment vector, as we

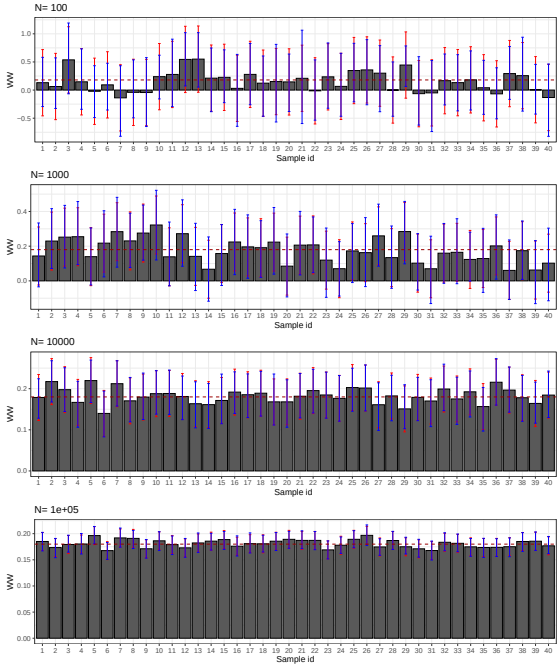


Figure 2.15: Bootstrapped confidence intervals of $W\hat{W}$ for $\delta = 0.99$ over sample replications for various sample sizes (true confidence intervals in red)

have seen in Section 2.1.3. In practice, the idea is simply to look at how the treatment effect that we estimate varies when we visit all the possible allocations of the treament dummy in the sample. For each new allocation, we are going to compute the with/without estimator using the observed outcomes and the newly allocated treatment dummy. It means that some actually treated observations are going to enter into the computation of the control group mean, while some actually untreated observations are going to enter into the computation of the treatment group mean. As a consequence, the resulting distribution will be centered at zero. Under the assumption of a constant treatment effect, the distribution of the parameter obtained using randomization inference will be an exact estimation of sampling noise for the sample treatment effect.

Notice how beautilful the result is: randomization inference yields an **exact** measure of sampling noise. The resulting estimate of sampling noise is not an approximation that is going to become better as sample size increases. No, it is the **actual** value of sampling noise in the sample.

There are two ways to compute a confidence interval using Fisher's permutation approach. One is to form symmetric intervals using our estimate of sampling noise as presented in Section 2.1.4. Another approach is to directly use the quantiles of the distribution of the parameter centered around the estimated treatment effect, in the same spirit as bootstrapped confidence intervals using the percentile approach. This last approach accomodates possible asymetries in the finite sample distribution of the treatment effect.

Computing the value of the treatment effect for all possible treatment allocations can take a lot of time with large samples. That's why we in general compute the test statistic for a reasonably large number of random allocations.

Remark. Fisher's original approach is slightly different from the one I delineate here. Fisher wanted to derive a test statistic for whether the treatment effect was zero, not to estimate sampling noise. Under the null that the treatment has absolutely no effect whatsoever on any unit, any test statistic whose value should be zero if the two distributions where identical can be computed on the actual sample and its distribution can be derived using Fisher's permutation approach. The test statistic can be the difference in means, standard deviations, medians, ranks, the T-stat, the Kolmogorov-Smirnov test statistic or any other test statistic that you might want to compute. Comparing the actual value of the test statistic to its distribution under the null gives a p-value for the validity of the null.

Remark. Imbens and Rubin propose a more complex procedure to derive the confidence interval for the treatment effect using randomization inference. They propose to compute Fisher's p-value for different values of the treatment effect, and to set the confidence interval as the values of the treatment effect under and above which the p-value is smaller than δ. When using the with/without estimator as the test statistic, the two approches should be equivalent. Is is possible that the estimates using statistics less influenced by outliers are more

precise though.

Remark. Note that we pay two prices for having an exact estimation of sampling noise:

1. We have to assume that the treatment effect is constant, e.g. we have to assume homoskedasticity. This is in general not the case. Whether this is in general a big issue depends on how large the difference is between homoskedastic and heteroskedastic standard errors. One way around this issue would be to add a small amount of noise to the observations that are in the group with the lowest variance. Whether this would work in practice is still to be demonstrated.
2. We have to be interested only in the sampling noise of the sample treatment effect. The sampling noise of the population treatment effect is not estimated using Fisher's permutation approach. As we have seen in Section 2.1.3, there is no practical difference between these two sampling noises in our example. Whether this is the case in general deserves further investigation.

Example 2.10. In practice, randomization inference is very close to a bootstrap procedure, except that instead of resampling with replacement from the original sample, we only change the vector of treatment allocation at each replication.

```
fisher.fun.ww.1 <- function(seed,data){
  set.seed(seed,kind="Wichmann-Hill")
  data$D <- rbinom(nrow(data),1,mean(data$Ds))
  ols.ww <- lm(y~D,data=data)
  ww <- ols.ww$coef[[2]]
  return(ww)
}

fisher.fun.ww <- function(Nfisher,data,delta){
  fisher <- unlist(lapply(1:Nfisher,fisher.fun.ww.1,data=data))
  ols.ww <- lm(y~Ds,data=data)
  ww <- ols.ww$coef[[2]]
  fisher <- fisher+ ww
  fisher.CI.ww <- c(quantile(fisher,prob=(1-delta)/2),quantile(fisher,prob=(1+delta)/2)
  fisher.samp.noise.ww <- quantile(fisher,prob=(1+delta)/2)-quantile(fisher,prob=(1-del
  return(list(fisher,fisher.CI.ww,fisher.samp.noise.ww))
}

Nfisher <- 500
ww.fisher <- fisher.fun.ww(Nfisher,data,delta)
```

Over 500 replications, the 99% confidence interval based on Fisher's permutation approach is $\{-0.052; 0.3\}$. As a consequence, the bootstrapped estimate of 99% sampling noise is of 0.352. Remember that, with $N = 1000$, sampling noise is actually equal to 0.39.

In order to assess the global precision of Fisher's permutation method, we are going to resort to Monte Carlo simulations.

```
monte.carlo.ww.fisher <- function(s,N,param,Nfisher,delta){
  set.seed(s)
  mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
  UB <- rnorm(N,0,sqrt(param["sigma2U"]))
  yB <- mu + UB
  YB <- exp(yB)
  Ds <- rep(0,N)
  V <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]+param["sigma2U"]))
  Ds[V<=log(param["barY"])] <- 1
  epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
  eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
  U0 <- param["rho"]*UB + epsilon
  y0 <- mu +  U0 + param["delta"]
  alpha <- param["baralpha"]+  param["theta"]*mu + eta
  y1 <- y0+alpha
  Y0 <- exp(y0)
  Y1 <- exp(y1)
  y <- y1*Ds+y0*(1-Ds)
  Y <- Y1*Ds+Y0*(1-Ds)
  data <- as.data.frame(cbind(y,Ds,yB))
  ww.fisher <- fisher.fun.ww(Nfisher,data,delta)
  return(c((1/sum(Ds))*sum(y*Ds)-(1/sum(1-Ds))*sum(y*(1-Ds)),var(y[Ds==1]),var(y[Ds==0]),mean(Ds)
}

sf.simuls.ww.N.fisher <- function(N,Nsim,Nfisher,delta,param){
  sfInit(parallel=TRUE,cpus=2*ncpus)
  sfExport("fisher.fun.ww","fisher.fun.ww.1")
  sim <- as.data.frame(matrix(unlist(sfLapply(1:Nsim,monte.carlo.ww.fisher,N=N,Nfisher=Nfisher,de
  sfStop()
  colnames(sim) <- c('WW','V1','V0','p','fisher.lCI','fisher.uCI','fisher.samp.noise')
  return(sim)
}

simuls.ww.fisher <- lapply(N.sample,sf.simuls.ww.N.fisher,Nsim=Nsim,param=param,Nfisher=Nfisher,d
```

We can now graph our bootstrapped estimate of sampling noise in all of our samples, the average bootstrapped estimates of sampling noise and of confidence intervals, as Figures 2.16, 2.17 and 2.18 show. The results are pretty good. On average, estimates of sampling noise using Randomization Inference are pretty accurate, as Figure 2.17 shows. It seems that sampling noise is underestimated by Randomization Inference when $N = 1000$, without any clear reason why.

```
par(mfrow=c(2,2))
for (i in 1:4){
```

```
  hist(simuls.ww.fisher[[i]][,'fisher.samp.noise'],main=paste('N=',as.character(N.sampl
  abline(v=table.noise[i,colnames(table.noise)=='sampling.noise'],col="red")
}
```

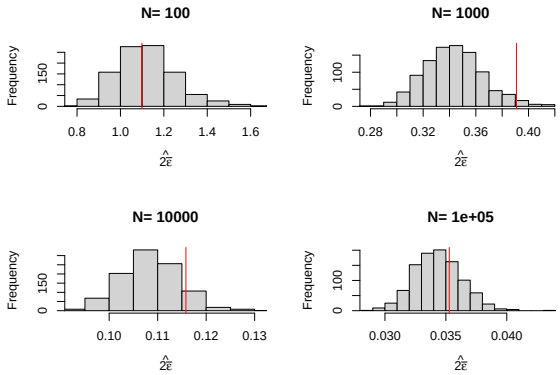


Figure 2.16: Distribution of the estimates of sampling noise using Randomization Inference over replications of samples of different sizes (true sampling noise in red)

```
for (k in (1:length(N.sample))){
  table.noise$fisher.noise[k] <- mean(simuls.ww.fisher[[k]]$fisher.samp.noise)
}
ggplot(table.noise, aes(x=as.factor(N), y=TT)) +
  geom_bar(position=position_dodge(), stat="identity", colour='black') +
  geom_errorbar(aes(ymin=TT-sampling.noise/2, ymax=TT+sampling.noise/2), width=.2,posit
  geom_errorbar(aes(ymin=TT-fisher.noise/2, ymax=TT+fisher.noise/2), width=.2,position=
  xlab("Sample Size")+
  theme_bw()
```

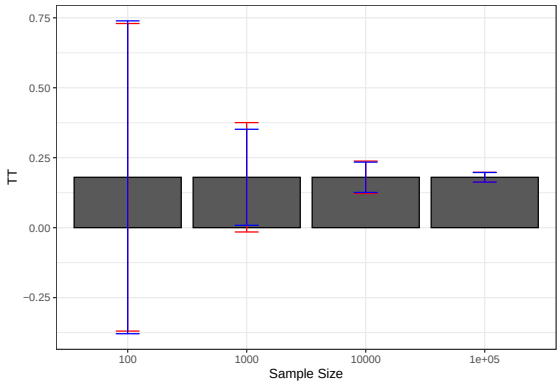


Figure 2.17: Average estimates of sampling noise using Randomization Inference over replications of samples of different sizes (true sampling noise in red)

```
N.plot <- 40
plot.list <- list()

for (k in 1:length(N.sample)){
 set.seed(1234)
 test.fisher <- simuls.ww.fisher[[k]][sample(N.plot),c('WW','fisher.lCI','fisher.uCI')]
 test.fisher <- as.data.frame(cbind(test.fisher,rep(samp.noise(simuls.ww.fisher[[k]][,'WW'],delta
 colnames(test.fisher) <- c('WW','fisher.lCI','fisher.uCI','sampling.noise')
 test.fisher$id <- 1:N.plot
 plot.test.fisher <- ggplot(test.fisher, aes(x=as.factor(id), y=WW)) +
     geom_bar(position=position_dodge(), stat="identity", colour='black') +
     geom_errorbar(aes(ymin=WW-sampling.noise/2, ymax=WW+sampling.noise/2), width=.2,position=pos
     geom_errorbar(aes(ymin=fisher.lCI, ymax=fisher.uCI), width=.2,position=position_dodge(.9),co
     geom_hline(aes(yintercept=delta.y.ate(param)), colour="#990000", linetype="dashed")+
     xlab("Sample id")+
     theme_bw()+
     ggtitle(paste("N=",N.sample[k]))
 plot.list[[k]] <- plot.test.fisher
}
plot.CI <- plot_grid(plot.list[[1]],plot.list[[2]],plot.list[[3]],plot.list[[4]],ncol=1,nrow=leng
print(plot.CI)
```

TO DO: ALTERNATIVE APPROACH USING p-VALUES

2.2.4.3 Subsampling

TO DO: ALL

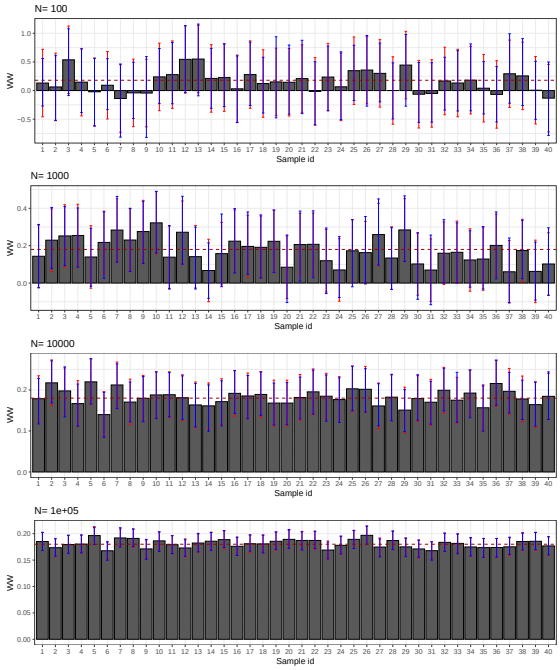


Figure 2.18: Confidence intervals of $\hat{WW}$ for $\delta = 0.99$ estimated using Randomization Inference over sample replications for various sample sizes (true confidence intervals in red)

Part II

Methods of Causal Inference

Chapter 3

Randomized Controlled Trials

The most robust and rigorous method that has been devised by social scientists to estimate the effect of an intervention on an outcome is the Randomized Controlled Trial (RCT). RCTs are used extensively in the field to evaluate a wide array of programs, from development, labor and education interventions to environmental nudges to website and search engine features.

The key feature of an RCT is the introduction by the researcher of randomness in the allocation of the treatment. Individuals with $R_i = 1$, where R_i denotes the outcome of a random event, such as a coin toss, have a higher probability of receiving the treatment. Potential outcomes have the same distribution in both $R_i = 1$ and $R_i = 0$ groups. If we observe different outcomes between the treatment and control group, it has to be because of the causal effect of the treatment, since both groups only differ by the proportion of treated and controls.

The most attractive feature of RCTs is that researchers enforce the main identification assumption (we do not have to assume that it holds, we can make sure that it does). This property of RCTs distinguishes them from all the other methods that we are going to learn in this class.

In this lecture, we are going to study how to estimate the effect of an intervention on an outcome using RCTs. We are especially going to study the various types of designs and what can be recovered from them using which technique. For each design, we are going to detail which treatment effect it enables us to identify, how to obtain a sample estimate of this treatment effect and how to estimate the associated sampling noise. The main substantial difference between these four designs are the types of treatment effect parameters that they enable us to recover. Sections 3.1 to 3.4 of this lecture introduces the four designs and how

to analyze them.

Unfortunately, RCTs are not bullet proof. They suffer from problems that might make their estimates of causal effects badly biased. Section **??** surveys the various threats and what we can do to try to minimize them.

3.1 Brute Force Design

In the Brute Force Design, eligible individuals are randomly assigned to the treatment irrespective of their willingness to accept it and have to comply with the assignment. This is a rather dumb procedure but it is very easy to analyze and that is why I start with it. With the Brute Force Design, you can recover the average effect of the treatment on the whole population. This parameter is generally called the Average Treatment Effect (ATE).

In this section, I am going to detail the assumptions required for the Brute Force Design to identify the ATE, how to form an estimator of the ATE and how to estimate its sampling noise.

3.1.1 Identification

In the Brute Force Design, we need two assumptions for the ATE to be identified in the population: Independence and Brute Force Validity.

Definition 3.1 (Independence)**.** We assume that the allocation of the program is independent of potential outcomes:

$$R_i \perp\!\!\!\perp (Y_i^0, Y_i^1).$$

Here, $\perp\!\!\!\perp$ codes for independence or random variables. Independence can be enforced by the randomized allocation.

We need a second assumption for the Brute Force Design to work:

Definition 3.2 (Brute Force Validity)**.** We assume that the randomized allocation of the program is mandatory and does not interfere with how potential outcomes are generated:

$$Y_i = \begin{cases} Y_i^1 & \text{if } R_i = 1 \\ Y_i^0 & \text{if } R_i = 0 \end{cases}$$

with Y_i^1 and Y_i^0 the same potential outcomes as defined in Lecture~0 with a routine allocation of the treatment.

Under both Idependence and Brute Force Validity, we have the follwing result:

Theorem 3.1 (Identification in the Brute Force Design)**.** *Under Assumptions 3.1 and 3.2, the WW estimator identifies the Average Effect of the Treatment*

(ATE):

$$\Delta^Y_{WW} = \Delta^Y_{ATE},$$

with:

$$\Delta^Y_{WW} = \mathbb{E}[Y_i|R_i = 1] - \mathbb{E}[Y_i|R_i = 0]$$
$$\Delta^Y_{ATE} = \mathbb{E}[Y^1_i - Y^0_i].$$

Proof.

$$\begin{aligned}\Delta^Y_{WW} &= \mathbb{E}[Y_i|R_i = 1] - \mathbb{E}[Y_i|R_i = 0] \\ &= \mathbb{E}[Y^1_i|R_i = 1] - \mathbb{E}[Y^0_i|R_i = 0] \\ &= \mathbb{E}[Y^1_i] - \mathbb{E}[Y^0_i] \\ &= \mathbb{E}[Y^1_i - Y^0_i],\end{aligned}$$

where the first equality uses Assumption 3.2, the second equality Assumption 3.1 and the last equality the linearity of the expectation operator. □

Remark. As you can see from Theorem 3.1, ATE is the average effect of the treatment on the whole population, those who would be eligible for it and those who would not. ATE differs from TT because the effect of the treatment might be correlated with treatment intake. It is possible that the treatment has a bigger (resp. smaller) effect on treated individuals. In that case, ATE is higher (resp. smaller) than TT.

Remark. Another related design is the Brute Force Design among Eligibles. In this design, you impose the treatment status only among eligibles, irrespective of whether they want the treatment or not. It can be operationalized using the selection rule used in Section 3.2.

Example 3.1. Let's use the example to illustrate the concept of ATE. Let's generate data with our usual parameter values without allocating the treatment yet:

```
param <- c(8,.5,.28,1500,0.9,0.01,0.05,0.05,0.05,0.1)
names(param) <- c("barmu","sigma2mu","sigma2U","barY","rho","theta","sigma2epsilon","sigma2eta","
```

```
set.seed(1234)
N <-1000
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
Ds <- rep(0,N)
Ds[YB<=param["barY"]] <- 1
```

```
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
```

In the sample, the ATE is the average difference between y_i^1 and y_i^0, or – the expectation operator being linear – the difference between average y_i^1 and average y_i^0. In our sample, the former is equal to 0.179 and the latter to 0.179.

In the population, the ATE is equal to:

$$\begin{aligned}\Delta^y_{ATE} &= \mathbb{E}[Y_i^1 - Y_i^0] \\ &= \mathbb{E}[\alpha_i] \\ &= \bar{\alpha} + \theta\bar{\mu}.\end{aligned}$$

Let's write a function to compute the value of the ATE and of TT (we derived the formula for TT in the previous lecture):

```
delta.y.ate <- function(param){
  return(param["baralpha"]+param["theta"]*param["barmu"])
}
delta.y.tt <- function(param){
  return(param["baralpha"]+param["theta"]*param["barmu"]-param["theta"]*((param["sigma
}
```

In the population, with our parameter values, $\Delta^y_{ATE} = 0.18$ and $\Delta^y_{TT} = 0.172$. In our case, selection into the treatment is correlated with lower outcomes, so that $TT \leq ATE$.

In order to implement the Brute Force Design in practice in a sample, we simply either draw a coin repeatedly for each member of the sample, assigning for example, all "heads" to the treatment and all "tails" to the control. Because it can be a little cumbersome, it is possible to replace the coin toss by a pseudo-Random Number Generator (RNG), which is is an algorithm that tries to mimic the properties of random draws. When generating the samples in the numerical exmples, I actually use a pseudo-RNG. For example, we can draw from a uniform distribution on $[0, 1]$ and allocate to the treatment all the individuals whose draw is smaller than 0.5:

$$R_i^* \sim \mathcal{U}[0,1]$$
$$R_i = \begin{cases} 1 & \text{if } R_i^* \leq .5 \\ 0 & \text{if } R_i^* > .5 \end{cases}$$

The advantage of using a uniform law is that you can set up proportions of treated and controls easily.

Example 3.2. In our numerical example, the following R code generates two random groups, one treated and one control, and imposes the Assumption of Brute Force Validity:

```
# randomized allocation of 50% of individuals
Rs <- runif(N)
R <- ifelse(Rs<=.5,1,0)
y <- y1*R+y0*(1-R)
Y <- Y1*R+Y0*(1-R)
```

Remark. It is interesting to stop for one minute to think about how the Brute Force Design solves the FPCI. First, with the ATE, the counterfactual problem is more severe than in the case of the TT. In the routine mode of the program, where only eligible individuals receive the treatment, both parts of the ATE are unobserved:

- $\mathbb{E}[Y_i^1]$ is unobserved since we only observe the expected value of outcomes for the treated $\mathbb{E}[Y_i^1|D_i = 1]$, and they do not have to be the same.
- $\mathbb{E}[Y_i^0]$ is unobserved since we only observe the expected value of outcomes for the untreated $\mathbb{E}[Y_i^0|D_i = 0]$, and they do not have to be the same.

What the Brute Force Design does, is that it allocates randomly one part of the sample to the treatment, so that we see $\mathbb{E}[Y_i^1|R_i = 1] = \mathbb{E}[Y_i^1]$ and one part to the control so that we see $\mathbb{E}[Y_i^0|R_i = 0] = \mathbb{E}[Y_i^0]$.

3.1.2 Estimating ATE

3.1.2.1 Using the WW estimator

In order to estimate ATE in a sample where the treatment has been randomized using a Brute Force Design, we simply use the sample equivalent of the With/Without estimator:

$$\hat{\Delta}_{WW}^Y = \frac{1}{\sum_{i=1}^N R_i} \sum_{i=1}^N Y_i R_i - \frac{1}{\sum_{i=1}^N (1 - R_i)} \sum_{i=1}^N Y_i (1 - R_i).$$

Example 3.3. In our numerical example, the WW estimator can be computed as follows in the sample:

```
delta.y.ww <- mean(y[R==1])-mean(y[R==0])
```

The WW estimator of the ATE in the sample is equal to 0.156. Let's recall that the true value of ATE is 0.18 in the population and 0.179 in the sample.

We can also see in our example how the Brute Force Design approximates the counterfactual expectation $\mathbb{E}[y_i^1]$ and its sample equivalent mean $\frac{1}{\sum_{i=1}^N}\sum_{i=1}^N y_i^1$ by the observed mean in the treated sample $\frac{1}{\sum_{i=1}^N R_i}\sum_{i=1}^N y_i R_i$. In our example, the sample value of the counterfactual mean potential outcome $\frac{1}{\sum_{i=1}^N}\sum_{i=1}^N y_i^1$ is equal to 8.222 and the sample value of its observed counterpart is 8.209. Similarly, the sample value of the counterfactual mean potential outcome $\frac{1}{\sum_{i=1}^N}\sum_{i=1}^N y_i^0$ is equal to 8.043 and the sample value of its observed counterpart is 8.054.

3.1.2.2 Using OLS

As we have seen in Lecture 0, the WW estimator is numerically identical to the OLS estimator of a linear regression of outcomes on treatment: The OLS coefficient β in the following regression:

$$Y_i = \alpha + \beta R_i + U_i$$

is the WW estimator.

Example 3.4. In our numerical example, we can run the OLS regression as follows:

```
reg.y.R.ols <- lm(y~R)
```

$\hat{\Delta}^y_{OLS} = 0.156$ which is exactly equal, as expected, to the WW estimator: 0.156.

3.1.2.3 Using OLS conditioning on covariates

The advantage of using OLS other the direct WW comparison is that it gives you a direct estimate of sampling noise (see next section) but also that it enables you to condition on additional covariates in the regression: The OLS coefficient β in the following regression:

$$Y_i = \alpha + \beta R_i + \gamma' X_i + U_i$$

is a consistent (and even unbiased) estimate of the ATE.

proof needed, especially assumption of linearity. Also, is interaction between X_i and R_i needed?

Example 3.5. In our numerical example, we can run the OLS regression conditioning on y_i^B as follows:

```
reg.y.R.ols.yB <- lm(y~R + yB)
```

$\hat{\Delta}^y_{OLSX} = 0.177$. Note that $\hat{\Delta}^y_{OLSX} \neq \hat{\Delta}^y_{WW}$. There is no numerical equivalence between the two estimators.

Remark. Why would you want to condition on covariates in an RCT? Indeed, covariates should be balanced by randomization and thus there does not seem to be a rationale for conditioning on potential confounders, since there should be none. The main reason why we condition on covariates is to decrease sampling noise. Remember that sampling noise is due to imbalances between confounders in the treatment and control group. Since these imbalances are not systematic, the WW estimator is unbiased. We can also make the bias due to these unbalances as small as we want by choosing an adequate sample size (the WW estimator is consistent). But for a given sample size, these imbalances generate sampling noise around the true ATE. Conditioning on covariates helps decrease sampling noise by accounting for imbalances due to observed covariates. If observed covariates explain a large part of the variation in outcomes, conditioning on them is going to prevent a lot of sampling noise from occuring.

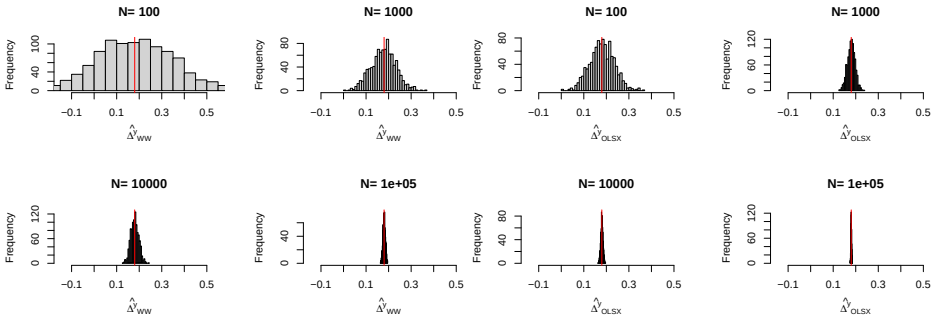
Example 3.6. In order to make the gains in precision from conditioning on covariates apparent, let's use Monte Carlo simulations of our numerical example.

```
monte.carlo.brute.force.ww <- function(s,N,param){
  set.seed(s)
  mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
  UB <- rnorm(N,0,sqrt(param["sigma2U"]))
  yB <- mu + UB
  YB <- exp(yB)
  Ds <- rep(0,N)
  Ds[YB<=param["barY"]] <- 1
  epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
  eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
  U0 <- param["rho"]*UB + epsilon
  y0 <- mu +  U0 + param["delta"]
  alpha <- param["baralpha"]+  param["theta"]*mu + eta
  y1 <- y0+alpha
  Y0 <- exp(y0)
  Y1 <- exp(y1)
  # randomized allocation of 50% of individuals
  Rs <- runif(N)
  R <- ifelse(Rs<=.5,1,0)
  y <- y1*R+y0*(1-R)
  Y <- Y1*R+Y0*(1-R)
  reg.y.R.ols <- lm(y~R)
  return(c(reg.y.R.ols$coef[2],sqrt(vcovHC(reg.y.R.ols,type='HC2')[2,2])))
}
```

```
simuls.brute.force.ww.N <- function(N,Nsim,param){
  simuls.brute.force.ww <- as.data.frame(matrix(unlist(lapply(1:Nsim,monte.carlo.brute
  colnames(simuls.brute.force.ww) <- c('WW','se')
  return(simuls.brute.force.ww)
}

sf.simuls.brute.force.ww.N <- function(N,Nsim,param){
  sfInit(parallel=TRUE,cpus=2*ncpus)
  sfLibrary(sandwich)
  sim <- as.data.frame(matrix(unlist(sfLapply(1:Nsim,monte.carlo.brute.force.ww,N=N,pa
  sfStop()
  colnames(sim) <- c('WW','se')
  return(sim)
}

Nsim <- 1000
#Nsim <- 10
N.sample <- c(100,1000,10000,100000)
#N.sample <- c(100,1000,10000)
#N.sample <- c(100,1000)
#N.sample <- c(100)

simuls.brute.force.ww <- lapply(N.sample,sf.simuls.brute.force.ww.N,Nsim=Nsim,param=pa
names(simuls.brute.force.ww) <- N.sample
```

```
monte.carlo.brute.force.ww.yB <- function(s,N,param){
  set.seed(s)
  mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
  UB <- rnorm(N,0,sqrt(param["sigma2U"]))
  yB <- mu + UB
  YB <- exp(yB)
  Ds <- rep(0,N)
  Ds[YB<=param["barY"]] <- 1
  epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
  eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
  U0 <- param["rho"]*UB + epsilon
  y0 <- mu +  U0 + param["delta"]
  alpha <- param["baralpha"]+  param["theta"]*mu + eta
  y1 <- y0+alpha
  Y0 <- exp(y0)
  Y1 <- exp(y1)
  # randomized allocation of 50% of individuals
  Rs <- runif(N)
  R <- ifelse(Rs<=.5,1,0)
  y <- y1*R+y0*(1-R)
```

```
  Y <- Y1*R+Y0*(1-R)
  reg.y.R.yB.ols <- lm(y~R + yB)
  return(c(reg.y.R.yB.ols$coef[2],sqrt(vcovHC(reg.y.R.yB.ols,type='HC2')[2,2])))
}

simuls.brute.force.ww.yB.N <- function(N,Nsim,param){
  simuls.brute.force.ww.yB <- as.data.frame(matrix(unlist(lapply(1:Nsim,monte.carlo.brute.force.w
  colnames(simuls.brute.force.ww.yB) <- c('WW','se')
  return(simuls.brute.force.ww.yB)
}

sf.simuls.brute.force.ww.yB.N <- function(N,Nsim,param){
  sfInit(parallel=TRUE,cpus=2*ncpus)
  sfLibrary(sandwich)
  sim <- as.data.frame(matrix(unlist(sfLapply(1:Nsim,monte.carlo.brute.force.ww.yB,N=N,param=para
  sfStop()
  colnames(sim) <- c('WW','se')
  return(sim)
}

Nsim <- 1000
#Nsim <- 10
N.sample <- c(100,1000,10000,100000)
#N.sample <- c(100,1000,10000)
#N.sample <- c(100,1000)
#N.sample <- c(100)

simuls.brute.force.ww.yB <- lapply(N.sample,sf.simuls.brute.force.ww.yB.N,Nsim=Nsim,param=param)
names(simuls.brute.force.ww.yB) <- N.sample
```



(a) WW (b) OLSX

Figure 3.1: Distribution of the WW and $OLSX$ estimators in a Brute Force design over replications of samples of different sizes

Figure 3.1 shows that the gains in precision from conditioning on y_i^B are spectacular in our numerical example. They basically correspond to a gain in one order of magnitude of sample size: the precision of the $OLSX$ estimator conditioning on y_i^B with a sample size of 100 is similar to the precision of the OLS estimator not conditioning on y_i^B with a sample size of 1000. This large gain in precision is largely due to the fact that y_i and y_i^B are highly correlated. Not all covariates perform so well in actual samples in the social sciences.

Remark. The ability to condition on covariates in order to decrease sampling noise is a blessing but can also be a curse when combined with significance testing. Indeed, you can now see that you can run a lot of regressions (with and without some covariates, interactions, etc) and maybe report only the statistically significant ones. This is a bad practice that will lead to publication bias and inflated treatment effects. Several possibilities in order to avoid that:

1. Pre-register your analysis and explain which covariates you are going to use (with which interactions, etc) so that you cannot cherry pick your favorite results ex-post.
2. Use a stratified design for your RCT (more on this in Lecture 6) so that the important covariates are already balanced between treated and controls.
3. If unable to do all of the above, report results from regressions without controls and with various sets of controls. We do not expect the various treatment effect estimates to be the same (they cannot be, otherwise, they would have similar sampling noise), but we expect the following pattern: conditioning should systematically decrease sampling noise, not increase the treatment effect estimate. If conditioning on covariates makes a treatment effect significant, pay attention to why: is it because of a decrease in sampling noise (expected and OK) or because of an increase in treatment effect (beware specification search).

Revise that especially in light of Chapter ??

Remark. You might not be happy with the assumption of linearity needed to use OLS to control for covariates. I have read somewhere (forgot where) that this should not be much of a problem since covariates are well balanced between groups by randomization, and thus a linear first approximation to the function relating X_i to Y_i should be fine. I tend not to buy that argument much. I have to run simulations with a non linear relation between outcomes and controls and see how linear OLS performs. If you do not like the linearity assumption, you can always use any of the nonparametric observational methods presented in Chapter **??**.

3.1.2.4 Estimating Sampling Noise

In order to estimate sampling noise, you can either use the CLT-based approach or resampling, either using the bootstrap or randomization inference. In Section 2.2, we have already discussed how to estimate sampling noise when using the WW estimator that we are using here. We are going to use the default

and heteroskedasticity-robust standard errors from OLS, which are both CLT-based. Only the heteroskedasticity-robust standard errors are valid under the assumptions that we have made so far. Homoskedasticity would require constant treatment effects. Heteroskedasticity being small in our numerical example, that should not matter much, but it could in other applications.

Example 3.7. Let us first estimate sampling noise for the simple WW estimator without control variables, using the OLS estimator.

```
sn.BF.simuls <- 2*quantile(abs(simuls.brute.force.ww[['1000']][,'WW']-delta.y.ate(param)),probs=c
sn.BF.OLS.hetero <- 2*qnorm((delta+1)/2)*sqrt(vcovHC(reg.y.R.ols,type='HC2')[2,2])
sn.BF.OLS.homo <- 2*qnorm((delta+1)/2)*sqrt(vcov(reg.y.R.ols)[2,2])
```

The true value of the 99% sampling noise with a sample size of 1000 and no control variables is stemming from the simulations is 0.274. The 99% sampling noise estimated using heteroskedasticity robust OLS standard errors is 0.295. The 99% sampling noise estimated using default OLS standard errors is 0.294.

Let us now estimate sampling noise for the simple WW estimator conditioning on y_i^B, using the OLS estimator.

```
sn.BF.simuls.yB <- 2*quantile(abs(simuls.brute.force.ww.yB[['1000']][,'WW']-delta.y.ate(param)),p
sn.BF.OLS.hetero.yB <- 2*qnorm((delta+1)/2)*sqrt(vcovHC(reg.y.R.ols.yB,type='HC2')[2,2])
sn.BF.OLS.homo.yB <- 2*qnorm((delta+1)/2)*sqrt(vcov(reg.y.R.ols.yB)[2,2])
```

The true value of the 99% sampling noise with a sample size of 1000 and no control variables is stemming from the simulations is 0.088. The 99% sampling noise estimated using heteroskedasticity robust OLS standard errors is 0.092. The 99% sampling noise estimated using default OLS standard errors is 0.091.

Let's see how all of this works on average. Figure 3.2 shows that overall the sampling nois eis much lower with $OLSX$ than with WW, as expected from Figure 3.1. The CLT-based estimator of sampling noise accounting for heteroskedasticity (in blue) recovers true sampling noise (in red) pretty well. Figure 3.3 shows that the CLT-based estimates of sampling noise are on point, except for $N = 10000$, where the CLT slightly overestimates true sampling noise. Figure 3.4 shows what happens when conditioning on Y^B in a selection of 40 samples. The reduction in samplong noise is pretty drastic here.

```
for (k in (1:length(N.sample))){
  simuls.brute.force.ww[[k]]$CLT.noise <- 2*qnorm((delta+1)/2)*simuls.brute.force.ww[[k]][,'se']
  simuls.brute.force.ww.yB[[k]]$CLT.noise <- 2*qnorm((delta+1)/2)*simuls.brute.force.ww.yB[[k]][,
}

samp.noise.ww.BF <- sapply(lapply(simuls.brute.force.ww,`[`,,1),samp.noise,delta=delta)
precision.ww.BF <- sapply(lapply(simuls.brute.force.ww,`[`,,1),precision,delta=delta)
names(precision.ww.BF) <- N.sample
signal.to.noise.ww.BF <- sapply(lapply(simuls.brute.force.ww,`[`,,1),signal.to.noise,delta=delta,
names(signal.to.noise.ww.BF) <- N.sample
```

```
table.noise.BF <- cbind(samp.noise.ww.BF,precision.ww.BF,signal.to.noise.ww.BF)
colnames(table.noise.BF) <- c('sampling.noise', 'precision', 'signal.to.noise')
table.noise.BF <- as.data.frame(table.noise.BF)
table.noise.BF$N <- as.numeric(N.sample)
table.noise.BF$ATE <- rep(delta.y.ate(param),nrow(table.noise.BF))
for (k in (1:length(N.sample))){
  table.noise.BF$CLT.noise[k] <- mean(simuls.brute.force.ww[[k]]$CLT.noise)
}
table.noise.BF$Method <- rep("WW",nrow(table.noise.BF))

samp.noise.ww.BF.yB <- sapply(lapply(simuls.brute.force.ww.yB,`[`,,1),samp.noise,delta=
precision.ww.BF.yB <- sapply(lapply(simuls.brute.force.ww.yB,`[`,,1),precision,delta=de
names(precision.ww.BF.yB) <- N.sample
signal.to.noise.ww.BF.yB <- sapply(lapply(simuls.brute.force.ww.yB,`[`,,1),signal.to.no
names(signal.to.noise.ww.BF.yB) <- N.sample
table.noise.BF.yB <- cbind(samp.noise.ww.BF.yB,precision.ww.BF.yB,signal.to.noise.ww.BF
colnames(table.noise.BF.yB) <- c('sampling.noise', 'precision', 'signal.to.noise')
table.noise.BF.yB <- as.data.frame(table.noise.BF.yB)
table.noise.BF.yB$N <- as.numeric(N.sample)
table.noise.BF.yB$ATE <- rep(delta.y.ate(param),nrow(table.noise.BF.yB))
for (k in (1:length(N.sample))){
  table.noise.BF.yB$CLT.noise[k] <- mean(simuls.brute.force.ww.yB[[k]]$CLT.noise)
}
table.noise.BF.yB$Method <- rep("OLSX",nrow(table.noise.BF))

table.noise.BF.tot <- rbind(table.noise.BF,table.noise.BF.yB)
table.noise.BF.tot$Method <- factor(table.noise.BF.tot$Method,levels=c("WW","OLSX"))

ggplot(table.noise.BF.tot, aes(x=as.factor(N), y=ATE,fill=Method)) +
  geom_bar(position=position_dodge(), stat="identity", colour='black') +
  geom_errorbar(aes(ymin=ATE-sampling.noise/2, ymax=ATE+sampling.noise/2), width=.2,pos
  geom_errorbar(aes(ymin=ATE-CLT.noise/2, ymax=ATE+CLT.noise/2), width=.2,position=posi
  xlab("Sample Size")+
  theme_bw()+
  theme(legend.position=c(0.85,0.88))
```

```
par(mfrow=c(2,2))
for (i in 1:length(simuls.brute.force.ww)){
  hist(simuls.brute.force.ww[[i]][,'CLT.noise'],main=paste('N=',as.character(N.sample[i
  abline(v=table.noise.BF[i,colnames(table.noise)=='sampling.noise'],col="red")
}
par(mfrow=c(2,2))
for (i in 1:length(simuls.brute.force.ww.yB)){
  hist(simuls.brute.force.ww.yB[[i]][,'CLT.noise'],main=paste('N=',as.character(N.sampl
  abline(v=table.noise.BF.yB[i,colnames(table.noise)=='sampling.noise'],col="red")
```

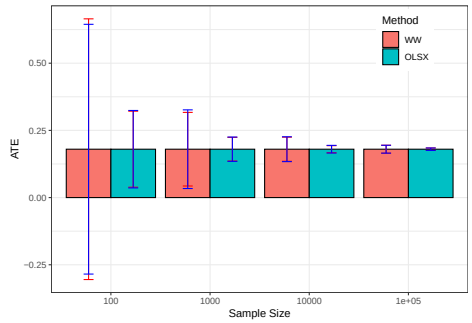
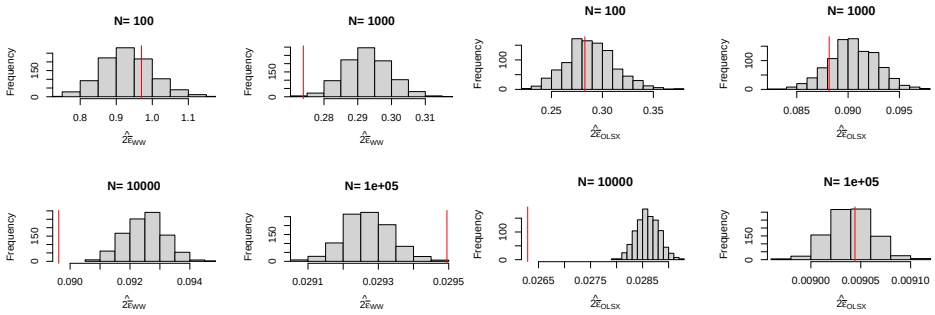


Figure 3.2: Average CLT-based approximations of sampling noise in the Brute Force design for WW and $OLSX$ over replications of samples of different sizes (true sampling noise in red)

```
}
```



(a) WW (b) OLSX

Figure 3.3: Distribution of the CLT approximation of sampling noise in the Brute Force design for WW and $OLSX$ over replications of samples of different sizes (true sampling noise in red)

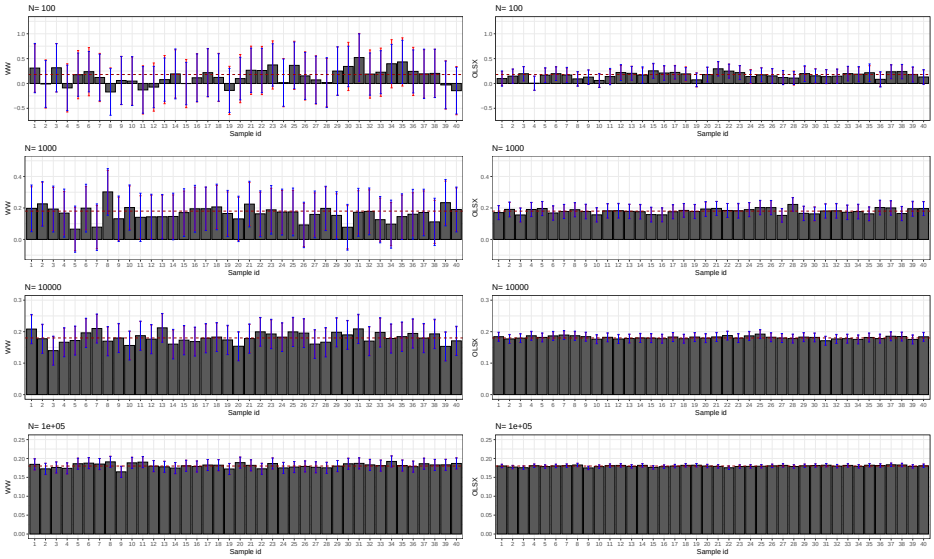
```
N.plot <- 40
plot.list <- list()
limx <- list(c(-0.65,1.25),c(-0.1,0.5),c(0,0.30),c(0,0.25))

for (k in 1:length(N.sample)){
  set.seed(1234)
  test.CLT.BF <- simuls.brute.force.ww[[k]][sample(N.plot),c('WW','CLT.noise')]
  test.CLT.BF <- as.data.frame(cbind(test.CLT.BF,rep(samp.noise(simuls.brute.force.ww[[k]][,'WW']
  colnames(test.CLT.BF) <- c('WW','CLT.noise','sampling.noise')
  test.CLT.BF$id <- 1:N.plot
  plot.test.CLT.BF <- ggplot(test.CLT.BF, aes(x=as.factor(id), y=WW)) +
```

```
      geom_bar(position=position_dodge(), stat="identity", colour='black') +
      geom_errorbar(aes(ymin=WW-sampling.noise/2, ymax=WW+sampling.noise/2), width=.2,p
      geom_errorbar(aes(ymin=WW-CLT.noise/2, ymax=WW+CLT.noise/2), width=.2,position=po
      geom_hline(aes(yintercept=delta.y.ate(param)), colour="#990000", linetype="dashed
      ylim(limx[[k]][1],limx[[k]][2])+
      xlab("Sample id")+
      theme_bw()+
      ggtitle(paste("N=",N.sample[k]))
  plot.list[[k]] <- plot.test.CLT.BF
}
plot.CI.BF <- plot_grid(plot.list[[1]],plot.list[[2]],plot.list[[3]],plot.list[[4]],nco
print(plot.CI.BF)

plot.list <- list()
for (k in 1:length(N.sample)){
  set.seed(1234)
  test.CLT.BF.yB <- simuls.brute.force.ww.yB[[k]][sample(N.plot),c('WW','CLT.noise')]
  test.CLT.BF.yB <- as.data.frame(cbind(test.CLT.BF.yB,rep(samp.noise(simuls.brute.forc
  colnames(test.CLT.BF.yB) <- c('WW','CLT.noise','sampling.noise')
  test.CLT.BF.yB$id <- 1:N.plot
  plot.test.CLT.BF.yB <- ggplot(test.CLT.BF.yB, aes(x=as.factor(id), y=WW)) +
      geom_bar(position=position_dodge(), stat="identity", colour='black') +
      geom_errorbar(aes(ymin=WW-sampling.noise/2, ymax=WW+sampling.noise/2), width=.2,p
      geom_errorbar(aes(ymin=WW-CLT.noise/2, ymax=WW+CLT.noise/2), width=.2,position=po
      geom_hline(aes(yintercept=delta.y.ate(param)), colour="#990000", linetype="dashed
      ylim(limx[[k]][1],limx[[k]][2])+
      xlab("Sample id")+
      ylab("OLSX")+
      theme_bw()+
      ggtitle(paste("N=",N.sample[k]))
  plot.list[[k]] <- plot.test.CLT.BF.yB
}
plot.CI.BF.yB <- plot_grid(plot.list[[1]],plot.list[[2]],plot.list[[3]],plot.list[[4]]
print(plot.CI.BF.yB)
```

3.2 Self-Selection design

In a Self-Selection design, individuals are randomly assigned to the treatment after having expressed their willingness to receive it. This design is able to recover the average effect of the Treatment on the Treated (TT).

In order to explain this design clearly, and especially to make it clear how it differs from the following one (randomization after eligibility), I have to introduce a slightly more complex selection rule that we have seen so far, one that includes self-selection, *i.e.* take-up decisions by agents. We are going to assume that



(a) WW (b) OLSX

Figure 3.4: CLT-based confidence intervals of $\hat{WW}$ and $\hat{OLSX}$ for $\delta = 0.99$ over sample replications for various sample sizes (true confidence intervals in red)

there are two steps in agents' participation process:

- Eligibility: agents' eligibility is assessed first, giving rise to a group of eligible individuals ($E_i = 1$) and a group of non eligible individuals ($E_i = 0$).
- Self-selection: eligible agents can then decide whether they want to take-up the proposed treatment or not. $D_i = 1$ for those who do. $D_i = 0$ for those who do not. By convention, ineligibles have $D_i = 0$.

Example 3.8. In our numerical example, here are the equations operationalizing these notions:

$$
\begin{aligned}
E_i &= \mathbb{1}[y_i^B \leq \bar{y}] \\
D_i &= \mathbb{1}[\underbrace{\bar{\alpha} + \theta\bar{\mu} - C_i}_{D_i^*} \geq 0 \wedge E_i = 1] \\
C_i &= \bar{c} + \gamma\mu_i + V_i \\
V_i &\sim \mathcal{N}(0, \sigma_V^2)
\end{aligned}
$$

Eligibility is still decided based on pre-treatment outcomes being smaller than a threshold level $\bar{y}$. Self-selection among eligibles is decided by the net utility of the treatment D_i^* being positive. Here, the net utility is composed of the average

gain from the treatment (assuming agents cannot foresee their idiosyncratic gain from the treatment) $\bar{\alpha} + \theta\bar{\mu}$ minus the cost of participation C_i. The cost of participation in turn depends on a constant, on μ_i and on a random shock orthogonal to everything else V_i. This cost might represent the administrative cost of applying for the treatment and the opportunity cost of participating into the treatment (foregone earnings and/or cost of time). Conditional on eligiblity, self-selection is endogenous in this model since both the gains and the cost of participation depend on μ_i. Costs depend on μ_i since most productive people may face lower administrative costs but a higher opportunity cost of time.

Let's choose some values for the new parameters:

```
param <- c(param,-6.25,0.9,0.5)
names(param) <- c("barmu","sigma2mu","sigma2U","barY","rho","theta","sigma2epsilon","si
```

and let's generate a new dataset:

```
set.seed(1234)
N <-1000
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
E <- ifelse(YB<=param["barY"],1,0)
V <- rnorm(N,0,param["sigma2V"])
Dstar <- param["baralpha"]+param["theta"]*param["barmu"]-param["barc"]-param["gamma"]*m
Ds <- ifelse(Dstar>=0 & E==1,1,0)
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
```

Let's compute the value of the TT parameter in this new model:

$$\Delta^y_{TT} = \bar{\alpha} + \theta\mathbb{E}[\mu_i | \mu_i + U_i^B \leq \bar{y} \wedge \bar{\alpha} + \theta\bar{\mu} - \bar{c} - \gamma\mu_i - V_i \geq 0]$$

To compute the expectation of a doubly censored normal, I use the package `tmvtnorm`.

$$(\mu_i, y_i^B, D_i^*) = \mathcal{N}\left(\bar{\mu}, \bar{\mu}, \bar{\alpha} + (\theta - \gamma)\bar{\mu} - \bar{c}, \begin{pmatrix} \sigma_\mu^2 & \sigma_\mu^2 & -\gamma\sigma_\mu^2 \\ \sigma_\mu^2 & \sigma_\mu^2 + \sigma_U^2 & -\gamma\sigma_\mu^2 \\ -\gamma\sigma_\mu^2 & -\gamma\sigma_\mu^2 & \gamma^2\sigma_\mu^2 + \sigma_V^2 \end{pmatrix}\right)$$

```
mean.mu.yB.Dstar <- c(param['barmu'],param['barmu'],param['baralpha']- param['barc']+(param['thet
cov.mu.yB.Dstar <- matrix(c(param['sigma2mu'],param["sigma2mu"],-param['gamma']*param["sigma2mu"]
                            param["sigma2mu"],param['sigma2mu']+param['sigma2U'],-param['gamma']*
                            -param['gamma']*param["sigma2mu"],-param['gamma']*param["sigma2mu"],p
lower.cut <- c(-Inf,-Inf,0)
upper.cut <- c(Inf,log(param['barY']),Inf)
moments.cut <- mtmvnorm(mean=mean.mu.yB.Dstar,sigma=cov.mu.yB.Dstar,lower=lower.cut,upper=upper.c
delta.y.tt <- param['baralpha']+ param['theta']*moments.cut$tmean[1]
delta.y.ww.self.select <- mean(y[R==1 & Ds==1])-mean(y[R==0 & Ds==1])
```

The value of Δ^y_{TT} in our illustration is now 0.17.

3.2.1 Identification

In a Self-Selection design, identification requires two assumptions:

Definition 3.3 (Independence Among Self-Selected)**.** We assume that the randomized allocation of the program among applicants is well done:

$$R_i \perp\!\!\!\perp (Y_i^0, Y_i^1) | D_i = 1.$$

Independence can be enforced by the randomized allocation of the treatment among the eligible applicants.

We need a second assumption:

Definition 3.4 (Self-Selection design Validity)**.** We assume that the randomized allocation of the program does not interfere with how potential outcomes and self-selection are generated:

$$Y_i = \begin{cases} Y_i^1 & \text{if } (R_i = 1 \text{ and } D_i = 1) \\ Y_i^0 & \text{if } (R_i = 0 \text{ and } D_i = 1) \text{ or } D_i = 0 \end{cases}$$

with Y_i^1, Y_i^0 and D_i the same potential outcomes and self-selection decisions as in a routine allocation of the treatment.

Under these assumptions, we have the following result:

Theorem 3.2 (Identification in a Self-Selection design)**.** *Under Assumptions 3.3 and 3.4, the WW estimator among the self-selected identifies TT:*

$$\Delta^Y_{WW|D=1} = \Delta^Y_{TT},$$

with:

$$\Delta^{Y}_{WW|D=1} = \mathbb{E}[Y_i|R_i = 1, D_i = 1] - \mathbb{E}[Y_i|R_i = 0, D_i = 1].$$

Proof.

$$\begin{aligned}\Delta^{Y}_{WW|D=1} &= \mathbb{E}[Y_i|R_i = 1, D_i = 1] - \mathbb{E}[Y_i|R_i = 0, D_i = 1] \\ &= \mathbb{E}[Y_i^1|R_i = 1, D_i = 1] - \mathbb{E}[Y_i^0|R_i = 0, D_i = 1] \\ &= \mathbb{E}[Y_i^1|D_i = 1] - \mathbb{E}[Y_i^0|D_i = 1] \\ &= \mathbb{E}[Y_i^1 - Y_i^0|D_i = 1],\end{aligned}$$

where the second equality uses Assummption3.4, the third equality Assumption 3.3 and the last equality the linearity of the expectation operator. □

Remark. The key intuitions for how the Self-Selection design solves the FPCI are:

- By allowing for eligibilty and self-selection, we identify the agents that would benefit from the treatment in routine mode (the treated).
- By randomly denying the treatment to some of the treated, we can estimate the counterfactual outcome of the treated by looking at the counterfactual outcome of the denied applicants: $\mathbb{E}[Y_i^0|D_i = 1] = \mathbb{E}[Y_i|R_i = 0, D_i = 1]$.

Remark. In practice, we use a pseudo-RNG to generate a random allocation among applicants:

$$\begin{aligned}R_i^* &\sim \mathcal{U}[0, 1] \\ R_i &= \begin{cases} 1 & \text{if } R_i^* \leq .5 \wedge D_i = 1 \\ 0 & \text{if } R_i^* > .5 \wedge D_i = 1 \end{cases}\end{aligned}$$

Example 3.9. In our numerical example, the following R code generates two random groups, one treated and one control, and imposes the Assumption of Self-Selection design Validity:

```
#random allocation among self-selected
Rs <- runif(N)
R <- ifelse(Rs<=.5 & Ds==1,1,0)
y <- y1*R+y0*(1-R)
Y <- Y1*R+Y0*(1-R)
```

3.2.2 Estimating TT

3.2.2.1 Using the WW Estimator

As in the case of the Brute Force Design, we can use the WW estimator to estimate the effect of the program with a Self-Selection design, except that this time the WW estimator is applied among applicants to the program only:

$$\hat{\Delta}^Y_{WW|D=1} = \frac{1}{\sum_{i=1}^N D_i R_i} \sum_{i=1}^N Y_i D_i R_i - \frac{1}{\sum_{i=1}^N D_i (1-R_i)} \sum_{i=1}^N D_i Y_i (1-R_i).$$

Example 3.10. In our numerical example, we can form the WW estimator among applicants:

```
delta.y.ww.self.select <- mean(y[R==1 & Ds==1])-mean(y[R==0 & Ds==1])
```

WW among applicants is equal to 0.085. It is actually rather far from the true value of 0.17, which reminds us that unbiasedness does not mean that a given sample will not suffer from a large bias. We just drew a bad sample where confounders are not very well balanced.

3.2.2.2 Using OLS

As in the Brute Force Design with the ATE, we can estimate the TT parameter with a Self-Selection design using the OLS estimator. In the following regression run among applicants only (with $D_i = 1$), β estimates TT:

$$Y_i = \alpha + \beta R_i + U_i.$$

As a matter of fact, the OLS estimator without control variables is numerically equivalent to the WW estimator.

Example 3.11. In our numerical example, here is the OLS regression:

```
reg.y.R.ols.self.select <- lm(y[Ds==1]~R[Ds==1])
```

The value of the OLS estimator is 0.085, which is identical to the WW estimator among applicants.

3.2.2.3 Using OLS Conditioning on Covariates

We might want to condition on covariates in order to reduce the amount of sampling noise. Parametrically, we can run the following OLS regression among applicants (with $D_i = 1$):

$$Y_i = \alpha + \beta R_i + \gamma' X_i + U_i.$$

β estimates the TT.

Needed: proof. Especially check whether we need to center covariates at the mean of the treatment group. I think so.

We can also use Matching to obtain a nonparametric estimator.

Example 3.12. Let us first compute the OLS estimator conditioning on y_i^B:

```
reg.y.R.yB.ols.self.select <- lm(y[Ds==1] ~ R[Ds==1] + yB[Ds==1])
```

Our estimate of TT after conditioning on y_i^B is 0.145. Conditioning on y_i^B has been able to solve part of the bias of the WW problem estimator.

Let's now check whether conditioning on OLS has brought an improvement in terms of decreased sampling noise.

```
monte.carlo.self.select.ww <- function(s,N,param){
  set.seed(s)
  mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
  UB <- rnorm(N,0,sqrt(param["sigma2U"]))
  yB <- mu + UB
  YB <- exp(yB)
  E <- ifelse(YB<=param["barY"],1,0)
  V <- rnorm(N,0,param["sigma2V"])
  Dstar <- param["baralpha"]+param["theta"]*param["barmu"]-param["barc"]-param["gamma"]
  Ds <- ifelse(Dstar>=0 & E==1,1,0)
  epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
  eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
  U0 <- param["rho"]*UB + epsilon
  y0 <- mu +  U0 + param["delta"]
  alpha <- param["baralpha"]+  param["theta"]*mu + eta
  y1 <- y0+alpha
  Y0 <- exp(y0)
  Y1 <- exp(y1)

  #random allocation among self-selected
  Rs <- runif(N)
  R <- ifelse(Rs<=.5 & Ds==1,1,0)
  y <- y1*R+y0*(1-R)
  Y <- Y1*R+Y0*(1-R)
  return(mean(y[R==1 & Ds==1])-mean(y[R==0 & Ds==1]))
}

simuls.self.select.ww.N <- function(N,Nsim,param){
```

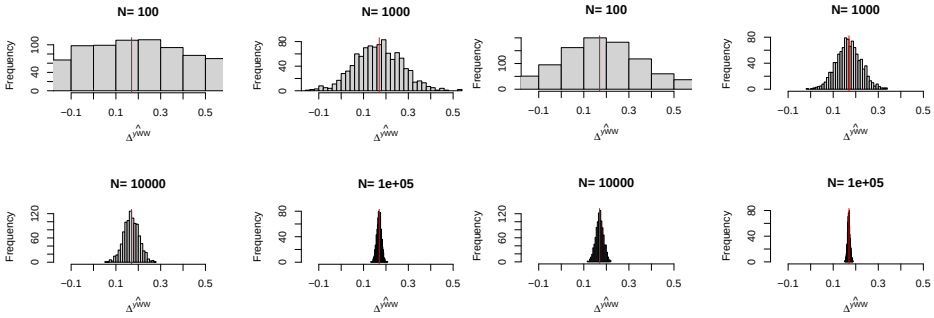
```
  simuls.self.select.ww <- matrix(unlist(lapply(1:Nsim,monte.carlo.self.select.ww,N=N,param=param
  colnames(simuls.self.select.ww) <- c('WW')
  return(simuls.self.select.ww)
}

sf.simuls.self.select.ww.N <- function(N,Nsim,param){
  sfInit(parallel=TRUE,cpus=8)
  sim <- matrix(unlist(sfLapply(1:Nsim,monte.carlo.self.select.ww,N=N,param=param)),nrow=Nsim,nco
  sfStop()
  colnames(sim) <- c('WW')
  return(sim)
}

Nsim <- 1000
#Nsim <- 10
N.sample <- c(100,1000,10000,100000)
#N.sample <- c(100,1000,10000)
#N.sample <- c(100,1000)
#N.sample <- c(100)

simuls.self.select.ww <- lapply(N.sample,sf.simuls.self.select.ww.N,Nsim=Nsim,param=param)
names(simuls.self.select.ww) <- N.sample
```

```
monte.carlo.self.select.yB.ww <- function(s,N,param){
  set.seed(s)
  mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
  UB <- rnorm(N,0,sqrt(param["sigma2U"]))
  yB <- mu + UB
  YB <- exp(yB)
  E <- ifelse(YB<=param["barY"],1,0)
  V <- rnorm(N,0,param["sigma2V"])
  Dstar <- param["baralpha"]+param["theta"]*param["barmu"]-param["barc"]-param["gamma"]*mu-V
  Ds <- ifelse(Dstar>=0 & E==1,1,0)
  epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
  eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
  U0 <- param["rho"]*UB + epsilon
  y0 <- mu +  U0 + param["delta"]
  alpha <- param["baralpha"]+  param["theta"]*mu + eta
  y1 <- y0+alpha
  Y0 <- exp(y0)
  Y1 <- exp(y1)

  #random allocation among self-selected
  Rs <- runif(N)
  R <- ifelse(Rs<=.5 & Ds==1,1,0)
```

```
  y <- y1*R+y0*(1-R)
  Y <- Y1*R+Y0*(1-R)
  reg.y.R.yB.ols.self.select <- lm(y[Ds==1] ~ R[Ds==1] + yB[Ds==1])
  return(reg.y.R.yB.ols.self.select$coef[2])
}

simuls.self.select.yB.ww.N <- function(N,Nsim,param){
  simuls.self.select.yB.ww <- matrix(unlist(lapply(1:Nsim,monte.carlo.self.select.yB.w
  colnames(simuls.self.select.yB.ww) <- c('WW')
  return(simuls.self.select.yB.ww)
}

sf.simuls.self.select.yB.ww.N <- function(N,Nsim,param){
  sfInit(parallel=TRUE,cpus=8)
  sim <- matrix(unlist(sfLapply(1:Nsim,monte.carlo.self.select.yB.ww,N=N,param=param))
  sfStop()
  colnames(sim) <- c('WW')
  return(sim)
}

Nsim <- 1000
#Nsim <- 10
N.sample <- c(100,1000,10000,100000)
#N.sample <- c(100,1000,10000)
#N.sample <- c(100,1000)
#N.sample <- c(100)

simuls.self.select.yB.ww <- lapply(N.sample,sf.simuls.self.select.yB.ww.N,Nsim=Nsim,pa
names(simuls.self.select.yB.ww) <- N.sample
```

```
par(mfrow=c(2,2))
for (i in 1:length(simuls.self.select.ww)){
  hist(simuls.self.select.ww[[i]][,'WW'],breaks=30,main=paste('N=',as.character(N.samp
  abline(v=delta.y.tt,col="red")
}
par(mfrow=c(2,2))
for (i in 1:length(simuls.self.select.yB.ww)){
  hist(simuls.self.select.yB.ww[[i]][,'WW'],breaks=30,main=paste('N=',as.character(N.s
  abline(v=delta.y.tt,col="red")
}
```

Figure 3.5 shows that, in our example, conditioning on covariates improves precision by the same amount as an increase in sample size by almost one order of magnitude.



(a) WW (b) $OLSX$

Figure 3.5: Distribution of the WW and $OLSX$ estimator in a Self-Selection design over replications of samples of different sizes

3.2.3 Estimating Sampling Noise

In order to estimate precision, we can either use the CLT, deriving sampling noise from the heteroskedasticity-robust standard error OLS estimates, or we can use some form of resampling as the bootstrap or randomization inference.

Example 3.13. Let us derive the CLT-based estimates of sampling noise using the OLS standard errors without conditioning on covariates first. I'm using the sample size with $N = 1000$ as an example.

```
sn.RASS.simuls <- 2*quantile(abs(simuls.self.select.ww[['1000']][,'WW']-delta.y.tt),probs=c(0.99)
sn.RASS.OLS.homo <- 2*qnorm((.99+1)/2)*sqrt(vcov(reg.y.R.ols.self.select)[2,2])
sn.RASS.OLS.hetero <- 2*qnorm((.99+1)/2)*sqrt(vcovHC(reg.y.R.ols.self.select,type='HC2')[2,2])
```

True 99% sampling noise (from the simulations) is 0.548. 99% sampling noise estimated using default OLS standard errors is 0.578. 99% sampling noise estimated using heteroskedasticity robust OLS standard errors is 0.58.

Conditioning on covariates:

```
sn.RASS.simuls.yB <- 2*quantile(abs(simuls.self.select.yB.ww[['1000']][,'WW']-delta.y.tt),probs=c
sn.RASS.OLS.homo.yB <- 2*qnorm((.99+1)/2)*sqrt(vcov(reg.y.R.yB.ols.self.select)[2,2])
sn.RASS.OLS.hetero.yB <- 2*qnorm((.99+1)/2)*sqrt(vcovHC(reg.y.R.yB.ols.self.select,type='HC2')[2,
```

True 99% sampling noise (from the simulations) is 0.295. 99% sampling noise estimated using default OLS standard errors is 0.294. 99% sampling noise estimated using heteroskedasticity robust OLS standard errors is 0.299.

3.3 Eligibility design

In an Eligibility design, we randomly select two groups among the eligibles. Members of the treated group are informed that they are eligible to the program

and are free to self-select into it. Members of the control group are not enformed that they are eligible and cannot enroll into the program. In an Eligibility design, we can still recover the TT despite the fact that we have not randomized access to the programs among the applicants. This is the magic of instrumental variables. Let us detail the mechanics of this beautiful result.

3.3.1 Identification

In order to state the identification results in the Randomization After Eligibility design rigorously, I need to define new potential outcomes:

- $Y_i^{d,r}$ is the value of the outcome Y when individual i belongs to the program group d ($d \in \{0,1\}$) and has been randomized in group r ($r \in \{0,1\}$).
- D_i^r is the value of the program participation decision when individual i has been assigned randomly to group r.

3.3.1.1 Identification of TT

In an Eligiblity design, we need three assumptions to ensure identification of the TT:

Definition 3.5 (Independence Among Eligibles)**.** We assume that the randomized allocation of the program among eligibles is well done:

$$R_i \perp\!\!\!\perp (Y_i^{0,0}, Y_i^{0,1}, Y_i^{1,0}, Y_i^{1,1}, D_i^1, D_i^0) | E_i = 1.$$

Independence can be enforced by the randomized allocation of information about eligibility among the eligibles.

We need a second assumption:

Definition 3.6 (Randomization After Eligibility Validity)**.** We assume that no eligibles that has been randomized out can take the treatment and that the randomized allocation of the program does not interfere with how potential outcomes and self-selection are generated:

$$\begin{aligned} &D_i^0 = 0, \forall i, \\ &D_i = D_i^1 R_i + (1 - R_i) D_i^0 \\ &Y_i = \begin{cases} Y_i^{1,1} & \text{if } (R_i = 1 \text{ and } D_i = 1) \\ Y_i^{0,1} & \text{if } (R_i = 1 \text{ and } D_i = 0) \\ Y_i^{0,0} & \text{if } R_i = 0 \end{cases} \end{aligned}$$

with $Y_i^{1,1}$, $Y_i^{0,1}$, $Y_i^{0,0}$, D_i^1 and D_i^0 the same potential outcomes and self-selection decisions as in a routine allocation of the treatment.

We need a third assumption:

Definition 3.7 (Exclusion Restriction of Eligibility)**.** We assume that there is no direct effect of being informed about eligibliity to the program on outcomes:

$$Y_i^{1,1} = Y_i^{1,0} = Y_i^1$$
$$Y_i^{0,1} = Y_i^{0,0} = Y_i^0.$$

Under these assumptions, we have the following result:

Theorem 3.3 (Identification of TT With Randomization After Eligibility)**.** *Under Assumptions 3.5, 3.6 and 3.7, the Bloom estimator among eligibles identifies TT:*

$$\Delta^Y_{Bloom|E=1} = \Delta^Y_{TT},$$

with:

$$\Delta^Y_{Bloom|E=1} = \frac{\Delta^Y_{WW|E=1}}{\Pr(D_i = 1|R_i = 1, E_i = 1)}$$
$$\Delta^Y_{WW|E=1} = \mathbb{E}[Y_i|R_i = 1, E_i = 1] - \mathbb{E}[Y_i|R_i = 0, E_i = 1].$$

Proof. I keep the conditioning on $E_i = 1$ implicit all along to save notation.

$$\begin{aligned}
\mathbb{E}[Y_i|R_i = 1] &= \mathbb{E}[Y_i^{1,1}D_i + Y_i^{0,1}(1 - D_i)|R_i = 1] \\
&= \mathbb{E}[Y_i^0 + D_i(Y_i^1 - Y_i^0)|R_i = 1] \\
&= \mathbb{E}[Y_i^0|R_i = 1] + \mathbb{E}[Y_i^1 - Y_i^0|D_i = 1, R_i = 1] \Pr(D_i = 1|R_i = 1) \\
&= \mathbb{E}[Y_i^0] + \mathbb{E}[Y_i^1 - Y_i^0|D_i = 1] \Pr(D_i = 1|R_i = 1),
\end{aligned}$$

where the first equality uses Assumption 3.6, the second equality Assumption 3.7 and the last equality Assumption 3.5 and the fact that $D_i = 1 \Rightarrow R_i = 1$. Using the same reasoning, we also have:

$$\begin{aligned}
\mathbb{E}[Y_i|R_i = 0] &= \mathbb{E}[Y_i^{1,0}D_i + Y_i^{0,0}(1 - D_i)|R_i = 0] \\
&= \mathbb{E}[Y_i^0|R_i = 0] \\
&= \mathbb{E}[Y_i^0].
\end{aligned}$$

A direct application of the formula for the Bloom estimator proves the result. □

3.3.1.2 Identification of ITE

The previous proof does not give a lot of intuition of how TT is identified in the Randomization After Eligibility design. In order to gain more insight, we are going to decompose the Bloom estimator, and have a look at its numerator. The numerator of the Bloom estimator is a With/Without comparison, and it identifies, under fairly light conditions, another causal effect, the Intention to Treat Effect (ITE).

Let me first define the ITE:

Definition 3.8 (Intention to Treat Effect)**.** In a Randomization After Eligibility design, the Intention to Treat Effect (ITE) is the effect of receiving information about eligiblity among eligibles:

$$\Delta^Y_{ITE} = \mathbb{E}[Y_i^{D_i^1,1} - Y_i^{D_i^0,0} | E_i = 1].$$

Receiving information about eligibility has two impacts, in the general framework that we have delineated so far: first, it triggers some individuals into the treatment (those for which $D_i^1 \neq 0$); second, it might have a direct effect on outcomes ($Y_i^{d,1} \neq Y_i^{d,0}$). This second effect is the effect of annoucing eligiblity that does not goes through participation into the program. For example, it is possible that announcing eligibility to a retirement program makes me save more for retirement, even if I end up not taking up the proposed program.

The two causal channels that are at work within the ITE can be seen more clearly after some manipulations:

$$\begin{aligned}
\Delta^Y_{ITE} &= \mathbb{E}[Y_i^{1,1} D_i^1 + Y_i^{0,1}(1 - D_i^1) - (Y_i^{1,0} D_i^0 + Y_i^{0,0}(1 - D_i^0)) | E_i = 1] \\
&= \mathbb{E}[Y_i^{1,1} D_i^1 + Y_i^{0,1}(1 - D_i^1) - (Y_i^{0,0}(D_i^1 + 1 - D_i^1)) | E_i = 1] \\
&= \mathbb{E}[(Y_i^{1,1} - Y_i^{0,0}) D_i^1 + (Y_i^{0,1} - Y_i^{0,0})(1 - D_i^1) | E_i = 1] \\
&= \mathbb{E}[Y_i^{1,1} - Y_i^{0,0} | D_i^1 = 1, E_i = 1] \Pr(D_i^1 = 1 | E_i = 1) \\
&\quad + \mathbb{E}[Y_i^{0,1} - Y_i^{0,0} | D_i^1 = 0, E_i = 1] \Pr(D_i^1 = 0 | E_i = 1), \qquad (3.1)
\end{aligned}$$

where the first equality follows from Assumption 3.6 and the second equality uses the fact that $D_i^0 = 0$, $\forall i$.

We can now see that the ITE is composed of two terms: the first term captures the effect of announcing eligibility on those who decide to participate into the program; the second term captures the effect of announcing eligibility on those who do not participate into the program. Both of these effects are weighted by the respective proportions of those reacting to the eligibility annoucement by participating and by not participating respectively.

Now, in order to see how the ITE "contains" the TT, we can use the following theorem:

Theorem 3.4 (From ITE to TT)**.** *Under Assumptions 3.5, 3.6 and 3.7, ITE is equal to TT multiplied by the proportion of individuals taking up the treatment after eligibility has been announced:*

$$\Delta^Y_{ITE} = \Delta^Y_{TT} \Pr(D^1_i = 1 | E_i = 1).$$

Proof. Under Assumption 3.7, Equation (3.1) becomes:

$$\begin{aligned}
\Delta^Y_{ITE} &= \mathbb{E}[Y^1_i - Y^0_i | D^1_i = 1, E_i = 1] \Pr(D^1_i = 1 | E_i = 1) \\
&\quad + \mathbb{E}[Y^0_i - Y^0_i | D^1_i = 0, E_i = 1] \Pr(D^1_i = 0 | E_i = 1) \\
&= \mathbb{E}[D^1_i (Y^1_i - Y^0_i) | R_i = 1, E_i = 1] \\
&= \mathbb{E}[Y^1_i - Y^0_i | D^1_i = 1, R_i = 1, E_i = 1] \Pr(D^1_i = 1 | R_i = 1, E_i = 1) \\
&= \mathbb{E}[Y^1_i - Y^0_i | D_i = 1, E_i = 1] \Pr(D^1_i = 1 | E_i = 1),
\end{aligned}$$

where the first equality follows from Assumption 3.7, the second from Bayes' rule and Assumptions 3.5, the third from Bayes' rule and the last from the fact that $D^1_i = 1, R_i = 1 \Leftrightarrow D_i = 1$. □

The previous theorem shows that Assumption 3.7 shuts down any direct effect of the announcement of eligibility on outcomes. As a consequence of this assumption, the only impact that an eligibility annoucement has on outcomes is through participation into the program. Hence, the ITE is equal to TT multiplied by the proportion of people taking up the treatment when eligibility is announced.

In order to move from the link between TT and ITE to the mechanics of the Bloom estimator, we need two additional identification results. The first result shows that ITE can be identified under fairly light conditions by a WW estimator. The second result shows that the proportion of people taking up the treatment when eligiblity is announced is also easily estimated from the data.

Theorem 3.5 (Identification of ITE with Randomization After Eligibility)**.** *Under Assumptions 3.5 and 3.6, ITE is identified by the With/Without comparison among eligibles:*

$$\Delta^Y_{ITE} = \Delta^Y_{WW|E=1}.$$

Proof.

$$\begin{aligned}\Delta^Y_{WW|E=1} &= \mathbb{E}[Y_i|R_i=1,E_i=1] - \mathbb{E}[Y_i|R_i=0,E_i=1] \\ &= \mathbb{E}[Y_i^{D_i^1,1}|R_i=1,E_i=1] - \mathbb{E}[Y_i^{D_i^0,0}|R_i=0,E_i=1] \\ &= \mathbb{E}[Y_i^{D_i^1,1}|E_i=1] - \mathbb{E}[Y_i^{D_i^0,0}|E_i=1],\end{aligned}$$

where the second equality follows from Assumption 3.6 and the third from Assumption 3.5. □

Theorem 3.6 (Identification of $\Pr(D_i^1=1|E_i=1)$)**.** *Under Assumptions 3.5 and 3.6,* $\Pr(D_i^1=1|E_i=1)$ *is identified by the proportion of people taking up the offered treatment when informed about their eligibility status:*

$$\Pr(D_i^1=1|E_i=1) = \Pr(D_i=1|R_i=1,E_i=1).$$

Proof.

$$\begin{aligned}\Pr(D_i=1|R_i=1,E_i=1) &= \Pr(D_i^1=1|R_i=1,E_i=1) \\ &= \Pr(D_i^1=1|E_i=1),\end{aligned}$$

where the first equality follows from Assumption 3.6 and the second from Assumption 3.5. □

Corollary 3.1 (Bloom estimator and ITE)**.** *It follows from Theorems 3.5 and 3.6 that, under Assumptions 3.5 and 3.6, the Bloom estimator is equal to the ITE divided by the propotion of agents taking up the program when eligible:*

$$\Delta^Y_{Bloom|E=1} = \frac{\Delta^Y_{ITE}}{\Pr(D_i^1=1|E_i=1)}.$$

As a consequence of Corollary 3.1, we see that the Bloom estimator reweights the ITE, the effect of receiving information about eligibility, by the proportion of people reacting to the eligibility by participating in the program. From Theorem 3.4, we know that this ratio will be equal to TT if the Assumption 3.7 also holds, so that all the impact of the eligibility annoucement stems from entering the program. The eligibility annoucement serves as an instrument for program participation.

Remark. The design using Randomization After Eligibility seems like magic. You do not assign randomly the program, but information about the eligiblity status, but you can recover the effect of the program anyway. How does this magic work? Randomization After Eligibility is also less intrusive than Self-Selection

design. With the latter design, you have to actively send away individuals that have expressed an interest for entering the program. This is harsh. With Randomization After Eligibility, you do not have to send away people expressing interest after being informed. And it seems that you are not paying a price for that, since you are able to recover the same TT parameter. Well, actually, you are going to pay a price in terms of larger sampling noise.

The intuition for all that can be delineated using the very same apparatus that we have developed so far. So here goes. Under the assumptions made so far, it is easy to show that (omitting the conditioning on $E_i = 1$ for simplicity):

$$\begin{aligned}\Delta^{Y}_{WW|E=1} = &\mathbb{E}[Y_i^{1,1}|D_i^1 = 1, R_i = 1]\Pr(D_i^1 = 1|R_i = 1)\\ &- \mathbb{E}[Y_i^{0,0}|D_i^1 = 1, R_i = 0]\Pr(D_i^1 = 1|R_i = 0)\\ &+ \mathbb{E}[Y_i^{0,1}|D_i^1 = 0, R_i = 1]\Pr(D_i^1 = 0|R_i = 1)\\ &- \mathbb{E}[Y_i^{0,0}|D_i^1 = 0, R_i = 0]\Pr(D_i^1 = 0|R_i = 0).\end{aligned}$$

The first part of the equation is due to the difference in outcomes between the two treatment arms for people that take up the program when eligibility is announced. The second part is due to the difference in outcomes between the two treatment arms for people that do not take up the program when eligibility is announced. This second part cancels out under Assumption 3.5 and 3.7.

But this cancelling out only happens in the population. In a given sample, the sample equivalents to the two members of the second part of the equation do not have to be equal, and thus they do not cancel out, generating additional sampling noise compared to the Self-Selection design. Indeed, in the Self-Selection design, you observe the population with $D_i^1 = 1$ in both the treatment and control arms (you actually observe this population before randomizing the treatment within it), and you can enforce that the effect on $D_i^1 = 0$ should be zero, under your assumptions. In an Eligibility design, you do not observe the population with $D_i^1 = 1$ in the control arm, and you cannot enforce the equality of the outcomes for those with $D_i^1 = 0$ present in both arms. You have to rely on the sampling estimates to make this cancellation, and that generates sampling noise.

Remark. In practice, we use a pseudo-RNG to allocate the randomized annoucement of the eligibility status:

$$\begin{aligned}&R_i^* \sim \mathcal{U}[0,1]\\ &R_i = \begin{cases}1 & \text{if } R_i^* \leq .5 \wedge E_i = 1\\ 0 & \text{if } R_i^* > .5 \wedge E_i = 1\end{cases}\\ &D_i = \mathbb{1}[\bar{\alpha} + \theta\bar{\mu} - C_i \geq 0 \wedge E_i = 1 \wedge R_i = 1]\end{aligned}$$

Example 3.14. In our numerical example, we can actually use the same sample as we did for Self-Selection design. I have to generate it again, though, since I am going to allocate R_i differently.

```
set.seed(1234)
N <-1000
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
E <- ifelse(YB<=param["barY"],1,0)
V <- rnorm(N,0,param["sigma2V"])
Dindex <- param["baralpha"]+param["theta"]*param["barmu"]-param["barc"]-param["gamma"]*
Dstar <- ifelse(Dindex>=0 & E==1,1,0)
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
```

The value of TT in our example is the same as the one in the Self-Selection design case. TT in the population is equal to 0.17.

Let's now compute the value of ITE in the population. In our model, exclusion restriction holds, so that we can use the fact that $ITE = TT \Pr(D_i^1 = 1|E_i = 1)$. We thus only need to compute $\Pr(D_i^1 = 1|E_i = 1)$:

$$\Pr(D_i^1 = 1|E_i = 1) = \Pr(D_i^* \geq 0|y_i^B \leq \bar{y}).$$

I can again use the package `tmvtnorm` to compute that probability. It is indeed equal to $1 - \Pr(D_i^* < 0|y_i^B \leq \bar{y})$, where $\Pr(D_i^* < 0|y_i^B \leq \bar{y})$ is the cumulative density of D_i^* conditional on $y_i^B \leq \bar{y}$, *i.e.* the marginal cumulative of the third variable of the truncated trivariate normal (μ_i, y_i^B, D_i^*) where the first variable is not truncated and the second one is truncated at $\bar{y}$.

```
lower.cut <- c(-Inf,-Inf,-Inf)
upper.cut <- c(Inf,log(param['barY']),Inf)
prD1.elig <- 1-ptmvnorm.marginal(xn=0,n=3,mean=mean.mu.yB.Dstar,sigma=cov.mu.yB.Dstar,l
delta.y.ite <- delta.y.tt*prD1.elig
```

$\Pr(D_i^1 = 1|E_i = 1) = 0.459$. As a consequence, ITE in the population is equal to 0.17 * 0.459 $\approx$ 0.078. In the sample, the value of ITE and TT are equal to:

```
delta.y.tt.sample <- mean(y1[E==1 & Dstar==1]-y0[E==1 & Dstar==1])
delta.y.ite.sample <- delta.y.tt.sample*mean(Dstar[E==1])
```

$\Delta^y_{ITE_s} = 0.068$ and $\Delta^y_{TT_s} = 0.187$.

Now, we can allocate the randomized treatment and let potential outcomes be realized:

```
#random allocation among eligibles
Rs <- runif(N)
R <- ifelse(Rs<=.5 & E==1,1,0)
Ds <- ifelse(Dindex>=0 & E==1 & R==1,1,0)
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
```

3.3.2 Estimating the ITE and the TT

In general, we start the analysis of an Eligibility design by estimating the ITE. Then, we provide the TT by dividing the ITE by the proportion of participants among the eligibles.

Actually, this procedure is akin to an instrumental variables estimator and we will see that the Bloom estimator is actually an IV estimator. The ITE estimation step corresponds to the reduced form in a classical IV approach. Estimation of the proportion of participants is the first stage in a IV approach. Estimation of the TT corresponds to the structural equation step of an IV procedure.

3.3.2.1 Estimating the ITE

Estimation of the ITE relies on the WW estimator, in general implemented using OLS. It is similar to the estimation of ATE and TT in the Brute Force and Self-Selection designs.

3.3.2.1.1 Using the WW estimator Estimation of the ITE can be based on the WW estimator among eligibles.

$$\hat{\Delta}^Y_{WW|E=1} = \frac{1}{\sum_{i=1}^N E_i R_i} \sum_{i=1}^N Y_i E_i R_i - \frac{1}{\sum_{i=1}^N E_i (1-R_i)} \sum_{i=1}^N E_i Y_i (1-R_i).$$

Example 3.15. In our numerical example, we can form the WW estimator among eligibles:

```
delta.y.ww.elig <- mean(y[R==1 & E==1])-mean(y[R==0 & E==1])
```

WW among eligibles is equal to 0.069.

3.3.2.1.2 Using OLS As we have already seen before, the WW estimator is equivalent to OLS with one constant and no control variables. As a consequence, we can estimate the ITE using the OLS estimate of β in the following regression run on the sample with $E_i = 1$:

$$Y_i = \alpha + \beta R_i + U_i.$$

By construction, $\hat{\beta}_{OLSR|E=1} = \hat{\Delta}^Y_{WW|E=1}$.

Example 3.16. In our numerical example, we can form the WW estimator among eligibles:

```
reg.y.ols.elig <- lm(y[E==1]~R[E==1])
delta.y.ols.elig <- reg.y.ols.elig$coef[2]
```

$\hat{\beta}_{OLSR|E=1}$ is equal to 0.069. Remember that ITE in the population is equal to 0.078.

3.3.2.1.3 Using OLS conditioning on covariates Again, as in the previous designs, we can compute ITE by using OLS conditional on covariates. Parametrically, we can run the following OLS regression among eligibles (with $E_i = 1$):

$$Y_i = \alpha + \beta R_i + \gamma' X_i + U_i.$$

The OLS estimate of β estimates the ITE.

Again: Needed: proof. Especially check whether we need to center covariates at the mean of the treatment group. I think so.

We can also use Matching to obtain a nonparametric estimator.

Example 3.17. Let us compute the OLS estimator conditioning on y_i^B:

```
reg.y.R.yB.ols.elig <- lm(y[E==1] ~ R[E==1] + yB[E==1])
```

Our estimate of ITE after conditioning on y_i^B is 0.065. I do not have time to run the simulations, but it is highly likely that the sampling noise is lower after conditioning on y_i^B.

I do not have time to run the simulations, but it is highly likely that the sampling noise is lower after conditioning on y_i^B.

3.3.2.2 Estimating TT

We can estimate TT either using the Bloom estimator, or using the IV estimator, which is equivalent to a Bloom estimator in the Eligibility design.

3.3.2.2.1 Using the Bloom estimator Using the Bloom estimator, we simply compute the numerator of the Bloom estimator and divide it by the estimated proportion of eligible individuals with $R_i = 1$ that have chosen to take the program.

$$\hat{\Delta}^Y_{WW|D=1} = \frac{\frac{1}{\sum_{i=1}^N E_i R_i} \sum_{i=1}^N Y_i E_i R_i - \frac{1}{\sum_{i=1}^N E_i(1-R_i)} \sum_{i=1}^N E_i Y_i (1-R_i)}{\frac{1}{\sum_{i=1}^N E_i R_i} \sum_{i=1}^N D_i E_i R_i}.$$

Example 3.18. Let's see how the Boom estimator works in our example.

The numerator of the Bloom estimator is the ITE that we have just computed: 0.069. The denominator of the Bloom estimator is equal to the proportion of eligible individuals with $R_i = 1$ that have chosen to take the program: 0.342.

```
delta.y.R.bloom.elig <- (mean(y[R==1 & E==1])-mean(y[R==0 & E==1]))/mean(Ds[R==1 & E==1])
```

The resulting estimate of TT is 0.203. It is rather far from the population or sample estimates: 0.17 and 0.187 respectively. What happened? The error seems to come from noise in the denominator of the Bloom estimator. In the ITE estimation, the true ITEs in the population and sample are 0.078 and 0.068 respectively and our estimate is equal to 0.069, so that's fine. In the denominator, the proportion of randomized eligibles that take the program is equal to 0.342 while the true proportions in the population and in the sample are 0.459 and 0.364 respectively. So we do not have enough invited eligibles getting into the program, and the ones who do have unusually large outcomes. These two sampling errors combine to blow up the estimate of TT.

3.3.2.2.2 Using IV There is a very useful results, similar to the one stating that the WW estimator is equivalent to an OLS estimator: in the Eligiblity design, the Bloom estimator is equivalent to an IV estimator:

Theorem 3.7 (Bloom is IV)**.** *Under the assumption that there is at least one individual with $R_i = 1$ and with $D_i = 1$, the coefficient β in the following regression estimated among eligibles using R_i as an IV*

$$Y_i = \alpha + \beta D_i + U_i$$

is the Bloom estimator in the Eligibility Design:

$$\begin{aligned}\hat{\beta}_{IV} &= \frac{\frac{1}{\sum_{i=1}^N E_i}\sum_{i=1}^N E_i\left(Y_i - \frac{1}{\sum_{i=1}^N E_i}\sum_{i=1}^N E_i Y_i\right)\left(R_i - \frac{1}{\sum_{i=1}^N E_i}\sum_{i=1}^N E_i R_i\right)}{\frac{1}{\sum_{i=1}^N E_i}\sum_{i=1}^N E_i\left(D_i - \frac{1}{\sum_{i=1}^N E_i}\sum_{i=1}^N E_i D_i\right)\left(R_i - \frac{1}{\sum_{i=1}^N E_i}\sum_{i=1}^N E_i R_i\right)} \\ &= \frac{\frac{1}{\sum_{i=1}^N E_i R_i}\sum_{i=1}^N Y_i R_i E_i - \frac{1}{\sum_{i=1}^N (1-R_i)E_i}\sum_{i=1}^N Y_i(1-R_i)E_i}{\frac{1}{\sum_{i=1}^N E_i R_i}\sum_{i=1}^N D_i R_i E_i}.\end{aligned}$$

Proof. The proof is straightforward using Theorem 3.15 below and setting $D_i = 0$ when $R_i = 0$. □

Example 3.19. In our numerical example, we have:

```
reg.y.R.2sls.elig <- ivreg(y[E==1]~Ds[E==1]|R[E==1])
```

$\hat{\beta}_{IV} = 0.203$ which is indeed equal to the Bloom estimator ($\hat{\Delta}^y_{Bloom} = 0.203$).

3.3.2.2.3 Using IV conditional on covariates We can improve on the precision of our 2SLS estimator by conditioning on observed covariates. Parametrically estimating the following equation with R_i and X_i as instruments on the sample with $E_i = 1$:

$$Y_i = \alpha + \beta D_i + \gamma' X_i + U_i.$$

Proof? Do we need to center covariates to their mean in the treatment group?

Nonparametric estimation using Frolich's Wald matching estimator.}

Example 3.20. In our numerical example, we have:

```
reg.y.R.yB.2sls.elig <- ivreg(y[E==1] ~ Ds[E==1] + yB[E==1] | R[E==1] + yB[E==1])
```

As a consequence, $\hat{\Delta}^y_{Bloom(X)} = 0.191$.

Does conditioning on covariates improve precision? Let's run some Monte-Carlo somulations in order to check for that.

```
monte.carlo.elig <- function(s,N,param){
  set.seed(s)
  mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
  UB <- rnorm(N,0,sqrt(param["sigma2U"]))
  yB <- mu + UB
  YB <- exp(yB)
  E <- ifelse(YB<=param["barY"],1,0)
```

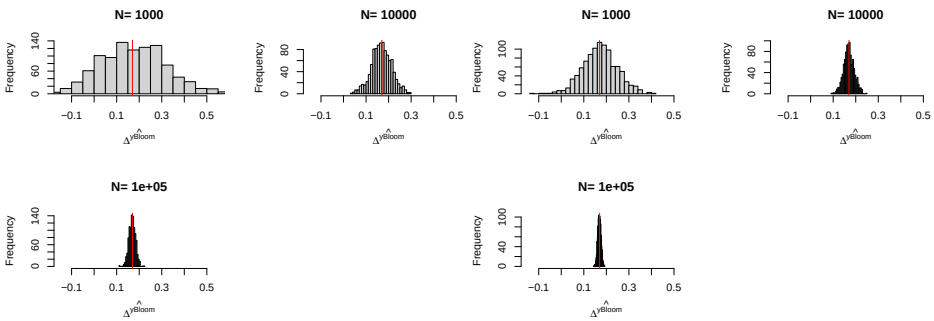
```
  V <- rnorm(N,0,param["sigma2V"])
  Dindex <- param["baralpha"]+param["theta"]*param["barmu"]-param["barc"]-param["gamma"]*mu-V
  Dstar <- ifelse(Dindex>=0 & E==1,1,0)
  epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
  eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
  U0 <- param["rho"]*UB + epsilon
  y0 <- mu +  U0 + param["delta"]
  alpha <- param["baralpha"]+  param["theta"]*mu + eta
  y1 <- y0+alpha
  Y0 <- exp(y0)
  Y1 <- exp(y1)

  #random allocation among self-selected
  Rs <- runif(N)
  R <- ifelse(Rs<=.5 & E==1,1,0)
  Ds <- ifelse(Dindex>=0 & E==1 & R==1,1,0)
  y <- y1*Ds+y0*(1-Ds)
  Y <- Y1*Ds+Y0*(1-Ds)
  reg.y.R.2sls.elig <- ivreg(y[E==1]~Ds[E==1]|R[E==1])
  return(reg.y.R.2sls.elig$coef[2])
}

simuls.elig.N <- function(N,Nsim,param){
  simuls.elig <- matrix(unlist(lapply(1:Nsim,monte.carlo.elig,N=N,param=param)),nrow=Nsim,ncol=1,
  colnames(simuls.elig) <- c('Bloom')
  return(simuls.elig)
}

sf.simuls.elig.N <- function(N,Nsim,param){
  sfInit(parallel=TRUE,cpus=8)
  sfLibrary(AER)
  sim <- matrix(unlist(sfLapply(1:Nsim,monte.carlo.elig,N=N,param=param)),nrow=Nsim,ncol=1,byrow=
  sfStop()
  colnames(sim) <- c('Bloom')
  return(sim)
}

Nsim <- 1000
#Nsim <- 10
#N.sample <- c(100,1000,10000,100000)
N.sample <- c(1000,10000,100000)
#N.sample <- c(100,1000)
#N.sample <- c(100)

simuls.elig <- lapply(N.sample,sf.simuls.elig.N,Nsim=Nsim,param=param)
```

```
names(simuls.elig) <- N.sample
```

```
monte.carlo.elig.yB <- function(s,N,param){
  set.seed(s)
  mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
  UB <- rnorm(N,0,sqrt(param["sigma2U"]))
  yB <- mu + UB
  YB <- exp(yB)
  E <- ifelse(YB<=param["barY"],1,0)
  V <- rnorm(N,0,param["sigma2V"])
  Dindex <- param["baralpha"]+param["theta"]*param["barmu"]-param["barc"]-param["gamma
  Dstar <- ifelse(Dindex>=0 & E==1,1,0)
  epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
  eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
  U0 <- param["rho"]*UB + epsilon
  y0 <- mu +  U0 + param["delta"]
  alpha <- param["baralpha"]+  param["theta"]*mu + eta
  y1 <- y0+alpha
  Y0 <- exp(y0)
  Y1 <- exp(y1)

  #random allocation among self-selected
  Rs <- runif(N)
  R <- ifelse(Rs<=.5 & E==1,1,0)
  Ds <- ifelse(Dindex>=0 & E==1 & R==1,1,0)
  y <- y1*Ds+y0*(1-Ds)
  Y <- Y1*Ds+Y0*(1-Ds)
  reg.y.R.yB.2sls.elig <- ivreg(y[E==1] ~ Ds[E==1] + yB[E==1] | R[E==1] + yB[E==1])
  return(reg.y.R.yB.2sls.elig$coef[2])
}

simuls.elig.yB.N <- function(N,Nsim,param){
  simuls.elig.yB <- matrix(unlist(lapply(1:Nsim,monte.carlo.elig.yB,N=N,param=param)),
  colnames(simuls.elig.yB) <- c('Bloom')
  return(simuls.elig.yB)
}

sf.simuls.elig.yB.N <- function(N,Nsim,param){
  sfInit(parallel=TRUE,cpus=8)
  sfLibrary(AER)
  sim <- matrix(unlist(sfLapply(1:Nsim,monte.carlo.elig.yB,N=N,param=param)),nrow=Nsim
  sfStop()
  colnames(sim) <- c('Bloom')
  return(sim)
}
```

```
Nsim <- 1000
#Nsim <- 10
#N.sample <- c(100,1000,10000,100000)
N.sample <- c(1000,10000,100000)
#N.sample <- c(100,1000)
#N.sample <- c(100)

simuls.elig.yB <- lapply(N.sample,sf.simuls.elig.yB.N,Nsim=Nsim,param=param)
names(simuls.elig.yB) <- N.sample
```

```
par(mfrow=c(2,2))
for (i in 1:length(simuls.elig)){
  hist(simuls.elig[[i]][,'Bloom'],breaks=30,main=paste('N=',as.character(N.sample[i])),xlab=expre
  abline(v=delta.y.tt,col="red")
}
par(mfrow=c(2,2))
for (i in 1:length(simuls.elig.yB)){
  hist(simuls.elig.yB[[i]][,'Bloom'],breaks=30,main=paste('N=',as.character(N.sample[i])),xlab=ex
  abline(v=delta.y.tt,col="red")
}
```



(a) *Bloom* (b) *Bloom*(X)

Figure 3.6: Distribution of the *Bloom* and *Bloom*(X) estimators with randomization after eligibility over replications of samples of different sizes

We can take three things from Figure 3.6:

1. Problems with the IV estimator appear with $N = 100$ (probably because there are some samples where no one is treated).
2. Sampling noise from randomization after eligibility is indeed larger than sampling noise from Self-Selection design.
3. Conditioning on covariates helps.

3.3.3 Estimating sampling noise

As always, we can estimate samling noise either using the CLT or resampling methods. Using the CLT, we can derive the following formula for the distribution of the Bloom estimator:

Theorem 3.8 (Asymptotic Distribution of $\hat{\Delta}^Y_{Bloom}$)**.** *Under Assumptions 3.5, 3.6 and 3.7 and assuming that there is at least one individual with $R_i = 1$ and one individual with $D_i = 1$, we have (keeping the conditioning on $E_i = 1$ implicit):*

$$\sqrt{N}(\hat{\Delta}^Y_{Bloom} - \Delta^Y_{TT}) \xrightarrow{d} \mathcal{N}\left(0, \frac{1}{(p^D_1)^2}\left[\left(\frac{p^D}{p^R}\right)^2 \frac{\mathbb{V}[Y_i|R_i=0]}{1-p^R} + \left(\frac{1-p^D}{1-p^R}\right)^2 \frac{\mathbb{V}[Y_i|R_i=1]}{p^R}\right]\right),$$

with $p^D = \Pr(D_i = 1)$, $p^R = \Pr(R_i = 1)$ and $(p^D_1 = \Pr(D_i = 1|R_i = 1)$.

Proof. The proof is immediate using Theorem 3.16, setting $p^{AT} = 0$. □

Remark. Theorem 3.8 shows that there is a price to pay for not randomizing after self-selection. This price is a decrease in precision. The variance of the estimator is weighted by $\frac{1}{(\Pr(D_i=1|R_i=1))^2}$. This means that the effective sample size is equal to the number of individuals that take up the treatment when offered. We generaly call these individuals "compliers," since they comply with the treatment assignment. Sampling noise is of the same order of magnitude as the number of compliers You might have very low precision despite a very large sample size if you have a very small proportion of compliers.

Remark. In order to compute an estimate of the sampling noise of the Bloom estimator, we can either use the plug-in formula from Theorem 3.8 or use the IV standard errors robust to heteroskedasticity. Here is a simple function in order to compute the plug-in estimator:

```
var.RAE.plugin <- function(pD1,pD,pR,V0,V1,N){
  return(((pD/pR)^2*(V0/(1-pR))+((1-pD)/(1-pR))^2*(V1/pR))/(N*pD1^2))
}
```

Example 3.21. Let us derive the CLT-based estimates of sampling noise using both the plug-in estimator and the IV standard errors without conditioning on covariates first. For the sake of the example, I'm working with a sample of size $N = 1000$.

```
sn.RAE.simuls <- 2*quantile(abs(simuls.elig[['1000']][,'Bloom']-delta.y.tt),probs=c(0.9
sn.RAE.IV.plugin <- 2*qnorm((.99+1)/2)*sqrt(var.RAE.plugin(pD1=mean(Ds[E==1 & R==1]),pl
sn.RAE.IV.homo <- 2*qnorm((.99+1)/2)*sqrt(vcov(reg.y.R.2sls.elig)[2,2])
sn.RAE.IV.hetero <- 2*qnorm((.99+1)/2)*sqrt(vcovHC(reg.y.R.2sls.elig,type='HC2')[2,2])
```

True 99% sampling noise (from the simulations) is 0.757. 99% sampling noise estimated using the plug-in estimator is 0.921. 99% sampling noise estimated

using default IV standard errors is 1.069. 99% sampling noise estimated using heteroskedasticity robust IV standard errors is 0.92.

Conditioning on covariates:

```
sn.RAE.simuls.yB <- 2*quantile(abs(simuls.elig.yB[['1000']][,'Bloom']-delta.y.tt),probs=c(0.99))
sn.RAE.IV.homo.yB <- 2*qnorm((.99+1)/2)*sqrt(vcov(reg.y.R.yB.2sls.elig)[2,2])
sn.RAE.IV.hetero.yB <- 2*qnorm((.99+1)/2)*sqrt(vcovHC(reg.y.R.yB.2sls.elig,type='HC2')[2,2])
```

True 99% sampling noise (from the simulations) is 0.393. 99% sampling noise estimated using default IV standard errors is 0.457. 99% sampling noise estimated using heteroskedasticity robust IV standard errors is 0.454.

Remark. Sampling noise in the Randomization After Eligibility design seems larger than sampling noise in the Self-Selection design.

In the Self-Selection design, sampling noise with $N = 1000$ is equal to 0.55. In the Eligibility design, sampling noise with $N = 1000$ is equal to 0.76. Why such a difference? Both designs have the same effective sample size.

In the Self-Selection design, the effective sample size N_e^{SS} is the number of eligible individuals that apply to take up the program: $N_e^{SS} = N \Pr(D_i = 1|E_i = 1) \Pr(E_i = 1)$. In our example, $N_e^{SS} = 1000 * 0.459 * 0.218 = 100$.

In the Eligibility design, the sample size on which the regressions are performed is N^E, the number of eligible individuals: $N^E = N \Pr(E_i = 1)$. In our example, $N^E = 1000 * 0.459 = 459$. But the effective sample size for the Randomization After Eligibility design is actually equal to the one in the Self-Selection design because only compliers matter for the precision of the Bloom estimator, as Theorem 3.8 shows. Thus $N_e^{SS} = N_e^E$.

Why then is sampling noise much larger in the Randomization After Eligibility design? Probably because the Bloom estimator cannot enforce the fact that the impact of the program on non compliers is zero. It has to estimate the average outcome of non compliers in both treatment arms and hope that they cancel. In real samples, they won't, increasing the size of sampling noise.

3.4 Encouragement Design

In an Encouragement Design, we randomly select two groups among the eligibles, as in Randomization After Eligibility. Treated individuals randomly receive an encouragement to participate in the program and decide whether they want to comply with the encouragement and join the program. Individuals in the control group do not receive an encouragement, but they can still decide to self-select in the program. The Encouragement design differs from the Randomization After Eligibility design mainly by not barring entry into the programs to individuals in the control group. If successful, the encouragement generates a higher level of take up of the program in the treatment group than in the control group. Examples of encouragements are additional reminders that the program exists,

help in subscribing the program, financial incentives for subscribing the program, etc.

In an Encouragement Design, we can recover the causal effect of the treatment not on all the treated but on the treated whose participation into the program has been triggered by the encouragement. The individuals reacting to the encouragement by participating in the program are usually called compliers. The effect of the treatment on the compliers is called the Local Average Treatment Effect. The main identification result for Encouragement designs is that a Wald ratio (an IV estimator) recovers the LATE. It is due to Imbens and Angrist (1994). A key assumption for this result is exclusion restriction: there has to be zero impact of the encouragement on the outcome, except through participation in the treatment. A second key assumption is that no one individual is driven away from participating in the treatment because of the encouragement. This assumption is called monotonicity.

Let's detail these assumptions, the identification result and the estimation strategy.

3.4.1 Identification

3.4.1.1 Identification of the Local Average Treatment Effect

Before stating the identification results, let's go through some definitions and assumptions. We are going to denote $R_i = 1$ when individual i receives the encouragement and $R_i = 0$ when she does not. As in Section 3.3, we have four potential outcomes for Y_i: $Y_i^{d,r}$, $(d, r) \in \{0, 1\}^2$, where d denotes receiving the treatment and r receiving the encouragement. We also have two potential outcomes for D_i: D_i^r, $r \in \{0, 1\}$. D_i^1 indicates whether individual i takes the treatment when receving the encouragement and D_i^0 whether she takes the treatment when not receiving the encouragement. These potential outcomes define four possible types of individuals, that I'm going to denote with the random variable T_i:

- **Always takers**, who take up the program whether they receive the encouragement or not. They are such that $D_i^1 = D_i^0 = 1$. I denote them $T_i = a$.
- **Never takers**, who do not take up the program whether they receive the encouragement or not. They are such that $D_i^1 = D_i^0 = 0$. I denote them $T_i = n$.
- **Compliers**, who take up the program when they receive the encouragement and do not when they do not receive the encouragement. They are such that $D_i^1 - D_i^0 = 1$. I denote them $T_i = c$.
- **Defiers**, who do not take up the program when they receive the encouragement and take it up when they do not receive the encouragement. They are such that $D_i^1 - D_i^0 = -1$. I denote them $T_i = d$.

We are now ready to state the assumptions needed for identification of the LATE.

Definition 3.9 (Encouragement Validity)**.** We assume that the randomized allocation of the program does not interfere with how potential outcomes and self-selection are generated:

$$D_i = D_i^1 R_i + (1 - R_i) D_i^0$$
$$Y_i = \begin{cases} Y_i^{1,1} & \text{if } (R_i = 1 \text{ and } D_i = 1) \\ Y_i^{0,1} & \text{if } (R_i = 1 \text{ and } D_i = 0) \\ Y_i^{1,0} & \text{if } (R_i = 0 \text{ and } D_i = 1) \\ Y_i^{0,0} & \text{if } (R_i = 0 \text{ and } D_i = 0) \end{cases}$$

with $Y_i^{1,1}$, $Y_i^{0,1}$, $Y_i^{1,0}$, $Y_i^{0,0}$, D_i^1 and D_i^0 the same potential outcomes and self-selection decisions as in a routine allocation of the treatment.

Definition 3.10 (Independence of Encouragement)**.** We assume that the randomized allocation of the program is well done:

$$(Y_i^{1,1}, Y_i^{0,1}, Y_i^{0,0}, Y_i^{1,0}, D_i^1, D_i^0) \perp\!\!\!\perp R_i | E_i = 1.$$

Definition 3.11 (Exclusion Restriction)**.** We assume that the randomized allocation of the program does not alter potential outcomes:

$$Y_i^{d,r} = Y_i^d, \forall (r,d) \in \{0,1\}^2 .$$

Definition 3.12 (First Stage)**.** We assume that the encouragement does manage to increase participation:

$$\Pr(D_i = 1 | R_i = 1, E_i = 1) > \Pr(D_i = 1 | R_i = 0, E_i = 1).$$

Definition 3.13 (Monotonicity)**.** We assume that the encouragement either increases participation for everyone or decreases participation for everyone:

$$\text{either } \forall i,\ D_i^1 \geq D_i^0 \text{ or } \forall i,\ D_i^1 \leq D_i^0.$$

Assumption 3.13 means that we cannot have simultaneously individuals that are pushed by the encouragement into the treatment and individuals that are pushed out of the treatment. As a consequence, there cannot be compliers and defiers at the same time. There can only be compliers or defiers. For simplicity, in what follows, I assume that there are no defiers. This is without loss of generality, since, under Assumption 3.13, a redefinition of the treatment ($\tilde{D}_i = -D_i$) moves the model in this section from one with only defiers to one with only compliers.

Theorem 3.9 (Identification in an Encouragement Design)**.** *Under Assumptions 3.9, 3.10, 3.11, 3.12 and 3.13, the Wald estimator identifies the LATE:*

$$\Delta^Y_{Wald} = \Delta^Y_{LATE},$$

with:

$$\Delta^Y_{Wald} = \frac{\mathbb{E}[Y_i|R_i = 1, E_i = 1] - \mathbb{E}[Y_i|R_i = 0, E_i = 1]}{\Pr(D_i = 1|R_i = 1, E_i = 1) - \Pr(D_i = 1|R_i = 0, E_i = 1)}$$
$$\Delta^Y_{LATE} = \mathbb{E}[Y^1_i - Y^0_i|T_i = c, E_i = 1].$$

Proof. See Section A.2.1. □

Remark. Theorem 3.9 is pretty amazing. It shows that there exists a set of assumptions under which we can use an encouragement design to recover the effect of the treatment (D_i) on outcomes, despite the fact that we have NOT randomized D_i. The assumptions needed for that to happen are intuitive:

1. The encouragement has to have no direct effect on the outcomes (Assumption 3.11)
2. The encouragement has to have an effect on treatment uptake (Assumption 3.12)
3. The encouragement does not generate two-way flows in and out of the treatment, but only a one-way flow (Assumption 3.13)

Under these assumptions, the only way that we can see a difference in outcomes between those that receive the encouragement and those that do not is that the treatment has had an effect on those that have taken it because of the encouragement. It cannot be because of the encouragement itself, because of Assumption 3.11. It cannot be because some people with particularly low outcomes have exited the program because of the encouragement, Assumption 3.13 forbids it. And if we see no effect of the encouragement, it has to be that the treatment has no effect on the compliers as well, because Assumption 3.12 implies that they have received the treatment in the encouragement group and that they have not in the group without encouragement.

Remark. Less nice with Theorem 3.9 is that we recover the effect only for a subgroup of individuals, the compliers. This raises two issues:

1. The effect on the compliers (or LATE) is not the effect on the treated (TT). When the treatment is given in routine mode, without the encouragement, TT is actually equal to the effect on the always takers. There is nothing that tells us that the always takers react in the same way to the treatment as the compliers. As soon as the expected benefits of the treatment enter

the decision of taking it up, always takers have larger treatment effects than compliers.
2. The identity of the compliers is unobserved. We cannot decide to allocate the treatment only to the compliers because they are defined by their counterfactual response to the encouragement. In both treatment arms, we do not know who the compliers are. We know they are among those who take up the program in the group receiving the encouragement. But there are also always takers that take up the program in this group. We know that they are among those that do not take up the program in the group that does not receive the encouragement. But never takers behave in the same way in that group.

The only way to direct the treatment at the compliers is to use the encouragement. So, we end up evaluating the effect of the encouragement itself and not of the program. In that case, we do not need Assumptions 3.11, 3.12 and 3.13, because they are not needed to identify the effect of the encouragement (see Section 3.4.1.2 below).

In general, researchers believe that LATE tells them something about the magnitude of the effect beyond compliers. This is not warranted by the maths, but one can understand how a bayesian decision-maker may use the information from some subpopulation to infer what would happen to another. Comparing LATEs and TTs for similar treatments is an active area for reasearch. I know of no paper doing that extensively and nicely.

To generalize from the LATE to the TT, we can make the assumption that the impact on always takers is equal to the impact on compliers, but that seems a little far-fetched. Angrist and Fernandez-Val propose to assume that the effect on compliers is equal to the effect on always takers conditional on some observed covariates. When outcomes are bounded (for example because they are between zero and one), we can try to bound the TT using the $LATE$ (see Huber, Laffers and Mellace (2017)).

Remark. If you see a connexion between the conditions for the Wald estimator to identify LATE and the assumptions behind an IV estimator, you're correct. The Wald estimator is actually an IV estimator (see Theorem 3.15 below).

3.4.1.2 Identification of the Intention to Treat Effect

In this section, we are going to delineate how to identify the Intention to Treat Effect (ITE) in an Encouragement design. In an Encouragement design, ITE is the effect of receiving the encouragement. It is defined in a similar manner as in a Randomization After Eligibility design (see Definition 3.8): $\Delta^Y_{ITE} = \mathbb{E}[Y_i^{D_i^1,1} - Y_i^{D_i^0,0} | E_i = 1]$.

Under Assumption 3.9, receiving the encouragement has several impacts:

1. Some individuals (the compliers) decide to enter the program,
2. Some individuals (the defiers) decide to exit the program,

3. The encouragement might have a direct effect on outcomes ($Y_i^{d,1} \neq Y_i^{d,0}$).

This last effect is the effect of receiving the encouragement that does not goes through participation into the program. For example, it is possible that sending an encouragement to take up a retirement program makes me save more for retirement, even if I end up not taking up the proposed program.

The two causal channels that are at work within the ITE can be seen more clearly when decomposing the ITE to make each type appear. We can do that because the four types define a partition of the sample space, that is a collection of mutually exclusive events whose union spans the whole space. As a consequence of that, conditioning on the union of the four types is the same thing as not conditioning on anything. Using this trick, we have:

$$\begin{aligned}
\Delta^Y_{ITE} &= \mathbb{E}[Y_i^{D_i^1,1} - Y_i^{D_i^0,0} | (T_i = a \cup T_i = c \cup T_i = d \cup T_i = n) \cap E_i = 1] \\
&= \mathbb{E}[Y_i^{D_i^1,1} - Y_i^{D_i^0,0} | T_i = a, E_i = 1] \Pr(T_i = a | E_i = 1) \\
&\quad + \mathbb{E}[Y_i^{D_i^1,1} - Y_i^{D_i^0,0} | T_i = c, E_i = 1] \Pr(T_i = c | E_i = 1) \\
&\quad + \mathbb{E}[Y_i^{D_i^1,1} - Y_i^{D_i^0,0} | T_i = d, E_i = 1] \Pr(T_i = d | E_i = 1) \\
&\quad + \mathbb{E}[Y_i^{D_i^1,1} - Y_i^{D_i^0,0} | T_i = n, E_i = 1] \Pr(T_i = n | E_i = 1) \\
&= \mathbb{E}[Y_i^{1,1} - Y_i^{1,0} | T_i = a, E_i = 1] \Pr(T_i = a | E_i = 1) \\
&\quad + \mathbb{E}[Y_i^{1,1} - Y_i^{0,0} | T_i = c, E_i = 1] \Pr(T_i = c | E_i = 1) \\
&\quad + \mathbb{E}[Y_i^{0,1} - Y_i^{1,0} | T_i = d, E_i = 1] \Pr(T_i = d | E_i = 1) \\
&\quad + \mathbb{E}[Y_i^{0,1} - Y_i^{0,0} | T_i = n, E_i = 1] \Pr(T_i = n | E_i = 1), \qquad (3.2)
\end{aligned}$$

where the first equality follows from the four types defining a partition of the sample space, the second equality from the usual rule of conditional expectations and the fact that types are disjoint events, and the third equality from Assumption 3.6.

We can now see that ITE is composed of four terms:

1. The effect of receiving the encouragment on the always takers. This effect is only the direct effect of the encouragement, and not the effect of the program since the always takers always take the program. This term cancels under Assumption 3.11, when there is no direct effect of the encouragement on outcomes.
2. The effect of receiving the encouragement on compliers. This is both the effect of the encouragement and of the program. This is equal to the LATE under Assumption 3.11.
3. The effect of receiving the encouragement on defiers. This is the difference between the direct effect of the encouragement and the effect of the program. This is equal to the opposite of the effect of the treatment on the defiers under Assumption 3.11.

4. The effect of receiving the encouragement on never takers. This effect is only the direct effect of the encouragement, and not the effect of the program since the never takers never take the program. This term cancels under Assumption 3.11.

All these effects are weighted by the respective proportions of the types in the population. ITE is linked to LATE. This link can be made clearer:

Theorem 3.10 (From ITE to Compliers and Defiers)**.** *Under Assumptions 3.9 and 3.11, ITE is equal to the effect on compliers minus the effect on defiers weighted by their respective proportions in the population:*

$$\begin{aligned}\Delta^Y_{ITE} = &\mathbb{E}[Y_i^1 - Y_i^0 | T_i = c, E_i = 1] \Pr(T_i = c | E_i = 1) \\ &- \mathbb{E}[Y_i^1 - Y_i^0 | T_i = d, E_i = 1] \Pr(T_i = d | E_i = 1).\end{aligned}$$

Proof. Under Assumption 3.11, Equation (3.2) becomes:

$$\begin{aligned}\Delta^Y_{ITE} = &\mathbb{E}[Y_i^1 - Y_i^1 | T_i = a, E_i = 1] \Pr(T_i = a | E_i = 1) \\ &+ \mathbb{E}[Y_i^1 - Y_i^0 | T_i = c, E_i = 1] \Pr(T_i = c | E_i = 1) \\ &+ \mathbb{E}[Y_i^0 - Y_i^1 | T_i = d, E_i = 1] \Pr(T_i = d | E_i = 1) \\ &+ \mathbb{E}[Y_i^0 - Y_i^0 | T_i = n, E_i = 1] \Pr(T_i = n | E_i = 1)\end{aligned}$$

which proves the result. □

The previous theorem shows that Assumption 3.11 shuts down any direct effect of receiving the encouragement on outcomes. As a consequence of this assumption, the only impact that receiving the encouragement has on outcomes is through participation into the program. Hence, ITE is equal to the impact of the program on those who react to the encouragement: the compliers and the defiers, weighted by their respective proportions.

The problem with the result of Theorem 3.10 is that ITE contains two-way flows in and out of the program. If we want to know something about the effect of the program, and not only about the effect of the encouragement, we need to assume that defiers do not exist. That's what Assumption 3.13 does, as the following theorem shows:

Theorem 3.11 (From ITE to LATE)**.** *Under Assumptions 3.9, 3.11 and 3.13, ITE is equal to the LATE multiplied by the proportion of compliers in the population:*

$$\Delta^Y_{ITE} = \Delta^Y_{LATE} \Pr(T_i = c | E_i = 1).$$

Proof. The result is straighforward using Assumption 3.13 and Theorem 3.10. Indeed, Assumption 3.13 implies that $\forall i$, $D_i^1 - D_i^0 \geq 0$ (choosing only the first "either" statement, without loss of generality). As a consequence, $\Pr(T_i = d|E_i = 1) = \Pr(D_i^1 - D_i^0 = -1|E_i = 1) = 0$. □

In order to move from the link between LATE and ITE to the mechanics of the Wald estimator, we need two additional identification results. The first result shows that ITE can be identified under fairly light conditions by a WW estimator. The second result shows that the proportion of people taking up the treatment when eligiblity is announced is also easily estimated from the data.

Theorem 3.12 (Identification of ITE in an Encouragement Design)**.** *Under Assumptions 3.9 and 3.10, ITE is identified by the With/Without comparison among eligibles:*

$$\Delta^Y_{ITE} = \Delta^Y_{WW|E_i=1}.$$

Proof.

$$\begin{aligned}
\Delta^Y_{WW|E=1} &= \mathbb{E}[Y_i|R_i = 1, E_i = 1] - \mathbb{E}[Y_i|R_i = 0, E_i = 1] \\
&= \mathbb{E}[Y_i^{D_i^1,1}|R_i = 1, E_i = 1] - \mathbb{E}[Y_i^{D_i^0,0}|R_i = 0, E_i = 1] \\
&= \mathbb{E}[Y_i^{D_i^1,1}|E_i = 1] - \mathbb{E}[Y_i^{D_i^0,0}|E_i = 1],
\end{aligned}$$

where the second equality follows from Assumption 3.9 and the third from Assumption 3.10. □

Theorem 3.13 (Identification of the Proportion of Compliers)**.** *Under Assumptions 3.9, 3.10 and 3.13, the proportion of compliers is identified by the difference between the proportion of people taking up the program among those receiving the encouragement and the proportion of individuals taking up the program among those not receiving the encouragement:*

$$\Pr(T_i = c|E_i = 1) = \Pr(D_i = 1|R_i = 1, E_i = 1) - \Pr(D_i = 1|R_i = 0, E_i = 1).$$

Proof.

$$\begin{aligned}
\Pr(D_i = 1|R_i = 1, E_i = 1) &= \Pr(D_i = 1 \cap (T_i = a \cup T_i = c \cup T_i = d \cup T_i = n)|R_i = 1, E_i = 1) \\
&= \Pr(D_i = 1 \cap T_i = a|R_i = 1, E_i = 1) \\
&\quad + \Pr(D_i = 1 \cap T_i = c|R_i = 1, E_i = 1) \\
&\quad + \Pr(D_i = 1 \cap T_i = d|R_i = 1, E_i = 1) \\
&\quad + \Pr(D_i = 1 \cap T_i = n|R_i = 1, E_i = 1) \\
&= \Pr(T_i = a|R_i = 1, E_i = 1) \\
&\quad + \Pr(T_i = c|R_i = 1, E_i = 1) \\
&= \Pr(T_i = a|E_i = 1) \\
&\quad + \Pr(T_i = c|E_i = 1),
\end{aligned}$$

where the first equality follows the types being a partition of the sample space; the second equality from the fact that the types are disjoint sets; the third equality from the fact that $T_i = a|R_i = 1 \Rightarrow D_i = 1$ (so that $\Pr(D_i = 1 \cap T_i = a|R_i = 1, E_i = 1) = \Pr(T_i = a|R_i = 1, E_i = 1)$), $T_i = c|R_i = 1 \Rightarrow D_i = 1$ (so that $\Pr(D_i = 1 \cap T_i = c|R_i = 1, E_i = 1) = \Pr(T_i = c|R_i = 1, E_i = 1)$), $T_i = d|R_i = 1 \Rightarrow D_i = 0$ (so that $\Pr(D_i = 1 \cap T_i = d|R_i = 1, E_i = 1) = 0$) and $T_i = n|R_i = 1 \Rightarrow D_i = 0$ (so that $\Pr(D_i = 1 \cap T_i = n|R_i = 1, E_i = 1) = 0$); and the fourth equality from Assumption 3.10 and Lemma A.6. □

Corollary 3.2 (Wald estimator and ITE)**.** *It follows from Theorems 3.12 and 3.13 that, under Assumptions 3.9, 3.10 and 3.13, the Wald estimator is equal to the ITE divided by the propotion of compliers:*

$$\Delta^Y_{Wald|E=1} = \frac{\Delta^Y_{ITE}}{\Pr(T_i = c|E_i = 1)}.$$

As a consequence of Corollary 3.2, we see that the Wald estimator reweights the ITE, the effect of receiving an encouragement, by the proportion of people reacting to the encouragement by participating in the program, the compliers. From Theorem 3.10, we know that this ratio will be equal to LATE if the Assumption 3.11 also holds, so that all the impact of the encouragement stems from entering the program. The encouragement serves as an instrument for program participation.

Remark. The Encouragement design seems like magic. You do not assign randomly the program, but only an encouragement to take it, and you can recover the effect of the program anyway. The Encouragement design is less intrusive than the Self-Selection and Eligibility designs. In an Encouragement design, you do not have to refuse the program to agents in the control group. You pay two types of prices for that:

1. You only recover LATE, not TT

2. You have larger sampling noise.

The intuition for this second point can be delineated using the very same apparatus that we have developed so far. So here goes. Under the assumptions made so far, it is easy to show that (omitting the conditioning on $E_i = 1$ for simplicity):

$$\begin{aligned}
\Delta^{Y}_{WW|E=1} = &\mathbb{E}[Y_i^{1,1}|T_i = a, R_i = 1] \Pr(T_i = a, |R_i = 1) \\
&- \mathbb{E}[Y_i^{1,0}|T_i = a, R_i = 0] \Pr(T_i = a, |R_i = 0) \\
&+ \mathbb{E}[Y_i^{1,1}|T_i = c, R_i = 1] \Pr(T_i = c|R_i = 1) \\
&- \mathbb{E}[Y_i^{0,0}|T_i = c, R_i = 0] \Pr(T_i = c|R_i = 0) \\
&+ \mathbb{E}[Y_i^{0,1}|T_i = d, R_i = 1] \Pr(T_i = d|R_i = 1) \\
&- \mathbb{E}[Y_i^{1,0}|T_i = d, R_i = 0] \Pr(T_i = d|R_i = 0) \\
&+ \mathbb{E}[Y_i^{0,1}|T_i = n, R_i = 1] \Pr(T_i = n|R_i = 1) \\
&- \mathbb{E}[Y_i^{0,0}|T_i = n, R_i = 0] \Pr(T_i = n|R_i = 0).
\end{aligned}$$

The four parts of the equation account for comparisons among each type between the two treatment arms. The parts due to always takers and never takers cancel out under Assumptions 3.10 and 3.11. But this cancelling out only happens in the population. In a given sample, the sample equivalents of the two members of each difference do not have to be equal, and thus they do not cancel out, generating sampling noise. Ideally, we would like to enforce that the effect of the encouragement on always takers and never takers is null, as Assumption 3.11 imposes, but that would require observing the type variable T_i. Unfortunately, we cannot now the type of each individual in the sample, since it is defined counterfactually. Maybe someday we'll be able to use prior responses to the encouragement to identify the type of each individual and thus improve the precision of the Wald estimator.

Explain de Chaisemartin.

Remark. what if we fear there are defiers. de Chaisemartin

Remark. In practice, we use a pseudo-RNG to allocate the randomized annoucement of the encouragement:

$$\begin{aligned}
R_i^* &\sim \mathcal{U}[0,1] \\
R_i &= \begin{cases} 1 & \text{if } R_i^* \leq .5 \wedge E_i = 1 \\ 0 & \text{if } R_i^* > .5 \wedge E_i = 1 \end{cases} \\
D_i &= \mathbb{1}[\bar{\alpha} + \theta\bar{\mu} + \psi R_i - C_i \geq 0 \wedge E_i = 1]
\end{aligned}$$

ψ denotes the increase in agents' valuation of the program after receiving the encouragement.

Example 3.22. Let's see how the encouragement design works in our numerical example. Let's choose a value for ψ and add it to the vector of parameters.

```
param <- c(param,0.6)
names(param) <- c("barmu","sigma2mu","sigma2U","barY","rho","theta","sigma2epsilon","sigma2eta","
```

Let's first compute the value of LATE in this new model. Let's denote $D_i^{*0} = \bar{\alpha}+\theta\bar{\mu}-C_i$ the utility of agent i absent the encouragement, with $C_i = \bar{c}+\gamma\mu_i+V_i$. In order to be a complier, you have to have a utility of the program that is insufficient to make you apply for the program when you receive no encouragement ($D_i^{*0} < 0$) and a positive utility of applying to the treatment after receiving the encouragement ($D_i^{*0} + \psi \geq 0$). Compliers are thus such that $-\psi \leq D_i^{*0} < 0$. LATE can thus be written as follows in our model:

$$\Delta^y_{LATE} = \bar{\alpha} + \theta\mathbb{E}[\mu_i|\mu_i + U_i^B \leq \bar{y} \wedge -\psi \leq D_i^{*0} < 0],$$

Using the same approach, we can also compute the proportion of compliers among eligibles. In our model, we indeed have:

$$\Pr(T_i = c|E_i = 1) = \Pr(-\psi \leq D_i^{*0} < 0|\mu_i + U_i^B \leq \bar{y}).$$

Since all errors terms are normally distributed in our model, we can compute the package `tmvtnorm` to compute both LATE and the proportion of compliers among eligibles.

$$(\mu_i, y_i^B, D_i^{*0}) \sim \mathcal{N}\left(\bar{\mu}, \bar{\mu}, \bar{\alpha} + (\theta-\gamma)\bar{\mu} - \bar{c}, \begin{pmatrix} \sigma_\mu^2 & \sigma_\mu^2 & -\gamma\sigma_\mu^2 \\ \sigma_\mu^2 & \sigma_\mu^2+\sigma_U^2 & -\gamma\sigma_\mu^2 \\ -\gamma\sigma_\mu^2 & -\gamma\sigma_\mu^2 & \gamma^2\sigma_\mu^2+\sigma_V^2 \end{pmatrix}\right)$$

```
mean.mu.yB.Dstar <- c(param['barmu'],param['barmu'],param['baralpha']- param['barc']+(param['thet
cov.mu.yB.Dstar <- matrix(c(param['sigma2mu'],param["sigma2mu"],-param['gamma']*param["sigma2mu"]
                            param["sigma2mu"],param['sigma2mu']+param['sigma2U'],-param['gamma']*
                            -param['gamma']*param["sigma2mu"],-param['gamma']*param["sigma2mu"],p
# late
lower.cut <- c(-Inf,-Inf,-param['psi'])
upper.cut <- c(Inf,log(param['barY']),0)
moments.cut <- mtmvnorm(mean=mean.mu.yB.Dstar,sigma=cov.mu.yB.Dstar,lower=lower.cut,upper=upper.c
delta.y.late <- param['baralpha']+ param['theta']*moments.cut$tmean[1]
# proportion of compliers
```

```
lower.cut <- c(-Inf,-Inf,-Inf)
upper.cut <- c(Inf,log(param['barY']),Inf)
pr.compliers <- ptmvnorm.marginal(xn=0,n=3,mean=mean.mu.yB.Dstar,sigma=cov.mu.yB.Dstar
delta.y.ite <- delta.y.late*pr.compliers
```

The value of Δ^y_{LATE} in the population is thus 0.173. The proportion of compliers among eligibles in the population is 0.272. As a consequence of Corollary 3.2, we can compute ITE as the product of LATE and the proportion of compliers. In our example, ITE is thus equal to 0.047 in the population.

Now let's simulate a new sample with the encouragement delivered randomly among eligibles. I'm also defining the potential outcomes D^1_i and D^0_i and the types T_i for later use.

```
# I'm changing the seed because with the usual one, I get a negative estimate of the t
set.seed(12345)
N <-1000
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
eta<- rnorm(N,0,sqrt(param["sigma2eta"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)

#random allocation of encouragement among eligibles
E <- ifelse(YB<=param["barY"],1,0)
Rs <- runif(N)
R <- ifelse(Rs<=.5 & E==1,1,0)
V <- rnorm(N,0,param["sigma2V"])
Dindex <- param["baralpha"]+param["theta"]*param["barmu"]+param["psi"]*R-param["barc"]-
Ds <- ifelse(Dindex>=0 & E==1,1,0)
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
D <- Ds

# types
Dindex1 <- param["baralpha"]+param["theta"]*param["barmu"]+param["psi"]-param["barc"]-p
Dindex0 <- param["baralpha"]+param["theta"]*param["barmu"]-param["barc"]-param["gamma"]
D1 <- ifelse(Dindex1>=0 & E==1,1,0)
D0 <- ifelse(Dindex0>=0 & E==1,1,0)
```

```
AT <- ifelse(D1==1 & D0==1,1,0)
NT <- ifelse(D1==0 & D0==0,1,0)
Co <- ifelse(D1==1 & D0==0,1,0)

# figures
Ncompliers <- sum(Co)
NElig <- sum(E)
PrCoElig <- Ncompliers/NElig
LATEs <- mean(alpha[Co==1])
ITEs <- LATEs*PrCoElig
```

In our sample of $N = 1000$ individuals, there are only 216 eligibles, and among them 67 compliers. The proportion of compliers among eligibles is thus 0.31. Sample size decreases fast in an encouragement design. The sample LATE is equal to 0.14. The sample ITE is equal to 0.044.

3.4.2 Estimating the Local Average Treatment Effect and the Intention to Treat Effect

Classically, we present the results of an Encouragement design in three stages:

1. We show the **first stage** regression of D_i on R_i: this estimates the impact of the encouragement on participation into the program and estimates the proportion of compliers.
2. We show the **reduced form** regression of Y_i on R_i: this estimates the impact of the encouragement on outcomes, also called ITE.
3. We finally show the **structural** regression of Y_i on D_i using R_i as an instrument, which estimates the LATE.

3.4.2.1 First stage regression

The first stage regression is simply to get an estimate of the effect of the encouragement on participation into the program. If there is no effect of the encouragement on participation, we might as well stop there, since there will be no compliers and no effect to estimate. Note that if we observe an effect on the encouragement on outcomes without any effect on participation, we have to accept the fact that the encouragement might have had a direct effect on outcomes and thus that the exclusion restriction assumption does not hold.

Let's denote this effect of R_i on D_i $\Delta_{TT}^{D,R} = \mathbb{E}[D_i^1 - D_i^0 | R_i = 1, E_i = 1]$. It is a treatment on the treated since we want to estimate the effect of the cnouragement onthose who have received it. Actually, $\Delta_{TT}^{D,R}$ is also equal to $\Delta_{ATE}^{D,R}$, since those who have received the encouragement are a random sample of the eligibles.

How to estimate the effect of R_i on D_i? When estimating the effect of the encouragement, we are in a Brute Force design among eligibles, so that the appropriate estimator is the With/Without estimator among eligibles:

Theorem 3.14 (Identification of the First Stage Effect in an Encouragment Design)**.** *Under Assumptions 3.9 and 3.10, the WW estimator identifies the First Stage Effect (the effect of R_i on D_i):*

$$\Delta_{WW}^{D,R} = \Delta_{TT}^{D,R}.$$

Proof. This is a direct consequence of Theorem 3.1. □

As we have seen in Chapter 1, the WW estimator is identical to an OLS estimator: The OLS coefficient β in the following regression:

$$D_i = \alpha + \beta R_i + U_i$$

is thus the WW estimator.

Finally, the advantage of using OLS other the direct WW comparison is that it gives you a direct estimate of sampling noise (see next section) but also that it enables you to condition on additional covariates in the regression: The OLS coefficient β in the following regression:

$$D_i = \alpha + \beta R_i + \gamma' X_i + U_i$$

is a consistent (and even unbiased) estimate of the ATE.

Center covariates at mean?

Example 3.23. In our numerical example, we can compare all these estimators.

```
WW.D.R <- mean(D[E==1 & R==1])-mean(D[E==1 & R==0])
reg.D.R.ols <- lm(D[E==1]~R[E==1])
reg.D.R.ols.yB <- lm(D[E==1]~R[E==1]+yB[E==1])
```

$\hat{\Delta}_{WW}^{D,R} = 0.213$, while $\hat{\Delta}_{OLS}^{D,R} = 0.213$ which is exactly equal, as expected, to the WW estimator. When controlling for y_i^B, we have: $\hat{\Delta}_{OLS(y^B)}^{D,R} = 0.233$.

Under monotonicity, $\Delta_{TT}^{D,R}$ is equal to the proportion of compliers among eligibles. Indeed, this is the proportion of eligibles that participate when receiving the encouragement and that does not participate when not receiving it. In our example, the proportion of compliers among eligibles is 0.272 in the population and 0.31 in the sample. We are thus underestimating the true proportino of compliers, which is going to make us overestimate the LATE.

3.4.2.2 Reduced form regression

The reduced form regression aims at estimating the ITE, that is the impact of receiving the encouragement on outcomes. From Theorem 3.12, we know that the WW estimator among eligibles identifies the ITE in the population. As a consequence of now classical results, the OLS estimator without control variables is equivalent to the WW estimator and the OLS estimator conditioning on covariates might increase precision.

Example 3.24. In our numerical example, we can compare all these estimators.

```
WW.y.R <- mean(y[E==1 & R==1])-mean(y[E==1 & R==0])
reg.y.R.ols <- lm(y[E==1]~R[E==1])
reg.y.R.ols.yB <- lm(y[E==1]~R[E==1]+yB[E==1])
```

$\hat{\Delta}^{y,R}_{WW} = 0.179$, while $\hat{\Delta}^{y,R}_{OLS} = 0.179$ which is exactly equal, as expected, to the WW estimator. When controlling for y^B_i, we have: $\hat{\Delta}^{y,R}_{OLS(y^B)} = 0.108$. In our example, the ITE is 0.047 in the population and 0.044 in the sample. Without conditioning on Y^B_i, we are thus overestimating the true ITE by a lot. The consequence is that we are going to overestimate the LATE as well.

3.4.2.3 Structural regression

There are four ways to compute the LATE:

1. We can directly compute the sample equivalent to the Wald estimator defined in Theorem 3.9.
2. We can divide our estimate of the ITE by the proportion of compliers, as Corollary 3.2 suggests.
3. We can run a regression of Y on D using R as an instrumental variable.
4. We can run a regression of Y on D using R as an instrumental variable and controlling for some variables X.

It turns out that, in the absence of control variables, the first three estimators are fully equivalent. Corollary 3.2 has already shown that the first two approaches are equivalent in the population. Theorem 3.15 below shows that the Wald estimator is equivalent to an IV estimator.

For simplicity, in all that follows, I am working only in the subgroup of eligible individuals. That means that I'm setting $E_i = 1$ for everyone, so that N is the number of eligible individuals.

3.4.2.3.1 Using the Wald estimator The empirical counterpart to the Wald estimator is the difference in mean outcomes between treatment and controls divided by the difference in participation rates between the two groups:

$$\hat{\Delta}^{Y}_{Wald} = \frac{\frac{1}{\sum_{i=1}^{N} R_i}\sum_{i=1}^{N} Y_i R_i - \frac{1}{\sum_{i=1}^{N}(1-R_i)}\sum_{i=1}^{N} Y_i(1-R_i)}{\frac{1}{\sum_{i=1}^{N} R_i}\sum_{i=1}^{N} D_i R_i - \frac{1}{\sum_{i=1}^{N}(1-R_i)}\sum_{i=1}^{N} D_i(1-R_i)}$$

Example 3.25. Let's check how this works in our numerical example.

```
mean.y.R.1 <- mean(y[E==1 & R==1])
mean.y.R.0 <- mean(y[E==1 & R==0])
mean.D.R.1 <- mean(D[E==1 & R==1])
```

```
mean.D.R.0 <- mean(D[E==1 & R==0])
delta.y.Wald <- (mean.y.R.1-mean.y.R.0)/(mean.D.R.1-mean.D.R.0)
```

The numerator of the Wald estimator is equal to $7.059 - 6.88 = 0.179$. The denominator of the Wald estimator is equal to $0.704 - 0.491 = 0.213$. Overall, the Wald estimator of the LATE is equal to $0.179 \div 0.213 = 0.841$.

Remember that the true LATE is equal to 0.173. We are thus severely overestimating the LATE. We'll understand why in the next section.

3.4.2.3.2 Using the ITE We know from Corollary 3.2 that dividing the ITE by the proportion of compliers gives the Wald estimator. From Theorem 3.12, we know that the ITE can be estimated using the sample equivalent to the With/Without estimator as follows:

$$\hat{\Delta}^{Y,R}_{WW} = \frac{1}{\sum_{i=1}^{N} R_i} \sum_{i=1}^{N} Y_i R_i - \frac{1}{\sum_{i=1}^{N} (1 - R_i)} \sum_{i=1}^{N} Y_i (1 - R_i).$$

From Theorem 3.13, we also know that the proportion of compliers can be estimated using the sample equivalent to the With/Without estimator as follows:

$$\hat{\Delta}^{D,R}_{WW} = \frac{1}{\sum_{i=1}^{N} R_i} \sum_{i=1}^{N} D_i R_i - \frac{1}{\sum_{i=1}^{N} (1 - R_i)} \sum_{i=1}^{N} D_i (1 - R_i).$$

Example 3.26. Let's check that this unfolds in our numerical example.

We already know that the estimated ITE is equal to 0.179, which is equal to the numerator of the Wald estimator. We also now that the proportion of compliers in our sample is equal to 0.213. As a consequence, again, the Wald estimator of the LATE is equal to $0.179 \div 0.213 = 0.841$. Without surprise, we obtain exactly the same results as when using the Wald estimator directly. The two approaches are numerically equivalent.

Again, our estimator of the LATE, the Wald estimator, severelt overestimates the LATE. The Wald estimator is equal to 0.841 while the true LATE is equal to 0.173. What is the reason for this mistake? There are actually two:

1. We are overestimating the ITE (truth: 0.047; estimate: 0.179).
2. We are underestimating the proportion of compliers (truth: 0.272; estimate: 0.213).

The combination of these two mistakes generates the large discrepancy that we see between our estimate of the LATE and its true value. This error comes for covariates that are not distributed identically in the treatment and control

groups. Maybe controlling for some of them would improve our fit. In order to to that, we need the IV estimator.

3.4.2.3.3 Using the IV estimator A very useful result is that the Wald estimator can be computed as an IV estimator. The following theorem proves that point:

Theorem 3.15 (Wald is IV)**.** *Under the assumption that there is at least one individual with $R_i = 1$ and $D_i = 1$, the coefficient β in the following regression estimated using R_i as an IV:*

$$Y_i = \alpha + \beta D_i + U_i$$

is the Wald estimator:

$$\begin{aligned}
\hat{\beta}_{IV} &= \frac{\frac{1}{N}\sum_{i=1}^{N}\left(Y_i - \frac{1}{N}\sum_{i=1}^{N} Y_i\right)\left(R_i - \frac{1}{N}\sum_{i=1}^{N} R_i\right)}{\frac{1}{N}\sum_{i=1}^{N}\left(D_i - \frac{1}{N}\sum_{i=1}^{N} D_i\right)\left(R_i - \frac{1}{N}\sum_{i=1}^{N} R_i\right)} \\
&= \frac{\frac{1}{\sum_{i=1}^{N} R_i}\sum_{i=1}^{N} Y_i R_i - \frac{1}{\sum_{i=1}^{N}(1-R_i)}\sum_{i=1}^{N} Y_i(1-R_i)}{\frac{1}{\sum_{i=1}^{N} R_i}\sum_{i=1}^{N} D_i R_i - \frac{1}{\sum_{i=1}^{N}(1-R_i)}\sum_{i=1}^{N} D_i(1-R_i)} \\
&= \hat{\Delta}^{Y}_{Wald}
\end{aligned}$$

Proof. See in section A.2.2 in the appendix. □

Theorem 3.15 is super powerful since it enables us to directly use the IV estimator to compute the Wald estimator. In order to do so, we're going to use the estimator `ivreg` in the `AER` package.

Example 3.27. Let's see how the IV estimator performs in our numerical example.

```
reg.y.R.2sls.encourage <- ivreg(y[E==1]~Ds[E==1]|R[E==1])
beta.IV <- reg.y.R.2sls.encourage$coef[2]
```

$\hat{\beta}_{IV} = 0.841$, which is equal to the Wald estimator, as Theorem 3.15 predicted.

3.4.2.3.4 Using the IV estimator conditioning on covariates One nice thing about the IV estimator is that we can use it to control for additional covariates X. Estimating the following equation with R_i and X_i as instruments:

$$Y_i = \alpha + \beta D_i + \gamma' X_i + U_i$$

recovers $\beta_{IV}(X)$, which is an estimate of the LATE under linearity assumptions on the potential outcomes.

Expand on that Center covariates at mean?

Example 3.28. Let's see how this work in our numerical example, when we condition on y_i^B.

```
reg.y.R.yB.2sls.encourage <- ivreg(y[E==1] ~ Ds[E==1] + yB[E==1] | R[E==1] + yB[E==1])
beta.IV.yB <- reg.y.R.yB.2sls.encourage$coef[2]
```

$\hat{\beta}_{IV}(y^B) = 0.464$. Remember that the value of Δ^y_{LATE} in the population is thus 0.173. All of our estimators have overshoot. The worse are the ones not conditioning on y^B. It seems that conditioning on y^B improves the estimator slightly. So part of the estimation error in the Wald estimator probably comes from an imbalance in y_i^B between the treatment and control groups. Let's check that this is the case.

```
reg.yB.R.ols.encourage <- lm(yB[E==1] ~ R[E==1])
delta.yB.WW.R <- reg.yB.R.ols.encourage$coef[2]
```

The difference in y_i^B among treated and controls in our example is 0.075. This is enough to account for the bias on the ITE.

Expand on that

Remark. One key question that remains is that whether the structural parameter $\beta(X)$ is still equal to the ratio of the reduced form parameter and the first stage parameter obtained by running OLS conditionnal on X.

Example 3.29. Let's examine what happens in our example.

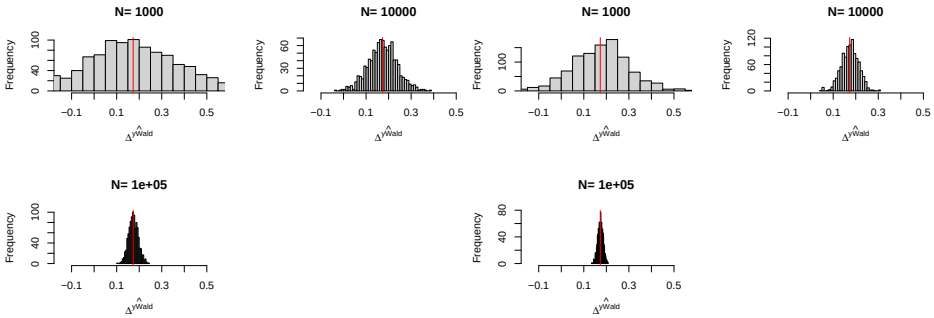
```
reg.y.R.yB.ols.encourage <- lm(y[E==1] ~ R[E==1] + yB[E==1])
ITE.yB <- reg.y.R.yB.ols.encourage$coef[2]

reg.D.R.yB.ols.encourage <- lm(D[E==1] ~ R[E==1] + yB[E==1])
prCo.yB <- reg.D.R.yB.ols.encourage$coef[2]

Wald.yB <- ITE.yB/prCo.yB
```

We find that the ITE conditional on y_i^B is equal to 0.108 while the proportion of compliers conditioning on y_i^B is equal to 0.233. Overall the ratio of these two, which we could call the Wald ratio after conditioning on y_i^B is equal to 0.464. This is actually equal to the IV estimator including y_i^B as a covariate: $\hat{\beta}_{IV}(y^B) = 0.464$. So running the reduced form and first stage regressions separately and dividing the coefficients on R_i recovers the LATE even when conditioning on covariates? That's pretty neat and opens up the route for a variety of new estimation techniques called `split sample` estimators, developed by Angrist and Krueger. We'll take more about them later.

Remark. We might want to control nonparametrically on the covariates instead of imposing a linear regression. Frolich's Wald matching estimator enables to do just that. Its implementation will become clearer after Chapter **??**.

Remark. The last thing we want to check is whether conditioning on covariates improve precision. It seems to be the case in our example with one dataset. Let's see what happens over sampling repetitions.

Example 3.30. Let's run some Monte Carlo simulations for the sampling noise of IV with and without conditining on y_i^B.



(a) $Wald$ (b) $Wald(X)$

Figure 3.7: Distribution of the $Wald$ and $Wald(X)$ estimator in an encouragement design over replications of samples of different sizes

Comment on the results

3.4.3 Estimating sampling noise

As always, we can estimate sampling noise either using the CLT or resampling methods. Using the CLT, we can derive the following formula for the distribution of the Bloom estimator: Theorem 3.16 shows the asymptotic distribution of $\hat{\Delta}^Y_{Wald}$:

Theorem 3.16 (Asymptotic Distribution of $\hat{\Delta}^Y_{Wald}$)**.** *Under Assumptions 3.9, 3.10, 3.11, 3.12, we have:*

$$\sqrt{N}(\hat{\Delta}^Y_{Wald} - \Delta^Y_{LATE}) \xrightarrow{d} \mathcal{N}\left(0, \frac{1}{(p_1^D - p_0^D)^2}\left[\left(\frac{p^D}{p^R}\right)^2 \frac{\mathbb{V}[Y_i|R_i=0]}{1-p^R} + \left(\frac{1-p^D}{1-p^R}\right)^2 \frac{\mathbb{V}[Y_i|R_i=1]}{p^R}\right]\right)$$

Adding Assumption 3.13, the variance of the Wald estimator can be further decomposed as follows:

$$\sigma^2_{\hat{\Delta}^Y_{Wald}} = \left(\frac{p^D}{p^R}\right)^2 \frac{\mathbb{V}[Y_i^0|T_i = C]}{p^C(1-p^R)} + \left(\frac{1-p^D}{1-p^R}\right)^2 \frac{\mathbb{V}[Y_i^1|T_i = C]}{p^C p^R} + \frac{(p^{AT}(1-p^R) - p^{NT}p^R)^2 + p^R(1-p^R)}{(p^C p^R(1-p^R))^2} \left[p^{AT}\mathbb{V}[Y_i^1|T_i = AT] + p^{NT}\mathbb{V}[Y_i^0|T_i = NT]\right]$$

with $p^D = \Pr(D_i = 1)$, $p^R = \Pr(R_i = 1)$, $p_1^D = \Pr(D_i = 1|R_i = 1)$, $p_0^D = \Pr(D_i = 1|R_i = 0)$, $p^t = \Pr(T_i = t)$, with $t \in \{AT, NT, C, D\}$.

Proof. See Section A.2.3. □

Remark. Theorem 3.16 shows that the effective sample size of an encouragement design is equal to the number of compliers. Indeed, the denominator of the variance of the Wald Estimatior depends on $p_1^D - p_0^D$, which is an estimate of the proportion of compliers, under Assumption 3.13.

Theorem 3.16 also shows that there is a price to pay for the fact that we cannot enforce the effect on always takers and never takers is actually zero. Indeed, as the second formula shows, it is not only the variance of the oucomes of the compliers that appears in the formula, but also the variances of the outcomes of the always takers and never takers, therefore increasing sampling noise.

Remark. In order to compute the formula in Theorem 3.16, we can use a plug-in estimator or the IV standard error estimate robist to heteroskedasticity. Here is a simple function in order to compute the plug-in estimator:

Example 3.31. Let us derive the CLT-based estimates of sampling noise using both the plug-in estimator and the IV standard errors without conditioning on covariates first. For the sake of the example, I'm working with a sample of size $N = 1000$.

```
sn.Encourag.simuls <- 2*quantile(abs(simuls.encourage[['1000']][,'Wald']-delta.y.late)
sn.Encourag.IV.plugin <- 2*qnorm((.99+1)/2)*sqrt(var.Encourage.plugin(pD1=mean(D[E==1 &
sn.Encourag.IV.homo <- 2*qnorm((.99+1)/2)*sqrt(vcov(reg.y.R.2sls.encourage)[2,2])
sn.Encourag.IV.hetero <- 2*qnorm((.99+1)/2)*sqrt(vcovHC(reg.y.R.2sls.encourage,type='HC
```

True 99% sampling noise (from the simulations) is 1.166. 99% sampling noise estimated using the plug-in estimator is 2.124. 99% sampling noise estimated using default IV standard errors is 23.134. 99% sampling noise estimated using heteroskedasticity robust IV standard errors is 2.119.

Conditioning on y_i^B:

```
sn.Encourag.simuls.yB <- 2*quantile(abs(simuls.encourage.yB[['1000']][,'Wald']-delta.y
sn.Encourag.IV.homo.yB <- 2*qnorm((.99+1)/2)*sqrt(vcov(reg.y.R.yB.2sls.encourage)[2,2])
sn.Encourag.IV.hetero.yB <- 2*qnorm((.99+1)/2)*sqrt(vcovHC(reg.y.R.yB.2sls.encourage,ty
```

True 99% sampling noise (from the simulations) is 0.705. 99% sampling noise estimated using default IV standard errors is 0.988. 99% sampling noise estimated using heteroskedasticity robust IV standard errors is 0.975.

Chapter 4

Natural Experiments

Natural Experiments are situations due to the natural course of events that approximate the conditions of a randomized controlled trial. In the economists' toolkit, we generally make a distinction between:

1. Instrumental variables (IV), that rely on finding a plausibly exogeneous source of variation in treatment intake.
2. Regression Discontinuity Designs (RDD), that exploit a discontinuity in the eligibility to the treatment.
3. Difference In Differences (DID), that make use of the differential exposure of some groups to the treatment of interest over time.

Remark. The term *Natural Experiments* seems to be mostly used by economists. It dates back to Haavelmo (1944)'s paper on the Probability Approach to Econometrics, where he makes a distinction between the experiments we'd like to make as social scientists and the experiments that Nature provides us with, that are in general a subset of the experiments we'd like to make. This raises the question of our ability to **identify** the relationships of interest from the variation that is present in the data, a traditional problem in classical econometrics that has echoes in treatment effect estimation, where we also try to *identify* treatment effect parameters. At the time of Haavelmo, and until the beginning of the 1990s, there was no real discussion of the plausibility of the *identifying assumptions* (or restrictions) required for identification of certain relations, outside of a discussion of their theoretical plausiblility. With the credibility revolution brought about by Angrist (1990)'s paper and summarized in Angrist and Krueger (2001)'s review paper, the notion of natural experiment made a come back, with the idea that we might be able to look for specific set of events produced by Nature that more credibly identify a relationship of interest, *i.e.* that closely approximate true experimental conditions.

Remark. Outside of economics, Natural Experiments have also flourished, but without the term, and were compiled in the early textbook on research methods

by Campbell (1966). Both Difference In Differences and Regression Discontinuity Designs have been actually developed outside of economics, mostly in education research. Instrumental Variables have had a separate history in economics and in genetics, were it is called the method of path coefficients.

4.1 Instrumental Variables

Instrumental Variables rely on finding a plausibly exogeneous source of variation in treatment intake. In the simple case of a binary instrument, the identification and estimation parts are actually identical to Encouragements designs in RCTs, that we have already studied in Section 3.4. As a consequence, unless we make very strong assumptions, an IV design is going to recover a Local Average Treatment Effect. Our classical assumptions are going to show up again: Independence, Exclusion Restriction, Monotonicity.

Remark. Examples of Instrumental Variables are:

- Distance to college or to school for studying the impact of college or school enrollement on education, earnings and other outcomes.
- Random draft lottery number for investigating the impact of military experience on earnings and other outcomes.
- Randomized encouragement to participate in order to study the impact of a program.

Remark. The crucial part of an IV design is to justify the credibility of the exclusion restriction and independence assumptions. It is in general very difficult to justify these assumptions, especially the exclusion restriction assumption. In the examples above, one could argue that schools or colleges might be built where they are necessary, i.e. close to destitute populations, or, on the contrary, that they are built far from difficult neighbourhoods. As soon as distance to school becomes correlated with other determinants of schooling, such as parental income and education, the independence assumption is violated.

Even if school placement is truly independent of potential education and earnings outcomes at first, parents, by choosing where to live, will sort themselves such as the parents that pay more attention to education end up located closer to school. As a consequence, the independence assumption might be violated again.

Even when the instrument is truly random, such as a draft lottery number, and thus the independence assumption seems fine, the instrument may directly affect the outcomes by other ways than the treatment of interest. For example, receiving a low draft lottery number makes one more likely to be drafted. In response, one might decide to increase their length of stay in college in order to use the waiver for the draft reserved for students. If receiving a low draft lottery number increases the number of years of education, and in turn subsequent earnings, then the exclusion restriction assumption is violated.

In this section, I'm going to denote Z_i a binary instrument that can either take

value 0 or 1. In general, we try to reserve the value 1 for the instrument value that increases participation in the treatment of interest. In our examples, that would be when for example, the distance to college is low, the draft lottery number is low, or someone receives an encouragement to enter a program.

4.1.1 An example where Monotonicity does not hold

Since Monotonicity is going to play such a particular role, and since we have already explored this assumption a little in Chapter 3, I am going to use as an example a model where the Monotonicity assumption actually does not hold. It will, I hope, help us understand better the way Monotonicity works and how it interacts with the other assumptions. The key component of the model that makes Monotonicity necessary is the fact that treatment effects are heterogeneous and correlated with participation in the treatment. We'll see later that Monotonicity is unnecessary when treatment effects are orthogonal to take up.

Example 4.1. Let's see how we can generate a model without Monotonicity:

$$
\begin{aligned}
y_i^1 &= y_i^0 + \bar{\alpha} + \theta\mu_i + \eta_i \\
y_i^0 &= \mu_i + \delta + U_i^0 \\
U_i^0 &= \rho U_i^B + \epsilon_i \\
y_i^B &= \mu_i + U_i^B \\
U_i^B &\sim \mathcal{N}(0, \sigma_U^2) \\
D_i &= \mathbb{1}[y_i^B + \kappa_i Z_i + V_i \leq \bar{y}] \\
\kappa_i &= \begin{cases} -\bar{\kappa} & \text{if } \xi_i = 1 \\ \underline{\kappa} & \text{if } \xi_i = 0 \end{cases} \\
\xi &\sim \mathcal{B}(p_\xi) \\
V_i &= \gamma(\mu_i - \bar{\mu}) + \omega_i \\
(\eta_i, \omega_i) &\sim \mathcal{N}(0, 0, \sigma_\eta^2, \sigma_\omega^2, \rho_{\eta,\omega}) \\
Z_i &\sim \mathcal{B}(p_Z) \\
Z_i &\perp\!\!\!\perp (y_i^0, y_i^1, y_i^B, V_i) \\
\xi_i &\perp\!\!\!\perp (y_i^0, y_i^1, y_i^B, V_i, Z_i)
\end{aligned}
$$

The key component of the model that generates a failure of Monotonicity is the coefficient κ_i, that determines how individuals' participation into the program reacts to the instrument Z_i. κ_i is a coefficient whose value varies accross the population. In my simplified model, κ_i can take only two values, $-\bar{\kappa}$ or $\underline{\kappa}$. When $-\bar{\kappa}$ and $\underline{\kappa}$ have opposite signs (let's say $-\bar{\kappa} < 0$ and $\underline{\kappa} > 0$), then individuals with $\kappa_i = -\bar{\kappa}$ are going to be more likely to enter the program when they receive

an encouragement (when $Z_i = 1$) while individuals with $\kappa_i = \underline{\kappa}$ will be less likely to enter the program when $Z_i = 1$. When $-\bar{\kappa}$ and $\underline{\kappa}$ have different signs, we have four types of reactions when the instrumental variable moves from $Z_i = 0$ to $Z_i = 1$, holding everything else constant. These four types of reactions define four types of individuals:

- **Always takers** ($T_i = a$): individuals that participate in the program both when $Z_i = 0$ and $Z_i = 1$.
- **Never takers** ($T_i = n$): individuals that do not participate in the program both when $Z_i = 0$ and $Z_i = 1$.
- **Compliers** ($T_i = c$): individuals that do not participate in the program when $Z_i = 0$ but that participate in the program when $Z_i = 1$.
- **Defiers** ($T_i = d$): individuals that participate in the program when $Z_i = 0$ but that do not participate in the program when $Z_i = 1$.

In our model, these four types are a function of $y_i^B + V_i$ and κ_i. In order to see this let's define, as in Section 3.4, D_i^z the participation decision of individual i when the instrument is exogenously set to $Z_i = z$, with $z \in \{0, 1\}$. When $\kappa_i = -\bar{\kappa} < 0$, we have three types of reactions to the instrument. It turns out that each of type can be defined by where $y_i^B + V_i$ lies with respect to a series of thresholds:

- **Always takers** ($T_i = a$) are such that $D_i^1 = \mathbb{1}[y_i^B - \bar{\kappa} + V_i \leq \bar{y}] = 1$ and $D_i^0 = \mathbb{1}[y_i^B + V_i \leq \bar{y}] = 1$, so that they actually are such that: $y_i^B + V_i \leq \bar{y}$. This is because $y_i^B + V_i \leq \bar{y} \Rightarrow y_i^B + V_i \leq \bar{y} + \bar{\kappa}$, when $\bar{\kappa} > 0$.
- **Never takers** ($T_i = n$) are such that $D_i^1 = \mathbb{1}[y_i^B - \bar{\kappa} + V_i \leq \bar{y}] = 0$ and $D_i^0 = \mathbb{1}[y_i^B + V_i \leq \bar{y}] = 0$, so that they actually are such that: $y_i^B + V_i > \bar{y} + \bar{\kappa}$. This is because $y_i^B + V_i > \bar{y} + \bar{\kappa} \Rightarrow y_i^B + V_i > \bar{y}$, when $\bar{\kappa} > 0$.
- **Compliers** ($T_i = c$) are such that $D_i^1 = \mathbb{1}[y_i^B - \bar{\kappa} + V_i \leq \bar{y}] = 1$ and $D_i^0 = \mathbb{1}[y_i^B + V_i \leq \bar{y}] = 0$, so that they actually are such that: $\bar{y} < y_i^B + V_i \leq \bar{y} + \bar{\kappa}$.

When $\kappa_i = \underline{\kappa} > 0$, we have three types defined by where V_i lies with respect to a series of thresholds:

- **Always takers** ($T_i = a$) are such that $D_i^1 = \mathbb{1}[y_i^B + \underline{\kappa} + V_i \leq \bar{y}] = 1$ and $D_i^0 = \mathbb{1}[y_i^B + V_i \leq \bar{y}] = 1$, so that they actually are such that: $y_i^B + V_i \leq \bar{y} - \underline{\kappa}$. This is because $y_i^B + V_i \leq \bar{y} - \underline{\kappa} \Rightarrow y_i^B + V_i \leq \bar{y}$, when $\underline{\kappa} > 0$.
- **Never takers** ($T_i = n$) are such that $D_i^1 = \mathbb{1}[y_i^B - \bar{\kappa} + V_i \leq \bar{y}] = 0$ and $D_i^0 = \mathbb{1}[y_i^B + V_i \leq \bar{y}] = 0$, so that they actually are such that: $y_i^B + V_i > \bar{y}$. This is because $y_i^B + V_i > \bar{y} \Rightarrow y_i^B + V_i \leq \bar{y} - \underline{\kappa}$, when $\underline{\kappa} > 0$.
- **Defiers** ($T_i = d$) are such that $D_i^1 = \mathbb{1}[y_i^B + \underline{\kappa} + V_i \leq \bar{y}] = 0$ and $D_i^0 = \mathbb{1}[y_i^B + V_i \leq \bar{y}] = 1$, so that they actually are such that: $\bar{y} - \underline{\kappa} < V_i + y_i^B \leq \bar{y}$.

Let's visualize how this works in a plot. Before that, let's generate some data according to this process. For that, let's choose values for the new parameters.

```
param <- c(8,.5,.28,1500,0.9,0.01,0.05,0.05,0.05,0.1,0.1,7.98,0.5,1,0.5,0.9,0.28,0)
names(param) <- c("barmu","sigma2mu","sigma2U","barY","rho","theta","sigma2epsilon","sigma2eta","
```

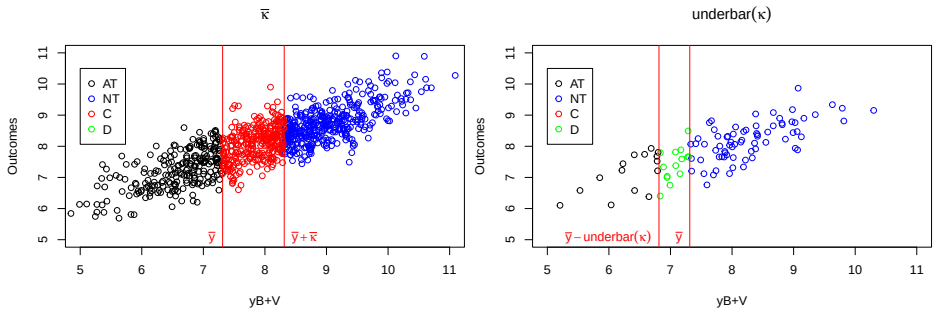
```
set.seed(1234)
N <-1000
cov.eta.omega <- matrix(c(param["sigma2eta"],param["rhoetaomega"]*sqrt(param["sigma2eta"]*param["
eta.omega <- as.data.frame(mvrnorm(N,c(0,0),cov.eta.omega))
colnames(eta.omega) <- c('eta','omega')
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
Ds <- rep(0,N)
Z <- rbinom(N,1,param["pZ"])
xi <- rbinom(N,1,param["pxi"])
kappa <- ifelse(xi==1,-param["barkappa"],param["underbarkappa"])
V <- param["gamma"]*(mu-param["barmu"])+eta.omega$omega
Ds[yB+kappa*Z+V<=log(param["barY"])] <- 1
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta.omega$eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
```

We can now define the types variable T_i:

```
D1 <- ifelse(yB+kappa+V<=log(param["barY"]),1,0)
D0 <- ifelse(yB+V<=log(param["barY"]),1,0)
AT <- ifelse(D1==1 & D0==1,1,0)
NT <- ifelse(D1==0 & D0==0,1,0)
C <- ifelse(D1==1 & D0==0,1,0)
D <- ifelse(D1==0 & D0==1,1,0)
Type <- ifelse(AT==1,'a',
            ifelse(NT==1,'n',
                   ifelse(C==1,'c',
                          ifelse(D==1,'d',""))))

data.non.mono <- data.frame(cbind(Type,C,NT,AT,D1,D0,Y,y,Y1,Y0,y0,y1,yB,alpha,U0,eta.omega$eta,ep
```

```
#ggplot(data.non.mono, aes(x=V, y=yB),color(as.factor(Type))) +
#    geom_point(shape=1)+
#    facet_grid(.~ as.factor(kappa))
```

```
plot(yB[AT==1 & kappa==-param["barkappa"]]+V[AT==1 & kappa==-param["barkappa"]],y[AT==
points(yB[NT==1 & kappa==-param["barkappa"]]+V[NT==1 & kappa==-param["barkappa"]],y[NT=
points(yB[C==1 & kappa==-param["barkappa"]]+V[C==1 & kappa==-param["barkappa"]],y[C==1
points(yB[D==1 & kappa==-param["barkappa"]]+V[D==1 & kappa==-param["barkappa"]],y[D==1
abline(v=log(param["barY"]),col='red')
abline(v=log(param["barY"])+param['barkappa'],col='red')
#abline(v=log(param["barY"])-param['underbarkappa'],col='red')
text(x=c(log(param["barY"]),log(param["barY"])+param['barkappa']),y=c(5,5),labels=c(exp
legend(5,10.5,c('AT','NT','C','D'),pch=c(1,1,1,1),col=c('black','blue','red','green'),
title(expression(kappa=bar(kappa)))

plot(yB[AT==1 & kappa==param["underbarkappa"]]+V[AT==1 & kappa==param["underbarkappa"]]
points(yB[NT==1 & kappa==param["underbarkappa"]]+V[NT==1 & kappa==param["underbarkappa"
points(yB[C==1 & kappa==param["underbarkappa"]]+V[C==1 & kappa==param["underbarkappa"]]
points(yB[D==1 & kappa==param["underbarkappa"]]+V[D==1 & kappa==param["underbarkappa"]]
abline(v=log(param["barY"]),col='red')
#abline(v=log(param["barY"])-param['barkappa'],col='red')
abline(v=log(param["barY"])-param['underbarkappa'],col='red')
text(x=c(log(param["barY"]),log(param["barY"])-param['underbarkappa']),y=c(5,5),labels=
legend(5,10.5,c('AT','NT','C','D'),pch=c(1,1,1,1),col=c('black','blue','red','green'),
title(expression(kappa=underbar(kappa)))
```



(a) $\kappa = \bar{\kappa}$ (b) $\kappa = \underline{\kappa}$

Figure 4.1: Types

As Figure 4.1 shows how the different types interact with κ_i. When $\kappa_i = -\bar{\kappa}$, individuals with $y_i^B + V_i$ below $\bar{y}$ always take the program. Even when $Z_i = 1$ and $\bar{\kappa}$ is subtracted from their index, it is still low enough so that they get to participate. When $y_i^B + V_i$ is in between $\bar{y}$ and $\bar{y} + \bar{\kappa}$, the individuals are such that their index without subtracting $\bar{\kappa}$ is above $\bar{y}$, but it is below $\bar{y}$ when $\bar{\kappa}$ is subtracted from it. These individuals participate when $Z_i = 1$ and do not participate when $Z_i = 0$: they are compliers. Individuals such that $y_i^B + V_i$ is above $\bar{y} + \bar{\kappa}$ will have an index above $\bar{y}$ whether we substract $\bar{\kappa}$ from it or not.

They are never takers.

When $\kappa_i = \underline{\kappa}$, individuals with $y_i^B + V_i$ below $\bar{y} - \underline{\kappa}$ always take the program. Even when $Z_i = 0$ and $\underline{\kappa}$ is not subtracted from their index, it is still low enough so that they get to participate. When $y_i^B + V_i$ is in between $\bar{y} - \underline{\kappa}$ and $\bar{y}$, the individuals are such that their index without adding $\underline{\kappa}$ is below $\bar{y}$, but it is above $\bar{y}$ when $\underline{\kappa}$ is added to it. These individuals participate when $Z_i = 0$ and do not participate when $Z_i = 1$: they are defiers. Individuals such that $y_i^B + V_i$ is above $\bar{y}$ will have an index above $\bar{y}$ whether we add $\underline{\kappa}$ from it or not. They are never takers.

4.1.2 Identification

We need several assumptions for identification in an Instrumental Variable framework. We are going to explore two sets of assumption that secure the identification of two different parameters:

- The Average Treatment Effect on the Treated (TT): identification will happen through the assumption of independence of treatment effects from potential treatment choice
- The Local Average Treatment Effect ($LATE$)

Hypothesis 4.1 (First Stage Full Rank)**.** We assume that the instrument Z_i has a direct effect on treatment participation:

$$\Pr(D_i = 1|Z_i = 1) \neq \Pr(D_i = 1|Z_i = 0).$$

Example 4.2. Let's see how this assumption works in our example. Let's first compute the average values of Y_i and D_i as a function of Z_i, for later use.

```
means.IV <- c(mean(Ds[Z==0]),mean(Ds[Z==1]),mean(y0[Z==0]),mean(y0[Z==1]),mean(y[Z==0]),mean(y[Z=
means.IV <- matrix(means.IV,nrow=2,ncol=4,byrow=FALSE,dimnames=list(c('Z=0','Z=1'),c('D','y0','y'
means.IV <- as.data.frame(means.IV)
```

Figure 4.2 shows that the proportion of treated when $Z_i = 1$ in our sample is equal to 0.53 while the proportion of treated when $Z_i = 0$ is equal to 0.28, in accordance with Assumption 4.1. In the population, the proportion of treated when $Z_i = 1$ depends on the value of κ_i. Let's derive its value:

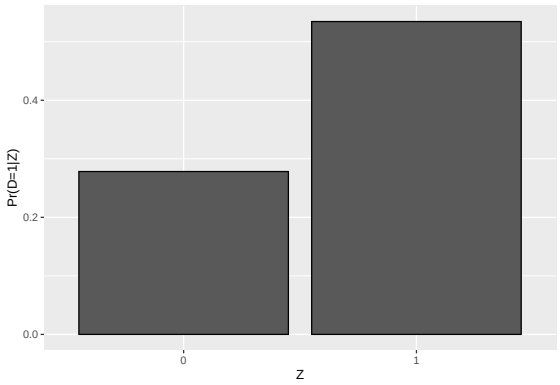


Figure 4.2: Proportion of participants as a function of Z_i

$$
\begin{aligned}
\Pr(D_i = 1|Z_i = 1) &= \Pr(y_i^B + \kappa_i Z_i + V_i \leq \bar{y}|Z_i = 1) \\
&= \Pr(y_i^B + \kappa_i + V_i \leq \bar{y}) \\
&= \Pr(y_i^B + V_i \leq \bar{y} + \bar{\kappa}|\xi_i = 1)\Pr(\xi_i = 1) + \Pr(y_i^B + V_i \leq \bar{y} - \underline{\kappa}|\xi_i = 0)\Pr(\xi_i = 0) \\
&= \Pr(y_i^B + V_i \leq \bar{y} + \bar{\kappa})p_\xi + \Pr(y_i^B + V_i \leq \bar{y} - \underline{\kappa})(1 - p_\xi) \\
&= p_\xi \Phi\left(\frac{\bar{y} + \bar{\kappa} - \bar{\mu}}{\sqrt{(1+\gamma^2)\sigma_\mu^2 + \sigma_U^2 + \sigma_\omega^2}}\right) + (1 - p_\xi)\Phi\left(\frac{\bar{y} - \underline{\kappa} - \bar{\mu}}{\sqrt{(1+\gamma^2)\sigma_\mu^2 + \sigma_U^2 + \sigma_\omega^2}}\right)
\end{aligned}
$$

where the second equality follows from Z_i being independent of $(y_i^0, y_i^1, y_i^B, V_i)$, the third equality follows from ξ_i being independent from $(y_i^0, y_i^1, y_i^B, V_i, Z_i)$ and the last equality follows from the formula for the cumulative of a normal distribution. The formula for $\Pr(D_i = 1|Z_i = 0)$ is the same except for $\bar{\kappa}$ and $\underline{\kappa}$ that are set to zero.

Let's write two functions to compute these probabilities:

```
prob.D.Z.1 <- function(param){
  part.1 <- param['pxi']*pnorm((log(param["barY"])+param['barkappa']-param['barmu'])/sc
  part.2 <- (1-param['pxi'])*pnorm((log(param["barY"])-param['underbarkappa']-param['ba
  return(part.1+part.2)
}
prob.D.Z.0 <- function(param){
  part.1 <- param['pxi']*pnorm((log(param["barY"])-param['barmu'])/sqrt((1+param['gamma
  part.2 <- (1-param['pxi'])*pnorm((log(param["barY"])-param['barmu'])/sqrt((1+param['g
  return(part.1+part.2)
}
```

With these functions, we know that, in the population, $\Pr(D_i = 1|Z_i = 1) =$

0.57 and $\Pr(D_i = 1|Z_i = 0) = 0.25$, which is not far from what we have found in our sample.

Our next set of assumptions imposes that the instrument has no direct effect on the outcome and that it is not correlated with all the potential outcomes. Let's start with the exclusion restriction:

Hypothesis 4.2 (Exclusion Restriction)**.** We assume that there is no direct effect of Z_i on outcomes:

$$\forall d, z \in \{0, 1\}, Y_i^{d,z} = Y_i^d.$$

Example 4.3. In our example, this assumption is automatically satisfied.

Indeed, $y_i^{d,z} = y_i^0 + d(y_i^1 - y_i^0)$ which is parameterized as $y_i^{d,z} = \mu_i + \delta + U_i^0 + d(\bar{\alpha} + \theta\mu_i + \eta_i)$. Since $y_i^{d,z}$ does not depend on z, we have $y_i^{d,z} = y_i^d$, $\forall d, z \in \{0, 1\}$. The assumption would not be satisfied if Z_i entered the equations for y_i^0 or y_i^1. For example, if Z_i is the Vietnam draft lottery number (high or low) used by Angrist to study the impact of army experience on earnings, the exclusion restriction would not work if Z_i was directly influencing outcomes, independent of miitary experience, by example by generating a higher education level. In that case, we could have $E_i = \alpha + \beta Z_i + v_i$, where E_i is education, and, for example, $y_i^0 = \mu_i + \delta + \lambda E_i + U_i^0$. We then have $y_i^{d,z} = \mu_i + \delta + \lambda(\alpha + \beta z + v_i) + U_i^0 + d(\bar{\alpha} + \theta\mu_i + \eta_i)$ which depends on z and thus the exclusion restriction does not hold any more.

Let us now state the independence assumption:

Hypothesis 4.3 (Independence)**.** We assume that Z_i is independent from the other determinants of Y_i and D_i:

$$(Y_i^1, Y_i^0, D_i^1, D_i^0) \perp\!\!\!\perp Z_i.$$

Remark. Why do we say that independence from the potential outcomes is the same as independence from the other determinants of Y_i and D_i? Because the only sources of variation that remain in Y_i^d and D_i^z are the other sources of variations (that is not the treatment $D_i = d$ nor the instrument variable $Z_i = z$).

Example 4.4. In our example, this assumption is also satisfied.

If we assumed that unobserved determinants of earnings contained in U_i^0 are correlated with the instrument value, then we would have a problem. For example, if children that leave close to college have also rich parents, or parents that spend a lot of time with them, or parents with large networks, there probably is a correlation between distance to college and earnings in the absence of the program. For the draft lottery example, you might have that people with a high draft lottery number who have well-connected parents obtain discharges on

special medical grounds. Is that a violation of the independence assumption? Actually no. Indeed, these individuals are simply going to become never takers (they avoid the draft whatever their lottery number). But Z_i is still independent from the level of connections of the parents. For the independence assumption to fail in the draft lottery number example, you would need that children of well-connected parents obtain lower lottery numbers because the lottery is rigged. In that case, since well-connected individuals would have had higher earnings even absent the lottery, there is a negative correlation between y_i^0 and having a high draft lottery number (Z_i).

The last assumption we need in order to identify the Local Average Treatment Effect is that of Monotonicity. We already know this assumption:

Hypothesis 4.4 (Monotonicity)**.** We assume that the instrument moves everyone in the population in the same direction:

$$\forall i, \text{ either } D_i^1 \geq D_i^0 \text{ or } D_i^1 \leq D_i^0.$$

Without loss of generality, we generally assume that $\forall i$, $D_i^1 \geq D_i^0$. As a consequence, there are no defiers.

Example 4.5. In our example, this assumption is not satisfied.

There are defiers, as Figure 4.1 shows, when $\xi_i = 0$ and thus $\kappa_i = \underline{\kappa}$. Indeed, in that case, for the individuals who are such that $\bar{y} - \underline{\kappa} < y_i^B + V_i \leq \bar{y}$, we have $D_i^1 = \mathbb{1}[y_i^B + \underline{\kappa} + V_i \leq \bar{y}] = 0$ and $D_i^0 = \mathbb{1}[y_i^B + V_i \leq \bar{y}] = 1$. This would happen for example if some people would go to college less if their house is located closer to the college, maybe for example because they have a preference not to stay at their parents' house.

Remark. Why are defiers a problem for the instrumental variable strategy? Because the Intention to Treat Effect that measures the difference in expected outcomes at the two levels of the instrument is going to be characterized by two-way flows in and out of the program, as we have already seen with Theorem 3.10. This means that some treatment effects will have negative weights in the ITE formula. In that case, you might have a negative Intention to Treat Effect despite the treatment having positive effects for everyone, or you might under estimate the true effect of the treatment. This matters only when the treatment effects are heterogeneous.

Example 4.6. Let us detail how non-monotonicity and the existence defiers act on the ITE in our example, since we now have defiers. The first very important thing to understand is that all the problems we have happend because treatment effects are heterogeneous **AND** they are correlated with the type of individuals: defiers and compliers do not have the same distribution of treatment effects and, case in point, they do not have the same average treatment effects. The average effects of the treatment on compliers and defiers are not the same. Let us first

look at the distribution of treatment effects among compliers and defiers in the sample and in the population.

In order to derive the distribution of α_i conditional on Type in the population, we need to derive the joint distribution of α_i and $y_i^B + V_i$ and use the **trmtvnorm** package to recover its density when it is truncated. This distribution is normal and fully characterized by its mean and covariance matrix.

$$(\alpha_i, y_i^B + V_i) \sim \mathcal{N}\left(\bar{\alpha} + \theta\bar{\mu}, \bar{\mu}, \begin{pmatrix} \theta^2\sigma_\mu^2 + \sigma_\eta^2 & (\theta+\gamma)\sigma_\mu^2 + \rho_{\eta,\omega}\sigma_\eta^2\sigma_\omega^2 \\ (\theta+\gamma)\sigma_\mu^2 + \rho_{\eta,\omega}\sigma_\eta^2\sigma_\omega^2 & (1+\gamma^2)\sigma_\mu^2 + \sigma_U^2 + \sigma_\omega^2 \end{pmatrix}\right)$$

Let us write a function to generate them.

```
mean.alpha.yBV <- c(param['baralpha']+param['theta']*param['barmu'],param['barmu'])
cov.alpha.yBV <- matrix(c((param['theta']^2)*param['sigma2mu']+param['sigma2eta'],
                          (param['theta']+param['gamma'])*param['sigma2mu']+param['rhoetaomega']*
                            (param['theta']+param['gamma'])*param['sigma2mu']+param['rhoetaomega'
                            (1+param['gamma']^2)*param['sigma2mu']+param['sigma2U']+param['sigma2
# density of alpha for compliers
lower.cut.comp <- c(-Inf,log(param['barY']))
upper.cut.comp <- c(Inf,log(param['barY'])+param['barkappa'])
d.alpha.compliers <- function(x){
  return(dtmvnorm.marginal(xn=x,n=1,mean=mean.alpha.yBV,sigma=cov.alpha.yBV,lower=lower.cut.comp,
}
# density of alpha for defiers
lower.cut.def <- c(-Inf,log(param['barY']-param['underbarkappa']))
upper.cut.def <- c(Inf,log(param['barY']))
d.alpha.defiers <- function(x){
  return(dtmvnorm.marginal(xn=x,n=1,mean=mean.alpha.yBV,sigma=cov.alpha.yBV,lower=lower.cut.def,u
}
```

Let us now plot the empirical and theoretical distributions of the treatment effects for compliers and defiers.

```
# building the data frame
alpha.types <- as.data.frame(cbind(alpha,C,D,AT,NT)) %>%
                mutate(
                  Type = ifelse(AT==1,"Always Takers",
                                ifelse(NT==1,"Never Takers",
                                       ifelse(C==1,"Compliers","Defiers")))
                ) %>%
                mutate(Type = as.factor(Type))

ggplot(filter(alpha.types,Type=="Compliers" | Type=="Defiers"), aes(x=alpha, colour=Type)) +
  geom_density(linetype="dashed") +
  geom_function(fun = d.alpha.compliers, colour = "red") +
```

```
  geom_function(fun = d.alpha.defiers, colour = "blue") +
  ylab('density') +
  theme_bw()
```

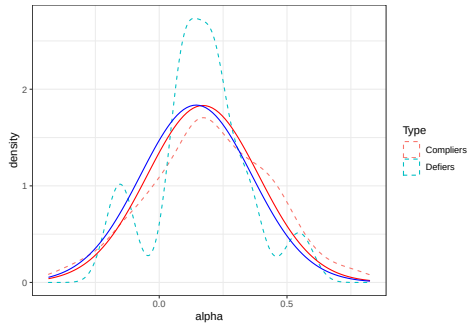


Figure 4.3: Distribution of treatment effects by Type in the sample (dashed line) and in the population (full line)

Figure 4.3 shows that the two distributions are actually very similar in our example. The distribution for the compliers is slightly above that for the defiers, meaning that the defiers should have lower expected outcomes in the population. Let us check that by computing the average outcomes of compliers and defiers both in the sample and in the population.

```
# sample means
mean.alpha.compliers.samp <- mean(alpha[C==1])
mean.alpha.defiers.samp <- mean(alpha[D==1])

# population means
mean.alpha.compliers.pop <- mtmvnorm(mean=mean.alpha.yBV,sigma=cov.alpha.yBV,lower=lowe
mean.alpha.defiers.pop <-  mtmvnorm(mean=mean.alpha.yBV,sigma=cov.alpha.yBV,lower=lowe
```

In the population, the average treatment effect for compliers is equal to 0.17 and the average treatment effect for defiers is equal to 0.14. In the sample, the average treatment effect for compliers is equal to 0.2 and the average treatment effect for defiers is equal to 0.16.

The difference between the treatment effect for compliers and defiers is a problem for the Wald estimator. Let's look at how the Wald estimator behaves in the population (in order to avoid considerations due to sampling noise). By Theorem 3.10, the numerator of the Wald estimator is equal to the difference between the average treatment on compliers and the average treatment effect on defiers weighted by their respective proportions in the population. In order to be able to compute the Wald estimator, we need to compute the proportion of compliers and of defiers in the population. These proportions are equal to:

$$\begin{aligned}
\Pr(T_i = c) &= \Pr(\bar{y} < y_i^B + V_i \leq \bar{y} + \bar{\kappa} \cap \kappa_i = -\bar{\kappa}) \\
&= \Pr(\bar{y} < y_i^B + V_i \leq \bar{y} + \bar{\kappa}) p_\xi \\
\Pr(T_i = d) &= \Pr(\bar{y} - \underline{\kappa} < y_i^B + V_i \leq \bar{y} \cap \kappa_i = \underline{\kappa}) \\
&= \Pr(\bar{y} - \underline{\kappa} < y_i^B + V_i \leq \bar{y})(1 - p_\xi),
\end{aligned}$$

where the second equality follows from the fact that ξ is independent from $y_i^B + V_i$ and uses the fact that $\Pr(A \cap B) = \Pr(A|B)\Pr(B)$. Since $y_i^B + V_i$ is normally distributed and we know its mean and variance, these proportions can be computed as:

$$\Pr(T_i = c) = p_\xi \left(\Phi \left(\frac{\bar{y} + \bar{\kappa} - \bar{\mu}}{\sqrt{(1+\gamma^2)\sigma_\mu^2 + \sigma_U^2 + \sigma_\omega^2}} \right) - \Phi \left(\frac{\bar{y} - \bar{\mu}}{\sqrt{(1+\gamma^2)\sigma_\mu^2 + \sigma_U^2 + \sigma_\omega^2}} \right) \right)$$

$$\Pr(T_i = d) = (1 - p_\xi) \left(\Phi \left(\frac{\bar{y} - \bar{\mu}}{\sqrt{(1+\gamma^2)\sigma_\mu^2 + \sigma_U^2 + \sigma_\omega^2}} \right) - \Phi \left(\frac{\bar{y} - \underline{\kappa} - \bar{\mu}}{\sqrt{(1+\gamma^2)\sigma_\mu^2 + \sigma_U^2 + \sigma_\omega^2}} \right) \right).$$

Let's write functions to compute these objects:

```
# proportion compliers
Prop.Comp <- function(param){
  first <- pnorm((log(param['barY'])+param['barkappa']-param['barmu'])/(sqrt((1+param['gamma']^2)
  second <- pnorm((log(param['barY'])-param['barmu'])/(sqrt((1+param['gamma']^2)*param['sigma2mu'
  return(param['pxi']*(first - second))
}

# proportion defiers
Prop.Def <- function(param){
  first <- pnorm((log(param['barY'])-param['barmu'])/(sqrt((1+param['gamma']^2)*param['sigma2mu']
  second <- pnorm((log(param['barY'])-param['underbarkappa']-param['barmu'])/(sqrt((1+param['gamm
  return((1-param['pxi'])*(first - second))
}
```

In our example, the proportion of compliers is equal to 0.33 and the proportion of defiers is equal to 0.01. As a consequence, the population value of the numerator of the Wald estimator is equal to 0.05. In the Wald estimator, this quantity is divided by the difference between the proportion of participants when $Z_i = 1$ and when $Z_i = 0$. We have already computed this quantity earlier, but it is nice to try to compute it in a different way using the types. The difference in the proportion of participants when $Z_i = 1$ and when $Z_i = 0$ is indeed equal to the difference in the proportion of compliers and the proportion of defiers. The difference between the proportion of compliers and the proportion of defiers is

equal to 0.32, while the difference between the proportion of participants when $Z_i = 1$ and when $Z_i = 0$ is equal to 0.32. It is reassuring that we find the same thing (actually, full disclosure, I did not find the same thing at first, and this help me spot a mistake in the formulas for the proportions of participants: mistakes are normal and natural and that is how we learn and grow).

So we are now equipped to compute the value of the Wald estimator in the population in our model without monotonicity. It is equal to 0.172. In practice, the bias of the Wald estimator is rather small for the average treatment effect on the compliers (remember that it is equal to 0.171). In order to understand why, it is useful to see that the bias of the Wald estimator for the average treatment effect on the compliers is equal to:

$$\mathbb{E}[\Delta_i^Y | T_i = c] - \Delta_{Wald}^Y = \mathbb{E}[\Delta_i^Y | T_i = c] + (\mathbb{E}[\Delta_i^Y | T_i = c] - \mathbb{E}[\Delta_i^Y | T_i = d]) \frac{\Pr(T_i = d)}{\Pr(T_i = c) - \Pr(T_i = d)},$$

where the equality follows from Theorem 3.10 and some algebra. In the absence of Monotonicity, when the impact on defiers is smaller than the impact of compliers, the Wald estimator is baised upward for the effect on the compliers (as it happens in our example). In a model in which the effect of the treatment is larger on defiers than on compliers, the Wald estimator is biased downwards for the effect on compliers because defiers make the outcome of the control group seem too good. In the extreme, when $\mathbb{E}[\Delta_i^Y | T_i = d] > \mathbb{E}[\Delta_i^Y | T_i = c](1 + \frac{\Pr(T_i=c)-\Pr(T_i=d)}{\Pr(T_i=d)})$, the Wald estimator can be negative whereas the effects on compliers and on defiers are both positive. This happens when the effect on defiers is $1 + \frac{\Pr(T_i=c)-\Pr(T_i=d)}{\Pr(T_i=d)}$ times larger than the effect on compliers. In our case, that means that the effect on defiers should be 26 times larger than the effect on compliers for the Wald estimator to be negative, that is to say the effect on defiers should be equal to 4.41, really much much much larger than the effect on compliers.

From there, we are going to explore three strategies in order to identify some true effect of the treatment using the Wald estimator:

- The first strategy has been recently proposed by de Chaisemartin (2017). It is valid in a model without monotonicity.
- The second strategy assumes that the heterogeneity in treatment effects is uncorrelated to the treatment.
- The last strategy is due to Imbens and Angrist (1994) and assumes that Monotonicity holds.

Let's review these solutions in turn.

4.1.2.1 Identification without Monotonicity

The approach delineated by de Chaisemartin (2017) does not assume away non-monotonicity. Clement instead assumes that we can divide the population

of compliers in two-subpopulations: the **compliers-defiers** ($T_i = cd$) and the **surviving-compliers** ($T_i = sc$). The main assumption in Clement's approach is that *(i)* the compliers-defiers are in the same proportion as the defiers and *(ii)* that the average effect of the treatment on the compliers defiers is equal as the average effect of the treatment on the defiers. These two assumptions can be formalized as follows:

Hypothesis 4.5 (Compliers-defiers)**.** We assume that there exists as subpopulation of compliers that are in the same proportion as the defiers and for whom the average effect of the treatment is equal as the average effect of the treatment on the defiers:

$$\begin{aligned}(T_i = c) &= (T_i = cd) \cup (T_i = sc)\\ \Pr(T_i = cd) &= \Pr(T_i = d)\\ \mathbb{E}[Y_i^1 - Y_i^0 | T_i = cd] &= \mathbb{E}[Y_i^1 - Y_i^0 | T_i = d].\end{aligned}$$

The first equation in Assumption 4.5 imposes that the compliers-defiers and the surviving-compliers are a partition of the population of compliers. From Assumption 4.5, we can prove the following theorem:

Theorem 4.1 (Identification of the effect on the surviving-compliers)**.** *Under Assumptions 4.1, 4.2, 4.3 and 4.5, the Wald estimator identifies the effect of the treatment on the surviving-compliers:*

$$\Delta^Y_{Wald} = \Delta^Y_{sc},$$

with:

$$\begin{aligned}\Delta^Y_{Wald} &= \frac{\mathbb{E}[Y_i | Z_i = 1] - \mathbb{E}[Y_i | Z_i = 0]}{\Pr(D_i = 1 | Z_i = 1) - \Pr(D_i = 1 | Z_i = 0)}\\ \Delta^Y_{sc} &= \mathbb{E}[Y_i^1 - Y_i^0 | T_i = sc].\end{aligned}$$

Proof. Under Assumptions 4.2 and 4.3, Theorems 3.10 and 3.12 imply that the numerator of the Wald estimator is equal to Δ^Y_{ITE} with:

$$\Delta^Y_{ITE} = \mathbb{E}[Y_i^1 - Y_i^0 | T_i = c] \Pr(T_i = c) - \mathbb{E}[Y_i^1 - Y_i^0 | T_i = d] \Pr(T_i = d).$$

Now, we have that the effect on compliers can be decomposed in the effect on surviving-compliers and the effect on compliers-defiers using the Law of Iterated Expectations and the fact that $T_i = sc \Rightarrow T_i = c$ and $T_i = cd \Rightarrow T_i = c$:

$$\Delta_c^Y = \mathbb{E}[Y_i^1 - Y_i^0 | T_i = sc] \Pr(T_i = sc | T_i = c) + \mathbb{E}[Y_i^1 - Y_i^0 | T_i = cd] \Pr(T_i = cd | T_i = c),$$

Now, using the fact that $\Pr(T_i = sc|T_i = c) \Pr(T_i = c) = \Pr(T_i = sc)$ and $\Pr(T_i = cd|T_i = c) \Pr(T_i = c) = \Pr(T_i = cd)$ (because $\Pr(A|B) \Pr(B) = \Pr(A \cap B)$ and $\Pr(A \cap B) = \Pr(A)$ if $A \Rightarrow B$), we have:

$$\begin{aligned}\Delta_{ITE}^Y = &\mathbb{E}[Y_i^1 - Y_i^0 | T_i = sc] \Pr(T_i = sc) \\ &+ \mathbb{E}[Y_i^1 - Y_i^0 | T_i = cd] \Pr(T_i = cd) - \mathbb{E}[Y_i^1 - Y_i^0 | T_i = d] \Pr(T_i = d).\end{aligned}$$

The second part of the right-hand side of the above equation is equal to zero by virtue of Assumption 4.5. Now, under Assumptions 4.1, 4.2 and 4.3, we know, from the proof of Theorem 3.9, that $\Pr(D_i = 1|Z_i = 1) - \Pr(D_i = 1|Z_i = 0) = \Pr(T_i = c) - \Pr(T_i = d)$. Under Assumption 4.5, we have $\Pr(T_i = c) = \Pr((T_i = cd) \cup (T_i = sc)) = \Pr(T_i = cd) + \Pr(T_i = sc)$. Replacing $\Pr(T_i = c)$ gives $\Pr(D_i = 1|Z_i = 1) - \Pr(D_i = 1|Z_i = 0) = \Pr(T_i = sc)$. Dividing Δ_{ITE}^Y by $\Pr(T_i = sc)$ gives the result. □

Remark. de Chaisemartin (2017) shows in his Theorem 2.1 that the reciprocal of Theorem 4.1 is actually valid: if there exists surviving-compliers such that their effect is estimated by the Wald estimator and their proportion is equal to the denominator of the Wald estimator, then it has to be that there exists a sub-population of compliers-defiers that are in the same proportion as the defiers and have the same average treatment effect.

Example 4.7. Let us now see if the conditions in de Chaisemartin (2017) are verified in our numerical example.

I have bad news: they are not. It is not super easy to see why, but an intuitive explanation is that the average effect on the defiers in our model is taken conditional on $y_i^B + V_i \in]\bar{y} - \underline{\kappa}, \bar{y}]$ while the effect on compliers is taken conditional on $y_i^B + V_i \in]\bar{y}, \bar{y} + \bar{\kappa}]$. These two intervals do not overlap. Since the expected value of the treatment effect conditional on $y_i^B + V_i = v$ is monotonous in v (because both variables come from a bivariate normal distribution), then all the effects on the defiers interval are either smaller or larger than all the effects on the compliers interval, making it impossible to find a sub-population of compliers that have the same average effect of the treatment as the defiers.

More formally, it is possible to prove this result by using the concept of Marginal Treatment Effect developed by Heckman and Vytlacil (1999). I might devote a specific section of the book to the MTE and its derivations. For now, I let it as a possibility.

What can we do then? Probably the best that we can do is to find κ^* such that $\Pr(\bar{y} < y_i^B + V \leq \bar{y} + \kappa^*) p_\xi = \Pr(T_i = d)$, that is the value such that the

interval of values of $y_i^B + V$ that are for compliers and closest to the interval for defiers and that contains the same proportion of compliers as there are defiers. This value is going to produce an average effect for compliers-defiers as close as possible to the average effect on defiers. It can be computed as follows:

$$\kappa^* = \bar{\mu} - \bar{y} + \sqrt{(1+\gamma^2)\sigma_\mu^2 + \sigma_U^2 + \sigma_\omega^2}\Phi^{-1}\left(\Phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{(1+\gamma^2)\sigma_\mu^2 + \sigma_U^2 + \sigma_\omega^2}}\right)\right.$$
$$\left. + \frac{1-p_\xi}{p_\xi}\left(\Phi\left(\frac{\bar{y}-\bar{\mu}}{\sqrt{(1+\gamma^2)\sigma_\mu^2 + \sigma_U^2 + \sigma_\omega^2}}\right) - \Phi\left(\frac{\bar{y}-\underline{\kappa}-\bar{\mu}}{\sqrt{(1+\gamma^2)\sigma_\mu^2 + \sigma_U^2 + \sigma_\omega^2}}\right)\right)\right)$$

Let's write functions to compute κ^*, the implied proportion of compliers-defiers and the average effect of the treatment on compliers-defiers and on surviving-compliers:

```
# kappa star
KappaStar <- function(param){
  prop.def <- Prop.Def(param)
  prop.below.bary <- pnorm((log(param['barY'])-param['barmu'])/(sqrt((1+param['gamma']^2)*param['
  st.dev.yB.V <- sqrt((1+param['gamma']^2)*param['sigma2mu']+param['sigma2U']+param['sigma2omega'
  return(param['barmu']-log(param['barY'])+st.dev.yB.V*qnorm(prop.below.bary+prop.def/param['pxi'
}

# proportion of compliers-defiers
Prop.Comp.Def <- function(param){
  first <- pnorm((log(param['barY'])+KappaStar(param)-param['barmu'])/(sqrt((1+param['gamma']^2)*
  second <- pnorm((log(param['barY'])-param['barmu'])/(sqrt((1+param['gamma']^2)*param['sigma2mu'
  return(param['pxi']*(first - second))
}

# mean impact on compliers-defiers
lower.cut.comp.def <- c(-Inf,log(param['barY']))
upper.cut.comp.def <- c(Inf,log(param['barY'])+KappaStar(param))
mean.alpha.comp.def.pop <- mtmvnorm(mean=mean.alpha.yBV,sigma=cov.alpha.yBV,lower=lower.cut.comp.

# mean impact on surviving compliers
lower.cut.surv.comp <- c(-Inf,log(param['barY'])+KappaStar(param))
upper.cut.surv.comp <- c(Inf,log(param['barY'])+param['barkappa'])

mean.alpha.surv.comp.pop <- mtmvnorm(mean=mean.alpha.yBV,sigma=cov.alpha.yBV,lower=lower.cut.surv
```

The first have that $\kappa^* = 0.0452$. For this value of κ^*, we have that $\Pr(T_i = cd) = 0.0128$. As expected, this is very close to the proportion of compliers in the population: $\Pr(T_i = d) = 0.0128$. Finally, the average treatment effect on the

compliers-defiers is equal to: $\Delta^y_{cd} = 0.1457$. As expected, but luckily enough, since it was absolutely not sure, it is very close to the to the average treatment effect on the defiers: $\Delta^y_d = 0.1445$. So, in our model, Assumption 4.5 is almost satisfied, and so does Theorem 4.1. As a consequence, the Wald estimator is very close to the effect on the surviving-compliers. Indeed, the Wald estimator, in the population, is equal to $\Delta^y_{Wald} = 0.172156$, while the average effect on surviving-compliers is equal to $\Delta^y_{sc} = 0.172108$.

4.1.2.2 Identification under Independence of treatment effects

Another way to get around the issue of Non-Monotonicity is simply to assume away any meaningful role for treatment effect heterogeneity. One approach to that would simply be to assume that treatment effects are constant across individuals. I leave to the reader to prove that in that case, the Wald estimator would recover the treatment effect under only Independence and Exclusion Restriction. We are going to use a slightly more general approach here by assuming that treatment effect heterogeneity is unrelated to the reaction to the instrument:

Hypothesis 4.6 (Independent Treatment Effects)**.** We assume that the treatment effect is independent from potential reactions to the instrument:

$$\Delta^Y_i \perp\!\!\!\perp (D^1_i, D^0_i).$$

We can now prove that, under Assumption 4.6, the Wald estimator identifies the Average Treatment Effect (ATE), the average effect of the Treatment on the Treated (TT) and the average effect on compliers and on defiers. The first thing to know before we state the result is that, under Assumption 4.6, all these average treatment effects are equal to each other. This is a direct implication of the following lemma:

Lemma 4.1 (Independence of Treatment Effects from Types)**.** *Under Assumption 4.6, the treatment effect is independent from types:*

$$\Delta^Y_i \perp\!\!\!\perp T_i.$$

Proof. Lemma (4.2) in Dawid (1979) states that if $X \perp\!\!\!\perp Y|Z$ and U is a function of X, then $U \perp\!\!\!\perp Y|Z$. Since T_i is a function of (D^1_i, D^0_i) under Assumption 4.6, Lemma 4.1 follows. □

A direct corollary of Lemma 4.1 is:

Corollary 4.1 (Independence of Treatment Effects and Average Effects)**.** *Under Assumption 4.6, the Average Treatment Effect (ATE), the average effect of the*

Treatment on the Treated (TT) and the average effect on compliers and on defiers are all equal:

$$\Delta^Y_{ATE} = \Delta^Y_{TT(1)} = \Delta^Y_{TT(0)} = \Delta^Y_c = \Delta^Y_d .$$

with:

$$\Delta^Y_{TT(z)} = \mathbb{E}[Y^1_i - Y^0_i | D_i = 1, Z_i = z].$$

Proof. Using Lemma 4.1, we have that:

$$\Delta^Y_c = \Delta^Y_d = \Delta^Y_{at} = \Delta^Y_{nt}.$$

Because T_i is a partition, we have $\Delta^Y_{ATE} = \Delta^Y_c \Pr(T_i = c) + \Delta^Y_d \Pr(T_i = d) + \Delta^Y_{at} \Pr(T_i = at) + \Delta^Y_{nt} \Pr(T_i = nt) = \Delta^Y_c$ (since $\Pr(T_i = c) + \Pr(T_i = d) + \Pr(T_i = at) + \Pr(T_i = nt) = 1$). Finally, we also have that $\Delta^Y_{TT(1)} = \Delta^Y_c \Pr(T_i = c | D_i = 1, Z_i = 1) + \Delta^Y_{at} \Pr(T_i = at | D_i = 1, Z_i = 1) = \Delta^Y_c$ and $\Delta^Y_{TT(0)} = \Delta^Y_d \Pr(T_i = d | D_i = 1, Z_i = 0) + \Delta^Y_{at} \Pr(T_i = at | D_i = 1, Z_i = 0) = \Delta^Y_c$, since $(D_i = 1) \cap (Z_i = 1) \Rightarrow (T_i = c) \cup (T_i = at)$ and $(D_i = 1) \cap (Z_i = 0) \Rightarrow (T_i = d) \cup (T_i = at)$. □

We are now equioped to state the final result of this section:

Theorem 4.2 (Identification under Independent Treatment Effect)**.** *Under Assumptions 4.1, 4.2, 4.3 and 4.6, the Wald estimator identifies the average effect of the Treatment on the Treated:*

$$\Delta^Y_{Wald} = \Delta^Y_{TT}.$$

Proof. Using the formula for the Wald estimator, we have, for the two components of its numerator:

$$\begin{aligned}
\mathbb{E}[Y_i | Z_i = 1] &= \mathbb{E}[Y^0_i + (Y^1_i - Y^0_i) D_i | Z_i = 1] \\
&= \mathbb{E}[Y^0_i | Z_i = 1] + \mathbb{E}[\Delta^Y_i | D_i = 1, Z_i = 1] \Pr(D_i = 1 | Z_i = 1) \\
&= \mathbb{E}[Y^0_i | Z_i = 1] + \Delta^Y_{TT(1)} \Pr(D_i = 1 | Z_i = 1) \\
\mathbb{E}[Y_i | Z_i = 0] &= \mathbb{E}[Y^0_i + (Y^1_i - Y^0_i) D_i | Z_i = 0] \\
&= \mathbb{E}[Y^0_i | Z_i = 0] + \mathbb{E}[\Delta^Y_i | D_i = 0, Z_i = 1] \Pr(D_i = 1 | Z_i = 0) \\
&= \mathbb{E}[Y^0_i | Z_i = 0] + \Delta^Y_{TT(0)} \Pr(D_i = 1 | Z_i = 0),
\end{aligned}$$

where the first equalities use Assumption 4.2. Now, under Assumption 4.6, Corollary 4.1 implies that $\Delta^Y_{TT(0)} = \Delta^Y_{TT(1)} = \Delta^Y_{TT}$. We thus have that the numerator of the Wald estimator is equal to:

$$\begin{aligned}\mathbb{E}[Y_i|Z_i = 1] - \mathbb{E}[Y_i|Z_i = 0] = \Delta^Y_{TT}(\Pr(D_i = 1|Z_i = 1) - \Pr(D_i = 1|Z_i = 0)) \\ + \mathbb{E}[Y_i^0|Z_i = 1] - \mathbb{E}[Y_i^0|Z_i = 0].\end{aligned}$$

Assumption 4.3 implies that $\mathbb{E}[Y_i^0|Z_i = 1] = \mathbb{E}[Y_i^0|Z_i = 0]$. Using Assumption 4.1 proves the result. □

4.1.2.3 Identification under Monotonicity

The classical approach to identification using instrumental variables is due to Imbens and Angrist (1994) and Angrist, Imbens and Rubin (1996). It rests on Assumption 4.4 or Monotonicity that we are now familiar with, that requires that the effect of the instrument on treatment participation moves everyone in the same direction.

Remark. For the rest of the section, we will assume that $\forall i$, $D_i^1 \geq D_i^0$. It is without loss of generality, since if the initial treatment does not comply with this requirement, you can simply redefine a new treatment equal to $-D_i$.

Under Monotonicity, there are no defiers. This is what the following lemma shows:

Lemma 4.2. *Under Assumption 4.4, there are no defiers a.s.:*

$$\Pr(T_i = c) = 0.$$

Proof. Under Assumption 4.4, $\forall i$, $D_i^1 \geq D_i^0$. As a consequence, $\Pr(D_i^1 < D_i^0) = 0$. Since defiers are defined as $D_i^1 < D_i^0$, the result follows. □

In the absence of defiers, the Wald estimator identifies the average effect of the treatment on the compliers, also called the Local Average Treatment Effect:

Theorem 4.3. *Under Assumptions 4.1, 4.2, 4.3 and 4.4, the Wald estimator identifies the average effect of the treatment on the compliers, also called the Local Average Treatment Effect:*

$$\Delta^Y_{Wald} = \Delta^Y_{LATE}.$$

Proof. Using Theorem 3.9 directly proves the result. □

Remark. The magic of the instrumental variables setting applies again. By moving the instrument, we are able to learn something about the causal effect of the treatment. Monotonicity is a very strong assumption though, as are Independence and Exclusion Restriction. They are very rarely met in practice. Even the case of RCTs with encouragement design, where Independence holds by design, might be affected by failures of Exclusion Restriction and/or Monotonicity.

Example 4.8. Let's see how monotonicity works in our example.

First, we have to generate a model in which monotonicity holds. For that, we need to shut down heterogeneous reactions to the instrument. In practice, we are going to replace the participation equation in our model, which was characterized by a random coefficient, by the following one, which has a constant coefficient:

$$D_i = \mathbb{1}[y_i^B - \bar{\kappa} Z_i + V_i \leq \bar{y}]$$

As a consequence, we have no more defiers and monotonicity holds. Let us now generate the data from the model with monotonicity:

```
set.seed(12345)
N <-1000
cov.eta.omega <- matrix(c(param["sigma2eta"],param["rhoetaomega"]*sqrt(param["sigma2eta"]*param["
eta.omega <- as.data.frame(mvrnorm(N,c(0,0),cov.eta.omega))
colnames(eta.omega) <- c('eta','omega')
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
Ds <- rep(0,N)
Z <- rbinom(N,1,param["pZ"])
V <- param["gamma"]*(mu-param["barmu"])+eta.omega$omega
Ds[yB-param["barkappa"]*Z+V<=log(param["barY"])] <- 1
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta.omega$eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
```

We can now define the types variable T_i:

```
D1 <- ifelse(yB-param["barkappa"]+V<=log(param["barY"]),1,0)
D0 <- ifelse(yB+V<=log(param["barY"]),1,0)
```

```
AT <- ifelse(D1==1 & D0==1,1,0)
NT <- ifelse(D1==0 & D0==0,1,0)
C <- ifelse(D1==1 & D0==0,1,0)
D <- ifelse(D1==0 & D0==1,1,0)
Type <- ifelse(AT==1,'a',
            ifelse(NT==1,'n',
                   ifelse(C==1,'c',
                          ifelse(D==1,'d',""))))

data.mono <- data.frame(cbind(Type,C,NT,AT,D1,D0,Y,y,Y1,Y0,y0,y1,yB,alpha,U0,eta.omega
```

The first thing we can check is that there are no defiers. For that, let's count the number of individuals who have $T_i = 1$. It is equal to 0.

One thing that helped me understand how the IV approach under monotonicity works is the following graph:

```
plot(yB[AT==1]+V[AT==1],y[AT==1],pch=1,xlim=c(5,11),ylim=c(5,11),xlab="yB+V",ylab="Out
points(yB[NT==1]+V[NT==1],y[NT==1],pch=1,col='blue')
points(yB[C==1 & Ds==1]+V[C==1 & Ds==1],y[C==1 & Ds==1],pch=1,col='red')
points(yB[C==1 & Ds==0]+V[C==1 & Ds==0],y[C==1 & Ds==0],pch=1,col='green')
abline(v=log(param["barY"]),col="red")
abline(v=log(param["barY"])+param['barkappa'],col="red")
text(x=c(log(param["barY"]),log(param["barY"])+param['barkappa']),y=c(5,5),labels=c(ex
legend(5,10.5,c('AT','NT','C|D=1','C|D=0'),pch=c(1,1,1,1),col=c("black",'blue',"red",'
```

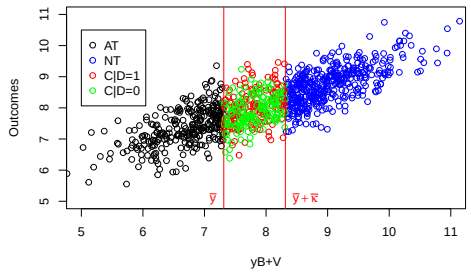


Figure 4.4: Types under Monotonicity

What 4.4 shows is that the IV acts as a randomized controlled trial among compliers. Within the population of compliers, whether one receives the treatment or not is as good as random. If we actually knew who the compliers were, we could directly estimate the effect of the treatment by comparing the outcomes of the treated compliers to the outcomes of the untreated compliers. Actually, this approach, applied in our sample, yields an estimated treatment effect on the compliers of 0.14, whereas the simple comparison of participants and non participants would give an estimate of -0.93. In our sample, the average effect of the treatment on compliers is actually equal to 0.18.

Let us finally check that Theorem 4.3 works in the population in our new model. We need to compute the various parts of the Wald estimator and the average effect of the treatment on the compliers. The key to understand the Wald estimator is to see that its numerator is composed of the difference between two means, with both means containing the average outcomes of always takers and never takers weighted by their respective proportions in the population, as shown in the proof of Theorem 4.3. These two means cancel out, leaving only the differences in the means of the compliers in and out of the treatment, weighted by their proportion in the population. The denominator of the Wald estimator simply provides an estimate of the proportion of compliers. In order to illustrate these intuitions in our example, I am going to use the formula for a truncated multivariate normal variable and the package `tmvtnorm`. The most important thing to notice here is that $(y_i^0, y_i^1, y_i^B + V_i) \sim \mathcal{N}(\bar{\mu} + \delta, \bar{\mu}(1+\theta) + \delta + \bar{\alpha}, \bar{\mu}, \mathbf{C})$ with:

$$\mathbf{C} = \begin{pmatrix} \sigma_\mu^2 + \rho^2\sigma_U^2 + \sigma_\epsilon^2 & (1+\theta)\sigma_\mu^2 + \rho^2\sigma_U^2 + \sigma_\epsilon^2 & (1+\gamma)\sigma_\mu^2 + \rho\sigma_U^2 \\ (1+\theta)\sigma_\mu^2 + \rho^2\sigma_U^2 + \sigma_\epsilon^2 & (1+\theta^2)\sigma_\mu^2 + \rho^2\sigma_U^2 + \sigma_\epsilon^2 + \sigma_\eta^2 & (1+\theta+\gamma)\sigma_\mu^2 + \rho\sigma_U^2 + \rho_{\eta,\omega}\sigma_\eta^2\sigma_\omega^2 \\ (1+\gamma)\sigma_\mu^2 + \rho\sigma_U^2 & (1+\theta+\gamma)\sigma_\mu^2 + \rho\sigma_U^2 + \rho_{\eta,\omega}\sigma_\eta^2\sigma_\omega^2 & (1+\gamma^2)\sigma_\mu^2 + \sigma_U^2 + \sigma_\omega^2 \end{pmatrix}$$

We now simply have to derive the mean outcomes and proportions of each type in the population in order to form the Wald estimator. Let me first derive the joint distribution of the portential outcomes and the means and proportions of each type in the population.

```
mean.y0.y1.yBV <- c(param['barmu']+param['delta'],param['barmu']*(1+param['theta'])+param['delta'
cov.y0.y1.yBV <- matrix(c(param['sigma2mu']+param['rho']^2*param['sigma2U']+param['sigma2epsilon'
                          (1+param['theta'])*param['sigma2mu']+param['rho']^2*param['sigma2U']+pa
                          (1+param['gamma'])*param['sigma2mu']+param['rho']*param['sigma2U'],
                          (1+param['theta'])*param['sigma2mu']+param['rho']^2*param['sigma2U']+pa
                          (1+param['theta']^2)*param['sigma2mu']+param['rho']^2*param['sigma2U']+
                          (1+param['theta']+param['gamma'])*param['sigma2mu']+param['rho']*param[
                          (1+param['gamma'])*param['sigma2mu']+param['rho']*param['sigma2U'],
                          (1+param['theta']+param['gamma'])*param['sigma2mu']+param['rho']*param[
                          (1+param['gamma']^2)*param['sigma2mu']+param['sigma2U']+param['sigma2om

# cuts
#always takers
lower.cut.at <- c(-Inf,-Inf,-Inf)
upper.cut.at <- c(Inf,Inf,log(param['barY']))
# compliers
lower.cut.comp <- c(-Inf,-Inf,log(param['barY']))
upper.cut.comp <- c(Inf,Inf,log(param['barY'])+param['barkappa'])
# never takers
lower.cut.nt <- c(-Inf,-Inf,log(param['barY'])+param['barkappa'])
upper.cut.nt <- c(Inf,Inf,Inf)
```

```
# means by types
#always takers
mean.y1.at.pop <- mtmvnorm(mean=mean.y0.y1.yBV,sigma=cov.y0.y1.yBV,lower=lower.cut.at,u
mean.y0.at.pop <- mtmvnorm(mean=mean.y0.y1.yBV,sigma=cov.y0.y1.yBV,lower=lower.cut.at,u
# never takers
mean.y1.nt.pop <- mtmvnorm(mean=mean.y0.y1.yBV,sigma=cov.y0.y1.yBV,lower=lower.cut.nt,u
mean.y0.nt.pop <- mtmvnorm(mean=mean.y0.y1.yBV,sigma=cov.y0.y1.yBV,lower=lower.cut.nt,u
#compliers
mean.y1.comp.pop <- mtmvnorm(mean=mean.y0.y1.yBV,sigma=cov.y0.y1.yBV,lower=lower.cut.co
mean.y0.comp.pop <- mtmvnorm(mean=mean.y0.y1.yBV,sigma=cov.y0.y1.yBV,lower=lower.cut.co

# Proportion of each types
# always takers
prop.at.pop <- ptmvnorm.marginal(log(param['barY']),n=3,mean=mean.y0.y1.yBV,sigma=cov.y
# never takers
prop.nt.pop <- 1-ptmvnorm.marginal(log(param['barY'])+param['barkappa'],n=3,mean=mean.y
# compliers
prop.comp.pop <- ptmvnorm.marginal(log(param['barY'])+param['barkappa'],n=3,mean=mean.y

# LATE
late.pop <- mean.y1.comp.pop-mean.y0.comp.pop
late.prop.comp.pop <- late.pop*prop.comp.pop
# Wald
num.Wald.pop <- (mean.y1.comp.pop*prop.comp.pop+mean.y1.at.pop*prop.at.pop+mean.y0.nt.p
denom.Wald.pop <- (prop.at.pop+prop.comp.pop-prop.at.pop)
Wald.pop <- num.Wald.pop/denom.Wald.pop
```

We are now equipped to compute the Wald estimator in the population. Before that, let us compute the LATE. We have $\Delta^Y_{LATE} = 0.179$. The Wald estimator is equal to $\Delta^Y_{Wald} = 0.179$. They are obviously equal. This is because the numerator of the Wald is equal to the product of the LATE multiplied by the proportion of compliers (which is equal to 0.066). This is because the outcomes of never takers and always takers cancel out on each separate term of the numerator of the Wald estimator. Indeed, we have that the numerator of the Wald estimator is equal to: 0.066.

4.1.3 Estimation

Estimation of the LATE under the IV assumptions closely follows the same steps that we have delineated in Section 3.4.2:

1. **First stage** regression of D_i on Z_i: this estimates the impact of the instrument on participation into the program and estimates the proportion of compliers.
2. **Reduced form** regression of Y_i on Z_i: this estimates the impact of the instrument on outcomes, *a.k.a* the ITE.

3. **Structural** regression of Y_i on D_i using Z_i as an instrument, which estimates the LATE.

Let's take these three steps in turn.

4.1.3.1 First stage regression

The first stage regression regresses D_i on Z_i and thus estimates the impact of the instrument on treatment participation, which is equal to the proportion of compliers. It can be run using the With/Without estimator or OLS (both are numerically equivalent as Lemma A.3 shows) or OLS conditioning on observed covariates.

Example 4.9. Let's see how these three approaches fare in our example.

```
# WW first stage
WW.First.Stage.IV <- mean(Ds[Z==1])-mean(Ds[Z==0])
# Simple OLS
OLS.D.Z.IV <- lm(Ds~Z)
OLS.First.Stage.IV <- coef(OLS.D.Z.IV)[[2]]
# OLS conditioning on yB
OLS.D.Z.yB.IV <- lm(Ds~Z+yB)
OLSX.First.Stage.IV <- coef(OLS.D.Z.yB.IV)[[2]]
```

The WW estimator of the first stage impact of Z_i on D_i is equal to 0.374. The OLS estimator of the first stage impact of Z_i on D_i is equal to 0.374. The OLS estimator of the first stage impact of Z_i on D_i conditioning on y_i^B is equal to 0.339. Remember that the true proportion of compliers in the population in our model is equal to 0.366.

4.1.3.2 Reduced form regression

The reduced form regression regresses Y_i on Z_i and thus estimates the impact of the instrument on outcomes, which is equal to the ITE. It can be run using the With/Without estimator or OLS (both are numerically equivalent as Lemma A.3 shows) or OLS conditioning on observed covariates.

Example 4.10. Let's see how these three approaches fare in our example.

```
# WW reduced form
WW.Reduced.Form.IV <- mean(y[Z==1])-mean(y[Z==0])
# Simple OLS
OLS.y.Z.IV <- lm(y~Z)
OLS.Reduced.Form.IV <- coef(OLS.y.Z.IV)[[2]]
# OLS conditioning on yB
OLS.y.Z.yB.IV <- lm(y~Z+yB)
OLSX.Reduced.Form.IV <- coef(OLS.y.Z.yB.IV)[[2]]
```

The WW estimator of the reduced form impact of Z_i on y_i is equal to -0.029.

The OLS estimator of the reduced form impact of Z_i on y_i is equal to -0.029. The OLS estimator of the reduced form impact of Z_i on y_i conditioning on y_i^B is equal to 0.058. Remember that the true ITE in the population in our model is equal to 0.066.

4.1.3.3 Structural regression

The final step of the analysis is to estimate the impact of D_i on Y_i using Z_i as an instrument. This can be done either by directly using the Wald estimator, by dividing the estimate of the reduced form by the result of the first stage, or by directly using the IV estimator (which is equivalent to the Wald estimator as Theorem 3.15 shows) or the IV estimator conditional on covariates.

Example 4.11. Let's see how these four approaches fare in our example.

```
# Wald structural form
Wald.Structural.Form.IV <- (mean(y[Z==1])-mean(y[Z==0]))/(mean(Ds[Z==1])-mean(Ds[Z==0]))
# Simple IV
TSLS.y.D.Z.IV <- ivreg(y~Ds|Z)
TSLS.Structural.Form.IV <- coef(TSLS.y.D.Z.IV)[[2]]
# IV conditioning on yB
TSLS.y.D.Z.yB.IV <- ivreg(y~Ds+yB|Z+yB)
TSLSX.Structural.Form.IV <- coef(TSLS.y.D.Z.yB.IV)[[2]]
```

The Wald estimator of the LATE is equal to $\hat{\Delta}^y_{Wald}$ = -0.078. The IV estimator of the LATE is equal to $\hat{\Delta}^y_{IV}$ = -0.078, and is numerically identical to the Wald estimator, as expected. The IV estimator of the LATE conditioning on y_i^B is equal to 0.172. Remember that the true LATE in the population in our model is equal to 0.179.

Remark. The last thing we might want to check is what the sampling noise of the IV estimator looks like and whether it is reduced by conditioning on observed covariates.

Example 4.12. Let's see how sampling noise moves in our example.

Do it

4.1.4 Estimation of sampling noise

Remark. The framework we have seen here as been extended to multivalued instruments or treatments by several papers. Imbens and Angrist (1994) extend the framework to an ordered instrument. They show that the 2SLS estimator is a weighted average of LATEs for each values of the instrument, with positive weights summing to one. Angrist and Imbens (1995) extend the framework to he case where the treatment is an ordered discrete variable and there are multiple dichotomous instruments. They again show that the 2SLS estimator is a weighted average of LATEs with positive weights summing to one. Heckman and Vytlacil (1999) extend the framework to a case with a continuous instrument

and show that one can the define a Marginal Treatment Effect (or MTE) that is equal to the effect of the treatment on individuals that have the same unobserved propensity to take the treatment. They show that the MTE can be identified by a limiting form of Wald estimator that they call a Local Instrumental Variable estimator. They also show that average treatment effect parameters such as TT, ATE and LATE are all weighted averages of the MTE, with positive weights summing to one. Under strong support conditions on the side of the instrument, one can thus in principle recover all treatment effect parameters with a continuous instrument.

Remark. One important concern with the first stage regression is that of weak instruments. When Assumption 4.1 does not hold and the impact on the instrument on take up is actually zero in the population, the Wald estimator is not well-defined.

Expand

4.2 Regression Discontinuity Designs

4.3 Difference In Differences

In Difference In Differences (a.k.a. DID), the difference between treated and untreated before the treatment is used to approximate selection bias. As a consequence, DID works by correcting the With/Without comparison after treatment by the With/Without comparison before treatment and hopes that it is enough to recover the TT. Hence the name Difference in Differences (DID), since the estimator, in its simplest form, is a difference between two differences. In this section, we are going to look at identification using DID, estimation and estimation of sampling noise. At first, we are going to assume that we have only access to two time periods. In that case, estimation and inference are pretty straightforward. We will then examine the case of several time periods, but we will first allow for only one treatment date. In that case, we will introduce the standard tools used by applied researchers to analyze these types of designs: the event study graph and the Two-Way Fixed Effects estimator (a.k.a. TWFE). We will determine which effect is estimated by the TWFE estimator and what are the goals of the event study graph. We will then look at the most complex case: the staggered design, where we have several time periods (strictly more than two) and the date of treatment differs across units. In the staggered design, troubles start appearing for the TWFE estimator. We will survey these problems and the proposed solutions to address them. Finally, we will look at the combination of DID with instrumental variables (the DID-IV estimator) and see which specific types of problems happen there as well. Let's get to it.

4.3.1 Difference In Differences with two time periods

Before getting into the rigorous derivations, let's start with a very simple illustration using our workhorse example.

Example 4.13. How does DID perform and what does it look like in our example model?

Let's first generate a dataset with selection bias.

$$
\begin{aligned}
y_i^1 &= y_i^0 + \bar{\alpha} + \theta\mu_i + \eta_i \\
y_i^0 &= \mu_i + \delta + U_i^0 \\
U_i^0 &= \rho U_i^B + \epsilon_i \\
y_i^B &= \mu_i + U_i^B \\
U_i^B &\sim \mathcal{N}(0, \sigma_U^2) \\
D_i &= \mathbb{1}[y_i^B + V_i \leq \bar{y}] \\
V_i &= \gamma(\mu_i - \bar{\mu}) + \omega_i \\
(\eta_i, \omega_i) &\sim \mathcal{N}(0, 0, \sigma_\eta^2, \sigma_\omega^2, \rho_{\eta,\omega})
\end{aligned}
$$

Let's see how DID works on this data.

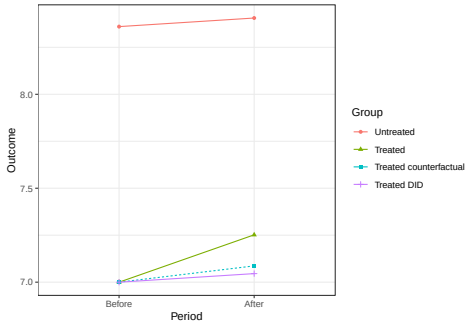


Figure 4.5: Evolution of average outcomes in the treated and control group

Figure 4.5 shows the evolution of the mean log-outcomes for the treated and untreated groups over time in our simulated dataset. We can see that in the **Before** period, outcomes (y_i^B in that case) are much higher for the non participants than for the participants, in agreement with the selection rule that makes participation into the program more likely for individuals with lower pre-treatment outcomes. The With/Without difference in outcomes before the program takes place is $\hat{\Delta}_{WW}^{y^B}$ = -1.361. Second, we see that the difference between participants and non-participants decreases after receiving the treatment. This is because the outcomes of the participants increase faster than the outcomes of the

non participants. As a consequence, the With/Without difference in outcomes after the program takes place is $\hat{\Delta}^y_{WW}$ = -1.154. The DID estimator is built by comparing these two differences. In our example, $\hat{\Delta}^y_{DID} = 0.206$. It is not too far from the true treatment effect of $\hat{\Delta}^y_{TT} = 0.165$.

Figure 4.5 also demonstrates that the DID estimator can also be seen as the difference between the Before/After differences in outcomes of the treated and the untreated. The Before/After difference in outcomes for the non participants is $\hat{\Delta}^y_{BA|D=0} = 0.046$ while the Before/After difference for the participants is $\hat{\Delta}^y_{BA|D=1} = 0.252$, leading to the same DID estimand. One way to understand the DID estimator is to see it as recreating the counterfactual trajectory of the participants (show as a discontinuous line on Figure 4.5) by using the trajectory of the non participants and making it start at the pre-treatment level of the participants. This estimated counterfactual trajectory is shown as the purple continuous line at the bottom of Figure 4.5. In our example, the true counterfactual trajectory (the discontinuous line) and the estimated counterfactual trajectory almost coincide, making the estimated counterfactual outcome of the participants very close to their true counterfactual outcome (7.046 vs 7.087). The difference between these two quantities measures the bias of the DID estimator, and we can see that it is very low in our example. The fact that the Before/After difference in outcomes for the non participants approximates well the counterfactual Before/After difference in outcomes for the participants is **THE** crucial assumption of the DID estimator. It is called the parallel trends assumption.

4.3.1.1 Identification

The formal setting for introducing the DID estimator is to start with two time periods, **Before** and **After** ($t = B$ and $t = A$ respectively). Outcomes with and without the treatment in both periods are denoted $Y^d_{i,t}$, for $d \in \{0, 1\}$ and $t \in \{B, A\}$. Treatment participation in both periods is denoted $D_{i,t}$ for $t \in \{B, A\}$. In the Before period, the treatment is unavailable, so that we get to observe the potential outcomes of the agents in the absence of the treatment. These two very specific requirements of DID are encoded in the following way:

Hypothesis 4.7 (No Treatment in the Before Period)**.** We assume that no unit in the population receives the treatment in the Before period: $D_{i,B} = 0$, $\forall i$.

Under Assumption 4.7, and without loss of generality, we are going to write $D_i = D_{i,A}$.

Hypothesis 4.8 (No Anticipation Effects)**.** We assume that, in the Before period, agents cannot anticipate that the program will happen in the After period, or that they do not change their behavior as a consequence: $Y_{i,B} = Y^0_{i,B}$, $\forall i$.

A consequence of Assumptions 4.7 and 4.8 is that we can write observed outcomes

as a function of treatment and potential outcomes using the usual switching equation:

$$Y_{i,t} = Y_{i,t}^1 D_{i,t} + Y_{i,t}^0 (1 - D_{i,t}). \tag{4.1}$$

The final very important assumption that we can make is to assume that the trends in the potential outcomes in the absence the treatment are the same for the treated and the untreated units:

Hypothesis 4.9 (Parallel Trends)**.** We assume that the trends in the potential outcomes in the absence the treatment are the same for the treated and the untreated units:

$$\mathbb{E}[Y_{i,A}^0 | D_i = 1] - \mathbb{E}[Y_{i,B}^0 | D_i = 1] = \mathbb{E}[Y_{i,A}^0 | D_i = 0] - \mathbb{E}[Y_{i,B}^0 | D_i = 0].$$

Assumption 4.9 is actually equivalent to assuming that selection bias is constant over time. This is what this very simple lemma shows:

Lemma 4.3 (Parallel Trends is Constant Selection Bias)**.** *Assumption 4.9 is equivalent to assuming that selection bias is constant over time:*

$$\mathbb{E}[Y_{i,A}^0 | D_i = 1] - \mathbb{E}[Y_{i,A}^0 | D_i = 0] = \mathbb{E}[Y_{i,B}^0 | D_i = 1] - \mathbb{E}[Y_{i,B}^0 | D_i = 0].$$

Proof. The proof follows immediately by adding $\mathbb{E}[Y_{i,B}^0 | D_i = 1] - \mathbb{E}[Y_{i,A}^0 | D_i = 0]$ to both sides of the equation in Assumption 4.9. □

Under these assumptions, we are ready to state the main identification result of this section:

Theorem 4.4 (DID identifies TT)**.** *Under Assumptions 4.7, 4.8 and 4.9, the DID estimator identifies the average effect of the Treatment on the Treated after the treatment:*

$$\Delta_{DID}^{Y} = \Delta_{TT}^{Y_A},$$

with:

$$\Delta_{DID}^{Y} = \mathbb{E}[Y_{i,A} | D_i = 1] - \mathbb{E}[Y_{i,B} | D_i = 1] - (\mathbb{E}[Y_{i,A} | D_i = 0] - \mathbb{E}[Y_{i,B} | D_i = 0]),$$
$$\Delta_{TT}^{Y_A} = \mathbb{E}[Y_{i,A}^1 - Y_{i,A}^0 | D_i = 1].$$

Proof.

$$\begin{aligned}\Delta^Y_{DID} &= \mathbb{E}[Y_{i,A}|D_i=1] - \mathbb{E}[Y_{i,B}|D_i=1] - (\mathbb{E}[Y_{i,A}|D_i=0] - \mathbb{E}[Y_{i,B}|D_i=0]) \\ &= \mathbb{E}[Y^1_{i,A}|D_i=1] - \mathbb{E}[Y^0_{i,B}|D_i=1] - (\mathbb{E}[Y^0_{i,A}|D_i=0] - \mathbb{E}[Y^0_{i,B}|D_i=0]) \\ &= \mathbb{E}[Y^1_{i,A}|D_i=1] - \big(\mathbb{E}[Y^0_{i,A}|D_i=0] + (\mathbb{E}[Y^0_{i,B}|D_i=1] - \mathbb{E}[Y^0_{i,B}|D_i=0])\big)\end{aligned}$$

where the second equality follows from Assumptions 4.7 and 4.8 and the switching equation, and the third equality follows from Lemma 4.3. Under Assumption 4.9, we have:

$$\mathbb{E}[Y^0_{i,A}|D_i=1] = \mathbb{E}[Y^0_{i,A}|D_i=0] + (\mathbb{E}[Y^0_{i,B}|D_i=1] - \mathbb{E}[Y^0_{i,B}|D_i=0])$$

As a consequence, we have:

$$\begin{aligned}\Delta^Y_{DID} &= \mathbb{E}[Y^1_{i,A}|D_i=1] - \mathbb{E}[Y^0_{i,A}|D_i=1] \\ &= \mathbb{E}[Y^1_{i,A} - Y^0_{i,A}|D_i=1] \\ &= \Delta^{Y_A}_{TT}.\end{aligned}$$

□

Example 4.14. How does the DID estimator behave in our example?

The Before/After comparison among the participants is equal to $\hat{\Delta}^y_{BA|D=1} =$ 0.252. The Before/After comparison among the non-participants is equal to $\hat{\Delta}^y_{BA|D=0} = 0.046$. The DID estimator is thus equal to $\hat{\Delta}^y_{DID} = \hat{\Delta}^y_{BA|D=1} - \hat{\Delta}^y_{BA|D=0} = 0.252 - 0.046 = 0.206$. It is also equal to the difference between the before and after With/Without estimators. The Before With/Without estimator is equal to $\hat{\Delta}^{y^B}_{WW}$ = -1.361. The After With/Without estimator is equal to $\hat{\Delta}^y_{WW}$ = -1.154. The DID estimator is thus equal to $\hat{\Delta}^y_{DID} = \hat{\Delta}^y_{WW} - \hat{\Delta}^{y^B}_{WW} =$ -1.154 −(-1.361) = 0.206. This is not too far from the true effect of the treatment in the sample which is equal to $\hat{\Delta}^y_{TT} = 0.165$.

Now, another very important question is whether the DID estimator is consistent, that is whether it is equal to Δ^y_{TT} in our model. A necessary and sufficient condition for that is for the Parallel Trends Assumption 4.9 to hold. Indeed, it can be shown that the bias of the DID estimator is $\Delta^y_{B(DID)} = \Delta^y_{DID} - \Delta^y_{TT} = \mathbb{E}[y^0_i|D_i=1] - \mathbb{E}[y^B_i|D_i=1] - (\mathbb{E}[y^0_i|D_i=0] - \mathbb{E}[y^B_i|D_i=0])$. Let us derive $\Delta^y_{B(DID)}$ in our example. Let us compute the trend in potential outcomes among the treated:

$$\begin{aligned}
&\mathbb{E}[y^0_{i,A}|D_i=1] - \mathbb{E}[y^0_{i,B}|D_i=1] \\
&\qquad = \mathbb{E}[y^0_i|D_i=1] - \mathbb{E}[y^B_i|D_i=1] \\
&\qquad = \mathbb{E}[\mu_i + \delta + U^0_i|D_i=1] - \mathbb{E}[\mu_i + U^B_i|D_i=1] \\
&\qquad = \mathbb{E}[\mu_i|D_i=1] + \delta + \mathbb{E}[U^0_i|D_i=1] \\
&\qquad\quad - \mathbb{E}[\mu_i|D_i=1] - \mathbb{E}[U^B_i|D_i=1] \\
&\qquad = \delta + \mathbb{E}[\rho U^B_i + \epsilon_i|D_i=1] - \mathbb{E}[U^B_i|D_i=1] \\
&\qquad = \delta - (1-\rho)\mathbb{E}[U^B_i|D_i=1].
\end{aligned}$$

Following the same line of reasoning, the trend in potential outcomes among the untreated is:

$$\mathbb{E}[y^0_i|D_i=0] - \mathbb{E}[y^B_i|D_i=0] = \delta - (1-\rho)\mathbb{E}[U^B_i|D_i=0].$$

As a consequence, the bias of the DID estimator in our model is:

$$\begin{aligned}
\Delta^y_{B(DID)} &= -(1-\rho)(\mathbb{E}[U^B_i|D_i=1] - \mathbb{E}[U^B_i|D_i=0]) \\
&= -(1-\rho)(\mathbb{E}[U^B_i|\mu_i + U^B_i + V_i \leq \bar{y}] - \mathbb{E}[U^B_i|\mu_i + U^B_i + V_i > \bar{y}])
\end{aligned}$$

Is this zero? The answer actually is that it is not. In order to see why, notice intuitively that the conditional expectation of U^B_i is taken conditional on something correlated with U^B_i being above or below some threshold. As a consequence, the two values whose difference is taken in the parenthesis cannot be equal. More formally, let us derive the formula for the bias of the DID estimator in our model, using the formula for the expectation of a truncated bivariate normal distribution:

$$\begin{aligned}
\Delta^y_{B(DID)} &= -(1-\rho)(\mathbb{E}[U^B_i|\mu_i + U^B_i + V_i \leq \bar{y}] - \mathbb{E}[U^B_i|\mu_i + U^B_i + V_i > \bar{y}]) \\
&= (1-\rho)\left(\frac{\sigma^2_U}{(1+\gamma^2)\sigma^2_\mu + \sigma^2_U + \sigma^2_\omega}\right)\left(\frac{\phi\left(\frac{\bar{y}-\bar{\mu}}{(1+\gamma^2)\sigma^2_\mu+\sigma^2_U+\sigma^2_\omega}\right)}{\Phi\left(\frac{\bar{y}-\bar{\mu}}{(1+\gamma^2)\sigma^2_\mu+\sigma^2_U+\sigma^2_\omega}\right)} + \frac{\phi\left(\frac{\bar{y}-\bar{\mu}}{(1+\gamma^2)\sigma^2_\mu+\sigma^2_U+\sigma^2_\omega}\right)}{1-\Phi\left(\frac{\bar{y}-\bar{\mu}}{(1+\gamma^2)\sigma^2_\mu+\sigma^2_U+\sigma^2_\omega}\right)}\right)
\end{aligned}$$

In order to compute the value of this parameter, and of the average treatment effect, we are going to use the package `tmtvnorm` which provides the moments from a truncated multivariate normal variable. Here, we use the distribution of $(\alpha_i, U^B_i, \mu_i + U^B_i + V_i)$ which is normal with mean $(\bar{\alpha} + \theta\bar{\mu}, 0, \bar{\mu})$ and covariance matrix $\mathbf{D}$ with:

$$\mathbf{D} = \begin{pmatrix} \theta^2\sigma_\mu^2 + \sigma_\eta^2 & 0 & (\theta+\gamma\theta)\sigma_\mu^2 + \rho_{\eta,\omega}\sigma_\eta\sigma_\omega \\ 0 & \sigma_U^2 & \sigma_U^2 \\ (\theta+\gamma\theta)\sigma_\mu^2 + \rho_{\eta,\omega}\sigma_\eta\sigma_\omega & \sigma_U^2 & (1+\gamma^2)\sigma_\mu^2 + \sigma_U^2 + \sigma_\omega^2 \end{pmatrix}$$

```
mean.alpha.UB.yBV <- c(param['baralpha']+param['barmu']*param['theta'],0,param['barmu'])
cov.alpha.UB.yBV <- matrix(c(param['theta']^2*param['sigma2mu']+param['sigma2eta'],
                             0,
                             (param['theta']+param['gamma']*param['theta'])*param['sigma2mu']+par
                             0,
                             param['sigma2U'],
                             param['sigma2U'],
                             (param['theta']+param['gamma']*param['theta'])*param['sigma2mu']+par
                             param['sigma2U'],
                            (1+param['gamma']^2)*param['sigma2mu']+param['sigma2U']+param['sigma2

# cuts
#non participants
lower.cut.D0 <- c(-Inf,-Inf,log(param['barY']))
upper.cut.D0 <- c(Inf,Inf,Inf)
# participants
lower.cut.D1 <- c(-Inf,-Inf,-Inf)
upper.cut.D1 <- c(Inf,Inf,log(param['barY']))

# means
TT <- mtmvnorm(mean=mean.alpha.UB.yBV,sigma=cov.alpha.UB.yBV,lower=lower.cut.D1,upper=upper.cut.D
mean.UB.D0 <- mtmvnorm(mean=mean.alpha.UB.yBV,sigma=cov.alpha.UB.yBV,lower=lower.cut.D0,upper=upp
mean.UB.D1 <- mtmvnorm(mean=mean.alpha.UB.yBV,sigma=cov.alpha.UB.yBV,lower=lower.cut.D1,upper=upp
B.DID <- -(1-param['rho'])*(mean.UB.D1-mean.UB.D0)
```

In our example, the population TT is equal to $\Delta^y_{TT} = 0.173$. The DID estimator is equal to $\Delta^y_{DID} = 0.219$. As a consequence, the bias of the DID estimator is equal to $\Delta^y_{B(DID)} = 0.046$.

In order to make the DID estimator consistent for the TT parameter, we need to impose that $\rho = 1$. When shocks are permanent, their bias remains constant over time and thus DID can estimate it without error. Let us generate new data that are compatible with that assumption.

```
set.seed(1234)
N <-1000
param["rho"] <- 1
cov.eta.omega <- matrix(c(param["sigma2eta"],param["rhoetaomega"]*sqrt(param["sigma2eta"]*param['
eta.omega <- as.data.frame(mvrnorm(N,c(0,0),cov.eta.omega))
colnames(eta.omega) <- c('eta','omega')
mu <- rnorm(N,param["barmu"],sqrt(param["sigma2mu"]))
```

```
UB <- rnorm(N,0,sqrt(param["sigma2U"]))
yB <- mu + UB
YB <- exp(yB)
Ds <- rep(0,N)
V <- param["gamma"]*(mu-param["barmu"])+eta.omega$omega
Ds[yB+V<=log(param["barY"])] <- 1
epsilon <- rnorm(N,0,sqrt(param["sigma2epsilon"]))
U0 <- param["rho"]*UB + epsilon
y0 <- mu +  U0 + param["delta"]
alpha <- param["baralpha"]+  param["theta"]*mu + eta.omega$eta
y1 <- y0+alpha
Y0 <- exp(y0)
Y1 <- exp(y1)
y <- y1*Ds+y0*(1-Ds)
Y <- Y1*Ds+Y0*(1-Ds)
```

Let's see how DID works on this data.

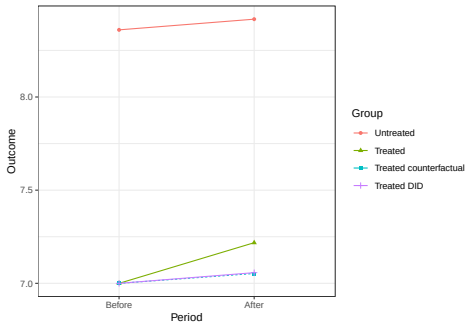


Figure 4.6: Evolution of average outcomes in the treated and control group when the Parallel Trends Assumption holds

Now, the counterfactual change in outcome for the treated and its approximation using the trend experienced by the untreated are extremely close, as the curves *Treated counterfactual* and *Treated DID* show on Figure 4.6.

4.3.1.2 Estimation

Estimation of TT under the DID assumptions can be performed in a variety of ways: using directly the DID formula, using OLS with group fixed effects, using OLS with individual and time dummy variables, using first differences and using the within transformation (also known as the Two-Way Fixed Effects or TWFE estimator). With only two periods of data and a fully balanced panel, all of these estimators are actually numerically equivalent. Let's examine them in turn.

4.3.1.2.1 Using the DID formula One could go directly and use the DID formula of Theorem 4.4. The sample DID estimator is thus equal to:

$$\hat{\Delta}^{Y}_{DID} = \frac{\sum_{i=1}^{N} Y_{i,A} D_i}{\sum_{i=1}^{N} D_i} - \frac{\sum_{i=1}^{N} Y_{i,B} D_i}{\sum_{i=1}^{N} D_i} - \left(\frac{\sum_{i=1}^{N} Y_{i,A}(1-D_i)}{\sum_{i=1}^{N}(1-D_i)} - \frac{\sum_{i=1}^{N} Y_{i,B}(1-D_i)}{\sum_{i=1}^{N}(1-D_i)} \right).$$

Example 4.15. In our example, let's see how this estimator works.

The Before/After comparison among the participants is equal to $\hat{\Delta}^{y}_{BA|D=1} = 0.218$. The Before/After comparison among the non-participants is equal to $\hat{\Delta}^{y}_{BA|D=0} = 0.057$. The DID estimator is thus equal to $\hat{\Delta}^{y}_{DID} = \hat{\Delta}^{y}_{BA|D=1} - \hat{\Delta}^{y}_{BA|D=0} = 0.218 - 0.057 = 0.161$. It is also equal to the difference between the before and after With/Without estimators. The Before With/Without estimator is equal to $\hat{\Delta}^{y^B}_{WW}$ = -1.361. The After With/Without estimator is equal to $\hat{\Delta}^{y}_{WW}$ = -1.2. The DID estimator is thus equal to $\hat{\Delta}^{y}_{DID} = \hat{\Delta}^{y}_{WW} - \hat{\Delta}^{y^B}_{WW}$ = -1.2 −(-1.361) = 0.161. This is not too far from the true effect of the treatment in the sample which is equal to $\hat{\Delta}^{y}_{TT} = 0.165$. In the population, the TT parameter has not changed, since its computation does not involve ρ. We still have $\Delta^{y}_{TT} = 0.165$.

4.3.1.2.2 Using the Least Squares pooling DID estimator The most basic regression-based way to implement DID is to run a linear regression of outcomes on a treatment group dummy, a time dummy and their interaction. The interaction captures the effect of the treatment estimated using DID. The way it works is as follows: estimate the following equation using OLS and use $\hat{\beta}_{OLS}$ as your DID estimate: $\hat{\beta}_{OLS} = \hat{\Delta}^{Y}_{DID}$.

$$Y_i = \alpha + \mu D_i + \delta T_i + \beta D_i T_i + \epsilon_i.$$

D_i is our usual treatment indicator while T_i takes value one when observation i is observed in the *After* and zero otherwise.

Example 4.16. Let's see how this works in our example.

Before estimating the model, we need to build a data frame with all the necessary variables.

```
# building a data frame
data.DID <- as.data.frame(cbind(c(y,yB),c(Ds,Ds),c(rep(1,N),rep(0,N))))
colnames(data.DID) <- c('y','D','T')

# running the OLS regression
reg.DID <- lm(y ~ D + T + D*T,data = data.DID)

# coefficients
```

```
yB.D0.reg <-  coef(reg.DID)[[1]]
WW.before.reg <-  coef(reg.DID)[[2]]
BA.untreated.reg <- coef(reg.DID)[[3]]
DID.est.reg <- coef(reg.DID)[[4]]

# comparisons
yB.D0.sample <- mean(yB[Ds==0])
```

The estimate of $\hat{\beta}_{OLS}$ in our sample is equal to 0.161. It is exactly equal to $\hat{\Delta}^{y}_{DID}$ as estimated just above. What is interesting with the regression-based DID approach is that the other coefficients in the regression have a direct interpretation. For example, the constant α estimates the mean outcome in the untreated group before the treatment. In our case, we have $\hat{\alpha}_O LS = 8.36$. Remember that in our sample, the average outcomes of the untreated before the treatment is equal to $\hat{\mathbb{E}}[y_i^B | D_i = 0] = 8.36$. μ, the coefficient in front of the D_i dummy, estimates the With/Without estimator before the treatment. In our case, we have $\hat{\mu}_{OLS}$ = -1.361. Remember that in our sample, the With/Without estimator before the treatment is equal to $\hat{\Delta}^{y^B}_{WW}$ = -1.361. δ, the coefficient in front of the T_i dummy, estimates the Before/After change in outcomes among the untreated. In our case, we have $\hat{\delta}_{OLS} = 0.057$. Remember that in our sample, the Before/After estimator among the untreated is equal to $\hat{\Delta}^{y}_{BA|D=0} = 0.057$.

Remark. A pretty cool property of the regression-based DID estimator is that is does not require panel data. It works even with repeated cross sections, *i.e.* when observations are drawn from the same population in both periods but are not the same.

4.3.1.2.3 Using First Differences In the presence of panel data, an alternative to the regression-based DID estimator is the first-difference estimator. It simply regresses the change over time in outcomes on the treatment dummy:

$$Y_{i,A} - Y_{i,B} = \alpha^{FD} + \beta^{FD} D_i + \epsilon_i^{FD}.$$

The coefficient β^{FD} estimated by OLS is an estimate of the DID parameter.

Example 4.17. Let's see how this works in our example.

Before running the model, we need to generate first the differenced estimates. One very simple way to do that is simply to take the difference between the before and the after outcome vectors.

```
# building a data frame
data.FD <- as.data.frame(cbind(y-yB,Ds))
colnames(data.FD) <- c('BAy','D')

# running the OLS regression
reg.FD <- lm(BAy ~ D,data = data.FD)
```

```
# coefficients
BA.untreated.FD <- coef(reg.FD)[[1]]
DID.est.FD <- coef(reg.FD)[[2]]
```

The estimate of $\hat{\beta}_{OLS}^{FD}$ in our sample is equal to 0.161. It is exactly equal to $\hat{\Delta}_{DID}^{y}$ as estimated just above. Note also that α^{FD} estimates the Before/After change in outcomes among the untreated. In our case, we have $\hat{\alpha}_{OLS}^{FD} = 0.057$. Remember that in our sample, the Before/After estimator among the untreated is equal to $\hat{\Delta}_{BA|D=0}^{y} = 0.057$.

4.3.1.2.4 Using the Least Squares Dummy Variables estimator

One very computer-intensive way to estimate TT in a DID setting is to use the OLS estimator supplemented with dummies for each observation and for each time period, also called the Least-Squares Dummy Variables estimator. In practice, the estimator is based on the following regression:

$$Y_{i,t} = \sum_{j=1}^{N} \mu_j \mathbb{1}[j = i] + \sum_{l=0}^{1} \delta_l \mathbb{1}[l = t] + \beta^{LSDV} D_{i,t} + \epsilon_{i,t}^{LSDV}.$$

The notation is generally simplified as follows:

$$Y_{i,t} = \mu_i + \delta_t + \beta^{TWFE} D_{i,t} + \epsilon_{i,t}^{TWFE},$$

This last estimator is generally called the Two-Way Fixed Effects estimator, since it has two-sets of so-called fixed effects (individual fixed effects, μ_i, and time fixed effects δ_t). I will write it using this second, more compact, formulation, but I think the first formulation encapsulates better how the Least-Squares Dummy Variables estimator works. In what follows, we will see other ways of estimating the Two-Way Fixed Effects model, but for now, let us focus on the Least-Squares Dummy Variables estimator. The way it works is simply by throwing a bunch of dummy variables in the regression.

Example 4.18. Let's see how the Least Squares Dummy Variable works in our example. For that, we need to generate one dummy variable for each individual i in our sample. This is made simple by the `factor` function in `R`. We are also going to run the model without a constant, so that all the fixed effects are identified.

```
# adding one column to the DID data frame with the individual index for each observation of the s
data.DID$indiv <- as.factor(c(1:N,1:N))
# generating Dit (time varying)
data.DID$Dit <- data.DID$D*data.DID$T
# running the LSDV estimator
reg.LSDV <- lm(y~-1 + Dit + as.factor(T) + indiv,data=data.DID)
# result
DID.est.LSDV <- coef(reg.LSDV)[[1]]
```

The Least-Squares Dummy Variables estimate of TT is equal to: $\hat{\beta}^{LSDV} = 0.161$.

Remark. The term *fixed effect* is specific to the panel data literature in econometrics. It refers to the fact that both μ_i and δ_t are allowed to be correlated with $D_{i,t}$ in this model. This is in contrast to the *random effects model* where μ_i and δ_t are assumed to be independent of the regressors of interest.

4.3.1.2.5 Using the Within estimator You might have noticed that the Least-Squares Dummy Variables estimator took some time to compute on your computer. This is because it requires the inversion of a very large matrix, as large as the number of fixed effects plus one. The size of this computation increases as the number of observation and time periods increases, meaning that this computation might become practically unfeasible in very large datasets. Several tricks have been developed to decrease the computational burden of the estimation of the Two-Way Fixed Effects model. One approach is to use the First Difference estimator. Another approach is the Within estimator. The way the Within estimator works is by taking the difference between each observation and its mean over time or over individuals. More precisely, the Within estimator estimates the following model by OLS:

$$Y_{i,t} - \frac{1}{2}\sum_{t=0}^{1} Y_{i,t} = \delta_t^W + \beta^W (D_{i,t} - \frac{1}{2}\sum_{t=0}^{1} D_{i,t}) + \epsilon_{i,t}^W.$$

The reason why this trick works is because of the shape of the Two-Way Fixed Effects model. Indeed, taking the average of the Two-Way Fixed Effects model over time gives:

$$\frac{1}{2}\sum_{t=0}^{1} Y_{i,t} = \mu_i + \frac{1}{2}\sum_{t=0}^{1} \delta_t + \beta^{TWFE}\frac{1}{2}\sum_{t=0}^{1} D_{i,t} + \frac{1}{2}\sum_{t=0}^{1} \epsilon_{i,t}^{TWFE}.$$

Taking the difference between the Two-Way Fixed Effects model and its time-averaged version gives the Within estimator. The key is that the differencing gets rid of the individual fixed effects parameter μ_i and thus makes it unnecessary to estimate it. The set of parameters to estimate is thus much smaller than in the Least-Squares Dummy Variables estimator.

Example 4.19. Let's see how the Within estimator works in our example. For that, we need to compute the average over time of the outcome and of the treatment for each observation in our dataset. This is made simple by the `summarize` function of the `dplyr` package.

```
# generating the time means of Y and Dit
TimeMeansYDit <- data.DID %>%
                  group_by(indiv) %>%
                  summarize(
                    TimeMeanY = mean(y),
```

```
                    TimeMeanDit = mean(Dit)
                  )
# doubling the observations to be able to take the difference in both periods
TimeMeansYDit <- rbind(TimeMeansYDit,TimeMeansYDit)
# taking the difference in both periods
data.DID$W.y <- data.DID$y-TimeMeansYDit$TimeMeanY
data.DID$W.Dit <- data.DID$Dit-TimeMeansYDit$TimeMeanDit
# running the within estimator
reg.W <- lm(W.y~-1 + W.Dit + as.factor(T),data=data.DID)
# result
DID.est.W <- coef(reg.W)[[1]]
```

The Within estimate of TT is equal to: $\hat{\beta}^W = 0.161$.

The `plm` package directly implements the Within transformation. The same package also estimates the First Difference model and the Least Squares pooling DID estimator. Let's see how this works.

```
# running the within estimator
reg.W.plm <- plm(y ~  Dit + as.factor(T) , data = data.DID, index= c("indiv", "T"), model = "with
# result
DID.est.W.plm <- coef(reg.W.plm)[[1]]

# running the first difference estimator
reg.FD.plm <- plm(y ~  Dit + as.factor(T) , data = data.DID, index= c("indiv", "T"), model = "fd"
# result
DID.est.FD.plm <- coef(reg.FD.plm)[[2]]

# running the OLS pooling DID estimator
reg.OLS.plm <- plm(y ~ as.factor(T) + D + Dit , data = data.DID, index= c("indiv", "T"), model =
# result
DID.est.OLS.plm <- coef(reg.OLS.plm)[[4]]
```

As expected, `plm` gives the following estimates for TT: $\hat{\beta}^W = 0.161$, $\hat{\beta}^{FD} = 0.161$ and $\hat{\beta}^{OLS} = 0.161$.

4.3.1.2.6 Using fast estimators of the Two-Way Fixed Effects model

All the estimators of the TWFE model that we have seen so far have issues. The OLS pooling DID estimator does not account for the panel structure of the data when it exists. It does not alter the precision of the estimator but it makes it mode difficult to account for more dimensions of fixed effects than two. The First Difference estimator, similarly, cannot easily account for more than two sets of fixed effects. The Least Squares Dummy variable is slow because of the very large matrix inversion problem. Therefore, applied econometricians tend to prefer using the Within estimator in practice. The Within estimtor of the Two-Way Fixed Effects model is not without problems as well. As the sample

size grows large, or the number of fixed effects increases, it becomes more and more difficult to compute the within transformation. As a consequence, recent packages have proposed to optimize the computation of the TWFE model using various computational tricks. Let's examine two in turn.

4.3.1.2.6.1 The Alternating Projections method The `lfe` package in `R` implements an alternating projections method to estimate the N-Way Fixed effects model. It is based on an algorithm proposed by Gaure (2013). The basic idea of Gaure (2013) is to repeat centering on the means of the fixed effects (the within operation) in an alternating manner between the various fixed effects dimensions until convergence.

Example 4.20. Let's see how the `lfe` estimator works in our example.

```
# running the within estimator
reg.W.lfe <- felm(y ~  Dit + as.factor(T) | indiv , data = data.DID)
# result
DID.est.W.lfe <- coef(reg.W.lfe)[[1]]
```

As expected, `lfe` gives the following estimate for TT: $\hat{\beta}^{AP} = 0.161$.

4.3.1.2.6.2 The Likelihood Concentration method One problem with the `lfe` package is that it works only for linear models. The `fixest` package in `R` proposes a solution for estimating fixed effects models in non-linear cases as well. The solution is based on the concentrated likelihood as explained in Berge (2018). The intuition is as follows. We first postulate a value for the treatment effect and the coefficient on the time dummies and we estimate each of the individual fixed effects using maximum likelihood. We then use maximum likelihood to find the treatment effect using the values of the fixed effects estimated in the previous step. This seems complicated but the key idea is to separate the estimation of the fixed effects from the estimation of the parameters of interest.

Example 4.21. Let's see how the `fixest` estimator works in our example.

```
# running the within estimator
reg.W.fixest <- feols(y ~  Dit + as.factor(T) | indiv , data = data.DID)
# result
DID.est.W.fixest <- coef(reg.W.fixest)[[1]]
```

As expected, `fixest` gives the following estimate for TT: $\hat{\beta}^{LC} = 0.161$.

4.3.1.2.7 Equivalence between the various DID methods with two time periods The above results suggest that all DID estimators are equivalent when working with two time periods. The following theorem actually states this result rigorously:

Theorem 4.5 (All DID estimators are numerically equivalent with two time periods)**.** *Under Assumptions 4.7, 4.8 and 4.9, in a panel with only two periods*

of data, all the DID estimators are numerically equivalent: $\hat{\beta}^{OLS} = \hat{\beta}^{FD} = \hat{\beta}^{W} = \hat{\beta}^{LSDV} = \hat{\beta}^{AP} = \hat{\beta}^{LC} = \hat{\Delta}^{Y}_{DID}$.

Proof. See Section A.3.1. □

Finally, let's see how our estimator varies across sampling replications. A key difference is whether we have access to panel data or not. Indeed, estimates from a repeated cross section are going to be more noisy since they are going to sample different people in different periods and thus are going to be affected by sampling noise stemming from the fixed effects. This is not going to be the case with panel data, since all the estimators based on the TWFE estimator differentiate out the individual fixed effects.

Do the simulations

4.3.1.3 Inference

Derive the distribution of the estimators with repeated cross sections and panels

4.3.2 Differences In Differences with more than two-time periods but only one treatment date

In real life, we generally have access to several time periods before and after the treatment date. What happens to DID in that case? The first thing is that, with several periods after the treatment date, we now have a time-varying TT: $\Delta^{TT}_{Y_t} = \mathbb{E}[Y^1_{i,t} - Y^0_{i,t} | D_i = 1]$.

Event Study graph and period specific treatment effects

What are the weights for the TWFE estimator? *Should be derived from the staggered case*

4.3.3 Differences In Differences with a staggered design

4.3.4 Difference In Differences with Instrumental Variables

Chapter 5

Observational Methods

Chapter 6

Threats to the validity of Causal Inference

In this final section, I want to discuss more generally about the possible threats to the validity of methods of causaliInference. Most of these threast stem from the fact that, much as particles do when physicists try to measure their position and velocity, human beings react to our experimental devices in sometimes unexpected ways. It is classical to make a disctinction between two threat and one specific set of problems:

1. Threats to internal validity: these are the threats that vitiate the result of the experiment in the sample at hand, even when only looking at the Intention to Treat Effect. They also include threats to the exclusion restriction assumption.
2. Threats to the measurement of precision
3. Threats to external validity: these are the threats that make the extension of the results from one experiment to the same population at another period of to another population dubious.
4. Ethical and political issues

6.1 Threats to internal validity

Threats to internal validity are the problems that might make the results of the experiment not measure the effect of the treatment of interest in the ongoing sample. Let's examine the most important ones in turn.

6.1.1 Survey bias

There is survey bias if the mere fact of having to answer to a survey alters the outcomes of the surveyed individuals. For example, administering a health

survey might make you pay more attention to your health and as a consequence improve it. Survey bias alters the measured impact of the treatment since it alters the behavior of the control group. As a result, the estimated effect of the treatment might be biased.

Remark. Note that survey bias might affect all the estimators presented in this book, including natural experimental and observational estimators. All estimators rely on measuring something and thus might be affected by survey bias.

Actually, RCTs might be able to avoid survey bias whereas other methods generally cannot. Indeed, survey bias generally occurs with repeated sampling: surveying at baseline might trigger a response by individuals, and thus bias the measurement at the end of the experiment. RCTs can avoid this issue by bypassing the baseline survey, or at least collecting baseline information on a subsample of the experimental sample. Other estimators that might be able to avoid this problem are DID, where the repeated cross section estimator eschews survey bias, and RDD and IV, which generally use only cross sectional information. Matching in general requires observations for the same individual over time, so that avoiding possible survey bias is impossible.

Remark. Do we have evidence of the existence and extent of survey bias? A paper by Zwane et al (2010) shows that there is extensive survey bias in 2 out of 5 experiments.

In the first example, the authors show that being surveyed more frequently (every two weeks *vs* every six months) for the extent of diarrhea prevalence and use of chlorine decontamination increases the use of chlorine, decreases diarrhea prevalence and decreases the effect of a spring protection program, to the extent that it becomes null in the frequent survey sample, whereas it is large and positive in the infrequent survey sample. The authors speculate that the frequent surveys act as a reminder for chlorinating water, which is a substitute for well protection.

In a second experiment, the authors randomly run a baseline survey on 80% of the households tha would be later included in a RCT where health insurance would be offered at randomly selected prices. The baseline survey includes questions about health and health insurance, but does not mention the particular product that will be offered later. The authors find a small imprecise increase in insurance take up in the group having undergone the baseline survey (5% $\pm$ 6.8), non significant at usual levels of confidence. They also find no evidence of impact of the baseline survey on the response of households to the price incentive.

In a third experiment, the authors report on the effect of being surveyed with a survey continaing questions on health status and health insurance on the subsequent adoption of health insurance. They find evidence that the baseline survey has increased the adoption of health insurance by 6.7% $\pm$ 6.6, from a mean of 26.4% in the control group. The effect dissipates over time though and becomes much smaller 15 to 18 months after the treatment.

In a fourth experiment, the authors randomly selected 60% of targeted households to be included in a baseline survey including wuestions on household finances and borrowing opportunities.
The sample was then prospected by a micro-credit firm. The aurhors do not find higher micro-credit take-up among households surveyed at baseline (-0.009 $\pm$ 0.048, with a baseline take up rate of 0.166). Note that the estimates are imprecise though.

In a fifth experiment, the authors randomly assigned the order in which households had to be contacted for a baseline survey, along with a fixed number of households to be sruveyed by village. The survey contained questions about finance and microfinancne use. Also, the survey explicitely mentioned that households were interviewed because they were patrons of a micro-finance provider. Sunbsequently, households had to decide whether or not to renew their loans from the micro-finance provider. The authors do not evidence of an effect of being surveyed on the subsequent decision to renew the loan (-0.005 $\pm$ 0.026 from a baseline rate of 0.769).

6.1.2 Experimenter bias

6.1.3 Substitution bias

6.1.4 Diffusion bias

6.2 Threats to the measurement of precision

6.2.1 Insufficient precision

6.2.2 Clustering

6.3 Threats to external validity

6.3.1 Randomization bias

6.3.2 Equilibrium effects

6.3.3 Context effects

6.3.4 Site selection bias

6.3.5 Publication bias

6.3.6 Ethical and political issues

Part III

Additional Topics

Chapter 7

Power Analysis

Chapter 8

Placebo Tests

Chapter 9

Clustering

Chapter 10

LaLonde Tests

Chapter 11

Diffusion effects

Chapter 12

Distributional effects

Chapter 13

Meta-analysis and Publication Bias

When several research teams work on a similar topic, they obtain and publish several estimates for the same program of for similar programs. For example, teams of doctors regularly test the same treatment on different samples or populations in order to refine the estimated effect. Similarly, economists report on the effects of similar types of programs (Conditional and Unconditional Cash Transfers, Job Training Programs, microcredit, etc) implemented in different countries.

Meta-analysis aims at summarizing and synthetizing the available evidence with two main goals in mind:

1. Increasing precision by providing an average estimated effect combining several estimates
2. Explaining variations in treatment effectiveness by relating changes in effect size to changes in sample characteristics.

One key issue that meta-analysis has to face – actually, we all have to face it, meta-analysis simply makes it more apparent – is that of publication bias. Publication bias is due to the fact that referees and editors have a marked preference for publishing statistically significant results. The problem with this approach is that the distribution of published results is going to be censored on the left: we will have more statistically significant results in the published record, and as a consequence, the average published result will be an upward biased estimate of the true treatment effect in the population. This is potentially a very severe problem if the amount of censoring due to publication bias is large. Eventually, this hinges on the true distribution of treatment effects: if it is centered on zero or close to zero, we run the risk of having very large publication bias.

In this chapter, I present first the tools for meta-analysis, and I then move on to testing and correcting for publication bias. Most of the material presented here stems from the reference book by Hedges and Olkin. When needed, I update this book with new references that I then cite. the R code comes mainly from a wonderful set of slides explaining of the `metafor` package works.

13.1 Meta-analysis

There are several approaches and refinements to meta-analysis. In this section, I am going to present only the most important ones. I'll defer the reader to other more specialized publications if needed.

I first present the basics of meta-analysis: the constitution and structure of the sample. Second, I present the problems of the intuitive "vote-counting" method. Third, I present the methods used when treatment effects are homogeneous across studies, called fixed effects models. Fourth, I move to the methods used when effects are heterogeneous across studies, or random effects models, and the tests used to decide whether we are in a fixed or random effects framework. Fifth, I present meta-regression, that tries to capture treatment effect heterogeneity by including covariates. Finally, I present constantly updated meta-analysis, a way to aggregate results of individual studies as they come.

13.1.1 Basic setting

The basic setting for a meta-analysis is that you have access to a list of estimates for the effect of a given program and for their precision. These estimates come from the literature, searching published and unpublished sources alike. This data is usually collected after an extensive search of bibliographic databases. Then, one has to select among all the studies selected by the search the ones that are actualy relevant. This is the most excruciating part of a meta-analysis, since a lot of the studies selected by hte search algorithm are actually irrelevant. Finally, one has to extract from each relevant paper an estimate of the effect of the treatment and of its precision. In general, one tries to choose standardized estimates such as the effect size (see Section 2.1.6 for a definition) and its standard error. After all this process, we should end up with a dataset like: $\left\{(\hat{\theta}_k, \hat{\sigma}_k)\right\}_{k=1}^{N}$, with $\hat{\theta}_k$ the estimated effect size, $\hat{\sigma}_k$ its estimated standard error, and N the number of included studies.

Example 13.1. Let's see how such a dataset would look like? Let's build one from our simulations.

```
N.sample <- c(100,1000,10000,100000)
N.plot.ES.CLT <- c(10,7,2,1)
data.meta <- data.frame(ES=numeric(),
                        se=numeric())
```

```
se.ww.CLT.ES <- function(N,v1,v0,p){
  return(sqrt((v1/p+v0/(1-p))/N)/v0)
}

for (k in 1:length(N.sample)){
  set.seed(1234)
  simuls.ww[[k]]$se.ES <- se.ww.CLT.ES(N.sample[[k]],simuls.ww[[k]][,'V1'],simuls.ww[[k]][,'V0'],
  test.ES <- simuls.ww[[k]][sample(N.plot.ES.CLT[[k]]),c('ES','se.ES')]
  test.ES$N <- rep(N.sample[[k]],N.plot.ES.CLT[[k]])
  data.meta <- rbind(data.meta,test.ES)
  }

data.meta$id <- 1:nrow(data.meta)
#data.meta$N <- factor(data.meta$N,levels(N.sample))

  ggplot(data.meta, aes(x=as.factor(id), y=ES)) +
      geom_bar(position=position_dodge(), stat="identity", colour='black') +
      geom_errorbar(aes(ymin=ES-qnorm((delta.2+1)/2)*se.ES, ymax=ES+qnorm((delta.2+1)/2)*se.ES),
      geom_hline(aes(yintercept=ES(param)), colour="#990000", linetype="dashed")+
      xlab("Studies")+
      ylab("Effect size")+
      theme_bw()
```

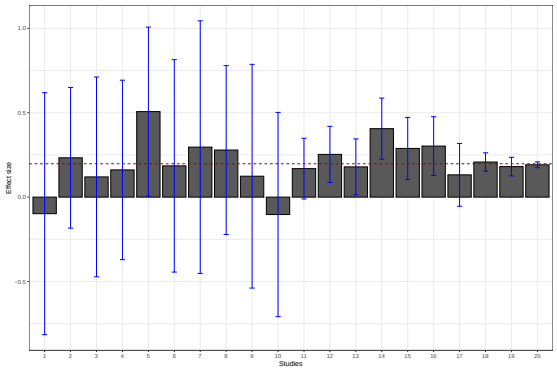


Figure 13.1: Example data set: effect sizes and confidence intervals with $\delta = 0.95$

Figure 13.1 shows the resulting sample. I've selected 10 studies with $N = 100$, 7 studies with $N = 1000$, 2 studies with $N = 10^4$, and 1 study with $N = 10^5$. The studies are represented in that order, mimicking the increasing sample size of studies that accumulate evidence on a treatment, probably with studies with a small sample size at first, and only large studies at the end for the most promising treatments.

13.1.2 Why vote-counting does not work

Vote-counting is an alternative to weighted average or meta-regression. The term, coined by Light and Smith (1971), refers to the practice of counting the number of studies that fall under one of three categories:

- Significant and positive,
- Insignificant,
- Significant and negative.

A vote-counting approach concludes that there is evidence in favor of the treatment when the majority of effects fall in the first category, that there is no evidence that the treatment has an impact whenthe majority of studies fall in the second category, and that there is evidence that the treatment is defavorable when the majority of studies fall in the third category. In general, majority is evaluated at 33%.

The main problem with the vote counting approach is that it does not give more weight to more precise studies. As a consequence, there is a very realistic possibility that the probability of finding the truth decrease as we add more studies to the meta-analysis.

Let's see how this could happen with a simulation taken from HEdges and Olkin's book. Let p be the probaility that a given result is significant and positive. p depends on the sample size n of the study, and on the true treatment effect, θ:

$$p = \int_{C_\alpha}^{\infty} f(t; \theta, n),$$

where f is the density of the test statistic T used to evaluate whether the effect is significant or not, and C_α is the critical value of the test T. If n and θ are constant over studies (for simplicity), the process of accumulating significant results can be modelled as a binomial with parameter p. The probability that over K studies, we have a proportion of significant results larger than a pre-specified threshold (let's say C_0) is equal to:

$$\Pr(\frac{X}{K} > C_0) = \sum_{k=\text{int}(C_0 k)+1}^{K} \left(\begin{array}{c} K \\ k \end{array} \right) p^k (1-p)^{K-k},$$

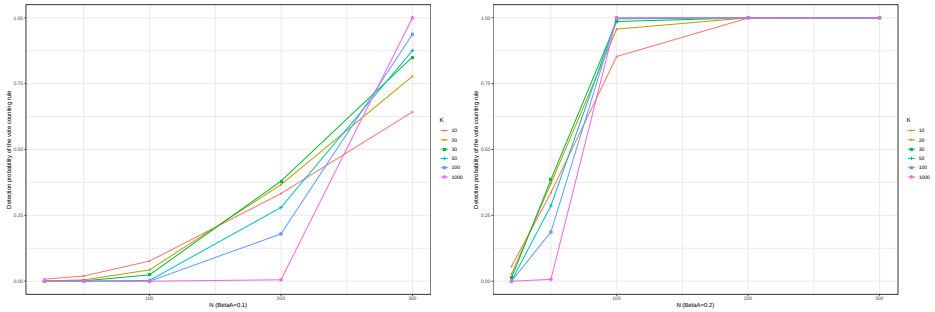
where $\text{int}(a)$ is the greatest integer larger or equal to a and $0 \leq C_0 \leq 1$. In order to use this formula, we simply have to choose a test. Let's choose the two-sided t-test of a zero treatment effect in an RCT with equal tozes for treated and control groups. In that case, p is simply the power of the test. In Chapter 7, we have derived a formula for the power of this test when N is large:

$$\kappa = \Phi\left(\frac{\beta_A}{\sqrt{\mathbb{V}[\hat{E}]}} - \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\right),$$

with $\mathbb{V}[\hat{E}] = \frac{C(\hat{E})}{N}$ and $C(\hat{E})$ the variance of the estimator across sampling replications. Let's make the simplifying assumption that the treatment effect is constant, so that the variance of the estimator is basically the variance of the outcomes. Let's also assume that we are working with effect sizes, so that our outcomes are normalized to have mean zero and variance one. Under these assumptions, $C(\hat{E}) = 1$ and we can implement the power formula:

```
PowerTwoside <- function(betaA,alpha,N,CE=1){
  return(pnorm(-betaA/sqrt(CE/N)-qnorm(1-alpha/2))+pnorm(betaA/sqrt(CE/N)-qnorm(1-alpha/2)))
}

PowerTwosideStudent <- function(betaA,alpha,N,CE=1){
  return(pt(-betaA/sqrt(CE/N)-qnorm(1-alpha/2),df=N-1)+pt(betaA/sqrt(CE/N)-qnorm(1-alpha/2),df=N-
}

VoteCounting <- function(betaA,C0,K,...){
  return(pbinom(q=C0*K,size=K,prob=PowerTwosideStudent(betaA=betaA,...),lower.tail = FALSE))
}

PowerTwosideStudent(betaA=0.1,alpha=0.05,N=300)
VoteCounting(C0=.33,K=3000,betaA=0.1,alpha=0.05,N=300)

Sample.size <- c(20,50,100,200,300)
BetaA <- seq(0.1,1.5,by=0.1)
K.list <- c(10,20,30,50,100,1000)

power.vote <- data.frame("Power"= 0,'BetaA'= 0,'N'= 0,'K'= 0)

#power.vote <- sapply(BetaA,VoteCounting,C0=.33,K=K.list[[1]],alpha=0.05,N=Sample.size[[1]])
#power.vote <- cbind(power.vote,BetaA,Sample.size[[1]],K.list[[1]])

for (j in (1:length(K.list))){
  for (k in (1:length(Sample.size))){
    power.vote.int <- sapply(BetaA,VoteCounting,C0=.33,K=K.list[[j]],alpha=0.05,N=Sample.size[[k]
    power.vote.int <- cbind(power.vote.int,BetaA,Sample.size[[k]],K.list[[j]])
    colnames(power.vote.int) <- c('Power','BetaA','N','K')
    power.vote <- rbind(power.vote,power.vote.int)
  }
}
```

```
power.vote <- power.vote[-1,]
power.vote$K.int <- power.vote$K
power.vote$K <- as.factor(power.vote$K)

#ggplot(data=filter(power.vote,K==10),aes(x=N,y=Power,group=as.factor(BetaA),shape=as.
#  geom_line()+
#  geom_point()

ggplot(data=filter(power.vote,BetaA==0.1),aes(x=N,y=Power,group=K,shape=K,color=K))+
  geom_line()+
  geom_point()+
  xlab("N (BetaA=0.1)")+
  ylab("Detection probability of the vote counting rule")+
  theme_bw() +
  scale_fill_discrete(name="K")


ggplot(data=filter(power.vote,BetaA==0.2),aes(x=N,y=Power,group=K,shape=K,color=K))+
  geom_line()+
  geom_point()+
  xlab("N (BetaA=0.2)")+
  ylab("Detection probability of the vote counting rule")+
  theme_bw()
```



(a) $\beta_A = 0.1$ (b) $\beta_A = 0.2$

Figure 13.2: Detection probability of the vote counting rule

Figure 13.2 shows that the vote counting rule has a very inconvenient property: when the power of the test is lower than 33%, the probability that the vote counting rule detects a true effect decreases with the number of studies included in the meta-analysis, and converges to zero when the number of studies gets large.

For example, when $N = 100$ and $\beta_A = 0.1$, the probability of detecting the effect using the vote counting method is equal to 0.076 with $K = 10$ studies

and decreases to 0.043 when $K = 20$, and 0 when $K = 100$. The pattern is reverse for more powerful studies, such as when $N = 300$ and $\beta_A = 0.1$ or when $N = 100$ and $\beta_A = 0.2$. The intuition for this result is that the vote counting method does not average out the sampling noise in each individual study.

13.1.3 Meta-analysis when treatment effects are homogeneous: the fixed effects approach

The key idea of meta-analysis with fixed effects is to combine the effect size estimates stemming from different studies, weighing them by their relative precision.

Definition 13.1 (Weighted Meta-Analytic Estimator)**.** The weighted meta-analytic estimator is

$$\bar{\theta} = \sum_{k=1}^{N} w_k \hat{\theta}_k \text{ with } w_k = \frac{\frac{1}{\hat{\sigma}_k^2}}{\sum_{k=1}^{N} \frac{1}{\hat{\sigma}_k^2}}.$$

Under some assumptions, the estimator $\bar{\theta}$ converges to the true effect of the treatment. Let's delineate these assumptions.

Definition 13.2 (Homogeneous Treatment Effect)**.** Each $\hat{\theta}_k$ converges to the same treatment effect θ.

Assumption 13.2 imposes that all the studies have been drawn from the same population, where the treatment effect is a constant.

Definition 13.3 (Independence of Estimates)**.** The $\hat{\theta}_k$ are independent from each other.

Assumption 13.3 imposes that all the studies estimates are independent from each other. That means that they do not share sampling units and that they are not affected by common shocks.

Under these assumptions, we can show two important results.

Theorem 13.1 (Consistency of the Weighted Meta-Analytic Estimator)**.** *Under Assumptions 13.2 and 13.3, when the sample size of each study goes to infinity,* $\bar{\theta} \approx \theta$.

Proof. The Law of Large Number applied to each sample gives the fact that the estimator is a weighted sum of θ with weights summing to one. Hence the result. □

Theorem 13.1 says that the error we are making around the true effect of the treatment goes to zero as the sample size in each study decrease. This is great: aggregating the studies is thus going to get us to the truth.

Remark. One interesting question is whether Theorem 13.1 also holds when the size of the individual studies remains fixed and the number of studies goes to infinity, which seems a more natural way to do asymptotics in a meta-analysis. I'm pretty sure that is the case. Indeed, the studies constitute an enormous sample in which we take the average outcomes of the treated on the one hand and of the untreated on the other. These averages differ from the usual ones in the Law of Large Numbers only by the fact that the weights are not equal to one. But they (i) are independent from the outcomes and (ii) sum to one. As a consequence, I'm pretty sure the Law of Large Numbers also apply in this dimension.

Check if this is a consequence of Kolmogorov's Law of Large Numbers.

Theorem 13.2 (Asymptotic Distribution of the Weighted Meta-Analytic Estimator)**.** *Under Assumptions 13.2 and 13.3, when the sample size of each study goes to infinity,* $\bar{\theta} \xrightarrow{d} \mathcal{N}(\theta, \sigma^2)$*, with*

$$\sigma^2 = \frac{1}{\sum_{k=1}^{N} \frac{1}{\sigma_k^2}}.$$

Proof. **To do using the Lindenberg-Levy version of the Central Limit Theorem.** □

Theorem 13.2 shows that the distribution of the weighted meta-analytic estimator converges to a normal, which is very convenient in order to compute sampling noise. In order to obtain an estimator $\hat{\sigma}^2$ of the variance of the meta-analytic estimator, we can simply replace the individual variance terms by their estimates: $\hat{\sigma}_k^2$.

Remark. I've taken Theorem 13.2 from Hedges and Olkin, but I think it is much more interesting and correct when the asymptotics goes in the number of studies.

Remark. According to Hedges and Olkin, the weighted meta-analytic estimator is the most efficient estimator available.

```
wmae <- function(theta,sigma2){
  return(c(weighted.mean(theta,(1/sigma2)/(sum(1/sigma2))),1/sum(1/sigma2)))
}
```

Example 13.2. Let's use our meta-analytic estimator to estimate the effect size of our treatment.

The estimated treatment effect size with our sample is 0.19 ± 0.02. A very simple way to implement such an estimator in R is to use the `rma` command of the `metafor` package.

```
data.meta$var.ES <- data.meta$se.ES^2
meta.example.FE <- rma(yi = data.meta$ES,vi=data.meta$var.ES,method="FE")
summary(meta.example.FE)
```

```
##
## Fixed-Effects Model (k = 20)
##
##   logLik  deviance       AIC       BIC      AICc
##  16.1375   12.7060  -30.2751  -29.2793  -30.0529
##
## I^2 (total heterogeneity / total variability):   0.00%
## H^2 (total variability / sampling variability):  0.67
##
## Test for Heterogeneity:
## Q(df = 19) = 12.7060, p-val = 0.8533
##
## Model Results:
##
## estimate      se     zval    pval   ci.lb   ci.ub
##   0.1950  0.0079  24.6975  <.0001  0.1795  0.2104  ***
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As seen above, the `metafor` package yields a meta-analytic estimate of 0.19 ± 0.02, as we have found using the weighted meta-analytic estimator.

It is customary to present the results of a meta-analysis using a forest plot. A forest plows all the individual estimates along with the aggregated estimate. Figure 13.3 presents the forest plot for our example using the very convenient `forest` function in the `metafor` package:

```
forest(meta.example.FE,slab = paste('Study',data.meta$id,sep=' '),xlab='Estimated Meta-analytic P
```

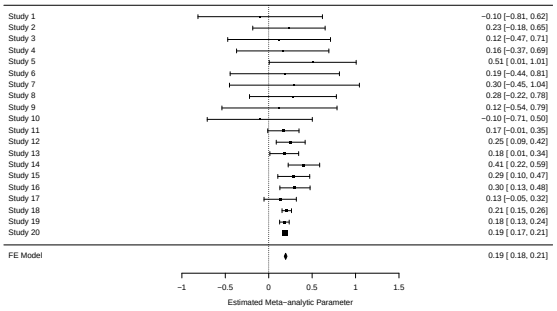


Figure 13.3: Example data set: forest plot

13.1.4 Meta-analysis when treatment effects are heterogeneous: the random effects approach

One key assumption that we have just made so far is that of homogeneous treatment effect. We have worked under the assumption that each study was drawn from the same population, where the treatment effect is a constant. Why would the treatment effects differ in each study?

1. We do not study exactly the same treatment, but a family of similar treatments. Each individual study covers a particular iteration of the treatment, each with its idiosyncratic parameterization. The particular value of the transfer in a Cash Transfer program, or of the conditions to receive it, or the length of payment, whether it is in one time or over some period, might make a difference, for example. The same is true for Job Training Programs, Payments for Environmental Services, microcredit, graduation programs, nudges, etc. Actually, most programs that economists study differ from one implementation to the next. In psychology and medecine, most treatments are accompanied by a rigorous protocol that makes them much more homogeneous.
2. The population on which the treatment is applied varies. For example, similar Job Training Programs or microcredit initiatives might have very different outcomes depending on the business cycle. Education interventions might have very different effects depending on the background of the students on which they are tested. A drug might interact with patients' phenotype and genotype to generate different effects, and the populations from which the experimental samples are drawn do not have to be similar. As an extreme example, think of a vaccine tested in a population where the prevalence of a disease is null. The treatment effect is zero. Now, test the vaccine in a population where the disease is endemic: the treatment effect might be huge.

When each study draws a treatment effect from a distinct population, meta-analysis has to take into account that treatment effects are heterogeneous. The main consequence of treatment effect heterogeneity is that the weighting approach we have used so far underestimates the uncertainty around the true effect, since it does not acknowledge that there is additional variation within each study.

There are two main ways to account for heterogeneity in meta-analysis:

1. **Random effects** allowing for additional random noise in each study.
2. **Meta-regression** trying to capture the heterogeneity in treatment effects with observed covariates.

In this section, we study the random effects estimator, and the next section will cover the meta-regression estimator. Before implementing the random effects estimator, we need to decide whether there is heterogeneity in treatment effects or not.

Generate noise right now and show the plot.

13.1.4.1 Estimating the heterogeneity of treatment effects

A necessary first step is to estimate the variance in treatment effects that is due to treatment effect heterogeneity, beyond sampling noise. The observed effect size estimate for a given study k is modelled as follows:

$$\hat{\theta}_k = \alpha + \epsilon_k + \nu_k,$$

where ϵ_k is due to sampling noise and ν_k is due to the heterogeneity in effect sizes across sites, while α is the average of the effect size accross all populations. We denote the variance of ν_k as τ^2. ν_k is the random effect that gives the random effects approach its name.

There are several ways to estimate this variation. I'm gooing to start with the most intuitive one, Hedges' estimator, and I'll then move on to the other ones available. I'll conclude with the formal statistical tests used to decide whether treatment effects are heterogeneous or not.

13.1.4.1.1 Hedges' estimator of treatment effect heterogeneity Since Hedges, τ^2 is estimated as the residual variance in effect sizes that is not explained by sampling noise. In order to compute this estimator, first estimate the overall variance in $\hat{\theta}_k$, then estimate the component of the variance due to sampling noise and finally take the difference between the two. Hedges' estimator of the overall variance in effect sizes is:

$$\hat{\tau}^2 = \hat{\sigma}^2_{tot} - \hat{\sigma}^2_{\epsilon},$$

with

$$\hat{\sigma^2_{tot}} = \frac{1}{N}\sum_{k=1}^{N}(\hat{\theta}_k - \bar{\theta}_u)^2$$
$$\bar{\theta}_u = \frac{1}{N}\sum_{k=1}^{N}\hat{\theta}_k$$
$$\hat{\sigma^2_{\epsilon}} = \frac{1}{N}\sum_{k=1}^{N}\hat{\sigma}^2_k.$$

Remark. Hedges actually uses the unbiased estimator adapted to small samples and thus replaces N by $N-1$ in the first equation.

Example 13.3. Let's compute Hedges' esimator for τ^2 in our numerical example.

Let's first define a few functions to compute each part:

```
tau.2 <- function(theta,vartheta){
  return(var(theta)-mean(vartheta))
}
tau.2.theta <- tau.2(data.meta$ES,data.meta$se.ES^2)
```

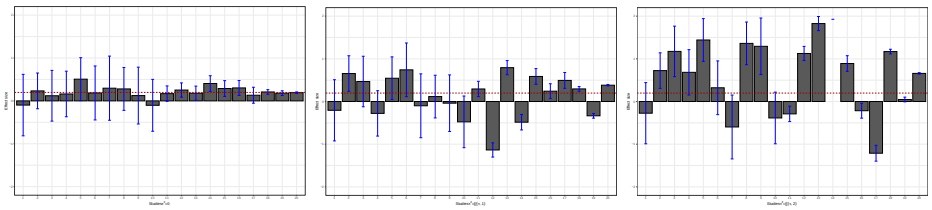
Our estimate of τ^2 in our example is thus -0.03. This estimate is small, suggesting that there is no additional variance in the treatment effects on top of sammling variation, as we know is the case and has already been suggested by the results of the Q statistic. Let's now create a new sample of effect sizes where we add noise to each estimate stemming not from sampling, but from heterogeneity in treatment effects across sites and studies.

```
tau <- c(0.5,1)
set.seed(1234)
data.meta$theta.1 <- data.meta$ES + rnorm(nrow(data.meta),mean=0,sd=tau[[1]])
data.meta$theta.2 <- data.meta$ES + rnorm(nrow(data.meta),mean=0,sd=tau[[2]])
```

I've simulated two new vectors of estimates for θ, both obtained adding a mean-zero normally distributed noise to the initial estimates of θ, one with a standard deviation of 0.5 and the other of 1. Let's visualize our two new datasets:

```
  ggplot(data.meta, aes(x=as.factor(id), y=ES)) +
      geom_bar(position=position_dodge(), stat="identity", colour='black') +
      geom_errorbar(aes(ymin=ES-qnorm((delta.2+1)/2)*se.ES, ymax=ES+qnorm((delta.2+1)/
      geom_hline(aes(yintercept=ES(param)), colour="#990000", linetype="dashed")+
      xlab(expression(paste('Studies',tau^2,'=',0,sep=' ')))+
      ylab("Effect size")+
      theme_bw()+
      ylim(-2,2)

  ggplot(data.meta, aes(x=as.factor(id), y=theta.1)) +
      geom_bar(position=position_dodge(), stat="identity", colour='black') +
      geom_errorbar(aes(ymin=theta.1-qnorm((delta.2+1)/2)*se.ES, ymax=theta.1+qnorm((de
      geom_hline(aes(yintercept=ES(param)), colour="#990000", linetype="dashed")+
      xlab(expression(paste('Studies',tau^2,'=',tau[[1]],sep=' ')))+
      ylab("Effect size")+
      theme_bw()+
      ylim(-2,2)

  ggplot(data.meta, aes(x=as.factor(id), y=theta.2)) +
      geom_bar(position=position_dodge(), stat="identity", colour='black') +
      geom_errorbar(aes(ymin=theta.2-qnorm((delta.2+1)/2)*se.ES, ymax=theta.2+qnorm((de
      geom_hline(aes(yintercept=ES(param)), colour="#990000", linetype="dashed")+
      xlab(expression(paste('Studies',tau^2,'=',tau[[2]],sep=' ')))+
      ylab("Effect size")+
      theme_bw()+
```

```
      ylim(-2,2)
```



(a) initial (b) $\tau^2 = 0.5$ (c) $\tau^2 = 1$

Figure 13.4: Datasets with treatment effect heterogeneity

Let's see now how Hedge's estimator performs:

```
tau.2.theta.1 <- tau.2(data.meta$theta.1,data.meta$se.ES^2)
tau.2.theta.2 <- tau.2(data.meta$theta.2,data.meta$se.ES^2)
```

Hedges' estimates of τ^2 in our examples are thus 0.2 and 0.73 respectively, while the true values are, respectively 0.25 and 1.

13.1.4.1.2 Other estimators of treatment effects heterogeneity τ^2 is a pretty difficult measure of treatment effect heterogeneity to interpret. That's why other indicators have been built that are easier to interpret. We are going to review several of them in this section.

The first alternative or complement to τ^2 is Higgin's I^2:

$$I^2 = \frac{Q - (N-1)}{Q} * 100$$

The interpretation of I^2 is pretty straightforward: it is the distance between the actual value of the Q statistic and its value under the null of treatment effect homogeneity (it is equal to the number of studies N, with a correction for degress of freedom). It can also be interpreted as the fraction of the overall variance (remember that Q is the sum of variance ratios) that is not explained by within study sampling noise.

Another complement to τ^2 is H^2:

$$H^2 = \frac{Q}{N-1}$$

If H^2 is above one, then there is unexplained heterogeneity, again by the fact that Q has mean $N-1$ under the null of treatment effect homogeneity.

Finally, we can also define the Intra Class Correlation (ICC), which precisely measures the share of total variance attributable to treatment effect heterogeneity:

$$ICC = \frac{\tau^2}{\tau^2 + S^2}$$

Where S^2 is the amount of variance due to sampling noise. An estimator for S^2 is:

$$S^2 = \frac{(N-1)\sum_{k=1}^{N} \frac{1}{\sigma_k^2}}{(\sum_{k=1}^{N} \frac{1}{\sigma_k^2})^2 - \sum_{k=1}^{N} (\frac{1}{\sigma_k^2})^2}.$$

I do not understand the formula for S^2. Why does it estimate what we want? I'd take the average variance.

ICC and I^2 are related by the following very simple relation: $I^2 = ICC * 100$.

Example 13.4. Let's see how these three estimators look like in our example. The cool thing is that `rma` computes these estimators by default, so that a simple call to `summary()` is going to show them. The default random effects estimator is `REML`, which is deemed to be the best of them all according to simulations (Viechtbauer, 2002).

```
meta.example.RE.ES <- rma(yi = data.meta$ES,vi=data.meta$var.ES)
meta.example.RE.theta.1 <- rma(yi = data.meta$theta.1,vi=data.meta$var.ES)
meta.example.RE.theta.2 <- rma(yi = data.meta$theta.2,vi=data.meta$var.ES)

tau2.hat <- c(meta.example.RE.ES$tau2,meta.example.RE.theta.1$tau2,meta.example.RE.thet
I2 <-  c(meta.example.RE.theta.1$I2,meta.example.RE.theta.2$I2,meta.example.RE.ES$I2)
H2 <-  c(meta.example.RE.theta.1$H2,meta.example.RE.theta.2$H2,meta.example.RE.ES$H2)

# illustration of results returned by summary
summary(meta.example.RE.theta.2)
```

```
##
## Random-Effects Model (k = 20; tau^2 estimator: REML)
##
##   logLik  deviance       AIC       BIC      AICc
## -24.7208   49.4417   53.4417   55.3305   54.1917
##
## tau^2 (estimated amount of total heterogeneity): 0.7507 (SE = 0.2583)
## tau (square root of estimated tau^2 value):      0.8664
## I^2 (total heterogeneity / total variability):   99.59%
## H^2 (total variability / sampling variability):  241.82
```

```
##
## Test for Heterogeneity:
## Q(df = 19) = 1927.7020, p-val < .0001
##
## Model Results:
##
## estimate      se    zval    pval   ci.lb   ci.ub
##   0.6015  0.1997  3.0127  0.0026  0.2102  0.9929  **
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The estimate of I^2 in our example is of 0 when τ^2 is equal to 0, of 98.71 when τ^2 is equal to 0.25 and of 99.59 when τ^2 is equal to 1. The estimate of H^2 in our example is of 1 when τ^2 is equal to 0, of 77.4 when τ^2 is equal to 0.25 and of 241.82 when τ^2 is equal to 1.

13.1.4.1.3 Testing for the homogeneity of treatment effects What can we do in order to test whether there is heterogeneity in treatment effects? One way is to build an index comparing the usual variation in treatment effects stemming from sampling noise to the one stemming from variation between studies. If we find that the variation between studies dwarves the variation due to sampling noise in each study, then there is some heterogeneity for sure. One statistics that does that is the Q statistic where the variation in treatment effects between studies is estimated using the difference between the individual effect size and the average one squared:

$$Q = \sum_{k=1}^{N} \frac{(\hat{\theta}_k - \bar{\theta})^2}{\hat{\sigma}_k^2}.$$

What is great with the Q statistic is that, under the Null hypothesis that all the treatment effects are equal to the same constant, it is distributed asymptotically as a χ^2 distribution with $N-1$ degrees of freedom, and thus it can directly be used to test for the hypothesis of homogeneous treatment effects.

Example 13.5. In our example, we have already computed the Q statistic when we have used the `rma` function in the `metafor` package. In order to access it, we just need to extract it using `meta.example.FE$QE` for the Q statistic and `meta.example.FE$QEp` for its p-value.

The Q statistic in our example has value 12.71, with associated p-value 0.85. We end up not rejecting homogeneity, which is correct.

Remark. The problem with using test statistics for testing for treatment effect homogeneity is that, when precision increases, we might end up rejecting

homogeneity despite the fact that it is there.

Test with $N = 10^5$.

Remark. The χ^2 distribution with k degrees of freedom is asymptotically distributed as a normal with mean k and variance $2k$. So, when k is large, a good rule of thumb for assessing the homogeneity of the treatment effect estimates is to compare the Q statistic to the number of studies. If it is much larger, homogeneity is probably not guaranteed.

13.1.4.2 Random effects models

Hedges proposes a new estimator for the average effect of the treatment, an estimator that accounts for the additional noise due to heterogeneous treatment effects accross sites.

Definition 13.4 (Hedges Weighted Meta-Analytic Estimator)**.** Hedges weighted meta-analytic estimator for in the presence of random effects is

$$\bar{\theta}_H = \sum_{k=1}^{N} v_k \hat{\theta}_k \text{ with } v_k = \frac{\frac{1}{\hat{\sigma}_k^2 + \hat{\tau}^2}}{\sum_{k=1}^{N} \frac{1}{\hat{\sigma}_k^2 + \hat{\tau}^2}}.$$

```
Hwmae <- function(theta,sigma2,tau2){
  return(c(weighted.mean(theta,(1/sigma2)/(sum(1/(sigma2+tau2))),1/sum(1/sigma2+tau2))
}
ES.H.theta.1 <- Hwmae(data.meta$theta.1,data.meta$se.ES^2,tau.2.theta.1)
ES.H.theta.2 <- Hwmae(data.meta$theta.2,data.meta$se.ES^2,tau.2.theta.2)
```

Example 13.6. Let's see how Hedges estimator performs in our example.

Hedges' estimates of the average effect size is equal to 0.3 and 0.65 respectively, while the true value is 0.2. The main problem with Hedges' estimator when treatment effects are heterogeneous is that very large effects for the more precise estimators dramatically affect the estimate.

Remark. Hedges' estimate of τ^2 is slightly negative, which is problem, since a variance is always positive. Other estimators of τ^2 have been proposed in the literature to account for this fact and to respond to various shortcomings of Hedges' approach. We will present them succinctly since they are part of the `metafor` package. These other estimators have bames such as . They are very well described in this amazing set of slides. Besides Hedges' (denoted 'HE' in R), the other estimators are named:

- DerSimonian-Laird ('DL')
- Hunter-Schmidt ('HS')
- Sidik-Jonkman ('SJ')
- Maximum-likelihood ('ML')
- Restricted maximum-likelihood ('REML')
- Empirical Bayes ('EB')

I'll detail how they work later.

Detail other estimators of tau.

Example 13.7. For the moment, let's see how they perform in our numerical example.

```
estimators <- c("DL", "REML", "HE", "HS", "SJ", "ML", "EB")
meta.example.RE.theta.1.tau2 <- sapply(estimators,function(method){return(rma(yi = data.meta$thet
meta.example.RE.theta.2.tau2 <- sapply(estimators,function(method){return(rma(yi = data.meta$thet
#meta.example.RE <- sapply(estimators,function(method){return(rma(yi = data.meta$theta.1,vi=data.
#meta.example.RE.tau2.test <- unlist(lapply(meta.example.RE,'[[','tau2'))

result.RE <- data.frame(Method=rep(estimators,2),tau2hat=c(meta.example.RE.theta.1.tau2,meta.exam

ggplot(data=result.RE, aes(x=Method, y=tau2hat, fill=as.factor(tau2))) +
    geom_bar(stat="identity", position=position_dodge())+
    ylim(0,1)
```

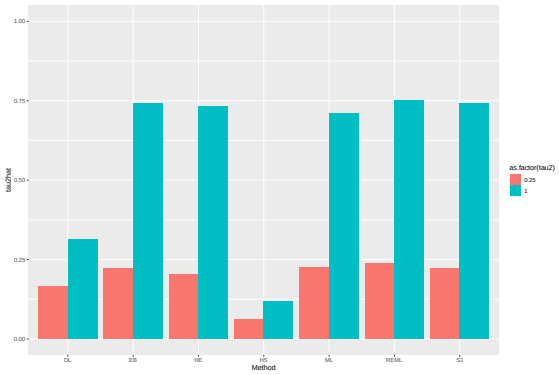


Figure 13.5: Various estimators of τ^2

We are ready to estimate the overall treatment effect using random effects.

```
estimators <- c("DL", "REML", "HE", "HS", "SJ", "ML", "EB")
meta.example.RE.theta.1.ES <- sapply(estimators,function(method){return(rma(yi = data.meta$theta.
meta.example.RE.theta.2.ES <- sapply(estimators,function(method){return(rma(yi = data.meta$theta.
#meta.example.RE.tau2.test <- unlist(lapply(meta.example.RE,'[[','tau2'))

result.RE$ES.RE <- c(meta.example.RE.theta.1.ES,meta.example.RE.theta.2.ES)

ggplot(data=result.RE, aes(x=Method, y=ES.RE, fill=as.factor(tau2))) +
    geom_bar(stat="identity", position=position_dodge())
```

Add error bars here.

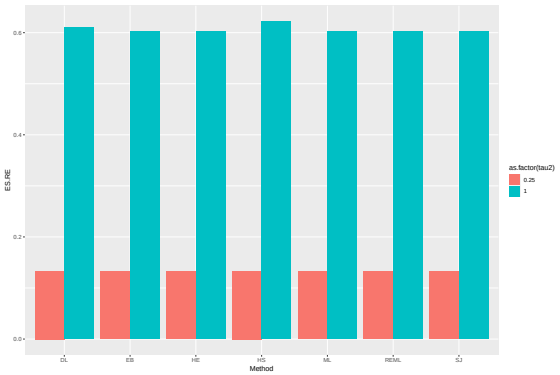
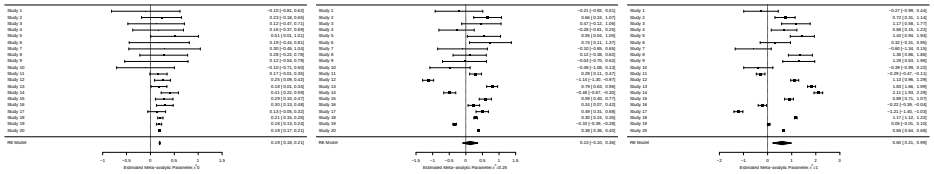


Figure 13.6: Various estimators of the treatment effect with random effects

13.1.4.2.1 Presenting the results of a random effects meta-analysis

In order to illustrate the results of a random effects meta-analysis, you can first show the forest plot. Let's see how it works in our example:

```
forest(meta.example.RE.ES,slab = paste('Study',data.meta$id,sep=' '),xlab=expression(pa
forest(meta.example.RE.theta.1,slab = paste('Study',data.meta$id,sep=' '),xlab=expressi
forest(meta.example.RE.theta.2,slab = paste('Study',data.meta$id,sep=' '),xlab=expressi
```



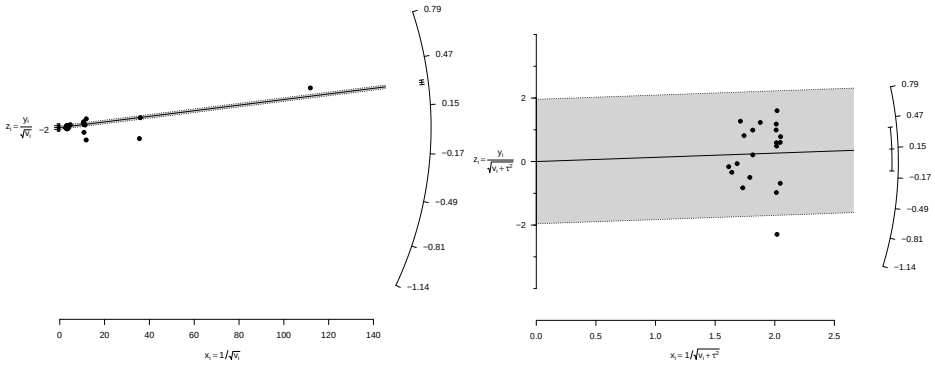
(a) τ^2 =0 (b) τ^2 =0.25 (c) τ^2 =1

Figure 13.7: Forest plots with random effects

Another very nice and useful graphical presentation device is a radial (or Galbraith) plot. It relates the invserse of the standard errors to the effect sizes normalized by their standard errors. Each data point is also related a radius by the line passing through the origin. The Radial plot enables to visualize the noise in the dataset, and is especially useful when comparing a fixed and a random effects estimator for the same study.

```
meta.example.FE.theta.1 <- rma(yi = data.meta$theta.1,vi=data.meta$var.ES,method="FE")
radial(meta.example.FE.theta.1)
radial(meta.example.RE.theta.1)
```

Figure 13.8 shows how the mechanics of the fixed effects estimator differs from the mechanics of the random effects one. In the presence of treatment effect heterogeneity, the fixed effect estimator faces two issues:



(a) Fixed effects (b) Random effects

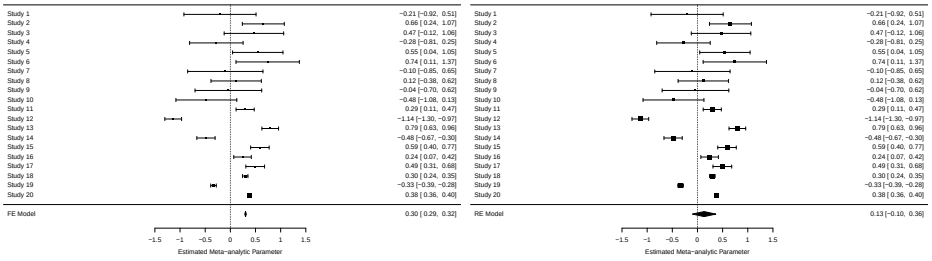
Figure 13.8: Radial plots with fixed and random effects $\tau^2 = 0.25$

1. It gives too much weight to very precise estimators. The random effects estimator undoes part of this importance by adding τ^2 to the weights of each observation.
2. It overestimates overall precision by ignoring the sampling variance stemming from treatment effect heterogeneity across sites. The random effects estimator corrects for that by estimating τ^2 and adding it to the estimate of the total variance of the treatment effect.

Example 13.8. Let's see how big a difference using random versus fixed effects does to the estimation of treatment effects.

Let's plot the two forest plots for the example with $\tau = 0.25$.

```
forest(meta.example.FE.theta.1,slab = paste('Study',data.meta$id,sep=' '),xlab='Estimated Meta-an
forest(meta.example.RE.theta.1,slab = paste('Study',data.meta$id,sep=' '),xlab='Estimated Meta-an
```



(a) Fixed effects (b) Random effects

Figure 13.9: Fixed vs random effects with $\tau^2 = 0.25$

Figure 13.9 clearly shows that the inclusion of τ^2 in the weights and precision estimates makes a huge difference to the meta-analytic estimate. The fixed effects estimator yields an estimate of our treatment effect of 0.3 ± 0.02. The

random effects estimator yields an estimate of our treatment effect of 0.13 ± 0.23. With $\tau^2 = 1$, the random effects estimator yields an estimate of our treatment effect of 0.6 ± 0.39. Remember that the true effect size of our treatment is 0.2. With $\tau^2 = 1$, the random effects estimator barely contains the truth in its 95 % confidence interval.

13.1.5 Meta-regression

A Meta-regression tries to explain the heterogeneity in treatment effects across studies using observed covariates. The idea is to identify characteristics of the studies or of the sites that are correlated with how treatment effects vary.

13.1.5.1 The Meta-regression model

The main equation that we want to estimate is as follows (Raudenbusch, 2009):

$$\hat{\theta}_k = \mathbf{X}_k\beta + \epsilon_k + \nu_k, \tag{13.1}$$

Center regressors at the mean?

where $\mathbf{X}_k$ is a line vector containing the value of the variables suspected to be correlated with treatment effect heterogeneity for study k and β is a column vector of the corresponding coefficients, of the same dimension as $\mathbf{X}_k$. $\mathbf{X}_k$ contains a 1 as its first term, so that β_0, the first component of the vector β measures the effect of the treatment when all other regressors are set to zero. It might thus be a good idea to set the regressors as deviations around their means if we want β_0 to capture the average effect of the treatment. The error term ϵ_k captures the heterogeneity in estimated effect sizes that is due to sampling noise. The error term ν_k captures the heterogeneity in effect sizes across sites that remains after conditioning on $\mathbf{X}_k$. In addition, it is generally assumed that $\epsilon_k \sim \mathbf{N}(0, \hat{\sigma}_k^2)$ and $\nu_k \sim \mathbf{N}(0, \tau^2)$.

This model is in general called the **mixed effects linear model**. It contains at the same time fixed effects captured by $\mathbf{X}_k\beta$ and random effects captured by ν_k. Setting τ^2 to zero generates a **fixed effects linear model**. It is possible, as usual, to test for whether τ^2 is null or not, which is a test of whether the added covariates fully capture the heterogeneity in treatment effects across studies.

13.1.5.2 Estimating the meta-regression model

There are at least four ways to estimate the meta-regression model:

1. Weighted Least squares (WLS): mostly used for fixed effects models, where τ^2 is assumed to be zero.
2. Full Maximum Likelihood Estimator (FMLE)
3. Restricted Maximum Likelihood Estimator (RMLE)
4. Method Of Moments (MOM)

13.1.5.2.1 Weighted Least Squares The Weighted Least Squares (WLS) estimator imposes that $\tau^2 = 0$. It is thus appropriate when we have a fixed effects linear model. It is also used as a starting point for estimating the other models.

The WLS estimator of β is written as follows:

$$\hat{\beta}_{WLS} = \left(\sum_{k=1}^{N} \frac{1}{\hat{\sigma}_k^2} \mathbf{X}_k' \mathbf{X}_k \right)^{-1} \sum_{k=1}^{N} \frac{1}{\hat{\sigma}_k^2} \mathbf{X}_k' \hat{\theta}_k.$$

The WLS estimator is similar to the standard OLS estimator, except that it gives more weight to mmore precise estimates of the treatment effect. This is a generalization of the weighted average that we have studied in Section 13.1.3.

13.1.5.2.2 Full Maximum Likelihood Estimator The Full Maximum Likelihood Estimator (FMLE) is also a weighted estimator, but, as the random effects estimator presented in Section 13.1.4.2, it uses as weigths not only the precision estimates ($\frac{1}{\hat{\sigma}_k^2}$), but the inverse of the sum of the variance due to sampling noise and the variance due to variation in treatment effects across sites. In order to make all of this clearer, let's define $\omega_k = \epsilon_k + \nu_k$, and let's denote $\zeta_k^2 = \hat{\sigma}_k^2 + \tau^2$ the variance of ω_k. The estimatingn equations for the FMLE estimator are:

$$\hat{\beta}_{FMLE} = \left(\sum_{k=1}^{N} \frac{1}{\hat{\zeta}_k^2} \mathbf{X}_k' \mathbf{X}_k \right)^{-1} \sum_{k=1}^{N} \frac{1}{\hat{\zeta}_k^2} \mathbf{X}_k' \hat{\theta}_k,$$

$$\hat{\tau}_{FMLE}^2 = \frac{\sum_{k=1}^{N} \frac{1}{\hat{\zeta}_k^4} \left((\hat{\theta}_k - \mathbf{X}_k \beta)^2 - \hat{\sigma}_k^2 \right)}{\sum_{k=1}^{N} \frac{1}{\hat{\zeta}_k^4}}$$

where $\hat{\zeta}_k^2$ is an estimate of ζ_k^2. In general, the FEML model is estimated by using a first guess for β, for example $\hat{\beta}_{WLS}$. Using this first estimate, we can compute a first estimate of $\hat{\tau}^2$ and update the set of weights, and iterate until convergence.

13.1.5.2.3 Restricted Maximum Likelihood Estimator The Restricted Maximum Likelihood Estimator (RMLE) is a weigthed estimator that is very similar to the FMLE estimator, except that the estimation procedure focuses on estimating τ^2 first. As a consequence, the formula for the τ^2 estimator is different:

$$\hat{\beta}_{RMLE} = \left(\sum_{k=1}^{N} \frac{1}{\hat{\zeta}_k^2} \mathbf{X}_k' \mathbf{X}_k \right)^{-1} \sum_{k=1}^{N} \frac{1}{\hat{\zeta}_k^2} \mathbf{X}_k' \hat{\theta}_k,$$

$$\hat{\tau}^2_{RMLE} = \frac{\sum_{k=1}^{N} \frac{1}{\hat{\zeta}_k^4} \left((\hat{\theta}_k - \mathbf{X}_k \beta)^2 - \hat{\sigma}_k^2 \right) + \mathrm{tr} \left[\left(\sum_{k=1}^{N} \frac{1}{\hat{\zeta}_k^2} \mathbf{X}_k' \mathbf{X}_k \right)^{-1} \sum_{k=1}^{N} \frac{1}{\hat{\zeta}_k^2} \mathbf{X}_k' \mathbf{X}_k \right]}{\sum_{k=1}^{N} \frac{1}{\hat{\zeta}_k^4}}.$$

Again, this estimator an be computed in a recursive way, starting with an initial guesstimate for the parameters β, for example the simple WLS estimator.

13.1.5.2.4 Method Of Moments (MOM) The Methods Of Moments estimator (MOM) does not require to assume that the distirbution of ν_k is normal. MOM only assumes that the distribution of ν_k is i.i.d. with mean zero and variance τ^2. The MOM estimator is a three-step estimator:

1. Estimate β using a simple regression that does require knowing τ^2.
2. Estimate τ^2 from the residuals of this regression.
3. Run a Weighted Least Squares regression including the new estimate of τ^2 in the weights.

When the first step uses a simple OLS estimator, we have:

$$\hat{\beta}_{OLS} = \left(\sum_{k=1}^{N} \mathbf{X}_k' \mathbf{X}_k \right)^{-1} \sum_{k=1}^{N} \mathbf{X}_k' \hat{\theta}_k$$

$$\hat{\tau}^2_{OLS} = \frac{RSS - \sum_{k=1}^{N} \hat{\sigma}_k^2 - \mathrm{tr}(S)}{k - p - 1},$$

where RSS is the Residual Sum of Squares of the OLS regression, p is the number of covariates and:

$$S = \left(\sum_{k=1}^{N} \mathbf{X}_k' \mathbf{X}_k \right)^{-1} \sum_{k=1}^{N} \mathbf{X}_k' \mathbf{X}_k.$$

When the first step uses the WLS estimator, we have:

$$\hat{\tau}^2_{WLS} = \frac{WRSS - (k - p - 1)}{\mathrm{tr}(M)},$$

where $WRSS$ is the Residual Sum of Squares of the WLS regression and:

$$\text{tr}(M) = \sum_{k=1}^{N} \frac{1}{\hat{\sigma}_k^2} - \text{tr}\left(\left(\sum_{k=1}^{N} \frac{1}{\hat{\sigma}_k^2}\mathbf{X}_k'\mathbf{X}_k\right)^{-1} \sum_{k=1}^{N} \frac{1}{\hat{\sigma}_k^4}\mathbf{X}_k'\mathbf{X}_k\right).$$

13.1.5.3 Estimating sampling noise in the meta-regression model

13.1.5.3.1 Under homoskedasticity Under homoskedasticity, we're assuming that the variance of the treatment effect at various sites does not depend on the site characteristics $\mathbf{X}_k$. In that case, the variance of the estimated coefficients is estimated by:

$$\hat{\text{Var}}_{Homo}(\hat{\beta}) = \left(\sum_{k=1}^{N} \frac{1}{\hat{\sigma}_k^2 + \hat{\tau}^2}\mathbf{X}_k'\mathbf{X}_k\right)^{-1}.$$

13.1.5.3.2 Under heteroskedasticity Under heteroskedasticity, we allow the variance τ^2 to depend on $\mathbf{X}_k$. One correct estimator under that assumption is the Huber-White sandwich estimator:

$$\hat{\text{Var}}_{HW}(\hat{\beta}) = \left(\sum_{k=1}^{N} \frac{1}{\hat{\sigma}_k^2 + \hat{\tau}^2}\mathbf{X}_k'\mathbf{X}_k\right)^{-1} \sum_{k=1}^{N} \left(\frac{1}{\hat{\sigma}_k^2 + \hat{\tau}^2}\right)^2 \mathbf{X}_k'(\hat{\theta}_k - \mathbf{X}_k\hat{\beta})^2\mathbf{X}_k \left(\sum_{k=1}^{N} \frac{1}{\hat{\sigma}_k^2 + \hat{\tau}^2}\mathbf{X}_k'\mathbf{X}_k\right)^{-1}.$$

Example 13.9. Let's see how all of these estimators work in our example. In order to run a regression, I first need a covariate. I'm going to use the exact value of the noise that I've added to the regressions, so that I should be able to perfectly capture the heterogeneity in treatment effects. Let's see how this works.

```
# Let me generate the noise as a deviation from the true treatment effect
data.meta$nu.1 <- data.meta$theta.1 - data.meta$ES
data.meta$nu.2 <- data.meta$theta.2 - data.meta$ES

# Let me now run a meta regression
metaReg.example.RE.theta.1.ES <- lapply(estimators,function(method){return(rma(theta.1 ~ nu.1,dat
metaReg.example.RE.theta.2.ES <- lapply(estimators,function(method){return(rma(theta.2 ~ nu.2,dat

#Let's see what the estimation looks like when we ran an REML regression:
summary(metaReg.example.RE.theta.1.ES[[2]])
```

```
##
## Mixed-Effects Model (k = 20; tau^2 estimator: REML)
##
```

```
##   logLik  deviance       AIC       BIC      AICc
##  12.3736  -24.7471  -18.7471  -16.0760  -17.0329
##
## tau^2 (estimated amount of residual heterogeneity):     0 (SE = 0.0005)
## tau (square root of estimated tau^2 value):             0
## I^2 (residual heterogeneity / unaccounted variability): 0.00%
## H^2 (unaccounted variability / sampling variability):   1.00
## R^2 (amount of heterogeneity accounted for):            100.00%
##
## Test for Residual Heterogeneity:
## QE(df = 18) = 11.7947, p-val = 0.8577
##
## Test of Moderators (coefficient 2):
## QM(df = 1) = 1009.6599, p-val < .0001
##
## Model Results:
##
##          estimate      se     zval    pval   ci.lb   ci.ub
## intrcpt    0.1981  0.0085  23.1790  <.0001  0.1813  0.2148  ***
## nu.1       0.9708  0.0306  31.7751  <.0001  0.9109  1.0307  ***
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We can see that the estimated coefficient for the noise is large and almost equal to one, that the estimation of residual inter-site variance becomes zero and that the precision of our estimared treatment effect becomes much greater (since all variance due to site effects has been absorbed by the regressor).

Let's now look at the estimated coefficients. For that, we are going to use the function `coef(summary())` that extracts a dataframe of the coefficients along with their standard errors.

```
list.coef.tot.1 <- lapply(metaReg.example.RE.theta.1.ES,function(res){return(coef(summa
list.coef.tot.2 <- lapply(metaReg.example.RE.theta.2.ES,function(res){return(coef(summa

list.coef.1 <- unlist(lapply(list.coef.tot.1,'[[',c(1,1)))
list.se.1 <- unlist(lapply(list.coef.tot.1,'[[',c(2,1)))
list.coef.2 <- unlist(lapply(list.coef.tot.2,'[[',c(1,1)))
list.se.2 <- unlist(lapply(list.coef.tot.2,'[[',c(2,1)))

result.Meta <- data.frame(Method=rep(estimators,2),ES.Meta=c(list.coef.1,list.coef.2),s

ggplot(data=result.Meta, aes(x=Method, y=ES.Meta, group=as.factor(tau2), color=as.facto
    geom_point(stat="identity", position=position_dodge(0.7))+
    geom_errorbar(aes(min=ES.Meta-qnorm((1+delta.2)/2)*se.ES,max=ES.Meta+qnorm((1+delta
    geom_hline(aes(yintercept=ES(param)), colour="#990000", linetype="dashed")+
```

```
    expand_limits(y=0)
```

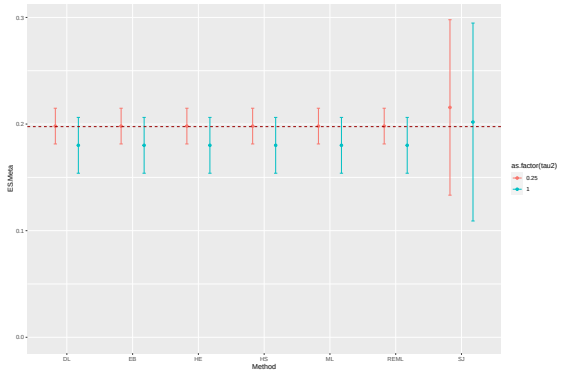


Figure 13.10: Various estimators of Effect Size in a Meta-Regression

Figure 13.10 shows that all estimators perform very well and deliver a precise estimate of the true effect.

I think SJn is the MOM estimator, check that.

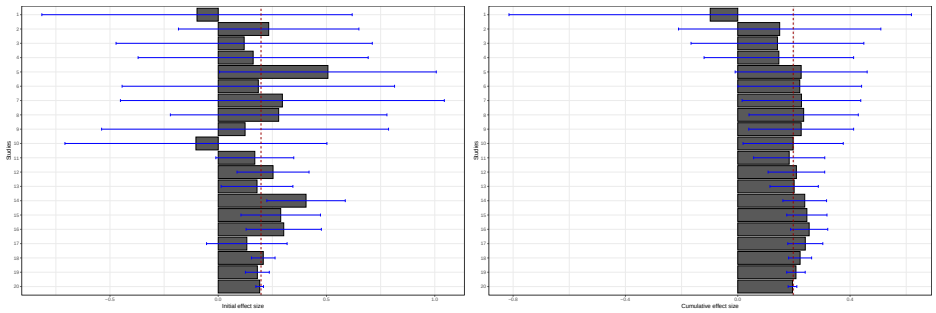
13.1.6 Constantly updated meta-analysis

Constantly updated meta-analysis performs the meta-analysis in a progressive manner, as the results keep arriving. This is a very important tool that enables us to aggregate constantly the information coming from different studies. Moreover, restrospectively, it helps us to assess when we would have reached enough precision so that we could have foregone an additional study. The way constantly updated meta-analysis works is simply by performing a new meta-analysis each time a new results pops up.

Example 13.10. Figure 13.11 shows how constantly updated meta-analysis works in our example.

```
cum.wmae.1 <- function(k,theta,sigma2){
  return(c(weighted.mean(theta[1:k],(1/sigma2[1:k])/(sum(1/sigma2[1:k]))),1/sum(1/sigma2[1:k])))
}

cum.wmae <- function(theta,sigma2){
  return(sapply(1:length(theta),cum.wmae.1,theta=theta,sigma2=sigma2))
}

cum.test <- as.data.frame(t(cum.wmae(data.meta$ES,data.meta$se.ES^2)))
colnames(cum.test) <- c('cum.ES','cum.var')
cum.test$id <- 1:nrow(cum.test)
cum.test$cum.se.ES <- sqrt(cum.test$cum.var)
```

```
ggplot(data.meta, aes(x=forcats::fct_rev(as.factor(id)), y=ES)) +
    geom_bar(position=position_dodge(), stat="identity", colour='black') +
    geom_errorbar(aes(ymin=ES-qnorm((delta.2+1)/2)*se.ES, ymax=ES+qnorm((delta.2+1)/2
    geom_hline(aes(yintercept=ES(param)), colour="#990000", linetype="dashed")+
    xlab("Studies")+
    ylab("Initial effect size")+
    theme_bw()+
    coord_flip()

ggplot(cum.test, aes(x=forcats::fct_rev(as.factor(id)), y=cum.ES)) +
    geom_bar(position=position_dodge(), stat="identity", colour='black') +
    geom_errorbar(aes(ymin=cum.ES-qnorm((delta.2+1)/2)*cum.se.ES, ymax=cum.ES+qnorm(
    geom_hline(aes(yintercept=ES(param)), colour="#990000", linetype="dashed")+
    xlab("Studies")+
    ylab("Cumulative effect size")+
    theme_bw()+
    coord_flip()
```



(a) Initial data set (b) Constantly updated dataset

Figure 13.11: Constantly updated meta-analysis

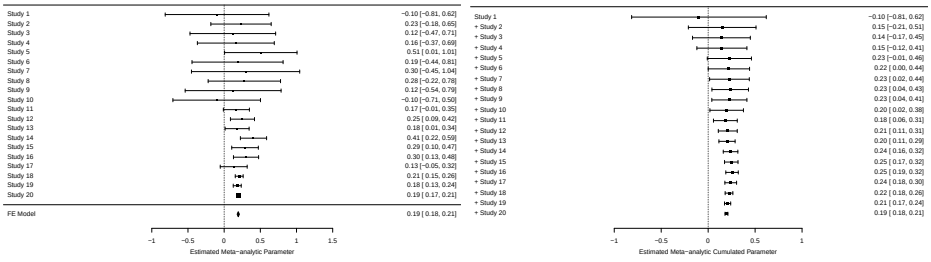
Figure 13.11 shows that combining several imprecise estimates might help you reach the same precision as running a larger experiment.
For instance, cumulating the first 10 studies with a small sample size ($N = 100$), the meta-analytic effect is estimated at 0.2 ± 0.18. This is very close to the individual estimate obtained from the first estimate with a larger sample size (sample 11 on Figure 13.11, with $N = 1000$): 0.17 ± 0.18. Both estimates actually have the exact same precision (because they actually have the same sample size). The same is true when combining the first 17 studies. The meta-analytic effect is estimated at 0.24 ± 0.06, while the effect estimated using one unique RCT with a larger sample size (sample 18 on Figure 13.11, with $N = 10^4$) is 0.21 ± 0.05. Finally, the same result occurs when combining the first 19 studies. The meta-analytic effect is estimated at 0.21 ± 0.03, while the effect

estimated using one unique RCT with a larger sample size (sample 20 on Figure 13.11, with $N = 10^5$) is 0.19 ± 0.02.

As a conclusion, constantly updated meta-analysis would have each time delivered the same result than the one found with a much larger study, rendering this additional study almost irrelevant. This is a very important result: beyond the apparent messiness of the first noisy estimates in Figures 13.1 and 13.3 lies an order that can be retrieved and made apparent using constantly updated meta-analysis. Sometimes, the answer is right there in front of our eyes, we just lack the ability to see it. Constantly updated meta-analysis serves as a binocular to magnify what is there. Think about how costly it woud be to run a very large study, just to find out that the we did not really need it because we had known the result all along.

Remark. Something pretty cool is that I can reproduce Figure 13.11 using the `metafor` package with much less lines of code.

```
forest(meta.example.FE,slab = paste('Study',data.meta$id,sep=' '),xlab='Estimated Meta-analytic P
cumul.meta.example.FE <- cumul(meta.example.FE, order=data.meta$id)
forest(cumul.meta.example.FE,slab = paste('Study',data.meta$id,sep=' '),xlab='Estimated Meta-anal
```



(a) Initial data set (b) Constantly updated dataset

Figure 13.12: Constantly updated meta-analysis with the 'metafor' package

You can also call each of the individual results of the cumulative meta-analysis using `cumul.meta.example.FE$estimate`. For example, the cumulated effect size after the 10 first studies is equal to 0.2 ± 0.18.

13.2 Publication bias and site selection bias

Up to now, we have made the assumption that a meta-analysis can access the results of **ALL** of the studies conducted on a topic. Problems appear when the publisehd record does not contain **ALL** of the studies conducted on a topic, but only a non-representative sample of them.

In the first section below, I detail the two main types of biases: publication bias and site selection bias. In the second section, I present methods that help to

detect and correct for publication bias. In the third section, I present methods tha help to detect and correct for site selection bias. In the last section, I take a step back and ask whether publication bias can be somehow optimal.

13.2.1 Sources of publication bias and of site selection bias and Questionable Research Practices

This section explains the sources of publication bias and site selection bias. I also expalin how they trigger the use of Questionable Research Practices that bias the published record even more.

13.2.1.1 Publication bias

There is publication bias when the eventual publication of the results of a research project depends on the results themselves. In general, the probability that a result is published increases drastically when the results reach the usual levels of statistical significance. On the contrary, the probability that a non significant result is published decreases drastically.

Give evidence of that behavior.

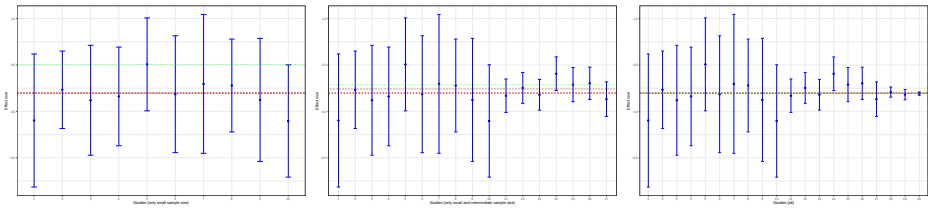
The reasons for this behavior are pretty well understood: editors and referees consider that only statistically significant results are of scientific interest, and that non significant results bring close to no information on a topic, especially if they are imprecise. Knowing this, most researchers choose not to invest time in trying to send a paper with a non significant result for publication.

What are the consequences of publishing only statistically significant results? Well, among imprecisely estimated effects, only the largest ones are going to reach publication, generating a pattern of overestimation of the true treatment effect. They key trade-off is whether the resulting bias is very large or not.

Example 13.11. What does publication bias look like in our example? Let's assume that only statistically significant effects are published. Would it change our estimate? In order to see whether that is the case, let's build Figure 13.1 with the addition of fixed effects estimator using all results and using only statistically significant results.

```
meta.example.FE.pubbias <- rma(yi = data.meta$ES[abs(data.meta$ES/sqrt(data.meta$var.ES

meta.example.FE.small <- rma(yi = filter(data.meta,id<=10)$ES,vi=filter(data.meta,id<=1
meta.example.FE.small.pubbias <- rma(yi = filter(data.meta,id<=10)$ES[abs(data.meta$ES/

meta.example.FE.interm <- rma(yi = filter(data.meta,id<=17)$ES,vi=filter(data.meta,id<=
meta.example.FE.interm.pubbias <- rma(yi = filter(data.meta,id<=17)$ES[abs(data.meta$ES

  ggplot(filter(data.meta,id<=10), aes(x=as.factor(id), y=ES)) +
      geom_point(position=position_dodge(), stat="identity", colour='blue') +
```

```
    geom_errorbar(aes(ymin=ES-qnorm((delta.2+1)/2)*se.ES, ymax=ES+qnorm((delta.2+1)/2)*se.ES),
    geom_hline(aes(yintercept=ES(param)), colour="#990000", linetype="dashed")+
    geom_hline(aes(yintercept=coef(meta.example.FE.small)), colour="#990000", linetype="dotted"
    geom_hline(aes(yintercept=coef(meta.example.FE.small.pubbias)), colour="green", linetype="d
    xlab("Studies (only small sample size)")+
    ylab("Effect size")+
    theme_bw()

ggplot(filter(data.meta,id<=17), aes(x=as.factor(id), y=ES)) +
    geom_point(position=position_dodge(), stat="identity", colour='blue') +
    geom_errorbar(aes(ymin=ES-qnorm((delta.2+1)/2)*se.ES, ymax=ES+qnorm((delta.2+1)/2)*se.ES),
    geom_hline(aes(yintercept=ES(param)), colour="#990000", linetype="dashed")+
    geom_hline(aes(yintercept=coef(meta.example.FE.interm)), colour="#990000", linetype="dotted
    geom_hline(aes(yintercept=coef(meta.example.FE.interm.pubbias)), colour="green", linetype="
    xlab("Studies (only small and intermediate sample size)")+
    ylab("Effect size")+
    theme_bw()

ggplot(data.meta, aes(x=as.factor(id), y=ES)) +
    geom_point(position=position_dodge(), stat="identity", colour='blue') +
    geom_errorbar(aes(ymin=ES-qnorm((delta.2+1)/2)*se.ES, ymax=ES+qnorm((delta.2+1)/2)*se.ES),
    geom_hline(aes(yintercept=ES(param)), colour="#990000", linetype="dashed")+
    geom_hline(aes(yintercept=coef(meta.example.FE)), colour="#990000", linetype="dotted")+
    geom_hline(aes(yintercept=coef(meta.example.FE.pubbias)), colour="green", linetype="dotted"
    xlab("Studies (all)")+
    ylab("Effect size")+
    theme_bw()
```



(a) All studies (b) Only imprecise studies (c) All studies

Figure 13.13: Illustration of publication bias

Figure 13.13 shows that publication bias can be a sizable problem. Remember that the true effect that we are trying to estimate is 0.2. When only imprecise studies with small sample size are available, the effect estimated using only the statistically significant studies (actually, the only study that reports a statistically significant result) is equal to 0.51 ± 0.5, while the effect estimated all the 10 studies with a small sample size is 0.2 ± 0.18. When studies with small

and intermediate sample size are available, the effect estimated using only the statistically significant studies is equal to 0.29 ± 0.08, while the effect estimated all the 17 studies with a small and intermediate sample size is 0.24 ± 0.06. It is only when studies with large and very large sample size are added to the estimation that publication bias is not a problem anymore. The effect estimated using only the statistically significant studies is equal to 0.2 ± 0.02, while the effect estimated all the studies is 0.19 ± 0.02.

As a conclusion of Figure 13.13, publication bias biases the true effect by:

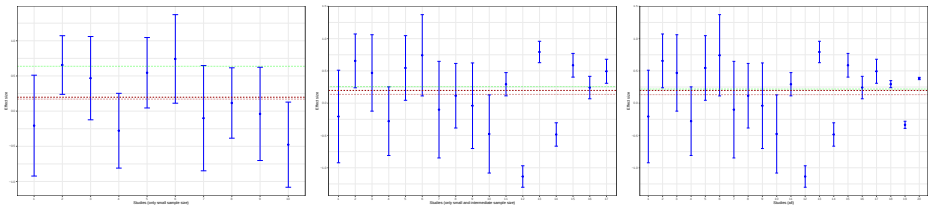
- 156 %, or 0.31 of a standard deviation, with studies with a small sample size,
- 45 %, or 0.09 of a standard deviation, with studies with a small or intermediate sample size,
- 1 %, or 0 of a standard deviation, with all studies.

With random effects, this behavior becomes even more severe, since only the sites at which the program has worked are going to appear in the published record, thereby biasing downards the true heterogeneity in treatment effects.

Example 13.12. Here is how that impacts the truth in our example:

```
meta.example.RE <- rma(yi = data.meta$theta.1,vi=data.meta$var.ES,method="REML")
meta.example.RE.pubbias <- rma(yi = data.meta$theta.1[abs(data.meta$theta.1/sqrt(data.m

meta.example.RE.small <- rma(yi = filter(data.meta,id<=10)$theta.1,vi=filter(data.meta
meta.example.RE.small.pubbias <- rma(yi = filter(data.meta,id<=10)$theta.1[abs(data.met

meta.example.RE.interm <- rma(yi = filter(data.meta,id<=17)$theta.1,vi=filter(data.meta
meta.example.RE.interm.pubbias <- rma(yi = filter(data.meta,id<=17)$theta.1[abs(data.me

  ggplot(filter(data.meta,id<=10), aes(x=as.factor(id), y=theta.1)) +
      geom_point(position=position_dodge(), stat="identity", colour='blue') +
      geom_errorbar(aes(ymin=theta.1-qnorm((delta.2+1)/2)*se.ES, ymax=theta.1+qnorm((de
      geom_hline(aes(yintercept=ES(param)), colour="#990000", linetype="dashed")+
      geom_hline(aes(yintercept=coef(meta.example.RE.small)), colour="#990000", linetyp
      geom_hline(aes(yintercept=coef(meta.example.RE.small.pubbias)), colour="green", l
      xlab("Studies (only small sample size)")+
      ylab("Effect size")+
      theme_bw()

  ggplot(filter(data.meta,id<=17), aes(x=as.factor(id), y=theta.1)) +
      geom_point(position=position_dodge(), stat="identity", colour='blue') +
      geom_errorbar(aes(ymin=theta.1-qnorm((delta.2+1)/2)*se.ES, ymax=theta.1+qnorm((de
      geom_hline(aes(yintercept=ES(param)), colour="#990000", linetype="dashed")+
      geom_hline(aes(yintercept=coef(meta.example.RE.interm)), colour="#990000", linety
      geom_hline(aes(yintercept=coef(meta.example.RE.interm.pubbias)), colour="green",
      xlab("Studies (only small and intermediate sample size)")+
```

```
      ylab("Effect size")+
      theme_bw()

  ggplot(data.meta, aes(x=as.factor(id), y=theta.1)) +
      geom_point(position=position_dodge(), stat="identity", colour='blue') +
      geom_errorbar(aes(ymin=theta.1-qnorm((delta.2+1)/2)*se.ES, ymax=theta.1+qnorm((delta.2+1)/2
      geom_hline(aes(yintercept=ES(param)), colour="#990000", linetype="dashed")+
      geom_hline(aes(yintercept=coef(meta.example.RE)), colour="#990000", linetype="dotted")+
      geom_hline(aes(yintercept=coef(meta.example.RE.pubbias)), colour="green", linetype="dotted"
      xlab("Studies (all)")+
      ylab("Effect size")+
      theme_bw()
```



(a) All studies (b) Only imprecise studies (c) All studies

Figure 13.14: Illustration of publication bias with Random Effects

Figure 13.14 shows that publication bias can be a sizable problem with random effects as well. Remember that the true effect that we are trying to estimate is 0.2. When only imprecise studies with small sample size are available, the effect estimated using only the statistically significant studies is equal to 0.64 $\pm$ 0.29, while the effect estimated all the 10 studies with a small sample size is 0.17 $\pm$ 0.27. When studies with small and intermediate sample size are available, the effect estimated using only the statistically significant studies is equal to 0.26 $\pm$ 0.39, while the effect estimated all the 17 studies with a small and intermediate sample size is 0.14 $\pm$ 0.27. It is only when studies with large and very large sample size are added to the estimation that publication bias is not a problem anymore. The effect estimated using only the statisticaly significant studies is equal to 0.22 $\pm$ 0.31, while the effect estimated all the studies is 0.13 $\pm$ 0.23.

As a conclusion of Figure 13.13, publication bias biases the true effect by:

- 223 %, or 0.44 of a standard deviation, with studies with a small sample size,
- 30 %, or 0.06 of a standard deviation, with studies with a small or intermediate sample size,
- 11 %, or 0.02 of a standard deviation, with all studies.

13.2.1.2 Site selection bias

There is site selection bias when researchers only implement an intervention in sites where they expect it to work. How can they do so? There are several informations that one can use to select sites for implementing a treatment and maximizing its effectiveness. First, researchers might only be able to work with highly motivated implementation agents. This might generate larger effects of the treatment. Second, researchers might have an informal knowledge on the types of individuals who react to the treatment well, and might decide to include them preferentially in the experimental study. Third, researchers might try out several different treatments in small informal pilots, and choose to run at scale only the most effective one(s). Finally, researchers, by conducting an extensive diagnosis of the problem that they face on the ground, might end up selecting a treatment that is more appropriate than a randomly selected treatment.

What are the consequences of site selection bias? If the selection process remains undocumented, a policy-maker trying to implement a treatment with a proven track record might fail to obtain the expected results because the site on which she decides to implement it is not representative of the distribution of sites in which the program has been evaluated. Ommitting to detail the process of site selection is akin to not explaining the recommendations of use, or worse the diagnosis of the disease, for a drug. If we do not know which disease the drug is effective against, we might end up expecting great results of a cold medecine against cancer.

Simulations.

13.2.1.3 Questionable Research Practices

Publication bias triggers and is aggravated by the use of Questionable Research Practices (QRPs). QRPs enable researchers (sometimes unknowingly) to obtain more statistically significant results than should be the case in view of the true effect of the treatment that they are looking at and the power of their test. Normally, when a treatment has no effect, only 5% of the treatment effects are going to turn out positive and significant when using a standard two-sided t-test. But, with QRPs, this figure can increase to 10, 20 or even, 50% in some cases.

References.

What are the most usual QRPs?

- Choosing a sample that generates significant effects: that includes stopping data collection when an effet of interest is found or deciding on critera of inclusion of observations based on statistical singificance. Sometimes, simply stopping to do robustness checks when results are significant is enough to bias usual tests of statistical significance.
- Choosing an outcome because the effect of the treatment is statistically significant. If we test a treatment on 100 outcomes for which the true effect of the treatment is null, between 2 and 3 outcomes are expected to

turn out with positive effects just by the sheer property of the tests that we are using.
- Choosing an identification strategy that generates significant treatment effects. Researcher smight try out various instruments and various natural experiments before settling down on the one that yields a statistically significant result.
- Choosing a subgroup for which significant effects are obtained. Analysis by subgroups offers a lot of opportunities for finding spurious significant effects.

The key question is whether these QRPs only move borderline significant results into the realm of significance, and thus have small effects of the size of the treatment effect, or if they enable to transform small effects into much larger ones. Note though that even if the QRPs only transform barely non-significant results in barely significant ones, the sheer repetition of these results in a meta-analysis is going to overestimate precision and might yield eventually to a confidence interval that does not contain the true effect, maybe by a large margin.

Simulations.

13.2.2 Detection of and correction for publication bias

Over the years, researchers have become aware of the problem that publication bias raises for meta-analyses and they have developed methods to detect and correct for it.

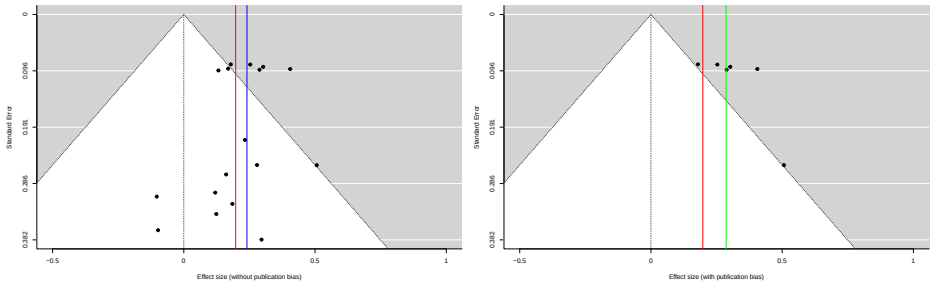
13.2.2.1 Funnel plot asymmetry

The first tool to identify the extent of publication bias is the funnel plot. The funnel plot plots the effect size as a function of its precision (or standard error). In the absence of publication bias, results should be distributed symetrically around the mean treatment effect estimate. We say that in this case the funnel plot is symmetric. In the presence of publication bias, results that are not statistically significant will be missing. They will be concentrated on the lower left part of the plot, were standard errors are large and estimated effects small. Missing results generate an asymetric funnel plot.

Example 13.13. Let's see how the funnel plot works in our example.

```
funnel(meta.example.FE.interm,xlab='Effect size (without publication bias)',xlim=c(-0.5,1),ylim=c
abline(v=ES(param),col="red")
abline(v=coef(meta.example.FE.interm),col="blue")
funnel(meta.example.FE.interm.pubbias,xlab='Effect size (with publication bias)',xlim=c(-0.5,1),y
abline(v=ES(param),col="red")
abline(v=coef(meta.example.FE.interm.pubbias),col="green")
```

Figure 13.15 shows how a funnel plot works. The x-axis presents the effect size of



(a) Without publication bias (b) With publication bias

Figure 13.15: Funnel plot with and without publication bias (homogeneous treatment effects, small and intermediate precision)

each study (here, in the homogeneous treatment effect case, analyzed using fixed effects). The y-axis presents the standard error, in an inverted scale, so that the most precise studies appear at the top of the graph. The two diagonal lines stemming out of zero present the 95% confidence intervals arounf zero, a.k.a. the two sided tests of statistical significance. In the plot, we focus of studies with small to intermediate precision. In our example, very precise studies are so much more precise that they make the problem of publication bias vanish.

When there is no publication bias, the funnel plot does not seem to exhibit asymmetry: there are as many imprecise studies on the left and on the right of the average effect. When there is publication bias, all the studies that fall within the confidence interval compatible with a zero treatment effect disappear. As a consequence, the remaining treatment effects are inflated versions of the truth. Moreover, we see that there is an increasing relationship between standard error and effect size. This is a sign of funnel plot asymmetry.

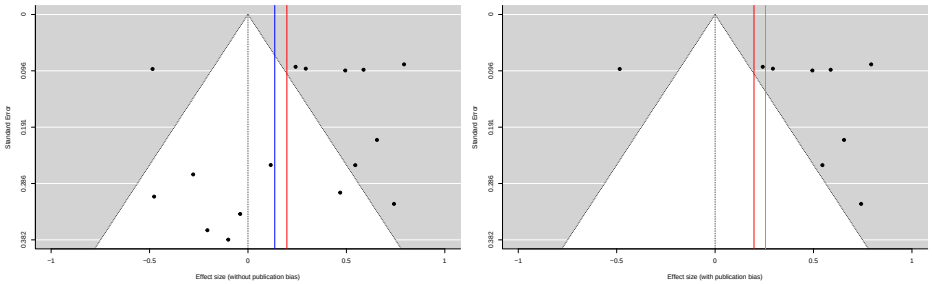
For the sake of completeness, Figure 13.16 shows what the funnel plot looks like with heterogeneous treatment effects analyzed using a random effects approach.

```
funnel(meta.example.RE.interm,xlab='Effect size (without publication bias)',xlim=c(-1,
abline(v=ES(param),col="red")
abline(v=coef(meta.example.RE.interm),col="blue")
funnel(meta.example.RE.interm.pubbias,xlab='Effect size (with publication bias)',xlim=
abline(v=ES(param),col="red")
abline(v=coef(meta.example.RE.interm.pubbias),col="green")
```

How do we implement these intuitions rigorously? The next section present the tools developed to do just that.

13.2.2.2 FAT-PET-PEESE

Docouliagos and Stanley (2012) have developed a method based on funnel plot



(a) Without publication bias (b) With publication bias

Figure 13.16: Funnel plot with and without publication bias (heterogeneous treatment effects, small and intermediate precision)

asymmetry to detect publication bias and correct for it. Their approach is based on three steps:

1. The Funnel Asymmetry Test (FAT) that tests whether there is a relationship between effect sizes and their precision.
2. The Precision-Effect Test (PET) that estimates the effect corrected for publication bias and tests for its existence.
3. The Precision-Effect Estimate with Standard Error (PEESE) that estimates the effect corrected for publication bias using a non-linear model for the standard error. When there is a genuine effect, PEESE offers a less biased estimate than PET.

The authors suggest to implement these procedures in a sequence, starting with the existence of publication bias, evidence for the existence of a non-zero effect once publication bias is accounted for and then estimate the bias-corrected effect when it is detected to be non-zero. Let's examine these approaches in turn.

The FAT and the PET are based on the following meta-regression:

$$\hat{\theta}_k = \alpha_0 + \alpha_1 \hat{\sigma}_k + \epsilon_k + \nu_k,$$

The PEESE is based on the following meta-regression:

$$\hat{\theta}_k = \beta_0 + \beta_1 \hat{\sigma}_k^2 + \epsilon_k + \nu_k,$$

Whether we assume that τ^2, the variance of ν_k is zero or not makes the FAT model a fixed or a random effects model. We run this regression with either Weighted Least Squares (in the fixed effects model) or with one of the methods appropriate for random effects (I'm going to use REML in what follows).

The FAT tests the assumption that $\alpha_1 = 0$ using a standard two-sided t-test. Rejecting the null means that there is sign of publication bias. The PET tests whether $\alpha_0 = 0$. Rejecting the null means that there is evidence of a true effect. The PEESE estimates the bias-corrected effect size by using $\hat{\beta}_1$.

Example 13.14. Let's see in practice how FAT, PET and PEESE work in our example. We are going first to run the regressions on the sample with homogeneous treatment effects, and thus we are going to use the simple Weighted Least Squares approach.

I'm focusing on the case with only small and intermediate precision estimates, as in the funnel plots in Figure 13.15.

```
FAT.PET.FE.interm <- rma(ES ~ sqrt(var.ES), data= filter(data.meta,id<=17),vi=filter(da
FAT.PET.FE.interm.pubbias <- rma(ES ~ sqrt(var.ES), data = filter(data.meta,id<=17,abs

PEESE.FE.interm <- rma(ES ~ var.ES, data= filter(data.meta,id<=17),vi=filter(data.meta
PEESE.FE.interm.pubbias <- rma(ES ~ var.ES, data = filter(data.meta,id<=17,abs(data.me

summary(FAT.PET.FE.interm)
```

```
##
## Fixed-Effects with Moderators Model (k = 17)
##
##   logLik  deviance       AIC       BIC      AICc
##   8.6124    9.5293  -13.2247  -11.5583  -12.3676
##
## I^2 (residual heterogeneity / unaccounted variability): 0.00%
## H^2 (unaccounted variability / sampling variability):   0.64
##
## Test for Residual Heterogeneity:
## QE(df = 15) = 9.5293, p-val = 0.8483
##
## Test of Moderators (coefficient 2):
## QM(df = 1) = 0.4952, p-val = 0.4816
##
## Model Results:
##
##               estimate      se     zval    pval    ci.lb   ci.ub
## intrcpt         0.2791  0.0634   4.4053  <.0001   0.1549  0.4033  ***
## sqrt(var.ES)   -0.3397  0.4828  -0.7037  0.4816  -1.2861  0.6066
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(FAT.PET.FE.interm.pubbias)
```

```
##
```

```
## Fixed-Effects with Moderators Model (k = 6)
##
##   logLik  deviance       AIC       BIC      AICc
##   6.4279    3.0645   -8.8557   -9.2722   -4.8557
##
## I^2 (residual heterogeneity / unaccounted variability): 0.00%
## H^2 (unaccounted variability / sampling variability):   0.77
##
## Test for Residual Heterogeneity:
## QE(df = 4) = 3.0645, p-val = 0.5471
##
## Test of Moderators (coefficient 2):
## QM(df = 1) = 1.1380, p-val = 0.2861
##
## Model Results:
##
##               estimate      se    zval    pval    ci.lb   ci.ub
## intrcpt         0.1352  0.1470  0.9195  0.3578  -0.1530  0.4233
## sqrt(var.ES)    1.6351  1.5327  1.0668  0.2861  -1.3691  4.6392
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(PEESE.FE.interm)
```

```
##
## Fixed-Effects with Moderators Model (k = 17)
##
##   logLik  deviance       AIC       BIC      AICc
##   8.6741    9.4058  -13.3483  -11.6818  -12.4911
##
## I^2 (residual heterogeneity / unaccounted variability): 0.00%
## H^2 (unaccounted variability / sampling variability):   0.63
##
## Test for Residual Heterogeneity:
## QE(df = 15) = 9.4058, p-val = 0.8554
##
## Test of Moderators (coefficient 2):
## QM(df = 1) = 0.6187, p-val = 0.4315
##
## Model Results:
##
##          estimate      se     zval    pval    ci.lb   ci.ub
## intrcpt    0.2568  0.0379   6.7699  <.0001   0.1825  0.3312  ***
## var.ES    -0.9426  1.1983  -0.7866  0.4315  -3.2912  1.4061
##
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(PEESE.FE.interm.pubbias)
```

```
##
## Fixed-Effects with Moderators Model (k = 6)
##
##   logLik  deviance       AIC       BIC      AICc
##   6.3347    3.2508   -8.6694   -9.0859   -4.6694
##
## I^2 (residual heterogeneity / unaccounted variability): 0.00%
## H^2 (unaccounted variability / sampling variability):   0.81
##
## Test for Residual Heterogeneity:
## QE(df = 4) = 3.2508, p-val = 0.5168
##
## Test of Moderators (coefficient 2):
## QM(df = 1) = 0.9516, p-val = 0.3293
##
## Model Results:
##
##          estimate      se    zval    pval    ci.lb    ci.ub
## intrcpt    0.2460  0.0569  4.3210  <.0001   0.1344   0.3576  ***
## var.ES     4.3828  4.4928  0.9755  0.3293  -4.4229  13.1885
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The results of the analysis are as expected, even though the small sample size prevents us from drawing conclusive results. When running the regression on the whole sample, in the absence of publication bias, we find that the estimated coefficient for the standard error in the meta-analytic regression is -0.34 $\pm$ 0.95. As a consequence, the FAT detects no sign of publication bias, with a pretty decent precision level. When running the regression on the sample with publication bias, we find that the estimated coefficient for the standard error in the meta-analytic regression is 1.64 $\pm$ 3. The coefficient is positive, as expected if larger results occur with smaller sample size, but the precision of this coefficient is too low for the FAT to be able to detect publication bias. This is a characteristic of the FAT to have low power, especially in our case where only one observation with small sample size drives all the results.

In the absence of publication bias, the PET detects a positive effect (0.28 $\pm$ 0.12) that is significantly different from zero, which is a sign of existence of a true effect. The PEESE is of 0.26 $\pm$ 0.07 . Following the practice suggested by Docouliagos and Stanley, we should refrain from using these estimates and focus only on the simple meta-analytic one (0.24 $\pm$ 0.06), since the FAT has not

detected signs of publication bias. In the presence of publication bias, the PET does not detect a positive effect (0.14 ± 0.29). The PEESE is of 0.25 ± 0.11 . Again, following the practice suggested by Docouliagos and Stanley, we should refrain from using these estimates and focus only on the simple meta-analytic one (0.29 ± 0.08), since the FAT has not detected signs of publication bias. Note nevertheless that in both cases the PEESE is almost as good as the meta-analytic estimate.

Let's now look at what happens when we are in a random effects world.

```
FAT.PET.RE.interm <- rma(theta.1 ~ sqrt(var.ES), data= filter(data.meta,id<=17),vi=filter(data.me
FAT.PET.RE.interm.pubbias <- rma(theta.1 ~ sqrt(var.ES), data = filter(data.meta,id<=17,abs(data.
FAT.PET.RE.interm.pubbias.pos <- rma(theta.1 ~ sqrt(var.ES), data = filter(data.meta,id<=17,abs(d

PEESE.RE.interm <- rma(theta.1 ~ var.ES, data= filter(data.meta,id<=17),vi=filter(data.meta,id<=1
PEESE.RE.interm.pubbias <- rma(theta.1 ~ var.ES, data = filter(data.meta,id<=17,abs(data.meta$the
PEESE.RE.interm.pubbias.pos <- rma(theta.1 ~ var.ES, data = filter(data.meta,id<=17,abs(data.meta

summary(FAT.PET.RE.interm)
```

```
##
## Mixed-Effects Model (k = 17; tau^2 estimator: REML)
##
##   logLik  deviance       AIC       BIC      AICc
## -12.7096   25.4191   31.4191   33.5433   33.6010
##
## tau^2 (estimated amount of residual heterogeneity):     0.2872 (SE = 0.1228)
## tau (square root of estimated tau^2 value):             0.5359
## I^2 (residual heterogeneity / unaccounted variability): 94.22%
## H^2 (unaccounted variability / sampling variability):   17.30
## R^2 (amount of heterogeneity accounted for):            0.00%
##
## Test for Residual Heterogeneity:
## QE(df = 15) = 391.5159, p-val < .0001
##
## Test of Moderators (coefficient 2):
## QM(df = 1) = 0.0212, p-val = 0.8842
##
## Model Results:
##
##               estimate      se     zval    pval    ci.lb   ci.ub
## intrcpt         0.1733  0.2929   0.5915  0.5542  -0.4009  0.7474
## sqrt(var.ES)   -0.1876  1.2878  -0.1457  0.8842  -2.7116  2.3364
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(FAT.PET.RE.interm.pubbias)
```

```
##
## Mixed-Effects Model (k = 10; tau^2 estimator: REML)
##
##   logLik  deviance       AIC       BIC      AICc
##  -7.3071   14.6142   20.6142   20.8526   26.6142
##
## tau^2 (estimated amount of residual heterogeneity):     0.3548 (SE = 0.1874)
## tau (square root of estimated tau^2 value):             0.5956
## I^2 (residual heterogeneity / unaccounted variability): 97.16%
## H^2 (unaccounted variability / sampling variability):   35.27
## R^2 (amount of heterogeneity accounted for):            5.24%
##
## Test for Residual Heterogeneity:
## QE(df = 8) = 367.0309, p-val < .0001
##
## Test of Moderators (coefficient 2):
## QM(df = 1) = 1.4973, p-val = 0.2211
##
## Model Results:
##
##               estimate      se     zval    pval    ci.lb   ci.ub
## intrcpt        -0.1548  0.3876  -0.3993  0.6897  -0.9145  0.6050
## sqrt(var.ES)    3.0194  2.4676   1.2236  0.2211  -1.8170  7.8558
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(PEESE.RE.interm)
```

```
##
## Mixed-Effects Model (k = 17; tau^2 estimator: REML)
##
##   logLik  deviance       AIC       BIC      AICc
## -12.6655   25.3311   31.3311   33.4552   33.5129
##
## tau^2 (estimated amount of residual heterogeneity):     0.2860 (SE = 0.1221)
## tau (square root of estimated tau^2 value):             0.5348
## I^2 (residual heterogeneity / unaccounted variability): 94.21%
## H^2 (unaccounted variability / sampling variability):   17.28
## R^2 (amount of heterogeneity accounted for):            0.00%
##
## Test for Residual Heterogeneity:
## QE(df = 15) = 392.0762, p-val < .0001
##
```

```
## Test of Moderators (coefficient 2):
## QM(df = 1) = 0.0801, p-val = 0.7772
##
## Model Results:
##
##          estimate      se     zval    pval    ci.lb   ci.ub
## intrcpt    0.1801  0.2103   0.8562  0.3919  -0.2321  0.5923
## var.ES    -0.8540  3.0178  -0.2830  0.7772  -6.7687  5.0607
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(PEESE.RE.interm.pubbias)
```

```
##
## Mixed-Effects Model (k = 10; tau^2 estimator: REML)
##
##   logLik  deviance       AIC       BIC      AICc
##  -7.3644   14.7288   20.7288   20.9671   26.7288
##
## tau^2 (estimated amount of residual heterogeneity):     0.3597 (SE = 0.1896)
## tau (square root of estimated tau^2 value):             0.5998
## I^2 (residual heterogeneity / unaccounted variability): 97.21%
## H^2 (unaccounted variability / sampling variability):   35.87
## R^2 (amount of heterogeneity accounted for):            3.91%
##
## Test for Residual Heterogeneity:
## QE(df = 8) = 370.0068, p-val < .0001
##
## Test of Moderators (coefficient 2):
## QM(df = 1) = 1.3503, p-val = 0.2452
##
## Model Results:
##
##          estimate      se    zval    pval    ci.lb    ci.ub
## intrcpt    0.0678  0.2540  0.2669  0.7895  -0.4301   0.5657
## var.ES     7.5975  6.5381  1.1620  0.2452  -5.2169  20.4118
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In the absence of publication bias, the FAT estimate of the coefficient for the standard error in the meta-analytic regression is -0.19 $\pm$ 2.52. As a consequence, the FAT detects no sign of publication bias. The PET does not detect a positive effect but its estimate is close to the truth, even if imprecise (0.28 $\pm$ 0.57). We would interpret this as absence of evidence for an effect. The PEESE is of 0.18 $\pm$ 0.41, close to the truth but highly imprecise. Following the practice suggested

by Docouliagos and Stanley, we should refrain from using these estimates and should focus only on the simple meta-analytic one (0.14 ± 0.27), since the FAT has not detected signs of publication bias.

In the presence of publication bias, the FAT estimate of the coefficient for the standard error in the meta-analytic regression is 3.02 ± 4.84. The coefficient is positive, as expected if larger results occur with smaller sample size, but the precision of this coefficient is too low for the FAT to be able to detect publication bias. The PET does not detect a positive effect, and even returns a negative one (-0.15 ± 0.76), however extremely imprecise. The PEESE at least returns a positive even though imprecise effect of 0.07 ± 0.5. Again, following the practice suggested by Docouliagos and Stanley, we should refrain from using these estimates and focus only on the simple meta-analytic one (0.26 ± 0.39), since the FAT has not detected signs of publication bias. In both cases the PEESE contains the true value in its confidence interval, but it does much less well than in the fixed effects case.

Some simulations would be great here in order to assess whether the estimated sampling noise of PEESE is actually of the same magnitude as what would stem from Monte Carlos.

I'd like to end this section on FAT-PET-PEESE by giving a graphical intuition of how this estimator corrects for publication bias. I'll supplement the graphical intuition with some intuition stemming from Heckman's selection model. The key intuition for understanding the FAT-PET and especially the PEESE estimator is the fact that, in the presence of publication bias, the meta-regression is akin to a censored or truncated model. As a consequence, and as Stanley and Docouliagos explain, we have something like:

$$\mathbb{E}[\hat{\theta}_k | |\frac{\hat{\theta}_k}{\hat{\sigma}_k}| > 1.96] = \alpha_0 + \alpha_1 \hat{\sigma}_k \lambda(\hat{\theta}_k, \hat{\sigma}_k) + \epsilon_k + \nu_k,$$

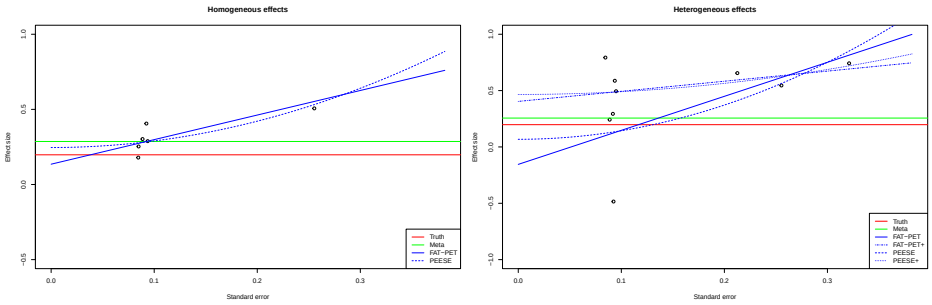
Do the derivation.

with λ the Inverted Mills Ratio. Approximating the nonlinear function of $\hat{\sigma}_k$ by a second order polynomial whose minium is when $\hat{\sigma}_k = 0$ gives rise to PEESE. FAT-PET approximate this function linearly instead. One way to see how this operates is to add the FAT-PET and PEESE estimates to the funnel plots.

Example 13.15. Let's see how the funnel plot works in our example.

```
plot(filter(data.meta,id<=17,abs(data.meta$ES/sqrt(data.meta$var.ES))>=qnorm((1+delta.
abline(h=ES(param),col="red")
abline(h=coef(meta.example.FE.interm.pubbias),col="green")
curve((coef(FAT.PET.FE.interm.pubbias)[1]+coef(FAT.PET.FE.interm.pubbias)[2]*x),col="bl
curve(expr=coef(PEESE.FE.interm.pubbias)[1]+coef(PEESE.FE.interm.pubbias)[2]*x^2,col="
legend("bottomright",
```

```
       legend = c("Truth", "Meta","FAT-PET","PEESE"),
       col = c('red', 'green','blue','blue'),
       lty= c(1,1,1,2),
       bg = "white")

plot(filter(data.meta,id<=17,abs(data.meta$theta.1/sqrt(data.meta$var.ES))>=qnorm((1+delta.2)/2))
abline(h=ES(param),col="red")
abline(h=coef(meta.example.RE.interm.pubbias),col="green")
curve((coef(FAT.PET.RE.interm.pubbias)[1]+coef(FAT.PET.RE.interm.pubbias)[2]*x),col="blue", add =
curve((coef(FAT.PET.RE.interm.pubbias.pos)[1]+coef(FAT.PET.RE.interm.pubbias.pos)[2]*x),col="blue
curve(expr=coef(PEESE.RE.interm.pubbias)[1]+coef(PEESE.RE.interm.pubbias)[2]*x^2,col="blue",lty=2
curve(expr=coef(PEESE.RE.interm.pubbias.pos)[1]+coef(PEESE.RE.interm.pubbias.pos)[2]*x^2,col="blu
legend("bottomright",
       legend = c("Truth", "Meta","FAT-PET","FAT-PET+","PEESE","PEESE+"),
       col = c('red', 'green','blue','blue','blue','blue'),
       lty= c(1,1,1,4,2,3),
       bg = "white")
```



(a) Fixed effects (b) Random effects

Figure 13.17: Funnel plot with PET and PEESE

On Figure 13.17, we see how PET and PEESE operate to deliver an estimate corrected for publication bias: they fit a line (PET) or a curve (PEESE) and use the intercept of this line or curve as an estimate of the true treatment effect. The plot for the hetergeneous treatment effects case suggests that both FAT-PET and PEESE are biased by a statistically significant negative result. I think there is a good case to be made for focusing only on results of the same sign when using these tools. When we get rid of that observation from the sample, the FAT estimate of the coefficient for the standard error in the meta-analytic regression is 0.9 ± 2.38. The PET estimate is now 0.4 ± 0.34. The PEESE estimates an effect of 0.47 ± 0.21. This correction does not seem to improve the estimator much in our example.

Nevertheless, it is worth to investigate further how PEESE behaves when observations from the over side of zero enter the picture. They seem to introduce

a lot of noise. I'd advocate for always using only values from one side, but we need theory and simulations to prove that intuition.

13.2.2.3 P-curving

P-curving has been proposed by Uri Simonsohn, Leif Nelson and Joseph Simmons in order to measure the evidential value of a set of published results. The basic idea is rather simple: when there is a true effect, the distribution of p-values of statistically significant results should be denser at lower p-values. This is because when there is a true effect, the density of the distribution of the p-values of statistically significant results decreases with the p-values. When there is no effect and in the absence of QRPs, p-values of statistically significant results are uniformly distributed, and their density is thus flat. When there is no effect and there are QRPs, the density of the distribution of the p-values of statistically significant results increases with the p-values. P-curving interprets the shape of teh p-curve as showing signs of true effect (we say it has evidential value), no effect, or QRPs. P-curving has two applications: detection of publication bias and QRPs and correction for publication bias and QRPs.

13.2.2.3.1 Proof of evidential value using p-curving The basic idea behind using p-curving for measuring whether a result has evidential value rests on the fact that, when there is no effects and no QPRs, p-values of statistically significant results are distributed uniformly. This is because, in the absence of any effect and of QRPs, the p-value measures the probability that a result of the same size or higher happens. When the effect is non existent and there are no QRPs, a p-value of 0.05 will occur 5% of the time and a p-value of 0.04 will occur 4% of the time So, p-values between 0.05 and 0.04 will occur 1% of the time, as p-values between 0.04 and 0.03 and so on. When there is a true effect, more small p-values are observed than larger ones. When there is no effet and there are QRPs, more p-values are observed closer to 0.05 than further away.

How to go from this intuition to testing for the existence of evidential value? One first very simple approach would simply be to separate the set of statistically significant p-values $[0, 0.05]$ in half. In the absence of effect and of QRPs, the probability that a statistically significant p-value falls into one of these two sets ($[0, 0.025]$ and $]0.025, 0.05]$) is 0.5. Comparing the actual proportion of p-values falling in these sets to the theoretical uniform value gives a first simple test of evidential value.

A rigorous test can be built by computing the probability that an event such as observed would have happened under the null of no effect and no QRPs. This can be done using the Binomial law, since under the null of no effect and no QRPs, X, the number of results falling in the $]0.025, 0.05]$ set, follows a binomial $Bi(n, p)$, with n the number of studies and $p = 0.5$. The probability of observing x studies among n in the $]0.025, 0.05]$ set is thus $\Pr(X = x) = b(x, n, p)$, where $b(x, n, p)$ is the density of the binomial distribution. The probability of observing x studies

or more among n in the $]0.025, 0.05]$ set is $\Pr(X \geq x) = 1 - B(x-1, n, p)$, where $B(x, n, p)$ is the cumulative of the binomial distribution.

For example, if we have 6 studies, with five of them falling in the $]0.025, 0.05]$ set, we have $5/6 = 83$ % of studies close to 0.05. Under the null of no effect and no QRPs, this would have happened with probability 0.09. If we define the alternative to be the existence of QRPs, this or something worse (meaning more QRPs) would have happened with probability 0.11. This is not definitive evidence against the null and in favor of QRPs, but we're getting there. If we define the alternative as being a true effect, we would obtain the same results per symmetry of the binomail distribution. Let's write a function that takes a vector of p-values and returns the binomial test statistic and p-value for the null of no effect and no QRPs.

```
pcurve.binom <- function(pvalues,alter='True'){
  p.upper <- ifelse(pvalues>0.025,1,0)
  p.lower <- ifelse(pvalues<=0.025,1,0)
  pbinom.True <- pbinom(sum(p.lower),length(pvalues),0.5)
  if (alter=='QRP'){
    pbinom.True <- 1-pbinom(sum(p.upper)-1,length(pvalues),0.5)
  }
  return(pbinom.True)
}
```

Another test use the distribution of the p-values of the p-values, or pp-value. The test works as follows. Let's say you have a set of p-values p_i. For each p_i, compute the probability to observe this p-value or a more extreme one if the null were true. This is not too hard since p_i is distributed uniformly on $[0, 0.05]$ under the null and thus both its density and cumulative are known. The only twist you have to pay attention to is how you define extreme. This depends on what is your alternative hypothesis. If you are comparing the null to a case with QRPs, then more extreme means a p-value closer to 0.05. If you are comparing the null to a case where there is a true effect, then more extreme means a p-value closer to 0. In the latter case, the pp-value of $p_i = p_k$ is $pp_k^r = \Pr(p_i \leq p) = p_k/0.05$, from the cumulative of a uniform. In the former case, the pp-value of $p_i = p_k$ is $pp_k^l = \Pr(p_i \geq p) = 1 - p_k/0.05$. Now, you can aggregate the pp-values using Fisher's method: $F_{pp}^s = -2\sum_k \ln(pp_k^s)$, for $s \in \{l, r\}$. F_{pp}^s is distributed $\chi^2(2k)$ under the null.

```
pp.test <- function(pvalues,alter='True'){
  pp <- pvalues/0.05
  if (alter=='QRP'){
    pp <- 1-pp
  }
  Fpp <- -2*sum(log(pp))
  dfChis <- 2*length(pvalues)
  pChisquare.Fpp <- pchisq(Fpp,dfChis,lower.tail=F)
```

```
  qChisquare.5 <- qchisq(0.05,dfChis,lower.tail=F)
  return(c(pChisquare.Fpp,Fpp,dfChis,qChisquare.5))
}
```

Imagine for example that we have three studies with p-values 0.001, 0.002 and 0.04. Let's compute the test against both alternatives:

```
pvalex <- c(0.001,0.002,0.04)
p.binom.test.True <- pcurve.binom(pvalex,alter='True')
p.binom.test.QRP <- pcurve.binom(pvalex,alter='QRP')

pp.ex.QRP <- pp.test(pvalex,alter='QRP')
pp.ex.True <- pp.test(pvalex,alter='True')
```

The Chi-square statistic against QRPs is 3.34 and the corresponding p-value is 0.76. The Chi-square statistic against a true effect is 14.71 and the corresponding p-value is 0.02.

There is a last test based on the p-curve tool that compares the actual distribution of statistically significant p-values to that that would be generated by a real but small effect, one that we would be powered to detect in only 33% of the samples. The test simply reverses the null and alternative of the previous test when the alternative was that there exists a true effect. I do not really see what one has to gain from this additional test so I'm going to abstain from encoding for now. It uses non-central distributions to compute the pp-values.

Code the additional pp-value test.

Example 13.16. Let's see how these tests work in our example.

We first have to compute the p-values for each statistically significant effect. Then, we can implement our tests.

```
data.meta$p.FE <- 2*pnorm(abs(data.meta$ES/sqrt(data.meta$var.ES)),lower.tail=F)
data.meta$p.RE <- 2*pnorm(abs(data.meta$theta.1/sqrt(data.meta$var.ES)),lower.tail=F)

pvalex.FE <- filter(data.meta,id<=17,abs(data.meta$ES/sqrt(data.meta$var.ES))>=qnorm((
pvalex.RE <- filter(data.meta,id<=17,abs(data.meta$theta.1/sqrt(data.meta$var.ES))>=qn

p.binom.test.True.FE <- pcurve.binom(pvalex.FE,alter='True')
p.binom.test.QRP.FE <- pcurve.binom(pvalex.FE,alter='QRP')
pp.ex.QRP.FE <- pp.test(pvalex.FE,alter='QRP')
pp.ex.True.FE <- pp.test(pvalex.FE,alter='True')

p.binom.test.True.RE <- pcurve.binom(pvalex.RE,alter='True')
p.binom.test.QRP.RE <- pcurve.binom(pvalex.RE,alter='QRP')
pp.ex.QRP.RE <- pp.test(pvalex.RE,alter='QRP')
pp.ex.True.RE <- pp.test(pvalex.RE,alter='True')
```

In the homogeneous effects case, the p-value of the null of an absence of an effect versus QRPs is of 0.76. The p-value of a null of an absence of an effect versus a true effect is 0. In the heterogeneous effects case, the p-value of a null of an absence of an effect versus QRPs is of 1 while the test statistic of a null of an absence of an effect versus a true effect is 0. In both case, we clearly reject the absence of an effect. As a consequence, the set of p-values has evidential value. We also reject the existence of QRPs. That means that there is no p-hacking creating an undue mass of p-values close to 0.05, but that does not mean that there is no publicaiton bias. P-curving has nothing to say about publication bias. It can only say whether there is a true effect or not and whether there are signs of QRPs.

A cool way to present the results of p-curving is to draw the density of the statistically significant p-values against a uniform and the density that would occur under 33% power. Let me try and build such a graph in our example. First, we have to split the overall set $[0, 0.05]$ into equal-sized p-values bins, lets say $[0, 0.01[$, $[0.01, 0.02[$, $[0.02, 0.03[$, $[0.03, 0.04[$, $[0.04, 0.05]$. I'm gonna name each interval after its higher end point. Second, we have to compute how many of our observations fall in each of the bins. Third, just plot the corresponding density.

The addition of the density of p-values if the real test had 33% power is slightly more involved, because it requires the notions of power and MDE that we studied in Chapter 7. As in Chapter 7, we're going to use the CLT approximation to the distribution of the treatment effect estimate over sampling replications. The key idea is to recognize that, with a power of κ for a two-sided test, the distribution of the treatment effect divided by its standard error $\sqrt{V[\hat{E}]}$ is a standard normal centered at $MDE^n_{\kappa,\alpha} = \frac{MDE_{\kappa,\alpha}}{\sqrt{V[\hat{E}]}} = \Phi^{-1}(\kappa) + \Phi^{-1}(1-\alpha/2)$. This is an approximation that assumes away the mass of the distribution that lies below zero. This appromixation is useful since it delivers closed form solutions and it most of the time is accurate enough. The lower below $\kappa = 33\%$ we're going, the likelier it is that this approximation is at fault. Now, the probability that this distribution gives a p-value of 0.05 or smaller for a two-sided test of the true effect being zero with size 5% is equal to $\Phi(MDE^n_{\kappa,\alpha} - \Phi^{-1}(1-\alpha/2)) = \kappa$ by definition. The probability that it gives a p-value of p for the same test is of $\Phi(MDE^n_{\kappa,\alpha} - \Phi^{-1}(1-p/2))$. Conditionnal on having a p-value inferior to 5% (*i.e.* a statistically significant result), the probability of having a p-value between p_1 and p_2 (*i.e.* the pp-value) is thus:

$$pp_{\kappa,\alpha}(p_1, p_2) = \frac{1}{\kappa}\left(\Phi(MDE^n_{\kappa,\alpha} - \Phi^{-1}(1-p_2/2)) - \Phi(MDE^n_{\kappa,\alpha} - \Phi^{-1}(1-p_1/2))\right).$$

Let's write a function that gives us this result.

```
MDE.var <- function(alpha=0.05,kappa=0.33,varE=1){
  return((qnorm(kappa)+qnorm(1-alpha/2))*sqrt(varE))
}

ppCurvePower <- function(p1,p2,alpha=0.05,kappa=0.33,varE=1){
  return((pnorm(MDE.var(alpha=alpha,kappa=kappa)-qnorm(1-p2/2))
          -pnorm(MDE.var(alpha=alpha,kappa=kappa)-qnorm(1-p1/2)))/kappa)
}
```

Now, let's plot the p-curve plot.

```
pCurve.hist <- function(pvalues,power=.33){
  dens1 <- sum(ifelse(abs(pvalues-0.005)<0.005,1,0)/length(pvalues))
  dens2 <- sum(ifelse(abs(pvalues-0.015)<0.005,1,0)/length(pvalues))
  dens3 <- sum(ifelse(abs(pvalues-0.025)<0.005,1,0)/length(pvalues))
  dens4 <- sum(ifelse(abs(pvalues-0.035)<0.005,1,0)/length(pvalues))
  dens5 <- sum(ifelse(abs(pvalues-0.045)<0.005,1,0)/length(pvalues))
  dens <- c(dens1,dens2,dens3,dens4,dens5)
  p.hist.1 <- cbind(c(0.01,0.02,0.03,0.04,0.05),dens)
  p.hist.1 <- as.data.frame(p.hist.1)
  colnames(p.hist.1) <- c('p','density')
  p.hist.1$Data <- c("Observed")
  p.hist.2 <- cbind(c(0.01,0.02,0.03,0.04,0.05),0.2)
  p.hist.2 <- as.data.frame(p.hist.2)
  colnames(p.hist.2) <- c('p','density')
  p.hist.2$Data <- c("Uniform")
  dens331 <- ppCurvePower(0,0.01)
  dens332 <- ppCurvePower(0.01,0.02)
  dens333 <- ppCurvePower(0.02,0.03)
  dens334 <- ppCurvePower(0.03,0.04)
  dens335 <- ppCurvePower(0.04,0.05)
  dens33 <- c(dens331,dens332,dens333,dens334,dens335)
  p.hist.3 <- cbind(c(0.01,0.02,0.03,0.04,0.05),dens33)
  p.hist.3 <- as.data.frame(p.hist.3)
  colnames(p.hist.3) <- c('p','density')
  p.hist.3$Data <- c("Power33")
  p.hist <- rbind(p.hist.1,p.hist.2,p.hist.3)
  return(p.hist)
}

p.hist.FE <- pCurve.hist(pvalex.FE)
p.hist.FE$Effect <- "Homogeneous"
p.hist.RE <- pCurve.hist(pvalex.RE)
p.hist.RE$Effect <- "Heterogeneous"
p.hist.ex <- rbind(p.hist.FE,p.hist.RE)
p.hist.ex$Effect <- factor(p.hist.ex$Effect,levels=c("Homogeneous","Heterogeneous"))
```

```
p.hist.ex$Data <- factor(p.hist.ex$Data,levels=c("Observed","Uniform","Power33"))

ggplot(data=p.hist.ex, aes(x=p, y=density, color=Data)) +
    geom_point() +
    geom_line() +
    facet_grid(. ~ Effect)+
    theme_bw()
```

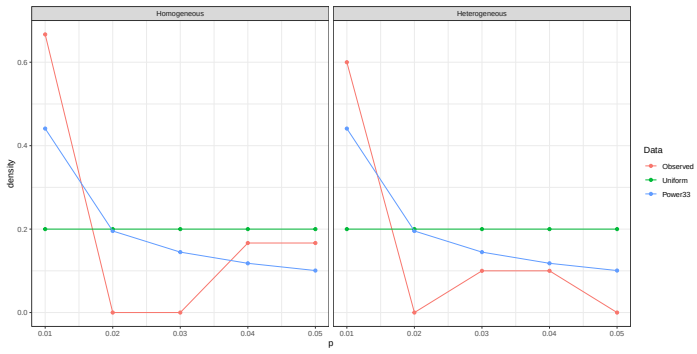


Figure 13.18: P-curve plot in our example

13.2.2.3.2 Correction for publication bias using p-curving In a separate paper, Simonsohn, Nelson and Simmons proposed a way to use p-curve to correct treatment effect estimates from publication bias. The underlying idea is rather simple, but also brilliant: the shape of the p-curve changes with the true underlying effect. It goes from uniform in the absence of any effect to right-skewed when there is an effect. The strength of the right-skewness tells us something about the underlying strength of the measured effect. For a given sample size, an increase in right-skewness will mean an increasae in effect size. The key difficulty is to separate the impact of sample size from that of effect size on the shape of the p-curve. Since sample size is known, it should be doable. Let's see how.

The key technical intuition behind the p-curve approach to correction for publication bias is to notice that the pp-curve computed for the true treatment effect should be uniform on $[0, 1]$. We have seen in the previous section that the pp-curve (the proportion of p-values that fall within identical intervals) is uniform when the true effect is zero. If we compute the pp-curve with a different assumption and apply it to the actual p-values that we observe, it is going to be uniform only for the actual treatment effect. So the only thing to do is to take all of the significant p-values and to compute their pp-curve for various levels of treatment effect, or, better, to look for the treatment effect that minimizes the distance between the observed pp-curve and a uniform. The authors propose to minimize a Kolmogorov-Smirnov metric to do so.

Let's first explore the way the concept works. For each treatment effect $\hat{\theta}_k$ and its estimated standard error $\hat{\sigma}_k$ we know from the CLT that they are distributed approximately as a normal. If we assume that the true treatment effect is θ_c, with c for candidate value, we know that $\frac{\hat{\theta}_k - \theta_c}{\hat{\sigma}_k}$ is distributed as a centered standardized normal distribution, under the assumption of homogeneous treatment effect across studies. Homogeneity is a crucial assumption for the pp-curve approach to correction for publication bias. I'll try to see how we can relax it later, but for now, I'm going to assume $\tau^2 = 0$. One way to compute the pp-value is to start directly with the treatment effect and its standard error, and to recover the pp-value from there. The pp-value is the probability that we have a draw of an estimator $\hat{\theta}$ of mean θ_c and standard error $\hat{\sigma}_k$ that is greater or equal to $\hat{\theta}_k$ given that it is a statistically significant result:

$$pp(\hat{\theta}_k, \hat{\sigma}_k, \theta_c) = \Pr\left(\hat{\theta} \geq \hat{\theta}_k \middle| \left|\frac{\hat{\theta}}{\hat{\sigma}_k}\right| \geq \Phi^{-1}\left(1 - \frac{\alpha}{2}\right)\right).$$

Let's assume that we are only looking at statistically significant results for two-sided t-tests, but that are located on the positive side of the threshold: $\frac{\hat{\theta}_k}{\hat{\sigma}_k} \geq \Phi^{-1}\left(1 - \frac{\alpha}{2}\right)$. This assumption will be innocuous in most cases of homogeneous treatment effect when the true effect we examine is positive since the mass of the distribution will fall on the positive side of the threshold, especially for significant results.

How do we compute the pp-value for a candidate value θ_c?

Theorem 13.3 (pp-value with homogeneous positive treatment effect)**.** *Under the assumption that the treatment effect is homogeneous across studies and equal to θ_c, that the estimated effects $\hat{\theta}_k$ are approximately normally distributed with mean θ_c and standard error $\hat{\sigma}_k$ and that we use only effects that are positive and significant at the level α following a two-sided test, the pp-value of a result with estimated effect $\hat{\theta}_k$ and estimated standard error is $\hat{\sigma}_k$:*

$$pp(\hat{\theta}_k, \hat{\sigma}_k, \theta_c) = \frac{\Phi\left(\frac{\theta_c - \hat{\theta}_k}{\hat{\sigma}_k}\right)}{\Phi\left(\frac{\theta_c}{\hat{\sigma}_k} - \Phi^{-1}\left(1 - \frac{\alpha}{2}\right)\right)}.$$

Proof.

$$\begin{aligned}
pp(\hat{\theta}_k, \hat{\sigma}_k, \theta_c) &= \Pr\left(\hat{\theta} \geq \hat{\theta}_k \middle| \frac{\hat{\theta}}{\hat{\sigma}_k} \geq \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\right) \\
&= \frac{\Pr\left(\hat{\theta} \geq \hat{\theta}_k \wedge \hat{\theta} \geq \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\hat{\sigma}_k\right)}{\Pr\left(\hat{\theta} \geq \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\hat{\sigma}_k\right)} \\
&= \frac{\Pr\left(\hat{\theta} \geq \hat{\theta}_k\right)}{\Pr\left(\hat{\theta} \geq \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\hat{\sigma}_k\right)} \\
&\approx \frac{1-\Phi\left(\frac{\hat{\theta}_k-\theta_c}{\hat{\sigma}_k}\right)}{1-\Phi\left(\frac{\Phi^{-1}\left(1-\frac{\alpha}{2}\right)\hat{\sigma}_k-\theta_c}{\hat{\sigma}_k}\right)} \\
&\approx \frac{\Phi\left(\frac{\theta_c-\hat{\theta}_k}{\hat{\sigma}_k}\right)}{\Phi\left(\frac{\theta_c}{\hat{\sigma}_k}-\Phi^{-1}\left(1-\frac{\alpha}{2}\right)\right)}.
\end{aligned}$$

The second equality stems from Bayes theorem. The third equality stems from the fact that $\hat{\theta} \geq \hat{\theta}_k \Rightarrow \hat{\theta} \geq \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\hat{\sigma}_k$ since $\hat{\theta}_k \geq \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\hat{\sigma}_k$ by assumption. The fourth equality stems from using the normality approximation to the distribution of $\hat{\theta}$. The fifth equality uses the usual property of the cumulative of the standardized and centered normal distribution that $1-\Phi(x) = \Phi(-x)$. □

Another way to compute the pp-value is to start from the p-value. This approach is especially useful when we do not have access to an estimate of the treatment effect, but only to the p-value of a two-sided t-test and to the sample size. In this case, we can generally only recover the effect size of the treatment effect d_c, that is the treatment effect scaled by the standard error of the outcome σ_Y (under the assumption of homogeneous treatment effects, there is no heteroskedasticity, and the variance of outcomes is identical in both the treatment and control groups). In order to do so, we use the fact that $d_c = \frac{\theta_c}{\sqrt{N}\hat{\sigma_k}}$ since $\hat{\sigma_k} \approx \frac{\sigma_Y}{\sqrt{N}}$ for the With/Without estimator without control variables (using the CLT). Note that this is a highly restrictive assumption, excluding that the With/Without estimator controls for covariates, or the use of other estimators.

Corollary 13.1 (Building pp-value from p-values)**.** *Under the assumption that the effect size is homogeneous across studies and equal to d_c, that the estimated effects sizes $\hat{d}_k$ are approximately normally distributed with mean d_c, that we use only effects are positive and significant at the level α following a two-sided test, and that $\hat{\sigma_k} \approx \frac{\sigma_Y}{\sqrt{N}}$, the pp-value of a result with sample size N_k and p-value $\hat{p}_k$ is:*

$$pp_p(\hat{p}_k, N_k, d_c) \approx \frac{\Phi\left(\sqrt{N}d_c - \Phi^{-1}\left(1 - \frac{p_k}{2}\right)\right)}{\Phi\left(\sqrt{N}d_c - \Phi^{-1}\left(1 - \frac{\alpha}{2}\right)\right)}.$$

Proof. From the formula for two-sided t-tests, we have that:

$$p_k = 2\left(1 - \Phi\left(\left|\frac{\hat{\theta}_k}{\hat{\sigma}_k}\right|\right)\right).$$

As a consequence:

$$\left|\frac{\hat{\theta}_k}{\hat{\sigma}_k}\right| = \Phi^{-1}\left(1 - \frac{p_k}{2}\right).$$

Using the assumption that all significant effects are positive, and the fact that $\sqrt{N}d_c \approx \frac{\theta_c}{\hat{\sigma}_k}$ under the assumption that $\hat{\sigma_k} \approx \frac{\sigma_Y}{\sqrt{N}}$, we obtain the result by using Theorem 13.3. □

Remark. Using only positive values is a benefit, since the preferred direction is less likely to have been slectively underreported.

Remark. The authors use Student t distributions instead of a normal approximation. This will not matter as loong as sample size is large enough. Generalizing the results of this section to Student-t distributions is simple, but the normal approximation should work most of the time.

Remark. There seems to be a mistake in the numerator in the original paper: the authors subtract power whereas it does not seem to be required. At least, I do not understand their derivation. Both appraoches yield similar results in their example, except for one case.

Remark. The assumption of homogeneous treatment effects is key here: it enables to use the standard normal as an approximation. The test has to be modified to account for heterogeneous treatment effects. Heterogeneity in treatment effects might bias the estimate.

In order to estimate the parameter θ_c (or d_c if using p-values only), the authors make use of the fact that, under the true θ_c, the pp-values are uniformly distributed on $[0, 1]$. The authors propose to choose the value $\hat{\theta}_c$ that makes the pp-curve as close to a uniform as possible as an estimator of θ_c. As a metric for estimating the distance between the observed pp-curve and the uniform $[0, 1]$, the authors propose to use the Kolmogorov-Smirnov statistic: the maximum value of

the absolute difference between the empirical cdf of pp-values and the theoretical values of the cdf of the uniform $[0, 1]$. The objective function proposed by the authors minimizes this distance.

Let's write the functions to compute just that:

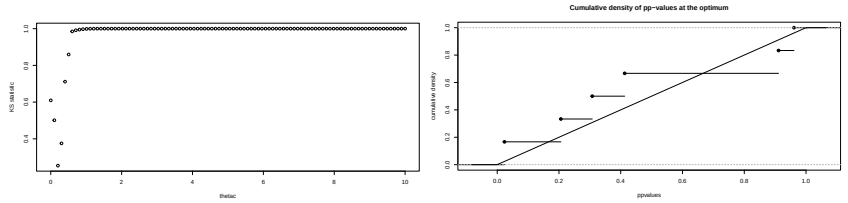
```
ppCurveEst <- function(thetac,thetak,sigmak,alpha=0.05){
  return((pnorm((thetac-thetak)/sigmak)/pnorm(thetac/sigmak-qnorm(1-alpha/2))))
}
#KS statistic
KS.stat.unif <- function(vector){
  return(ks.test(x=vector,y=punif)$statistic)
}
ppCurve.Loss.KS <- function(thetac,thetak,sigmak,alpha=0.05){
  ppvalues <- ppCurveEst(thetac=thetac,thetak=thetak,sigmak=sigmak,alpha=alpha)
  return(KS.stat.unif(ppvalues))
}
#Estimating thetac that minimizes the KS distance by brute grid search first
# will program the optimize function after
ppCurveEstES <- function(thetak,sigmak,thetacl,thetach,alpha=0.05,ngrid=100){
  # break thetac values in a grid
  thetac.grid <- seq(from=thetacl,to=thetach,length.out=ngrid)
  # computes the ppcurve for each point of the grid: outputs a matrix where columns are the ppcu
  ppCurve.grid <- sapply(thetac.grid,ppCurveEst,thetak=thetak,sigmak=sigmak,alpha=alpha)
  # compute KS stat for each value of thetac (over columns)
  KS.grid <- apply(ppCurve.grid,2,KS.stat.unif)
  # computes the value of thetac for which the KS stat is minimum (match identifies the rank of t
  min.theta.c <- thetac.grid[match(min(KS.grid),KS.grid)]
  # optimizes over KS stat to find value of thetac that minimizes the KS stat
  thetahat <- optimize(ppCurve.Loss.KS,c(min.theta.c-0.1,min.theta.c+0.1),thetak=thetak,sigmak=si
  # returns the optimal thetac, the grid of thetac, the KS stats on the grid, for potential plot,
  return(list(thetahat$minimum,thetac.grid,KS.grid,ecdf(ppCurve.grid[,match(min(KS.grid),KS.grid)
}
```

Example 13.17. Let's see how this approach works in our example.

Let's start with the homogeneous treatment effect case

```
# I'm keeping only significant and positive estimates
# Maybe this could be enforced within the function for ease of reading and use
ppCurveBiasCorrFE <- ppCurveEstES(thetak=filter(data.meta,id<=17,data.meta$ES>0,abs(data.meta$ES/
plot(ppCurveBiasCorrFE[[2]],ppCurveBiasCorrFE[[3]],xlab="thetac",ylab="KS statistic")
plot(ppCurveBiasCorrFE[[4]],xlab = "ppvalues",ylab="cumulative density",main="Cumulative density
curve(punif,add=T)
```

The bias corrected estimate using p-curving in the homogeneous treatment effect case is equal to 0.2, which is spot on. Remember the true treatment effect is 0.2. Let's see what happens when effects are heterogeneous.

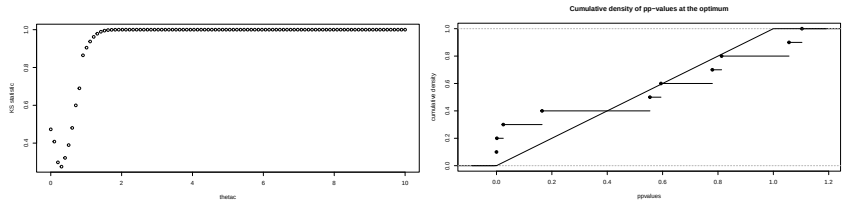


(a) KS objective function

(b) Cumulative distribution of pp-values at the optimum

Figure 13.19: Correction for publication bias using p-curving with homogeneous effects

```
# I'm keeping only significant and positive estimates
# Maybe this could be enforced within the function for ease of reading and use
ppCurveBiasCorrRE <- ppCurveEstES(thetak=filter(data.meta,id<=17,data.meta$ES>0,abs(da
plot(ppCurveBiasCorrRE[[2]],ppCurveBiasCorrRE[[3]],xlab="thetac",ylab="KS statistic")
plot(ppCurveBiasCorrRE[[4]],xlab = "ppvalues",ylab="cumulative density",main="Cumulati
curve(punif,add=T)
```



(a) KS objective function

(b) Cumulative distribution of pp-values at the optimum

Figure 13.20: Correction for publication bias using p-curving with heterogeneous effects

The bias corrected estimate using p-curving in the heterogeneous treatment effect case is equal to 0.35.

Remark. The approach might be incorrect in the heterogeneous treatment effect case since we do not normalize using τ^2. I think using an estimate of τ^2 to normalize the estimates would restore the validity of the procedure. It is also possible that the distribution of significant p-values is uniform even under heterogeneity of the treatment effect. In that case, the p-curve approach would still be valid. This is scope for further reasearch: first some simulations would be welcome. Simulations by the authors seem to suggest that p-curve does fine when treatment effects are heterogeneous: see here

Remark. Anther problem with p-curving when correcting for publication bias

is the existence of QRPs: QRPs might bias the bias correction because of an excess mass around 0.05. Simulations by the author in the original paper show that this biases the estimate downward.

Remark. Another approach than using the KS statistic could use the distance between the observed pp-curve in Figure 13.18 and the pp-curve with various levels of power. Once the power is identified, the effect size is identified. Still another approach would be to use the standardized distribution of effects (distributed as a standardized normal) to estimate the mean of the distribution and then recover the treatment effect.

13.2.2.4 Z-curving

See here.

13.2.2.5 Selection models

Publication bias generates a usual pattern for economists: a selection model. The probability that a result is published depends on several properties, let's say its significance, as measured for example by the t-statistic of a two-sided t-test of the estimated parameter being null. The resulting distribution of observed effect sizes is a truncated or censored distribution compared to the distribution of true effect sizes. It has been a long goal of statisticians and economists to try to recover properties of the distribution of a latent unobserved variable from what is observed (think distribution of wages for women, when labor force participation for women was far lower than it is today).

Statisticians have been using selection models to try to correct for publication bias since at least Hedges. In this section, I'm going to follow closely Andrews and Kasy's approach. Andrews and Kasy carefully delineate non-parameteric identification of a selection model in the case of heterogeneous treatment effects. They then present ways to estimate the parameters of this model and propose a web-app to perform their estimation strategy.

Andrews and Kasy assume that there is a true treatment effect in all the populations that is equal to θ_c^*. In each study k, the true treatment effect θ_k^* is drawn from a distribution with mean θ_c^*. If the distribution of θ_k^* is degenerate, then we have homogeneous treatment effects. The estimator of θ_k^* in each study, $\hat{\theta}_k^*$, is distributed as a normal centered at θ_k^* with variance σ_k^{*2}, whose estimator in the sample k is $\hat{\sigma}_k^{*2}$. The normality assumption is not too crazy here: it follows from the CLT.

Andrews and Kasy posit that, because of selection bias, we observe only a subset of these latent effects, noted $\hat{\theta}_k$, those for which $D_k = 1$. D_k is distributed as a Bernoulli random variable, with probability of success $p(\hat{Z}_k^*)$, where $\hat{Z}_k^* = \frac{\hat{\theta}_k^*}{\hat{\sigma}_k^*}$ is the test statistic of t-test for the null assumption that $\theta_k = 0$. So Andrews and Kasy assume that all publication bias is driven by the value of the t-statistic $\hat{Z}_k^*$.

Note that it is equivalent to assuming that it is driven by the p-value of this test, since one is a monotone transformation of the other.

As a consequence of the assumed selection model, the density of observed t-stats is (noting $Z_k^* = \frac{\theta_k^*}{\sigma_k^*}$ and $Z_k = \frac{\theta_k}{\sigma_k}$):

$$\begin{aligned} f_{\hat{Z}|Z}(\hat{z}|z) &= f_{\hat{Z}^*|Z^*,D=1}(\hat{z}|z) \\ &= \frac{\Pr(D_k = 1|\hat{Z}_k^* = \hat{z}, Z_k^* = z)}{\Pr(D_k = 1|Z_k^* = z)}\phi(\hat{z} - z) \\ &= \frac{p(\hat{z})}{\mathbb{E}[p(\hat{Z}_k^*)|Z_k^* = z]}\phi(\hat{z} - z). \end{aligned}$$

The first equality is obtained by using Bayes' equality twice (once to undo the conditioning on $D = 1$ and once to generate the conditioning on $Z_k^* = z$) and the fact that $f_{\hat{Z}^*|Z^*}$ is normally distributed with mean Z^* and variance 1.

The key result in Andrews and Kasy is their Proposition 3:

Proposition 13.1 (Identification of the true effect in meta-analysis)**.** *Under the assumption that $\theta_k^* \perp\!\!\!\perp \sigma_k^*$, and that the support of σ_k contains an open interval, $p(.)$ is identified up to scale and the distribution of θ_k^* is identified.*

Proof. See Andrews and Kasy's supplementary material. Let's detail the proof somehow. The proof works by using the way the density of observed $\hat{Z}$ changes with precision ($\hat{\pi}_k = \frac{1}{\hat{\sigma}_k}$). Without loss of generality, the authors choose to look at the density of $\hat{Z}$ when $\hat{\sigma}_k = 1$. They define $h(z) = f_{\hat{Z}^*|\hat{\sigma}_k^*}(z|1)$. The first insight of the proof is that identifying $h(.)$ identifies $p(.)$ and the distribution of θ_k^*, f_{θ^*}. When $h(.)$ is identified, f_{θ^*} is identified by deconvolution since $h = f_{\theta^*} * \phi$, where $*$ is the convolution operator. This is because we can think of $\hat{\theta}_k^* = \theta_k^* + \epsilon_k^*$, where ϵ_k^* is independent from θ_k^* (since $\theta_k^* \perp\!\!\!\perp \sigma_k^*$) and follows a normal with mean zero and variance $\hat{\sigma}_k^{*2}$, here one. The density of a sum of independent random variables is the convolution of their densities, hence the result. Now, we have:

$$\begin{aligned} f_{\hat{Z}|\hat{\sigma}}(z|s) &= f_{\hat{Z}^*|\hat{\sigma}^*,D=1}(z|z) \\ &= \frac{\Pr(D_k = 1|\hat{Z}_k^* = z, \hat{\sigma}_k^* = s)}{\Pr(D_k = 1|\hat{\sigma}_k^* = s)} f_{\hat{Z}^*|\hat{\sigma}^*}(z|s) \\ &= \frac{p(z)}{\mathbb{E}[p(\hat{Z}_k^*)|\hat{\sigma}_k^* = s]} f_{\hat{Z}^*|\hat{\sigma}^*}(z|s). \end{aligned}$$

As a consequence, we have:

$$p(z) = \mathbb{E}[p(\hat{Z}_k^*)|\hat{\sigma}_k^* = s]\frac{f_{\hat{Z}|\hat{\sigma}}(z|s)}{h(z)}.$$

So, once we know $h(z)$, we know $p(z)$ up to a constant, since $f_{\hat{Z}|\hat{\sigma}}(z|s)$ is known by definition, and $\mathbb{E}[p(\hat{Z}_k^*)|\hat{\sigma}_k^* = s]$ does not change with z.

In order to identify $h(z)$, we look at how the density of observed effects changes when precision changes:

$$\begin{aligned} g(z) &= \frac{\partial \ln f_{\hat{Z}|\hat{\sigma}}(z|\frac{1}{\pi})}{\partial \pi}|_{\pi=1} \\ &= C_1 + \frac{\partial \ln f_{\hat{Z}^*|\hat{\sigma}^*}(z|\frac{1}{\pi})}{\partial \pi}|_{\pi=1}. \end{aligned}$$

C_1 is a constant in z. This is because $p(z)$ does not depend on π and because $\mathbb{E}[p(\hat{Z}_k^*)|\hat{\sigma}_k^* = s]$ does not depend on z. Note that $g(z)$ is identified in the population.

Now, using the fact that, because $\theta_k^* \perp\!\!\!\perp \sigma_k^*$, we have $h = f_{\theta^*} * \phi$, and thus $f_{\hat{Z}^*|\hat{\sigma}^*}(z|\frac{1}{\pi}) = \int \phi(z - t\pi)df_{\theta^*}(t)$, and the fact that $\phi'(z) = -z\phi(z)$, we have:

$$\begin{aligned} \frac{\partial f_{\hat{Z}^*|\hat{\sigma}^*}(z|1)}{\partial z} &= -\int (z-t)\phi(z-t)df_{\theta^*}(t) \\ \frac{\partial^2 f_{\hat{Z}^*|\hat{\sigma}^*}(z|1)}{\partial z^2} &= -f_{\hat{Z}^*|\hat{\sigma}^*}(z|1) + \int (z-t)^2\phi(z-t)df_{\theta^*}(t) \\ \frac{\partial f_{\hat{Z}^*|\hat{\sigma}^*}(z|\frac{1}{\pi})}{\partial \pi}|_{\pi=1} &= \int t(z-t)\phi(z-t)df_{\theta^*}(t) \\ &= -\left[f_{\hat{Z}^*|\hat{\sigma}^*}(z|1) + z\frac{\partial f_{\hat{Z}^*|\hat{\sigma}^*}(z|1)}{\partial z} + \frac{\partial^2 f_{\hat{Z}^*|\hat{\sigma}^*}(z|1)}{\partial z^2}\right]. \end{aligned}$$

The last equation comes from rearranging all the terms in the various terms and factoring what remains. Note that $f_{\hat{Z}^*|\hat{\sigma}^*}(z|1)$ disappears when you add $\frac{\partial^2 f_{\hat{Z}^*|\hat{\sigma}^*}(z|1)}{\partial z^2}$. REgrouping under the intergal sign, factoring and simplifying gives the result.

Now, using the expression for $g(z)$ above, we have a second order differential equation in $h(.)$:

$$h''(z) = (C_1 - 1 - g(z))h(z) - zh'(z).$$

Given C_1 and initial conditions $h(0) = h_0$ and $h'(0) = h'_0$, there is a unique solution to this equation, thereby identifying $h(.)$, $p(.)$ and f_{θ^*}. The rest of the proof in Andrews and Kasy's supplementary material shows that C_1, h_0 and h'_0 are all identified. The proof builds new differential equations involving the second order derivative of $f_{\hat{Z}|\frac{1}{\pi}}$ with respect to π. The constants are identified after successive derivations with respect to z so that we have an equation for them that depends on the third order derivative of g. □

Remark. The authors derive an equation for the case where θ^* is normally distributed with mean θ and variance τ^2. The second order differential equation becomes:

$$-\frac{1}{\tau^2+1} = C_1 - g(z) - 1 + z\frac{z-\theta}{\tau^2+1} - \left(\frac{z-\theta}{\tau^2+1}\right).$$

The authors argue that evaluating this equation for different values of z pins down θ and τ^2. It seems not enough to prove identification since we need uniqueness of the parameter values obtained. There are already two values of θ compatible for agiven z and τ^2. We need more to ensure uniqueness.

Remark. Note that $p(z)$ does not depend on π is not a trivial assumption. It stems from assuming that the probability of publication only depends on π through $\hat{Z}^*$. It means for example that editors and authors do not look at precision independently of its effect on the t-statistic. The authors study identification in this case, assuming independence of the probabilities of selection based on both approaches.

For estimation, Andrews and Kasy follow the approach in Hedges and estimate their model by parametric maximum likelihood. They also propose in their supplementary material an approach based on a Generalized Method of Moments estimator that tries to emulate their identification strategy. Finally, they offer a web-app to implement their most straightforward estimators.

The likelihood can be written as:

$$f_{\hat{\theta},\hat{\sigma}}(t,s) = \frac{p\left(\frac{t}{s}\right)\int \phi\left(\frac{t-\theta}{s}\right) f_{\theta^*}(\theta)d\theta}{\int p\left(\frac{t'}{s}\right)\int \phi\left(\frac{t'-\theta}{s}\right) f_{\theta^*}(\theta)d\theta dt'} f_\sigma(s).$$

Under the assumption that θ^* is normally distributed with mean θ^*_c and variance τ^2, we have the following likelihood:

$$f^n_{\hat{\theta},\hat{\sigma}}(t,s) = \frac{p\left(\frac{t}{s}\right)\phi\left(\frac{t-\theta^*_c}{\sqrt{s^2+\tau^2}}\right)}{\int p\left(\frac{t'}{s}\right)\phi\left(\frac{t'-\theta^*_c}{\sqrt{s^2+\tau^2}}\right)dt'} f_\sigma(s).$$

Assuming that $p(.)$ is a step function such that $p(z) = p_1$ if $z < 1.96$ and $p(z) = 1$ if $z \geq 1.96$, we have:

$$f^n_{\hat{\theta},\hat{\sigma}}(t,s) = \frac{p\left(\frac{t}{s}\right)\phi\left(\frac{t-\theta_c^*}{\sqrt{s^2+\tau^2}}\right)}{p_1\Phi\left(\frac{1.96s-\theta_c^*}{\sqrt{s^2+\tau^2}}\right)+1-\Phi\left(\frac{1.96s-\theta_c^*}{\sqrt{s^2+\tau^2}}\right)} f_\sigma(s).$$

The likelihood is simply the product of this term computed at each values $t = \hat{\theta}_k$ and $s = \hat{\sigma}_k$:

$$\mathcal{L}(p_1, \theta_c^*, \tau^2) = \Pi_{k=1}^N f^n_{\hat{\theta},\hat{\sigma}}(\hat{\theta}_k, \hat{\sigma}_k)$$

Taking logs, we see that $f_\sigma(s)$ is a constant that does not contribute to the likelihood. We solve for the optimal vector of parameters by using a nonlinear optimisation routine. The authors use `nlminb`. One could also probably use `optim`. What is nice with these procedures is that they do not require computing the first and second order derivatives of the objective function: they compute them numerically.

Let's write an R function that maximizes this log likelihood:

```
# log-likelihood
Lk <- function(thetak,sigmak,p1,thetac,tau){
  f <- ifelse(thetak/sigmak<qnorm(1-0.05/2),p1,1)*dnorm((thetak-thetac)/sqrt(sigmak^2+tau^2))/(1-
  return(sum(log(f)))
}
#log-likelihood prepared for nlminb: vector of parameters and minimization
Lk.param <- function(param,thetak,sigmak){
  f <- Lk(thetak=thetak,sigmak=sigmak,p1=param[[1]],thetac=param[2],tau=param[3])
  return(-f)
}
```

Example 13.18. Let's see how this works in our example. Let's first prepare the sample. We are going to simulate two procedures of censoring: one with $p_1 = 0.5$ and one with $p_1 = 0$.

```
# sample with p1=0: only positive significant results
# homogeneous effects
thetak.FE.0 <- filter(data.meta,id<=17,data.meta$ES>0,abs(data.meta$ES/sqrt(data.meta$var.ES))>=q
sigmak.FE.0 <- sqrt(filter(data.meta,id<=17,abs(data.meta$ES/sqrt(data.meta$var.ES))>=qnorm((1+de
# heterogeneous effects
thetak.RE.0 <- filter(data.meta,id<=17,data.meta$theta.1>0,abs(data.meta$theta.1/sqrt(data.meta$v
sigmak.RE.0 <- sqrt(filter(data.meta,id<=17,abs(data.meta$theta.1/sqrt(data.meta$var.ES))>=qnorm(
```

Table 13.1: Parameter estimates of Andrews and Kasy selection model

	p_1	θ_c	τ
FE0	0.00	-100842.0	15537.35
FE50	0.04	-245889.4	219861.04
RE0	0.00	-5.4	0.19
RE50	0.03	-352375.7	277781.96

```
# sample with p1=0.1, for insignificant or negative results
p1 <- 0.5
# drawing 10% among insignificant and negative observations
set.seed(1234)
set.FE <- ifelse(runif(length(filter(data.meta,id<=17,data.meta$ES/sqrt(data.meta$var.
set.seed(1234)
set.RE <- ifelse(runif(length(filter(data.meta,id<=17,data.meta$theta.1/sqrt(data.meta

# homogeneous effects
thetak.FE.1 <- c(thetak.FE.0,filter(data.meta,id<=17,data.meta$ES/sqrt(data.meta$var.ES
sigmak.FE.1 <- c(sigmak.FE.0,sqrt(filter(data.meta,id<=17,data.meta$ES/sqrt(data.meta$v
# heterogeneous effects
thetak.RE.1 <- c(thetak.RE.0,filter(data.meta,id<=17,data.meta$theta.1/sqrt(data.meta$v
sigmak.RE.1 <- c(sigmak.RE.0,sqrt(filter(data.meta,id<=17,data.meta$theta.1/sqrt(data.m
```

```
# optimization procedure using nlminb
MaxEval<-10^5
MaxIter<-10^5
Tol<-10^(-8)
stepsize<-10^(-6)
lower.b <- c(0,-Inf,0)
upper.b <- c(1,Inf,Inf)
start.val <- c(0.5,1,1)

optim.Lk.FE.0 <- nlminb(objective=Lk.param, start=start.val,lower=lower.b,upper=upper.b
optim.Lk.FE.1 <- nlminb(objective=Lk.param, start=start.val,lower=lower.b,upper=upper.b
optim.Lk.RE.0 <- nlminb(objective=Lk.param, start=start.val,lower=lower.b,upper=upper.b
optim.Lk.RE.1 <- nlminb(objective=Lk.param, start=start.val,lower=lower.b,upper=upper.b
```

```
paramAK <- rbind(optim.Lk.FE.0$par,optim.Lk.FE.1$par,optim.Lk.RE.0$par,optim.Lk.RE.1$pa
colnames(paramAK) <- c("$p_1$","$\\theta_c$","$\\tau$")
rownames(paramAK) <- c("FE0","FE50","RE0","RE50")
knitr::kable(paramAK,digits=2,caption='Parameter estimates of Andrews and Kasy selectio
```

Table 13.1 shows the parameter estimates of the model for various data configurations (no treatment effect heterogeneity vs treatment effect heterogeneity and 0%

or 50% of non significant observations published). The results do not look great. The estimates of p_1 are correct when no non significant effects are published, by they are not nearly large enough when 50% of insignificant observations are published. The estimates of θ_c are completely crazy: all negative and large in asbolute value while the true value of θ_c is $\theta_c = 0.2$. The estimates of τ are also all misleading. For fixed effects, the estimates should be zero. For random effects, the true τ is $\tau = 0.5$. The estimates are much too large, apart from the third one that is close to home. Overall, barring a coding error, selection models do not look super promising here.

13.2.2.6 Fukumura

13.2.2.7 Trim and fill

13.2.3 Getting rid of publication bias: registered reports and pre-analysis plans

13.2.4 Detection of and correction for site selection bias

13.2.5 Vote counting and publication bias

13.2.6 The value of a statistically significant result

Publication bias and random effects

Chapter 14

Bounds

Chapter 15

Mediation Analysis

When we have estimated the treatment effect of a program, we sometimes wonder by which channels the program impact has been obtained. For example, has a Job Training Program been successful because it has increases the human capital of an agent, or simply by signalling to employers her motivation? The question of separating between the various channels into which a program impact can be decomposed becomes especially important when a program has several components, and we wish to ascertain which one is the more important. Another reason why we might be interested in which channel precisely is responsible for the program impact is because which channel dominates might give us indications about which theoretical mechanism is at play.

In this chapter, I am going to first delineate the general framework for mediation analysis and the way mediation analysis can be undertaken in the ideal case of a Randomized Controlled Trial. Then, I am going to present the fundamental problem of mediation analysis (which turns out to be one version of the confounders problem we know all too well) and the various techniques that have been developed in order to solve for it.

Appendix A

Proofs

A.1 Proofs of results in Chapter 2

A.1.1 Proof of Theorem 2.3

In order to use Theorem 2.2 for studying the behavior of $\Delta^{\hat{Y}}_{WW}$, we have to prove that it is unbiased and we have to compute $\mathbb{V}[\Delta^{\hat{Y}}_{WW}]$. Let's first prove that the WW estimator is an unbiased estimator of TT:

Lemma A.1 (Unbiasedness of $\Delta^{\hat{Y}}_{WW}$)**.** *Under Assumptions 1.7, 2.3 and 2.4,*

$$\mathbb{E}[\Delta^{\hat{Y}}_{WW}] = \Delta^{Y}_{TT}.$$

Proof. In order to prove Lemma A.1, we are going to use a trick. We are going to compute the expectation of the WW estimator conditional on a given treatment allocation. Because the resulting estimate is independent of treatment allocation, we will have our proof. This trick simplifies derivations a lot and is really natural: think first of all the samples with the same treatment allocation, then average your results over all possible treatment allocations.

$$
\begin{aligned}
\mathbb{E}[\Delta^{\hat{Y}}_{WW}] &= \mathbb{E}[\mathbb{E}[\Delta^{\hat{Y}}_{WW}|\mathbf{D}]] \\
&= \mathbb{E}[\mathbb{E}[\frac{1}{\sum_{i=1}^{N} D_i}\sum_{i=1}^{N} Y_i D_i - \frac{1}{\sum_{i=1}^{N}(1-D_i)}\sum_{i=1}^{N} Y_i(1-D_i)|\mathbf{D}]] \\
&= \mathbb{E}[\mathbb{E}[\frac{1}{\sum_{i=1}^{N} D_i}\sum_{i=1}^{N} Y_i D_i|\mathbf{D}] - \mathbb{E}[\frac{1}{\sum_{i=1}^{N}(1-D_i)}\sum_{i=1}^{N} Y_i(1-D_i)|\mathbf{D}]] \\
&= \mathbb{E}[\frac{1}{\sum_{i=1}^{N} D_i}\mathbb{E}[\sum_{i=1}^{N} Y_i D_i|\mathbf{D}] - \frac{1}{\sum_{i=1}^{N}(1-D_i)}\mathbb{E}[\sum_{i=1}^{N} Y_i(1-D_i)|\mathbf{D}]] \\
&= \mathbb{E}[\frac{1}{\sum_{i=1}^{N} D_i}\sum_{i=1}^{N}\mathbb{E}[Y_i D_i|\mathbf{D}] - \frac{1}{\sum_{i=1}^{N}(1-D_i)}\sum_{i=1}^{N}\mathbb{E}[Y_i(1-D_i)|\mathbf{D}]] \\
&= \mathbb{E}[\frac{1}{\sum_{i=1}^{N} D_i}\sum_{i=1}^{N}\mathbb{E}[Y_i D_i|D_i] - \frac{1}{\sum_{i=1}^{N}(1-D_i)}\sum_{i=1}^{N}\mathbb{E}[Y_i(1-D_i)|D_i]] \\
&= \mathbb{E}[\frac{1}{\sum_{i=1}^{N} D_i}\sum_{i=1}^{N} D_i\mathbb{E}[Y_i|D_i=1] - \frac{1}{\sum_{i=1}^{N}(1-D_i)}\sum_{i=1}^{N}(1-D_i)\mathbb{E}[Y_i|D_i=0]] \\
&= \mathbb{E}[\frac{\sum_{i=1}^{N} D_i}{\sum_{i=1}^{N} D_i}\mathbb{E}[Y_i|D_i=1] - \frac{\sum_{i=1}^{N}(1-D_i)}{\sum_{i=1}^{N}(1-D_i)}\mathbb{E}[Y_i|D_i=0]] \\
&= \mathbb{E}[\mathbb{E}[Y_i|D_i=1] - \mathbb{E}[Y_i|D_i=0]] \\
&= \mathbb{E}[Y_i|D_i=1] - \mathbb{E}[Y_i|D_i=0] \\
&= \Delta^{Y}_{TT}.
\end{aligned}
$$

The first equality uses the Law of Iterated Expectations (LIE). The second and fourth equalities use the linearity of conditional expectations. The third equality uses the fact that, conditional on $\mathbf{D}$, the number of treated and untreated is a constant. The fifth equality uses Assumption 2.4. The sixth equality uses the fact that $\mathbb{E}[Y_i D_i|D_i] = D_i\mathbb{E}[Y_i * 1|D_i = 1] + (1 - D_i)\mathbb{E}[Y_i * 0|D_i = 0]$. The seventh and ninth equalities use the fact that $\mathbb{E}[Y_i|D_i = 1]$ is a constant. The last equality uses Assumption 1.7. □

Let's now compute the variance of the WW estimator:

Lemma A.2 (Variance of $\Delta^{\hat{Y}}_{WW}$)**.** *Under Assumptions 1.7, 2.3 and 2.4,*

$$
\mathbb{V}[\Delta^{\hat{Y}}_{WW}] = \frac{1-(1-\Pr(D_i=1))^N}{N\Pr(D_i=1)}\mathbb{V}[Y_i^1|D_i=1] + \frac{1-\Pr(D_i=1)^N}{N(1-\Pr(D_i=1))}\mathbb{V}[Y_i^0|D_i=0].
$$

Proof. Same trick as before, but now using the Law of Total Variance (LTV):

$$
\begin{aligned}
\mathbb{V}[\Delta^{\hat{Y}}_{WW}] &= \mathbb{E}[\mathbb{V}[\Delta^{\hat{Y}}_{WW}|\mathbf{D}]] + \mathbb{V}[\mathbb{E}[\Delta^{\hat{Y}}_{WW}|\mathbf{D}]] \\
&= \mathbb{E}[\mathbb{V}[\frac{1}{\sum_{i=1}^{N} D_i}\sum_{i=1}^{N} Y_i D_i - \frac{1}{\sum_{i=1}^{N}(1-D_i)}\sum_{i=1}^{N} Y_i(1-D_i)|\mathbf{D}]] \\
&= \mathbb{E}[\mathbb{V}[\frac{1}{\sum_{i=1}^{N} D_i}\sum_{i=1}^{N} Y_i D_i|\mathbf{D}]] + \mathbb{E}[\mathbb{V}[\frac{1}{\sum_{i=1}^{N}(1-D_i)}\sum_{i=1}^{N} Y_i(1-D_i)|\mathbf{D}]] \\
&\quad + \mathbb{E}[\mathbb{C}[\frac{1}{\sum_{i=1}^{N} D_i}\sum_{i=1}^{N} Y_i D_i, \frac{1}{\sum_{i=1}^{N}(1-D_i)}\sum_{i=1}^{N} Y_i(1-D_i)|\mathbf{D}]] \\
&= \mathbb{E}[\frac{1}{(\sum_{i=1}^{N} D_i)^2}\mathbb{V}[\sum_{i=1}^{N} Y_i D_i|\mathbf{D}]] + \mathbb{E}[\frac{1}{(\sum_{i=1}^{N}(1-D_i))^2}\mathbb{V}[\sum_{i=1}^{N} Y_i(1-D_i)|\mathbf{D}]] \\
&= \mathbb{E}[\frac{1}{(\sum_{i=1}^{N} D_i)^2}\mathbb{V}[\sum_{i=1}^{N} Y_i D_i|D_i]] + \mathbb{E}[\frac{1}{(\sum_{i=1}^{N}(1-D_i))^2}\mathbb{V}[\sum_{i=1}^{N} Y_i(1-D_i)|D_i]] \\
&= \mathbb{E}[\frac{1}{(\sum_{i=1}^{N} D_i)^2}\sum_{i=1}^{N} D_i\mathbb{V}[Y_i|D_i=1]] + \mathbb{E}[\frac{1}{(\sum_{i=1}^{N}(1-D_i))^2}\sum_{i=1}^{N}(1-D_i)\mathbb{V}[Y_i|D_i=0]] \\
&= \mathbb{V}[Y_i|D_i=1]\mathbb{E}[\frac{1}{\sum_{i=1}^{N} D_i}] + \mathbb{V}[Y_i|D_i=0]\mathbb{E}[\frac{1}{\sum_{i=1}^{N}(1-D_i)}] \\
&= \frac{1-(1-\Pr(D_i=1))^N}{N\Pr(D_i=1)}\mathbb{V}[Y_i^1|D_i=1] + \frac{1-\Pr(D_i=1)^N}{N(1-\Pr(D_i=1))}\mathbb{V}[Y_i^0|D_i=0].
\end{aligned}
$$

The first equality stems from the LTV. The second and third equalities stems from the definition of the WW estimator and of the variance of a sum of random variables. The fourth equality stems from Assumption 2.4, which means that the covariance across observations is zero, and from the formula for a variance of a random variable multiplied by a constant. The fifth and sixth equalities stems from Assumption 2.4 and from $\mathbb{V}[Y_i D_i|D_i] = D_i\mathbb{V}[Y_i * 1|D_i = 1] + (1-D_i)\mathbb{V}[Y_i * 0|D_i = 0]$. The seventh equality stems from $\mathbb{V}[Y_i|D_i=1]$ and $\mathbb{V}[Y_i|D_i=0]$ being constant. The last equality stems from the formula for the expectation of the inverse of a sum of Bernoulli random variables with at least one of them taking value one which is the case under Assumption 2.3. □

Using Theorem 2.2, we have:

$$
\begin{aligned}
2\epsilon &\leq 2\sqrt{\frac{1}{N(1-\delta)}\left(\frac{1-(1-\Pr(D_i=1))^N}{\Pr(D_i=1)}\mathbb{V}[Y_i^1|D_i=1] + \frac{1-\Pr(D_i=1)^N}{(1-\Pr(D_i=1))}\mathbb{V}[Y_i^0|D_i=0]\right)} \\
&\leq 2\sqrt{\frac{1}{N(1-\delta)}\left(\frac{\mathbb{V}[Y_i^1|D_i=1]}{\Pr(D_i=1)} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{(1-\Pr(D_i=1))}\right)},
\end{aligned}
$$

where the second equality stems from the fact that $\frac{(1-\Pr(D_i=1))^N}{\Pr(D_i=1)}\mathbb{V}[Y_i^1|D_i = 1] + \frac{\Pr(D_i=1)^N}{(1-\Pr(D_i=1))}\mathbb{V}[Y_i^0|D_i = 0] \geq 0$. This proves the result.

A.1.2 Proof of Theorem 2.5

Before proving Theorem 2.5, let me state a very useful result: $\hat{WW}$ can be computed using OLS:

Lemma A.3 (WW is OLS)**.** *Under Assumption 2.3, the OLS coefficient β in the following regression:*

$$Y_i = \alpha + \beta D_i + U_i$$

is the WW estimator:

$$\begin{aligned}\hat{\beta}_{OLS} &= \frac{\frac{1}{N}\sum_{i=1}^N \left(Y_i - \frac{1}{N}\sum_{i=1}^N Y_i\right)\left(D_i - \frac{1}{N}\sum_{i=1}^N D_i\right)}{\frac{1}{N}\sum_{i=1}^N \left(D_i - \frac{1}{N}\sum_{i=1}^N D_i\right)^2} \\ &= \Delta_{WW}^{\hat{Y}}.\end{aligned}$$

Proof. In matrix notation, we have:

$$\underbrace{\begin{pmatrix} Y_1 \\ \vdots \\ Y_N \end{pmatrix}}_{Y} = \underbrace{\begin{pmatrix} 1 & D_1 \\ \vdots & \vdots \\ 1 & D_N \end{pmatrix}}_{X} \underbrace{\begin{pmatrix} \alpha \\ \beta \end{pmatrix}}_{\Theta} + \underbrace{\begin{pmatrix} U_1 \\ \vdots \\ U_N \end{pmatrix}}_{U}$$

The OLS estimator is:

$$\hat{\Theta}_{OLS} = (X'X)^{-1}X'Y$$

Under the Full Rank Assumption, $X'X$ is invertible and we have:

$$\begin{aligned}(X'X)^{-1} &= \begin{pmatrix} N & \sum_{i=1}^N D_i \\ \sum_{i=1}^N D_i & \sum_{i=1}^N D_i^2 \end{pmatrix}^{-1} \\ &= \frac{1}{N\sum_{i=1}^N D_i^2 - \left(\sum_{i=1}^N D_i\right)^2}\begin{pmatrix} \sum_{i=1}^N D_i^2 & -\sum_{i=1}^N D_i \\ -\sum_{i=1}^N D_i & N \end{pmatrix}\end{aligned}$$

For simplicity, I omit the summation index:

$$\begin{aligned}\hat{\Theta}_{OLS} &= \frac{1}{N\sum D_i^2 - \left(\sum D_i\right)^2}\begin{pmatrix} \sum D_i^2 & -\sum D_i \\ -\sum D_i & N \end{pmatrix}\begin{pmatrix} \sum Y_i \\ \sum Y_i D_i \end{pmatrix} \\ &= \frac{1}{N\sum D_i^2 - \left(\sum D_i\right)^2}\begin{pmatrix} \sum D_i^2 \sum Y_i - \sum D_i \sum_{i=1}^{N} Y_i D_i \\ -\sum D_i \sum Y_i + N\sum Y_i D_i \end{pmatrix}\end{aligned}$$

Using $D_i^2 = D_i$, we have:

$$\begin{aligned}\hat{\Theta}_{OLS} &= \begin{pmatrix} \frac{\left(\sum D_i\right)\left(\sum Y_i - \sum Y_i D_i\right)}{\left(\sum D_i\right)\left(N - \sum D_i\right)} \\ \frac{N\sum Y_i D_i - \sum D_i \sum Y_i}{N\sum D_i - \left(\sum D_i\right)^2} \end{pmatrix} = \begin{pmatrix} \frac{\sum (Y_i D_i + Y_i(1-D_i)) - \sum Y_i D_i}{\sum (1-D_i)} \\ \frac{N^2}{N^2}\frac{\frac{1}{N}\sum Y_i D_i - \frac{1}{N}\sum D_i \frac{1}{N}\sum Y_i + \frac{1}{N}\sum D_i \frac{1}{N}\sum Y_i - \frac{1}{N}\sum D_i \frac{1}{N}\sum Y_i}{\frac{1}{N}\sum D_i - 2\left(\frac{1}{N}\sum D_i\right)^2 + \left(\frac{1}{N}\sum D_i\right)^2} \end{pmatrix} \\ &= \begin{pmatrix} \frac{\sum Y_i(1-D_i)}{\sum (1-D_i)} \\ \frac{\frac{1}{N}\sum\left(Y_i D_i - D_i \frac{1}{N}\sum Y_i - Y_i \frac{1}{N}\sum D_i + \frac{1}{N}\sum D_i \frac{1}{N}\sum Y_i\right)}{\frac{1}{N}\sum\left(D_i - 2D_i \frac{1}{N}\sum D_i + \left(\frac{1}{N}\sum D_i\right)^2\right)} \end{pmatrix} = \begin{pmatrix} \frac{\sum Y_i(1-D_i)}{\sum (1-D_i)} \\ \frac{\frac{1}{N}\sum\left(Y_i - \frac{1}{N}\sum Y_i\right)\left(D_i - \frac{1}{N}\sum D_i\right)}{\frac{1}{N}\sum\left(D_i - \frac{1}{N}\sum D_i\right)^2} \end{pmatrix},\end{aligned}$$

which proves the first part of the lemma. Now for the second part of the lemma:

$$\begin{aligned}\hat{\beta}_{OLS} &= \frac{\sum Y_i D_i - \frac{1}{N}\sum D_i \sum Y_i}{\sum D_i\left(1 - \frac{1}{N}\sum D_i\right)} = \frac{\sum Y_i D_i - \frac{1}{N}\sum D_i \sum (Y_i D_i + (1-D_i)Y_i)}{\sum D_i\left(1 - \frac{1}{N}\sum D_i\right)} \\ &= \frac{\sum Y_i D_i\left(1 - \frac{1}{N}\sum D_i\right) - \frac{1}{N}\sum D_i \sum (1-D_i)Y_i}{\sum D_i\left(1 - \frac{1}{N}\sum D_i\right)} \\ &= \frac{\sum Y_i D_i}{\sum D_i} - \frac{\frac{1}{N}\sum (1-D_i)Y_i}{\left(1 - \frac{1}{N}\sum D_i\right)} \\ &= \frac{\sum Y_i D_i}{\sum D_i} - \frac{\frac{1}{N}\sum (1-D_i)Y_i}{\frac{1}{N}\sum (1-D_i)} \\ &= \frac{\sum Y_i D_i}{\sum D_i} - \frac{\sum (1-D_i)Y_i}{\sum (1-D_i)} \\ &= \Delta_{WW}^{\hat{Y}},\end{aligned}$$

which proves the result. □

Now, let me state the most important lemma behind the result in Theorem 2.5:

Lemma A.4 (Asymptotic Distribution of the OLS Estimator)**.** *Under Assumptions 1.7, 2.3, 2.4 and 2.5, we have:*

$$\sqrt{N}(\hat{\Theta}_{OLS} - \Theta) \xrightarrow{d} \mathcal{N}\left(\begin{array}{c} 0 \\ 0 \end{array}, \sigma_{XX}^{-1} \mathbf{V_{xu}} \sigma_{XX}^{-1} \right),$$

with

$$\sigma_{XX}^{-1} = \left(\begin{array}{cc} \frac{\Pr(D_i=1)}{\Pr(D_i=1)(1-\Pr(D_i=1))} & -\frac{\Pr(D_i=1)}{\Pr(D_i=1)(1-\Pr(D_i=1))} \\ -\frac{\Pr(D_i=1)}{\Pr(D_i=1)(1-\Pr(D_i=1))} & \frac{1}{\Pr(D_i=1)(1-\Pr(D_i=1))} \end{array} \right)$$

$$\mathbf{V_{xu}} = \mathbb{E}[U_i^2 \left(\begin{array}{cc} 1 & D_i \\ D_i & D_i \end{array} \right)]$$

Proof.

$$\begin{aligned} \sqrt{N}(\hat{\Theta}_{OLS} - \Theta) &= \sqrt{N}((X'X)^{-1}X'Y - \Theta) \\ &= \sqrt{N}((X'X)^{-1}X'(X\Theta + U) - \Theta) \\ &= \sqrt{N}((X'X)^{-1}X'X\Theta + (X'X)^{-1}X'U) - \Theta) \\ &= \sqrt{N}(X'X)^{-1}X'U \\ &= N(X'X)^{-1}\frac{\sqrt{N}}{N}X'U \end{aligned}$$

Using Slutsky's Theorem, we can study both terms separately. Slutsky's Theorem states that if $Y_N \xrightarrow{d} y$ and $\text{plim}(X_N) = x$, then:

1. $X_N + Y_N \xrightarrow{d} x + y$
2. $X_N Y_N \xrightarrow{d} xy$
3. $\frac{Y_N}{X_N} \xrightarrow{d} \frac{x}{y}$ if $x \neq 0$

Using this theorem, we have:

$$\sqrt{N}(\hat{\Theta}_{OLS} - \Theta) \xrightarrow{d} \sigma_{XX}^{-1} xu,$$

Where σ_{XX}^{-1} is a matrix of constants and xu is a random variable.

Let's begin with $\frac{\sqrt{N}}{N}X'U \xrightarrow{d} xu$:

$$\frac{\sqrt{N}}{N}X'U = \sqrt{N}\left(\begin{array}{c} \frac{1}{N}\sum_N^{i=1} U_i \\ \frac{1}{N}\sum_N^{i=1} D_i U_i \end{array} \right)$$

In order to determine the asymptotic distribution of $\frac{\sqrt{N}}{N}X'U$, we are going to use the vector version of the CLT:

If X_i and Y_i are two i.i.d. random variables with finite first and second moments, we have:

$$\sqrt{N}\begin{pmatrix} \frac{1}{N}\sum_{i=1}^{N} X_i - \mathbb{E}[X_i] \\ \frac{1}{N}\sum_{i=1}^{N} Y_i - \mathbb{E}[Y_i] \end{pmatrix} \xrightarrow{d} \mathcal{N}\left(\begin{array}{c} 0 \\ 0 \end{array}, \mathbf{V} \right),$$

where $\mathbf{V}$ is the population covariance matrix of X_i and Y_i.

We know that, under Assumption 1.7, both random variables have mean zero:

$$\begin{aligned} \mathbb{E}[U_i] &= \mathbb{E}[U_i|D_i = 1]\Pr(D_i = 1) + \mathbb{E}[U_i|D_i = 0]\Pr(D_i = 0) = 0 \\ \mathbb{E}[U_i D_i] &= \mathbb{E}[U_i|D_i = 1]\Pr(D_i = 1) = 0 \end{aligned}$$

Their covariance matrix $\mathbf{V_{xu}}$ can be computed as follows:

$$\begin{aligned} \mathbf{V_{xu}} &= \mathbb{E}[\begin{pmatrix} U_i \\ UiD_i \end{pmatrix} \begin{pmatrix} U_i & UiD_i \end{pmatrix}] - \mathbb{E}[\begin{pmatrix} U_i \\ UiD_i \end{pmatrix}]\mathbb{E}[\begin{pmatrix} U_i & UiD_i \end{pmatrix}] \\ &= \mathbb{E}[\begin{pmatrix} U_i^2 & U_i^2 D_i \\ Ui^2 D_i & U_i^2 D_i^2 \end{pmatrix}] = \mathbb{E}[U_i^2 \begin{pmatrix} 1 & D_i \\ D_i & D_i^2 \end{pmatrix}] = \mathbb{E}[U_i^2 \begin{pmatrix} 1 & D_i \\ D_i & D_i \end{pmatrix}] \end{aligned}$$

Using the Vector CLT, we have that $\frac{\sqrt{N}}{N} X'U \xrightarrow{d} \mathcal{N}\left(\begin{array}{c} 0 \\ 0 \end{array}, \mathbf{V_{xu}} \right)$.

Let's show now that $\text{plim} N(X'X)^{-1} = \sigma_{XX}^{-1}$:

$$\begin{aligned} N(X'X)^{-1} &= \frac{N}{N\sum_{i=1}^{N} D_i - \left(\sum_{i=1}^{N} D_i\right)^2} \begin{pmatrix} \sum_{i=1}^{N} D_i & -\sum_{i=1}^{N} D_i \\ -\sum_{i=1}^{N} D_i & N \end{pmatrix} \\ &= \frac{1}{N} \frac{1}{\frac{1}{N}\sum_{i=1}^{N} D_i - \left(\frac{1}{N}\sum_{i=1}^{N} D_i\right)^2} \begin{pmatrix} \sum_{i=1}^{N} D_i & -\sum_{i=1}^{N} D_i \\ -\sum_{i=1}^{N} D_i & N \end{pmatrix} \\ &= \frac{1}{\frac{1}{N}\sum_{i=1}^{N} D_i - \left(\frac{1}{N}\sum_{i=1}^{N} D_i\right)^2} \begin{pmatrix} \frac{1}{N}\sum_{i=1}^{N} D_i & -\frac{1}{N}\sum_{i=1}^{N} D_i \\ -\frac{1}{N}\sum_{i=1}^{N} D_i & 1 \end{pmatrix} \\ \text{plim} N(X'X)^{-1} &= \frac{1}{\text{plim}\frac{1}{N}\sum_{i=1}^{N} D_i - \left(\text{plim}\frac{1}{N}\sum_{i=1}^{N} D_i\right)^2} \begin{pmatrix} \text{plim}\frac{1}{N}\sum_{i=1}^{N} D_i & -\text{plim}\frac{1}{N}\sum_{i=1}^{N} D_i \\ -\text{plim}\frac{1}{N}\sum_{i=1}^{N} D_i & 1 \end{pmatrix} \\ &= \frac{1}{\Pr(D_i = 1) - \Pr(D_i = 1)^2} \begin{pmatrix} \Pr(D_i = 1) & -\Pr(D_i = 1) \\ -\Pr(D_i = 1) & 1 \end{pmatrix} \\ &= \sigma_{XX}^{-1} \end{aligned}$$

The fourth equality uses Slutsky's Theorem. The fifth equality uses the Law of Large Numbers (LLN): if Y_i are i.i.d. variables with finite first and second moments, $\text{plim}_{N\to\infty} \frac{1}{N}\sum_{i=1}^{N} Y_i = \mathbb{E}[Y_i]$.

In order to complete the proof, we have to use the Delta Method Theorem. This theorem states that:

$$\sqrt{N}\begin{pmatrix} \bar{X}_N - \mathbb{E}[X_i] \\ \bar{Y}_N - \mathbb{E}[Y_i] \end{pmatrix} \xrightarrow{d} \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \mathbf{V}\right)$$
$$\Rightarrow \sqrt{N}(g(\bar{X}_N, \bar{Y}_N) - g(\mathbb{E}[X_i], \mathbb{E}[Y_i])) \xrightarrow{d} \mathcal{N}(0, G'\mathbf{V}G)$$

where $G(u) = \frac{\partial g(u)}{\partial u}$ and $G = G(\mathbb{E}[X_i], \mathbb{E}[Y_i])$.

In our case, $g(xu) = \sigma_{XX}^{-1} xu$, so $G(xu) = \sigma_{XX}^{-1}$. The results follows from that and from the symmetry of σ_{XX}^{-1}. □

A last lemma uses the previous result to derive the asymptotic distribution of $W\hat{W}$:

Lemma A.5 (Asymptotic Distribution of $W\hat{W}$)**.** *Under Assumptions 1.7, 2.3, 2.4 and 2.5, we have:*

$$\sqrt{N}(\Delta_{WW}^{\hat{Y}} - \Delta_{TT}^{Y}) \xrightarrow{d} \mathcal{N}\left(0, \frac{\mathbb{V}[Y_i^1|D_i=1]}{\Pr(D_i=1)} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{1-\Pr(D_i=1)}\right).$$

Proof. In order to derive the asymptotic distribution of WW, I use first Lemma A.3 which implies that the asymptotic distribution of WW is the same as that of $\hat{\beta}_{OLS}$. Now, from Lemma A.4, we know that $\sqrt{N}(\hat{\beta}_{OLS} - \beta) \xrightarrow{d} \mathcal{N}(0, \sigma_\beta^2)$, where σ_β^2 is the lower diagonal term of $\sigma_{XX}^{-1}\mathbf{V_{xu}}\sigma_{XX}^{-1}$. Using the convention $p = \Pr(D_i = 1)$, we have:

$$\begin{aligned}
\sigma_{XX}^{-1}\mathbf{V_{xu}}\sigma_{XX}^{-1} &= \begin{pmatrix} \frac{p}{p(1-p)} & -\frac{p}{p(1-p)} \\ -\frac{p}{p(1-p)} & \frac{1}{p(1-p)} \end{pmatrix} \mathbb{E}[U_i^2 \begin{pmatrix} 1 & D_i \\ D_i & D_i \end{pmatrix}] \begin{pmatrix} \frac{p}{p(1-p)} & -\frac{p}{p(1-p)} \\ -\frac{p}{p(1-p)} & \frac{1}{p(1-p)} \end{pmatrix} \\
&= \frac{1}{(p(1-p))^2} \begin{pmatrix} p\mathbb{E}[U_i^2] - p\mathbb{E}[U_i^2 D_i] & p\mathbb{E}[U_i^2 D_i] - p\mathbb{E}[U_i^2 D_i] \\ -p\mathbb{E}[U_i^2] + \mathbb{E}[U_i^2 D_i] & -p\mathbb{E}[U_i^2 D_i] + \mathbb{E}[U_i^2 D_i] \end{pmatrix} \begin{pmatrix} p & -p \\ -p & 1 \end{pmatrix} \\
&= \frac{1}{(p(1-p))^2} \begin{pmatrix} p^2(\mathbb{E}[U_i^2] - \mathbb{E}[U_i^2 D_i]) & p^2(\mathbb{E}[U_i^2 D_i] - \mathbb{E}[U_i^2]) \\ p^2(\mathbb{E}[U_i^2 D_i] - \mathbb{E}[U_i^2]) & p^2\mathbb{E}[U_i^2] + (1-2p)\mathbb{E}[U_i^2 D_i] \end{pmatrix}
\end{aligned}$$

The final result comes from the fact that:

$$\begin{aligned}\mathbb{E}[U_i^2] &= \mathbb{E}[U_i^2|D_i=1]p + (1-p)\mathbb{E}[U_i^2|D_i=0] \\ &= p\mathbb{V}[Y_i^1|D_i=1] + (1-p)\mathbb{V}[Y_i^0|D_i=0] \\ \mathbb{E}[U_i^2 D_i] &= \mathbb{E}[U_i^2|D_i=1]p \\ &= p\mathbb{V}[Y_i^1|D_i=1].\end{aligned}$$

As a consequence:

$$\begin{aligned}\sigma_\beta^2 &= \frac{1}{(p(1-p))^2}\left(\mathbb{V}[Y_i^1|D_i=1]p(p^2-2p+1) + p^2(1-p)\mathbb{V}[Y_i^0|D_i=0]\right) \\ &= \frac{1}{(p(1-p))^2}\left(\mathbb{V}[Y_i^1|D_i=1]p(1-p)^2 + p^2(1-p)\mathbb{V}[Y_i^0|D_i=0]\right) \\ &= \frac{\mathbb{V}[Y_i^1|D_i=1]}{p} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{1-p}.\end{aligned}$$

□

Using the previous lemma, we can now approximate the confidence level of $\hat{WW}$:

$$\begin{aligned}&\Pr(|\Delta_{WW}^{\hat{Y}} - \Delta_{TT}^{Y}| \leq \epsilon) = \Pr(-\epsilon \leq \Delta_{WW}^{\hat{Y}} - \Delta_{TT}^{Y} \leq \epsilon) \\ &= \Pr\left(-\frac{\epsilon}{\frac{1}{\sqrt{N}}\sqrt{\frac{\mathbb{V}[Y_i^1|D_i=1]}{\Pr(D_i=1)} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{1-\Pr(D_i=1)}}} \leq \frac{\Delta_{WW}^{\hat{Y}} - \Delta_{TT}^{Y}}{\frac{1}{\sqrt{N}}\sqrt{\frac{\mathbb{V}[Y_i^1|D_i=1]}{\Pr(D_i=1)} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{1-\Pr(D_i=1)}}} \leq \frac{\epsilon}{\frac{1}{\sqrt{N}}\sqrt{\frac{\mathbb{V}[Y_i^1|D_i=1]}{\Pr(D_i=1)} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{1-\Pr(D_i=1)}}}\right) \\ &\approx \Phi\left(\frac{\epsilon}{\frac{1}{\sqrt{N}}\sqrt{\frac{\mathbb{V}[Y_i^1|D_i=1]}{\Pr(D_i=1)} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{1-\Pr(D_i=1)}}}\right) - \Phi\left(-\frac{\epsilon}{\frac{1}{\sqrt{N}}\sqrt{\frac{\mathbb{V}[Y_i^1|D_i=1]}{\Pr(D_i=1)} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{1-\Pr(D_i=1)}}}\right) \\ &= \Phi\left(\frac{\epsilon}{\frac{1}{\sqrt{N}}\sqrt{\frac{\mathbb{V}[Y_i^1|D_i=1]}{\Pr(D_i=1)} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{1-\Pr(D_i=1)}}}\right) - 1 + \Phi\left(\frac{\epsilon}{\frac{1}{\sqrt{N}}\sqrt{\frac{\mathbb{V}[Y_i^1|D_i=1]}{\Pr(D_i=1)} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{1-\Pr(D_i=1)}}}\right) \\ &= 2\Phi\left(\frac{\epsilon}{\frac{1}{\sqrt{N}}\sqrt{\frac{\mathbb{V}[Y_i^1|D_i=1]}{\Pr(D_i=1)} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{1-\Pr(D_i=1)}}}\right) - 1.\end{aligned}$$

As a consequence,

$$\delta \approx 2\Phi\left(\frac{\epsilon}{\frac{1}{\sqrt{N}}\sqrt{\frac{\mathbb{V}[Y_i^1|D_i=1]}{\Pr(D_i=1)} + \frac{\mathbb{V}[Y_i^0|D_i=0]}{1-\Pr(D_i=1)}}}\right) - 1.$$

Hence the result.

A.2 Proofs of results in Chapter 3

A.2.1 Proof of Theorem 3.9

In order to prove the theorem, it is going to be very helpful to prove the following lemma:

Lemma A.6 (Unconfounded Types)**.** *Under Assumptions 3.9 and 3.10, the types T_i are independent of the allocation of the treatment:*

$$(Y_i^{1,1}, Y_i^{0,1}, Y_i^{0,0}, Y_i^{1,0}, T_i) \perp\!\!\!\perp R_i | E_i = 1.$$

Proof. Lemma 4.2 in Dawid (1979) shows that if $X \perp\!\!\!\perp Y|Z$ and U is a function of X then $U \perp\!\!\!\perp Y|Z$. The fact that T_i is a function of (D_i^1, D_i^0) proves the result. □

The four sets defined by T_i are a partition of the sample space. As a consequence, we have (ommitting the conditioning on $E_i = 1$ all along for simplicity):

$$\begin{aligned}
\mathbb{E}[Y_i|R_i = 1] = {} & \mathbb{E}[Y_i|T_i = a, R_i = 1] \Pr(T_i = a|R_i = 1) \\
& + \mathbb{E}[Y_i|T_i = c, R_i = 1] \Pr(T_i = c|R_i = 1) \\
& + \mathbb{E}[Y_i|T_i = d, R_i = 1] \Pr(T_i = d|R_i = 1) \\
& + \mathbb{E}[Y_i|T_i = n, R_i = 1] \Pr(T_i = n|R_i = 1) \\
\mathbb{E}[Y_i|R_i = 0] = {} & \mathbb{E}[Y_i|T_i = a, R_i = 0] \Pr(T_i = a|R_i = 0) \\
& + \mathbb{E}[Y_i|T_i = c, R_i = 0] \Pr(T_i = c|R_i = 0) \\
& + \mathbb{E}[Y_i|T_i = d, R_i = 0] \Pr(T_i = d|R_i = 0) \\
& + \mathbb{E}[Y_i|T_i = n, R_i = 0] \Pr(T_i = n|R_i = 0).
\end{aligned}$$

Let's look at all these terms in turn:

$$\begin{aligned}
\mathbb{E}[Y_i|T_i = a, R_i = 1] &= \mathbb{E}[Y_i^{1,1} D_i R_i + Y_i^{1,0} D_i(1 - R_i) + Y_i^{0,1}(1 - D_i)R_i + Y_i^{0,0}(1 - D_i)(1 - R_i)|T_i = a, R \\
&= \mathbb{E}[Y_i^{1,1}(D_i^1 R_i + D_i^0(1 - R_i))R_i + Y_i^{0,1}(1 - (D_i^1 R_i + D_i^0(1 - R_i)))R_i|T_i = a, R_i = 1 \\
&= \mathbb{E}[Y_i^{1,1} D_i^1 R_i^2 + Y_i^{0,1}(1 - D_i^1 R_i)R_i|D_i^1 = D_i^0 = 1, R_i = 1] \\
&= \mathbb{E}[Y_i^{1,1}|T_i = a, R_i = 1] \\
&= \mathbb{E}[Y_i^{1,1}|T_i = a],
\end{aligned}$$

where the first equality uses Assumption 3.9, the second equality uses the fact that $R_i = 1$ in the conditional expectation and Assumption 3.9, the third equality uses

the fact that $R_i = 1$, the fourth equality uses the fact that $T_i = a \Leftrightarrow D_i^1 = D_i^0 = 1$ and the last equality uses Lemma A.6.

Using a similar reasoning, we have:

$$\begin{aligned}
\mathbb{E}[Y_i|T_i = c, R_i = 1] &= \mathbb{E}[Y_i^{1,1}|T_i = c] \\
\mathbb{E}[Y_i|T_i = d, R_i = 1] &= \mathbb{E}[Y_i^{0,1}|T_i = d] \\
\mathbb{E}[Y_i|T_i = n, R_i = 1] &= \mathbb{E}[Y_i^{0,1}|T_i = n] \\
\mathbb{E}[Y_i|T_i = a, R_i = 0] &= \mathbb{E}[Y_i^{1,0}|T_i = c] \\
\mathbb{E}[Y_i|T_i = c, R_i = 0] &= \mathbb{E}[Y_i^{0,0}|T_i = c] \\
\mathbb{E}[Y_i|T_i = d, R_i = 0] &= \mathbb{E}[Y_i^{1,0}|T_i = d] \\
\mathbb{E}[Y_i|T_i = n, R_i = 0] &= \mathbb{E}[Y_i^{0,0}|T_i = n].
\end{aligned}$$

Also, Lemma A.6 implies that $\Pr(T_i = a|R_i) = \Pr(T_i = a)$, and the same is true for all other types. As a consequence, we have:

$$\begin{aligned}
\mathbb{E}[Y_i|R_i = 1] = {} & \mathbb{E}[Y_i^{1,1}|T_i = a] \Pr(T_i = a) \\
& + \mathbb{E}[Y_i^{1,1}|T_i = c] \Pr(T_i = c) \\
& + \mathbb{E}[Y_i^{0,1}|T_i = d] \Pr(T_i = d) \\
& + \mathbb{E}[Y_i^{0,1}|T_i = n] \Pr(T_i = n) \\
\mathbb{E}[Y_i|R_i = 0] = {} & \mathbb{E}[Y_i^{1,0}|T_i = a] \Pr(T_i = a) \\
& + \mathbb{E}[Y_i^{0,0}|T_i = c] \Pr(T_i = c) \\
& + \mathbb{E}[Y_i^{1,0}|T_i = d] \Pr(T_i = d) \\
& + \mathbb{E}[Y_i^{0,0}|T_i = n] \Pr(T_i = n).
\end{aligned}$$

And thus:

$$\begin{aligned}
\mathbb{E}[Y_i|R_i = 1] - \mathbb{E}[Y_i|R_i = 0] = {} & (\mathbb{E}[Y_i^{1,1}|T_i = a] - \mathbb{E}[Y_i^{1,0}|T_i = a]) \Pr(T_i = a) \\
& + (\mathbb{E}[Y_i^{1,1}|T_i = c] - \mathbb{E}[Y_i^{0,0}|T_i = c]) \Pr(T_i = c) \\
& - (\mathbb{E}[Y_i^{1,0}|T_i = d] - \mathbb{E}[Y_i^{0,1}|T_i = d]) \Pr(T_i = d) \\
& + (\mathbb{E}[Y_i^{0,1}|T_i = n] - \mathbb{E}[Y_i^{0,0}|T_i = n]) \Pr(T_i = n).
\end{aligned}$$

Using Assumption 3.11, we have:

$$\begin{aligned}\mathbb{E}[Y_i|R_i=1]-\mathbb{E}[Y_i|R_i=0] &= (\mathbb{E}[Y_i^1|T_i=a]-\mathbb{E}[Y_i^1|T_i=a])\Pr(T_i=a)\\ &\quad+(\mathbb{E}[Y_i^1|T_i=c]-\mathbb{E}[Y_i^0|T_i=c])\Pr(T_i=c)\\ &\quad-(\mathbb{E}[Y_i^1|T_i=d]-\mathbb{E}[Y_i^0|T_i=d])\Pr(T_i=d)\\ &\quad+(\mathbb{E}[Y_i^0|T_i=n]-\mathbb{E}[Y_i^0|T_i=n])\Pr(T_i=n)\\ &= \mathbb{E}[Y_i^1-Y_i^0|T_i=c]\Pr(T_i=c)\\ &\quad-\mathbb{E}[Y_i^1-Y_i^0|T_i=d]\Pr(T_i=d).\end{aligned}$$

Under Assumption 3.13, we have:

$$\begin{aligned}\mathbb{E}[Y_i|R_i=1]-\mathbb{E}[Y_i|R_i=0] &= \mathbb{E}[Y_i^1-Y_i^0|T_i=c]\Pr(T_i=c)\\ &= \Delta^Y_{LATE}\Pr(T_i=c).\end{aligned}$$

We also have:

$$\begin{aligned}\Pr(D_i=1|R_i=1) &= \Pr(D_i^1=1|R_i=1)\\ &= \Pr(D_i^1=1\cap(D_i^0=1\cup D_i^0=0)|R_i=1)\\ &= \Pr(D_i^1=1\cap D_i^0=1\cup D_i^1=1\cap D_i^0=0|R_i=1)\\ &= \Pr(D_i^1=D_i^0=1\cup D_i^1-D_i^0=0|R_i=1)\\ &= \Pr(T_i=a\cup T_i=c|R_i=1)\\ &= \Pr(T_i=a|R_i=1)+\Pr(T_i=c|R_i=1)\\ &= \Pr(T_i=a)+\Pr(T_i=c),\end{aligned}$$

where the first equality follows from Assumption 3.9 and the fact that $D_i = R_iD_i^1 + (1-R_i)D_i^0$, so that $D_i|R_i=1 = D_i^1$. The second equality follows from the fact that $\left\{D_i^0=1, D_i^0=0\right\}$ is a partition of the sample space. The third equality follows from usual rules of logic and the fourth equality from the fact that D_i^1 and D_i^0 can only take values zero and one. The fifth equality follows from the definition of T_i. The sixth equaity follows from the rule of addition in probability and the fact that $T_i=a$ and $T_i=c$ are disjoint. The final equality follows from Lemma A.6.

Using a similar reasoning, we have:

$$\Pr(D_i=1|R_i=0)=\Pr(T_i=a)+\Pr(T_i=d).$$

As a consequence, under Assumption 3.13, we have:

$$\Pr(D_i = 1|R_i = 1) - \Pr(D_i = 1|R_i = 0) = \Pr(T_i = c).$$

Using Assumption 3.12 proves the result.

A.2.2 Proof of Theorem 3.15

In matrix notation, we have:

$$\underbrace{\begin{pmatrix} Y_1 \\ \vdots \\ Y_N \end{pmatrix}}_{Y} = \underbrace{\begin{pmatrix} 1 & D_1 \\ \vdots & \vdots \\ 1 & D_N \end{pmatrix}}_{X} \underbrace{\begin{pmatrix} \alpha \\ \beta \end{pmatrix}}_{\Theta} + \underbrace{\begin{pmatrix} U_1 \\ \vdots \\ U_N \end{pmatrix}}_{U}$$

and

$$\begin{pmatrix} D_1 \\ \vdots \\ D_N \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & R_1 \\ \vdots & \vdots \\ 1 & R_N \end{pmatrix}}_{R} \begin{pmatrix} \gamma \\ \tau \end{pmatrix} + \begin{pmatrix} V_1 \\ \vdots \\ V_N \end{pmatrix}$$

The IV estimator is:

$$\hat{\Theta}_{IV} = (R'X)^{-1}R'Y$$

If there is at least one observation with $R_i = 1$ and $D_i = 1$, $R'X$ is invertible (its determinant is non null) and we have (ommitting the summation index for simplicity):

$$\begin{aligned}(R'X)^{-1} &= \begin{pmatrix} N & \sum D_i \\ \sum R_i & \sum D_i R_i \end{pmatrix}^{-1} \\ &= \frac{1}{N\sum D_i R_i - \sum D_i \sum R_i} \begin{pmatrix} \sum D_i R_i & -\sum D_i \\ -\sum R_i & N \end{pmatrix}\end{aligned}$$

Since:

$$R'Y = \begin{pmatrix} \sum Y_i \\ \sum Y_i R_i \end{pmatrix},$$

we have:

$$\hat{\Theta}_{IV} = \begin{pmatrix} \frac{\sum Y_i \sum D_i R_i - \sum D_i \sum Y_i R_i}{N \sum D_i R_i - \sum D_i R_i} \\ \frac{N \sum Y_i R_i - \sum R_i \sum Y_i}{N \sum D_i R_i - \sum D_i R_i} \end{pmatrix}$$

As a consequence, $\hat{\beta}_{IV}$ is equal to the ratio of two OLS estimators (Y_i on R_i and a constant and D_i on the same regressors) (see the proof of Lemma A.3 in section A.1.2, just after "Using $D_i^2 = D_i$"). We can use Lemma A.3 stating that the OLS estimator is the WW estimator to prove the result.

A.2.3 Proof of Theorem 3.16

In order to derive the asymptotic distribution of the Wald estimator, I first use Theorem 3.15 which implies that the asymptotic distribution of Wald is the same as that of $\hat{\beta}_{IV}$. Now, I'm going to derive the asymptotic distribution of the IV estimator.

Lemma A.7 (Asymptotic Distribution of the IV Estimator)**.** *Under Independence and Validity of the Instrument, Exclusion Restriction and Full Rank, we have:*

$$\sqrt{N}(\hat{\Theta}_{IV} - \Theta) \xrightarrow{d} \mathcal{N}\left(\begin{array}{c} 0 \\ 0 \end{array}, (\sigma_{RX}^{-1})' \mathbf{V_{ru}} \sigma_{RX}^{-1} \right),$$

with

$$\sigma_{RX}^{-1} = \frac{\begin{pmatrix} \mathbb{E}[D_i R_i] & -\Pr(D_i = 1) \\ -\Pr(R_i = 1) & 1 \end{pmatrix}}{(\Pr(D_i = 1|R_i = 1) - \Pr(D_i = 1|R_i = 0)) \Pr(R_i = 1)(1 - \Pr(R_i = 1))}$$

$$\mathbf{V_{ru}} = \mathbb{E}[U_i^2 \begin{pmatrix} 1 & R_i \\ R_i & R_i \end{pmatrix}]$$

Proof.

$$\begin{aligned} \sqrt{N}(\hat{\Theta}_{IV} - \Theta) &= \sqrt{N}((R'X)^{-1}R'Y - \Theta) \\ &= \sqrt{N}((R'X)^{-1}R'(X\Theta + U) - \Theta) \\ &= \sqrt{N}((R'X)^{-1}R'X\Theta + (X'X)^{-1}X'U) - \Theta) \\ &= \sqrt{N}(R'X)^{-1}R'U \\ &= N(R'X)^{-1}\frac{\sqrt{N}}{N}R'U \end{aligned}$$

Using Slutsky's Theorem, we have:

$$\sqrt{N}(\hat{\Theta}_{IV} - \Theta) \xrightarrow{d} \sigma_{RX}^{-1} ru,$$

where σ_{RX}^{-1} is a matrix of constants and ru is a random variable.

We know that $\text{plim} N(R'X)^{-1} = \sigma_{RX}^{-1}$. So:

$$\begin{aligned} N(R'X)^{-1} &= \frac{N}{N\sum D_i R_i - \sum D_i \sum R_i} \begin{pmatrix} \sum D_i R_i & -\sum D_i \\ -\sum R_i & N \end{pmatrix} \\ &= \frac{1}{\frac{\sum D_i R_i}{N} - \frac{\sum D_i}{N}\frac{\sum R_i}{N}} \begin{pmatrix} \frac{\sum D_i R_i}{N} & -\frac{\sum D_i}{N} \\ -\frac{\sum R_i}{N} & 1 \end{pmatrix} \end{aligned}$$

$\frac{\sum D_i R_i}{N} - \frac{\sum D_i}{N}\frac{\sum R_i}{N}$ is equal to the numerator of the OLS coefficient of a regression of D_i on R_i and a constant (Proof of Lemma 3 in Lecture 0). As a consequence of Lemma 3 in Lecture 0, it can be written as the With/Without estimator multiplied by the denominator of the OLS estimator, which is simply the variance of R_i.

Let's turn to $\frac{\sqrt{N}}{N} R'U \xrightarrow{d} xu$:

$$\frac{\sqrt{N}}{N} R'U = \sqrt{N} \begin{pmatrix} \frac{1}{N}\sum_N^{i=1} U_i \\ \frac{1}{N}\sum_N^{i=1} R_i U_i \end{pmatrix}$$

We know that, under Validity of Randomization, both random variables have mean zero:

$$\begin{aligned} \mathbb{E}[U_i] &= \mathbb{E}[U_i|R_i = 1]\Pr(R_i = 1) + \mathbb{E}[U_i|R_i = 0]\Pr(R_i = 0) = 0 \\ \mathbb{E}[U_i R_i] &= \mathbb{E}[U_i|R_i = 1]\Pr(R_i = 1) = 0 \end{aligned}$$

Their covariance matrix $\mathbf{V_{ru}}$ can be computed as follows:

$$\begin{aligned} \mathbf{V_{ru}} &= \mathbb{E}[\begin{pmatrix} U_i \\ UiR_i \end{pmatrix} \begin{pmatrix} U_i & UiR_i \end{pmatrix}] - \mathbb{E}[\begin{pmatrix} U_i \\ UiR_i \end{pmatrix}]\mathbb{E}[\begin{pmatrix} U_i & UiR_i \end{pmatrix}] \\ &= \mathbb{E}[\begin{pmatrix} U_i^2 & U_i^2 R_i \\ Ui^2 R_i & U_i^2 R_i^2 \end{pmatrix}] = \mathbb{E}[U_i^2 \begin{pmatrix} 1 & R_i \\ R_i & R_i^2 \end{pmatrix}] = \mathbb{E}[U_i^2 \begin{pmatrix} 1 & R_i \\ R_i & R_i \end{pmatrix}] \end{aligned}$$

Using the Vector CLT, we have that $\frac{\sqrt{N}}{N} R'U \xrightarrow{d} \mathcal{N}\left(\begin{array}{c} 0 \\ 0 \end{array}, \mathbf{V_{ru}} \right)$. Using Slutsky's theorem and the LLN gives the result. □

From Lemma A.7, we know that $\sqrt{N}(\hat{\beta}_{IV} - \beta) \xrightarrow{d} \mathcal{N}(0, \sigma^2_\beta)$, where σ^2_β is the lower diagonal term of $(\sigma^{-1}_{RX})' \mathbf{V_{ru}} \sigma^{-1}_{RX}$. Using the convention $p^R = \Pr(R_i = 1)$, $p^D = \Pr(D_i = 1)$, $p^D_1 = \Pr(D_i = 1 | R_i = 1)$, $p^D_0 = \Pr(D_i = 1 | R_i = 0)$ and $p^{DR} = \mathbb{E}[D_i R_i]$, we have:

$$
\begin{aligned}
&(\sigma^{-1}_{RX})' \mathbf{V_{ru}} \sigma^{-1}_{RX} \\
&= \frac{1}{((p^D_1 - p^D_0) p^R (1 - p^R))^2} \begin{pmatrix} p^{DR} & -p^R \\ -p^D & 1 \end{pmatrix} \mathbb{E}[U_i^2 \begin{pmatrix} 1 & R_i \\ R_i & R_i \end{pmatrix}] \begin{pmatrix} p^{DR} & -p^D \\ -p^R & 1 \end{pmatrix} \\
&= \frac{1}{((p^D_1 - p^D_0) p^R (1 - p^R))^2} \begin{pmatrix} p^{DR}\mathbb{E}[U_i^2] - p^R \mathbb{E}[U_i^2 R_i] & \mathbb{E}[U_i^2 R_i](p^{DR} - p^R) \\ \mathbb{E}[U_i^2 R_i] - p^D \mathbb{E}[U_i^2] & \mathbb{E}[U_i^2 R_i](1 - p^D) \end{pmatrix} \begin{pmatrix} p^{DR} & -p^D \\ -p^R & 1 \end{pmatrix} \\
&= \frac{\begin{pmatrix} p^{DR}(p^{DR}\mathbb{E}[U_i^2] - p^R \mathbb{E}[U_i^2 R_i]) - p^R \mathbb{E}[U_i^2 R_i](p^{DR} - p^R) & \mathbb{E}[U_i^2 R_i](p^{DR} - p^R) - p^D(p^{DR}\mathbb{E}[U_i^2] - p \\ p^{DR}(\mathbb{E}[U_i^2 R_i] - p^D \mathbb{E}[U_i^2]) - p^R \mathbb{E}[U_i^2 R_i](1 - p^D) & \mathbb{E}[U_i^2 R_i](1 - p^D) - p^D(\mathbb{E}[U_i^2 R_i] - p^D \end{pmatrix}}{((p^D_1 - p^D_0) p^R (1 - p^R))^2}
\end{aligned}
$$

As a consequence:

$$
\begin{aligned}
\sigma^2_\beta &= \frac{\mathbb{E}[U_i^2 R_i](1 - p^D) - p^D(\mathbb{E}[U_i^2 R_i] - p^D \mathbb{E}[U_i^2])}{((p^D_1 - p^D_0) p^R (1 - p^R))^2} \\
&= \frac{(p^D)^2 \mathbb{E}[U_i^2] + (1 - 2p^D)\mathbb{E}[U_i^2 R_i]}{((p^D_1 - p^D_0) p^R (1 - p^R))^2} \\
&= \frac{(p^D)^2 (\mathbb{E}[U_i^2 | R_i = 1] p^R + \mathbb{E}[U_i^2 | R_i = 0](1 - p^R)) + (1 - 2p^D)\mathbb{E}[U_i^2 | R_i = 1] p^R}{((p^D_1 - p^D_0) p^R (1 - p^R))^2} \\
&= \frac{(p^D)^2 \mathbb{E}[U_i^2 | R_i = 0](1 - p^R) + (1 - 2p^D + (p^D)^2)\mathbb{E}[U_i^2 | R_i = 1] p^R}{((p^D_1 - p^D_0) p^R (1 - p^R))^2} \\
&= \frac{(p^D)^2 \mathbb{E}[U_i^2 | R_i = 0](1 - p^R) + (1 - p^D)^2 \mathbb{E}[U_i^2 | R_i = 1] p^R}{((p^D_1 - p^D_0) p^R (1 - p^R))^2} \\
&= \frac{1}{(p^D_1 - p^D_0)^2} \left[\left(\frac{p^D}{p^R} \right)^2 \frac{\mathbb{E}[U_i^2 | R_i = 0]}{1 - p^R} + \left(\frac{1 - p^D}{1 - p^R} \right)^2 \frac{\mathbb{E}[U_i^2 | R_i = 1]}{p^R} \right].
\end{aligned}
$$

Note that, under monotonicity, $p^C = p^D_1 - p^D_0$ and:

$$
\begin{aligned}
\mathbb{E}[U_i^2 | R_i = 1] &= p^{AT} \mathbb{V}[Y_i^1 | T_i = AT] + p^C \mathbb{V}[Y_i^1 | T_i = C] + p^{NT} \mathbb{V}[Y_i^0 | T_i = NT] \\
\mathbb{E}[U_i^2 | R_i = 0] &= p^{AT} \mathbb{V}[Y_i^1 | T_i = AT] + p^C \mathbb{V}[Y_i^0 | T_i = C] + p^{NT} \mathbb{V}[Y_i^0 | T_i = NT].
\end{aligned}
$$

The final result comes from the fact that:

$$\begin{aligned}
&\frac{1}{(p^C)^2}\left[\left(\frac{p^D}{p^R}\right)^2\frac{1}{1-p^R}+\left(\frac{1-p^D}{1-p^R}\right)^2\frac{1}{p^R}\right]\\
&\quad=\frac{(p^D)^2(1-p^R)+(1-p^D)^2p^R}{(p^Cp^R(1-p^R))^2}\\
&\quad=\frac{(p^D)^2-(p^D)^2p^R+p^R-2p^Dp^R+(p^D)^2p^R}{(p^Cp^R(1-p^R))^2}\\
&\quad=\frac{(p^D)^2+p^R-2p^Dp^R}{(p^Cp^R(1-p^R))^2}\\
&\quad=\frac{(p^D-p^R)^2+p^R-(p^R)^2}{(p^Cp^R(1-p^R))^2}\\
&\quad=\frac{(p^D-p^R)^2+p^R(1-p^R)}{(p^Cp^R(1-p^R))^2}\\
&\quad=\frac{(p^{AT}+p^Cp^R-p^R)^2+p^R(1-p^R)}{(p^Cp^R(1-p^R))^2}\\
&\quad=\frac{(p^{AT}+(1-p^{AT}-p^{NT})p^R-p^R)^2+p^R(1-p^R)}{(p^Cp^R(1-p^R))^2}\\
&\quad=\frac{(p^{AT}+(1-p^{AT}-p^{NT})p^R-p^R)^2+p^R(1-p^R)}{(p^Cp^R(1-p^R))^2}\\
&\quad=\frac{(p^{AT}+p^R-p^{AT}p^R-p^{NT}p^R-p^R)^2+p^R(1-p^R)}{(p^Cp^R(1-p^R))^2}\\
&\quad=\frac{(p^{AT}(1-p^R)-p^{NT}p^R)^2+p^R(1-p^R)}{(p^Cp^R(1-p^R))^2},
\end{aligned}$$

where the seventh equality uses the fact that $p^C+p^{AT}+p^{NT}=1$.

A.3 Proofs of results in Chapter 4

A.3.1 Proof of Theorem 4.5

Let us start with the proof that $\hat{\beta}^{FD}=\hat{\Delta}^{Y}_{DID}$. Using Lemma A.3, we have that $\hat{\beta}^{FD}=\hat{\Delta}^{Y_A-Y_B}_{WW}$. From there, since $\sum_{i=1}^{N}(Y_{i,A}-Y_{i,B})D_i=\sum_{i=1}^{N}Y_{i,A}D_i-\sum_{i=1}^{N}Y_{i,B}D_i$, we have $\hat{\beta}^{FD}=\hat{\Delta}^{Y}_{DID}$.

In order to prove the result for the OLS DID estimator, it is convenient to write the model in matrix form (where we rank all the observations from the first period in the first lines of each matrix and vector):

$$\underbrace{\begin{pmatrix} Y_{1,B} \\ \vdots \\ Y_{N,B} \\ Y_{1,A} \\ \vdots \\ Y_{N,A} \end{pmatrix}}_{Y} = \underbrace{\begin{pmatrix} 1 & D_1 & T_{1,B} & D_1T_{1,B} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & D_N & T_{N,B} & D_NT_{N,B} \\ 1 & D_1 & T_{1,A} & D_1T_{1,A} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & D_N & T_{N,A} & D_NT_{N,A} \end{pmatrix}}_{X} \underbrace{\begin{pmatrix} \alpha \\ \mu \\ \delta \\ \beta \end{pmatrix}}_{\Theta} + \underbrace{\begin{pmatrix} \epsilon_{1,B} \\ \vdots \\ \epsilon_{N,B} \\ \epsilon_{1,A} \\ \vdots \\ \epsilon_{N,A} \end{pmatrix}}_{\epsilon}$$

Now, using the fact that $T_{i,B} = 0$ and $T_{i,A} = 1$, $\forall i$, we can write matrix X as follows:

$$X = \begin{pmatrix} 1 & D_1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & D_N & 0 & 0 \\ 1 & D_1 & 1 & D_1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & D_N & 1 & D_N \end{pmatrix}$$

Doing some matrix multiplication and factoring N, we have:

$$X'X = N \underbrace{\begin{pmatrix} 2 & 2\bar{D} & 1 & \bar{D} \\ 2\bar{D} & 2\bar{D} & \bar{D} & \bar{D} \\ 1 & \bar{D} & 1 & \bar{D} \\ \bar{D} & \bar{D} & \bar{D} & \bar{D} \end{pmatrix}}_{x'x}$$

with $\bar{D} = \frac{1}{N}\sum_{i=1}^{N} D_i$, and using the fact that $D_i^2 = D_i$ since $D_i \in \{0, 1\}$. Using results on the inverse of a 4 by 4 matrix presented here and collecting terms patiently, we find that the determinant of xx is equal to:

$$\det(x'x) = \bar{D}^2(1 - \bar{D})^2$$

and its adjugate is equal to:

$$\tilde{x'x} = \bar{D}(1 - \bar{D}) \begin{pmatrix} \bar{D} & -\bar{D} & -\bar{D} & \bar{D} \\ -\bar{D} & 1 & \bar{D} & -1 \\ -\bar{D} & \bar{D} & 2\bar{D} & -2\bar{D} \\ \bar{D} & -1 & -2\bar{D} & 2 \end{pmatrix}$$

We also have that:

$$X'Y = N \begin{pmatrix} \bar{Y}_B + \bar{Y}_A \\ \bar{D}(\bar{Y}_B^1 + \bar{Y}_A^1) \\ \bar{Y}_A \\ \bar{D}\bar{Y}_A^1 \end{pmatrix}$$

with $\bar{Y}_t = \frac{1}{N}\sum_{i=1}^N Y_{i,t}$ and $\bar{Y}_t^1 = \frac{1}{\sum_{i=1}^N D_i}\sum_{i=1}^N D_i Y_{i,t}$ and $\bar{Y}_t^0 = \frac{1}{\sum_{i=1}^N (1-D_i)}\sum_{i=1}^N (1-D_i)Y_{i,t}$ and using the fact that $\sum_{i=1}^N D_i Y_{i,t} = N\bar{D}\bar{Y}_t^1$. Using the fact that $Y_{i,t} = D_i Y_{i,t} + (1-D_i)Y_{i,t}$, we have:

$$\begin{aligned} \bar{Y}_t &= \frac{\sum_{i=1}^N D_i}{N}\frac{\sum_{i=1}^N D_i Y_{i,t}}{\sum_{i=1}^N D_i} + \frac{\sum_{i=1}^N (1-D_i)}{N}\frac{\sum_{i=1}^N (1-D_i)Y_{i,t}}{\sum_{i=1}^N (1-D_i)} \\ &= \bar{D}\bar{Y}_t^1 + (1-\bar{D})\bar{Y}_t^0. \end{aligned}$$

We thus have:

$$X'Y = N \begin{pmatrix} \underbrace{\bar{Y}_B^0 + \bar{Y}_A^0 + \bar{D}(\bar{Y}_B^1 - \bar{Y}_B^0 + \bar{Y}_A^1 - \bar{Y}_A^0)}_{\mathbf{A}} \\ \underbrace{\bar{D}(\bar{Y}_B^1 + \bar{Y}_A^1)}_{\mathbf{B}} \\ \underbrace{\bar{Y}_A^0 + \bar{D}(\bar{Y}_A^1 - \bar{Y}_A^0)}_{\mathbf{C}} \\ \underbrace{\bar{D}\bar{Y}_A^1}_{\mathbf{D}} \end{pmatrix}$$

Using the fact that $(X'X)^{-1} = (Nx'x)^{-1} = \frac{1}{N}(x'x)^{-1} = \frac{1}{N}\frac{\tilde{x'x}}{\det(x'x)}$, we have:

$$\begin{aligned} \hat{\Theta}^{OLS} &= (X'X)^{-1}X'Y \\ &= \frac{1}{\bar{D}(1-\bar{D})} \begin{pmatrix} \bar{D}(\mathbf{A} - \mathbf{B} - \mathbf{C} + \mathbf{D}) \\ -\bar{D}\mathbf{A} + \mathbf{B} + \bar{D}\mathbf{C} - \mathbf{D} \\ \bar{D}(-\mathbf{A} + \mathbf{B} + 2\mathbf{C} - 2\mathbf{D}) \\ \bar{D}\mathbf{A} - \mathbf{B} - 2\bar{D}\mathbf{C} + 2\mathbf{D} \end{pmatrix} \end{aligned}$$

Let's take each term in turn:

$$\begin{aligned}\hat{\alpha}^{OLS} &= \frac{1}{1-\bar{D}}\left(\bar{Y}_B^0 + \bar{Y}_A^0 + \bar{D}(\bar{Y}_B^1 - \bar{Y}_B^0 + \bar{Y}_A^1 - \bar{Y}_A^0) - \bar{D}(\bar{Y}_B^1 + \bar{Y}_A^1) - (\bar{Y}_A^0 + \bar{D}(\bar{Y}_A^1 - \bar{Y}_A^0)) + \bar{D}\bar{Y}_A^1\right)\\ &= \frac{1}{1-\bar{D}}\left(\bar{Y}_B^0(1-\bar{D}) + \bar{Y}_A^0(1-\bar{D}-1+\bar{D}) + \bar{Y}_B^1(\bar{D}-\bar{D}) + \bar{Y}_A^1(\bar{D}-\bar{D}-\bar{D}+\bar{D})\right)\\ &= \bar{Y}_B^0\end{aligned}$$

$$\begin{aligned}\hat{\mu}^{OLS} &= \frac{1}{\bar{D}(1-\bar{D})}\left(-\bar{D}(\bar{Y}_B^0 + \bar{Y}_A^0 + \bar{D}(\bar{Y}_B^1 - \bar{Y}_B^0 + \bar{Y}_A^1 - \bar{Y}_A^0)) + \bar{D}(\bar{Y}_B^1 + \bar{Y}_A^1) + \bar{D}(\bar{Y}_A^0 + \bar{D}(\bar{Y}_A^1 - \bar{Y}_A^0)) - I\right.\\ &= \frac{1}{1-\bar{D}}\left(-\bar{Y}_B^0(1-\bar{D}) + \bar{Y}_A^0(-1+\bar{D}+1-\bar{D}) + \bar{Y}_B^1(1-\bar{D}) + \bar{Y}_A^1(-\bar{D}+1+\bar{D}-1)\right)\\ &= \bar{Y}_B^1 - \bar{Y}_B^0\end{aligned}$$

$$\begin{aligned}\hat{\delta}^{OLS} &= \frac{1}{1-\bar{D}}\left(-(\bar{Y}_B^0 + \bar{Y}_A^0 + \bar{D}(\bar{Y}_B^1 - \bar{Y}_B^0 + \bar{Y}_A^1 - \bar{Y}_A^0)) + (\bar{Y}_B^1 + \bar{Y}_A^1) + 2(\bar{Y}_A^0 + \bar{D}(\bar{Y}_A^1 - \bar{Y}_A^0)) - 2\bar{D}\bar{Y}_A^1\right)\\ &= \frac{1}{1-\bar{D}}\left(-\bar{Y}_B^0(1-\bar{D}) + \bar{Y}_A^0(2(1-\bar{D}) - (1-\bar{D})) + \bar{Y}_B^1(\bar{D}-\bar{D}) + \bar{Y}_A^1(\bar{D}-\bar{D}+2\bar{D}-2\bar{D})\right)\\ &= \bar{Y}_A^0 - \bar{Y}_B^0\end{aligned}$$

$$\begin{aligned}\hat{\beta}^{OLS} &= \frac{1}{\bar{D}(1-\bar{D})}\left(\bar{D}(\bar{Y}_B^0 + \bar{Y}_A^0 + \bar{D}(\bar{Y}_B^1 - \bar{Y}_B^0 + \bar{Y}_A^1 - \bar{Y}_A^0)) - \bar{D}(\bar{Y}_B^1 + \bar{Y}_A^1) - 2\bar{D}(\bar{Y}_A^0 + \bar{D}(\bar{Y}_A^1 - \bar{Y}_A^0)) + 2\right.\\ &= \frac{1}{1-\bar{D}}\left(\bar{Y}_B^0(1-\bar{D}) + \bar{Y}_A^0((1-\bar{D}) - 2(1-\bar{D})) + \bar{Y}_B^1(\bar{D}-1) + \bar{Y}_A^1(\bar{D}-1-2\bar{D}+2)\right)\\ &= \bar{Y}_A^1 - \bar{Y}_B^1 - (\bar{Y}_A^0 - \bar{Y}_B^0)\end{aligned}$$

This last results proves that $\hat{\beta}^{OLS} = \hat{\Delta}_{DID}^{Y}$.

For the within estimator, it can be written in matrix form as follows:

$$\underbrace{\begin{pmatrix} Y_{1,B} - \bar{Y}_1 \\ \vdots \\ Y_{N,B} - \bar{Y}_N \\ Y_{1,A} - \bar{Y}_1 \\ \vdots \\ Y_{N,A} - \bar{Y}_N \end{pmatrix}}_{Y^W} = \underbrace{\begin{pmatrix} 1 & 0 & -\bar{D}_1 \\ \vdots & \vdots & \vdots \\ 1 & 0 & -\bar{D}_N \\ 1 & 1 & D_1 - \bar{D}_1 \\ \vdots & \vdots & \vdots \\ 1 & 1 & D_N - \bar{D}_N \end{pmatrix}}_{X^W} \underbrace{\begin{pmatrix} \alpha^W \\ \delta^W \\ \beta^W \end{pmatrix}}_{\Theta^W} + \underbrace{\begin{pmatrix} \epsilon_{1,B}^W \\ \vdots \\ \epsilon_{N,B}^W \\ \epsilon_{1,A}^W \\ \vdots \\ \epsilon_{N,A}^W \end{pmatrix}}_{\epsilon^W}$$

We have:

$$X^{W\prime}X^{W} = N \underbrace{\begin{pmatrix} 2 & 1 & 0 \\ 1 & 1 & \frac{\bar{D}}{2} \\ 0 & \frac{\bar{D}}{2} & \frac{\bar{D}}{2} \end{pmatrix}}_{x^{W\prime}x^{W}}$$

This is because:

$$X^{W\prime}X^{W} = \begin{pmatrix} 2N & N & -\sum_{i=1}^{N}\bar{D}_i + \sum_{i=1}^{N}(D_i - \bar{D}_i) \\ N & N & \sum_{i=1}^{N}(D_i - \bar{D}_i) \\ -\sum_{i=1}^{N}\bar{D}_i + \sum_{i=1}^{N}(D_i - \bar{D}_i) & \sum_{i=1}^{N}(D_i - \bar{D}_i) & \sum_{i=1}^{N}\bar{D}_i^2 + \sum_{i=1}^{N}(D_i - \bar{D}_i)^2 \end{pmatrix}$$

and:

$$\begin{aligned} \sum_{i=1}^{N}\bar{D}_i &= \frac{1}{2}\sum_{i=1}^{N}(D_{i,B} + D_{i,A}) \\ &= \frac{1}{2}\sum_{i=1}^{N}D_i \\ &= \frac{1}{2}N\bar{D} \\ \sum_{i=1}^{N}(D_i - \bar{D}_i) &= N\bar{D} - \frac{1}{2}N\bar{D} \\ &= \frac{1}{2}N\bar{D} \\ \sum_{i=1}^{N}\bar{D}_i^2 &= \frac{1}{4}\sum_{i=1}^{N}(D_{i,B} + D_{i,A})^2 \\ &= \frac{1}{4}\sum_{i=1}^{N}D_i^2 \\ &= \frac{1}{4}N\bar{D} \\ \sum_{i=1}^{N}(D_i - \bar{D}_i)^2 &= \sum_{i=1}^{N}(D_i - \frac{1}{2}D_i)^2 \\ &= \frac{1}{4}N\bar{D} \end{aligned}$$

Now we can use the results here and here to compute the inverse of the $x^{W\prime}x^W$ matrix. Let us first compute the determinant:

$$\begin{aligned}\det(x^{W\prime}x^W) &= 2(\frac{\bar{D}}{2} - \frac{\bar{D}^2}{4}) - \frac{\bar{D}}{2} \\ &= \frac{1}{2}\bar{D}(1-\bar{D}).\end{aligned}$$

And then the adjugate:

$$x^{W\tilde{\prime}}x^W = \begin{pmatrix} \frac{\bar{D}}{2}(1-\frac{\bar{D}}{2}) & -\frac{\bar{D}}{2} & \frac{\bar{D}}{2} \\ -\frac{\bar{D}}{2} & \bar{D} & -\bar{D} \\ \frac{\bar{D}}{2} & -\bar{D} & 1 \end{pmatrix}$$

Let us now examine $X^{W\prime}Y^W$:

$$X^{W\prime}Y^W = \begin{pmatrix} \sum_{i=1}^N (Y_{i,B} - \bar{Y}_i) + \sum_{i=1}^N (Y_{i,A} - \bar{Y}_i) \\ \sum_{i=1}^N (Y_{i,A} - \bar{Y}_i) \\ -\sum_{i=1}^N \bar{D}_i(Y_{i,B} - \bar{Y}_i) + \sum_{i=1}^N (D_i - \bar{D}_i)(Y_{i,A} - \bar{Y}_i) \end{pmatrix}$$

We have:

$$\begin{aligned}
\sum_{i=1}^{N}(Y_{i,B}-\bar{Y}_i) &= N\bar{Y}_B - \frac{1}{2}N(\bar{Y}_B+\bar{Y}_A)\\
&= \frac{1}{2}N(\bar{Y}_B-\bar{Y}_A)\\
\sum_{i=1}^{N}(Y_{i,A}-\bar{Y}_i) &= \frac{1}{2}N(\bar{Y}_A-\bar{Y}_B)\\
\sum_{i=1}^{N}\bar{D}_i(Y_{i,B}-\bar{Y}_i) &= \sum_{i=1}^{N}\frac{1}{2}D_i(Y_{i,B}-\frac{1}{2}\sum_{i=1}^{N}(Y_{i,B}+Y_{i,A}))\\
&= \sum_{i=1}^{N}\frac{1}{2}D_i\frac{1}{2}(Y_{i,B}-Y_{i,A})\\
&= \frac{1}{4}\sum_{i=1}^{N}D_i(Y_{i,B}-Y_{i,A})\\
&= \frac{1}{4}N\bar{D}(\bar{Y}_B^1-\bar{Y}_A^1)\\
\sum_{i=1}^{N}(D_i-\bar{D}_i)(Y_{i,A}-\bar{Y}_i) &= \sum_{i=1}^{N}(D_i-\frac{1}{2}D_i)(Y_{i,A}-\frac{1}{2}\sum_{i=1}^{N}(Y_{i,B}+Y_{i,A}))\\
&= \frac{1}{4}\sum_{i=1}^{N}D_i(Y_{i,A}-Y_{i,B})\\
&= \frac{1}{4}N\bar{D}(\bar{Y}_A^1-\bar{Y}_B^1).
\end{aligned}$$

So, we have:

$$(X^{W\prime}X^W)^{-1}X^{W\prime}Y^W = \frac{2}{N\bar{D}(1-\bar{D})}\begin{pmatrix} \frac{\bar{D}}{2}(1-\frac{\bar{D}}{2}) & -\frac{\bar{D}}{2} & \frac{\bar{D}}{2} \\ -\frac{\bar{D}}{2} & \bar{D} & -\bar{D} \\ \frac{\bar{D}}{2} & -\bar{D} & 1 \end{pmatrix}\begin{pmatrix} 0 \\ \frac{N}{2}(\bar{Y}_A-\bar{Y}_B) \\ \frac{N}{2}\bar{D}(\bar{Y}_A^1-\bar{Y}_B^1) \end{pmatrix}$$

We thus have:

$$\begin{aligned}
\hat{\beta}^W &= \frac{2}{N\bar{D}(1-\bar{D})}\left(-\bar{D}\frac{N}{2}(\bar{Y}_A-\bar{Y}_B)+\frac{N}{2}\bar{D}(\bar{Y}_A^1-\bar{Y}_B^1)\right)\\
&= \frac{1}{1-\bar{D}}\left(\bar{Y}_A^1-\bar{Y}_B^1-(\bar{Y}_A-\bar{Y}_B)\right)
\end{aligned}$$

Using the fact that $\bar{Y}_t = \bar{D}\bar{Y}_t^1 + (1-\bar{D})\bar{Y}_t^0$, we have $\bar{Y}_A - \bar{Y}_B = (1-\bar{D})(\bar{Y}_A^0 - \bar{Y}_B^0) + \bar{D}(\bar{Y}_A^1 - \bar{Y}_B^1)$.

As a consequence:

$$\begin{aligned}\hat{\beta}^W &= \frac{1-\bar{D}}{1-\bar{D}}\left(\bar{Y}_A^1 - \bar{Y}_B^1 - (\bar{Y}_A^0 - \bar{Y}_B^0)\right) \\ &= \bar{Y}_A^1 - \bar{Y}_B^1 - (\bar{Y}_A^0 - \bar{Y}_B^0),\end{aligned}$$

which proves that $\hat{\beta}^W = \hat{\Delta}_{DID}^Y$.

Now for $\hat{\beta}^{LSDV}$, the estimator can be written in matrix form as follows:

$$\underbrace{\begin{pmatrix} Y_{1,B} \\ \vdots \\ Y_{N,B} \\ Y_{1,A} \\ \vdots \\ Y_{N,A} \end{pmatrix}}_{Y} = \underbrace{\begin{pmatrix} 1 & 0 & \dots & 0 & 1 & 0 & D_{1,B} \\ 0 & 1 & \dots & 0 & 1 & 0 & D_{2,B} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 1 & 0 & D_{N,B} \\ 1 & 0 & \dots & 0 & 0 & 1 & D_{1,A} \\ 0 & 1 & \dots & 0 & 0 & 1 & D_{2,A} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 & 1 & D_{N,A} \end{pmatrix}}_{X^{LSDV}} \underbrace{\begin{pmatrix} \mu_1^{LSDV} \\ \vdots \\ \mu_N^{LSDV} \\ \delta_B^{LSDV} \\ \delta_A^{LSDV} \\ \beta^{LSDV} \end{pmatrix}}_{\Theta^{LSDV}} + \underbrace{\begin{pmatrix} \epsilon_{1,B}^{LSDV} \\ \vdots \\ \epsilon_{N,B}^{LSDV} \\ \epsilon_{1,A}^{LSDV} \\ \vdots \\ \epsilon_{N,A}^{LSDV} \end{pmatrix}}_{\epsilon^{LSDV}}.$$

In order to prove the result, it is going to be very convenient to use Frish-Waugh-Lovell Theorem. It can be stated as follows:

Theorem A.1 (Frish-Waugh-Lovell)**.** *The coefficients on a set of variables X_2 estimated by OLS in a linear regression with another set of control variables X_1 is equal to the coefficients on the same set of variables estimated by OLS in a linear model where the outcome variable is the residual of regressing Y on X_1 by OLS and the explanatory variables are the residuals of regressing X_2 on X_1. More formally: $\hat{\beta}_2^{OLS} = \hat{\beta}_2^{OLS(MX_1)}$ where:*

$$\begin{aligned} Y &= X_1\beta_1 + X_2\beta_2 + \epsilon \\ M_1 Y &= M_1 X_2 \beta_2 + \epsilon^* \\ M_1 &= I - X_1(X_1'X_1)^{-1}X_1'. \end{aligned}$$

Proof. See Section 8.2.2 here. □

M_1 is called the **prediction** or the **residualizing** matrix.

In our case, let us call X_μ^{LSDV} the first N columns of X^{LSDV}. X_μ^{LSDV} is going to play the role of X_1 in Theorem A.1. Let us call $X_{\delta,D}^{LSDV}$ the matrix made

of the last three columns of X^{LSDV}. $X_{\delta,D}^{LSDV}$ is going to play the role of X_2 in Theorem A.1.

Let us first note that $X_\mu^{LSDV\prime} X_\mu^{LSDV} = 2I_N$, where I_N is the identity matrix of dimension N. As a consequence, $(X_\mu^{LSDV\prime} X_\mu^{LSDV})^{-1} = \frac{1}{2} I_N$. Now, let us compute $X_\mu^{LSDV\prime} Y$:

$$X_\mu^{LSDV\prime} Y = \begin{pmatrix} Y_{1,B} + Y_{1,A} \\ \vdots \\ Y_{N,B} + Y_{N,A} \end{pmatrix}.$$

As a consequence, we have:

$$\begin{aligned} M_\mu^{LSDV} Y &= Y - X_\mu^{LSDV} (X_\mu^{LSDV\prime} X_\mu^{LSDV})^{-1} X_\mu^{LSDV\prime} Y \\ &= Y - \frac{1}{2} X_\mu^{LSDV} I_N \begin{pmatrix} Y_{1,B} + Y_{1,A} \\ \vdots \\ Y_{N,B} + Y_{N,A} \end{pmatrix} \\ &= \begin{pmatrix} Y_{1,B} - \frac{1}{2}(Y_{1,B} + Y_{1,A}) \\ \vdots \\ Y_{N,B} - \frac{1}{2}(Y_{N,B} + Y_{N,A}) \\ Y_{1,A} - \frac{1}{2}(Y_{1,B} + Y_{1,A}) \\ \vdots \\ Y_{N,A} - \frac{1}{2}(Y_{N,B} + Y_{N,A}) \end{pmatrix}. \end{aligned}$$

And finally:

$$\begin{aligned} M_\mu^{LSDV} X_{\delta,D}^{LSDV} &= X_{\delta,D}^{LSDV} - X_\mu^{LSDV} (X_\mu^{LSDV\prime} X_\mu^{LSDV})^{-1} X_\mu^{LSDV\prime} X_{\delta,D}^{LSDV} \\ &= \begin{pmatrix} \frac{1}{2} & -\frac{1}{2} & D_{1,B} - \frac{1}{2}(D_{1,B} + D_{1,A}) \\ \vdots & \vdots & \vdots \\ \frac{1}{2} & -\frac{1}{2} & D_{N,B} - \frac{1}{2}(D_{1,B} + D_{1,A}) \\ -\frac{1}{2} & \frac{1}{2} & D_{1,A} - \frac{1}{2}(D_{1,B} + D_{1,A}) \\ \vdots & \vdots & \vdots \\ -\frac{1}{2} & \frac{1}{2} & D_{N,A} - \frac{1}{2}(D_{1,B} + D_{1,A}) \end{pmatrix}. \end{aligned}$$

Using Theorem A.1, we can rewrite the LSDV version of the TWFE model as follows:

$$M_\mu^{LSDV} Y = M_\mu^{LSDV} X_{\delta,D}^{LSDV} \begin{pmatrix} \delta_B^{LSDV} \\ \delta_A^{LSDV} \\ \beta^{LSDV} \end{pmatrix} + M_\mu^{LSDV} \epsilon^{LSDV}$$

In a more compact notation, we have, $\forall i \in [1, N]$ and $\forall t \in \{B, A\}$:

$$Y_{i,t} - \bar{Y}_i = \frac{1}{2}(\delta_A^{LSDV} - \delta_B^{LSDV})(\mathbb{1}[t = A] - \mathbb{1}[t = B]) + \beta^{LSDV}(D_{i,t} - \bar{D}_i) + \epsilon_{i,t}^{LSDV} - \bar{\epsilon}_i^{LSDV},$$

which we can rewrite, for simplicity, as:

$$Y_{i,t} - \bar{Y}_i = \tilde{\delta}_t^{LSDV} + \beta^{LSDV}(D_{i,t} - \bar{D}_i) + \epsilon_{i,t}^{LSDV} - \bar{\epsilon}_i^{LSDV},$$

with $\tilde{\delta}_A^{LSDV} = -\tilde{\delta}_B^{LSDV} = \bar{\delta}^{LSDV}$ and $\bar{\delta}^{LSDV} = \frac{1}{2}(\delta_A^{LSDV} - \delta_B^{LSDV})$.

In matrix form, we can thus rewrite the LSDV model transformed by the application of the Frich-Waugh theorem as follows:

$$\underbrace{\begin{pmatrix} Y_{1,B} - \bar{Y}_1 \\ \vdots \\ Y_{N,B} - \bar{Y}_N \\ Y_{1,A} - \bar{Y}_1 \\ \vdots \\ Y_{N,A} - \bar{Y}_N \end{pmatrix}}_{Y_r^{LSDV}} = \underbrace{\begin{pmatrix} 1 & 0 & -\bar{D}_1 \\ \vdots & \vdots & \vdots \\ 1 & 0 & -\bar{D}_N \\ 0 & 1 & D_1 - \bar{D}_1 \\ \vdots & \vdots & \vdots \\ 0 & 1 & D_N - \bar{D}_N \end{pmatrix}}_{X_r^{LSDV}} \underbrace{\begin{pmatrix} \tilde{\delta}_B^{LSDV} \\ \tilde{\delta}_A^{LSDV} \\ \beta^{LSDV} \end{pmatrix}}_{\Theta_r^{LSDV}} + \underbrace{\begin{pmatrix} \epsilon_{1,B}^{LSDV} - \bar{\epsilon}_1^{LSDV} \\ \vdots \\ \epsilon_{N,B}^{LSDV} - \bar{\epsilon}_N^{LSDV} \\ \epsilon_{1,A}^{LSDV} - \bar{\epsilon}_1^{LSDV} \\ \vdots \\ \epsilon_{N,A}^{LSDV} - \bar{\epsilon}_N^{LSDV} \end{pmatrix}}_{\epsilon_r^{LSDV}}$$

This is very close to the formula for the Within estimator we have seen above. The only difference is that we have two time fixed effects instead of a constant and the **After** time fixed effect. We are going to solve for the estimator in a very similar way. First:

$$X_r^{LSDV\prime} X_r^{LSDV} = N \underbrace{\begin{pmatrix} 1 & 0 & -\frac{\bar{D}}{2} \\ 0 & 1 & \frac{\bar{D}}{2} \\ -\frac{\bar{D}}{2} & \frac{\bar{D}}{2} & \frac{\bar{D}}{2} \end{pmatrix}}_{x_r^{LSDV\prime} x_r^{LSDV}}$$

The determinant of $x_r^{LSDV\prime} x_r^{LSDV}$ is:

$$\det(x_r^{LSDV\prime} x_r^{LSDV}) = \frac{1}{2}\bar{D}(1-\bar{D})$$

and its adjoint matrix is:

$$x_r^{LSDV\tilde{\prime}} x_r^{LSDV} = \begin{pmatrix} \frac{1}{2}\bar{D}(1-\frac{1}{2}\bar{D}) & -\frac{1}{4}\bar{D}^2 & \frac{1}{2}\bar{D} \\ -\frac{1}{4}\bar{D}^2 & \frac{1}{2}\bar{D}(1-\frac{1}{2}\bar{D}) & -\frac{1}{2}\bar{D} \\ \frac{1}{2}\bar{D} & -\frac{1}{2}\bar{D} & 1 \end{pmatrix}.$$

Finally, we have:

$$\begin{aligned} X_r^{LSDV\prime} Y_r^{LSDV} &= \begin{pmatrix} \sum_{i=1}^N (Y_{i,B} - \bar{Y}_i) \\ \sum_{i=1}^N (Y_{i,A} - \bar{Y}_i) \\ -\sum_{i=1}^N \bar{D}_i (Y_{i,B} - \bar{Y}_i) + \sum_{i=1}^N (D_i - \bar{D}_i)(Y_{i,A} - \bar{Y}_i) \end{pmatrix} \\ &= \begin{pmatrix} -\frac{1}{2}N(\bar{Y}_A - \bar{Y}_B) \\ \frac{1}{2}N(\bar{Y}_A - \bar{Y}_B) \\ \frac{1}{2}N\bar{D}(\bar{Y}_A^1 - \bar{Y}_B^1) \end{pmatrix} \end{aligned}$$

Using the fact that $\hat{\Theta}_r^{LSDV} = (X_r^{LSDV\prime} X_r^{LSDV})^{-1} X_r^{LSDV\prime} Y_r^{LSDV}$, we have:

$$\begin{aligned} \hat{\beta}^{LSDV} &= \frac{2}{N\bar{D}(1-\bar{D})}\left[-\frac{\bar{D}N}{2}(\bar{Y}_A - \bar{Y}_B) + \frac{\bar{D}N}{2}(\bar{Y}_A^1 - \bar{Y}_B^1)\right] \\ &= \frac{1}{1-\bar{D}}\left[\bar{Y}_A^1 - \bar{Y}_B^1 - (1-\bar{D})(\bar{Y}_A^0 - \bar{Y}_B^0) - \bar{D}(\bar{Y}_A^1 - \bar{Y}_B^1)\right] \\ &= \frac{1}{1-\bar{D}}\left[(1-\bar{D})(\bar{Y}_A^1 - \bar{Y}_B^1) - (1-\bar{D})(\bar{Y}_A^0 - \bar{Y}_B^0)\right] \\ &= \bar{Y}_A^1 - \bar{Y}_B^1 - (\bar{Y}_A^0 - \bar{Y}_B^0). \end{aligned}$$

The second equality uses the fact that $\bar{Y}_A - \bar{Y}_B = (1-\bar{D})(\bar{Y}_A^0 - \bar{Y}_B^0) + \bar{D}(\bar{Y}_A^1 - \bar{Y}_B^1)$. This proves the result.

To Do: the AP and LC estimators