Journal of Statistical Software

October 2017, Volume 81, Issue 1. doi: 10.18637/jss.v081.i01

ltmle: An R Package Implementing Targeted Minimum Loss-Based Estimation for Longitudinal Data

Samuel D. Lendle
University of California, Berkeley

Joshua Schwab
University of California, Berkeley

Maya L. Petersen
University of California, Berkeley

Mark J. van der Laan
University of California, Berkeley

Abstract

In recent years, targeted minimum loss-based estimation methodology has been used to develop estimators of parameters in longitudinal data structures (Gruber and van der Laan 2012; Petersen, Schwab, Gruber, Blaser, Schomaker, and van der Laan 2014; Schnitzer, Moodie, van der Laan, Platt, and Klein 2013). These methods are implemented in the **ltmle** package for R. The **ltmle** package provides methods to estimate intervention-specific means and measures of association including the average treatment effect, causal odds ratio and causal risk ratio and parameters of a longitudinal working marginal structural model. The package allows for multiple time point treatments, time-varying covariates and right censoring of the outcome. In this paper we described the usage of the **ltmle** package and provide examples.

Keywords: targeted minimum loss-based estimation, longitudinal data, causal inference, estimation, R.

1. Introduction

Targeted minimum loss-based estimation (TMLE) is a framework for constructing regular and asymptotically linear estimators for a parameter in a statistical model (van der Laan and Rose 2011; van der Laan and Rubin 2006). TMLE methodology has been applied to many parameters that are interpretable as causal quantities in observational studies under assumptions. Examples include the average treatment effect (ATE), average treatment effect

among the treated (ATT), controlled direct effects (CDE), natural direct and indirect effects (NDE and NIE), and causal effects of multiple time point interventions (van der Laan and Rose 2011; Moore and van der Laan 2009; van der Laan and Gruber 2012; Zheng and van der Laan 2012; Lendle, Subbaraman, and van der Laan 2013).

The **tmle** (Gruber and van der Laan 2012) package for R (R Core Team 2017) provides estimators for parameters when all potential confounders are baseline (pre-exposure) variables, including the ATE, the CDE, and the parameters of a marginal structural model (MSM) for a single time point intervention. The **tmle** package also allows for missing outcomes when variables needed for the missing at random assumption to hold are not affected by the exposure. The package is available on the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=tmle` and is described by Gruber and van der Laan (2012).

A new R package, **ltmle**, has been developed to estimate parameters in longitudinal data structures. The **ltmle** package implements an estimator for intervention-specific means developed by Robins (2000, 2002); Bang and Robins (2005); van der Laan and Gruber (2012) (and by extension the ATE, CDE, causal risk ratio and causal odds ratio,) and an estimator for parameters of a longitudinal working marginal structural model developed by Petersen *et al.* (2014). The **ltmle** package also extends some of the capabilities of **tmle** by allowing outcome missingness to depend on post-baseline covariates. In this article we describe usage of the **ltmle** package. We have omitted many technical details and focus on practical usage of the **ltmle** through examples. For details on the method, including examples working through the estimation procedure with a few time points, see Petersen *et al.* (2014) and Schnitzer *et al.* (2013). The package is available on CRAN at `cran.r-project.org/package=ltmle`.

The article is organized as follows: in Section 2 the observed data structure and causal and statistical models are defined, as well as notation used in later sections; in Section 3 intervention-specific mean parameters are introduced and the **ltmle** package is used to estimate those parameters in examples; in Section 4, we define an MSM and demonstrate estimation of the parameters though an example; in Section 5 we discuss planned extensions to the package.

2. Observed data and statistical and causal models

2.1. Observed data structure and notation

In longitudinal studies, data are collected on observations at multiple time points and can be subject to censoring. One observation O is coded as

$$O = (L(0), A(0), \ldots, L(K), A(K), L(K+1)).$$

We observe n independent and identically distributed copies of O with distribution P_0. The L variables, or nodes, are covariates and the outcome of interest, and the A nodes are intervention nodes. Baseline covariates are called $L(0)$. The outcome variable, if it is measured at time k, is $Y(k)$, which is part of $L(k)$ for $k = 1 \ldots, K+1$. Elements of $L(k)$ other than $Y(k)$ are time-varying covariates. Multiple time-varying covariates can be included, and different time-varying covariates can occur at different time points. If there are no measured covariates or outcome at time k, $L(k)$ is null.

The intervention node $A(k) = (A_1(k), A_2(k))$ can in general include both a binary treatment ($A_1(k)$) and a right censoring indicator ($A_2(k)$). However, an intervention node does not necessarily need to include both treatment and censoring, so treatment or censoring nodes can be back to back. Back to back binary treatment nodes can be used to code treatments with more than two levels. For example, a 4 level treatment can be coded with pairs of treatment nodes: $(0, 0)$, $(0, 1)$, $(1, 0)$, and $(1, 1)$. Throughout, by censoring we mean right censoring, but other types of censoring and missingness can be handled using the treatment nodes. We describe this in Section 5.

We use an overbar with a time index to denote the history of a variable or function from time 0 up to the given time. For example, $\bar{L}(k) = (L(0), L(1), \ldots, L(k))$. If the time index is omitted, the overbar denotes history up to the last time point, e.g., $\bar{L} = \bar{L}(K + 1)$ and $\bar{A} = \bar{A}(K)$.

We use $Pa(\cdot)$ to denote the parents of a node. The parents of L nodes are all previous nodes in the time ordering: $Pa(L(k)) = (\bar{L}(k - 1), \bar{A}(k - 1))$. The parents of an intervention node are a known subset of the nodes occurring before the intervention node and potentially affect it.

2.2. Causal model and counterfactuals

In the previous section, we discuss the distribution of observed data. Given a data set, we can estimate parameters of this distribution, which we call statistical parameters. Without more structure, statistical parameters do not have a causal interpretation. To formally define a causal quantity we want to estimate, we first specify a structural causal model (SCM) or non-parametric structural equation model (Pearl 1995, 2000, 2009).

An SCM is a way to encode knowledge about the relationships of variables. In our case, the SCM will allow us to write each observed variable as an unknown deterministic function of the past variables and an unobserved error.

With the SCM, we can define counterfactuals: outcomes that would have happened had some (treatment or exposure) variables taken a possibly different value than they had in reality. The SCM implies a set of possible probability distributions for the counterfactual random variables of interest. Using the SCM, we can translate a scientific question of interest into a well defined causal quantity, which is a function of the distribution of counterfactuals. We call this a causal parameter.

We suppose that each component of the observed data structure is generated by a function of its parents and an unobserved exogenous error. Let

$$L(k) = f_{L(k)}(Pa(L(k)), U_{L(k)}) \text{ for } k = 0, \ldots K + 1$$

and

$$A(k) = f_{A(k)}(Pa(A(k)), U_{A(k)}) \text{ for } k = 0, \ldots K.$$

The functions $f_{L(k)}$ and $f_{A(k)}$ are deterministic and non-parametric. The U components are the unobserved errors and are random with an unknown distribution.

The counterfactuals we are interested in are defined by a sequence of treatments that we choose, called a static treatment regime, or more generally by a deterministic rule that depends on L values at previous time points, called a dynamic regime. A dynamic regime is a collection of deterministic functions, $d = (d_k : k \in \{0, \ldots, K + 1\})$, for assigning treatment at time k as

a function of the past L nodes. For a dynamic regime d, define the counterfactual

$$L_d(k) = f_{L(k)}((\bar{L}_d(k-1), d_k(\bar{L}_d(k-1))), U_{L(k)}) \text{ for } k = 0, \ldots, K+1,$$

letting $\bar{L}_d(-1)$ be null for convenience. That is, we replace the intervention nodes in $f_{L(k)}$ with those set by our rule, and previous L nodes by their previously generated counterfactual values. The counterfactual values of L are then generated sequentially for each time point k.

Though the mathematical definition of a dynamic regime is more complicated than a static regime, dynamic regimes are often more realistic and intuitive because in practice, treatment decisions are usually made based on the current and past state of a patient, and not pre-specified as in a static regime. As an example, a rule might be to start a patient on treatment 1 at time 0, and to keep her on that treatment until some biomarker measured at each time point falls below a certain level. Once the biomarker crosses a threshold, the treatment is then switched to 0. Typically we are only interested in counterfactuals under no censoring, so we assume d_k always sets $A_2(k) = 0$.

A static regime is a special case of a dynamic regime, where treatment is set according to a predetermined sequence $\bar{a}$ that is not a function of past L nodes. For most of the remainder of the article we will limit the discussion to dynamic regimes, but we include an example of a static regime in Section 3.2.

2.3. Identifiability

As described in Section 2.2, the causal parameters we wish to estimate in the following sections are parameters of the distribution of counterfactuals $\bar{L}_d$ for $d \in \mathcal{D}$, a set of regimes of interest. In order to estimate these parameters from the observed data, we need to determine if the distribution of $\bar{L}_d$ is identifiable, meaning that we can express the distribution of $\bar{L}_d$ in terms of the distribution of the observed data. We briefly review identifiability assumptions here. For further discussion, please see (Robins 1986; van der Laan and Gruber 2012; Petersen *et al.* 2014).

Two key assumptions that allow us to identify the distribution of $\bar{L}_d$ are the sequential randomization assumption Robins (1986) (Assumption 1) and the positivity assumption (Assumption 2).

Assumption 1.

$$A(k) \perp L_d \mid Pa(A(k)) \text{ for } k = 1, \ldots, K.$$

Assumption 2.

$$P_0(A(k) = d_k(\bar{L}(k)) \mid \bar{L}(k), \bar{A}(k-1) = \bar{d}(\bar{L}(k-1))) > 0 \text{ almost everywhere.}$$

Under Assumptions 1 and 2, the distribution of the counterfactual $\bar{L}_d$ is identified by the longitudinal G-computation formula(Robins 1986). Informally, Assumption 1 means that we must be able to assume that at each time point k, all common causes of the L nodes and $A(k)$ are observed and included in our dataset. This is a "no unmeasured confounders" type assumption. Assumption 1 puts no restrictions on the observed data distribution P_0 so does not affect our statistical model. Assumption 2 can be interpreted as assuming that each observation has a positive probability of following rule d at each time point.

Though not required for identifiability, we may make additional assumptions about the conditional distribution of intervention nodes given the past on top of Assumption 2. For example, we may make a Markov type assumption by assuming the conditional probability of treatment only depends on a fixed number of recent time points. Our statistical model, the set of possible distributions for the observed data, is semiparametric in general.

One class of causal parameters of interest is the intervention-specific mean, $E_0(Y_d(t))$ for a particular intervention d and time t, where E_0 denotes expectation. We may be interested in the intervention-specific mean under different $d \in \mathcal{D}$ and $t \in \tau$, where $\tau \subseteq \{1, \ldots, K+1\}$ is the set of times of interest. If we are only interested in the counterfactual Ys at one time point, say the last, then $\tau = \{K+1\}$. If we are interested the counterfactual Ys as a function of time, τ can include more or all time points where an outcome Y is measured.

Bang and Robins (2005) show that under Assumptions 1 and 2, $E_0(Y_d(t))$ can be identified through a sequence of recursively defined conditional expectations, which we define here. For $t \in \tau$, let

$$\bar{Q}^{d,t}_{L(t)} = E_0(Y(t) \mid \bar{L}(t-1), \bar{A}(t-1) = \bar{d}_{t-1}(\bar{L}(t-1))).$$

This is the regression of $Y(t)$ given the observed past covariates, but with intervention nodes set based on rule d. This quantity, $\bar{Q}^{d,t}_{L(t)}$, is then regressed on covariates and intervention nodes set by d up to time $t-2$, then that object is regressed on $t-3$, and so on until time 0. For $k = t-1, \ldots, 1$,

$$\bar{Q}^{d,t}_{L(k)} = E_0(\bar{Q}^{d,t}_{L(k+1)} \mid \bar{L}(k-1), \bar{A}(k-1) = \bar{d}_{k-1}(\bar{L}(k-1))).$$

For notational convenience, let $\bar{L}(-1)$ and $\bar{A}(-1)$ be the null, so $\bar{Q}^{d,t}_{L(0)}$ is constant. Under the above assumptions, $E_0(Y_d(t)) = \bar{Q}^{d,t}_{L(0)}$. For more details and a derivation of these terms, see van der Laan and Gruber (2012) and Petersen *et al.* (2014).

A second class of causal parameters are the parameters of a working marginal structural model, which depend on $E_0(Y_d(t))$ for $d \in \mathcal{D}$, $t \in \tau$. The recursive conditional expectation representation of the target statistical parameter for intervention-specific means can be extended to working MSMs(Bang and Robins 2005; Petersen *et al.* 2014).

In Sections 3 and 4, the intervention-specific mean and working marginal structural model parameters are described in more detail, and we show how the **ltmle** package can be used to estimate them.

2.4. Additional notation

Here we define some additional notation we will use when describing the usage of the package.

Call the collection of all of these sequential regressions

$$\bar{Q} = (\bar{Q}^{d,t}_{L(k)} : t \in \tau, k \in \{1, \ldots, t\}, d \in \mathcal{D}).$$

For $k = 1, \ldots, K$, let

$$\begin{aligned} g_k(A(k) \mid Pa(A(k))) =& g_{1,k}(A_1(k) \mid Pa(A_1(k)))g_{2,k}(A_2(k) \mid Pa(A_2(k))) \\ =& P_0(A_1(k) \mid Pa(A_1(k)))P_0(A_2(k) \mid Pa(A_2(k))) \end{aligned}$$

be the conditional distribution of each intervention node $A(k)$ given its parents. These conditional distributions make up the so-called intervention distribution. The time ordering of

treatment and censoring nodes at a given time point can be chosen by the user in the implementation of the TMLE algorithm in the **ltmle** package. Let $g_k = (g_{1,k}, g_{2,k})$ and let $g = (g_k : k \in \{0, \dots K\})$ be the entire intervention mechanism.

The procedures in the **ltmle** package use estimates of the sequential regressions $\bar{Q}^{d,t}_{L(k)}$, and the intervention mechanism components $g_{1,k}$ and $g_{2,k}$. For some $t \in \tau$ and starting with $k = t$, an initial estimate of $\bar{Q}^{d,t}_{L(k)}$ is updated using an estimate of $g_{0:k-1} = \prod_{j=0}^{k-1} g_j$. Then, for $k = t-1$ down to 1 and using the updated estimate of $\bar{Q}^{d,t}_{L(k+1)}$ as the outcome, an initial estimate of $\bar{Q}^{d,t}_{L(k)}$ is computed and subsequently updated with an estimate of $g_{0:k-1}$ (Petersen *et al.* 2014). The user can specify how each component is estimated as we demonstrate in the following sections.

3. Intervention-specific means

3.1. Causal parameter

For a regime d, we may be interested in the mean counterfactual outcome at some time point if all observations in the population followed that rule. Without loss of generality, suppose this time is $K+1$, and let $Y = Y(K+1)$. We call this an intervention-specific mean, and it can be written as $E_0(Y_d)$.

We may also be interested in comparing two different regimes, d and d'. One possible comparison is the additive effect, the difference in intervention-specific means:

$$E_0(Y_d) - E_0(Y_{d'}).$$

If the outcome a binary indicator of an event, the intervention-specific mean is also the probability of the event, so two regimes can also be compared with a causal risk ratio,

$$\frac{E_0(Y_d)}{E_0(Y_{d'})},$$

or a causal odds ratio,

$$\frac{\frac{E_0(Y_d)}{1-E_0(Y_d)}}{\frac{E_0(Y_{d'})}{1-E_0(Y_{d'})}}.$$

In order to be able to interpret estimates based on the observed data as estimates of these quantities, Assumptions 1 and 2 must hold for d and d'.

3.2. Estimation using the ltmle package

Estimation of an intervention-specific mean or a comparison (additive effect, risk ratio or odds ratio) between two intervention-specific means with the **ltmle** package is performed with the `ltmle` function.

Specification of the data set

The data set is passed to `ltmle` through the `data` argument as a data frame, with $L(0)$ variables in the first (leftmost) columns, then $A(1)$, $L(1)$, and so on. Treatment nodes, $A_1(k)$, and censoring nodes, $A_2(k)$, are specified by the `Anodes` and `Cnodes` arguments, respectively. Censoring nodes should be coded as a factor with levels `'censored'` or `'uncensored'`. The values of variables after a censoring node with level `'censored'` are ignored and can be `NA`. Outcome nodes, $Y(k)$ are specified with the `Ynodes` argument, and time-varying covariates, the components of L nodes other than the Y nodes, are specified with the `Lnodes` argument. Baseline covariates, $L(0)$, are not included in `Lnodes`. These arguments are specified with column indexes or names in the data frame.

The `ltmle` function can handle continuous or binary outcomes, and it detects the type of outcome automatically. If all outcome variables listed in the `Ynodes` option are binary, then an additional option, `survivalOutcome` must be specified. If $Y(k)$ is an indicator of an event that could go from 1 back to 0 or occur more than once, `survivalOutcome` should be `FALSE`. If $Y(k)$ is an indicator of an event at or before time k that can only happen once such as death or first heart attack, then `survivalOutcome` should be `TRUE`. The second case is discussed in more detail in an example below.

The `ltmle` function assumes the time point of interest is at time $K + 1$. If the time point of interest is some other time, we can simply ignore variables from later time points, and only include variables up to the time point of interest in `data`.

Specification of the treatment regime

A static treatment regime is specified with the `abar` argument, which is a binary vector with one entry for each A_1 node. A dynamic regime is specified by the `rule` argument, which takes a function that operates on a single row of `data`, which is a named vector. The function returns a binary vector with one entry per A_1 node corresponding to the treatment that observation would receive at that node under the dynamic regime of interest. For example, for a data set with two treatment nodes, a regime might assign treatment at time 1 for every observation and then only assign treatment at time 2 to those observations for which a variable 'score' is greater than some threshold, say 10. This would be coded as

```
ltmle(..., rule = function(row) c(1, ifelse(row["score"] > 10, 1, 0)))
```

Alternatively, a dynamic regime can be specified by passing a matrix to the `abar` argument with one row per observation, where each row corresponds to the vector of treatments (with one entry per A_1 node) that observation would receive under the regime of interest.

Specification of estimation for sequential regressions and intervention mechanism

Both the sequential regressions $\bar{Q}$ and intervention mechanism g need to be estimated in the TMLE procedure. Consistency and efficiency of the final estimate depend on consistency of estimates for $\bar{Q}$ and g. By consistent, we mean that estimates of components of $\bar{Q}$ and g converge to the true values under P_0, $\bar{Q}_0$ and g_0. The components of both $\bar{Q}$ and g are all conditional means, so estimating these components comes down to regression. However, because our goal is to estimate a target parameter, which is averaged over the covariates, we are not particularly interested in interpreting the fit of any one of these regressions for a component of $\bar{Q}$ or g.

Components of $\bar{Q}$ and g can be estimated with parametric generalized linear models using R's `glm` function, but because the model is non-parametric, parametric models for each component are generally not correctly specified. In practice, more flexible estimation is recommended. The super learner algorithm (van der Laan, Polley, and Hubbard 2007) can also be used to estimate components of $\bar{Q}$ and g as implemented in the **SuperLearner** package. This is a data adaptive algorithm which can leverage a variety of candidate estimators already available in R including flexible machine learning algorithms and parametric models. The algorithm chooses the best weighted combination of candidate estimators via cross-validation. This combination is guaranteed to perform asymptotically as well or better than any algorithm in the library of candidate algorithms, and has also been demonstrated to perform well with realistic sample sizes (van der Laan *et al.* 2007).

Choosing between `glm` and the super learner algorithm (as well as which candidate estimators to include in super learner) involves a trade off between bias, variance, and computation time. The `glm` will generally have higher bias and lower variance than super learner as well as lower computation time. A super learner estimator with a library of diverse candidate estimators will generally have lower bias than a single parametric model, so super learner estimates are more likely to be consistent when estimating components of $\bar{Q}$ and g.

The TMLE is doubly robust, meaning consistent estimation only one of $\bar{Q}$ or g is needed for consistency of the final estimate. When both $\bar{Q}$ and g are estimated consistently the TMLE is efficient, meaning that the TMLE achieves the best possible variance asymptotically. As a result, using the super learner algorithm with a library of many candidate estimators is recommended.

When `glm` is used, components of the intervention mechanism $g_{1,k}$ and $g_{2,k}$ are estimated with logistic regression. Logistic regression is also used for components of $\bar{Q}$. First, Y variables are automatically transformed to be between 0 and 1. When an outcome is bounded by 0 and 1, the negative Bernoulli log likelihood is a valid loss function for the conditional mean (Gruber and van der Laan 2010), so logistic regression can also be used on any outcome that is between 0 and 1, even if it is not binary, such as $\bar{Q}_k^{K+1,d}$. Estimates based on logistic regression have the attractive property that all predicted means will also be bounded by 0 and 1, which is not the case for linear regression. Keeping the estimates for $\bar{Q}^{d,t}$ properly bounded ensures that the final parameter estimates will respect their bounds, and reduces bias and variance in small samples (Gruber and van der Laan 2010). If Y variables were originally transformed to be between 0 and 1, the package automatically transforms estimates back to the original scale.

The use of **SuperLearner** is determined by the `SL.library` argument, which is either `NULL`, indicating `glm` is to be used, or a list with elements `"Q"` for the library to be used to estimate $\bar{Q}$ and `"g"` for the library for estimating g. If one is `NULL`, then it is estimated with `glm`. We show an example using super learner below, and more information can be found in the documentation for the **ltmle** package (Schwab, Lendle, Petersen, and van der Laan 2017).

Formulas can be specified for estimating components of $\bar{Q}$ and g using the `Qform` and `gform` arguments, which are vectors of strings to be coerced to formulas with one formula for each time point. More details on the structure of the formulas are given in the package documentation(Schwab *et al.* 2017). When super learner is used, the functional form of the formulas is unimportant; all variables on the right hand side of the formulas are considered as predictor variables passed to the candidate estimators when estimating the regression. If other func-

tions of the predictor variables are desired when using super learner, for example interaction, polynomial, or spline terms, they are specified using the appropriate candidate estimators. For details on creating custom candidate estimators for super learner, see Polley, LeDell, and van der Laan (2017). If the `Qform` and `gform` arguments arguments are left unspecified, the formulas default to main terms regressions that include all parent nodes in the time ordering.

Finally we can choose to estimate $\bar{Q}^{d,t}_{L(k)}$ either using all observations or stratifying by regime, and only using observations following regime d. This is set by the `stratify` argument. When stratifying by regime, there will typically be less bias in the estimate of $\bar{Q}^{d,t}_{L(k)}$, resulting in less bias in the final TMLE estimate, but there may be more variance in the estimate, particularly if there are few observations following rule d. Similarly, without stratifying, an estimate of $\bar{Q}^{d,t}_{L(k)}$ will typically be less variable but bias may increase due to smoothing over all observations.

Additional options

In some cases, we may have additional information about the data. Specifically, we may know $g_{1,k}$ or $g_{2,k}$. For example, in a clinical trial, it may be possible to switch from treatment 1 to treatment 0 throughout the study, but not from treatment 0 to 1. For any patient with $A(k-1) = 0$, we know the probability that $A(k)$ is 1 given its parents is 0. This knowledge can be specified with the `deterministic.g.function` argument. Similarly, knowledge about $\bar{Q}$ can be specified with the `deterministic.Q.function` argument. Usage details are provided in the package documentation (Schwab *et al.* 2017).

Example 1: Static treatment regime

In this example we have a simulated dataset of 500 observations with two post-baseline time points and censoring. Baseline covariates are called `L0.a`, `L0.b`, and `L0.c`. Treatment assigned at baseline is `A0`, and `C0` is a censoring indicator. Covariates at time 1 are `L1.a`, `L1.b`, the value of the outcome at time 1 is called `Y1`, and treatment and censoring variables are `A1` and `C1`, respectively. The outcome of interest is called `Y2`. The Y variables are continuous and between 0 and 1. Code to generate the example dataset is given in the supplementary material.

Suppose we are interested in the mean outcome at time 2 if both treatment nodes are set to 1, i.e., $\bar{a} = (1, 1)$ and our target parameter is $E_0(Y_{\bar{a}})$. We begin by specifying L, A, and Y nodes and then call the `ltmle` function. A standard error and confidence interval is computed with `summary`. The variance estimate is based on the variance of the estimated influence curve.

```
R> Lnodes <- c("L1.a", "L1.b")
R> Anodes <- c("A0", "A1")
R> Cnodes <- c("C0", "C1")
R> Ynodes <- c("Y1", "Y2")
R> EY.11 <- ltmle(exData1, Anodes = Anodes, Cnodes = Cnodes, Lnodes = Lnodes,
+    Ynodes = Ynodes, abar = c(1, 1), estimate.time = FALSE)
R> print(summary(EY.11))

Estimator:  tmle
Call:
```

```
ltmle(data = exData1, Anodes = Anodes, Cnodes = Cnodes, Lnodes = Lnodes,
    Ynodes = Ynodes, abar = c(1, 1), estimate.time = FALSE)

   Parameter Estimate:  0.47833
    Estimated Std Err:  0.010726
              p-value:  <2e-16
    95% Conf Interval: (0.45731, 0.49935)
```

If instead we would like to estimate the average treatment effect comparing $\bar{a}$ to another treatment, say $\bar{a}' = (0, 0)$, we can pass a `list` with named elements `treatment` and `control` to the `abar` argument of `ltmle`. The ATE, $E_0(Y_{\bar{a}}) - E(Y_{\bar{a}'})$, is computed with `summary`. If the outcome is binary, this also computes the causal risk ratio and causal odds ratio. Again, the variance is estimated using the estimated influence curve.

```
R> ATE <- ltmle(exData1, Anodes = Anodes, Cnodes = Cnodes, Lnodes = Lnodes,
+    Ynodes = Ynodes, abar = list(treament = c(1, 1), control = c(0, 0)),
+      estimate.time = FALSE)
R> print(summary(ATE))

Estimator:  tmle
Call:
ltmle(data = exData1, Anodes = Anodes, Cnodes = Cnodes, Lnodes = Lnodes,
    Ynodes = Ynodes, abar = list(treament = c(1, 1), control = c(0,
        0)), estimate.time = FALSE)

Treatment Estimate:
   Parameter Estimate:  0.47833
    Estimated Std Err:  0.010726
              p-value:  <2e-16
    95% Conf Interval: (0.45731, 0.49935)

Control Estimate:
   Parameter Estimate:  0.30126
    Estimated Std Err:  0.007591
              p-value:  <2e-16
    95% Conf Interval: (0.28638, 0.31614)

Additive Treatment Effect:
   Parameter Estimate:  0.17707
    Estimated Std Err:  0.012274
              p-value:  <2e-16
    95% Conf Interval: (0.15302, 0.20113)
```

Example 2: Dynamic treatment regime and super learner

With the same data set, we may be interested in the mean counterfactual outcome, $E(Y_d)$, under a dynamic regime d where treatment at time 0 is set to 1, and at time 1, treatment is set

to 1 if `L1.b` is positive, and 0 otherwise. Additionally, we will now estimate the components of g with the super learner algorithm, and continue to estimate $\bar{Q}$ with `glm`. Instead of choosing the default library, we will specify one. The specified library requires that packages **nnet** (Ripley 2016), **gam** (Hastie 2017), and **glmnet** (Friedman, Hastie, and Tibshirani 2010) are installed.

```
R> d <- function(row) c(1, ifelse(row["L1.b"] > 0, 1, 0))
R> SL.lib <- c("SL.glm", "SL.stepAIC", "SL.nnet", "SL.gam", "SL.glmnet")
R> EY.d <- ltmle(exData1, Anodes = Anodes, Cnodes = Cnodes, Lnodes = Lnodes,
+    Ynodes = Ynodes, rule = d, SL.library = list(Q = NULL, g = SL.lib),
+    estimate.time = FALSE)
R> print(summary(EY.d))

Estimator: tmle
Call:
ltmle(data = exData1, Anodes = Anodes, Cnodes = Cnodes, Lnodes = Lnodes,
    Ynodes = Ynodes, rule = d, SL.library = list(Q = NULL, g = SL.lib),
    estimate.time = FALSE)

Parameter Estimate: 0.43755
Estimated Std Err: 0.010608
p-value: <2e-16
95% Conf Interval: (0.41676, 0.45834)
```

Example 3: Survival analysis

In this example, we use a modified version of the dataset in the previous example, with continuous Y nodes replaced with survival indicators. If the outcome of interest is survival, $Y(k)$ is a binary variable indicating that the event of interest occurred at or before time k. Then for a regime d, $1 - E(Y_d)$ is interpretable as a counterfactual survival probability, so $E_0(Y_d)$ is the probability of an event at or before time $K + 1$ under regime d.

A survival outcome is specified by setting the `survivalOutcome` argument to `TRUE`. This tells the procedure that once a Y node has jumped from 0 to 1, indicating that the event has occurred, it will remain at 1 at subsequent time points. The `ltmle` function will terminate with an error if the `data` argument does not conform to this structure.

Because the target parameter can be interpreted as a probability, when two regimes are compared, `summary` displays an estimated risk ratio and odds ratio in addition to the additive effect. Note that risk ratios and odds ratios are also displayed when the Y variables are binary but `survivalOutcome=FALSE`. We demonstrate this by comparing regimes $\bar{a} = (1, 1)$ and $\bar{a}' = (0, 0)$.

```
R> ATE.survival <- ltmle(exData2, Anodes = Anodes, Cnodes = Cnodes,
+    Lnodes = Lnodes, Ynodes = Ynodes, survivalOutcome = TRUE,
+    abar = list(treatment = c(1, 1), control = c(0, 0)),
+    estimate.time = FALSE)
R> print(summary(ATE.survival))
```

```
Estimator:  tmle
Call:
ltmle(data = exData2, Anodes = Anodes, Cnodes = Cnodes, Lnodes = Lnodes,
    Ynodes = Ynodes, survivalOutcome = TRUE, abar = list(treatment = c(1,
        1), control = c(0, 0)), estimate.time = FALSE)

Treatment Estimate:
   Parameter Estimate:  0.29047
    Estimated Std Err:  0.042383
              p-value:  7.2068e-12
    95% Conf Interval: (0.2074, 0.37354)

Control Estimate:
   Parameter Estimate:  0.021726
    Estimated Std Err:  0.011103
              p-value:  0.050366
    95% Conf Interval: (0, 0.043487)

Additive Treatment Effect:
   Parameter Estimate:  0.26875
    Estimated Std Err:  0.043813
              p-value:  8.5747e-10
    95% Conf Interval: (0.18287, 0.35462)

Relative Risk:
   Parameter Estimate:  13.37
  Est Std Err log(RR):  0.53145
              p-value:  1.0658e-06
    95% Conf Interval: (4.7179, 37.887)

Odds Ratio:
   Parameter Estimate:  18.434
  Est Std Err log(OR):  0.5614
              p-value:  2.0925e-07
    95% Conf Interval: (6.1341, 55.396)
```

3.3. Summary of key arguments to the `ltmle` function

For full details, see the documentation for the **ltmle** package (Schwab *et al.* 2017).

- `data`: A data frame where the order of the columns corresponds to the time-ordering of the model.
- `Anodes`: Names or indexes of treatment nodes.
- `Cnodes`: Names or indexes of censoring nodes.
- `Lnodes`: Names or indexes of time-varying covariate nodes.

- `Ynodes`: Names or indexes of outcome nodes.
- `survivalOutcome`: Set to `TRUE` if the outcome is an event that can occur only once, e.g., death or first diagnosis of a disease, or `FALSE`, the default, otherwise. Must be `FALSE` for continuous outcomes.
- `Qform`: A character vector of regression formulas for $\bar{Q}$.
- `gform`: A character vector of regression formulas for g.
- `abar`: A binary vector of length `length(Anodes)` or matrix of size n by `length(Anodes)` of counterfactual treatment assignments or a list of length 2 (to contrast two treatments).
- `rule`: A function to be applied to each row (a named vector) of `data` that returns a numeric vector of treatment assignments of length `length(Anodes)` or a list of length 2.
- `gbounds` A vector of lower and upper bounds on estimated g components.
- `Yrange`: Optionally specify the range of all Y nodes.
- `deterministic.g.function`: Optional information on A and C nodes that are deterministic.
- `stratify`: If `TRUE` stratify on following `abar` when estimating $\bar{Q}$ and g. If `FALSE`, the default, pool over `abar`.
- `SL.library`: Optional character vector of libraries to pass to use with the **SuperLearner** package. `NULL` indicates that `glm` should be used to estimate $\bar{Q}$ and g. `default` indicates a standard set of libraries. May be separately specified for $\bar{Q}$ and g.
- `estimate.time`: If `TRUE`, compute a rough estimate of runtime based on an initial estimate using only 50 observations.
- `deterministic.Q.function`: Optional information on $\bar{Q}$ given deterministically.
- `observation.weights`: Optional sampling weights for each observation.

4. Marginal structural models for static and dynamic regimes

4.1. Causal parameter

In some settings, the researcher might be interested in how the counterfactual expected outcome varies as a function of static or regime, or time. One way to specify such a target parameter is with a MSM (Robins 1998), or if we do not want to assume that model is correct, a working MSM (Neugebauer and van der Laan 2007). Marginal structural models were originally developed for static regimes and were later extended to dynamic regimes (Neugebauer and van der Laan 2007; Robins, Orellana, and Rotnitzky 2008).

The true dose response curve as a function of treatment regimes in a set $\mathcal{D}$ at time $t \in \tau$ is $(E_0(Y_d(t)) : d \in \mathcal{D}, t \in \tau)$. Recall that τ is the index set of time points of interest. We can

summarize this curve using an MSM, $\Theta = \{m_\beta : \beta\}$. When the outcome is binary or bounded by 0 and 1, we can use a logistic model, as implemented in the **ltmle** package:

$$\text{logit}\, m_\beta(d,t) = \sum_{j=0}^{J-1} \beta_j \phi_j(d,t).$$

Here logit is the log odds function: $\text{logit}(x) = \log(x/(1-x))$. When the outcome of interest is an indicator of survival at or before that time point and the probability of an observation having an event at a particular time point is small given that they are at still at risk for the event, the logistic working MSM can be used to approximate a time dependent Cox model by including a separate intercept term for each time point (Hernán, Brumback, and Robins 2000).

In general, m_β does not capture the true functional form of $E_0(Y_d(t))$, so we treat $\{m_\beta : \beta\}$ as a working model. The working model is a way of summarizing the true dose response curve with a few parameters. We define the causal quantity of as a projection onto the model:

$$\beta_0 = \arg\min_\beta -E_0 \sum_{t\in\tau} \sum_{d\in\mathcal{D}} h(d,t)\{Y_d(t)\log(m_\beta(d,t)) + (1-Y_d(t))\log((1-m_\beta(d,t))\}$$

where $h(d,t)$ is a user specified weight function. If the functional form of $E_0(Y_d(t))$ is correctly described by m_β, then h does not affect β_0 and will only affect the efficiency of the estimation procedure. The current version of the **ltmle** package weights by the empirical probability of following rule d by default, or uses constant weights, i.e., $h(d,t) = 1$, when the argument `weight.msm` is `FALSE`.

4.2. Estimation using the ltmle package

Estimation of the parameters of an MSM is done with the `ltmleMSM` function. The data set is specified in the same way as the `ltmle` function described in Section 3.2. Regimes are specified as functions as described in Section 3.2 but all regimes in $\mathcal{D}$ are passed to `ltmle` at once as a list via the `regimes` argument.

Estimation methods for components of $\bar{Q}$ and g are specified as described in Section 3.2. At a given time point k, the same formula is used for all $t \in \tau$ for $\bar{Q}_k^{d,t}$. For example, suppose time points of interest include times 2 and 3. The sequential regressions $\bar{Q}_1^{d,2}$ and $\bar{Q}_1^{d,3}$ are estimated using the same formula in the `Qform` argument corresponding to $k = 1$.

Specification of the MSM

The time points of interest, τ, can include all times with a Y node, or just a subset. This is specified with the argument `final.Ynodes`, which is a vector of the variable names of $Y(t)$ for $t \in \tau$.

The functional form of the MSM is specified via the `working.msm` argument as a character string to be coerced to a formula with `Y` on the left hand side. The variables on the right hand side are summary functions of d and t, $\phi_j(d,t)$, and the names are don't need to be the same as variable names in `data`. In fact, using different names is recommended to avoid confusion.

The values of $\phi_j(d,t)$ are specified with the `summary.measures` argument. This is a $|\mathcal{D}| \times m \times |\tau|$ array where m is the number of variables on the right hand side of `working.msm`. The dimension names for the 2nd dimension of `summary.measures` should correspond to the variable

names on the right hand side of the `working.msm` formula. The value of `summary.measures[i, "varname", k]` is the value of `varname`, a variable in `working.msm` for the ith regime in $\mathcal{D}$ at the kth time point in τ. We demonstrate the construction of `summary.measures` in the example below.

Additional options

Like in the `ltmle` function, knowledge about $\bar{Q}$ and g can be specified with the `deterministic.acnode.function` and `deterministic.Q.function` arguments.

We can choose to perform the targeting step of TMLE by pooling across regimes, which is the default, or to stratify by regime before performing the targeting step. The latter choice is the method described in Schnitzer *et al.* (2013). Stratifying by regime essentially yields a more saturated model for $\bar{Q}$ so bias may be decreased, but, particularly when there are only a few observations following some regimes, the variance can be higher in small samples. Petersen *et al.* (2014) discuss scenarios where one choice may perform better than the other in more detail. This is specified through the `pooled` argument.

Example 4: MSM

In this example, we use a simulated dataset included in the package (`sampleDataForLtmleMSM`) based on an example in Petersen *et al.* (2014). The real dataset cannot be distributed, but code from the original analysis is available via a web supplement (Petersen *et al.* 2014). Here $n = 200$. Prior to baseline, all patients were receiving treatment 0, and after baseline may be switched to treatment 1 at time 0, 1, or 2, or may never switch. The outcome is a survival outcome, so $Y(k)$ is an indicator of death at or before time k up to $K + 1 = 3$. There is one time-varying covariate, CD4 count at each time point and no censoring.

We are interested in how the risk of death is related to choice of switching time following immunological failure. Let $i = 0, \ldots, 3$ be the switch time, and define regime d^i as setting $A(k) = 0$ for $k < i$ and $A(k) = 1$ for $k \geq i$. The regime d^3 denotes never switching. Because these regimes do not depend on covariates, they are static regimes, but MSMs can also be specified with dynamic regimes in general. We choose as a working MSM

$$\operatorname{logit} m_\beta(d^i, t) = \beta_0 + \beta_1 t + \beta_2 \max(t - i, 0).$$

This is specified with the `working.msm` argument as `"Y ~ time + pmax(time - switch.time, 0)"`. We demonstrate the construction of the `summary.measures` array in the code below.

```
R> Lnodes <- c("CD4_1", "CD4_2")
R> Anodes <- c("A0", "A1", "A2")
R> Ynodes <- c("Y1", "Y2", "Y3")
R> D <- list(function(row) c(1, 1, 1), function(row) c(0, 1, 1),
+    function(row) c(0, 0, 1), function(row) c(0, 0, 0))
R> summary.measures <- array(dim = c(4, 2, 3))
R> dimnames(summary.measures)[[2]] <- c("switch.time", "time")
R> summary.measures[, , 1] <- cbind(0:3, rep(1, 4))
R> summary.measures[, , 2] <- cbind(0:3, rep(2, 4))
R> summary.measures[, , 3] <- cbind(0:3, rep(3, 4))
```

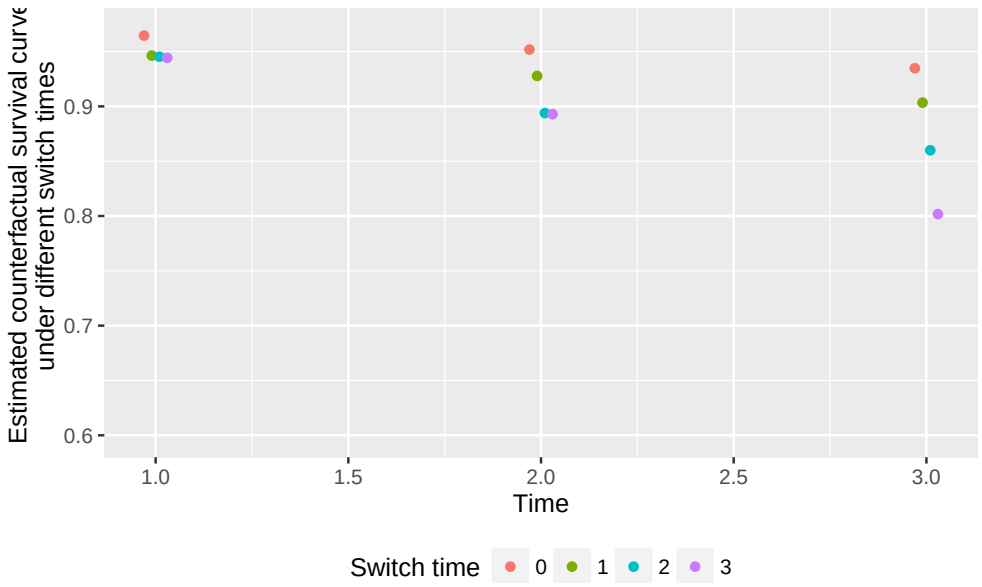

Figure 1: Estimated counterfactual survival curves for working MSM.

```
R> MSM.estimates <- ltmleMSM(sampleDataForLtmleMSM$data, Anodes = Anodes,
+    Lnodes = Lnodes, Ynodes = Ynodes, survivalOutcome = TRUE,
+    regimes = D, summary.measures = summary.measures, final.Ynodes = Ynodes,
+    working.msm = "Y ~ time + pmax(time - switch.time, 0)",
+    estimate.time = FALSE)
R> print(summary(MSM.estimates))

Estimator:  tmle
                            Estimate Std. Error CI 2.5% CI 97.5%  p-value
(Intercept)                  -3.6255     0.5133 -4.6316   -2.619 1.63e-12 ***
time                          0.7364     0.2163  0.3124    1.160 0.000664 ***
pmax(time - switch.time, 0)  -0.4155     0.2088 -0.8248   -0.006 0.046633 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Examining the parameter estimates, we see that the coefficient for `pmax(time - switch.time, 0)` is negative. This suggests switching treatments earlier generally reduces the odds of death. We plot the estimated counterfactual survival curves at discrete times 1, 2, and 3 in Figure 1. The plot was created using the **ggplot2** (Wickham 2009) package.

4.3. Summary of key arguments to the `ltmleMSM` function

For full details, see the documentation for the **ltmle** package (Schwab *et al.* 2017).

- `data`: A data frame where the order of the columns corresponds to the time-ordering of the model.
- `Anodes`: Names or indexes of treatment nodes.

- `Cnodes`: Names or indexes of censoring nodes.
- `Lnodes`: Names or indexes of time-varying covariate nodes.
- `Ynodes`: Names or indexes of outcome nodes.
- `survivalOutcome`: Set to `TRUE` if the outcome is an event that can occur only once, e.g., death or first diagnosis of a disease, or `FALSE`, the default, otherwise. Must be `FALSE` for continuous outcomes.
- `Qform`: A character vector of regression formulas for $\bar{Q}$.
- `gform`: A character vector of regression formulas for g.
- `gbounds` A vector of lower and upper bounds on estimated g components.
- `Yrange`: Optionally specify the range of all Y nodes.
- `deterministic.g.function`: Optional information on A and C nodes that are deterministic.
- `stratify`: If `TRUE` stratify on following `abar` when estimating $\bar{Q}$ and g. If `FALSE`, the default, pool over `abar`.
- `SL.library`: Optional character vector of libraries to pass to use with the **SuperLearner** package. `NULL` indicates that `glm` should be used to estimate $\bar{Q}$ and g. `default` indicates a standard set of libraries. May be separately specified for $\bar{Q}$ and g.
- `estimate.time`: If `TRUE`, compute a rough estimate of runtime based on an initial estimate using only 50 observations.
- `deterministic.Q.function`: Optional information on $\bar{Q}$ given deterministically.
- `observation.weights`: Optional sampling weights for each observation.
- `regimes`: A binary array of dimension n by `length(Anodes)` by number of regimes of counterfactual treatment assignments, or a list of `rule` functions.
- `working.msm`: A character formula for the working marginal structural model.
- `final.Ynodes`: A subset of `Ynodes` used in the MSM to pool over a set of outcome nodes. By default, all `Ynodes` are included.
- `summary.measures`: An array of dimension number of regimes by number of summary measure by `length(final.Ynodes)`. Each slice along the first dimension summarizes a regime, and can be used on the right hand side of `working.msm`.
- `msm.weights`: Weights for the working MSM. Defaults to `"empirical"`, where each regime is weighted by the empirical proportion of observations following that regime at each time point. If `NULL`, weights of 1 are used. User specified weights can be given as an array of dimension n by number of regimes by `length(final.Ynodes)`.

5. Discussion

The **ltmle** package was developed to provide researchers access to a flexible implementation of the TMLE algorithm for general longitudinal data structures. Two large classes of causal parameters are estimated by the package, namely the intervention-specific mean and comparisons of two intervention-specific means, estimated by the `ltmle` function, and the parameters of a logistic working MSM, estimated by the `ltmleMSM` function. The longitudinal data structure allows for baseline covariates, multiple time point treatments, time-varying covariates, and censoring.

The censoring nodes in the package handle right censoring. However, treatment nodes can be used to handle other types of censoring. If a variable can be missing at some time points and observed again in the future, we can create an additional treatment node at each time point whose value indicates whether the variable is missing or not. Then, we set this new treatment node to prevent missingness when define regimes of interest. For time-varying covariates other than the outcome that are subject to missingness, there is another option. We can encode that variable as a pair in an L node: an indicator of missingness, and observed value if it is not missing, or a dummy value, say 0, if it is missing. This information is always observed. In either case, Assumption 1, the sequential randomization assumption, and Assumption 2, the positivity assumption, must hold for the observed data structure including the new A or L nodes.

The TMLE estimates are doubly robust; thus the estimates are consistent if either of the estimates of $\bar{Q}$ or g are consistent. When both are consistent, the TMLE estimates are efficient. The procedures in **ltmle** provide access to data adaptive machine learning algorithms through the **SuperLearner** package, which can improve the chance that $\bar{Q}$ and g are estimated consistently.

In addition to TMLE estimates calculated by default, both the `ltmle` and `ltmleMSM` functions can calculate non-targeted substitution estimates using the `gcomp` argument, or inverse probability of treatment weighted (IPTW) estimates using the `iptw.only` argument. The non-targeted substitution estimates use only an estimate of $\bar{Q}$, and IPTW estimates use only an estimate of g. Since these estimates are only based on either $\bar{Q}$ or g, they are not doubly robust.

Variance and standard error estimates currently provided by the package are based on the estimated influence curve. These estimates are correct asymptotically when estimates of $\bar{Q}$ and g are both consistent, and conservative when only g is estimated consistently. However, in small samples, estimates of variance may be poor. Variance estimation is particularly challenging when the positivity assumption is nearly violated, or, for binary outcomes, when the outcome is very rare. This can lead to inflated type I errors and poor confidence interval coverage. The bootstrap is an alternate choice for variance and confidence interval estimation (Petersen *et al.* 2014). Planned extensions to the package include alternate variance estimates.

Additionally, when a binary outcome is rare, estimation of $\bar{Q}$ is difficult, and final TMLE estimates can have high bias and variance in moderate sample sizes. Another planned addition to the package will provide an alternate method for estimating $\bar{Q}$ in this setting, resulting in less bias and variance in the final TMLE estimates.

Computational details

- R version 3.4.1 (2017-06-30), `x86_64-pc-linux-gnu`
- Locale: `en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8`
- Base packages: **splines**, **methods**, **stats**, **graphics**, **grDevices**, **utils**, **datasets**, **base**
- Other packages: **MASS** 7.3-47, **ggplot2** 2.2.1, **glmnet** 2.0-10, **Matrix** 1.2-10, **foreach** 1.4.3, **ltmle** 0.9-9-3, **SuperLearner** 2.0-22, **nnls** 1.4, **nnet** 7.3-12, **gam** 1.14-4, **knitr** 1.16.4
- Loaded via a namespace (and not attached): **Rcpp** 0.12.11, **magrittr** 1.5, **colorspace** 1.3-2, **lattice** 0.20-35, **plyr** 1.8.3, **tools** 3.4.1, **iterators** 1.0.8, **matrixStats** 0.52.2, **lazyeval** 0.2.0, **tibble** 1.2, **labeling** 0.3, **stringi** 1.1.5, **munsell** 0.4.3, **stringr** 1.2.0, **grid** 3.4.1, **digest** 0.6.12, **codetools** 0.2-15, **compiler** 3.4.1, **speedglm** 0.3-2, **highr** 0.6, **gtable** 0.2.0, **assertthat** 0.2.0, **evaluate** 0.10, **scales** 0.4.1

6. Acknowledgments

This work was supported by NIH Grant # U01 AI069924 (NIAID, NICHD, NCI) (PIs: Egger and Davies) and NIH Grant # R01 AI074345-06 (NIAID) (PI: van der Laan). Maya Petersen is supported by Doris Duke Charitable Foundation Grant # : 2011042.

References

Bang H, Robins JM (2005). "Doubly Robust Estimation in Missing Data and Causal Inference Models." *Biometrics*, **61**(4), 962–73. ISSN 0006-341X. doi:10.1111/j.1541-0420.2005.00377.x.

Friedman J, Hastie T, Tibshirani R (2010). "Regularization Paths for Generalized Linear Models via Coordinate Descent." *Journal of Statistical Software*, **33**(1), 1–22. doi:10.18637/jss.v033.i01.

Gruber S, van der Laan MJ (2010). "A Targeted Maximum Likelihood Estimator of a Causal Effect on a Bounded Continuous Outcome." *The International Journal of Biostatistics*, **6**(1). doi:10.2202/1557-4679.1260.

Gruber S, van der Laan MJ (2012). "**tmle**: An R Package for Targeted Maximum Likelihood Estimation." *Journal of Statistical Software*, **51**(13), 1–35. doi:10.18637/jss.v051.i13.

Hastie T (2017). ***gam**: Generalized Additive Models*. R package version 1.14-4, URL https://CRAN.R-project.org/package=gam.

Hernán MÁ, Brumback B, Robins JM (2000). "Marginal Structural Models to Estimate the Causal Effect of Zidovudine on the Survival of HIV-Positive Men." *Epidemiology*, **11**(5), 561–570. doi:10.1097/00001648-200009000-00012.

Lendle SD, Subbaraman MS, van der Laan MJ (2013). "Identification and Efficient Estimation of the Natural Direct Effect among the Untreated." *Biometrics*, pp. 1–8. ISSN 1541-0420. doi:10.1111/biom.12022.

Moore KL, van der Laan MJ (2009). "Covariate Adjustment in Randomized Trials with Binary Outcomes: Targeted Maximum Likelihood Estimation." *Statistics in Medicine*, **28**(1), 39–64. doi:10.1002/sim.3445.

Neugebauer R, van der Laan M (2007). "Nonparametric Causal Effects Based on Marginal Structural Models." *Journal of Statistical Planning and Inference*, **137**(2), 419–434. doi:10.1016/j.jspi.2005.12.008.

Pearl J (1995). "Causal Diagrams for Empirical Research." *Biometrika*, **82**(4), 669–688. doi:10.1093/biomet/82.4.669.

Pearl J (2000). *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge.

Pearl J (2009). "Causal Inference in Statistics: An Overview." *Statistics Surveys*, **3**, 96–146. doi:10.1214/09-ss057.

Petersen M, Schwab J, Gruber S, Blaser N, Schomaker M, van der Laan M (2014). "Targeted Maximum Likelihood Estimation for Dynamic and Static Longitudinal Marginal Structural Working Models." *Journal of Causal Inference*, **2**(2). doi:10.1515/jci-2013-0007.

Polley E, LeDell E, van der Laan M (2017). ***SuperLearner***: *Super Learner Prediction*. R package version 2.0-22, URL https://CRAN.R-project.org/package=SuperLearner.

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Ripley B (2016). ***nnet***: *Feed-Forward Neural Networks and Multinomial Log-Linear Models*. R package version 7.3-12, URL https://CRAN.R-project.org/package=nnet.

Robins J (1986). "A New Approach to Causal Inference in Mortality Studies with a Sustained Exposure Period – Application to Control of the Healthy Worker Survivor Effect." *Mathematical Modelling*, **7**(9), 1393–1512. doi:10.1016/0270-0255(86)90088-6.

Robins J, Orellana L, Rotnitzky A (2008). "Estimation and Extrapolation of Optimal Treatment and Testing Strategies." *Statistics in Medicine*, **27**(23), 4678–4721. doi:10.1002/sim.3301.

Robins JM (1998). "Marginal Structural Models." In *1997 Proceedings of the American Statistical Association, Section on Bayesian Statistical Science*, pp. 1–10.

Robins JM (2000). "Robust Estimation in Sequentially Ignorable Missing Data and Causal Inference Models." In *Proceedings of the American Statistical Association Section on Bayesian Statistical Science*, volume 1999, pp. 6–10.

Robins JM (2002). "Commentary on 'Using Inverse Weighting and Predictive Inference to Estimate the Effects of Time-Varying Treatments on the Discrete-Time Hazard'." *Statistics in Medicine*, **21**(12), 1663–1680. doi:10.1002/sim.1110.

Schnitzer ME, Moodie EEM, van der Laan MJ, Platt RW, Klein MB (2013). "Modeling the Impact of Hepatitis C Viral Clearance on End-Stage Liver Disease in an HIV Co-Infected Cohort with Targeted Maximum Likelihood Estimation." *Biometrics*, **70**(1), 144–152. doi:10.1111/biom.12105.

Schwab J, Lendle S, Petersen M, van der Laan M (2017). ***ltmle**: Longitudinal Targeted Maximum Likelihood Estimation*. R package version 1.0-0, URL https://CRAN.R-project.org/package=ltmle.

van der Laan MJ, Gruber S (2012). "Targeted Minimum Loss Based Estimation of Causal Effects of Multiple Time Point Interventions." *The International Journal of Biostatistics*, **8**(1). ISSN 1557-4679. doi:10.1515/1557-4679.1370.

van der Laan MJ, Polley EC, Hubbard AE (2007). "Super Learner." *Statistical Applications in Genetics and Molecular Biology*, **6**(1), 1–21. doi:10.2202/1544-6115.1309.

van der Laan MJ, Rose S (2011). *Targeted Learning: Causal Inference for Observational and Experimental Data*. Springer-Verlag, New York. doi:10.1007/978-1-4419-9782-1.

van der Laan MJ, Rubin D (2006). "Targeted Maximum Likelihood Learning." *The International Journal of Biostatistics*, **2**(1). ISSN 1557-4679. doi:10.2202/1557-4679.1043.

Wickham H (2009). ***ggplot2**: Elegant Graphics for Data Analysis*. Springer-Verlag, New York.

Zheng W, van der Laan MJ (2012). "Targeted Maximum Likelihood Estimation of Natural Direct Effects." *The International Journal of Biostatistics*, **8**(1). ISSN 1557-4679. doi:10.2202/1557-4679.1361.

Affiliation:

Samuel D. Lendle
Group in Biostatistics University of California, Berkeley
Berkeley, CA 94720, United States of America
E-mail: lendle@berkeley.edu

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
October 2017, Volume 81, Issue 1 *Submitted:* 2013-12-31
doi:10.18637/jss.v081.i01 *Accepted:* 2016-07-20