Causal inference in the social and health sciences: A very brief practical primer in R

Theiss Bendixen

2023-01-23

1 Introduction

Causal inference is about what works, when, for whom and under what circumstances. When an event occurs – a medical treatment, social intervention, etc. – we want to know what happened as a result, if anything. But – and here's the crucial insight – we also need to know what would have happened *in the absence of* the event, all else being equal. That is, the causal effect of an event is the difference between what actually happened and the potential but actually-not-occurring counter-factual.

This working paper is a very brief practical primer on causal inference in the health and social sciences. We'll use minimal formal notation and instead emphasize practical computational implementation using the `R` scripting language. It assumes a basic understanding of Bayesian regression and generalized linear modeling in `R`. Most importantly, however, it assumes an interest in the question of how we can ever know whether something caused something else and to what extent.

Without further ado, let's dive in.

2 The fundamental issue of causal inference: Observed, potential and missing outcomes

We said just above that the causal effect of an event is the difference between what actually happened and the potential but actually-not-occurring counter-factual. But here immediately the fundamental issue of causal inference comes crashing down at our feet: We can never observe the counter-factual!

We can never re-play history or anything of the sort. For instance, we can never give a medical treatment to a patient and also *not* give the treatment to said patient. If we could, we could simply take the difference between the treatment and no treatment states to obtain an **individual treatment effect**. That'd be ideal. But it's not feasible, for obvious reasons. The counter-factual outcome is missing. Causal inference is, then, fundamentally a missing data problem.

So, the best we can do is to *infer* the counter-factual by estimation, approximation or informed (hopefully) guesswork and contrast that with the observed state. More precisely, the best we can do is to aim at an **average treatment effect** or variants thereof. An average treatment effect of, say, a medical treatment is the difference in outcomes under receiving the treatment vs. not receiving the treatment not within an individual, like the individual treatment effect, but instead *between* individuals. Further, these individuals have to be similar enough in relevant characteristics such that, whatever the difference in outcome between the treatment and no treatment groups is, in fact, explained by the treatment alone, and not by some other events or developments.

This, in a nutshell, is the fundamental issue of causal inference. It's simple. Deceptively so, because the real world is a mess. So we have assumptions to make. But in return we get something quite extraordinary: A rigorous, tried-and-tested framework for thinking about cause-and-effect-relationships. The fundamental issue of causal inference, then, is also a *promise*: A promise that, if we're careful, honest, and transparent, we might actually get causal answers to the questions that matter most in life. Or, alternatively, we'll be told that there exist no valid answers to the questions we posed. Either is a very valuable yield.

2.1 Some notation

To make the discussion more succinct, we'll use some simple formal notation and terminology. It's useful to learn these for another reason, too: perhaps you'll want to navigate the causal inference literature on your own, when you're done here.

Throughout, Y is our outcome on which our cause has or hasn't an effect, and X is the cause, our predictor of interest – say, receiving a particular medical treatment $X = 1$ or not $X = 0$. That is, in its simplest form, an average treatment effect is simply the difference in expected outcomes between receiving the treatment $Y^{X=1}$ or not $Y^{X=0}$. This quantity is often written as $E[Y^{X=1}] - E[Y^{X=0}]$, where $E(.)$ is the expectation operator, i.e. it computes the expected value.

2.2 DAGs and experimental randomization

One useful tool for thinking through a causal modeling problem is **DAG**s, *directed acyclic graphs.* A DAG graphs the assumed[1] causal relationships among variables and can thereby guide subsequent analytic strategies for recovering the causal effects of interest (or can reveal that no such effects can be recovered under anu circumstances, given the DAG). *Directed* means that the relationships, denoted by arrows, are causal. *Acyclic* means that variables can't cause themselves dynamically. And *graph* means… well, that should be obvious.

Figure 1 is one example of a DAG. It shows a simple but extremely common scenario in which both our cause X and outcome Y are influenced by a third variable Z. In other words, Z is a **confounder**. We'll discuss confounding in more depth in sections below. For now, suffice it to say that our estimate of the cause of X on Y will be biased if we do not take into account Z. How do we know that Z confounds the relationship between X and Y and thereby induces bias? Simple, we can read it off the DAG.

$$\begin{tikzcd}
 & Z \arrow[dl] \arrow[dr] & \\
X \arrow[rr] & & Y
\end{tikzcd}$$

Figure 1: Directed acyclic graph (DAG) of confounding.

To see this, note that Z has a causal effect on both X and Y. This is technically called a **backdoor path**. A backdoor path is any set of arrows that both points into the outcome Y and predictor of interest X. In a basic sense, causal inference is about identifying such backdoor paths and then "blocking' ' them, using study design or statistical adjustment. This simple insight makes analyzing DAGs a sort of party game or puzzle. It can actually *be fun* to draw out more or less complicated DAGs and then identify how and whether a causal effect can be recovered under this particular graph, through backdoor paths and the likes. There's more to graph analysis than looking for and blocking backdoor paths, as we'll see below. But it's a good start.

So how do we block backdoor paths? I said just above that we can, among other approaches, use particular study designs. Randomization is the best known example of such a study design. What randomization does, from the perspective of DAGs, is that it deletes any arrows that go into X, meaning that there can longer be any backdoor paths, by definition. Figure 2 illustrates this. It's similar to figure 1, except that there's no arrow from Z to X. Under this model, an estimate of the effect of X on Y is no longer biased and we don't need to measure Z[2].

Randomization allows us to delete arrows going into X from all other variables because randomization is, well, random. So, whether an individual receives a randomized medical treatment will not be dependent on anything else than the randomization mechanism. In the next section, we'll show in code and simple simulated data how this works.

[1] Based on prior studies, theory, commonsense, or other sources of inference.
[2] Although, on a technical aside, knowing Z potentially increases the precision of our estimate of X.

$$\begin{tikzcd} & Z \arrow[d] \\ X \arrow[r] & Y \end{tikzcd}$$

Figure 2: DAG of randomized X.

In sum, randomization is an efficient strategy for causal estimation, because it makes X independent of all other variables in the graph. This is the reason why randomized controlled studies are often referred to as the ‘‘gold-standard” for causal inference. But, unfortunately, our work does not end here. Randomization is often imperfect (e.g., participant might not perfectly adhere to the randomized treatment) or not feasible for practical or ethical reasons (e.g., could we assign people to smoke in a randomized fashion such that we could estimate the causal effect of smoking on, say, cancer?).

This does not mean that all hope is lost. Causal inference in observational or pseudo-randomized contexts is possible. But it does force us to make some assumptions and add some steps to our workflow. For instance, through relatively simple statistical techniques, we can block backdoor paths that couldn't be deleted by randomization. Next section introduces the promises and pitfalls of this approach, using regression modeling.

3 Blocking backdoors

This section riffs on – and in a few ways goes slightly beyond – see Cinelli's Crash Course and McElreath 2023 Rethinking lectures 5 and 6.

As such, it will discuss:

Multiple regression as an alternative to randomization for blocking back-door paths; Table 2 fallacy and the danger of garbage can regression (e.g., blocking a mediator, collider bias, conditioning on post-treatment); conditional vs. marginal effects; some promises (decomposing effects, mediation, front-door criterion).

One critical take-home is that causes are not in the data (for instance, you cannot distinguish between a pipe and a fork from the data alone, because they have similar implications). A DAG cannot be built on the basis of a set of regressions. Regressions can be used to test the implications of a DAG or a set of them (for a discussion of this point in the wild, see Purzycki, Bendixen, Lightner, 2022). All this means that variable selection – which variables to include in your adjustment set – cannot be automated; it strictly requires prior knowledge (see also books: What if?; Causal inference by design; etc.)

The *average* person (conditional) vs. people *on average* (marginal). Y ~ X is the marginal estimate. But adjustment is critical for de-confounding. But, adjustment returns only conditional estimates. Enter g-methods.

4 *G*-methods and marginal effects

As we saw above, multiple regression is a powerful tool for blocking backdoor paths. This allows for causal interpretation of the focal parameter(s) (but, importantly, *not* the control variables). However, regression has drawbacks that makes it insufficient for many common causal estimands, in particular when there are many covariates with complex relationships. Often, we're interested in marginal, not conditional, average treatment effects – that is, average treatment effects in a population as a whole, not just subsets thereof. In more complicated data structures, such as longitudinal studies with dynamical relationships between treatment and outcome, regression alone will also fall short. In all these cases, we need a few additional tools in the trunk.

One such family of tools is known as **g-methods** – *g* for *general* or *generalized.* Here, we'll focus on one g-method in particular, often referred to as g-computation or **standardization**, while we'll also very briefly introduce another method, known as **inverse probability of treatment weighting** (IPTW). Standardization and IPTW rely on the same assumptions and will yield similar if not identical results (indeed, in many simpler cases, they are mathematically identical), but they arrive at their results at quite different analytic routes.

We focus primarily on standardization for a few reasons. First, since we want to perform our data analysis in a Bayesian framework, standardization is an obvious choice since the IPT weights are not obviously compatible with Bayes theorem (I leave out many details and some exceptions, see REFERENCE for more).

Second, compared to IPTW, standardization and its practical implementation seem to be often overlooked in popular text books and primers on causal inference and econometrics in the social and health sciences (e.g., Morgan & Winship, Primer, Causal Inference by Design, Mostly Harmless, Epidemiology by Design, even McElreath, although the latter stresses contrasts. It is discussed in Hernan & Robins (What if? ch. 13) but without explicit, reproducible practical/programming application, though there's a g-methods package). However, given its popularity, it's useful to at least be aware of the nuts and bolts of IPTW, too. So let's briefly present the two methods in turn.

4.1 Inverse Probability Weighting

Recall that the main aim of a covariate-adjusted analysis is to obtain conditional exchangeability: When we account for imbalances in covariates between treatment and control group, we say the two groups are exchangeable conditionally on the covariates. This means that the two groups are comparable such that if we detect a difference between the groups after the intervention, we can interpret that difference as a causal effect of the intervention.

The IPTW method obtains conditional exchangeability by, in effect, creating a "pseudo-population" in which covariates are balanced between treatment and control. As the name hints at, this pseudo-population is created by estimating the probability of receiving the treatment conditional on covariates. This probability is then inverted and used as weights in a regression predicting the actual outcome of interest. In code, the procedure looks like this. First, we simulate a simple, confounded data structure, with a binary treatment X with a coefficient of βx, binary confounder Z and a continuous outcome Y.

```r
set.seed(123)

n <- 1e4

bX <- 0.5

Z <- rbinom(n, 1, 0.3)
X <- rbinom(n, 1, 0.4+Z*0.5)
Y <- rnorm(n, 10 + X*0.5 + Z*0.5)

gdat <- data.frame(Y=Y, X=X, Z=Z)
```

Next, we calculate IPT weights by, first, fitting a model that regresses X on Z. Then, for each row in the dataset, we get predictions from this model for the probability of receiving treatment. Then we compute the IPT weights and includes those weights in a regression that predicts the outcome by X. This latter step, the outcome regression, is known as a **marginal structural model** (MSM). It's a *marginal* model, because the coefficient of X is an average (or marginal, since the estimated weights *marginalizes over* the distribution of the covariate) treatment effect in the population; and *structural*, because X has a valid causal interpretation (*structural* is another word for *causal*).

```r
# inverse logit function
inv_logistic <- function(x) exp(x)/(1+exp(x))

# receiving treatment conditional on Z
treat.mod <- glm(X ~ Z, data = gdat, family = "binomial")

# probability of treatment
```

```r
gdat$pd <- predict(treat.mod) |> inv_logistic()

# compute inverse weights
gdat$w <- with(gdat, ifelse(X==1, 1/pd, 1/(1-pd)))

# MSM of outcome
glm(Y ~ X, data = gdat, weights = w)
```

The MSM recovers the simulated coefficient for X ($\beta x = 0.5$), even though the model does not adjust for Z in a traditional sense. This is because we adjusted for Z using weights. Note that this simple example could also, of course, be obtained using simple multiple regression that adjusted for X and Z. This, however, will not be so, when we tackle more complex longitudinal data in the next section.

There are at least two ways to improve on our simple IPTW workflow above. First, above we calculated unstabilized weights, but in many cases, so-called **stabilized weights** are preferred (and in all cases, it's good to know about both). The stabilized weights are different from the unstabilized in that the numerator is the unconditional inverse probability of treatment. They can be calculated with an intercept-only regression on treatment.

```r
# intercept-only model on treatment
treat.mod.sw <- glm(X ~ 1, data = gdat, family = "binomial")

# probability of treatment
gdat$pn <- predict(treat.mod.sw) |> inv_logistic()

# compute stabilized weights
gdat$sw <- with(gdat, ifelse(X==1, pn/pd, pn/(1-pd)))

# MSM of outcome with stabilized weight
glm(Y ~ X, data = gdat, weights = sw)
```

The other thing we can do to improve the workflow is to obtain a measure of uncertainty around our marginal estimate. However, since [**placeholder**], we can't rely on the standard error from our MSM. Instead, we must resort to **bootstrapping**, which involves running the same algorithmic routine R times. This will result in a distribution of estimates with length R. Here's one way to implement bootstrapping with the stabilized weights.

```r
library(boot)

# IPTW function
iptw_fun <- function(formula, data, indices) {
  d <- data[indices,]
  fit <- glm(formula, family="gaussian", weights=sw, data=d)
  return(coef(fit))
}

iptw.result <- boot(
                # set data
                data=gdat,
                # set function
                statistic=iptw_fun, # set function
                # set R
                R=1e3,
                # specify formula
                formula=Y ~ X)
```

```
# bootstrapped point estimate for bX
iptw.point <- iptw.result$t0[2]

# bootstrapped interval for bX
iptw.interval <- boot.ci(iptw.result,
                         type = "norm",
                         index = 2)$normal
```

That's it for IPTW for now. They are popular and reasonably straightforward to implement. However, as mentioned, since there's no very clear way of obtaining uncertainty in the weights and then propagate that uncertainty to the MSM in a formal Bayesian framework, we leave IPTW here.

4.2 Standardization

Standardization is our main g-method here. It's arguably even more straightforward to implement than IPTW, as it requires us to only fit a single regression of the outcome and then do some post-fitting simulation.

The conceptual steps in standardization are as follows:

- Fit a theoretically informed model of the outcome including the treatment, covariates and possibly non-linear relationships and functional forms (e.g., interaction terms, quadratic terms, etc.)
- Duplicate the original data, set $X = 1$ for all individuals, and then obtain predictions of the outcome using the fitted model holding covariate(s) Z as observed z, $E[Y^{Z=z,X=1}]$.
- Duplicate the original data, set $X = 0$ for all individuals, and then obtain predictions of the outcome using the fitted model holding covariate(s) Z as observed z, $E[Y^{Z=z,X=0}]$.
- Compute contrast corresponding to the estimand of interest, $E[Y^{X=1}] - E[Y^{X=0}]$.

In code, using Bayesian estimation with default priors via `Stan` and `brms`, the steps are these.

```
library(brms)
library(tidybayes)

# outcome model
stand.mod <- brm(Y ~ X + Z,
                 data = gdat,
                 family = "gaussian",
                 cores = 4)

# set X=1, Z=z
X1 <- transform(gdat, X=1)

# set X=0, Z=z
X0 <- transform(gdat, X=0)

# E[Y{Z=z, X=1}]
Y.x1 <- posterior_epred(stand.mod, X1)

# E[Y{Z=z, X=0}]
Y.x0 <- posterior_epred(stand.mod, X0)

# E[Y{X=1} - Y{X=0}]
mean_hdci(Y.x1[,1] - Y.x0[,1])
```

The distribution of contrasts calculated in that final step are marginal estimates in that the two expectations

$E[Y^{Z=z,X=1}]$ and $E[Y^{Z=z,X=0}]$ marginalize or average over the covariate(s) Z. We summarize the contrast by its posterior mean and HPDI, using the `tidybayes` package.

When performing standardization in a frequentist setting, we'd have to resort to bootstrapping in order to obtain valid uncertainty around the contrast, as with IPTW. However, in a Bayesian setting, we get uncertainty in one go, since the predicted values are valid posterior distributions of expectations.

Okay, now that we have introduced g-methods in a very simple data context, we're ready to tackle more complex data structures where regression modeling alone falls short.

4.3 Longitudinal analysis, time-varying treatments, doubly robust estimators

For illustrating a longitudinal data context with time-varying treatment, we'll simulate from the DAG in figure 3. Suppose that our treatment X is randomized but administered according to some time-varying covariate Z. We're interested in the *joint effects* of the treatment X_0 and X_1 on our outcome Y (that is, the effects of each treatment on the outcome not through the subsequent treatment), denoted by the dashed edges.

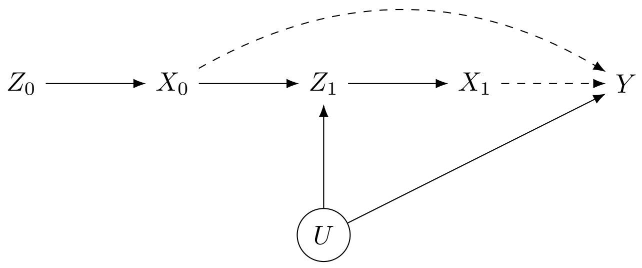
$$\begin{tikzcd} Z_0 \arrow[r] & X_0 \arrow[r] \arrow[rrr, dashed, bend left] & Z_1 \arrow[r] & X_1 \arrow[r, dashed] & Y \\ & & U \arrow[u] \arrow[rru] & & \end{tikzcd}$$

Figure 3: DAG of longitudinal treatment-confounder feedback.

However, some unobserved variable U affects both Z_1 and Y. This confounds the relationship between the treatment and the outcome via the backdoor path $Y \leftarrow U \rightarrow Z_1 \rightarrow X_1$. This in turn means that in order to estimate the hypothesized causal path $X_1 \rightarrow Y$, we'd want to adjust for Z_1.

But here's the issue: Z_1 is also a collider on the path $Y \leftarrow U \rightarrow Z_1 \leftarrow X_0$, implying that if we condition on Z_1, we open a non-causal path between X_0 and Y. All in all, under this DAG, we're not able to estimate the joint effects of X_0 and X_1 on Y using a single regression model, since the two treatment indicators imply different adjustment sets. G-methods to the rescue.

First, we simulate some data, under the model presented in figure 3. To keep things simple, we assume that the treatment, in fact, does not have any direct impact on Y (i.e., we could delete the dashed edges), such that the true coefficients of X_0 and X_1 should be (roughly) zero.

```r
set.seed(9)

n <- 1e4

U <- rnorm(n, 0, 1)
Z_0 <- rbinom(n, 1, 1/(1+exp(0.5)))
X_0 <- rbinom(n, 1, 1/(1+exp(0.5 + Z_0*0.5)))
Z_1 <- rbinom(n, 1, 1/(1+exp(0.5 + X_0*0.5+U*0.5)))
X_1 <- rbinom(n, 1, 1/(1+exp(0.5 + Z_1*0.5)))

Y <- rnorm(n, 10 + U*2)

lgdat <- data.frame(Y=Y, X_0=X_0, X_1=X_1, Z_0=Z_0, Z_1=Z_1, U=U)
```

Then, we verify the chaos that Z_1 can havoc, if we haphazardly estimated the effect of X_0 on while adjusting for Z_1. The coefficient is solidly non-zero.

```r
mx0 <- glm(Y ~ X_0 + Z_1, data = lgdat)
summary(mx0)
confint(mx0)
```

And similarly, we check that *not* adjusting for Z_1 can bias our estimate of X_1.

```r
mx1 <- glm(Y ~ X_1, data = lgdat)
summary(mx1)
confint(mx1)
```

Then, we apply standardization to the data. The logic is the same as discussed above, only now we have two time points.

So, we first fit a model for the treatment at time point 0 and then compute the contrast in expectations between receiving and not receiving the treatment, $E[Y^{X_0=1,Z_0=z_0}] - E[Y^{X_0=0,Z_0=z_0}]$. Next, we fit a model for receiving treatment at time point 1 and similarly compute the contrast in expectations between receiving and not receiving the treatment, $E[Y^{X_1=1,X_0=x_0,Z_1=z_1}] - E[Y^{X_1=0,X_0=x_0,Z_1=z_1}]$.

```r
# Outcome model for X_0
yX0model <- brm(Y ~ X_0 + Z_0,
                data = lgdat, cores = 4)

# E[Y{X_0=1, Z_0=z_0}]
yX0_1 <- posterior_epred(yX0model, newdata = transform(lgdat, X_0=1))

# E[Y{X_0=0, Z_0=z_0}]
yX0_0 <- posterior_epred(yX0model, newdata = transform(lgdat, X_0=0))

# Contrast: E[Y{X_0=1}] - E[Y{X_0=0}]
yX0contrast <- mean_hdci((yX0_1[,1] - yX0_0[,1]))

# Outcome model for X_1
yX1model <- brm(Y ~ X_0 + X_1 + Z_1,
                data = lgdat, cores = 4)

# E[Y{X_1 = 1, X_0=x_0, Z_1=z_1}]
yX1_1 <- posterior_epred(yX1model, newdata = transform(lgdat, X_0=1))

# E[Y{X_1=0, X_0=x_0, Z_1=z_1}]
yX1_0 <- posterior_epred(yX1model, newdata = transform(lgdat, X_0=0))

# Contrast: E[Y{X_1=1}] - E[Y{X_1=0}]
yX1contrast <- mean_hdci((yX1_1[,1] - yX1_0[,1]))
```

By breaking the procedure down into separate models – one for each time point and treatment – we by-pass the problem that a single regression runs into, namely that Z_1 is both a collider and a confounder. The resulting contrasts, stored in `yX0contrast` and `yX1contrast`, are valid causal estimates of the joint effects of each treatment on the outcome.

Now for the finale. The above example comes from What if?, and it's useful in that illustrates how regression modeling alone breaks if our estimands require different adjustment sets (e.g., if a variable is both a confounder that we want to adjust for and a collider that we want to remain unadjusted). But let's take standardization for a spin in a real-world data analysis example. Vanderweele et al. sets the stage for our example: religion and (mental) health.

The DAG could look something like figure 4.

$$\begin{tikzcd}
 & & Z_1 \arrow[r] \arrow[ddr] \arrow[ddrr] \arrow[ddrrr] & Z_2 \arrow[r] \arrow[ddr] \arrow[ddrr] & Z_3 \arrow[r] \arrow[ddr] & Z_4 \\
C \arrow[ur] \arrow[r] \arrow[dr] & {} & & & & \\
 & {} & R_1 \arrow[r] \arrow[uur] \arrow[uurr] \arrow[uurrr] & R_2 \arrow[r] \arrow[uur] \arrow[uurr] & R_3 \arrow[r] \arrow[uur] & R_4
\end{tikzcd}$$

Figure 4: DAG of longitudinal exposure-outcome feedback.

C is a set of baseline covariates comprising standard demographics, including age, gender, income and education; the indefinite arrows indicate confounding on all observed variables). Our exposure R_{1-4} represent level of religiosity ("How important is religion to you?", measured on a 1 ("not at all important") to 4 ("very important") scale.) measured at four time points, and Z_{1-4} are subjective general health ("How would you describe your current health?", measured on a 1 ("very bad") to 5 ("very good") scale.), a set of time-varying covariates. Note the complex causal relationships across time between R and Z.

To provide some background, this dataset comes from a panel study on a pseudo-representative sample of Danes during the COVID-19 pandemic (for more detail, see Mauritsen et al., *in review*). As investigators, we are interested in assessing whether religiosity was a protective factor in terms of subjective general well-being in the context of the stress and uncertainty of a pandemic. If that was so, we'd expect a positive effect of religiosity on subjective health.

Suppose, then, that our estimand of interest is the contrast in subjective well-being at the final measurement time (that is, $Y = Z_4$) between being "maximally religious" $Y^{R_1=R_2=R_3=4}$ and being "minimally religious" $Y^{R_1=R_2=R_3=1}$. That is, we're interested in the *joint effects* of being religious on health at all time points up to the final measurement, meaning partitioning the respective effects of religiosity level at each time point on the outcome, *except those running through the subsequent measurements of religiosity.*

Note that we could've made the causal diagram even more involved by hypothesizing that our two time-varying variables also caused future versions of themselves beyond the immediately subsequent time point (e.g., adding paths $R_1 \rightarrow R_3$, $R_1 \rightarrow R_4$, $Z_1 \rightarrow Z_3$, $Z_1 \rightarrow Z_4$, etc.).

However, even as it stands, this analysis is not straightforward. Consider for instance that some of the effect of R_1 on the outcome works through Z_2 (and Z_3) and that if we adjust for Z_2 (or Z_3), we'll block some of the effect of R_1. But note too that Z_2 (through Z_3) confounds the relationship between R_3 and the outcome, such that if we do not adjust for Z_2 (or Z_3), the estimate for the effect of R_3 will be biased.

All this implies that Z_2 is *both* a mediator that we'd want to keep unadjusted *and* a confounder that we'd want to adjust for. A single regression cannot handle this situation. But, as we've already seen, g-methods can.

First, we load the data, packages, and set the path to `cmdstan`, which helps speed up the sampling procedure. We also set the number of iterations to use in sampling. For now, we use only the complete cases (in all variables but for R_4, which we're not actually using at this moment); below, we'll discuss in more detail what a complete cases analysis implies and ways of handling missing covariate values when using standardisation.

```r
veluxdat <- read.csv("velux_data.csv")

d <- veluxdat[complete.cases(veluxdat[, !names(veluxdat) %in% c("religion_4")]),]

library(cmdstanr)
library(brms)
library(tidybayes)

set_cmdstan_path("C:\\cmdstan\\cmdstan-2.29.0")
```

```
iter <- 2000
```

Then, we fit the model for R_1 on the outcome Z_4. We're interested in estimating all paths from $R_1 \rightarrow Z_4$ except those through subsequent religiosity measurements, while controlling for the demographic confounders. This means that, in addition to the baseline covariates and our main exposure R_1, we only need to include R_2 in the model. Adjusting for R_2 blocks the paths from R_1 to the outcome through subsequent measures of religion. Had we assumed the path $R_1 \rightarrow R_3$, we'd have to include R_3, too, to block the path $R_1 \rightarrow R_3 \rightarrow Z_4$.

```
## Outcome model for religion_t1
r1model <- brm(health_4 ~ gender + age.c + education + household_income + religion_1 + religion_2,
               data = d,
               cores = 4, iter = iter, control = list(adapt_delta = 0.99),
               backend = "cmdstanr")

rstan::check_hmc_diagnostics(r1model$fit)
pp_check(r1model)

# Min. religious: E[Y{R_1=1, C=c, R_2=r_2}]
yr1_0 <- posterior_epred(r1model, newdata = transform(d, religion_1=1))

# Max. religious: E[Y{R_1=4, C=c, R_2=r_2}]
yr1_1 <- posterior_epred(r1model, newdata = transform(d, religion_1=4))

# Contrast: E[Y{R_1=4}] - E[Y{R_1=1}]
(yr1contrast <- mean_hdci((yr1_1[,1] - yr1_0[,1])))
```

Next, we fit a model for R_2. In addition to the main exposure and the demographic covariates, we need to adjust for both R_1 (because it's a confounder on the path $R_2 \leftarrow R_1 \rightarrow Z_4$), R_3 (to block the path $R_2 \rightarrow R_3 \rightarrow Z_4$) and Z_1 (because it's a confounder on the path $R_2 \leftarrow Z_1 \rightarrow Z_2 \rightarrow Z_3 \rightarrow Z_4$).

```
## Outcome model for religion_t2
r2model <- brm(health_4 ~ gender + age.c + education + household_income + religion_1 + religion_2 + reli
               data = d,
               cores = 4, iter = iter, control = list(adapt_delta = 0.99),
               backend = "cmdstanr")

rstan::check_hmc_diagnostics(r2model$fit)
pp_check(r2model)

# Min. religious: E[Y{R_2=1, C=c, H_1=h_1, R_1=r_1, R_3=r_3}]
yr2_0 <- posterior_epred(r2model, newdata = transform(d, religion_2=1))

# Max. religious: E[Y{R_2=4, C=c, H_1=h_1, R_1=r_1, R_3=r_3}]
yr2_1 <- posterior_epred(r2model, newdata = transform(d, religion_2=4))

# Contrast: E[Y{R_2=4}] - E[Y{R_2=1}]
(yr2contrast <- mean_hdci((yr2_1[,1] - yr2_0[,1])))
```

Finally, the model for R_3 adjusts for the demographic covariates as well as R_2 (because it's a confounder on the path $R_3 \leftarrow R_2 \rightarrow Z_4$) and Z_2 (because it's a confounder on the path $R_3 \leftarrow Z_2 \rightarrow Z_4$ through Z_3). Notice that, had we assumed a causal relationship $R_4 \rightarrow Z_4$, R_4 would have to be included, too.

```
# Outcome model for religion_t3
r3model <- brm(health_4 ~ gender + age.c + education + household_income + religion_2 + religion_3 + hea
               data = d,
               cores = 4, iter = iter, control = list(adapt_delta = 0.99),
```

```r
               backend = "cmdstanr")

rstan::check_hmc_diagnostics(r3model$fit)
pp_check(r3model)

# Min. religious: E[Y{R_3=1, C=c, H_2=h_2, R_2=r_2}]
yr3_0 <- posterior_epred(r3model, newdata = transform(d, religion_3=1))

# Max. religious: E[Y{R_3=4, C=c, H_2=h_2, R_2=r_2}]
yr3_1 <- posterior_epred(r3model, newdata = transform(d, religion_3=4))

# Contrast: E[Y{R_3=4}] - E[Y{R_3=1}]
(yr3contrast <- mean_hdci((yr3_1[,1] - yr3_0[,1])))
```

The objects `yr1contrast`, `yr2contrast` and `yr3contrast` store the posterior means and HPDIs of the joint effects of religiosity on subjective health. You can inspect the contrasts to see that they huddle mostly around zero, meaning that we find little evidence for the notion that religion on average works as a protective factor during the pandemic, at least in these data.

One last comment, before we move on: In our estimation above, we silently cut some corners, for the sake of illustrating the basic principles of standardisation in a context with time-varying confounders. There are some additional modeling complexities that we could (and perhaps should) incorporate: Monotonicity in predictors; ordinal (not gaussian) outcome; random intercepts for individuals; Bayesian imputation.

5 To do:

- double check notation throughout – within-document consistency and external consistency with e.g., What if? book
- add references
- a little bit more formal detail on the IPTW weights and the difference btw. unstabilized and stabilized weigths and why we can't rely on the standard errors (and therefore must resort to bootstrapping)
- Missing data MCAR, MAR, MNAR. See Morris et al. (2022, appendix) for discussion of missingness in X and/or Y in the context of g-comp.

Above, we conducted a complete cases analysis, under the assumption that missingness is unsystematic. One way to investigate this assumption a little bit further is to ask, *does health predict dropout at the final measurement*? This is not a bulletproof test, but it's a start.

```r
# 1 if health is missing at t4, 0 otherwise
veluxdat$health_4miss <- ifelse(is.na(veluxdat$health_4), 1, 0)

# Does health at previous measurements predict missingness at final follow-up?
healthmiss_mod <- brm(health_4miss ~ health_1 + health_2 + health_3,
                      data = veluxdat,
                      family = bernoulli,
                      cores = 4,
                      backend = "cmdstanr")
```

- Assumptions, assumptions, assumptions

See end of Standardization chapter in What if?

- Econometric techniques

Regression discontinuity, instrumental variables, synthetic controls

- Threats to identification and some solutions

Attrition (simple MNAR analysis), selection bias, spill-over, non-compliance (IV of randomization), etc.

- Transportability of treatment effects (see What if? chapter)
- Red herrings Testing for covariate imbalances
- Dictionary for common terminology in the causal inference literature

D-separation, Do-calculus,

- Principles for randomized trials/experiments

Adjusting for covariates?

6 Appendix: Velux IPTW

I said above that we'd leave IPTW behind., Well, not quite. Here's how to use IPTWs and a MSM to analyze the panel data on health and religiosity.

Note that we use only the complete cases. Also, this illustration differs from the IPTW above, because now we have a continuous, not a binary, exposure (well, it's categorical, but we assume gaussian for the sake of simplicity). The IPTWs are calculated slightly differently, when the exposure is continuous. All this said, the results are qualitatively (although not numerically, likely due to difference in estimation and also different sample subsets) similar to our standardization approach, in that we find no noteworthy effect of religiosity on health.

However, try and run the bootstrap and subsequent steps *without* the weights (i.e., delete the `weights` argument from `iptw_fun()`); you'll see that the model then picks up a "near-statistically significant" negative association between R_3 and the outcome, which we can only guess is spurious.

```r
diptw <- d[complete.cases(d),] # complete cases

## Computing IPTW with a continuous exposure: https://www.andrewheiss.com/blog/2020/12/01/ipw-binary-co

## the stabilized weights for religion_t1 is 1 (since the numerator and the denominator is the same)
## cf., VanderWeel et al. (online appendix)

## numerator model for religion_t2
r2exppn <- glm(religion_2 ~ gender + age.c + education + household_income + religion_1, data = diptw)
diptw$pn_r2 <- dnorm(diptw$religion_2,
             predict(r2exppn),
             sd(r2exppn$residuals))

## denominator model for religion_t2
r2exppd <- glm(religion_2 ~ gender + age.c + education + household_income + religion_1 + health_1, data
diptw$pd_r2 <- dnorm(diptw$religion_2,
             predict(r2exppd),
             sd(r2exppd$residuals))

## numerator model for religion_t3
r3exppn <- glm(religion_3 ~ gender + age.c + education + household_income + religion_2, data = diptw)
diptw$pn_r3 <- dnorm(diptw$religion_3,
             predict(r3exppn),
             sd(r3exppn$residuals))

## denominator model for religion_t3
```

```
r3exppd <- glm(religion_3 ~ gender + age.c + education + household_income + religion_2 + health_1 + heal
diptw$pd_r3 <- dnorm(diptw$religion_3,
             predict(r3exppd),
             sd(r3exppd$residuals))

## calculate weights
diptw$sw2 <- with(diptw, pn_r2/pd_r2)
diptw$sw3 <- with(diptw, pn_r3/pd_r3)
diptw$sw <- with(diptw, sw2*sw3)

# MSM without and with stabilized weights
velux_uw <- glm(health_4 ~ gender + age.c + education + household_income + religion_1 + religion_2 + rel
summary(velux_uw)

velux_msm <- glm(health_4 ~ gender + age.c + education + household_income + religion_1 + religion_2 + re
summary(velux_msm)

### bootstrapping, when weights are calculated
library(boot)

# same function as above
iptw_fun <- function(formula, data, indices) {
  d <- data[indices,]
  fit <- glm(formula, family="gaussian", weights = sw, data=d)
  return(coef(fit))
}

iptw.velux.result <- boot(data = diptw,
                 statistic = iptw_fun,
                 R = 1e4,
                 formula = health_4 ~ gender + age.c + education + household_income + religion_1 + relig

# bootstrapped point estimate for religion_t1
iptw.velux.r1point <- iptw.velux.result$t0[6]

# bootstrapped point estimate for religion_t2
iptw.velux.r2point <- iptw.velux.result$t0[7]

# bootstrapped point estimate for religion_t3
iptw.velux.r3point <- iptw.velux.result$t0[8]

# bootstrapped interval for religion_t1
iptw.velux.r1interval <- boot.ci(iptw.velux.result,
                        type = "norm",
                        index = 6)$normal

# bootstrapped interval for religion_t2
iptw.velux.r2interval <- boot.ci(iptw.velux.result,
                        type = "norm",
                        index = 7)$normal

# bootstrapped interval for religion_t3
iptw.velux.r3interval <- boot.ci(iptw.velux.result,
                        type = "norm",
                        index = 8)$normal
```