G-formula and Inverse Probability of Treatment Weighting
SASA '23 Short Course on Causal Inference

Erin E Gabriel and Andreas K Jensen

November 2023

1 Introduction to the exercise

In this exercise we are interested in estimating the average causal effect of quitting smoking on the amount of weight gain over a ten year period using observational data from the NHEFS study. We will look at two ways of estimating the causal effect, namely G-estimation and Inverse Probability of Treatment Weighting (IPTW).

In order to run the R commands in this exercise, you need to have the R packages `boot`, `geepack`, `stdReg2`, and `causaldata` installed. If you don't have them installed you should start by executing the following three lines

```
install.packages("boot")
install.packages("geepack")
remotes::install_github("sachsmc/stdReg2")
install.packages("causaldata")
```

You can now load the packages with the commands

```
library(boot)
library(geepack)
library(stdReg2)
library(causaldata)
```

We are now ready to do the exercise. We will first introduce the data set that is the foundation of the example.

2 Data set

We will be using data from the National Health and Nutrition Examination Survey Data I Epidemiologic Follow-up Study (NHEFS). The data was collected as part of a project by the National Center for Health Statistics and the National Institute on Aging in collaboration with other agencies of the United States Public Health Service. It was designed to investigate the relationships between clinical, nutritional and behavioral factors, and subsequent morbidity, mortality, and hospital utilization and changes in risk factors, functional limitation, and institutionalization.

The data set we use will has data from 1566 individuals and contains among others the following variables:

- `seqn`: person id
- `wt82_71`: weight gain in kilograms between 1971 and 1982
- `qsmk`: quit smoking between between 1st questionnaire and 1982, 1 = yes, 0 = no
- `sex`: 0 = male, 1 = female
- `race`: 0 = white, 1 = black or other
- `age`: age in years at baseline

- `education`: level of education in 1971, 1 = 8th grade or less, 2 = high school dropout, 3 = high school, 4 = college dropout, 5 = college or more
- `smokeintensity`: number of cigarettes smoked per day in 1971
- `smokeyrs`: number of years smoked
- `exercise`: level of physical activity in 1971, 0 = much exercise, 1 = moderate exercise, 2 = little or no exercise
- `active`: level of activity in 1971, 0 = very active, 1 = moderately active, 2 = inactive, 3 = missing information
- `wt71`: weight in kilograms in 1971
- `ht`: height in centimeters in 1971

You can load the data set by executing the following command

```
nhefs_dat <- causaldata::nhefs_complete
```

3 G-formula

Here we will use the G-formula to estimate the average causal effect of the exposure (quitting smoking) in the variable `qsmk` on the weight gain outcome in the variable `wt82_71`. First we will implement the G-formula directly and estimate the causal effect manually. Later we will see how this can also be done using the `stdReg2` package.

3.1 Implementing the G-formula directly

We recall that the implementation of the G-formula in the estimation of the average causal effect can be estimated as

$$\frac{1}{n}\sum_{i=1}^{n} E[Y \mid A = 1, L_i] - \frac{1}{n}\sum_{i=1}^{n} E[Y \mid A = 0, L_i]$$

In order to implement this, we start by making three copies of the data: one with the original exposure values (A_i in the above notation and `qsmk` in the data set), one where the exposure is provisionally set equal to 0, and one where it is set equal to 1. In the two data sets where we set the exposure provisionally equal to 0 and 1, we set the outcome `wt82_71` equal to missing (NA) since these values will be predicted from the model $E[Y_i \mid A_i, L_i]$.

The code below generates these data sets and append them together to form the data set `d_joint`.

```
d1 <- nhefs_dat
d2 <- nhefs_dat
d3 <- nhefs_dat

d1$interv <- -1

d2$interv <- 0
d2$qsmk <- 0
d2$wt82_71 <- NA

d3$interv <- 1
d3$qsmk <- 1
d3$wt82_71 <- NA

d_joint <- rbind(d1, d2, d3)
```

We will assume that the set of confounders in L is comprised of sex, race, age, education, number of cigarettes smoked per year, the number of years smoked, level of physical activity, and baseline weight in 1971. This is

equivalent to assuming that the counterfactual weight gain is independent of the exposure conditional on these variables.

For the specific forms of the conditional expectation required in the G-formula we assume a linear regression model with both linear and quadratic terms of the continuous covariates. We can fit this model as

```
m <- glm(wt82_71 ~ qsmk + sex + race + age + I(age^2) +
           as.factor(education) + smokeintensity + I(smokeintensity^2) +
           smokeyrs + I(smokeyrs^2) + as.factor(exercise) + as.factor(active) +
           wt71 + I(wt71^2),
         data = d_joint)
```

The predicted conditional expectations are obtained by

```
d_joint$meanY <- predict(m, d_joint)
```

and we can get the averages for each of the exposure conditions which is averaged over the empirical distribution of the confounders.

```
with(d_joint, tapply(meanY, list(interv), mean))
```

```
      -1        0        1
2.638300 1.747216 5.209838
```

The average treatment effect is then given by subtracting the two exposure levels means we we get that it is equal to $5.21 - 1.75 = 3.46$.

The standard inference is not applicable here because of the data augmentation. There are different other ways to obtain valid standard errors for the causal treatment effect. On general way is to apply Efron's bootstrap procedure, where we repeat the full procedure we just performed but on data sets resampled with replacement. The standard error can then consistently be estimated empirically from this bootstrap distribution.

In order to implement the bootstrap we first create a helper function that applies the G-formula on a resampled version of the data set.

```
boot.fun <- function(dat, index) {
  sampled.data <- dat[index,]
  m <- glm(wt82_71 ~ qsmk + sex + race + age + I(age^2) +
             as.factor(education) + smokeintensity + I(smokeintensity^2) +
             smokeyrs + I(smokeyrs^2) + as.factor(exercise) + as.factor(active) +
             wt71 + I(wt71^2),
           data = sampled.data)
  sampled.data$meanY <- predict(m, sampled.data, type = "response")
  std.mean <- with(sampled.data, tapply(meanY, list(interv), mean))
  unlist(c(std.mean, diff(std.mean[c(2,3)])))
}
```

The function boot from the `boot` package then performs the work for us. We set the seed for the random number generator to ensure reproducibility and we obtain 1,000 bootstrap estimates of the average treatment effects as well as their difference.

```
set.seed(12345)
bootres <- boot(d_joint, boot.fun, R = 1000)
bootres
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP
```

```
Call:
boot(data = d_joint, statistic = boot.fun, R = 1000)


Bootstrap Statistics :
    original       bias    std. error
t1* 2.638300 -0.002208276   0.1993378
t2* 1.747216 -0.005952713   0.2209957
t3* 5.209838 -0.012196515   0.4274693
t4* 3.462622 -0.006243803   0.4904186
```

95% confidence intervals using the normal approximation can be obtained as

```
boot.ci(bootres, index = 1, type = "norm")
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates

CALL :
boot.ci(boot.out = bootres, type = "norm", index = 1)

Intervals :
Level      Normal
95%   ( 2.250,  3.031 )
Calculations and Intervals on Original Scale
```

```
boot.ci(bootres, index = 2, type = "norm")
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates

CALL :
boot.ci(boot.out = bootres, type = "norm", index = 2)

Intervals :
Level      Normal
95%   ( 1.320,  2.186 )
Calculations and Intervals on Original Scale
```

```
boot.ci(bootres, index = 3, type = "norm")
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates

CALL :
boot.ci(boot.out = bootres, type = "norm", index = 3)

Intervals :
Level      Normal
95%   ( 4.384,  6.060 )
Calculations and Intervals on Original Scale
```

```
boot.ci(bootres, index = 4, type = "norm")
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates
```

```
CALL :
boot.ci(boot.out = bootres, type = "norm", index = 4)

Intervals :
Level      Normal
95%   ( 2.508,  4.430 )
Calculations and Intervals on Original Scale
```

3.2 Applying the stdReg package

We now show how the same calculation can be performed using the `stdReg2` package. The call to `standardize_glm` in the `stdReg2` package requires a regression model for the conditional expectation on the non-augmented data set. The command estimates the average treatment effect since we specifice `difference` and the contrast of interest.

```
m_stdreg <- standardize_glm(wt82_71 ~ qsmk + sex + race + age + I(age^2) +
                              as.factor(education) + smokeintensity +
                              I(smokeintensity^2) + smokeyrs + I(smokeyrs^2) +
                              as.factor(exercise) + as.factor(active) +
                              wt71 + I(wt71^2),
         data = nhefs_dat,
         values = list(qsmk = c(0,1)),
         contrasts = c("difference"),
         reference = 0)
m_stdreg
```

```
Outcome formula: wt82_71 ~ qsmk + sex + race + age + I(age^2) + as.factor(education) +
    smokeintensity + I(smokeintensity^2) + smokeyrs + I(smokeyrs^2) +
    as.factor(exercise) + as.factor(active) + wt71 + I(wt71^2)
Outcome family: gaussian
Outcome link function: identity
Exposure:  qsmk

Tables:
  qsmk Estimate Std.Error lower.0.95 upper.0.95
1    0     1.75     0.217       1.32       2.17
2    1     5.21     0.420       4.39       6.03

Reference level:  qsmk = 0
Contrast:  difference
  qsmk Estimate Std.Error lower.0.95 upper.0.95
1    0     0.00     0.000       0.00       0.00
2    1     3.46     0.466       2.55       4.38
```

We see that we obtain the same point estimate as our implementation by hand but slightly difference confiderence limits.

4 Inverse probability of treatment weighting (IPTW)

In this part of the exercise we will show how the average causal effect can be estimated by inverse probability of treatment weighting. Recall, that the IPTW estimator of the average causal effect is based on the estimates

$$\widehat{E[Y^a]} = \frac{\sum_{i:A_i=a} w_i Y_i}{\sum_{i:A_i=a} w_i}$$

where the weights w_i are given by

$$w_i = \frac{1}{P(A_i = 1 \mid L_i)} \quad \text{if } A_i = 1, \quad w_i = \frac{1}{1 - P(A_i = 1 \mid L_i)} \quad \text{if } A_i = 0$$

The probability of exposure given the confounders, $P(A_i = 1 \mid L_i)$, is called the propensity score and requires a specified model. Here the exposure is binary so we fit a logistic regression model to the variable `qsmk` and include the confounders in the same way are before.

```
fit <- glm(qsmk ~ sex + race + age + I(age^2) +
             as.factor(education) + smokeintensity + I(smokeintensity^2) +
             smokeyrs + I(smokeyrs^2) + as.factor(exercise) + as.factor(active) +
             wt71 + I(wt71^2),
           family = binomial(),
           data = nhefs_dat)
```

If is now possible to define the weights according to the definition

```
p.qsmk.obs <- ifelse(nhefs_dat$qsmk == 0,
                     1 - predict(fit, type = "response"),
                     predict(fit, type = "response"))

nhefs_dat$w <- 1 / p.qsmk.obs
```

The average causal effect can then be estimated by simply taking the differences in the weighted outcome averages.

```
wMean1 <- with(subset(nhefs_dat, nhefs_dat$qsmk == 1), sum(w*wt82_71) / sum(w))
wMean0 <- with(subset(nhefs_dat, nhefs_dat$qsmk == 0), sum(w*wt82_71) / sum(w))
wMean1 - wMean0
```

```
[1] 3.440535
```

You should now compare this result to the one you obtained by the G-formula.

Care must be taken when doing inference for the average treatment effect since the propensity scores themselves are estimated. One possibility is to use a so-called sandwich estimator for the variance. This can be done in the following way using the function `geeglm` from the `geepack` package.

```
gee.obj <- geeglm(wt82_71 ~ qsmk,
                  data = nhefs_dat,
                  std.err = "san.se",
                  weights = w,
                  id = seqn,
                  corstr = "independence")
summary(gee.obj)
```

```
Call:
geeglm(formula = wt82_71 ~ qsmk, data = nhefs_dat, weights = w,
    id = seqn, corstr = "independence", std.err = "san.se")

 Coefficients:
            Estimate Std.err  Wald Pr(>|W|)
(Intercept)   1.7800  0.2247 62.73 2.33e-15 ***
qsmk          3.4405  0.5255 42.87 5.86e-11 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation structure = independence
Estimated Scale Parameters:

            Estimate Std.err
(Intercept)    65.06   4.221
Number of clusters:   1566  Maximum cluster size: 1
```

```
beta <- coef(gee.obj)
SE <- sqrt(diag(summary(gee.obj)$cov.unscaled))
cbind(beta, beta - 1.96*SE, beta + 1.96*SE)
```

```
             beta
(Intercept) 1.780 1.340 2.220
qsmk        3.441 2.411 4.471
```

It is well-know that inference based on the sandwich estimator can lead to conservative confidence intervals, meaning that they have coverage probability greater than the nominal level and they are hence valid but wider than necessary.

A possible remedy to this issue is to base the inference on a bootstrap procedure similar to what we did for the G-formula. It is important here that everything is done in each bootstrap sample i.e., both estimating the propensity scores and calculating the difference in weighted averages. You can try to implement and run a similar bootstrap code as above to calculate a 95% confidence interval for the IPTW average treatment effect and compare it to the one obtained by using the sandwich estimator.

4.1 Stabilized weights

We will end this exercise by a short discussion about stabilized weights. A requirement for using the IPTW method is positivity, i.e., $0 < P(A_i \mid L_i) < 1$. However, even when positivity is satisfied, the estimator may still be unstable if the weights are close to zero or one.

One remedy to this problem is to use stabilized weights constructed by multiplying the usual weights with the marginal probability of exposure. The stabilized weights are formally defined as

$$w_i^S = \frac{P(A_i = 1)}{P(A_i = 1 \mid L_i)} \quad \text{if } A_i = 1, \quad w_i^S = \frac{1 - P(A_i = 1)}{1 - P(A_i = 1 \mid L_i)} \quad \text{if } A_i = 0$$

This can implemented in a straight forward manner as

```
denom.p <- predict(fit, type = "response")

numer.fit <- glm(qsmk ~ 1,
                 family = binomial(),
                 data = nhefs_dat)
numer.p <- predict(numer.fit, type = "response")

nhefs_dat$sw <- ifelse(nhefs_dat$qsmk == 0,
                       (1 - numer.p) / (1 - denom.p),
                       numer.p / denom.p)

gee.obj <- geeglm(wt82_71 ~ qsmk,
                  data = nhefs_dat,
                  std.err = "san.se",
                  weights = sw,
```

```
                  id = seqn,
                  corstr = "independence")

beta <- coef(gee.obj)
SE <- sqrt(diag(summary(gee.obj)$cov.unscaled))
cbind(beta, beta - 1.96*SE, beta + 1.96*SE)
```

```
             beta
(Intercept) 1.780 1.340 2.220
qsmk        3.441 2.411 4.471
```