

A guided tour in targeted learning territory

David Benkeser, Antoine Chambaz, Nima Hejazi

10/11/2017

Contents

1	Introduction	1
2	A simulation study	1
2.1	Reproducible experiment as a law.	1
2.2	The parameter of interest, first pass.	3
2.3	The parameter of interest, second pass.	4

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.

Use sectioning a lot to ease cross-referencing.

```
set.seed(54321) ## because reproducibility matters...
suppressMessages(library(R.utils)) ## 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:

```
drawFromExperiment <- function(n, full = FALSE) {
  ## preliminary
  n <- Arguments$getInteger(n, c(1, Inf))
  full <- Arguments$getLogical(full)
  ## ## 'gbar' and 'Qbar' factors
  gbar <- function(W) {
    expit(-0.3 + 2 * W - 1.5 * W^2)
```

```

}
Qbar <- function(AW) {
  A <- AW[, 1]
  W <- AW[, 2]
  A * cos(2 * pi * W) + (1 - A) * sin(2 * pi * W^2)
}
## sampling
## ## context
W <- runif(n)
## ## counterfactual rewards
zeroW <- cbind(A = 0, W)
oneW <- cbind(A = 1, W)
Yzero <- rnorm(n, mean = Qbar(zeroW), sd = 1)
Yone <- rnorm(n, mean = Qbar(oneW), sd = 1)
## ## 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 `drawFromExperiment` 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:

```

five.obs <- drawFromExperiment(5)
five.obs

##           W A           Y
## [1,] 0.4290078 1 -0.4410530
## [2,] 0.4984304 0  0.1905250
## [3,] 0.1766923 0 -0.1325567
## [4,] 0.2743935 1 -1.6415356
## [5,] 0.2165102 0  0.6829717
## attr(,"gbar")
## function (W)
## {
##   expit(-0.3 + 2 * W - 1.5 * W^2)
## }
## <bytecode: 0x3ed5508>
## <environment: 0x11dee08>
## attr(,"Qbar")
## function (AW)
## {

```

```
##      A <- AW[, 1]
##      W <- AW[, 2]
##      A * cos(2 * pi * W) + (1 - A) * sin(2 * pi * W^2)
## }
## <bytecode: 0x42947d0>
## <environment: 0x11dee08>
## attr("QW")
## function (x, min = 0, max = 1, log = FALSE)
## .Call(C_dunif, x, min, max, log)
## <bytecode: 0x4232218>
## <environment: namespace:stats>
```

The `attributes` of the object `obs` are visible 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

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

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:

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

Acting as oracles, we can compute explicitly 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 `drawFormExperiment` is useful to ram home what it means to run the “full” experiment by setting argument `full` to `TRUE` in a call to `drawFormExperiment`. 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

$$\psi_0 = E_{P_0}(Y_1 - Y_0).$$

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 `drawFromExperiment` 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 :

```
B <- 1e6
full.obs <- drawFromExperiment(B, full = TRUE)
psi.hat <- mean(full.obs[, "Yone"] - full.obs[, "Yzero"])
psi.hat
```

```
## [1] -0.170182
```

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

```
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)
psi.CI
```

```
## [1] -0.1738905 -0.1664735
```

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

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

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 ,

```
drawFromAnotherExperiment <- 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 <- rbeta(n, shape1 = 2, shape2 = 2)
  ## ## action undertaken
  A <- rbinom(n, size = 1, prob = gbar(W))
  ## ## reward
  QAW <- Qbar(cbind(A, W))
  Y <- rbeta(n, shape1 = 1, shape2 = (1 - QAW) / QAW)
  ## ## observation
  obs <- cbind(W = W, A = A, Y = Y)
  attr(obs, "gbar") <- gbar
  attr(obs, "Qbar") <- Qbar
  attr(obs, "QW") <- dunif
  ##
  return(obs)
}

```

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

```

obs.from.another.experiment <- drawFromAnotherExperiment(1)
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)
}
psi.Pi.zero <- integrate(integrand, lower = 0, upper = 1)$val
psi.Pi.zero

```

```
## [1] 0.1666667
```

(easy algebra reveals that $\Psi(\Pi_0) = 1/6$ indeed).

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 $P_h \rightarrow_h P$ in \mathcal{M} from a direction s when the real parameter $h \rightarrow 0$, then not only $\Psi(P_h) \rightarrow_h \Psi(P)$ (continuity), but also $h^{-1}[\Psi(P_h) - \Psi(P)] \rightarrow_h c$, where the real number c depends on P and s (differentiability).

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

```

approx <- seq(-1, 1, length.out = 1e2)
psi.Pi.h <- sapply(approx, function(t) {
  obs.from.another.experiment <- drawFromAnotherExperiment(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 <- (psi.Pi.h - psi.Pi.zero) / approx
slope <- slope[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, xend = 1, yend = psi.Pi.zero + slope),
            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])))

```

