R Lab 5 - TMLE

Laura B.Balzer

Biostat 683 - Intro. to Causal Inference

Goals:
1. Review the Causal Roadmap.
2. Code TMLE for the G-computation estimand.
3. Understand the basics of the `ltmle` package.
4. Use the `ltmle` package to explore the double robustness of TMLE.

Next lab:
We will implement the non-parametric bootstrap to estimate the standard error of the estimators. We will also use the sample variance of the estimated influence curve to obtain inference for TMLE.

1 Background

Dr. Alan Grant: "T-Rex doesn't want to be fed. He wants to hunt. Can't just suppress 65 million years of gut instinct." - Michael Crichton

We are interested in estimating the causal effect of prior experience with Dinosaurs on injury severity on Isla Nublar, the location of the InGen lab. Suppose we have data on the following variables:

- $W1$: sex (1 for male; 0 for female)
- $W2$: intelligence (scale from 0 to 1; with higher values for smarter)
- $W3$: handy/inventiveness (continuous and scaled; with larger, positive values for more MacGyver-ness)
- $W4$: running speed (continuous and scaled; with larger, positive values for faster)
- A: prior Dinosaur experience (1 for yes; 0 for no)
- Y: seriousness of injury (scale from 0 to 1; with higher values for more severe)

Let $W = (W1, W2, W3, W4)$ be the vector of baseline covariates.

http://www.thesambarnes.com/web-project-management/account-management-for-the-web-project-manager-part-1/

2 Causal Roadmap Rundown

1. **Specify the Question:**
   What is the causal effect of prior experience on injury severity in Jurassic Park?

2. **Specify the structural causal model (SCM) $\mathcal{M}^*$:**
   - Endogenous nodes: $X = (W, A, Y)$, where $W = (W1, W2, W3, W4)$ is the set of adjustment covariates (sex, intelligence, MacGyver-ness, running speed), A is prior Dinosaur experience, and Y is injury severity. For simplicity, we have condensed the baseline characteristics into a single node.
   - Background variables (unmeasured factors): $U = (U_W, U_A, U_Y) \sim \mathbb{P}_U$. We place no assumptions on the distribution $\mathbb{P}_U$.
   - Structural equations $\mathcal{F}$:

$$\begin{aligned} W &= f_W(U_W) \\ A &= f_A(W, U_A) \\ Y &= f_Y(W, A, U_Y) \end{aligned}$$

   We have not placed any restrictions on the functional forms.

3. **Specify the causal parameter of interest:**
   We are interested in the causal effect of prior Dinosaur experience on expected injury severity on Isla Nublar (i.e., the average treatment effect):

$$\Psi^*(\mathbb{P}^*) = \mathbb{E}^*(Y_1) - \mathbb{E}^*(Y_0)$$

   where Y_a is the counterfactual outcome (injury severity), if possibly contrary to fact, the participant had Dinosaur-experience $A = a$.

4. **Specify the link between the structural causal model (SCM) and the observed data:**
   We assume that the observed data $O = (W, A, Y) \sim \mathbb{P}_0$ were generated by sampling n times from a data generating compatible with the SCM. The statistical model $\mathcal{M}$ for the set of allowed distributions of the observed data is non-parametric.

5. **Assess identifiability:**
   In the original structural causal model $\mathcal{M}^*$, the target causal parameter is not identified from the observed data distribution. For identifiability to hold, we would need the following independence assumptions to hold: $U_A \perp\!\!\!\perp U_Y$ and (i) $U_A \perp\!\!\!\perp U_W$, or (ii) $U_Y \perp\!\!\!\perp U_W$. We also need the positivity assumption to hold

$$min_{a \in \mathcal{A}} \ \mathbb{P}_0(A = a | W = w) > 0$$

   for all w for which $\mathbb{P}_0(W = w) > 0$. In terms of our example, there must be a positive probability of being dinosaur-experienced and not being dinosaur-experienced within values of the adjustment covariates.

6. **Specify the statistical estimand:**
   Even though the identifiability assumptions do not hold in the original structural causal model $\mathcal{M}^*$, we can still specify a statistical estimand that would equal the wished-for causal effect if the identifiability assumptions did, in fact, hold. The target statistical estimand is given by the G-Computation formula:

$$\Psi(\mathbb{P}_0) = \mathbb{E}_0\big[\mathbb{E}_0(Y|A = 1, W) - \mathbb{E}_0(Y|A = 0, W)\big]$$

   In these simulations, the true value of the statistical estimand was -6.2%.

7. **Estimate the chosen parameter of the observed data distribution:**

   (a) **Simple substitution estimator based on the G-Computation formula:**

$$\hat{\Psi}_{SS}(\hat{\mathbb{P}}) = \frac{1}{n} \sum_{i=1}^{n} \left(\hat{\mathbb{E}}(Y|A = 1, W_i) - \hat{\mathbb{E}}(Y|A = 0, W_i) \right)$$

where $\hat{\mathbb{P}}$ is the empirical distribution and $\hat{\mathbb{E}}(Y|A,W)$ is the estimator of the conditional mean outcome given the exposure (experience with Dinosaurs or not) and baseline covariates $\mathbb{E}_0(Y|A,W)$.
- Consistency of the simple (non-targeted) substitution estimator depends on consistent estimation of the conditional mean outcome $\mathbb{E}_0(Y|A,W)$.

(b) **Inverse probability weighted estimator (IPTW)**:

$$\hat{\Psi}_{IPTW}(\hat{\mathbb{P}}) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\mathbb{I}(A_i=1)}{\hat{\mathbb{P}}(A=1|W_i)} - \frac{\mathbb{I}(A_i=0)}{\hat{\mathbb{P}}(A=0|W_i)}\right)Y_i$$

where $\hat{\mathbb{P}}(A=1|W)$ is the estimator of the exposure mechanism (i.e., the conditional probability of having Dinosaur experience given the baseline covariates).
- Consistency of IPTW estimators depends on consistent estimation of the exposure mechanism $\mathbb{P}_0(A|W)$.

(c) **Targeted maximum likelihood estimation (TMLE):**

$$\hat{\Psi}_{TMLE}(\hat{\mathbb{P}}) = \frac{1}{n}\sum_{i=1}^{n}\left[\hat{\mathbb{E}}^*(Y|A=1,W_i) - \hat{\mathbb{E}}^*(Y|A=0,W_i)\right]$$

where $\hat{\mathbb{E}}^*(Y|A,W)$ denotes the targeted estimator of the conditional mean outcome given the exposure and baseline covariates $\mathbb{E}_0(Y|A,W)$.
- Implementation requires estimation of both the conditional mean function $\mathbb{E}_0(Y|A,W)$ and the exposure mechanism $\mathbb{P}_0(A|W)$.
- Double robust estimators are consistent if either $\mathbb{E}_0(Y|A,W)$ or $\mathbb{P}_0(A|W)$ is estimated consistently.
- If both $\mathbb{E}_0(Y|A,W)$ and $\mathbb{P}_0(A|W)$ are estimated consistently at reasonable rates, TMLE will be efficient and achieve the lowest possible asymptotic variance over a large class of estimators.
- These asymptotic properties describe what happens when sample size goes to infinity and also translate into lower bias and variance in finite samples.

If we apply an estimator to our observed data (n i.i.d. copies of O drawn from $\mathbb{P}_0$), we get an estimate (i.e., a number). The estimator is function of random variables; so it is a random variable. It has a distribution, which we can study theoretically or using simulations.

Note: An estimator is *consistent* if the point estimates converge (in probability) to the estimand as sample size $n \to \infty$.

8. **Inference and interpret results:**
In the next lab, we will implement the non-parametric bootstrap for variance estimation for the three types of estimators. We will use the sample variance of the estimated influence curve to obtain inference for the TMLE.

3 Import and explore data set `RLab5.TMLE.csv`.

1. Set the seed to 252.
2. Use the `read.csv` function to import the dataset and assign it to dataframe `ObsData`.
3. Use the `head` and `summary` functions to explore the data.
4. Use the `nrow` function to count the number of participants in the data set.

Solution:

```
> set.seed(252)

> #  Import the data set and assign it to object ObsData; explore
> ObsData<- read.csv("RLab5.TMLE.csv")
> names(ObsData)

[1] "W1" "W2" "W3" "W4" "A"  "Y"

> head(ObsData)

  W1        W2         W3         W4 A           Y
1  0 0.7619739  0.1098437  0.1521844 0 0.422755804
2  0 0.9329098  0.4729021  0.3103004 1 0.128175775
3  1 0.4706785  0.2570524 -0.3910361 0 0.689871569
4  1 0.6035881 -0.2412886 -0.2351033 0 0.680570406
5  0 0.4849897  0.3973982  0.9405205 0 0.351130899
6  1 0.1088063  0.6658812  1.5389331 0 0.002012601

> summary(ObsData)

       W1              W2                  W3                 W4
 Min.   :0.000   Min.   :0.001837   Min.   :-2.93279   Min.   :-2.853309
 1st Qu.:0.000   1st Qu.:0.221433   1st Qu.:-0.59761   1st Qu.:-0.643772
 Median :0.000   Median :0.460235   Median : 0.01420   Median :-0.047242
 Mean   :0.496   Mean   :0.481484   Mean   : 0.04708   Mean   :-0.002612
 3rd Qu.:1.000   3rd Qu.:0.745637   3rd Qu.: 0.77019   3rd Qu.: 0.691945
 Max.   :1.000   Max.   :0.996077   Max.   : 3.92945   Max.   : 3.399766
       A               Y
 Min.   :0.000   Min.   :0.0000
 1st Qu.:0.000   1st Qu.:0.0520
 Median :0.000   Median :0.2978
 Mean   :0.312   Mean   :0.3009
 3rd Qu.:1.000   3rd Qu.:0.5183
 Max.   :1.000   Max.   :0.7692

> # can get the dimensions
> nrow(ObsData)

[1] 250
```

4 Implement TMLE for the G-computation estimand

1. **Load the `SuperLearner` package (Polley et al., 2018). Then specify the Super Learner library with the following algorithms: `SL.mean`, `SL.glm` and `SL.step.interaction`.** In practice, we would want to use a larger library with a mixture of simple (e.g., parametric) and more aggressive algorithms.

```
> library("SuperLearner")
> # specify the library
> SL.library<- c("SL.mean", "SL.glm", "SL.step.interaction")
```

2. **Use Super Learner to estimate $\mathbb{E}_0(Y|A,W)$, which is the expected injury severity given the exposure (prior experience) and baseline covariates.**

   (a) Create dataframe `X` consisting of the covariates $(W1, W2, W3, W4)$ and the exposure A. Also create dataframe `X1` where A has been set to 1, and create dataframe `X0` where A has been set to 0.

   (b) Estimate $\mathbb{E}_0(Y|A,W)$ by running `SuperLearner`. Call this object `SL.outcome`. Be sure to specify the `SL.library` and the appropriate `family`.

   ```
   > SL.outcome<- SuperLearner(Y=ObsData$Y, X=X, SL.library=SL.library, family="binomial")
   ```

   Note: we are using the logit-link even though the outcome Y is not binary, but it is bounded in [0,1]. See Chpt7 of 'Targeted Learning' book for details.

   (c) Use the `predict` function to obtain initial estimates of the expected outcome, given the observed exposure and covariates $\hat{\mathbb{E}}(Y|A,W)$.

   ```
   > expY.givenAW <- predict(SL.outcome, newdata=X)$pred
   ```

   The argument `newdata=X` specifies that we want to predict the outcome using as input the observed exposure and covariates.

   (d) Also obtain the initial estimates of the expected outcome for all participants under the exposure $\hat{\mathbb{E}}(Y|A=1,W)$. Now we specify `newdata=X1` to predict the outcome using as input `X1`, where $A=1$ for all participants.

   ```
   > expY.given1W<- predict(SL.outcome, newdata=X1)$pred
   ```

   (e) Finally, obtain the initial estimates of the expected outcome for all participants under no exposure $\hat{\mathbb{E}}(Y|A=0,W)$. Now we specify `newdata=X0` to predict the outcome using as input `X0`, where $A=0$ for all participants.

   ```
   > expY.given0W<- predict(SL.outcome, newdata=X0)$pred
   ```

   (f) Evaluate the simple substitution estimator by plugging these Super Learner-based estimates $\hat{\mathbb{E}}(Y|A=1,W)$ and $\hat{\mathbb{E}}(Y|A=0,W)$ into the target parameter mapping:

$$\hat{\Psi}_{SS}(\hat{\mathbb{P}}) = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{\mathbb{E}}(Y|A=1,W_i) - \hat{\mathbb{E}}(Y|A=0,W_i)\right)$$

   Note: This step is not part of the TMLE algorithm, but done for comparison.

3. **Estimate the propensity score $\mathbb{P}_0(A=1|W)$, which is the conditional probability of Dinosaur experience given baseline covariates.**

   (a) Estimate $\mathbb{P}_0(A=1|W)$ by running `SuperLearner`. Call this object `SL.exposure`. Since we are estimating the exposure mechanism, specify the-outcome-for-prediction with `Y=ObsData$A` and the predictors as the baseline covariates with `X=subset(ObsData, select= -c(A,Y))`. Use the same library for simplicity. (This is not a requirement.)

   ```
   > X <- subset(ObsData, select= -c(A,Y))
   > SL.exposure<- SuperLearner(Y=ObsData$A, X=X,
   +                            SL.library=SL.library, family="binomial")
   ```

   (b) The predicted probability of being Dinosaur experienced, given the participant's baseline characteristics $\hat{\mathbb{P}}(A=1|W)$, can be accessed with `SL.exposure$SL.predict`. (Equivalently, we could `predict` using as `newdata=X`.)

      i. Assign the predicted probability of being experienced $\hat{\mathbb{P}}(A=1|W)$ to `probA1.givenW`:

      ```
      > probA1.givenW<- SL.exposure$SL.predict
      ```

ii. Assign the predicted probability of not being experienced $\hat{\mathbb{P}}(A = 0|W)$ to `probA0.givenW`.

iii. Look at the distribution of estimated probabilities: $\hat{\mathbb{P}}(A = 1|W)$ and $\hat{\mathbb{P}}(A = 0|W)$.

4. **Use these estimates to create the clever covariate for each participant i:**

$$\hat{H}(A_i, W_i) = \left(\frac{\mathbb{I}(A_i = 1)}{\hat{\mathbb{P}}(A = 1|W_i)} - \frac{\mathbb{I}(A_i = 0)}{\hat{\mathbb{P}}(A = 0|W_i)} \right)$$

(a) Calculate `H.AW` for each participant:

```
> H.AW<- as.numeric(ObsData$A==1)/probA1.givenW - as.numeric(ObsData$A==0)/probA0.givenW
```

For participants with $A = 1$, the clever covariate is 1 over the predicted probability of being experienced given the baseline covariates. Among participants with $A = 0$, the clever covariate is -1 over the predicted probability of not being experienced given the baseline covariates.

Note: the form of the clever covariate is dependent on the statistical target parameter. See Appendix for an alternative implementation that simultaneously targets (under identifiability assumptions) the effect on the absolute (i.e., difference) and relative (i.e., ratio or odds ratio) scale.

(b) Also evaluate the clever covariate at $A = 1$ and $A = 0$ for all participants. Call the resulting vectors `H.1W` and `H.0W`, respectively.

(c) Evaluate the IPTW estimator by taking the empirical mean of the weighted observations:

$$\begin{aligned}\hat{\Psi}_{IPTW}(\hat{\mathbb{P}}) &= \frac{1}{n}\sum_{i=1}^{n}\left[\frac{\mathbb{I}(A_i = 1)}{\hat{\mathbb{P}}(A = 1|W_i)} - \frac{\mathbb{I}(A_i = 0)}{\hat{\mathbb{P}}(A = 0|W_i)}\right] Y_i \\ &= \frac{1}{n}\sum_{i=1}^{n} \hat{H}(A_i, W_i) \times Y_i\end{aligned}$$

As before, this is not part of the TMLE algorithm, but implemented for comparison.

5. **Target the initial estimator of the conditional mean outcome $\hat{\mathbb{E}}(Y|A, W)$ with information in the estimated propensity score $\hat{\mathbb{P}}(A = 1|W)$.**

(a) Run a univariate regression of the outcome Y on the clever covariate $\hat{H}(A, W)$ with the (logit of the) initial estimates as offset. Specifically, we estimate the coefficient ϵ by fitting the following logistic regression model

$$logit\big[\hat{\mathbb{E}}^*(Y|A, W)\big] = logit\big[\hat{\mathbb{E}}(Y|A, W)\big] + \epsilon\hat{H}(A, W).$$

Note there is no intercept (i.e., there is no β_0 term), and the coefficient on the (*logit* of the) initial estimator is set to 1.

```
> logitUpdate<- glm(ObsData$Y ~ -1 +offset(qlogis(expY.givenAW)) + H.AW,
+                   family='binomial')
```

- We are again calling the `glm` function to fit a generalized linear model.
- On the left hand side of the formula, we have the outcome Y.
- On the right hand side of the formula, we suppress the intercept by including -1 and use as `offset` the *logit* of our initial Super Learner estimates `expY.givenAW`.
- In `R`, $logit(x) = log(x/(1 - x))$ function is given by `qlogis(x)`.
- The only main term in the regression is the clever covariate $\hat{H}(A, W)$.
- Including `family='binomial'` runs logistic regression.
- Ignore any warning message.

```
> # we can examine the output by typing
> summary(logitUpdate)
```

(b) Let `epsilon` denote the resulting maximum likelihood estimate of the coefficient on the clever covariate `H.AW`.

```
> epsilon<- logitUpdate$coef
> epsilon
```

(c) Update the initial estimate of $\hat{\mathbb{E}}(Y|A,W)$ according to the fluctuation model:

$$logit\big[\hat{\mathbb{E}}^*(Y|A,W)\big] = logit\big[\hat{\mathbb{E}}(Y|A,W)\big] + \hat{\epsilon}\hat{H}(A,W)$$
$$\hat{\mathbb{E}}^*(Y|A,W) = logit^{-1}\Big[logit\big[\hat{\mathbb{E}}(Y|A,W)\big] + \hat{\epsilon}\hat{H}(A,W)\Big]$$

```
> expY.givenAW.star<- plogis(qlogis(expY.givenAW)+ epsilon*H.AW)
```

In `R`, the inverse-*logit* function is given by `plogis(x)`.

(d) Plug-in the estimated coefficient $\hat{\epsilon}$ to yield targeted estimates of the expected outcome under the exposure $\hat{\mathbb{E}}^*(Y|A=1,W)$ and under no exposure $\hat{\mathbb{E}}^*(Y|A=0,W)$:

$$\hat{\mathbb{E}}^*(Y|A=1,W) = logit^{-1}\Big[logit\big[\hat{\mathbb{E}}(Y|A=1,W)\big] + \hat{\epsilon}\hat{H}(1,W)\Big]$$
$$\hat{\mathbb{E}}^*(Y|A=0,W) = logit^{-1}\Big[logit\big[\hat{\mathbb{E}}(Y|A=0,W)\big] + \hat{\epsilon}\hat{H}(0,W)\Big]$$

Recall $\hat{H}(1,W)$ is the clever covariate evaluated for all participants under the exposure, and $\hat{H}(0,W)$ is the clever covariate evaluated for all participants under no exposure.

```
> expY.given1W.star <- plogis( qlogis(expY.given1W.star)+ epsilon*H.1W)
> expY.given0W.star <- plogis( qlogis(expY.given0W.star)+ epsilon*H.0W)
```

(e) *Optional*: Try updating again. What is updated $\hat{\epsilon}$?

6. **Step 6. Estimate the statistical parameter by substituting the targeted predictions into the G-Computation formula.**

Estimate $\Psi(\mathbb{P}_0)$ by averaging the difference in the targeted predictions:

$$\Psi_{TMLE}(\hat{\mathbb{P}}) = \frac{1}{n}\sum_{i=1}^{n}\Big[\hat{\mathbb{E}}^*(Y|A=1,W_i) - \hat{\mathbb{E}}^*(Y|A=0,W_i)\Big]$$

```
> PsiHat.TMLE <- mean(expY.given1W.star- expY.given0W.star)
> PsiHat.TMLE
```

Solution:

```
> #---------------------------------------
> # 1. Load the Super Learner package and specify the library
> #---------------------------------------
> library("SuperLearner")
> # specify the library
> SL.library<- c("SL.mean", "SL.glm", "SL.step.interaction")


> #---------------------------------------
> # 2. Estimate E_0(Y|A,W) with Super Learner
> #---------------------------------------
> # dataframe X with baseline covariates and exposure
> X<-subset(ObsData, select=c(A, W1, W2, W3, W4))
> # set the exposure=1 in X1 and the exposure=0 in X0
```

```
> X1 <- X0<-X
> X1$A <- 1          # under exposure
> X0$A <- 0         # under no exposure

> # call Super Learner
> SL.outcome<- SuperLearner(Y=ObsData$Y, X=X, SL.library=SL.library, family="binomial")
> SL.outcome

Call:
SuperLearner(Y = ObsData$Y, X = X, family = "binomial", SL.library = SL.library)



                             Risk       Coef
SL.mean_All              0.05513011 0.00000000
SL.glm_All               0.04892818 0.09096398
SL.step.interaction_All 0.01892797 0.90903602

> # get the expected injury severity, given the observed exposure and covariates
> expY.givenAW <- predict(SL.outcome, newdata=X)$pred
> # expected injury severity, given A=1 and covariates
> expY.given1W<- predict(SL.outcome, newdata=X1)$pred
> # expected injury severity, given A=0 and covariates
> expY.given0W<- predict(SL.outcome, newdata=X0)$pred

> # the fitted value at the observed exposure should equal the fitted value
> # under when A=a
> tail(data.frame(A=ObsData$A, expY.givenAW, expY.given1W, expY.given0W))

    A expY.givenAW expY.given1W expY.given0W
245 1  0.004996601  0.004996601   0.01053257
246 0  0.298184186  0.283277378   0.29818419
247 0  0.332424202  0.319709296   0.33242420
248 1  0.265414703  0.265414703   0.28050478
249 0  0.369757560  0.356621656   0.36975756
250 0  0.539733749  0.522843649   0.53973375

> # note the simple substitution estimator would be
> PsiHat.SS<-mean(expY.given1W - expY.given0W)
> PsiHat.SS

[1] -0.01391576

> #---------------------------------------
> # 3.  Estimate P_0(A=1|W) with Super Learner
> #---------------------------------------

> # call Super Learner for the exposure mechanism
> X <- subset(ObsData, select= -c(A,Y))
> SL.exposure<- SuperLearner(Y=ObsData$A, X=X,
+                       SL.library=SL.library, family="binomial")
> SL.exposure
```

```
Call:
SuperLearner(Y = ObsData$A, X = X, family = "binomial", SL.library = SL.library)



                            Risk      Coef
SL.mean_All             0.2163674 0.6130339
SL.glm_All              0.2271971 0.0000000
SL.step.interaction_All 0.2196667 0.3869661


> # generate the predicted prob of being experienced, given baseline cov
> probA1.givenW<- SL.exposure$SL.predict
> # above is equivalent to
> check <- predict(SL.exposure, newdata=X)$pred
> sum(probA1.givenW != check)


[1] 0


> # generate the predicted prob of not being experienced, given baseline cov
> probA0.givenW<- 1- probA1.givenW


> # summary of the predicted probabilities of being exposed/not, given the covariates
> summary(data.frame(probA1.givenW, probA0.givenW))


 probA1.givenW    probA0.givenW
 Min.   :0.2535   Min.   :0.4440
 1st Qu.:0.2825   1st Qu.:0.6739
 Median :0.2951   Median :0.7049
 Mean   :0.3120   Mean   :0.6880
 3rd Qu.:0.3261   3rd Qu.:0.7175
 Max.   :0.5560   Max.   :0.7465


> #-----------------------------------------------
> # 4. Create the clever covariate H(A,W) for each participant
> #-----------------------------------------------
> H.AW<- as.numeric(ObsData$A==1)/probA1.givenW - as.numeric(ObsData$A==0)/probA0.givenW
> # also want to evaluate the clever covariates at A=1 and A=0 for all participants
> H.1W<- 1/probA1.givenW
> H.0W<- -1/probA0.givenW


> tail(data.frame(ObsData$A, H.AW, H.1W, H.0W))


    ObsData.A      H.AW     H.1W      H.0W
245         1  1.798447 1.798447 -2.252431
246         0 -1.429244 3.329678 -1.429244
247         0 -1.403768 3.476668 -1.403768
248         1  3.315430 3.315430 -1.431885
249         0 -1.500869 2.996528 -1.500869
250         0 -1.393508 3.541245 -1.393508
```

```
> #IPTW estimator of the G-computation formula:
> PsiHat.IPTW <-mean( H.AW*ObsData$Y)
> PsiHat.IPTW


[1] -0.1344216


> #-----------------------------------------
> # 5. Update the initial estimator of E_0(Y|A,W)
> # run logistic regression of Y on H.AW using the logit of the esimates as offset
> #-----------------------------------------
> logitUpdate<- glm(ObsData$Y ~ -1 +offset(qlogis(expY.givenAW)) + H.AW,
+                   family='binomial')
> epsilon <- logitUpdate$coef
> epsilon


       H.AW
-0.07931157


> # obtain the targeted estimates
> expY.givenAW.star<- plogis(qlogis(expY.givenAW)+ epsilon*H.AW)
> expY.given1W.star<- plogis(qlogis(expY.given1W)+ epsilon*H.1W)
> expY.given0W.star<- plogis(qlogis(expY.given0W)+ epsilon*H.0W)


> # trying updating again:
> coef(glm(ObsData$Y ~ -1 +offset(qlogis(expY.givenAW.star)) + H.AW, family=binomial))


        H.AW
2.037722e-18


> # since the clever covariate is not changing, updating will not have any impact


> # 6. Estimate Psi(P_0) as the empirical mean of the difference in the targeted
> # outcomes under A=1 and A=0
> PsiHat.TMLE <- mean(expY.given1W.star - expY.given0W.star)


> # comparing the estimates...
> c(PsiHat.SS, PsiHat.IPTW, PsiHat.TMLE)


[1] -0.01391576 -0.13442160 -0.07992069
```

The point estimate from the simple substitution estimator using Super Learner for $\mathbb{E}_0(Y|A,W)$ was -1.4%. The point estimate from IPTW using Super Learner for $\mathbb{P}_0(A|W)$ was -13.4%. The point estimate from TMLE was -8%. In these simulations, the true value of the statistical estimand was -6.2%. To evaluate the performance of these estimators (e.g., bias and variance), we would draw another independent sample of size n, implement the 3 estimators (with the same Super Learner library), and repeat 500 or so times.

5 The basics of the `ltmle` package

The `ltmle` package (Schwab et al., 2017) expands the previous `tmle` package (Gruber and van der Laan, 2012). The `ltmle` package estimates parameters corresponding to point-treatment exposures, longitudinal exposures, marginal structural working models, dynamic treatment regimes, and much more!

1. **Reset the seed to 252. Load the `SuperLearner` and `ltmle` packages.**

```
> set.seed(252)
> library('SuperLearner')
> library('ltmle')
> # we can learn a lot more about the function by reading the help file
> ?ltmle
```

   - The basic input to the function is the dataset `data`, the exposure variable(s) `Anodes`, the outcome(s) `Ynodes`, and the exposure levels of interest `abar`.
   - The user can also specify censoring variables `Cnodes`, time-dependent covariates `Lnodes`, weights `observation.weights`, and the independent unit `id`. (See the help file for more information.)
   - Initial estimates of the conditional mean outcome $\mathbb{E}_0(Y|A,W)$ can be estimated according to a user-specified regression formula (`Qform`) or estimated with Super Learner (`SL.library`).
   - Initial estimates of the exposure mechanism $\mathbb{P}_0(A|W)$ can be estimated according to a user-specified regression formula (`gform`) or estimated with Super Learner (`SL.library`).

2. **Call the `ltmle` function using Super Learner to estimate the conditional mean outcome $\mathbb{E}_0(Y|A,W)$ and the exposure mechanism $\mathbb{P}_0(A|W)$. Use the `summary` function to obtain point estimates and get inference.**

```
> ltmle.SL<- ltmle(data=ObsData, Anodes='A', Ynodes='Y', abar=list(1,0),
+                       SL.library=SL.library)
> summary(ltmle.SL)
```

   Here, `abar=list(1,0)` specifies the comparison of interest: all exposed ($A = 1$) vs. all unexposed ($A = 0$).

3. **Use the tmle package to explore performance under model mis-specification**

   (a) Using main terms parametric regression

```
> ltmle.parametric<- ltmle(data=ObsData, Anodes='A', Ynodes='Y', abar=list(1,0),
+                       Qform=c(Y="Q.kplus1 ~ A+W1+W2+W3+W4"), gform="A~W1+W2+W3+W4")
> summary(ltmle.parametric)
```

   (b) Using unadjusted estimators

```
> # adding a dummy variable to observed data
> ObsData<- data.frame(U=1, ObsData)
> ltmle.unadj <- ltmle(data=ObsData, Anodes='A', Ynodes='Y', abar=list(1,0),
+                       Qform=c(Y="Q.kplus1 ~ A"), gform="A~U")
> summary(ltmle.unadj)
```

4. **Use the tmle package to explore double robustness.**

Solution:

```
> #----------------------------------------
> # 0. Re-loading the observed data for the workshop & resetting the seed
> #----------------------------------------
> ObsData<- read.csv("RLab5.TMLE.csv")
> set.seed(252)


> #----------------------------------------
> # 1. Load the Super Learner & ltmle packages
> #----------------------------------------
> library("SuperLearner")
> library("ltmle")


> ?ltmle


> #----------------------------------------
> # 2. call ltmle with Super Learner (same libraries)
> #----------------------------------------
> ltmle.SL<- ltmle(data=ObsData, Anodes='A', Ynodes='Y', abar=list(1,0),
+                       SL.library=SL.library, estimate.time = F)
> summary(ltmle.SL)


Estimator:  tmle
Call:
ltmle(data = ObsData, Anodes = "A", Ynodes = "Y", abar = list(1,
    0), SL.library = SL.library, estimate.time = F)

Treatment Estimate:
   Parameter Estimate:  0.24219
    Estimated Std Err:  0.016225
              p-value:  <2e-16
    95% Conf Interval: (0.21039, 0.27399)

Control Estimate:
   Parameter Estimate:  0.32226
    Estimated Std Err:  0.01614
              p-value:  <2e-16
    95% Conf Interval: (0.29063, 0.3539)

Additive Treatment Effect:
   Parameter Estimate:  -0.080076
    Estimated Std Err:  0.016186
              p-value:  7.5252e-07
    95% Conf Interval: (-0.1118, -0.048352)
```

The `ltmle` package provides estimates and inference for (under identifiability assumptions) the expected outcome under the exposure ("Treatment Estimate"), the expected outcome under no exposure ("Control Estimate"), and the Additive Treatment Effect. If the outcome is binary, the package will also return estimates of the risk ratio and odds ratio. (See Example 1 in the help file.)

The point estimates from `ltmle` package might differ from our code for several reasons. First, the `ltmle` package updates initial estimates by including the clever "covariate" as a weight (instead of as a main term). Second, the package also allows for estimation and inference of the marginal risk under the exposure, the

marginal risk under no exposure, the marginal risk difference, the marginal risk ratio and the marginal odds ratio (after controlling for measured confounders). In our code, we used a one-dimensional clever covariate for simplicity and to focus on the marginal risk difference. Third, the `ltmle` package bounds the estimated propensity scores. This bounding is included to deal with theoretical and practical positivity violations. For further discussion, see Petersen et al. (2012). Finally, the Super Learner algorithm could split the data into different folds. (This is why we reset the seed.)

```
> #----------------------------------------
> # 3a. call ltmle with main terms parametric regression for both E(Y|A,W) & P(A=1|W)
> #----------------------------------------
> ltmle.parametric<- ltmle(data=ObsData, Anodes='A', Ynodes='Y', abar=list(1,0),
+                       Qform=c(Y="Q.kplus1 ~ A+W1+W2+W3+W4"), gform="A~W1+W2+W3+W4",
+                       estimate.time = F)
> summary(ltmle.parametric)

Estimator:  tmle
Call:
ltmle(data = ObsData, Anodes = "A", Ynodes = "Y", Qform = c(Y = "Q.kplus1 ~ A+W1+W2+W3+W4"),
    gform = "A~W1+W2+W3+W4", abar = list(1, 0), estimate.time = F)

Treatment Estimate:
   Parameter Estimate:  0.19199
    Estimated Std Err:  0.023627
              p-value:  4.4455e-16
    95% Conf Interval: (0.14568, 0.2383)

Control Estimate:
   Parameter Estimate:  0.34724
    Estimated Std Err:  0.017655
              p-value:  <2e-16
    95% Conf Interval: (0.31263, 0.38184)

Additive Treatment Effect:
   Parameter Estimate:  -0.15525
    Estimated Std Err:  0.029348
              p-value:  1.2241e-07
    95% Conf Interval: (-0.21277, -0.097727)


> #----------------------------------------
> # 3b. call ltmle with unadjusted
> # adding a dummy variable to observed data
> #----------------------------------------
> ObsData<- data.frame(U=1, ObsData)
> ltmle.unadj <- ltmle(data=ObsData, Anodes='A', Ynodes='Y', abar=list(1,0),
+                       Qform=c(Y="Q.kplus1 ~ A"), gform="A~U",
+                       estimate.time = F)
> summary(ltmle.unadj)

Estimator:  tmle
Call:
ltmle(data = ObsData, Anodes = "A", Ynodes = "Y", Qform = c(Y = "Q.kplus1 ~ A"),
    gform = "A~U", abar = list(1, 0), estimate.time = F)
```

```
Treatment Estimate:
   Parameter Estimate:  0.19364
    Estimated Std Err:  0.022404
              p-value:  <2e-16
    95% Conf Interval: (0.14973, 0.23755)

Control Estimate:
   Parameter Estimate:  0.34948
    Estimated Std Err:  0.017844
              p-value:  <2e-16
    95% Conf Interval: (0.3145, 0.38445)

Additive Treatment Effect:
   Parameter Estimate:  -0.15584
    Estimated Std Err:  0.028641
              p-value:  5.297e-08
    95% Conf Interval: (-0.21197, -0.099703)


> #----------------------------------------
> # 4a. explore double robustness
> #----------------------------------------
> ltmle.DR<- ltmle(data=ObsData, Anodes='A', Ynodes='Y', abar=list(1,0),
+                  SL.library=SL.library,
+                  gform="A~U", estimate.time = F)
> summary(ltmle.DR)


Estimator:  tmle
Call:
ltmle(data = ObsData, Anodes = "A", Ynodes = "Y", gform = "A~U",
    abar = list(1, 0), SL.library = SL.library, estimate.time = F)

Treatment Estimate:
   Parameter Estimate:  0.24157
    Estimated Std Err:  0.015686
              p-value:  <2e-16
    95% Conf Interval: (0.21083, 0.27231)

Control Estimate:
   Parameter Estimate:  0.32255
    Estimated Std Err:  0.016329
              p-value:  <2e-16
    95% Conf Interval: (0.29055, 0.35456)

Additive Treatment Effect:
   Parameter Estimate:  -0.080982
    Estimated Std Err:  0.015799
              p-value:  2.9637e-07
    95% Conf Interval: (-0.11195, -0.050016)
```

Formally, an estimator is *consistent* if the point estimates converge (in probability) to the estimand as sample size $n \to \infty$. This is an asymptotic property. Here, we only have one sample of size $n = 1,000$.

To evaluate the consistency of TMLE, we would need to do multiple runs at increasing samples sizes, e.g., $n = 500$, $n = 5,000$, $n = 50,000$, $n = 500,000$.

6 Bonus: Food-for-thought

How would you modify TMLE to adjust for differential ascertainment (i.e., missingess) on the outcome?

7 Appendix A: Alternative TMLE implementations

Let $\psi_a = \mathbb{E}_0\big[\mathbb{E}_0(Y|A = a, W)\big]$ be the statistical estimand equal to the treatment-specific mean $\mathbb{E}^*(Y_a)$ under the identifiabilty assumptions.

7.1 Two-dimensional clever "covariate"

Suppose we are interested in contrasts of the treatment-specific means (under identifiability assumptions) ψ_1 and ψ_0 on the difference and ratio scale. We can use a two-dimensional clever covariate.

```
> # We already have Super Learner-based estimates of the
> # conditional mean outcome and the propensity score
>
> # calculate 2-dimensional clever covariate
> H.1W <- as.numeric(ObsData$A==1)/probA1.givenW
> H.0W <-  as.numeric(ObsData$A==0)/probA0.givenW
> # target
> logitUpdate<- glm(ObsData$Y~ -1 +offset(qlogis(expY.givenAW)) +
+                     H.0W + H.1W, family="binomial")
> eps<-logitUpdate$coef
> eps
```

```
       H.0W        H.1W
 0.06619084 -0.08691700
```

```
> # obtain the targeted estimates
> expY.givenAW.star <- plogis(qlogis(expY.givenAW) +
+                               eps['H.0W']*H.0W + eps['H.1W']*H.1W)
> expY.given1W.star <- plogis( qlogis(expY.given1W) +
+                                eps['H.1W']/probA1.givenW )
> expY.given0W.star <- plogis( qlogis(expY.given0W) +
+                                eps['H.0W']/probA0.givenW )
> TMLE2 <- data.frame(cbind(
+   psi1 = mean(expY.given1W.star),
+   psi0 = mean(expY.given0W.star),
+   diff = mean(expY.given1W.star) - mean(expY.given0W.star),
+   ratio = mean(expY.given1W.star) /mean(expY.given0W.star)
+   ))
> TMLE2
```

```
       psi1      psi0        diff     ratio
1 0.2416568 0.3222889 -0.08063215 0.7498141
```

```
> # comparison to before
> PsiHat.TMLE
```

```
[1] -0.07992069
```

7.2 Moving the clever "covariate" to the weight

Suppose our interest in the effect under the exposure $\mathbb{E}^*(Y_1)$; out statistical estimand would be Let $\psi_1 = \mathbb{E}_0\big[\mathbb{E}_0(Y|A=1,W)\big]$.

```
> # We already have Super Learner-based estimates of the
> # conditional mean outcome and the propensity score
>
> # we already have the clever covariate under the exposure of interest
> H.1W <- as.numeric(ObsData$A==1)/probA1.givenW
> # target
> logitUpdate<- glm(ObsData$Y~ offset(qlogis(expY.givenAW)),
+                     weights=H.1W, family="binomial")
> eps<-logitUpdate$coef
> eps
```

```
(Intercept)
 -0.2924544
```

```
> # obtain the targeted estimates
> expY.given1W.star <- plogis( qlogis(expY.given1W) + eps)
> mean(expY.given1W.star)
```

```
[1] 0.2422428
```

```
> # comparison with above
> TMLE2$psi1
```

```
[1] 0.2416568
```

Solution:

Appendix B: A specific data generating process

The following code was used to generate the data set `RLab5.TMLE.csv`. In this data generating process (one of many compatible with the SCM $\mathcal{M}^*$), *all background factors are independent.*

```
> library('MASS')
> #-------
> # generateData - function to generate the data
> # input: number of draws, whether or not there is a treatment effect
> # output: observed data + counterfactuals
> generateData<- function(n, effect=T){
+
+   W1 <- rbinom(n, size=1, prob=0.5)
+   W2<- runif(n, min=0, max=1)
+
+   # W3 and W4 are drawn from a multivariate normal (i.e., correlated)
+   s=1
+   Sigma<- matrix(0.85*s*s, nrow=2, ncol=2)
+   diag(Sigma)<- s^2
+   Z<- mvrnorm(n, rep(0,2), Sigma)
+   W3<- Z[,1]; W4<- Z[,2];
+
+   # generate the exposure mechanism P(A=1|W)
+   pscore<- plogis(-1.25 - .25*(W1+W2) +.5*W3*W4)
+
+   A<- rbinom(n, size=1, prob= pscore)
+
+   U.Y<- rnorm(n, 0, s)
+   # generate the counterfactual outcome with A=0
+   Y.0<- generateY(W1=W1, W2=W2, W3=W3, W4=W4, A=0, U.Y=U.Y)
+
+   if(!effect){ # if there is no effect, the counterfactual under txt =
+     # the counterfactual under the control
+     Y.1<- Y.0
+   }else{ # otherwise, generated the counterfactual outcome with A=1
+     Y.1<- generateY(W1=W1, W2=W2, W3=W3, W4=W4, A=1, U.Y=U.Y)
+   }
+
+   # assign the observed outcome based on the observed exposure
+   Y<- rep(NA, n)
+   Y[A==1]<- Y.1[A==1]
+   Y[A==0]<- Y.0[A==0]
+
+   data<- data.frame(W1, W2, W3, W4, A, Y, Y.1, Y.0)
+   data
+ }
> #---------
> # generateY: function to generate the outcome given the
> #   baseline covariates, exposure and background error U.Y
> #------------------
> generateY<- function(W1, W2, W3, W4, A, U.Y){
+   W1*plogis(0.25 +.5*W2 -1*W4 -0.5*A  -2*W4*W4 -.5*W4*A + .25*U.Y) +
+     (1-W1)*plogis(.25  -.5*W2 -1*W3 -0.5*A  -2*W3*W3 -.5*W3*A - .25*U.Y)
+ }


> #------------
> # Creation of the dataset
> #--------------
```

```
> set.seed(1)
> FullData<- generateData(n=250, effect= T)
> # remove unobservable counterfactuals
> ObsData<- subset(FullData, select=c(W1,W2,W3,W4, A, Y) )
> write.csv(ObsData, file="RLab5.TMLE.csv", row.names=F)
> #------------
```

We could obtain the true value of the causal parameter $\Psi^*(\mathbb{P}^*)$ by drawing a huge number of observations and taking the difference in the means of the counterfactual outcomes.

```
> set.seed(252)
> TrueData<- generateData(n=100000)
> # Treatment-specific means
> Psi.star.1 <- mean(TrueData$Y.1)
> Psi.star.0 <- mean( TrueData$Y.0)
> # difference scale
> Psi.star.diff <- Psi.star.1 - Psi.star.0
> # ratio scale
> Psi.star.ratio <- Psi.star.1 / Psi.star.0
> c(Psi.star.1, Psi.star.0, Psi.star.diff, Psi.star.ratio)

[1]  0.25689585  0.31924683 -0.06235098  0.80469352
```

The average treatment effect $\Psi^*(\mathbb{P}^*)$ is -6.2%. The expected injury severity would be 6.2% lower if all participants had prior Dinosaur experience than if none were experienced.

References

S. Gruber and M.J. van der Laan. tmle: An R package for targeted maximum likelihood estimation. *Journal of Statistical Software*, 51(13):1–35, 2012. doi: 10.18637/jss.v051.i13.

M.L. Petersen, K.E. Porter, S. Gruber, Y. Wang, and M.J. van der Laan. Diagnosing and responding to violations in the positivity assumption. *Statistical Methods in Medical Research*, 21(1):31–54, 2012. doi: 10.1177/0962280210386207.

E. Polley, E. LeDell, C. Kennedy, and M. van der Laan. *SuperLearner: Super Learner Prediction*, 2018. URL http://CRAN.R-project.org/package=SuperLearner. R package version 2.0-24.

J. Schwab, S. Lendle, M. Petersen, and M van der Laan. *ltmle: Longitudinal Targeted Maximum Likelihood Estimation*, 2017. URL http://CRAN.R-project.org/package=ltmle.