A guided tour in targeted learning territory

David Benkeser, Antoine Chambaz, Nima Hejazi

08/06/2018

Contents

1 Introduction

This is a very first draft of our article. The current *tentative* title is "A guided tour in targeted learning territory".

Explain our objectives and how we will meet them. Explain that the symbol indicates more delicate material.

Use sectioning a lot to ease cross-referencing.

Do we include exercises?

```r
set.seed(54321) ## because reproducibility matters...
suppressMessages(library(R.utils)) ## make sure it is installed
suppressMessages(library(tidyverse)) ## make sure it is installed
suppressMessages(library(ggplot2)) ## make sure it is installed
expit <- plogis
logit <- qlogis
```

2 A simulation study

blabla

2.1 Reproducible experiment as a law. We are interested in a reproducible experiment. The generic summary of how one realization of the experiment unfolds, our observation, is called O. We view O as a random variable drawn from what we call the law P_0 of the experiment. The law P_0 is viewed as an element of what we call the model. Denoted by $\mathcal{M}$, the model is the collection of *all* laws from which O can be drawn and that meet some constraints. The constraints translate the knowledge we have about the experiment. The more we know about the experiment, the smaller is $\mathcal{M}$. In all our examples, model $\mathcal{M}$ will put very few restrictions on the candidate laws.

Consider the following chunk of code:

```r
draw_from_experiment <- function(n, full = FALSE) {
  ## preliminary
  n <- Arguments$getInteger(n, c(1, Inf))
  full <- Arguments$getLogical(full)
  ## ## 'Gbar' and 'Qbar' factors
  Gbar <- function(W) {
    expit(-0.2 + 3 * sqrt(W) - 1.5 * W)
  }
  Qbar <- function(AW) {
    A <- AW[, 1]
    W <- AW[, 2]
    A * cos((1 + W) * pi / 5) + (1 - A) * sin((1 + W^2) * pi / 4)
  }
  ## sampling
  ## ## context
  W <- runif(n)
  ## ## counterfactual rewards
  zeroW <- cbind(A = 0, W)
  oneW <- cbind(A = 1, W)
  Qbar.zeroW <- Qbar(zeroW)
  Qbar.oneW <- Qbar(oneW)
  Yzero <- rbeta(n, shape1 = 1, shape2 = (1 - Qbar.zeroW) / Qbar.zeroW)
  Yone <- rbeta(n, shape1 = 1, shape2 = (1 - Qbar.oneW) / Qbar.oneW)
  ## ## action undertaken
  A <- rbinom(n, size = 1, prob = Gbar(W))
  ## ## actual reward
  Y <- A * Yone + (1 - A) * Yzero
  ## ## observation
  if (full) {
    obs <- cbind(W = W, Yzero = Yzero, Yone = Yone, A = A, Y = Y)
  } else {
    obs <- cbind(W = W, A = A, Y = Y)
  }
  attr(obs, "Gbar") <- Gbar
  attr(obs, "Qbar") <- Qbar
  attr(obs, "QW") <- dunif
  ##
  return(obs)
}
```

We can interpret `draw_from_experiment` as a law P_0 since we can use the function to sample observations from a common law. It is even a little more than that, because we can tweak the experiment, by setting its `full` argument to `TRUE`, in order to get what appear as intermediary (counterfactual) variables in the regular experiment. The next chunk of code runs the (regular) experiment five times independently:

```r
(five_obs <- draw_from_experiment(5))
```

```
##                W A         Y
## [1,] 0.4290078 0 0.9998700
## [2,] 0.4984304 1 0.9351501
## [3,] 0.1766923 0 0.9477263
## [4,] 0.2743935 1 0.8287541
## [5,] 0.2165102 1 0.9977092
## attr(,"Gbar")
## function (W)
## {
##     expit(-0.2 + 3 * sqrt(W) - 1.5 * W)
## }
## <bytecode: 0x4b1ac00>
## <environment: 0x5770f20>
## attr(,"Qbar")
## function (AW)
## {
##     A <- AW[, 1]
##     W <- AW[, 2]
##     A * cos((1 + W) * pi/5) + (1 - A) * sin((1 + W^2) * pi/4)
## }
## <bytecode: 0x6410dc8>
## <environment: 0x5770f20>
## attr(,"QW")
## function (x, min = 0, max = 1, log = FALSE)
## .Call(C_dunif, x, min, max, log)
## <bytecode: 0x6403cf8>
## <environment: namespace:stats>
```

We can view the `attributes` of object `five_obs` because, in this section, we act as oracles, *i.e.*, we know completely the nature of the experiment. From a probabilistic point of view, the attributes `Gbar`, `Qbar` and `QW` are infinite-dimensional features of P_0. There is more to P_0 than $\bar{G}_0$ (`Gbar`), $\bar{Q}_0$ (`Qbar`), formally defined by

$$\bar{G}_0(W) \equiv P_0(A = 1|W), \quad \bar{Q}_0(A, W) \equiv E_{P_0}(Y|A, W), \tag{1}$$

and the marginal distribution $Q_{0,W}$ of W under P_0 (`QW`), for instance the conditional distribution (not expectation) of Y given (A, W), but $\bar{G}_0$, $\bar{Q}_0$ and $Q_{0,W}$ will play a prominent role in our story.

2.2 The parameter of interest, first pass. It happens that we especially care for a finite-dimensional feature of P_0 that we denote by ψ_0. Its definition involves the aforementioned infinite-dimensional features:

$$\begin{aligned}\psi_0 &\equiv E_{P_0}\left(\bar{Q}_0(1, W) - \bar{Q}_0(0, W)\right) \\ &= \int \left(\bar{Q}_0(1, w) - \bar{Q}_0(0, w)\right) dQ_{0,W}(w).\end{aligned} \tag{2}$$

Acting as oracles, we can compute explicitely the numerical value of ψ_0.

Our interest in ψ_0 is of causal nature. Taking a closer look at `drawFromExperiment` reveals indeed that the random making of an observation O drawn from P_0 can be summarized by the following causal graph and nonparametric system of structural equations:

```
## plot the causal diagram
```

and, for some deterministic functions f_w, f_a, f_y and independent sources of randomness U_w, U_a, U_y,

1. sample the context where the rest of the experiment will take place, $W = f_w(U_w)$;

2. sample the two counterfactual rewards of the two actions that can be undertaken, $Y_0 = f_y(0, W, U_y)$ and $Y_1 = f_y(1, W, U_y)$;
3. sample which action is carried out in the given context, $A = f_a(W, U_a)$;
4. define the corresponding reward, $Y = AY_1 + (1 - A)Y_0$;
5. summarize the course of the experiment with the observation $O = (W, A, Y)$, thus concealing Y_0 and Y_1.

The above description of the experiment `draw_from_experiment` is useful to ram home what it means to run the "full" experiment by setting argument `full` to `TRUE` in a call to `draw_from_experiment`. Doing so triggers a modification of the nature of the experiment, enforcing that the counterfactual rewards Y_0 and Y_1 be part of the summary of the experiment eventually. In light of the above enumeration, $\mathbb{O} \equiv (W, Y_0, Y_1, A, Y)$ is output, as opposed to its summary measure O. This defines another experiment and its law, that we denote $\mathbb{P}_0$.

It is well known (do we give the proof or refer to other articles?) that

$$\psi_0 = E_{\mathbb{P}_0}(Y_1 - Y_0).$$

Thus, ψ_0 compares (additively) the averages of the two counterfactual rewards. In other words, ψ_0 quantifies the difference in average of the reward one would get in a world where one would always enforce action $a = 1$ with the reward one would get in a world where one would always enforce action $a = 0$. This said, it is worth emphasizing that ψ_0 is a well defined parameter beyond its causal interpretation.

To conclude this subsection, we draw advantage from the possibility to sample full observations from `draw_from_experiment` by setting its argument `full` to `TRUE` in order to numerically approximate ψ_0. By the law of large numbers, the following chunk of code approximates ψ_0:

```r
B <- 1e6
full_obs <- draw_from_experiment(B, full = TRUE)
(psi_hat <- mean(full_obs[, "Yone"] - full_obs[, "Yzero"]))
```

```
## [1] -0.2644049
```

In fact, the central limit theorem and Slutsky's lemma allow us to build a confidence interval with asymptotic level 95% for ψ_0:

```r
sd_hat <- sd(full_obs[, "Yone"] - full_obs[, "Yzero"])
alpha <- 0.05
(psi_CI <- psi_hat + c(-1, 1) * qnorm(1 - alpha / 2) * sd_hat / sqrt(B))
```

```
## [1] -0.2652679 -0.2635419
```

2.3 The parameter of interest, second pass. Suppose we know beforehand that O drawn from P_0 takes its values in $\mathcal{O} \equiv [0, 1] \times \{0, 1\} \times [0, 1]$ and that $P_0(A = 1|W)$ is bounded away from zero and one $Q_{0,W}$-almost surely (this is the case indeed). Then we can define model $\mathcal{M}$ as the set of all laws P on $\mathcal{O}$ such that $\bar{G}(W) \equiv P(A = 1|W)$ is bounded away from zero and one Q_W-almost surely, where Q_W is the marginal distribution of W under P.

Let us also define generically $\bar{Q}$ as

$$\bar{Q}(A, W) \equiv E_P(Y|A, W).$$

Central to our approach is viewing ψ_0 as the value at P_0 of the statistical mapping Ψ from $\mathcal{M}$ to $[0, 1]$ characterized by

$$\begin{aligned}\Psi(P) &\equiv E_P\left(\bar{Q}(1,W) - \bar{Q}(0,W)\right)\\ &= \int \left(\bar{Q}(1,w) - \bar{Q}(0,w)\right) dQ_W(w),\end{aligned}$$

a clear extension of (2). For instance, although the law $\Pi_0 \in \mathcal{M}$ encoded by default (*i.e.*, with `h=0`) in `drawFromAnotherExperiment` defined below differs starkly from P_0,

```r
draw_from_another_experiment <- function(n, h = 0) {
  ## preliminary
  n <- Arguments$getInteger(n, c(1, Inf))
  h <- Arguments$getNumeric(h)
  ## ## 'Gbar' and 'Qbar' factors
  Gbar <- function(W) {
    sin((1 + W) * pi / 6)
  }
  Qbar <- function(AW, hh = h) {
    A <- AW[, 1]
    W <- AW[, 2]
    expit( logit( A *  W + (1 - A) * W^2 ) +
           hh * 10 * sqrt(W) * A )
  }
  ## sampling
  ## ## context
  W <- runif(n, min = 1/10, max = 9/10)
  ## ## action undertaken
  A <- rbinom(n, size = 1, prob = Gbar(W))
  ## ## reward
  shape1 <- 4
  QAW <- Qbar(cbind(A, W))
  Y <- rbeta(n, shape1 = shape1, shape2 = shape1 * (1 - QAW) / QAW)
  ## ## observation
  obs <- cbind(W = W, A = A, Y = Y)
  attr(obs, "Gbar") <- Gbar
  attr(obs, "Qbar") <- Qbar
  attr(obs, "QW") <- function(x){dunif(x, min = 1/10, max = 9/10)}
  attr(obs, "shape1") <- shape1
  ##
  return(obs)
}
```

parameter $\Psi(\Pi_0)$ is well defined, and numerically approximated by `psi.Pi.zero` in the following chunk of code:

```r
five_obs_from_another_experiment <- draw_from_another_experiment(5)
integrand <- function(w) {
  Qbar <- attr(five_obs_from_another_experiment, "Qbar")
  QW <- attr(five_obs_from_another_experiment, "QW")
  ( Qbar(cbind(1, w)) - Qbar(cbind(0, w)) ) * QW(w)
}
(psi_Pi_zero <- integrate(integrand, lower = 0, upper = 1)$val)
```

```
## [1] 0.1966687
```

(easy algebra reveals that $\Psi(\Pi_0) = 59/300$ indeed).

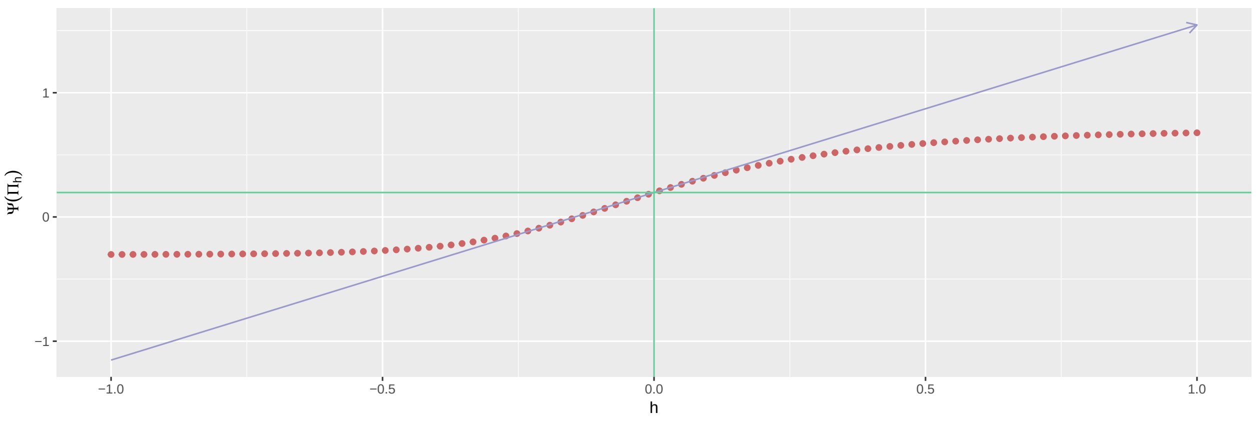

Figure 1: Evolution of statistical parameter Ψ along fluctuation $\{\Pi_h : h \in H\}$.

2.4 Being smooth, first pass. Luckily, the statistical mapping Ψ is well behaved, or smooth. Here, this colloquial expression refers to the fact that, for each $P \in \mathcal{M}$, if $P_h \to_h P$ in $\mathcal{M}$ from a direction s when the real parameter $h \to 0$, then not only $\Psi(P_h) \to_h \Psi(P)$ (continuity), but also $h^{-1}[\Psi(P_h) - \Psi(P)] \to_h c$, where the real number c depends on P and s (differentiability).

For instance, let $\Pi_h \in \mathcal{M}$ be the law encoded in `draw_from_another_experiment` with `h` ranging over $[-1, 1]$. We will argue shortly that $\Pi_h \to_h \Pi_0$ in $\mathcal{M}$ from a direction s when $h \to 0$. The following chunk of code evaluates and represents $\Psi(\Pi_h)$ for h ranging in a discrete approximation of $[-1, 1]$:

```r
approx <- seq(-1, 1, length.out = 1e2)
psi_Pi_h <- sapply(approx, function(t) {
  obs_from_another_experiment <- draw_from_another_experiment(1, h = t)
  integrand <- function(w) {
    Qbar <- attr(obs_from_another_experiment, "Qbar")
    QW <- attr(obs_from_another_experiment, "QW")
    ( Qbar(cbind(1, w)) - Qbar(cbind(0, w)) ) * QW(w)
  }
  integrate(integrand, lower = 0, upper = 1)$val
})
slope_approx <- (psi_Pi_h - psi_Pi_zero) / approx
slope_approx <- slope_approx[min(which(approx > 0))]
ggplot() +
  geom_point(data = data.frame(x = approx, y = psi_Pi_h), aes(x, y),
             color = "#CC6666") +
  geom_segment(aes(x = -1, y = psi_Pi_zero - slope_approx,
                   xend = 1, yend = psi_Pi_zero + slope_approx),
               arrow = arrow(length = unit(0.03, "npc")),
               color = "#9999CC") +
  geom_vline(xintercept = 0, color = "#66CC99") +
  geom_hline(yintercept = psi_Pi_zero, color = "#66CC99") +
  labs(x = "h", y = expression(Psi(Pi[h])))
```

The dotted curve represents the function $h \mapsto \Psi(\Pi_h)$. The blue line represents the tangent to the previous curve at $h = 0$, which is indeed differentiable around $h = 0$. It is derived by simple geometric arguments. In the next subsection, we formalize what it means to be smooth for the statistical mapping Ψ. Once the presentation is complete, we will be able to derive a closed-form expression for the slope of the blue curve from the chunk of code where `draw_from_another_experiment` is defined.

2.5 Being smooth, second pass. Let us now describe what it means for statistical mapping Ψ to be smooth at every $P \in \mathcal{M}$. The description necessitates the introduction of fluctuations.

For every direction[*] $s : \mathcal{O} \to \mathbb{R}$ such that $s \neq 0$[†], $E_P(s(O)) = 0$ and s bounded by, say, M, for every $h \in H \equiv]-M^{-1}, M^{-1}[$, we can define a law $P_h \in \mathcal{M}$ by setting $P_h \ll P$[‡] and

$$\frac{dP_h}{dP}(O) \equiv 1 + hs(O), \tag{3}$$

that is, P_h has density $(1 + hs)$ with respect to (w.r.t.) P. We call $\{P_h : h \in H\}$ a fluctuation of P in direction s because

$$(i)\ P_h|_{h=0} = P, \quad (ii)\ \left.\frac{d}{dh} \log \frac{dP_h}{dP}(O)\right|_{h=0} = s(O). \tag{4}$$

The fluctuation is a one-dimensional parametric submodel of $\mathcal{M}$.

Statistical mapping Ψ is smooth at every $P \in \mathcal{M}$ because, for each $P \in \mathcal{M}$, there exists a so called efficient influence curve[§] $D^*(P) : \mathcal{O} \to \mathbb{R}$ such that $E_P(D^*(P)(O)) = 0$ and, for any direction s as above, if $\{P_h : h \in H\}$ is defined as in (3), then the real-valued mapping $h \mapsto \Psi(P_h)$ is differentiable at $h = 0$, with a derivative equal to

$$E_P\left(D^*(P)(O)s(O)\right). \tag{5}$$

Interestingly, if a fluctuation $\{P_h : h \in H\}$ satisfies (4) for a direction s such that $s \neq 0$, $E_P(s(O)) = 0$ and $\mathrm{Var}_P(s(O)) < \infty$, then $h \mapsto \Psi(P_h)$ is still differentiable at $h = 0$ with a derivative equal to (5) (beyond fluctuations of the form (3)).

The influence curves $D^*(P)$ convey valuable information about Ψ. For instance, an important result from the theory of inference based on semiparametric models guarantees that if ψ_n is a regular[¶] estimator of $\Psi(P)$ built from n independent observations drawn from P, then the asymptotic variance of the centered and rescaled $\sqrt{n}(\psi_n - \Psi(P))$ cannot be smaller than the variance of the P-specific efficient influence curve, that is,

$$\mathrm{Var}_P(D^*(P)(O)). \tag{6}$$

In this light, an estimator ψ_n of $\Psi(P)$ is said *asymptotically efficient* at P if it is regular at P and such that $\sqrt{n}(\psi_n - \Psi(P))$ converges in law to the centered Gaussian law with variance (6), which is called the Cramér-Rao bound.

2.6 The efficient influence curve. It is not difficult to check (do we give the proof?) that the efficient influence curve $D^*(P)$ of Ψ at $P \in \mathcal{M}$ writes as $D^*(P) \equiv D_1^*(P) + D_2^*(P)$ where $D_1^*(P)$ and $D_2^*(P)$ are given by

[*]A direction is a measurable function.

[†]That is, $s(O)$ is not equal to zero P-almost surely.

[‡]That is, P_h is dominated by P: if an event A satisfies $P(A) = 0$, then necessarily $P_h(A) = 0$ too.

[§]It is a measurable function.

[¶]We can view ψ_n as the by product of an algorithm $\widehat{\Psi}$ trained on independent observations $O_1, \ldots, O_n$ drawn from P. The estimator is regular at P (w.r.t. the maximal tangent space) if, for any direction $s \neq 0$ such that $E_P(s(O)) = 0$ and $\mathrm{Var}_P(s(O)) < \infty$ and fluctuation $\{P_h : h \in H\}$ satisfying (4), the estimator $\psi_{n,1/\sqrt{n}}$ of $\Psi(P_{1/\sqrt{n}})$ obtained by training $\widehat{\Psi}$ on independent observations $O_1, \ldots, O_n$ drawn from $P_{1/\sqrt{n}}$ is such that $\sqrt{n}(\psi_{n,1/\sqrt{n}} - \Psi(P_{1/\sqrt{n}}))$ converges in law to a limit that does not depend on s.

$$D_1^*(P)(O) \equiv \bar{Q}(1,W) - \bar{Q}(0,W) - \Psi(P),$$
$$D_2^*(P)(O) \equiv \frac{2A-1}{\ell\bar{G}(A,W)}(Y - \bar{Q}(A,W)),$$

with shorthand notation $\ell\bar{G}(A,W) \equiv A\bar{G}(W) + (1-A)(1-\bar{G}(W))$. The following chunk of code enables the computation of the values of the efficient influence curve $D^*(P)$ at observations drawn from P (note that it is necessary to provide the value of $\Psi(P)$, or a numerical approximation thereof, through argument `psi`).

```
eic <- function(obs, psi) {
  Qbar <- attr(obs, "Qbar")
  Gbar <- attr(obs, "Gbar")
  QAW <- Qbar(obs[, c("A", "W")])
  gW <- Gbar(obs[, "W"])
  lgAW <- obs[, "A"] * gW + (1 - obs[, "A"]) * (1 - gW)
  ( Qbar(cbind(1, obs[, "W"])) - Qbar(cbind(0, obs[, "W"])) - psi ) +
    (2 * obs[, "A"] - 1) / lgAW * (obs[, "Y"] - QAW)
}

(eic(five_obs, psi = psi_hat))
```

```
## [1] -0.7207719  0.4762960 -0.4398213  0.3962634  0.6449063
```

```
(eic(five_obs_from_another_experiment, psi = psi_Pi_zero))
```

```
## [1]  0.02107056 -0.00342964  0.10731746  0.07596022  0.05989993
```

2.7 Computing and comparing Cramér-Rao bounds. We can use `eic` to numerically approximate the Cramér-Rao bound at P_0:

```
obs <- draw_from_experiment(B)
(cramer_rao_hat <- var(eic(obs, psi = psi_hat)))
```

```
## [1] 0.3652997
```

and the Cramér-Rao bound at Π_0:

```
obs_from_another_experiment <- draw_from_another_experiment(B)
(cramer_rao_Pi_zero_hat <- var(eic(obs_from_another_experiment, psi = 59/300)))
```

```
## [1] 0.09574321
```

```
(ratio <- sqrt(cramer_rao_Pi_zero_hat/cramer_rao_hat))
```

```
## [1] 0.5119521
```

We thus discover that of the statistical parameters $\Psi(P_0)$ and $\Psi(\Pi_0)$, the latter is easier to target than the former. Heuristically, for large sample sizes, the narrowest (efficient) confidence intervals for $\Psi(\Pi_0)$ are approximately 0.51 (rounded to two decimal places) smaller than their counterparts for $\Psi(P_0)$.

2.8 Revisiting Section 2.4. It is not difficult either (though a little cumbersome) (do we give the proof? I'd rather not) to verify that $\{\Pi_h : h \in [-1,1]\}$ is a fluctuation of Π_0 in the direction of σ_0 (in the sense of (3)) given, up to a constant, by

$$\sigma_0(O) \equiv -10\sqrt{W}A \times \beta_0(A,W)\left(\log(1-Y) + \sum_{k=0}^{3}(k+\beta_0(A,W))^{-1}\right) + \text{constant},$$

where $\beta_0(A,W) \equiv \dfrac{1-\bar{Q}_{\Pi_0}(A,W)}{\bar{Q}_{\Pi_0}(A,W)}$.

Consequently, the slope of the dotted curve in Figure 1 is equal to

$$E_{\Pi_0}(D^*(\Pi_0)(O)\sigma_0(O)) \tag{7}$$

(since $D^*(\Pi_0)$ is centered under Π_0, knowing σ_0 up to a constant is not problematic).

Let us check this numerically. In the next chunk of code, we implements direction s with `s_draw_from_another_experiment`, then we numerically approximate (7) (pointwise and with a confidence interval of asymptotic level 95%):

```
s_draw_from_another_experiment <- function(obs) {
  ## preliminary
  Qbar <- attr(obs, "Qbar")
  QAW <- Qbar(obs[, c("A", "W")])
  shape1 <- Arguments$getInteger(attr(obs, "shape1"), c(1, Inf))
  ## computations
  betaAW <- shape1 * (1 - QAW) / QAW
  out <- log(1 - obs[, "Y"])
  for (int in 1:shape1) {
    out <- out + 1/(int - 1 + betaAW)
  }
  out <- - out * shape1 * (1 - QAW) / QAW * 10 * sqrt(obs[, "W"]) * obs[, "A"]
  ## no need to center given how we will use it
  return(out)
}

vars <- eic(obs_from_another_experiment, psi = 59/300) *
  s_draw_from_another_experiment(obs_from_another_experiment)
sd_hat <- sd(vars)
(slope_hat <- mean(vars))
```

```
## [1] 1.358524
```

```
(slope_CI <- slope_hat + c(-1, 1) * qnorm(1 - alpha / 2) * sd_hat / sqrt(B))
```

```
## [1] 1.353257 1.363791
```

Equal to 1.349 (rounded to three decimal places), the first numerical approximation `slope_approx` is not too off.

2.9 Double-robustness The efficient influence curve $D^*(P)$ at $P \in \mathcal{M}$ enjoys another remarkable property: it is double-robust. Specifically, for all $P' \in \mathcal{M}$, it holds that

$$\Psi(P') - \Psi(P) = -E_P(D^*(P')(O)) + \mathrm{Rem}_P(\bar{Q}', \bar{G}') \tag{8}$$

where the so called remainder term $\mathrm{Rem}_P(\bar{Q}', \bar{G}')$ satisfies[‖]

[‖]For any (measurable) $f : \mathcal{O} \to \mathbb{R}$, we denote $\|f\|_P = E_P(f(O)^2)^{1/2}$.

$$\mathrm{Rem}_P(\bar{Q}', \bar{G}')^2 \leq \|\bar{Q}' - \bar{Q}\|_P^2 \times \|(\bar{G}' - \bar{G})/\ell\bar{G}'\|_P^2. \tag{9}$$

In particular, if

$$E_P(D^*(P')(O)) = 0, \tag{10}$$

and *either* $\bar{Q}' = \bar{Q}$ *or* $\bar{G}' = \bar{G}$, then $\mathrm{Rem}_P(\bar{Q}', \bar{G}') = 0$ hence $\Psi(P') = \Psi(P)$. In words, if P' solves the so called P-specific efficient influence curve equation (10) and if, in addition, P' has the same $\bar{Q}$-component or $\bar{G}$-component as P, then $\Psi(P') = \Psi(P)$ no matter how P' may differ from P otherwise. This property is useful to build consistent estimators of $\Psi(P)$.

However, there is much more to double-robustness than the above straightforward implication. Indeed, 8 is useful to build a consistent etimator of $\Psi(P)$ that, in addition, satisfies a central limit theorem and thus lends itsef to the construction of confidence intervals.

Let $P_n^0 \in \mathcal{M}$ be an element of model $\mathcal{M}$ of which the choice is data-driven, based on observing n independent draws from P. Equality 8 reveals that the statistical behavior of the corresponding *substitution* estimator $\psi_n^0 \equiv \Psi(P_n^0)$ is easier to analyze when the remainder term $\mathrm{Rem}_P(\bar{Q}_n^0, \bar{G}_n^0)$ goes to zero at a fast (relative to n) enough rate. In light of 9, this happens if the features $\bar{Q}_n^0$ and $\bar{G}_n^0$ of P_n^0 converge to their counterparts under P at rates of which *the product* is fast enough.

2.10 Targeted inference. blabla

```r
Gbar <- attr(obs, "Gbar")

psi_hat_ab <- obs %>% as_tibble() %>% mutate(id = 1:nrow(obs) %% 1e3) %>%
  mutate(lgAW = A * Gbar(W) + (1 - A) * (1 - Gbar(W))) %>% group_by(id) %>%
  summarize(est_a = mean(Y[A==1]) - mean(Y[A==0]),
            est_b = mean(Y * (2 * A - 1) / lgAW),
            std_b = sd(Y * (2 * A - 1) / lgAW),
            clt_b = sqrt(n()) * (est_b - psi_hat) / std_b)
std_a <- sd(psi_hat_ab$est_a)
psi_hat_ab <- psi_hat_ab %>%
  mutate(std_a = std_a,
         clt_a = (est_a - psi_hat) / std_a) %>%
  gather(key, value, -id) %>%
  extract(key, c("what", "type"), "([^_]+)_([ab])") %>%
  spread(what, value)

bias <- psi_hat_ab %>% group_by(type) %>% summarise(bias = mean(clt))

ggplot(psi_hat_ab, aes(clt, fill = type, colour = type)) +
  geom_density(alpha = 0.1) +
  geom_vline(aes(xintercept = bias, colour = type), bias) +
  labs(x = expression(paste(sqrt(n)*(psi[n]^{list(a, b)} - psi[0]))))
```

blabla

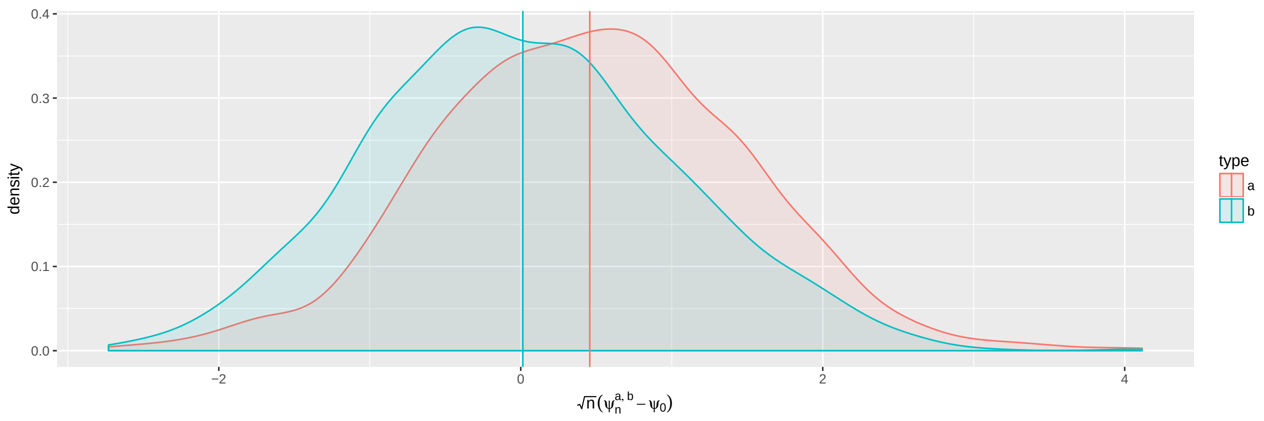

Figure 2: Of two estimators of ψ_0, one of them misconceived, the other assuming that $\bar{G}_0$ is known.