Day 4, Lecture 2

Longitudinal TMLE (LTMLE)

Longitudinal TMLE (LTMLE)

In this lecture, our goal is to:

1. Describe the implementation of the targeting algorithm for estimation of parameters defined under dynamic treatment interventions.
2. Explain the application of TMLE to longitudinal data structures to evaluate effects of static and dynamic effects in presence and without presence of right-censoring and with and without use of super learning.

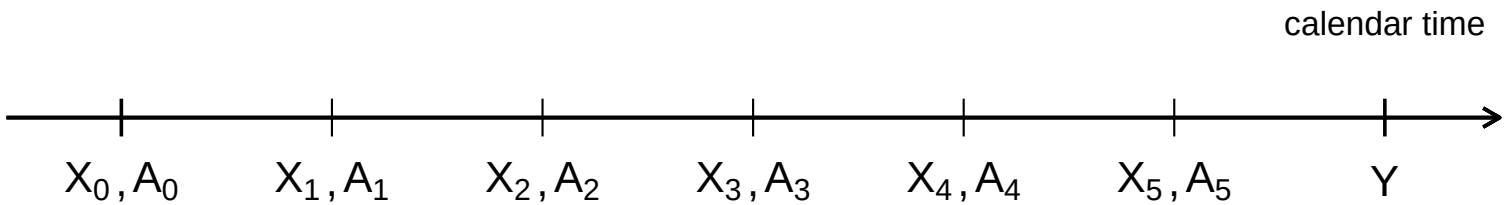
Overview

1. Targeting algorithm
    - Practical 2 (Task 1)
2. `ltmle` software package
    - Practical 2 (Task 2ff)
3. A note on right-censoring

Longitudinal data structure

Longitudinal data structure:

- $O = (X_0, A_0, X_1, A_1, \ldots, X_K, A_K, Y) \in (\mathbb{R}^d \times \{0,1\})^K \times \{0,1\}$
- Covariates $X = (X_0, X_1, \ldots, X_K)$ change over time
- Treatment decisions $A = (A_0, A_1, \ldots, A_K)$ are updated over time
- Covariates and treatment decisions interact in complex ways



NB: Still keeping right-censoring (and competing risks) out of the picture.

Longitudinal targeting

(For the representation in terms of iterated conditional expectations)

Targeting algorithm

Recall —

TMLE is a two-step procedure:

- Step 1 Construct initial estimator $\hat{P}_n$ for P.
- Step 2 Update the estimator $\hat{P}_n \mapsto \hat{P}_n^*$ such that $\hat{P}_n^*$ solves the efficient influence curve equation.

Step 1 = "initial estimation step"

Step 2 = "targeting step"

Targeting algorithm

Recall — for the ATE:

TMLE is a two-step procedure:

- Step 1 Construct initial estimators $\hat{f}_n, \hat{\pi}_n$ for f, π.
- Step 2 Update the estimator $\hat{f}_n \mapsto \hat{f}_n^*$ for f such that $\hat{f}_n^*$ for the fixed $\hat{\pi}_n$ solves the efficient influence curve equation.

Step 1 = "initial estimation step"

Step 2 = "targeting step"

Targeting algorithm

The relevant part of P needed to evaluate our target parameter:

$$\mathbb{E}_P[Y^{A_0=a_0^*,A_1=a_1^*,\ldots,A_K=a_K^*}] = \tilde{\Psi}(\bar{Q}),$$

with $\bar{Q} = (\bar{Q}_k)_{1\leq k\leq K+1}$.

Starting backwards from the last time-point:

$$\bar{Q}_{K+1}(\bar{x}_K, \bar{a}_K) = \mathbb{E}_P[Y \mid \bar{X}_K = \bar{x}_K, \bar{A}_K = \bar{a}_K]$$

and iteratively for $k = K, K-1, \ldots, 1$,

$$\bar{Q}_k(\bar{x}_{k-1}, \bar{a}_{k-1}) = \mathbb{E}_P[\bar{Q}_{k+1}(\bar{X}_k, a_k^*, \bar{A}_{k-1}) \mid \bar{X}_{k-1} = \bar{x}_{k-1}, \bar{A}_{k-1} = \bar{a}_{k-1}]$$

so that

$$\mathbb{E}_P[Y^{A_0=a_0^*,A_1=a_1^*,\ldots,A_K=a_K^*}] = \mathbb{E}_P[\bar{Q}_1(X_0, a_0^*)] = \tilde{\Psi}(\bar{Q}).$$

Targeting algorithm

We need the efficient influence function:

- Tells us what we need to estimate (to construct TMLE)
- Guides the construction of the targeting step

Construction of the targeting step for a given target parameter $\Psi : \mathcal{M} \to \mathbb{R}$ with efficient influence function $\phi^*(P)$ involves:

(i) A parametric submodel $\{\bar{Q}_\varepsilon : \varepsilon \in \mathbb{R}\} \subset \mathcal{M}$

(ii) A loss function $(O, \bar{Q}) \mapsto \mathscr{L}(\bar{Q})(O)$

such that: (1) $\bar{Q}_{\varepsilon=0} = \bar{Q}$, and, (2) $\left.\frac{d}{d\varepsilon}\right|_{\varepsilon=0} \mathscr{L}(\bar{Q}_\varepsilon)(O) = \phi^*(P)(O)$

Targeting algorithm

The efficient influence function is given by:

$$\phi^*(P)(O) = \tilde{\phi}^*(\bar{Q}, \pi)(O)$$

$$= \sum_{k=1}^{K+1} \left(\prod_{l=0}^{k-1} \frac{1\{A_l = a_l^*\}}{\pi_{A_l}(a_l^* \mid \bar{x}_l, \bar{a}_{l-1}^*)} \right) \left\{ \bar{Q}_{k+1}(\bar{X}_k, a_k^*, \bar{A}_{k-1}) - \bar{Q}_k(\bar{X}_{k-1}, \bar{A}_{k-1}) \right\}$$

$$+ \bar{Q}_1(X_0, a_0^*) - \Psi(P)$$

(with $\bar{Q}_{K+2} := Y$)

Targeting algorithm

The efficient influence function is given by:

$$\phi^*(P)(O) = \tilde{\phi}^*(\bar{Q}, \pi)(O)$$

$$= \sum_{k=1}^{K+1} \left(\prod_{l=0}^{k-1} \frac{1\{A_l = a_l^*\}}{\pi_{A_l}(a_l^* \mid \bar{x}_l, \bar{a}_{l-1}^*)} \right) \{ \bar{Q}_{k+1}(\bar{X}_k, a_k^*, \bar{A}_{k-1}) - \bar{Q}_k(\bar{X}_{k-1}, \bar{A}_{k-1}) \}$$

$$+ \bar{Q}_1(X_0, a_0^*) - \Psi(P)$$

(with $\bar{Q}_{K+2} := Y$)

- Need initial estimators for: $\pi = (\pi_{A_k})_{0 \le k \le K}$, $\bar{Q} = (\bar{Q}_k)_{1 \le k \le K+1}$

Targeting algorithm

The efficient influence function is given by:

$$\phi^*(P)(O) = \tilde{\phi}^*(\bar{Q}, \pi)(O)$$

$$= \sum_{k=1}^{K+1} \left(\prod_{l=0}^{k-1} \frac{1\{A_l = a_l^*\}}{\pi_{A_l}(a_l^* \mid \bar{x}_l, \bar{a}_{l-1}^*)} \right) \left\{ \bar{Q}_{k+1}(\bar{X}_k, a_k^*, \bar{A}_{k-1}) - \bar{Q}_k(\bar{X}_{k-1}, \bar{A}_{k-1}) \right\}$$

$$+ \bar{Q}_1(X_0, a_0^*) - \Psi(P)$$

(with $\bar{Q}_{K+2} := Y$)

- Need initial estimators for: $\pi = (\pi_{A_k})_{0 \le k \le K}$, $\bar{Q} = (\bar{Q}_k)_{1 \le k \le K+1}$
- Submodel and loss function for each $\bar{Q}_k$ in turn *to solve the k-specific part of the efficient influence curve equation*,

$$\tilde{\phi}_k^*(\bar{Q}, \pi)(O) = \left(\prod_{l=0}^{k-1} \frac{1\{A_l = a_l^*\}}{\pi_{A_l}(a_l^* \mid \bar{X}_l, \bar{a}_{l-1}^*)} \right) \left\{ \bar{Q}_{k+1}(\bar{X}_k, a_k^*, \bar{A}_{k-1}) - \bar{Q}_k(\bar{X}_{k-1}, \bar{A}_{k-1}) \right\}$$

Targeting algorithm

We construct a submodel $\bar{Q}_{k,\varepsilon}$ through a given $\bar{Q}_k$ and a loss function $(O, \bar{Q}_k) \mapsto \mathscr{L}_{\bar{Q}_{k+\mathbf{1}}}(\bar{Q}_k)(O)$ such that

$$(1) \quad \bar{Q}_{k,\varepsilon=0} = \bar{Q}_k, \quad \text{and,} \quad (2) \quad \frac{d}{d\varepsilon}\bigg|_{\varepsilon=0} \mathscr{L}_{\bar{Q}_{k+\mathbf{1}}}(\bar{Q}_{k,\varepsilon})(O) = \phi_k^*(\bar{Q}, \pi)(O)$$

Note that the loss function is indexed by $\bar{Q}_{k+1}$.

Targeting algorithm

We construct a submodel $\bar{Q}_{k,\varepsilon}$ through a given $\bar{Q}_k$ and a loss function $(O, \bar{Q}_k) \mapsto \mathscr{L}_{\bar{Q}_{k+\mathbf{1}}}(\bar{Q}_k)(O)$ such that

$$(1) \quad \bar{Q}_{k,\varepsilon=0} = \bar{Q}_k, \quad \text{and,} \quad (2) \quad \frac{d}{d\varepsilon}\bigg|_{\varepsilon=0} \mathscr{L}_{\bar{Q}_{k+\mathbf{1}}}(\bar{Q}_{k,\varepsilon})(O) = \phi_k^*(\bar{Q}, \pi)(O)$$

Note that the loss function is indexed by $\bar{Q}_{k+1}$.

We define:

$$\mathscr{L}_{\bar{Q}_{k+\mathbf{1}}}(\bar{Q}_k) = -\Big(\bar{Q}_{k+1}\log(\bar{Q}_k) + (1 - \bar{Q}_{k+1})\log(1 - \bar{Q}_k)\Big)$$

and,

$$\bar{Q}_{k,\varepsilon}(O) = \operatorname{expit}\Big(\operatorname{logit}(\bar{Q}_k(\bar{X}_{k-1}, \bar{A}_{k-1})) + \varepsilon H_k(\pi)(O)\Big)$$

with the "clever covariate": $H_k(\pi)(O) := \prod_{l=0}^{k-1} \frac{1\{A_l = a_l^*\}}{\pi_{A_l}(a_l^* \mid \bar{x}_l, \bar{a}_{l-1}^*)}$

Targeting algorithm

Another valid choice of loss function and submodel would be:

We define:

$$\mathscr{L}_{\bar{Q}_{k+1}}(\bar{Q}_k) = -H_k(\pi)(O)\Big(\bar{Q}_{k+1}\log(\bar{Q}_k) + (1-\bar{Q}_{k+1})\log(1-\bar{Q}_k)\Big)$$

and,

$$\bar{Q}_{k,\varepsilon}(O) = \text{expit}\Big(\text{logit}(\bar{Q}_k(\bar{X}_{k-1}, \bar{A}_{k-1})) + \varepsilon\Big)$$

using the "clever covariate" $H_k(\pi)(O) := \prod_{l=0}^{k-1} \frac{1\{A_l = a_l^*\}}{\pi_{A_l}(a_l^* \mid \bar{x}_l, \bar{a}_{l-1}^*)}$

as a weight.

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

- We start by targeting an estimator $\hat{\bar{Q}}_{K+1}$

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

- We start by targeting an estimator $\hat{\bar{Q}}_{K+1}$
- We set $\hat{\bar{Q}}_{K+2} := Y$

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

- We start by targeting an estimator $\hat{\bar{Q}}_{K+1}$
- We set $\hat{\bar{Q}}_{K+2} := Y$
- Regress Y on $\bar{X}_K, \bar{A}_K$ and obtain the predictions $\hat{\bar{Q}}_{K+1}(\bar{X}_k, \bar{A}_k)$

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

- We start by targeting an estimator $\hat{\bar{Q}}_{K+1}$
- We set $\hat{\bar{Q}}_{K+2} := Y$
- Regress Y on $\bar{X}_K, \bar{A}_K$ and obtain the predictions $\hat{\bar{Q}}_{K+1}(\bar{X}_k, \bar{A}_k)$
- Logistic regression of $\hat{\bar{Q}}_{K+2}$ on the clever covariate $H_{K+1}(\hat{\pi})(O)$ and offset $\text{logit}\{\hat{\bar{Q}}_{K+1}(\bar{X}_K, \bar{A}_K)\}$ to obtain estimate $\hat{\varepsilon}_n$ for ε

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

- We start by targeting an estimator $\hat{\bar{Q}}_{K+1}$
- We set $\hat{\bar{Q}}_{K+2} := Y$
- Regress Y on $\bar{X}_K, \bar{A}_K$ and obtain the predictions $\hat{\bar{Q}}_{K+1}(\bar{X}_k, \bar{A}_k)$
- Logistic regression of $\hat{\bar{Q}}_{K+2}$ on the clever covariate $H_{K+1}(\hat{\pi})(O)$ and offset $\text{logit}\{\hat{\bar{Q}}_{K+1}(\bar{X}_K, \bar{A}_K)\}$ to obtain estimate $\hat{\varepsilon}_n$ for ε
- Update according to the submodel: $\hat{\bar{Q}}^*_{K+1}(\bar{X}_k, a^*_k, \bar{A}_{k-1}) := \text{expit}\Big(\text{logit}\{\hat{\bar{Q}}_{K+1}(\bar{X}_k, a^*_k, \bar{A}_{k-1})\} + \hat{\varepsilon}_n H_{K+1}(\hat{\pi})(O)\Big)$

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

- We start by targeting an estimator $\hat{\bar{Q}}_{K+1}$
- We set $\hat{\bar{Q}}_{K+2} := Y$
- Regress Y on $\bar{X}_K, \bar{A}_K$ and obtain the predictions $\hat{\bar{Q}}_{K+1}(\bar{X}_k, \bar{A}_k)$
- Logistic regression of $\hat{\bar{Q}}_{K+2}$ on the clever covariate $H_{K+1}(\hat{\pi})(O)$ and offset $\text{logit}\{\hat{\bar{Q}}_{K+1}(\bar{X}_K, \bar{A}_K)\}$ to obtain estimate $\hat{\varepsilon}_n$ for ε
- Update according to the submodel: $\hat{\bar{Q}}^*_{K+1}(\bar{X}_k, a^*_k, \bar{A}_{k-1}) := \text{expit}\Big(\text{logit}\{\hat{\bar{Q}}_{K+1}(\bar{X}_k, a^*_k, \bar{A}_{k-1})\} + \hat{\varepsilon}_n H_{K+1}(\hat{\pi})(O)\Big)$

We then solve:

$$\frac{1}{n}\sum_{i=1}^{n} H_{K+1}(\hat{\pi})(O_i)\Big\{\hat{\bar{Q}}^*_{K+2} - \hat{\bar{Q}}^*_{K+1}(\bar{X}_{K,i}, \bar{A}_{K,i})\Big\} = 0$$

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

- We start by targeting an estimator $\hat{\bar{Q}}_{K+1}$
- We set $\hat{\bar{Q}}_{K+2} := Y$
- Regress Y on $\bar{X}_K, \bar{A}_K$ and obtain the predictions $\hat{\bar{Q}}_{K+1}(\bar{X}_k, \bar{A}_k)$
- Logistic regression of $\hat{\bar{Q}}_{K+2}$ on the clever covariate $H_{K+1}(\hat{\pi})(O)$ and offset $\text{logit}\{\hat{\bar{Q}}_{K+1}(\bar{X}_K, \bar{A}_K)\}$ to obtain estimate $\hat{\varepsilon}_n$ for ε
- Update according to the submodel: $\hat{\bar{Q}}^*_{K+1}(\bar{X}_k, a^*_k, \bar{A}_{k-1}) := \text{expit}\Big(\text{logit}\{\hat{\bar{Q}}_{K+1}(\bar{X}_k, a^*_k, \bar{A}_{k-1})\} + \hat{\varepsilon}_n H_{K+1}(\hat{\pi})(O)\Big)$

We then solve:

$$\frac{1}{n}\sum_{i=1}^{n} H_{K+1}(\hat{\pi})(O_i)\Big\{\hat{\bar{Q}}^*_{K+2} - \hat{\bar{Q}}^*_{K+1}(\bar{X}_{K,i}, \bar{A}_{K,i})\Big\} = 0$$

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

- Proceed iteratively from $k+1$ to k

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

- Proceed iteratively from $k+1$ to k
- From the $(k+1)th$ step, we have targeted estimator $\hat{\bar{Q}}^*_{k+1}(\bar{X}_k, \bar{A}_k)$

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

- Proceed iteratively from $k+1$ to k
- From the $(k+1)th$ step, we have targeted estimator $\hat{\bar{Q}}^*_{k+1}(\bar{X}_k, \bar{A}_k)$
- Regress $\hat{\bar{Q}}_{k+1}(\bar{X}_k, a^*_k, \bar{A}_{k-1})$ on $\bar{X}_{k-1}, \bar{A}_{k-1}$ and denote the prediction by $\hat{\bar{Q}}_k(\bar{X}_{k-1}, \bar{A}_{k-1})$

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

- Proceed iteratively from $k+1$ to k
- From the $(k+1)th$ step, we have targeted estimator $\hat{\bar{Q}}^*_{k+1}(\bar{X}_k, \bar{A}_k)$
- Regress $\hat{\bar{Q}}_{k+1}(\bar{X}_k, a^*_k, \bar{A}_{k-1})$ on $\bar{X}_{k-1}, \bar{A}_{k-1}$ and denote the prediction by $\hat{\bar{Q}}_k(\bar{X}_{k-1}, \bar{A}_{k-1})$
- Logistic regression of $\hat{\bar{Q}}_{k+1}(\bar{X}_k, a^*_k, \bar{A}_{k-1})$ on the clever covariate $H_k(\hat{\pi})(O)$ and offset $\text{logit}(\hat{\bar{Q}}_k(\bar{X}_{k-1}, \bar{A}_{k-1}))$ to obtain estimate $\hat{\varepsilon}_n$ for ε

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

- Proceed iteratively from $k+1$ to k
- From the $(k+1)th$ step, we have targeted estimator $\hat{\bar{Q}}^*_{k+1}(\bar{X}_k, \bar{A}_k)$
- Regress $\hat{\bar{Q}}_{k+1}(\bar{X}_k, a^*_k, \bar{A}_{k-1})$ on $\bar{X}_{k-1}, \bar{A}_{k-1}$ and denote the prediction by $\hat{\bar{Q}}_k(\bar{X}_{k-1}, \bar{A}_{k-1})$
- Logistic regression of $\hat{\bar{Q}}_{k+1}(\bar{X}_k, a^*_k, \bar{A}_{k-1})$ on the clever covariate $H_k(\hat{\pi})(O)$ and offset $\text{logit}(\hat{\bar{Q}}_k(\bar{X}_{k-1}, \bar{A}_{k-1}))$ to obtain estimate $\hat{\varepsilon}_n$ for ε
- Update according to the submodel:
$\hat{\bar{Q}}^*_k := \text{expit}\big(\text{logit}(\hat{\bar{Q}}_k(\bar{X}_{k-1}, \bar{A}_{k-1})) + \hat{\varepsilon}_n H_k(\hat{\pi})(O)\big)$

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

- Proceed iteratively from $k+1$ to k
- From the $(k+1)th$ step, we have targeted estimator $\hat{\bar{Q}}^*_{k+1}(\bar{X}_k, \bar{A}_k)$
- Regress $\hat{\bar{Q}}_{k+1}(\bar{X}_k, a^*_k, \bar{A}_{k-1})$ on $\bar{X}_{k-1}, \bar{A}_{k-1}$ and denote the prediction by $\hat{\bar{Q}}_k(\bar{X}_{k-1}, \bar{A}_{k-1})$
- Logistic regression of $\hat{\bar{Q}}_{k+1}(\bar{X}_k, a^*_k, \bar{A}_{k-1})$ on the clever covariate $H_k(\hat{\pi})(O)$ and offset $\text{logit}(\hat{\bar{Q}}_k(\bar{X}_{k-1}, \bar{A}_{k-1}))$ to obtain estimate $\hat{\varepsilon}_n$ for ε
- Update according to the submodel:
  $\hat{\bar{Q}}^*_k := \text{expit}\big(\text{logit}(\hat{\bar{Q}}_k(\bar{X}_{k-1}, \bar{A}_{k-1})) + \hat{\varepsilon}_n H_k(\hat{\pi})(O)\big)$

We then solve:

$$\frac{1}{n}\sum_{i=1}^{n} H_k(\hat{\pi})(O_i)\big\{\hat{\bar{Q}}^*_{k+1}(\bar{X}_{k,i}, a^*_k, \bar{A}_{k-1,i}) - \hat{\bar{Q}}^*_k(\bar{X}_{k-1,i}, \bar{A}_{k-1,i})\big\} = 0$$

Targeting algorithm

The targeting step becomes a targeting algorithm that proceeds iteratively along the sequence of iterated conditional expectations, *starting from the last time-point*.

This procedure gives a sequence of updated estimators $\hat{\bar{Q}}^* = (\hat{\bar{Q}}^*_{K+1}, \hat{\bar{Q}}^*_K, \ldots, \hat{\bar{Q}}^*_1)$ that solves the efficient influence curve equation:

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K+1}\left(\prod_{l=0}^{k-1}\frac{1\{A_{l,i}=a^*_l\}}{\hat{\pi}_{A_l}(a^*_l \mid \bar{X}_{l,i}, \bar{a}^*_{l-1})}\right)\Big\{\hat{\bar{Q}}^*_{k+1}(\bar{X}_{k,i}, a^*_k, \bar{A}_{k-1,i}) - \hat{\bar{Q}}^*_k(\bar{X}_{k-1}, \bar{A}_{k-1})\Big\} + \hat{\bar{Q}}^*_1(X_0, a^*_0) - \Psi(\hat{\bar{Q}}^*)$$

where

$$\tilde{\Psi}(\hat{\bar{Q}}^*) = \frac{1}{n}\sum_{i=1}^{n}\hat{\bar{Q}}^*_1(X_{0,i}, a^*_0).$$

Practical 2: Implementing the targeting step (Task 1)

Read the first page of **day3_practical2.pdf** and then go through the steps of **Task 1**.

`ltmle` software package

- Specifying static and dynamic interventions
- Initial estimation and super learning
- LTMLE and `ltmle` for right-censored data structures

ltmle software package

See also: https://cran.r-project.org/web/packages/ltmle/ltmle.pdf

```
install.packages(ltmle)
library(ltmle)
```

ltmle software package

Some useful arguments to know:

```
ltmle(data,
      Anodes, Lnodes, Cnodes, Ynodes,
      abar, rule,
      Qform, gform,
      SL.library,
      gbounds,
      ...)
```

`ltmle` software package

- `data`: data frame *in wide format*

  NB the order of columns correspond to the order of variables.

- `Anodes`: names of treatment variables
- `Lnodes`: names of covariates
- `Cnodes`: names of censoring variables
- `Ynodes`: names of outcome variables

NB All variables except baseline covariates must be specified in `Anodes`, `Lnodes`, `Cnodes` or `Ynodes`.

ltmle software package

For the simulated dataset from the practicals:

```
          X0.1        X0.2 X0.3 A0 X1.1 X1.2 A1 Y
1:  1.2915853 -0.29726781    0  0    1    0  0 1
2:  0.8407983 -0.48032616    0  0    0    1  0 0
3:  1.8633391 -0.50152323    0  1    1    1  0 1
4: -1.6855569 -1.57948481    0  1    0    0  1 1
5: -1.7854517  0.04438914    0  0    1    1  0 1
6:  0.2999129 -1.41470763    0  1    0    1  0 0
```

```
Anodes = paste0("A",0:1)
Lnodes = c(paste0("X0.", 1:3), paste0("X1.", 1:2))
Ynodes = "Y"
```

The ordering of columns dictates the temporal ordering of variables:

$$\begin{Bmatrix} X_{0,1} \\ X_{0,2} \\ X_{0,3} \end{Bmatrix} \rightarrow A_0 \rightarrow \begin{Bmatrix} X_{1,1} \\ X_{1,2} \end{Bmatrix} \rightarrow A \rightarrow Y$$

`ltmle` software package

Specifying interventions:

- `abar`: a binary vector of treatment assignments of length = length(`Anodes`) **or** a list of two elements to contrast treatment regimes
  - to specify *static treatment regimes*
- `rule`: a function that can be applied to each row of the data to return a binary vector of treatment assignments of length = length(`Anodes`)
  - to specify *dynamic treatment regimes*

ltmle software package

For example:

```
summary(ltmle(sim.data,
              Anodes=paste0("A",0:1),
              Lnodes=c(paste0("X0.", 1:3), paste0("X1.", 1:2)),
              Ynodes="Y",
              abar=list(treatment=c(1,1), control=c(0,0))))
```

[output next slide]

ltmle software package

```
Treatment Estimate:
   Parameter Estimate:  0.3029
    Estimated Std Err:  0.086738
              p-value:  0.00047919
    95% Conf Interval: (0.1329, 0.4729)

Control Estimate:
   Parameter Estimate:  0.36688
    Estimated Std Err:  0.015419
              p-value:  <2e-16
    95% Conf Interval: (0.33666, 0.3971)

Additive Treatment Effect:
   Parameter Estimate:  -0.063978
    Estimated Std Err:  0.088095
              p-value:  0.46769
    95% Conf Interval: (-0.23664, 0.10868)
```

`ltmle` software package

Here note that `Treatment Estimate` and `Control Estimate` were fitted completely separately, and that they could had been obtained with separate calls:

ltmle software package

Here note that `Treatment Estimate` and `Control Estimate` were fitted completely separately, and that they could had been obtained with separate calls:

`Treatment Estimate`:

```
summary(ltmle(...,
              abar=c(1,1)))
```

```
Parameter Estimate:  0.3029
 Estimated Std Err:  0.086738
           p-value:  0.00047919
 95% Conf Interval: (0.1329, 0.4729)
```

`Control Estimate`:

```
summary(ltmle(...,
              abar=c(0,0)))
```

```
Parameter Estimate:  0.36688
 Estimated Std Err:  0.015419
           p-value:  <2e-16
 95% Conf Interval: (0.33666, 0.3971)
```

- For each parameter we get the standard error,[a] the p-value of the null hypothesis that that quantity equals zero, and confidence intervals

[a] influence curve based if the argument `variance.method="ic"` is specified, otherwise a more robust variance estimator based on TMLE is used.

ltmle software package

Effect of a dynamic regime:

```
summary(ltmle(sim.data,
          Anodes=paste0("A",0:1),
          Lnodes=c(paste0("X0.", 1:3), paste0("X1.", 1:2)),
          Ynodes="Y",
          rule=function(row) c(1, ifelse(row["X1.1"]==1, 0,
    1))))
```

ltmle software package

Effect of a dynamic regime:

```
summary(ltmle(sim.data,
          Anodes=paste0("A",0:1),
          Lnodes=c(paste0("X0.", 1:3), paste0("X1.", 1:2)),
          Ynodes="Y",
          rule=function(row) c(1, ifelse(row["X1.1"]==1, 0,
    1))))
```

Static regimes can also be specified as a dynamic regime:

```
summary(ltmle(sim.data,
          Anodes=paste0("A",0:1),
          Lnodes=c(paste0("X0.", 1:3), paste0("X1.", 1:2)),
          Ynodes="Y",
          rule=list(
          treatment=function(row) c(1,1),
          control=function(row) c(0,0))))
```

`ltmle` software package

- Specifying static and dynamic interventions
- Initial estimation and super learning
- LTMLE and `ltmle` for right-censored data structures

ltmle software package

- Qform: character vector of regression formulas for the outcome regressions
  - Qform indicates what variables are included in each outcome regression
  - default is NULL which means that all variables from previous time-points are included
  - (does *not* mean that GLM is used)
- gform: character vector of regression formulas for the propensity scores
  - gform indicates what variables are included in each propensity score regression
  - default is NULL which means that all variables from previous time-points are included
  - (does *not* mean that GLM is used)

ltmle software package

Default:

```
fit.ltmle <- ltmle(sim.data,
            Anodes=paste0("A",0:1),
            Lnodes=c(paste0("X0.", 1:3), paste0("X1.", 1:2)),
            Ynodes="Y",
            abar=list(treatment=c(1,1), control=c(0,0)))
```

```
Qform not specified, using defaults:
formula for X1.1:
Q.kplus1 ~ X0.1 + X0.2 + X0.3 + A0
formula for Y:
Q.kplus1 ~ X0.1 + X0.2 + X0.3 + A0 + X1.1 + X1.2 + A1

gform not specified, using defaults:
formula for A0:
A0 ~ X0.1 + X0.2 + X0.3
formula for A1:
A1 ~ X0.1 + X0.2 + X0.3 + A0 + X1.1 + X1.2

speedglm failed, using glm instead. If you see a lot of this message, and
```

ltmle software package

Extracting the first (last time-point) regression fit:

$$\bar{Q}_2(\bar{X}_1, \bar{A}_1) = \mathbb{E}_P[Y \mid \bar{X}_1, \bar{A}_1]$$

```
fit.ltmle$fit$Q[[1]]$Y
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-0.91411304	0.10514428	-8.6938923	7.179800e-18
X0.1	0.03225723	0.04248034	0.7593449	4.477361e-01
X0.2	-0.05878768	0.04851366	-1.2117759	2.257419e-01
X0.3	0.03744604	0.12002695	0.3119803	7.550882e-01
A0	-0.29864339	0.10343002	-2.8873958	3.926343e-03
X1.1	1.16277977	0.10398452	11.1822394	3.378290e-28
X1.2	0.08077220	0.10204795	0.7915123	4.287394e-01
A1	-0.70956699	0.26311107	-2.6968345	7.059175e-03

ltmle software package

Extracting the next/last (first time-point) regression fit:

$$\bar{Q}_1(X_0, A_0) = \mathbb{E}_P[\bar{Q}_2(\bar{X}_1, \bar{A}_1) \mid X_0, A_0]$$

```
fit.ltmle$fit$Q[[1]]$X1.1
```

```
                Estimate Std. Error    t value             Pr(>|t|)
(Intercept) -1.45814424 0.01937218 -75.270032 0.0000000000000000
X0.1         0.04941305 0.01106743   4.464726 0.0000846490464004
X0.2        -0.04858049 0.01266330  -3.836320 0.0001287731282715 4
X0.3         0.11706316 0.03047067   3.841830 0.0001259363419614 6
A0           0.17059618 0.02525066   6.756106 0.0000000001853852
```

(If you ask me, the naming convention is quite strange).

ltmle software package

Extracting the propensity score regression fit:

$$\pi_{A_1}(1 \mid \bar{X}_1, A_0) = \mathbb{E}_P[A_1 \mid \bar{X}_1, A_0]$$

```
fit.ltmle$fit$g[[1]]$A1
```

```
                Estimate Std. Error     t value     Pr(>|t|)
(Intercept) -4.16188796 0.33230949 -12.5241321 1.074919e-34
X0.1         0.15421078 0.08538514   1.8060610 7.105961e-02
X0.2         0.09551047 0.10273564   0.9296722 3.526534e-01
X0.3         0.36500100 0.23542597   1.5503854 1.212078e-01
A0           4.84411485 0.34081015  14.2135287 9.610307e-44
X1.1        -5.64663180 0.51985544 -10.8619269 9.664175e-27
X1.2        -5.21307450 0.42763985 -12.1903383 5.081408e-33
```

ltmle software package

Extracting the propensity score regression fit:

$$\pi_{A_0}(1 \mid X_0) = \mathbb{E}_P[A_0 \mid X_0]$$

```
fit.ltmle$fit$g[[1]]$A0
```

```
                Estimate Std. Error    t value  Pr(>|t|)
(Intercept) -0.03114875 0.05043403 -0.6176137 0.5369005
X0.1         0.01069894 0.03928253  0.2723587 0.7853744
X0.2         0.01483267 0.04490924  0.3302810 0.7412223
X0.3        -0.04204218 0.11017986 -0.3815777 0.7028152
```

`ltmle` software package

- There are many opportunities for misspecification!!
- Double robustness helps a bit for consistency
- Inference still requires getting all of it right

`ltmle` software package

- `SL.library`: list with entries named `Q` and `g` specifying super learner libraries to pass to `SuperLearner` for the outcome regressions and the propensities scores

`ltmle` software package

- `SL.library`: list with entries named `Q` and `g` specifying super learner libraries to pass to `SuperLearner` for the outcome regressions and the propensities scores

You can see available models for the super learner here:
`https://cran.r-project.org/web/packages/SuperLearner/vignettes/Guide-to-SuperLearner.html` (Section 4)

NB Some algorithms are really very slow

NB Some algorithms may not converge (gives error messages)

NB Think about what you know about each particular algorithm and do not just blindly include a ton of heavy algorithms

You can also write your own algorithms.

ltmle software package

For example, one could specify:

```
sl.lib.Q <- c("SL.glm", "SL.mean", "SL.glm.interaction",
              "SL.glmnet", "SL.gam")
sl.lib.g <- c("SL.glm", "SL.mean", "SL.glmnet", "SL.gam")
```

And then call ltmle:

```
fit.ltmle.sl <- ltmle(dt,
                  Anodes=paste0("A",0:1),
                  Lnodes=c(paste0("X0.", 1:3),
                       paste0("X1.", 1:2)),
                  Ynodes="Y",
                  SL.library=list(Q=sl.lib.Q, g=sl.lib.g),
                  abar=list(treatment=c(1,1), control=c(0,0)))
```

ltmle software package

We can extract the super learner weights applied to each algorithm from the ltmle object:

```
fit.ltmle.sl$fit$Q[[1]]
```

```
$X1.1
                              Risk       Coef
SL.glm_All                0.01616588 0.97450398
SL.mean_All               0.01708647 0.02549602
SL.glm.interaction_All    0.01624704 0.00000000
SL.glmnet_All                     NA 0.00000000
SL.gam_All                0.01617732 0.00000000

$Y
                             Risk       Coef
SL.glm_All                0.2091238 0.74283811
SL.mean_All               0.2342586 0.02679133
SL.glm.interaction_All    0.2115791 0.00000000
SL.glmnet_All             0.2094033 0.00000000
SL.gam_All                0.2091607 0.23037056
```

ltmle software package

We can extract the super learner weights applied to each algorithm from the ltmle object:

```
fit.ltmle.sl$fit$g[[1]]
```

```
$A0
                   Risk Coef
SL.glm_All    0.2508915    0
SL.mean_All   0.2502013    1
SL.glmnet_All 0.2502013    0
SL.gam_All    0.2512009    0

$A1
                    Risk      Coef
SL.glm_All    0.03332285 0.000000
SL.mean_All   0.08031019 0.000000
SL.glmnet_All 0.03330272 0.822445
SL.gam_All    0.03337456 0.177555
```

`ltmle` software package

- Guidelines on what algorithms to use??[1]
- Trade-off between computation time and performance??
- Studying performance on simulated data can be useful.
  - (but this is not trivial either).

[1] Phillips, R. V., van der Laan, M. J., Lee, H., & Gruber, S. (2023). Practical considerations for specifying a super learner. International Journal of Epidemiology.

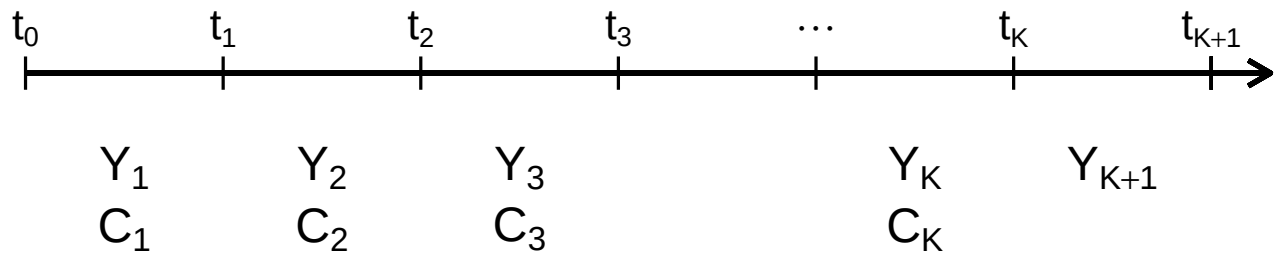
`ltmle` software package

- Specifying static and dynamic interventions
- Initial estimation and super learning
- LTMLE and `ltmle` for right-censored data structures

LTMLE for right-censored data

Outcome process Y_k can jump from 0 to 1 at any time-point t_k.

Censoring process C_k can jump from 0 to 1 at any time-point t_k.



- Time to event: $\tilde{T} = k' \wedge K + 1$ where $k' = \min(k : Y_k = 1 \text{ or } C_k = 1)$
- Event indicator: $\Delta = 1\{Y_{\tilde{T}} = 1\}$

LTMLE for right-censored data

One way that we can represent the survival data over a grid of $K = 7$ time-points (long format), e.g., for three different individuals:

```
   id k Y C
1:  2 1 0 0
2:  2 2 0 0
3:  2 3 0 0
4:  2 4 1 0
```

```
   id k Y C
1:  5 1 0 0
2:  5 2 0 0
3:  5 3 0 0
4:  5 4 0 0
5:  5 5 0 0
6:  5 6 0 0
7:  5 7 0 0
8:  5 8 0 0
```

```
   id k Y C
1:  7 1 0 0
2:  7 2 0 1
```

LTMLE for right-censored data

One way that we can represent the survival data over a grid of $K = 7$ time-points (long format), e.g., for three different individuals:

	id	k	Y	C
1:	2	1	0	0
2:	2	2	0	0
3:	2	3	0	0
4:	2	4	1	0

	id	k	Y	C
1:	5	1	0	0
2:	5	2	0	0
3:	5	3	0	0
4:	5	4	0	0
5:	5	5	0	0
6:	5	6	0	0
7:	5	7	0	0
8:	5	8	0	0

	id	k	Y	C
1:	7	1	0	0
2:	7	2	0	1

Same data may also be presented as (wide format):

	id	Y1	C1	Y2	C2	Y3	C3	Y4	C4	Y5	C5	Y6	C6	Y7	C7	Y8
1:	2	0	0	0	0	0	0	1	0	1	0	1	0	1	0	1
2:	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3:	7	0	0	0	1	0	1	0	1	0	1	0	1	0	1	0

LTMLE for right-censored data

One way that we can represent the survival data over a grid of $K = 7$ time-points (long format), e.g., for three different individuals:

```
   id k Y C
1:  2 1 0 0
2:  2 2 0 0
3:  2 3 0 0
4:  2 4 1 0
```

```
   id k Y C
1:  5 1 0 0
2:  5 2 0 0
3:  5 3 0 0
4:  5 4 0 0
5:  5 5 0 0
6:  5 6 0 0
7:  5 7 0 0
8:  5 8 0 0
```

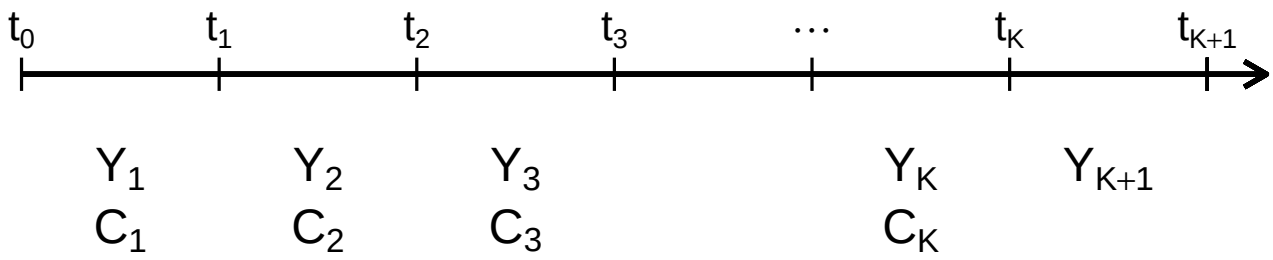
```
   id k Y C
1:  7 1 0 0
2:  7 2 0 1
```

Same data may also be presented as (wide format):

```
   id Y1 C1 Y2 C2 Y3 C3 Y4 C4 Y5 C5 Y6 C6 Y7 C7 Y8
1:  2  0  0  0  0  0  0  1  0  1  0  1  0  1  0  1
2:  5  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
3:  7  0  0  0  1  0  1  0  1  0  1  0  1  0  1  0
```

(The wide format is needed for `ltmle`).

LTMLE for right-censored data



Example:

	id	X0.1	X0.2	X0.3	A0	Y1	C1	Y2	C2	Y3	C3	Y4	C4	Y5	C5	Y6	C6	Y7	C7	Y8
1:	1	0.408	-0.196	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
2:	2	-1.220	0.595	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
3:	3	1.866	-1.609	0	1	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1
4:	4	0.604	0.041	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5:	5	-0.532	-1.251	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6:	6	1.955	0.133	1	1	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0

- Baseline covariate vector: $X_0 = (X_{0,1}, X_{0,2}, X_{0,3})$
- Baseline treatment decision: $A_0 \in \{0, 1\}$
- Time to event: $\tilde{T} = k' \wedge K + 1$ where $k' = \min(k : Y_k = 1 \text{ or } C_k = 1)$
- Event indicator: $\Delta = 1\{Y_{\tilde{T}} = 1\}$

LTMLE for right-censored data

Generally:

$$O = (X_0, A_0, Y_1, C_1, X_1, A_1, \ldots, Y_K, C_K, X_K, A_K, Y = X_{K+1}).$$

- Covariates $X = (X_0, X_1, \ldots, X_K)$ change over time.
- Treatment decisions $A = (A_0, A_1, \ldots, A_K)$ are updated over time.
- Censoring status $C = (C_1, C_2, \ldots, C_K)$ change over time.
- Outcome (death) status $Y = (Y_1, Y_2, \ldots, Y_K)$ change over time.

[After death/censoring, all variables are deterministically set to their last observed values].

LTMLE for right-censored data

Inferring on the uncensored event time is like imposing a simple static intervention on all right-censoring nodes to impose 'no censoring'.

Target causal parameter:

$$\Psi(P) = \mathbb{E}[\, Y_K^{A_0=a_0^*, C_1=0, A_1=a_k^*, \ldots, C_K=0, A_K=a_K^*}] - \mathbb{E}[\, Y_K^{A_0=0, C_1=0, A_0=0, \ldots, C_K=0, A_K=0}]$$

I.e., the effect of the treatment *had there been no loss to follow-up*.

= the absolute risk by time t_{k^*} if everyone had received treatment $(a_0^*, a_1^*, \ldots, a_K^*)$ contrasted to the absolute risk if everyone had not been treated.

LTMLE for right-censored data

The ordering of

$$O = (X_0, A_0, Y_1, C_1, X_1, A_1, \ldots, Y_K, C_K, X_K, A_K, Y = X_{K+1}),$$

implies a temporal ordering:

$$X_0 \to A_0 \to \cdots \to Y_k \to C_k \to X_k \to A_k \to \cdots \to Y_{K+1}.$$

- at each time-point t_k, a patient is only at risk of dying if they did not yet die and they were not yet right-censored $(Y_{k-1} = 0, C_{k-1} = 0)$.
- at each time-point t_k, a patient is only at risk of being right-censoring if they did not yet die at this time and they were not yet right-censored $(Y_k = 0, C_{k-1} = 0)$.

LTMLE for right-censored data

Factorization of the density p of $P \in \mathcal{M}$:

$$p(o) = \mu_{X_0}(x_0)\pi_{A_0}(a \mid x_0)$$
$$\prod_{k=1}^{K} \Big(\mu_{Y_k}(y_k \mid \bar{y}_{k-1}, \bar{c}_{k-1}, \bar{x}_{k-1}, \bar{a}_{k-1}) \pi_{C_k}(c_k \mid \bar{y}_k, \bar{c}_{k-1}, \bar{x}_k, \bar{a}_{k-1})$$
$$\times \mu_{X_k}(x_k \mid \bar{y}_k, \bar{c}_k, \bar{x}_{k-1}, \bar{a}_{k-1}) \pi_{A_k}(a_k \mid \bar{y}_k, \bar{c}_k, \bar{x}_k, \bar{a}_{k-1}) \Big)$$
$$\times \mu_{Y_{K+1}}(y_{K+1} \mid \bar{y}_K, \bar{c}_K, \bar{x}_K, \bar{a}_K)$$

Without going into too many details, note for example that:

- $\mu_{Y_k}(1 \mid \bar{0}_{k-1}, \bar{0}_{k-1}, \bar{x}_{k-1}, \bar{a}_{k-1})$ is the risk of outcome (dying) for a subject who did not yet die nor were right-censored plus had covariate and treatment history equal to $\bar{x}_{k-1}, \bar{a}_{k-1}$.
- $\pi_{C_k}(1 \mid \bar{0}_k, \bar{0}_{k-1}, \bar{x}_{k-1}, \bar{a}_{k-1})$ is the probability of being right-censored for a subject who did not yet die nor were right-censored plus had covariate and treatment history equal to $\bar{x}_{k-1}, \bar{a}_{k-1}$.

LTMLE for right-censored data

Identification of the target parameter as without censoring:

$$\bar{Q}_{K+1}(\bar{x}_K, \bar{a}_K, \bar{y}_K) = \mathbb{E}_P[Y_{K+1} \mid \bar{Y}_K = \bar{y}_K, \bar{C}_K = 0, \bar{X}_K = \bar{x}_K, \bar{A}_K = \bar{a}_K]$$

$$\bar{Q}_K(\bar{x}_{K-1}, \bar{a}_{K-1}, \bar{y}_{K-1}) = \mathbb{E}_P[Q_{K+1}(\bar{X}_K, a_K^*, \bar{A}_{K-1}, \bar{Y}_{K-1}) \mid \bar{Y}_{K-1} = \bar{y}_{K-1}, \bar{C}_{K-1} = 0, \bar{X}$$

$$\vdots$$

$$\bar{Q}_k(\bar{x}_{k-1}, \bar{a}_{k-1}, \bar{y}_{k-1}) = \mathbb{E}_P[Q_{k+1}(\bar{X}_k, a_k^*, \bar{A}_{k-1}, \bar{Y}_{k-1}) \mid \bar{Y}_{k-1} = \bar{y}_{k-1}, \bar{C}_{k-1} = 0, \bar{X}_{k-1}$$

$$\vdots$$

$$\bar{Q}_2(\bar{x}_1, \bar{a}_1, \bar{y}_1) = \mathbb{E}_P[Q_3(\bar{X}_2, a_2^*, A_1, Y_1) \mid \bar{Y}_1 = y_1, \bar{C}_1 = 0, \bar{X}_1 = \bar{x}_1, A_1 = a_1]$$

$$\bar{Q}_1(x_0, a_0) = \mathbb{E}_P[Q_2(\bar{X}_1, a_1^*) \mid X_0 = x_0, A_0 = a_0]$$

+ note that $\bar{Q}_k(\bar{x}_{k-1}, \bar{a}_{k-1}, \bar{y}_{k-1}) = 1$ if $y_{k'-1} = 1$ for $k' \leq k$.

LTMLE for right-censored data

The efficient influence function is given by:

$$\tilde{\phi}^*(\bar{Q}, \pi)(O)$$

$$= \sum_{k=1}^{K+1} 1\{Y_k = 0\} \left(\prod_{l=0}^{k-1} \frac{1\{A_l = a_l^*\} 1\{C_l = 0\}}{\pi_{A_l}(a_l^* \mid \bar{0}_l, \bar{0}_l, \bar{x}_l, \bar{a}_{l-1}^*) \pi_{C_l}(0 \mid \bar{0}_{l-1}, \bar{0}_{l-1}, \bar{x}_{l-1}, \bar{a}_{l-1}^*)} \right)$$

$$\times \left\{ \bar{Q}_{k+1}(\bar{X}_k, a_k^*, \bar{A}_{k-1}, 0) - \bar{Q}_k(\bar{X}_{k-1}, \bar{A}_{k-1}, 0) \right\} + \bar{Q}_0(X_0) - \Psi(P).$$

ltmle for right-censored data

```
ltmle(data,
      Anodes, Lnodes, Cnodes, Ynodes,
      abar, rule,
      Qform, gform,
      SL.library,
      gbounds,
      survivalOutcome=TRUE,
      ...)
```

- Right-censoring nodes are specified in the `Cnodes` argument.
- The formatting of `Cnodes` is a bit peculiar — it should be a factor variable with the values `0` and `1` and the labels `"uncensored"` and `"censored"`.
- Note that we further specify `survivalOutcome=TRUE`, so that `Ynodes` are treated as indicators of a terminating event.

ltmle for right-censored data

- Qform: character vector of regression formulas for the outcome regressions
  - Qform indicates what variables are included in each outcome regression
  - default is NULL which means that all variables from previous time-points are included
  - (does *not* mean that GLM is used)
- gform: character vector of regression formulas for the propensity scores and the hazards of censoring
  - gform indicates what variables are included in each propensity score regression and the hazards of censoring
  - default is NULL which means that all variables from previous time-points are included
  - (does *not* mean that GLM is used)

ltmle for right-censored data

For the simulated example:

	id	X0.1	X0.2	X0.3	A0	Y1	C1	Y2	C2	Y3	C3	Y4	C4	Y5	C5	Y6	C6	Y7	C7	Y8
1:	1	0.408	-0.196	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
2:	2	-1.220	0.595	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
3:	3	1.866	-1.609	0	1	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1
4:	4	0.604	0.041	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5:	5	-0.532	-1.251	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6:	6	1.955	0.133	1	1	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0

```
for (k in 1:7)
    sim.data[, (paste0("C", k)):=BinaryToCensoring(is.censored=get(
    paste0("C", k)))]
```

```
ltmle.fit <- ltmle(sim.data[, -"id"],
            Anodes="A0", Lnodes=paste0("X0.", 1:3),
            Cnodes=paste0("C", 1:7), Ynodes=paste0("Y", 1:8),
            abar=list(treatment=1, control=0),
            SL.library=list(Q=c("SL.glm", "SL.mean"),
                    g=c("SL.glm", "SL.mean")),
            survivalOutcome=TRUE)
```

ltmle for right-censored data

```
Treatment Estimate:
   Parameter Estimate:  0.40391
    Estimated Std Err:  0.031247
              p-value:  <2e-16
    95% Conf Interval: (0.34267, 0.46515)

Control Estimate:
   Parameter Estimate:  0.49545
    Estimated Std Err:  0.029052
              p-value:  <2e-16
    95% Conf Interval: (0.43851, 0.55239)

Additive Treatment Effect:
   Parameter Estimate:  -0.091539
    Estimated Std Err:  0.041653
              p-value:  0.027973
    95% Conf Interval: (-0.17318, -0.0099008)

Relative Risk:
   Parameter Estimate:  0.81524
  Est Std Err log(RR):  0.095263
              p-value:  0.032009
```

ltmle for right-censored data

```
ltmle.fit$fit$g[[1]]
```

```
$A0
                  Risk       Coef
SL.glm_All  0.2502361 0.4568067
SL.mean_All 0.2501919 0.5431933

$C1
                  Risk       Coef
SL.glm_All  0.1533290 0.98494501
SL.mean_All 0.1667804 0.01505499

$C2
                  Risk     Coef
SL.glm_All  0.1365783 0.956276
SL.mean_All 0.1428211 0.043724

$C3
                  Risk       Coef
SL.glm_All  0.1373765 0.92097765
SL.mean_All 0.1440681 0.07902235
```

Practical 2: Application of `ltmle` (Task 2ff)

In this part of the practical, we consider application of `ltmle`. There are three different 'topics' as follows:

1. Static and dynamic interventions (continuing Task 1)
2. Super learning
3. Right-censored data structures

Proceed from **Task 2** of **day3_practical2.pdf**.