

Causal biomarker discovery

Practical session · Causality in Biomedicine 2026 (EMBL-EBI)

2026-10-07

Table of contents

Day 3 Practical Session	4
The two parts	4
Notation	4
Data, model, and parameters	5
Estimators, nuisance parameters, and variance estimation	5
How the session works	5
Setup	6
1 Biomarker discovery as causal effect estimation	7
1.1 The data: smoking and DNA methylation	7
1.2 The target parameter	9
1.2.1 Exercise 1.1 	9
1.3 Efficient estimators: the one-step estimator and TMLE	10
1.3.1 A synthetic CpG with a known effect	11
1.3.2 Exercise 1.2 	12
1.3.3 Exercise 1.3 	14
1.4 Back to the real data	16
1.4.1 Exercise 1.4 	17
1.4.2 Exercise 1.5 	19
1.5 Variance moderation	19
1.5.1 Exercise 1.6 	20
1.5.2 Exercise 1.7 	21
1.5.3 Exercise 1.8  (<i>stretch</i>)	23
2 Stochastic-interventional vaccine efficacy	25
2.1 The question	26
2.2 The data: a simulated trial with a case-cohort design	26
2.3 The target parameter	28
2.3.1 Exercise 2.1 	29
2.4 Estimation with <code>lmt</code>	30
2.4.1 Exercise 2.2 	31
2.4.2 Exercise 2.3 	33
2.4.3 Exercise 2.4 	34
2.4.4 Exercise 2.5 	37
2.4.5 Exercise 2.6  (<i>stretch</i>)	38

2.5 Wrap-up	40
References	41

Day 3 Practical Session

This page accompanies the practical session on *causal biomarker discovery from expression data* (Wednesday, 7 October 2026) in the EMBO course [Causality in Biomedicine](#).

Earlier today, you heard two lectures: one on causal effects of *modifying* a continuous mediator, applied to immune correlates of protection in vaccine trials (slides [here](#)), and one on biomarker discovery with causal inference and machine learning (slides [here](#)). This session revisits the key ideas of both lectures, hands-on. We will **build each method from its parts**: you have all the pieces from earlier in the week (structural causal models, identification, model misspecification, targeted learning).

The two parts

1. **Biomarker discovery as causal effect estimation.** Which CpG sites in whole blood are affected by smoking? We will reconstruct the `biotmle` procedure: an efficient, machine-learning-based estimator of the average treatment effect (ATE) for each CpG, combined with the variance moderation that makes `limma` useful. Real data from Su et al. (2016), as analyzed in Hejazi, Boileau, et al. (2023).
2. **Stochastic-interventional vaccine efficacy (SVE).** How would vaccine efficacy change if a vaccine induced higher (or lower) antibody titers? We will estimate vaccine efficacy under a *modified treatment policy* (MTP) that shifts an immune marker, in a simulated vaccine efficacy trial.

A theme runs through both: each causal parameter we target **reduces to a familiar coefficient of a linear model** when that model is correct. The linear model is more accessible, but more brittle; the exercises ask you to see exactly where it breaks.

In Part 1, the biomarker is an *outcome* of the exposure. In Part 2, the biomarker lies *on the causal pathway* between vaccination and infection: it is a mediator.

Notation

We use the same notation in both parts, close to that of the lectures; each part also has a table of its own symbols.

Data, model, and parameters

- **Data and variables.** O is the data on one study unit (e.g., a patient in a clinical trial); we assume $O \sim P_0$, the true, unknown data-generating distribution, and that we observe n independent copies $O_1, \dots, O_n$, whose empirical distribution we denote P_n . In the simplest case, $O = (L, A, Y)$, with baseline covariates L , an exposure $A \in \{0, 1\}$, and an outcome Y .
- **Statistical model.** $\mathcal{P}$ is a collection of candidate probability distributions among which P_0 falls. Throughout, we assume $\mathcal{P}$ to be *nonparametric*, that is, we assume nothing about P_0 beyond what the study design guarantees.
- **Causal parameters.** A causal parameter is a contrast of counterfactual outcomes, $Y(a)$, for example, the ATE, $\mathbb{E}[Y(1) - Y(0)]$. Under assumptions, a causal parameter is *identified* by a *statistical* (or observed-data) parameter.
- **Statistical parameter:** A functional of the data-generating distribution P_0 , a statistical parameter is a mapping Ψ from the statistical model, $\mathcal{P}$, to an outcome space (e.g., $\mathbb{R}$), with true value $\psi_0 = \Psi(P_0)$. We use a naught subscript to denote a quantity with respect to P_0 , e.g., $\mathbb{E}_0$ and $\mathbb{V}_0$ for the mean and variance under P_0 .

Estimators, nuisance parameters, and variance estimation

- **Nuisance parameters.** Ψ depends on P_0 through simpler pieces that are not of interest in themselves but must be estimated (with GLMs or the Super Learner): the *propensity score* $g_0(L) = P_0(A = 1 \mid L)$ and the *outcome regression* $\bar{Q}_0(A, L) = \mathbb{E}_0[Y \mid A, L]$.
- **Estimators.** A subscript n marks an estimate from the n observations (e.g., $g_n, \bar{Q}_n, \psi_n$). A *plug-in* estimate substitutes initial nuisance estimates into Ψ ; the *one-step* estimator ψ_n^+ adds the empirical mean of the estimated efficient influence function to it; and the *TMLE* $\psi_n^* = \Psi(P_n^*)$ plugs in a *targeted* estimate, marked with a star (e.g., $\bar{Q}_n^*$). We write $\text{expit}(x) = 1/(1 + e^{-x})$, with inverse logit.
- **Variance estimation.** $D(O; P)$ is the *efficient influence function* (EIF) of Ψ at P . Its variance, $\sigma_0^2 = \mathbb{V}_0[D(O; P_0)]$, is the smallest asymptotic variance of a regular estimator; its empirical variance at the estimates, σ_n^2 , gives the standard error $\sqrt{\sigma_n^2/n}$ of an efficient estimator.

How the session works

Exercises come in two kinds:

-  **pen and paper:** short derivations or explanations, and
-  **coding:** fill in the blanks (___) in the code provided, then run it.

Work in pairs if you like. Course trainers will circulate. Please ask questions! Exercises marked *stretch* are for those who finish early.

Setup

All code should run on the course VM. Open RStudio, start a new script, and run:

```
library(SuperLearner)
library(lmtp)
library(ggplot2)

# helper functions and data for this session
source("practical/R/utils.R")
source("practical/R/moderate.R")
source("practical/R/learners.R")
source("practical/R/sim_cpg.R")
source("practical/R/sim_vaccine_trial.R")
methy <- read.csv("practical/data/su2016_methylation.csv.gz")
```

The data and helper scripts can also be downloaded directly: [su2016_methylation.csv.gz](#), [utils.R](#), [moderate.R](#), [learners.R](#), [sim_cpg.R](#), [sim_vaccine_trial.R](#).

1 Biomarker discovery as causal effect estimation

i In a nutshell

We rebuild the `biotmle` recipe from its parts. First, we estimate the average treatment effect (ATE) of smoking on methylation at each of many CpG sites by targeted maximum likelihood (TMLE) with machine learning. Then we test all CpGs with variance moderation.

Roadmap: (1) the target parameter, and when a regression coefficient equals it $\rightarrow$ (2) TMLE by hand, on a synthetic CpG with a known effect $\rightarrow$ (3) misspecified GLMs, correct GLMs, and Super Learner $\rightarrow$ (4) the real data, one CpG and then all of them $\rightarrow$ (5) variance moderation and multiple testing.

i Notation for this part

Symbol	Meaning
L	sex, age, race, and leukocyte proportions
A	smoking status (1 = ever, 0 = never)
$Y_b, b = 1, \dots, B$	methylation (M-value) at CpG site b
$\bar{Q}_{0,b}(A, L)$	outcome regression for CpG b
$\psi_{0,b} = \Psi_b(P_0)$	ATE of smoking on CpG b
$H_0(A, L)$	clever covariate, $A/g_0(L) - (1 - A)/(1 - g_0(L))$
$\sigma_{n,b}^2, \tilde{\sigma}_{n,b}^2$	EIF variance for CpG b , before and after moderation

1.1 The data: smoking and DNA methylation

Su et al. (2016) measured DNA methylation in whole blood, at roughly 470,000 CpG sites, in 253 people: 172 who had ever smoked and 81 who never had. We use their data to ask: **which CpG sites are affected by smoking?**

The data on one person are $O = (L, A, Y) \sim P_0 \in \mathcal{P}$, where P_0 is the true, unknown data-generating distribution and $\mathcal{P}$ is the nonparametric statistical model; we observe n i.i.d. copies $O_1, \dots, O_n$. Here,

- L are baseline covariates: sex, age, race, and the estimated proportions of six leukocyte types (CD8T, CD4T, NK, Bcell, Mono, Gran). Blood cell composition differs by smoking status and strongly affects methylation, so it is a confounder;
- $A \in \{0, 1\}$ is smoking status (**smoker**: 1 = ever, 0 = never);
- $Y = (Y_b : b = 1, \dots, B)$ are the methylation levels (M-values) at each CpG site b .

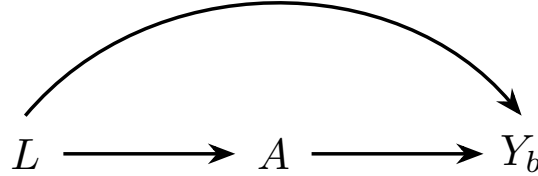


Figure 1.1: DAG for one CpG site b , in time order: covariates L confound the effect of smoking A on methylation Y_b .

To keep computation fast, the file contains $B = 300$ CpG sites: 150 chosen at random among those flagged by a `limma` screen in the full data (including four well-known smoking-related CpGs, e.g., `cg05575921` in the *AHRR* gene), and 150 chosen at random among those that were not.

```

covars <- c("sex", "age", "race", "CD8T", "CD4T", "NK", "Bcell", "Mono", "Gran")
cpgs <- grep("^cg", names(methy), value = TRUE)
A <- methy$smoker
L <- methy[, covars]
n <- nrow(methy)

dim(methy)

```

```
[1] 253 311
```

```
table(smoker = A)
```

```

smoker
 0    1
81 172

```

1.2 The target parameter

Let $g_0(L) = P_0(A = 1 | L)$ be the propensity score and $\bar{Q}_{0,b}(A, L) = \mathbb{E}_0[Y_b | A, L]$ the outcome regression for CpG b . For each CpG, our target is the ATE of smoking on methylation, which (under no unmeasured confounding and positivity) is identified by

$$\psi_{0,b} = \Psi_b(P_0) = \mathbb{E}_0[\bar{Q}_{0,b}(1, L) - \bar{Q}_{0,b}(0, L)].$$

This is a *variable importance measure*: it ranks CpG sites by how much smoking would change their methylation, without committing to a model for $\bar{Q}_{0,b}$. Throughout, a subscript n denotes an estimate (e.g., g_n , $\bar{Q}_{n,b}$, $\psi_{n,b}$).

1.2.1 Exercise 1.1

i When is a regression coefficient the ATE?

The usual approach (as in `limma`) fits, for each CpG, the working linear model $\bar{Q}_{0,b}(a, l) = \beta_0 + \beta_1 a + \gamma^\top l$, and tests $\beta_1 = 0$.

- Suppose this linear model is correct. Show that $\psi_{0,b} = \beta_1$.
- Now suppose the effect of smoking varies with age, so that the correct model is $\bar{Q}_{0,b}(a, l) = \beta_0 + \beta_1 a + \beta_2 a \cdot \text{age} + \gamma^\top l$. What is $\psi_{0,b}$ now? Is β_1 still the ATE?
- In (b), what would β_1 in the *main-terms* model of (a) estimate? Is there a study design under which it would still estimate the ATE?

💡 Solution

- $\bar{Q}_{0,b}(1, l) - \bar{Q}_{0,b}(0, l) = (\beta_0 + \beta_1 + \gamma^\top l) - (\beta_0 + \gamma^\top l) = \beta_1$ for every l , so averaging over L gives $\psi_{0,b} = \beta_1$.
- Now $\bar{Q}_{0,b}(1, l) - \bar{Q}_{0,b}(0, l) = \beta_1 + \beta_2 \text{age}$, so $\psi_{0,b} = \beta_1 + \beta_2 \mathbb{E}_0[\text{age}]$. The coefficient β_1 is the effect at age 0, which is rarely of scientific interest.
- The main-terms coefficient converges to (roughly) a weighted average of the age-specific effects, with weights proportional to the conditional variance of smoking, $\mathbb{V}_0(A | L)$: people whose smoking status is hard to predict from L count more. It is not $\psi_{0,b}$ in general. If A were randomized ($A \perp\!\!\!\perp L$), the main-terms coefficient would be consistent for the ATE even though the model is wrong. Smoking is not randomized, so here we need an estimator whose validity does not hinge on the form of $\bar{Q}_{0,b}$.

Take-away: $\psi_{0,b}$ is defined without reference to any model; β_1 is $\psi_{0,b}$ only if the linear model is right (or A is randomized).

1.3 Efficient estimators: the one-step estimator and TMLE

From the lecture, the efficient influence function (EIF) of $\psi_{0,b}$ is

$$D_b(O; P_0) = H_0(A, L)(Y_b - \bar{Q}_{0,b}(A, L)) + \bar{Q}_{0,b}(1, L) - \bar{Q}_{0,b}(0, L) - \psi_{0,b} ,$$

$$H_0(A, L) = \frac{A}{g_0(L)} - \frac{1 - A}{1 - g_0(L)} .$$

Given estimates g_n and $\bar{Q}_{n,b}$, the *one-step* estimator corrects the plug-in estimate by the average of the estimated EIF:

$$\psi_{n,b}^+ = \frac{1}{n} \sum_{i=1}^n \{ \bar{Q}_{n,b}(1, L_i) - \bar{Q}_{n,b}(0, L_i) \} + \frac{1}{n} \sum_{i=1}^n H_n(A_i, L_i)(Y_{b,i} - \bar{Q}_{n,b}(A_i, L_i)).$$

The correction removes the first-order bias of the plug-in, but the result can fall outside the parameter's range.

TMLE removes the same bias differently: it *updates* the initial fit $\bar{Q}_{n,b}$ to a targeted fit $\bar{Q}_{n,b}^*$ chosen so that the average of the estimated EIF is zero, and then plugs it in,

$$\psi_{n,b}^* = \frac{1}{n} \sum_{i=1}^n \{ \bar{Q}_{n,b}^*(1, L_i) - \bar{Q}_{n,b}^*(0, L_i) \}.$$

TMLE is a **substitution (plug-in) estimator**, $\psi_{n,b}^* = \Psi_b(P_n^*)$: it is the value of the target parameter at an estimate of the data distribution, so it always respects the parameter's bounds. Respecting the bounds also tends to make TMLE more stable in finite samples. It shares the one-step estimator's large-sample properties: it is efficient, its standard error is $\sqrt{\sigma_{n,b}^2/n}$ with $\sigma_{n,b}^2$ the empirical variance of the estimated EIF, and it is *doubly robust*: consistent if either g_n or $\bar{Q}_{n,b}$ is consistent.

For a continuous outcome, the update is made on the unit scale, so that the targeted fit respects the observed range of the outcome (Gruber and van der Laan 2010). The helper functions in `practical/R/utis.R` carry out the parts of each step that are not about TMLE itself:

1. Scale Y_b to $[0, 1]$, using its observed range.
 - *Helper*: `scale_to_unit()`.
2. Estimate g_0 and (on the unit scale) $\bar{Q}_{0,b}$. *Helpers*: `estimate_g()` and `estimate_Q()` fit the nuisance parameters with `SuperLearner()`, using the library of learners specified. `estimate_g()` passes its predictions through `bound_propensity()`, which truncates predicted probabilities away from 0 and 1, to avoid instability in H_n (note: truncation does trade a bit of bias for improved variance). `bound_precision()` keeps the predicted outcomes strictly inside $(0, 1)$, so that their logits (step 4) are finite.

3. Compute the *clever covariate* $H_n(A, L)$, at the observed A and at $A = 1$ and $A = 0$. *By hand.*
4. *Fluctuate*: fit the logistic regression $\text{logit } \bar{Q}_{n,b}(\epsilon)(A, L) = \text{logit } \bar{Q}_{n,b}(A, L) + \epsilon H_n(A, L)$, with the initial fit as an offset, giving ϵ_n . *By hand*, with `glm()`.
5. Update, $\bar{Q}_{n,b}^* = \bar{Q}_{n,b}(\epsilon_n)$, and plug in. Fitting ϵ by maximum likelihood makes the average of the EIF zero: TMLE *solves the EIF estimating equation*. *By hand.*
6. Compute the EIF at the targeted fit for the standard error, and scale back to the original units (multiplying by the range of Y_b , as `scale_from_unit()` does). *By hand.*

`tmle_ate()` wraps all six steps; we use it once we have done them by hand.

1.3.1 A synthetic CpG with a known effect

Before turning to the real data, we try TMLE on a **synthetic CpG** for which we know the true causal effect. In `sim_cpg()`, $L = (\text{bmi}_z, \text{sex})$, where bmi_z is a standardized body-mass index (a z -score), and

$$g_0(L) = \text{expit}(-1 + 0.7 \text{bmi}_z^2 + 0.4 \text{sex}) ,$$

$$\bar{Q}_0(A, L) = 0.5 - A + 0.3 \text{bmi}_z + 0.8 \text{bmi}_z^2 + 0.2 \text{sex} ,$$

so the true ATE is $\psi_0 = -1$. A main-terms GLM in $(A, \text{bmi}_z, \text{sex})$ omits the confounding through bmi_z^2 and is misspecified; the learner `SL.glm_correct` (in `practical/R/learners.R`) adds bmi_z^2 and is correctly specified.

```

cpg_sim <- sim_cpg()
a_sim <- cpg_sim$A
l_sim <- cpg_sim[, c("bmi_z", "sex")]
head(cpg_sim, 3)

```

	bmi_z	sex	A	Y
1	-0.6264538	0	1	-0.94129573
2	0.1836433	1	1	0.16435144
3	-0.8356286	0	1	0.09328663

1.3.2 Exercise 1.2

TMLE by hand

Complete the steps below to compute the TMLE of the ATE of the synthetic CpG, with correctly specified GLMs for both g_0 and $\bar{Q}_0$. Check that the average of the estimated EIF at the targeted fit is (numerically) zero.

```
# 1. scale the outcome to [0, 1]
y_range <- range(cpg_sim$Y)
y_s <- scale_to_unit(cpg_sim$Y, y_range)

# 2. initial estimates of g_0 and of Q_0 (on the unit scale)
g_n <- estimate_g(a_sim, l_sim, g_lib = "SL.glm.correct")
Q_n <- estimate_Q(y_s, a_sim, l_sim, q_lib = ___)
QA <- bound_precision(Q_n$QA)
Q1 <- bound_precision(Q_n$Q1)
Q0 <- bound_precision(Q_n$Q0)

# 3. the clever covariate, at the observed A, at A = 1, and at A = 0
H_A <- ___
H_1 <- 1 / g_n
H_0 <- -1 / (1 - g_n)

# 4. fluctuate: logistic regression on H_A, with the initial fit as offset
fluc <- glm(y_s ~ -1 + H_A + offset(qlogis(QA)), family = quasibinomial())
eps <- coef(fluc)

# 5. update the initial fit, and plug in
Q1_star <- plogis(qlogis(Q1) + eps * ___)
Q0_star <- plogis(qlogis(Q0) + eps * ___)
QA_star <- plogis(qlogis(QA) + eps * H_A)
psi_s <- mean(___)
```

```
# 6. the EIF at the targeted fit, and the estimate on the original scale
eif <- diff(y_range) * (H_A * (y_s - QA_star) + Q1_star - Q0_star - psi_s)
c(
  psi = diff(y_range) * psi_s,
  se = sd(eif) / sqrt(nrow(cpg_sim)),
  mean_eif = mean(eif)
)
```

Solution

```
# 1. scale the outcome to [0, 1]
y_range <- range(cpg_sim$Y)
y_s <- scale_to_unit(cpg_sim$Y, y_range)

# 2. initial estimates of g_0 and of Q_0 (on the unit scale)
g_n <- estimate_g(a_sim, l_sim, g_lib = "SL.glm_correct")
Q_n <- estimate_Q(y_s, a_sim, l_sim, q_lib = "SL.glm_correct")
QA <- bound_precision(Q_n$QA)
Q1 <- bound_precision(Q_n$Q1)
Q0 <- bound_precision(Q_n$Q0)

# 3. the clever covariate, at the observed A, at A = 1, and at A = 0
H_A <- a_sim / g_n - (1 - a_sim) / (1 - g_n)
H_1 <- 1 / g_n
H_0 <- -1 / (1 - g_n)

# 4. fluctuate: logistic regression on H_A, with the initial fit as offset
fluc <- glm(y_s ~ -1 + H_A + offset(qlogis(QA)), family = quasibinomial())
eps <- coef(fluc)

# 5. update the initial fit, and plug in
Q1_star <- plogis(qlogis(Q1) + eps * H_1)
Q0_star <- plogis(qlogis(Q0) + eps * H_0)
QA_star <- plogis(qlogis(QA) + eps * H_A)
psi_s <- mean(Q1_star - Q0_star)

# 6. the EIF at the targeted fit, and the estimate on the original scale
eif <- diff(y_range) * (H_A * (y_s - QA_star) + Q1_star - Q0_star - psi_s)
c(
  psi = diff(y_range) * psi_s,
  se = sd(eif) / sqrt(nrow(cpg_sim)),
  mean_eif = mean(eif)
)
```

psi	se	mean_eif
-1.003542e+00	3.256159e-02	4.871104e-17

```
# the initial (untargeted) plug-in estimate, for comparison
diff(y_range) * mean(Q1 - Q0)
```

```
[1] -1.002366
```

With correctly specified GLMs, the initial plug-in estimate is already close to -1 , and the targeting step barely moves it (ϵ_n is small). The mean of the EIF at the targeted fit is zero up to numerical error: that is what the fluctuation step is for.

As a check, `tmle_ate()`, given the propensity score estimates and a library for $\bar{Q}_0$, reproduces the computation above:

```
set.seed(11249)
g_n <- estimate_g(a_sim, l_sim, "SL.glm_correct")
tmle_ate(cpg_sim$Y, a_sim, l_sim, g_n, "SL.glm_correct")
```

	psi	se	mean_eif
	-1.003542e+00	3.256159e-02	4.871104e-17

1.3.3 Exercise 1.3

Misspecified GLMs, correct GLMs, and Super Learner

- Using `estimate_g()` and `tmle_ate()`, estimate the ATE of the synthetic CpG with all four combinations of misspecified ("SL.glm") and correct ("SL.glm_correct") GLMs for g_0 and $\bar{Q}_0$. Before running the code: which combinations do you expect to give a (nearly) unbiased estimate? How does the main-terms regression coefficient compare?
- In practice, we don't know which GLM is correct. Re-estimate the ATE with Super Learner for both nuisance parameters, using the library below, which includes the correct GLM *among* other candidates: the main-terms GLM, lasso and ridge regression (`glmnet`), MARS (`earth`), and a generalized additive model (`gam`). Which learners does the Super Learner weight most heavily?

```
# (a) GLMs
g_wrong <- estimate_g(a_sim, l_sim, "SL.glm")
g_right <- estimate_g(a_sim, l_sim, ___)
rbind(
  both_wrong = tmle_ate(cpg_sim$Y, a_sim, l_sim, g_wrong, "SL.glm"),
  both_right = tmle_ate(cpg_sim$Y, a_sim, l_sim, g_right, "SL.glm_correct"),
  q_wrong_g_right = tmle_ate(___),
  q_right_g_wrong = tmle_ate(___)
)
```

```

coef(lm(Y ~ A + bmi_z + sex, data = cpg_sim))["A"]

# (b) Super Learner
sl_lib <- c(
  "SL.mean",
  "SL.glm",
  "SL.glm_correct",
  "SL.glmnet",
  "SL.ridge",
  "SL.earth",
  "SL.gam"
)
g_sl <- estimate_g(____)
tmle_ate(cpg_sim$Y, a_sim, l_sim, g_sl, ____ )
round(estimate_Q(cpg_sim$Y, a_sim, l_sim, sl_lib)$coef, 2)

```

Solution

```

# (a) GLMs
g_wrong <- estimate_g(a_sim, l_sim, "SL.glm")
g_right <- estimate_g(a_sim, l_sim, "SL.glm_correct")
rbind(
  both_wrong = tmle_ate(cpg_sim$Y, a_sim, l_sim, g_wrong, "SL.glm"),
  both_right = tmle_ate(cpg_sim$Y, a_sim, l_sim, g_right, "SL.glm_correct"),
  q_wrong_g_right = tmle_ate(cpg_sim$Y, a_sim, l_sim, g_right, "SL.glm"),
  q_right_g_wrong = tmle_ate(cpg_sim$Y, a_sim, l_sim, g_wrong, "SL.glm_correct")
)

```

		psi	se	mean_eif
both_wrong	-0.1706585	0.05822168	-2.138787e-13	
both_right	-1.0035424	0.03256159	4.871104e-17	
q_wrong_g_right	-1.0428640	0.05916187	4.183320e-16	
q_right_g_wrong	-1.0023752	0.02211830	-1.510482e-12	

```
coef(lm(Y ~ A + bmi_z + sex, data = cpg_sim))["A"]
```

```
[1] -0.1762296
```

```
# (b) Super Learner
sl_lib <- c(
  "SL.mean",
  "SL.glm",
  "SL.glm_correct",
  "SL.glmnet",
  "SL.ridge",
  "SL.earth",
  "SL.gam"
)
g_sl <- estimate_g(a_sim, l_sim, sl_lib)
tmle_ate(cpg_sim$Y, a_sim, l_sim, g_sl, sl_lib)
```

```
      psi      se    mean_eif
-1.004047e+00  2.831007e-02 -1.483626e-11
```

```
round(estimate_Q(cpg_sim$Y, a_sim, l_sim, sl_lib)$coef, 2)
```

SL.mean_All	SL.glm_All	SL.glm_correct_All	SL.glmnet_All
0.00	0.00	0.99	0.00
SL.ridge_All	SL.earth_All	SL.gam_All	
0.00	0.00	0.01	

- By double robustness, every combination except “both wrong” is consistent. With both GLMs misspecified, the estimate is far from -1 : the omitted bmi_z^2 confounds the comparison, just as it does for the main-terms regression coefficient. With only g_n correct, the estimate is nearly unbiased but noisier: the targeting step then does all the work, through the weights H_n , which are large for some people.
- The Super Learner puts most of its weight on `SL.glm_correct`: by cross-validation, it finds the correctly specified GLM among the candidates and recovers the truth. Had no candidate been correct, it would still pick the best-predicting combination, and double robustness only needs one of g_n and $\bar{Q}_n$ to be (close to) right.

1.4 Back to the real data

For the methylation data, we do not know the correct model, so we rely on the Super Learner. We drop `SL.glm_correct`, which is specific to the synthetic CpG, and `SL.gam`, whose backfitting fails on the nearly collinear cell-type proportions (which sum to about one).

1.4.1 Exercise 1.4

i The linear model, and *AHRR*

- Fit the working linear model of Exercise 1.1(a) to each CpG, collecting $\beta_{n,1}$, its standard error, and its t -statistic.
- Notice that $g_0(L)$ does not depend on b : estimate it **once**, with the Super Learner.
- Estimate the ATE of smoking on cg05575921 (*AHRR*) by TMLE, with the Super Learner and with `q_lib = "SL.glm"`. Compare both to the linear model's $\beta_{n,1}$. Why is the "SL.glm" version so close to it?

```
# (a) the linear-model baseline
lm_one <- function(cpg) {
  fit <- lm(reformulate(c("smoker", covars), response = cpg), data = methy)
  coef(summary(fit))[___, c("Estimate", "Std. Error", "t value")]
}
lm_res <- t(sapply(cpgs, lm_one))
head(lm_res[order(-abs(lm_res[, "t value"]))], ]))

# (b) the propensity score, once
real_lib <- setdiff(sl_lib, c("SL.glm_correct", "SL.gam"))
g_n <- estimate_g(____, ____, real_lib)

# (c) one CpG
rbind(sl = tmle_ate(methy$cg05575921, A, L, g_n, real_lib), glm = tmle_ate(____))
lm_res["cg05575921", ]
```

 Solution

```
# (a) the linear-model baseline
lm_one <- function(cpg) {
  fit <- lm(reformulate(c("smoker", covars), response = cpg), data = methy)
  coef(summary(fit))["smoker", c("Estimate", "Std. Error", "t value")]
}
lm_res <- t(sapply(cpgs, lm_one))
head(lm_res[order(-abs(lm_res[, "t value"]))], ])
```

	Estimate	Std. Error	t value
cg05575921	-1.7162551	0.05852813	-29.323592
cg21566642	-0.7905914	0.04415236	-17.905982

```
cg03636183 -0.6788992 0.05405966 -12.558333
cg19859270 -0.6294083 0.06298618 -9.992800
cg08709672 -0.2169516 0.02496367 -8.690693
cg19614919 -0.3696805 0.04559097 -8.108634
```

```
# (b) the propensity score, once
real_lib <- setdiff(sl_lib, c("SL.glm_correct", "SL.gam"))
g_n <- estimate_g(A, L, real_lib)
summary(g_n)
```

```
Min. 1st Qu. Median Mean 3rd Qu. Max.
0.4426 0.6065 0.6850 0.6798 0.7586 0.8935
```

```
# (c) one CpG
rbind(
  sl = tmle_ate(methy$cg05575921, A, L, g_n, real_lib),
  glm = tmle_ate(methy$cg05575921, A, L, g_n, "SL.glm")
)
```

```
      psi      se    mean_eif
sl -1.749647 0.04974591 -5.693329e-13
glm -1.739676 0.04975121 -8.631651e-13
```

```
lm_res["cg05575921", ]
```

```
Estimate Std. Error t value
-1.71625514 0.05852813 -29.32359214
```

The *AHRR* CpG is at the top of the linear-model ranking, as in large epigenome-wide studies of smoking (Joehanes et al. 2016). With $\bar{Q}_{n,b}$ from the main-terms GLM, the initial plug-in estimate $\frac{1}{n} \sum_i \{\bar{Q}_{n,b}(1, L_i) - \bar{Q}_{n,b}(0, L_i)\}$ is exactly $\beta_{n,1}$ (Exercise 1.1a); the targeting step then moves it only as much as the data, through g_n , indicate that the linear model is misspecified. When the linear model fits well, that step is small.

1.4.2 Exercise 1.5

All CpGs

Apply `tmle_ate()`, with the Super Learner, to every CpG. (This takes a minute or two.) Since we will need the variance of the EIF, $\sigma_{n,b}^2$, recover it from the standard error.

```
tmle_res <- t(sapply(cpgs, function(cpg) tmle_ate(__)))
tmle_res <- cbind(tmle_res, s2 = n * tmle_res[, "se"]^2)
head(tmle_res)
```

Solution

```
tmle_res <- t(sapply(cpgs, function(cpg) {
  tmle_ate(methy[[cpg]], A, L, g_n, real_lib)
}))
tmle_res <- cbind(tmle_res, s2 = n * tmle_res[, "se"]^2)
head(tmle_res)
```

	psi	se	mean_eif	s2
cg05575921	-1.7434290	0.04970326	-7.168693e-13	0.6250148
cg21566642	-0.8296899	0.03834594	-4.300469e-17	0.3720139
cg03636183	-0.7174997	0.04960191	-7.159714e-10	0.6224684
cg19859270	-0.6361967	0.05935419	-5.446029e-11	0.8912987
cg13591352	0.2373700	0.03620747	2.863168e-13	0.3316781
cg04255770	-0.3362369	0.05061981	-3.911896e-12	0.6482784

```
cor(tmle_res[, "psi"], lm_res[, "Estimate"])
```

```
[1] 0.9980098
```

1.5 Variance moderation

With many biomarkers and few samples, some $\sigma_{n,b}^2$ will be tiny by chance, producing large test statistics for small effects. The moderated t -statistic of Smyth (2004) borrows strength across

biomarkers by shrinking each variance towards a common value,

$$\tilde{\sigma}_{n,b}^2 = \frac{d_b \sigma_{n,b}^2 + d_0 s_0^2}{d_b + d_0}, \quad \tilde{t}_{n,b} = \frac{\psi_{n,b}^*}{\sqrt{\tilde{\sigma}_{n,b}^2/n}},$$

where d_b are the degrees of freedom of $\sigma_{n,b}^2$, and the prior degrees of freedom d_0 and prior variance s_0^2 are estimated from all B variances by empirical Bayes. $\tilde{t}_{n,b}$ is compared to a t -distribution with $d_b + d_0$ degrees of freedom. `biotmle`'s key observation (Hejazi, Boileau, et al. 2023) is that nothing in this recipe requires $\sigma_{n,b}^2$ to come from a linear model: it can be the variance of the EIF.

The function `moderate(s2, df)` (in `practical/R/moderate.R`) implements this shrinkage, reproducing `limma::squeezeVar()`; it returns `s2_tilde`, `d0`, and `s0_2`.

1.5.1 Exercise 1.6

 How much shrinkage?

- What does $\tilde{\sigma}_{n,b}^2$ become as $d_0 \rightarrow 0$? As $d_0 \rightarrow \infty$?
- Here, $d_b = n - 1 = 252$. Suppose empirical Bayes estimates $d_0 \approx 5$. What fraction of $\tilde{\sigma}_{n,b}^2$ comes from the prior? When would moderation matter more?
- Why might the EIF-based variance $\sigma_{n,b}^2$ be *more* erratic than the linear model's residual variance, making moderation especially valuable?

 Solution

- As $d_0 \rightarrow 0$, $\tilde{\sigma}_{n,b}^2 \rightarrow \sigma_{n,b}^2$: no moderation, an ordinary t -statistic. As $d_0 \rightarrow \infty$, $\tilde{\sigma}_{n,b}^2 \rightarrow s_0^2$: every biomarker shares one variance, and tests rank biomarkers by effect size alone.
- The weight on the prior is $d_0/(d_b + d_0) = 5/257 \approx 2\%$, so moderation barely changes anything at $n = 253$. It matters when n is small (say, $n = 10$ – 30 , as in many expression studies), where d_b is comparable to d_0 .
- The EIF involves H_n , i.e., $1/g_n(L)$, which can be large for a few people (as in Exercise 1.3a). It also involves flexible $\bar{Q}_{n,b}$ fits, whose residuals vary across CpGs. The EIF variance estimator is only *asymptotically* valid: in small samples, it can be too small for some CpGs (false positives) and too large for others. Shrinkage stabilizes it without giving up model-agnostic inference.

1.5.2 Exercise 1.7

Moderated tests and multiple testing

- Moderate the EIF variances, compute moderated t -statistics and p -values, and adjust for multiple testing with the Benjamini–Hochberg procedure (Benjamini and Hochberg 1995).
- (Optional.) Applying the *same* function to the linear models' residual variances gives `limma`'s moderated t -statistic. Do so, and compare the two lists of discoveries.

```
# (a) biotmle: moderate the EIF variances
mod_eif <- moderate(tmle_res[, "s2"], df = __)
t_eif <- tmle_res[, "psi"] / sqrt(__ / n)
p_eif <- 2 * pt(-abs(t_eif), df = n - 1 + mod_eif$d0)
q_eif <- p.adjust(p_eif, method = "BH")

# (b) limma: moderate the residual variances of the linear models
df_lm <- n - length(covars) - 2
...
```

Solution

```
# (a) biotmle: moderate the EIF variances
mod_eif <- moderate(tmle_res[, "s2"], df = n - 1)
t_eif <- tmle_res[, "psi"] / sqrt(mod_eif$s2_tilde / n)
p_eif <- 2 * pt(-abs(t_eif), df = n - 1 + mod_eif$d0)
q_eif <- p.adjust(p_eif, method = "BH")
c(d0 = mod_eif$d0, s0_2 = mod_eif$s0_2)
```

```
      d0      s0_2
4.3651262 0.3660632
```

```

# (b) limma: moderate the residual variances of the linear models
df_lm <- n - length(covars) - 2
sigma_lm <- sapply(cpgs, function(cpg) {
  sigma(lm(reformulate(c("smoker", covars), response = cpg), data = methy))
})
unscaled_se <- lm_res[, "Std. Error"] / sigma_lm
mod_lm <- moderate(sigma_lm^2, df = df_lm)
t_lm <- lm_res[, "Estimate"] / (unscaled_se * sqrt(mod_lm$s2_tilde))
p_lm <- 2 * pt(-abs(t_lm), df = df_lm + mod_lm$d0)
q_lm <- p.adjust(p_lm, method = "BH")

table(biotmle = q_eif < 0.05, limma = q_lm < 0.05)

```

```

      limma
biotmle FALSE TRUE
FALSE   127    0
TRUE     8  165

```

```

res <- data.frame(cpg = cpgs, psi = tmle_res[, "psi"], q = q_eif)
ggplot(res, aes(psi, -log10(q), colour = q < 0.05)) +
  geom_point(alpha = 0.7) +
  geom_hline(yintercept = -log10(0.05), linetype = "dashed") +
  geom_text(
    data = subset(res, cpg == "cg05575921"),
    aes(label = "AHRR"),
    nudge_x = 0.15,
    colour = "black"
  ) +
  labs(
    x = "Estimated ATE of smoking (M-value scale)",
    y = expression(-log[10](q)),
    colour = "FDR < 5%"
  ) +
  theme_bw()

```

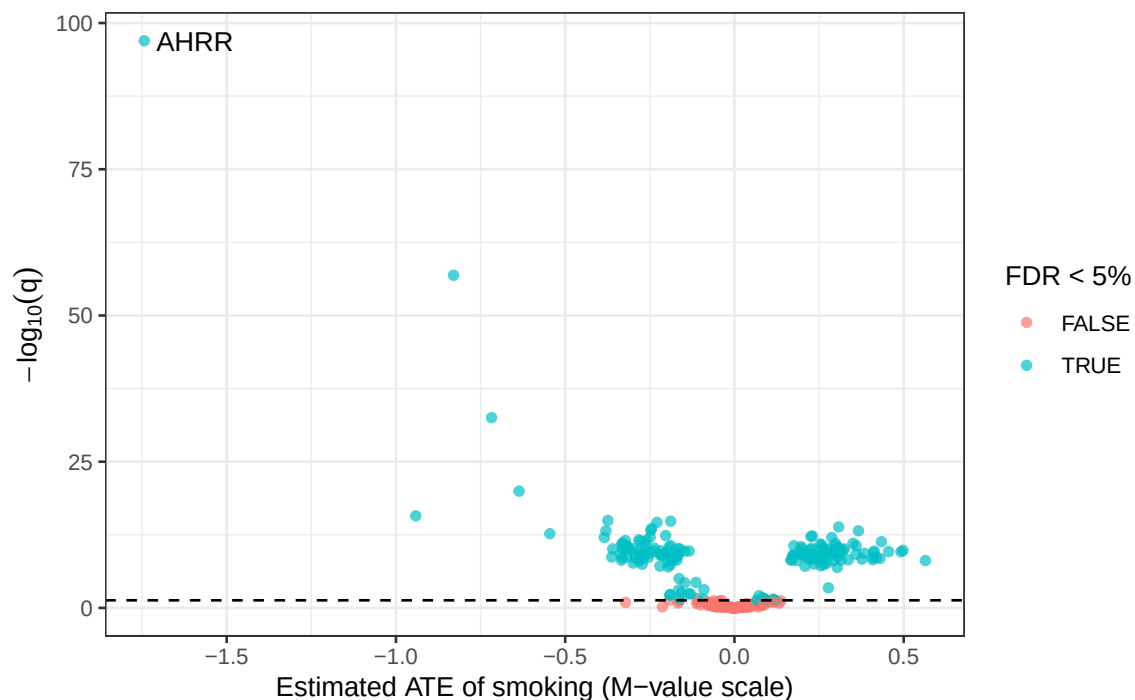


Figure 1.2: Volcano plot of the variance-moderated TMLE estimates.

The two procedures largely agree here: the linear model seems a reasonable working model for most CpGs, and n is large enough that moderation is mild. Where they disagree is precisely where the linear model's assumptions are doing work that the data cannot verify.

1.5.3 Exercise 1.8 (stretch)

Two-stage analyses

These 300 CpGs were themselves selected: half passed a screen applied to all 472,678 CpGs measured. The published analysis (Hejazi, Boileau, et al. 2023) screens with `limma`, runs `biotmle` on the 2,537 CpGs that pass, and adjusts p -values as if all 472,678 had been tested (setting $p = 1$ for those screened out) (Tuglus and van der Laan 2009). Why is adjusting only over the CpGs that pass the screen not enough?

Solution

The screen used the same data, and kept CpGs *because* their association with smoking looked strong. Testing only the survivors, and correcting only for their number, ignores the selection: the error rate would be computed over a set of hypotheses chosen to look significant. Counting every CpG measured in the multiplicity correction accounts for the first stage.

2 Stochastic-interventional vaccine efficacy

i In a nutshell

We estimate how vaccine efficacy would change had the vaccine shifted everyone's antibody titers. In our simulated trial, titers were measured only in a case-cohort sample, so we use `lmt` with weights for the sampling design.

Roadmap: (1) the population mean under a modified treatment policy (MTP), the SVE, and its linear-model analogue $\rightarrow$ (2) one MTP, with misspecified and correct outcome GLMs and Super Learner $\rightarrow$ (3) the SVE curve $\rightarrow$ (4) plug-in g-computation, and why it is more fragile $\rightarrow$ (5) what happens if the sampling design is ignored $\rightarrow$ (6) positivity.

i Notation for this part

Symbol	Meaning
L	age, high-risk behaviour, sex
A	vaccine (1) or placebo (0), randomized
Y	infection by the end of follow-up
S	immune marker ($\log_{10}$ antibody titer), a mediator of $A \rightarrow Y$
$R, g_{0,R}(Y, L)$	phase-two sampling indicator, and its known probability $P_0(R = 1 \mid Y, L)$
$\bar{Q}_{0,Y}(s, l)$	risk among vaccinees, $P_0(Y = 1 \mid S = s, A = 1, L = l)$
$g_{0,S}(s \mid l)$	density of S given $A = 1$ and $L = l$
$d(S, L) = S + \delta$	(generalized propensity score) the modified treatment policy (MTP): an additive shift of S
$\psi_{0,\delta}$	population mean under the MTP, $\mathbb{E}_0[Y(d(S, L)) \mid A = 1]$
SVE_δ	$1 - \psi_{0,\delta}/P_0(Y = 1 \mid A = 0)$

2.1 The question

In vaccine trials, an *immune correlate of protection* is a marker of the vaccine-induced immune response S (e.g., neutralizing antibody titer) that tracks protection against the outcome Y (e.g., infection) (Plotkin and Gilbert 2012). Correlates help us understand *how* a vaccine protects and can serve as surrogate endpoints for future trials (Prentice 1989; Gilbert et al. 2024). The question from the keynote:

How would the risk of infection have changed had the vaccine induced higher, or lower, antibody titers?

The vaccine A is randomized, but S is not: it is a *post-treatment* variable on the pathway $A \rightarrow S \rightarrow Y$, a mediator. Questions about intervening on S are questions about mediation.

2.2 The data: a simulated trial with a case-cohort design

Data from the real trials are not public, so we simulate a trial of 10,000 people. The *complete* data on one person would be $X = (L, A, S, Y)$:

- L : age (in years), a `high_risk` behaviour indicator, `sex`;
- A : vaccine (1) or placebo (0), randomized 1:1;
- S : $\log_{10}$ neutralizing antibody titer ($S = 0$, the assay floor, in the placebo arm);
- Y : infection by the end of follow-up.

The data are generated (in `practical/R/sim_vaccine_trial.R`) by the structural equations

$$\begin{aligned} \text{age} &\sim \text{Uniform}\{18, \dots, 75\}, \quad \text{high_risk} \sim \text{Bernoulli}(0.35), \quad \text{sex} \sim \text{Bernoulli}(0.5), \\ A &\sim \text{Bernoulli}(0.5), \\ S &= A \cdot \{1.4 - 0.02(\text{age} - 45) - 0.2\text{high_risk} + U_S\}, \quad U_S \sim \mathcal{N}(0, 0.5^2), \\ Y &\sim \text{Bernoulli}(\text{expit}\{-2.3 + 0.025(\text{age} - 45) + 0.9\text{high_risk} - 2.2\min(S, 1.2)\}), \\ R &\sim \text{Bernoulli}(g_{0,R}(Y, L)), \end{aligned}$$

with $g_{0,R}$ given below. The vaccine acts on Y only through S ; the risk of infection falls with titer up to a $\log_{10}$ titer of 1.2 and is flat above it; `sex` affects nothing.

Assaying blood samples is expensive, so S is measured only in a *phase-two* sample, indicated by $R \in \{0, 1\}$: a case-cohort design (Prentice 1986), a special case of two-phase sampling (White 1982; Breslow et al. 2003). The investigators choose who enters phase two, and they choose using only Y and L : every infected person (case) is sampled, and non-cases are sampled

at random within strata of `high_risk`. The sampling probabilities are therefore known by design,

$$g_{0,R}(Y, L) = P_0(R = 1 | Y, L) = \begin{cases} 1 & \text{if } Y = 1, \\ 0.25 & \text{if } Y = 0 \text{ and } \text{high_risk} = 1, \\ 0.15 & \text{if } Y = 0 \text{ and } \text{high_risk} = 0, \end{cases}$$

and recorded as `g_R`. So the *observed* data unit is

$$O = (L, A, RS, Y, R) \sim P_0 \in \mathcal{P},$$

where `RS` is `S` when `R` = 1 and missing otherwise.

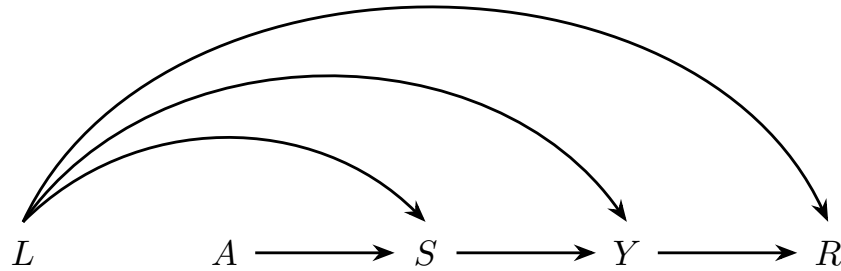


Figure 2.1: DAG for the vaccine trial, in time order. The vaccine `A` is randomized; covariates `L` confound the effect of the immune marker `S` on infection `Y`; inclusion `R` in the phase-two sample depends, by design, only on `Y` and `L`. (In our simulation, the vaccine acts on `Y` only through `S`.)

```
trial <- sim_vaccine_trial()
head(trial)
```

	age	high_risk	sex	A	S	Y	R	g_R
1	56		1	1	1.635746	0	1	0.25
2	60	0	0	0	NA	0	0	0.15
3	51	0	1	1	NA	0	0	0.15
4	44	0	0	0	NA	0	0	0.15
5	66	0	0	0	NA	0	0	0.15
6	53	1	0	0	0.000000	0	1	0.25

```
with(trial, table(vaccine = A, infected = Y))
```

	infected	
vaccine	0	1
0	4247	672
1	4921	160

```
with(trial, table(vaccine = A, phase_two = R))
```

```
      phase_two
vaccine    0    1
  0 3463 1456
  1 4069 1012
```

```
with(trial, table(phase_two = R, infected = Y)) # every case is sampled
```

```
      infected
phase_two    0    1
  0  7532    0
  1 1636   832
```

We focus on vaccine recipients in the phase-two sample:

```
vax <- subset(trial, A == 1 & R == 1)
nrow(vax)
```

```
[1] 1012
```

```
mean(vax$Y) # cases are heavily over-represented!
```

```
[1] 0.1581028
```

2.3 The target parameter

Let $\bar{Q}_{0,Y}(s, l) = P_0(Y = 1 \mid S = s, A = 1, L = l)$ be the risk of infection among vaccinees, and $g_{0,S}(s \mid l)$ the conditional density of S given $A = 1$ and $L = l$ (the *generalized propensity score*). A *modified treatment policy* (MTP) replaces each vaccinee's natural titer S with a new value $d(S, L)$ that may depend on it (Haneuse and Rotnitzky 2013; Díaz et al. 2023). Here, the MTP is the additive shift $d(S, L) = S + \delta$ (Díaz and van der Laan 2012). Our target parameter is the **stochastic-interventional vaccine efficacy** (Hejazi et al. 2021; Gilbert, Fong, et al. 2021),

$$\text{SVE}_\delta = 1 - \frac{\psi_{0,\delta}}{P_0(Y = 1 \mid A = 0)}.$$

- The **numerator** $\psi_{0,\delta} = \mathbb{E}_0[Y(d(S, L)) \mid A = 1]$ is a *population mean under the MTP*: the risk of infection among vaccinees had each vaccinee's titer been shifted by δ . Under no unmeasured confounding of S and Y given L , and positivity, it is identified by

$$\psi_{0,\delta} = \mathbb{E}_0 \left[\int \bar{Q}_{0,Y}(d(s, L), L) g_{0,S}(s \mid L) ds \mid A = 1 \right] :$$

vaccinees with covariates L , shift a given unit's observed titer value by δ units, and average the vaccinee risk at the shifted titer $s + \delta$ over L .

- The **denominator** is the risk of infection had everyone received placebo, $P_0(Y(0) = 1)$. Since A is randomized, it is identified most simply by the placebo-arm risk, $P_0(Y = 1 \mid A = 0)$.¹

At $\delta = 0$, this is the usual vaccine efficacy; as δ varies, it traces out how efficacy would change with the vaccine's immunogenicity. It was one of the immune correlates analyses of the COVE trial of the Moderna mRNA-1273 COVID-19 vaccine (Gilbert, Montefiori, et al. 2021; Hejazi, Shen, et al. 2023; Huang et al. 2023). Unlike the controlled vaccine efficacy from the keynote (Gilbert et al. 2023), which sets every vaccinee's titer to the *same* value s , an MTP moves each person's titer from where it naturally fell.

2.3.1 Exercise 2.1

MTPs, linear models, and sampling

- Suppose the outcome model is linear: $\bar{Q}_{0,Y}(s, l) = \beta_0 + \beta_1 s + \gamma^\top l$. Show that $\psi_{0,\delta} - \psi_{0,0} = \beta_1 \delta$. What does SVE_δ look like as a function of δ ?
- What goes wrong with the linear model as δ grows? (Hint: $\psi_{0,\delta}$ is a probability.)
- Among the phase-two vaccinees, what is the share of cases, compared to the full vaccine arm? What would happen if we estimated $\psi_{0,\delta}$ from **vax** as if it were a random sample of vaccinees? How do the weights $1/g_{0,R}(Y, L)$ fix this?
- Positivity for an MTP requires that, for each l , the shifted value $s + \delta$ is a value of S that could occur among people with $L = l$. Why is this harder to satisfy for large $|\delta|$?

Solution

- $\psi_{0,\delta} = \mathbb{E}_0[\beta_0 + \beta_1(S + \delta) + \gamma^\top L \mid A = 1] = \psi_{0,0} + \beta_1 \delta$. So $\text{SVE}_\delta = 1 - (\psi_{0,0} + \beta_1 \delta)/P_0(Y = 1 \mid A = 0)$ is a straight line in δ with slope $-\beta_1/P_0(Y = 1 \mid A = 0)$: the linear model's β_1 is the effect of the MTP with $\delta = 1$, just as β_1 was the ATE

¹Randomization also gives the covariate-adjusted identification $\mathbb{E}_0[P_0(Y = 1 \mid A = 0, L)]$. Estimating the denominator this way uses the prognostic information in L and can improve efficiency; we use the simpler unadjusted estimate.

in Part 1.

- b. A straight line eventually leaves $[0, 1]$: for large positive δ (with $\beta_1 < 0$), the predicted risk becomes negative and the SVE exceeds 100%; the model also forces each additional unit of titer to buy the same reduction in risk, whatever the starting titer.
- c. All cases are sampled but only 15–25% of non-cases, so cases are hugely over-represented (compare `mean(vax$Y)` with the infection rate in the vaccine arm). Ignoring this would badly overestimate risk. Weighting each phase-two person by $1/g_{0,R}(Y, L)$ makes them stand in for the $1/g_{0,R}(Y, L)$ people like them in the full trial (inverse probability weighting), recovering the full-cohort target.
- d. Shifting by δ asks about titers $s + \delta$; for large $|\delta|$, many shifted titers fall outside the range observed among similar people (e.g., below the assay floor, or far above any observed response), so the data contain no information about risk there and any estimate is an extrapolation.

2.4 Estimation with `lmt`

The `lmt` package (Diaz et al. 2023) implements estimators of population means under MTPs, including our additive shift. `lmt_tmle()` estimates the numerator $\psi_{0,\delta}$ by targeted maximum likelihood. It needs two nuisance estimates: $\bar{Q}_{n,Y}$, and an estimate of the density ratio $g_{0,S}(s - \delta | l)/g_{0,S}(s | l)$, which `lmt` obtains by classifying observed versus shifted titers. Both are fit with `SuperLearner` libraries (`learners_outcome`, `learners_trt`). `lmt_tmle()` accepts sampling weights through `weights`, which gives the inverse-probability-weighted version of the estimator for two-phase designs (Rose and van der Laan 2011).²

In our simulation, the risk of infection falls with titer only up to a threshold ($\log_{10}$ titer of 1.2) and is flat above it. A main-terms logistic regression ("`SL.glm`") for $\bar{Q}_{0,Y}$ is therefore misspecified; "`SL.glm_threshold`" (in `practical/R/learners.R`), which uses $\min(S, 1.2)$ in place of S , is correctly specified. For the density ratio, we use a main-terms logistic regression throughout. It is correctly specified here: S given $A = 1$ and L is normal, with a mean linear in L and constant variance, so the log density ratio is linear in (s, l) .

²The estimator in the `txshift` package (Hejazi and Benkeser 2020) also *augments* the weighted estimator with a model for the sampling mechanism, using information on the whole cohort to gain efficiency (Breslow et al. 2009; Hejazi et al. 2021). The reweighted version is valid here because $g_{0,R}$ is known by design.

2.4.1 Exercise 2.2

i One MTP, three outcome models

Estimate $\psi_{0,0.5}$, the risk among vaccinees had every vaccinee's titer been $0.5 \log_{10}$ units (about 3-fold) higher, using for $\overline{Q}_n$ (i) the misspecified GLM, (ii) the correctly specified GLM, and (iii) a Super Learner whose library includes the correct GLM. Compare to the truth, `true_sve(0.5)`.

```
shift_by <- function(delta) function(data, trt) ___
fit_shift <- function(delta, q_lib, weights = 1 / vax$g_R) {
  lmtp_tmle(
    data = vax,
    trt = "S",
    outcome = "Y",
    baseline = c("age", "high_risk", "sex"),
    shift = shift_by(delta),
    outcome_type = "binomial",
    learners_outcome = q_lib,
    learners_trt = "SL.glm",
    folds = 5,
    weights = weights
  )
}
sl_lib <- c(
  "SL.mean",
  "SL.glm",
  "SL.glm_threshold",
  "SL.glmnet",
  "SL.earth",
  "SL.gam"
)
fits_05 <- list(
  glm_wrong = fit_shift(0.5, "SL.glm"),
  glm_right = fit_shift(0.5, ___),
  super_learner = fit_shift(0.5, ___)
)
sapply(fits_05, function(fit) {
  c(psi = fit$estimate@x, se = fit$estimate@std_error)
})
true_sve(0.5)
```

💡 Solution

```
shift_by <- function(delta) function(data, trt) data[[trt]] + delta
fit_shift <- function(delta, q_lib, weights = 1 / vax$g_R) {
  lmtm_tmle(
    data = vax,
    trt = "S",
    outcome = "Y",
    baseline = c("age", "high_risk", "sex"),
    shift = shift_by(delta),
    outcome_type = "binomial",
    learners_outcome = q_lib,
    learners_trt = "SL.glm",
    folds = 5,
    weights = weights
  )
}
sl_lib <- c(
  "SL.mean",
  "SL.glm",
  "SL.glm_threshold",
  "SL.glmnet",
  "SL.earth",
  "SL.gam"
)
fits_05 <- list(
  glm_wrong = fit_shift(0.5, "SL.glm"),
  glm_right = fit_shift(0.5, "SL.glm_threshold"),
  super_learner = fit_shift(0.5, sl_lib)
)
sapply(fits_05, function(fit) {
  c(psi = fit$estimate@x, se = fit$estimate@std_error)
})
```

```
      glm_wrong  glm_right super_learner
psi 0.016524739 0.017072059 0.016768437
se 0.002261512 0.002239203 0.002252903
```

```
true_sve(0.5)
```

```
delta      psi      sve
1 0.5 0.01630634 0.8830436
```

All three are close to the truth, *including* the misspecified GLM. The TML estimator is doubly robust: it is consistent if either $\bar{Q}_{n,Y}$ or the density ratio is consistent, and here the density ratio model is correctly specified. Exercise 2.4 shows what happens without that protection.

2.4.2 Exercise 2.3

The SVE curve

Estimate $\psi_{0,\delta}$ over a grid of values of δ with the Super Learner, compute $\text{SVE}_{n,\delta}$ using the placebo-arm risk (estimable from the full trial, since S is not needed), and compare to the truth, which we know because we simulated the data (`true_sve()`).

```
deltas <- c(-0.5, -0.25, 0, 0.25, 0.5, 0.75)
risk_placebo <- mean(trial$Y[trial$A == ___])
psi_n <- sapply(deltas, function(delta) fit_shift(___)$estimate@x)
sve <- data.frame(
  delta = deltas,
  lmtp = 1 - psi_n / ___,
  truth = true_sve(deltas)$sve
)
sve
```

Solution

```
deltas <- c(-0.5, -0.25, 0, 0.25, 0.5, 0.75)
risk_placebo <- mean(trial$Y[trial$A == 0])
psi_n <- sapply(deltas, function(delta) fit_shift(delta, sl_lib)$estimate@x)
sve <- data.frame(
  delta = deltas,
  lmtp = 1 - psi_n / risk_placebo,
  truth = true_sve(deltas)$sve
)
sve
```

	delta	lmtp	truth
1	-0.50	0.4726627	0.5110695
2	-0.25	0.6384777	0.6742671
3	0.00	0.7464934	0.7813769

```

4  0.25 0.8215102 0.8466500
5  0.50 0.8770759 0.8830436
6  0.75 0.9097931 0.9012873

```

The estimated curve has the shape of the truth but sits a few points below it for most shifts. At $\delta = 0$, the SVE is just the observed vaccine efficacy, which in this sample falls below its true value by chance. For the largest shift, the estimate overtakes the truth: more vaccinees are shifted beyond the titers observed among people like them. This is a first sign of the positivity strain explored in Exercise 2.6a.

2.4.3 Exercise 2.4

The linear-model analogue

Now estimate the same curve by *g-computation*: plug an outcome regression into the identification formula, with no correction. Use parametric models fit by weighted regression to `vax`: (i) the linear probability model of Exercise 2.1(a), (ii) the misspecified main-terms logistic regression, and (iii) the correctly specified threshold logistic regression. Predict each vaccinee's risk at $S + \delta$, average (with the weights), and convert to SVE. Plot all the curves.

```

w <- 1 / vax$g_R
fit_lin <- lm(Y ~ S + age + high_risk + sex, data = vax, weights = w)
fit_logit <- glm(
  Y ~ S + age + high_risk + sex,
  data = vax,
  weights = w,
  family = quasibinomial()
)
fit_thresh <- glm(
  Y ~ pmin(S, 1.2) + age + high_risk + sex,
  data = vax,
  weights = w,
  family = quasibinomial()
)
gcomp_sve <- function(fit, delta) {
  risk <- predict(fit, newdata = transform(vax, S = __), type = "response")
  1 - weighted.mean(risk, w) / risk_placebo
}
sve$linear <- sapply(deltas, gcomp_sve, fit = fit_lin)

```

```
sve$logistic <- sapply(deltas, gcomp_sve, fit = fit_logit)
sve$threshold <- sapply(deltas, gcomp_sve, fit = fit_thresh)
```

💡 Solution

```
w <- 1 / vax$g_R
fit_lin <- lm(Y ~ S + age + high_risk + sex, data = vax, weights = w)
fit_logit <- glm(
  Y ~ S + age + high_risk + sex,
  data = vax,
  weights = w,
  family = quasibinomial()
)
fit_thresh <- glm(
  Y ~ pmin(S, 1.2) + age + high_risk + sex,
  data = vax,
  weights = w,
  family = quasibinomial()
)
gcomp_sve <- function(fit, delta) {
  risk <- predict(
    fit,
    newdata = transform(vax, S = S + delta),
    type = "response"
  )
  1 - weighted.mean(risk, w) / risk_placebo
}
sve$linear <- sapply(deltas, gcomp_sve, fit = fit_lin)
sve$logistic <- sapply(deltas, gcomp_sve, fit = fit_logit)
sve$threshold <- sapply(deltas, gcomp_sve, fit = fit_thresh)
round(sve, 3)
```

	delta	lmt	p	truth	linear	logistic	threshold
1	-0.50	0.473	0.511	0.565	0.491	0.481	
2	-0.25	0.638	0.674	0.658	0.641	0.644	
3	0.00	0.746	0.781	0.751	0.751	0.751	
4	0.25	0.822	0.847	0.844	0.830	0.818	
5	0.50	0.877	0.883	0.937	0.885	0.857	
6	0.75	0.910	0.901	1.030	0.923	0.876	

```
# the linear model's slope:  $SVE(\text{delta}) = SVE(0) - \text{beta}_1 * \text{delta} / P(Y = 1 \mid A = 0)$   
-coef(fit_lin)[["S"]] / risk_placebo
```

```
[1] 0.3722196
```

```
sve_long <- reshape(  
  sve,  
  direction = "long",  
  varying = names(sve)[-1],  
  v.names = "sve",  
  timevar = "method",  
  times = names(sve)[-1],  
  idvar = "delta"  
)  
ggplot(sve_long, aes(delta, sve, colour = method)) +  
  geom_line() +  
  geom_point() +  
  geom_hline(yintercept = 1, linetype = "dotted") +  
  labs(  
    x = expression(paste("Shift ", delta, " in ", log[10], " titer")),  
    y = "SVE",  
    colour = NULL  
  ) +  
  theme_bw()
```

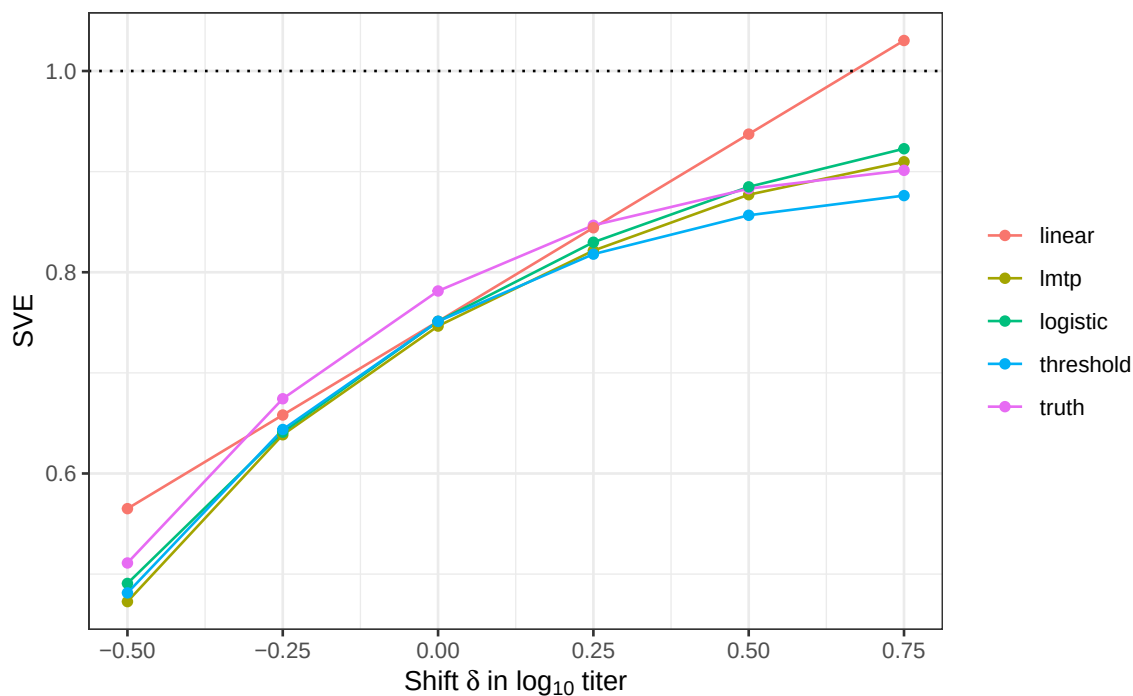


Figure 2.2: SVE as a function of the shift in $\log_{10}$ titer.

The linear model forces a straight line and soon claims an efficacy above 100%: its “risk” is negative. The misspecified logistic regression keeps predicting benefit from higher titers where there is none, overstating the SVE for positive shifts. The threshold model, like the truth, levels off above the threshold. Plug-in g-computation is only as good as its outcome model. In contrast, `lmtpl` was protected by its density ratio model in Exercise 2.2, even with the misspecified GLM.

2.4.4 Exercise 2.5

Forgetting the design

Re-estimate $\text{SVE}_{0.5}$ with `lmtpl_tmle()` and the Super Learner, **without** the sampling weights. What happens, and why?

💡 Solution

```
fit_unwt <- fit_shift(0.5, sl_lib, weights = NULL)
c(
  weighted = 1 - fits_05$super_learner$estimate@x / risk_placebo,
  unweighted = 1 - fit_unwt$estimate@x / risk_placebo,
  truth = true_sve(0.5)$sve
)
```

weighted	unweighted	truth
0.8772560	0.3543930	0.8830436

Without weights, the phase-two sample is treated as representative of all vaccinees, but cases are over-represented about five-fold, so the estimated risk is far too high and the SVE far too low. The estimator must account for the two-phase design.

2.4.5 Exercise 2.6 (stretch)

i Positivity and realistic MTPs

- Estimate SVE_1 , and compare to `true_sve(1)`. Repeat with `"SL.glm_threshold"` as the outcome model, and compare to its g-computation counterpart from Exercise 2.4. What might explain the result? (Hint: what is the range of `vax$S`?)
- An MTP that respects positivity only shifts people whose shifted titer remains plausible (Hejazi et al. 2021): $d(s, l) = s + \delta$ if $s + \delta \leq u$, and $d(s, l) = s$ otherwise. Implement it with u the 90th percentile of S in the phase-two sample, and re-estimate the SVE curve. How would you interpret this effect?

💡 Solution

```
# (a) a large shift
range(vax$S)
```

```
[1] -0.9046149  2.9612913
```

```

c(
  sl = 1 - fit_shift(1, sl_lib)$estimate@x / risk_placebo,
  glm_right = 1 - fit_shift(1, "SL.glm_threshold")$estimate@x / risk_placebo,
  gcomp_right = gcomp_sve(fit_thresh, 1),
  truth = true_sve(1)$sve
)

```

	sl	glm_right	gcomp_right	truth
	0.9347938	0.9270621	0.8844434	0.9093501

```

# (b) a positivity-respecting shift, capped at the 90th percentile
u <- quantile(vax$S, 0.9)
shift_capped <- function(delta) {
  function(data, trt) {
    ifelse(data[[trt]] + delta <= u, data[[trt]] + delta, data[[trt]])
  }
}
psi_capped <- sapply(deltas, function(delta) {
  lmtp_tmle(
    data = vax,
    trt = "S",
    outcome = "Y",
    baseline = c("age", "high_risk", "sex"),
    shift = shift_capped(delta),
    outcome_type = "binomial",
    learners_outcome = sl_lib,
    learners_trt = "SL.glm",
    folds = 5,
    weights = 1 / vax$g_R
  )$estimate@x
})
round(
  data.frame(
    delta = deltas,
    capped = 1 - psi_capped / risk_placebo,
    uncapped = sve$lmtp
  ),
  3
)

```

	delta	capped	uncapped

1	-0.50	0.473	0.473
2	-0.25	0.639	0.638
3	0.00	0.746	0.746
4	0.25	0.820	0.822
5	0.50	0.872	0.877
6	0.75	0.895	0.910

- a. Shifting titers up by a full $\log_{10}$ unit (10-fold) asks about titers well beyond those observed for many vaccinees. The density ratio becomes extreme and the TML estimate less stable, even with a correct outcome model. Plug-in g-computation with the correct model, by contrast, simply extrapolates its (correct) threshold. This is a near-violation of positivity: the data say little about such large shifts.
- b. The capped MTP answers a different, more realistic question: the effect of raising titers by δ *for those who could plausibly reach the higher titer*, leaving others unchanged. Its efficacy gains are smaller, because fewer people are shifted.

2.5 Wrap-up

- Both parts targeted causal parameters defined without a model: the ATE $\psi_{0,b}$ of smoking on each CpG, and the SVE, SVE_δ , under an MTP that shifts antibody titers. We estimated both with machine learning and efficient influence functions.
- In both, a linear model gives an appealing shortcut: $\psi_{0,b} = \beta_1$, and $\psi_{0,\delta} - \psi_{0,0} = \beta_1\delta$. These identities hold *only* if the linear model is correct; when it is not, the coefficient answers a different (or impossible) question.
- A misspecified GLM breaks a plug-in estimator, but not a doubly robust one whose other nuisance is right. Given a library that contains a good model, the Super Learner finds it.
- Ideas from classical biomarker analysis still matter: variance moderation for high-dimensional testing (Part 1), and the sampling design (Part 2).
- In Part 1, the biomarker was the *outcome*. In Part 2, it was a *mediator* of the vaccine's effect. Decomposing a total effect into pathways through biomarkers (direct and indirect effects) is the subject of causal mediation analysis; see, e.g., the `medoutcon` package.

References

- Benjamini, Yoav, and Yosef Hochberg. 1995. “Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing.” *Journal of the Royal Statistical Society: Series B (Methodological)* 57 (1): 289–300. <https://doi.org/10.1111/j.2517-6161.1995.tb02031.x>.
- Breslow, Norman E, Thomas Lumley, Christie M Ballantyne, Lloyd E Chambless, and Michal Kulich. 2009. “Using the Whole Cohort in the Analysis of Case-Cohort Data.” *American Journal of Epidemiology* 169 (11): 1398–405.
- Breslow, Norman, Brad McNeney, and Jon A Wellner. 2003. “Large Sample Theory for Semiparametric Regression Models with Two-Phase, Outcome Dependent Sampling.” *The Annals of Statistics* 31 (4): 1110–39.
- Díaz, Iván, and Mark J van der Laan. 2012. “Population Intervention Causal Effects Based on Stochastic Interventions.” *Biometrics* 68 (2): 541–49.
- Díaz, Iván, Nicholas Williams, Katherine L Hoffman, and Edward J Schenck. 2023. “Nonparametric Causal Effects Based on Longitudinal Modified Treatment Policies.” *Journal of the American Statistical Association* 118 (542): 846–57. <https://doi.org/10.1080/01621459.2021.1955691>.
- Gilbert, Peter B, Youyi Fong, David Benkeser, et al. 2021. *CoVPN COVID-19 Vaccine Efficacy Trial Immune Correlates Statistical Analysis Plan*. <https://doi.org/10.6084/m9.figshare.13198595>.
- Gilbert, Peter B, Youyi Fong, Nima S Hejazi, et al. 2024. “Four Statistical Frameworks for Assessing an Immune Correlate of Protection (Surrogate Endpoint) from a Randomized, Controlled, Vaccine Efficacy Trial.” *Vaccine* 42 (9): 2181–90. <https://doi.org/10.1016/j.vaccine.2024.02.071>.
- Gilbert, Peter B, Youyi Fong, Avi Kenny, and Marco Carone. 2023. “A Controlled Effects Approach to Assessing Immune Correlates of Protection.” *Biostatistics* 24 (4): 850–65. <https://doi.org/10.1093/biostatistics/kxac024>.
- Gilbert, Peter B, David C Montefiori, Adrian B McDermott, et al. 2021. “Immune Correlates Analysis of the mRNA-1273 COVID-19 Vaccine Efficacy Clinical Trial.” *Science* 375 (6576):

43–50. <https://doi.org/10.1126/science.abm3425>.

Gruber, Susan, and Mark J van der Laan. 2010. “A Targeted Maximum Likelihood Estimator of a Causal Effect on a Bounded Continuous Outcome.” *The International Journal of Biostatistics* 6 (1). <https://doi.org/10.2202/1557-4679.1260>.

Haneuse, Sebastian, and Andrea Rotnitzky. 2013. “Estimation of the Effect of Interventions That Modify the Received Treatment.” *Statistics in Medicine* 32 (30): 5260–77.

Hejazi, Nima S, and David C Benkeser. 2020. “txshift: Efficient Estimation of the Causal Effects of Stochastic Interventions in R.” *Journal of Open Source Software* 5 (54): 2447. <https://doi.org/10.21105/joss.02447>.

Hejazi, Nima S, Philippe Boileau, Mark J van der Laan, and Alan E Hubbard. 2023. “A Generalization of Moderated Statistics to Data Adaptive Semiparametric Estimation in High-Dimensional Biology.” *Statistical Methods in Medical Research* 32 (3): 539–54. <https://doi.org/10.1177/09622802221146313>.

Hejazi, Nima S, Xiaoying Shen, Lindsay N Carpp, et al. 2023. “Stochastic Interventional Approach to Assessing Immune Correlates of Protection: Application to the COVE mRNA-1273 Vaccine Trial.” *International Journal of Infectious Diseases* 137: 28–39. <https://doi.org/10.1016/j.ijid.2023.09.012>.

Hejazi, Nima S, Mark J van der Laan, Holly E Janes, Peter B Gilbert, and David C Benkeser. 2021. “Efficient Nonparametric Inference on the Effects of Stochastic Interventions Under Two-Phase Sampling, with Applications to Vaccine Efficacy Trials.” *Biometrics* 77 (4): 1241–53. <https://doi.org/10.1111/biom.13375>.

Huang, Ying, Nima S Hejazi, Bryan Blette, et al. 2023. “Stochastic Interventional Vaccine Efficacy and Principal Surrogate Analyses of Antibody Markers as Correlates of Protection Against Symptomatic COVID-19 in the COVE mRNA-1273 Trial.” *Viruses* 15 (10): 2029. <https://doi.org/10.3390/v15102029>.

Joehanes, Roby, Allan C. Just, Riccardo E. Marioni, et al. 2016. “Epigenetic Signatures of Cigarette Smoking.” *Circulation: Cardiovascular Genetics* 9 (5): 436–47. <https://doi.org/10.1161/CIRCGENETICS.116.001506>.

Plotkin, Stanley A, and Peter B Gilbert. 2012. “Nomenclature for Immune Correlates of Protection After Vaccination.” *Clinical Infectious Diseases* 54 (11): 1615–17.

Prentice, Ross L. 1986. “A Case-Cohort Design for Epidemiologic Cohort Studies and Disease Prevention Trials.” *Biometrika* 73 (1): 1–11.

- Prentice, Ross L. 1989. “Surrogate Endpoints in Clinical Trials: Definition and Operational Criteria.” *Statistics in Medicine* 8 (4): 431–40.
- Rose, Sherri, and Mark J van der Laan. 2011. “A Targeted Maximum Likelihood Estimator for Two-Stage Designs.” *The International Journal of Biostatistics* 7 (1): 1–21.
- Smyth, Gordon K. 2004. “Linear Models and Empirical Bayes Methods for Assessing Differential Expression in Microarray Experiments.” *Statistical Applications in Genetics and Molecular Biology* 3 (1): 1–25. <https://doi.org/10.2202/1544-6115.1027>.
- Su, Dan, Xuting Wang, Michelle R Campbell, et al. 2016. “Distinct Epigenetic Effects of Tobacco Smoking in Whole Blood and Among Leukocyte Subtypes.” *PLoS ONE* 11 (12): e0166486. <https://doi.org/10.1371/journal.pone.0166486>.
- Tuglus, Catherine, and Mark J van der Laan. 2009. “Modified FDR Controlling Procedure for Multi-Stage Analyses.” *Statistical Applications in Genetics and Molecular Biology* 8 (1). <https://doi.org/10.2202/1544-6115.1397>.
- White, J Emily. 1982. “A Two Stage Design for the Study of the Relationship Between a Rare Exposure and a Rare Disease.” *American Journal of Epidemiology* 115 (1): 119–28.