concrete: An R Package for Continuous-Time, Competing Risks Targeted Maximum Likelihood Estimation

by David Chen, Helene C. W. Rytgaard, Edwin Fong, Jens M. Tarp, Maya L. Petersen, Mark J. van der Laan, and Thomas A. Gerds

Abstract This article introduces the R package **concrete** which implements a recently developed targeted maximum likelihood-based estimator (TMLE) targeting the cause-specific absolute risks of time-to-event outcomes measured in continuous time. This package can be used to estimate the effects of static and dynamic interventions on a binary treatment given at baseline, quantified as causally-interpretable absolute risks, risk differences, and risk ratios. Cause-specific hazards are estimated by cross-validated Super Learner ensembles of Cox regressions, which are then used to compute g-formula plug-in and TMLE point estimates of absolute risks. Influence curve-based asymptotic inference is provided for TMLE estimates and simultaneous confidence bands can be computed for target estimands that span multiple multiple times or events. In this paper we review one-step continuous-time TMLE methodology as it is situated in the larger causal inference targeted learning workflow, describe how it is implemented in **concrete**, and demonstrate its use on the PBC dataset.

Introduction

In biomedical applications evaluating treatment effects on time-to-event outcomes, study subjects are often susceptible to competing risks such as all-cause mortality. In recent decades, several competing risk methods have been developed; including the Fine-Gray subdistributions model (Fine and Gray, 1999), cause-specific Cox regression (Benichou and Gail, 1990), pseudovalue (Klein and Andersen, 2005), and direct binomial (Scheike et al., 2008; Gerds et al., 2012) regressions; and authors have consistently cautioned against the use of standard survival estimands for causal questions involving competing risks. Nevertheless, reviews of clinical literature (Koller et al., 2012; Austin and Fine, 2017) found that most trials still fail to adequately address the effect of potential competing risks in their studies. Meanwhile, formal causal inference frameworks (Rubin, 1974; Pearl et al., 2016) gained recognition for their utility in translating clinical questions into statistical analyses and the targeted maximum likelihood estimation (TMLE) (Laan and Rose, 2011, 2018) methodology developed from the estimating equation and one-step estimator lineage of constructing semi-parametric efficient estimators through solving efficient influence curve (EIC) equations. The targeted learning roadmap (Petersen and van der Laan, 2014) combines these developments into a cohesive causal inference workflow and provides a structured way to think about statistical decisions. In this paper we apply the targeted learning roadmap to an analysis of time-to-event outcomes and demonstrate the R package **concrete**, which implements a recently developed continuous-time TMLE targeting cause-specific absolute risks (Rytgaard and van der Laan, 2021, 2022; Rytgaard et al., 2023).

Given identification and regularity assumptions, **concrete** can be used to efficiently estimate the treatment effect of interventions given at baseline. In short, the implemented one-step TMLE procedure consists of three stages: 1) an initial estimation of nuisance parameters, 2) a targeted update of the initial estimators to solve the estimating equation corresponding to the target statistical estimand's efficient influence curve (EIC) (Laan and Dudoit, 2003; Kennedy, 2016), and 3) a plug-in of the updated estimators into the original parameter mapping to produce a substitution estimator of the target estimand.

In **concrete** the initial nuisance parameter estimation is performed using Super Learning, a cross-validated machine-learning ensemble algorithm with asymptotic oracle guarantees (Laan and Dudoit, 2003; Laan et al., 2007; Polley et al., 2021); flexible machine-learning approaches such as Super Learners with robust candidate libraries and appropriate loss functions often give users the best chance of achieving the convergence rates needed for TMLE's asymptotic properties. The subsequent targeted update is based in semi-parametric efficiency theory; specifically that efficient regular and asymptotically linear (RAL) estimators must have influence curves equal to the efficient influence curve (EIC) of the target statistical estimand, see e.g. (Laan and Rose, 2011). This update step shifts initial nuisance parameter estimates to solve the estimating equation corresponding to the target EIC, thus recovering normal asymptotic inference (given that the initial estimators converge adequately quickly) while leveraging the power of flexible machine-learning algorithms for initial estimation. In Section 2.2.3 we outline how Super Learner is used to estimate nuisance parameters in **concrete** while more detailed guidance on how to best specify Super Learner estimators is provided in (Phillips et al.,

2022). Section **??** details the subsequent targeted update, with a full description provided in (Rytgaard and van der Laan, 2021).

Currently **concrete** can be used for estimands derived from cause-specific absolute risks, such as risk ratios and risk differences, under static and dynamic interventions on binary treatments given at baseline. Estimands can be jointly targeted at multiple times, up to full risk curves over an interval, and for multiple events in cases with competing risks. Methods are available to handle right censoring, competing risks, and confounding by baseline covariates. Point estimates can be computed using g-formula plug-in or one-step TMLE, and asymptotic inference for the latter is derived from the variance of the efficient influence curve (EIC).

concrete is not intended to be used for data with clustering, left trunctation (i.e. delayed entry) or interval censoring. Currently the Super Learners for estimating conditional hazards must be comprised of Cox regressions although the incorporation of penalized Cox (coxnet) and hazard estimators based on highly adaptive lasso (HAL) are planned in future package versions. Support for stochastic interventions and interventions on multinomial and continuous treatments are also forthcoming, while longitudinal methods to handle time-dependent treatment regimes and time-dependent confounding are in longer term development.

A concrete example: analyzing the PBC dataset

Below we illustrate the usage of **concrete** with the well-known Mayo Clinic Primary Biliary Cholangitis (PBC) data set (Fleming and Harrington, 1991; Therneau and Grambsch, 2000). We estimate the cause-specific counterfactual absolute risk differences, i.e. average treatment effects, under two levels of a binary treatment (randomization to placebo or D-penicillamine). The treatment column `"trt"` is transformed so that 0 indicates placebo and 1 indicates D-penicillamine, and where the two competing events are transplant (`"status"=1`) and death (`"status"=2`) in the presence of right censoring (`"status"=0`). We include outcomes for two estimators, g-computation plug-in and TMLE, as well as point-wise confidence intervals based on the estimated influence curve.

```
# Prepare Data
library(concrete)
data <- survival::pbc[, c("time", "status", "trt", "age", "sex", "albumin")]
data <- subset(data, subset = !is.na(data$trt))
data$trt <- data$trt - 1

# Specify Analysis
ConcreteArgs <- formatArguments(
    DataTable = data,
    EventTime = "time", # name of event time variable
    EventType = "status", # name of event status variable
    Treatment = "trt", # name of treatment variable
    Intervention = 0:1, # 2 different static interventions
    TargetTime = 365.25/2 * (6:16), # 11 target times: 3-8 years biannually
    TargetEvent = 1:2, # 2 different competing risks
    CVArg = list(V = 10)
)

# Compute
ConcreteEst <- doConcrete(ConcreteArgs)

# Return Output
ConcreteOut <- getOutput(ConcreteEst, Estimand = "RD", Simultaneous = FALSE)
plot(ConcreteOut, ask = FALSE)
```

Other packages

concrete is the first R package for implementing a continuous-time TMLE for survival and competing risk estimands, and thus extends a group of existing R packages implementing semi-parametric efficient estimators for time-to-event-outcomes. The **ltmle** (Schwab et al., 2020), **stremr** (Sofrygin et al., 2017), and **survtmle** (Benkeser and Hejazi, 2019) implement discrete-time TMLEs for survival estimands and can either natively or can be adapted to perform discrete-time TMLEs for right censored survival or competing risks estimands. Notably these packages all operate in discrete-time and would thus require discretization of continuous-time data, which if not performed carefully can negatively

impact both causal inference and estimator performance(Sofrygin et al., 2019; Ferreira Guerra et al., 2020; Sloma et al., 2021). **ltmle** and **stremr** use the method of iterated expectations while **survtmle** can target the hazard-based survival formulation.

In addition, the *Causal Inference* CRAN Task View lists **riskregression** (Gerds et al., 2022) as estimating treatment effect estimands in survival settings. **riskregression** implements the IPTW and double-robust AIPTW estimators. None of the packages listed on the *Survival* CRAN Task View are described as implementing efficient semi-parametric estimators, though available via Github are the R packages adjustedCurves (Denz et al., 2022) and CFsurvival (Westling et al., 2021), which implement the AIPTW and a cross-fitted doubly-robust estimator respectively.

Structure of this manuscript

This article is written for readers wishing to use the **concrete** package for their own analyses and for readers interested in an applied introduction to the one-step continuous-time TMLE method described in (Rytgaard and van der Laan, 2021). The [Targeted Learning][sec-targeted-learning] section outlines the targeted learning approach to time-to-event causal effect estimation, with subsection Estimation providing details on the implemented one-step TMLE. Usage of the **concrete** package and its features is then provided in [Using Concrete][sec-using-concrete], using the example of a simple competing risks analysis of the PBC dataset.

The Targeted Learning framework for survival analysis

At a high level, the targeted learning roadmap for analyzing continuous-time survival or competing risks consists of:

1. Specifying the causal model and defining a causal estimand (e.g. causal risk difference). Considerations include defining a time zero and time horizon, specifying the intervention (i.e., treatment) variable and the desired interventions (including on sources of right censoring), and specifying the target time(s) and event(s) of interest.
2. Defining a statistical model and statistical estimand, and evaluating the assumptions necessary for the statistical estimand to identify the causal estimand. Considerations include identifying confounding variables, establishing positivity for desired interventions, and formalizing knowledge about the statistical model (e.g. dependency structures or functional structures).
3. Performing estimation and providing inference. Considerations include pre-specification of an estimator and an inferential approach with desirable theoretical properties (e.g. consistency and efficiency within a desired class), and assessing via outcome-blind simulations the estimator's robustness and suitability for the data at hand.

In the following sections we discuss these three stages in greater detail.

The causal model: counterfactuals, interventions, and causal estimands

With time-to-event data, typical counterfactual outcomes are how long it would take for some event(s) to occur if subjects were hypothetically to receive some intervention, i.e. treatment. Let A be the treatment variable and let d be the hypothetical intervention rule of interest, i.e., the function that assigns treatment levels to each subject. The simplest interventions are static rules setting A to some value a in the space of treatment values $\mathcal{A}$, while more flexible dynamic treatment rules might assign treatments based on subjects' baseline covariates (which we denote as W), and stochastic treatment rules incorporate randomness and may even depend on the natural treatment assignment mechanism in so-called modified treatment policies. Additionally, our goal in time-to-event analyses is often to assess the causal effect of some treatment rule d on an event (or set of competing events) *in the absence of right censoring*. This "absence of right censoring" condition is in fact a static intervention to deterministically prevent right censoring, and is an implicit component to many interventions in time-to-event analyses.

Regardless of the type of intervention rule, the associated counterfactual survival data under intervention rule d, $X \sim P^d$, takes the general form

$$X = \left(T^d, \Delta^d, A^d, W\right) \tag{1}$$

where $T^d \in (0, t_{max}]$ is the counterfactual time-to-event under intervention d for the earliest of J competing events up to some maximum follow-up time t_{max}, $\Delta^d \in \{1, \ldots, J\}$ is the counterfactual

event index indicating which the J events would have hypothetically occurred first, and A^d is the treatment variable under intervention d (which for static and dynamic interventions will be a degenerate variable). Note that we differentiate between competing events (indexed $1, ..., J$) and sources of right censoring (not present in X), as our goal is to assess the causal effect of treatment rule d on the set of competing events in the absence of right censoring. For ideal experiments tracking just one event, i.e. $J = 1$, the causal setting is one of survival of a single risk; if instead mutually exclusive events would be allowed to compete, then the causal setting is one with competing risks.

With the counterfactual data defined, causal estimands can then be specified as functions of the counterfactual data. For instance, if we were interested in effects of interventions d_0 versus d_1 on time-to-event outcomes, the counterfactual data $\widetilde{X} \sim P^{0,1}$ might be represented as

$$\widetilde{X} = \left(T^{d_0},\, \Delta^{d_0},\, A^{d_0}, T^{d_1},\, \Delta^{d_1},\, A^{d_1}, W\right)$$

We could then define estimands such as the causal event j relative risks at time t

$$\tilde{\Psi}_{j,t}(P^{0,1}) = \frac{P(T^{d_1} \leq t, \Delta^{d_1} = j)}{P(T^{d_0} \leq t, \Delta^{d_0} = j)} \tag{2}$$

These estimands may be of interest at a single timepoint, at multiple timepoints, or over a time interval, and in the case of competing risks may involve multiple events (e.g. $\tilde{\Psi}_{j,t}(P^{0,1}) : t \in (0, t_{max}),\, j \in 1, \ldots, J$). In any case, once the desired causal quantity of interest has been expressed as a function of the counterfactual data, efforts can then be made to identify the causal estimand with a function of observed data, i.e. a statistical estimand.

Statistical model: observed data, identification, and statistical estimands

Observed time-to-event data $O \sim P_0$ with J competing events can be represented as:

$$O = \left(\widetilde{T},\, \tilde{\Delta},\, A,\, W\right) \tag{3}$$

where $\widetilde{T} \in (0, t_{max}]$ is the earlier of the first event time T or the right censoring time C, $\tilde{\Delta} \in \{0, \ldots, J\}$ indicates which event occurs (with 0 indicating right censoring), A is the observed treatment and W is the set of baseline covariates.

To link causal estimands such as Eq. (2) to statistical estimands, we need a set of identification assumptions to hold, informally: consistency, positivity, and conditional exchangeability (or their structural causal model analogs) . Readers can find a full discussion of these identification assumptions for absolute risk estimands in Section 3 of (Rytgaard and van der Laan, 2022). Given these assumptions, we can identify the cause-j absolute risk at time t under intervention d using the g-computation formula (Robins, 1986) as

$$P(T^d \leq t, \Delta^d = j) = \mathbb{E}_{\mathcal{W}}\left[\int_{\mathcal{A}} F_j(t|\, a,\, w)\, \pi^*(a \mid w)\, da\right] \tag{4}$$

where $\pi^*(a \mid w)$ is the treatment propensity implied by the intervention d. Here $F_j(t|\, a,\, w)$ is the conditional cause-j absolute risk

$$F_j(t|\, a,\, w) = \int_0^t \lambda_j(s|\, a,\, w)\, S(s\text{-}|\, a,\, w)\, ds\ ,$$

where the cause-j conditional hazard function λ_j is defined as

$$\lambda_j(t|\, a,\, w) = \lim_{h \to 0} \frac{1}{h} P(\widetilde{T} \leq t + h,\, \tilde{\Delta} = j \mid \widetilde{T} \geq t,\, a,\, w)\ ,$$

and the conditional event-free survival probability is given by

$$S(t|\, a,\, w) = \exp\left(-\int_0^t \sum_{j=1}^{J} \lambda_j(s|\, a,\, w)\, ds\right). \tag{5}$$

From Eq (4), it follows that we can identify the causal cause-j relative risk (2) at time t by

$$\Psi_{F_{j,t}}(P_0) = \frac{\mathbb{E}_{\mathcal{W}}\left[\int_{\mathcal{A}} F_j(t|\, a,\, w)\, \pi^*_{d_1}(a \mid w)\, da\right]}{\mathbb{E}_{\mathcal{W}}\left[\int_{\mathcal{A}} F_j(t|\, a,\, w)\, \pi^*_{d_0}(a \mid w)\, da\right]} \tag{6}$$

where $\pi^*_{d_0}$ and $\pi^*_{d_1}$ represent the treatment propensities implied by treatment rules d_0 and d_1 respec-

tively.

It should be noted that even without the identification assumptions for causal inference, statistical estimands such as Eq. (6) may still have valuable interpretations as standardized measures isolating the importance of the "intervention" variable (Laan, 2006).

Targeted estimation

The TMLE procedure for estimands derived from cause-specific absolute risks begins with estimating the treatment propensity π, the conditional hazard of censoring λ_c and the conditional hazards of events $\lambda_j : j = 1, \ldots, J$. In **concrete** these nuisance parameters are estimated using the Super Learner algorithm, which involves specifying a cross-validation scheme, compiling a library of candidate algorithms, and designating a cross-validation loss function and a Super Learner meta-learner.

Specifying Super Learners

For a simple V-fold cross-validation setup, let $Q_n = \{O_i\}_{i=1}^n \sim P_n$ be the observed n i.i.d observations of $O \sim P_0$ and let $B_n \in \{1, ..., V\}^n$ be a random vector that assigns the n observations into V validation folds. Then for each v in $1, ..., V$ we define a training set $Q_v^{\mathcal{T}} = \{O_i : B_n^i = v\} \sim P_v^{\mathcal{T}}$ and corresponding validation set $Q_v^{\mathcal{V}} = \{O_i : B_n^i \neq v\} \sim P_v^{\mathcal{V}}$.

Having specified a cross-validation scheme, the next steps are to construct the Super Learner candidate library, define an appropriate loss function, and select a Super Learner meta-learner. Super Learner libraries should be comprised of candidate algorithms that range in flexibility while respecting existing data-generating knowledge. For instance, candidate estimators should incorporate incorporate domain knowledge regarding covariates and interactions are predictive of outcomes. If the number independent observations n is small compared to the number of covariates, then Super Learner libraries should contain fewer candidates and either incorporate native penalization, e.g. regularized Cox regression (`coxnet`), or be paired with covariate screening algorithms (Phillips et al., 2022). If, on the other hand, the number of independent observations is large compared to the number of covariates, then Super Learner libraries can include more algorithms including highly flexible non-parametric algorithms such as highly adaptive lasso (HAL). It should be noted that using HAL for initial nuisance parameter estimation can achieve the necessary convergence rates (Laan, 2017; Bibaut and van der Laan, 2019; Rytgaard et al., 2023) for TMLE to be efficient. Super Learner loss functions should imply a risk that is minimized by the true data-generating process and define a loss-based dissimilarity tailored to the target parameter and a discrete selector that selects the best performing candidate should be used as the Super Learner meta-learner (Laan et al., 2007; Phillips et al., 2022). If computationally feasible, Super Learners using more flexible meta-learner algorithms can safely be nested as candidates within a larger Super Learner using a discrete meta-learner. Additional guidance on Super Learner specification is provided in (Phillips et al., 2022) and Chapter 3 of (Laan and Rose, 2011).

Currently the default cross-validation setup in **concrete** follows the guidelines laid out in (Phillips et al., 2022; Dudoit and van der Laan, 2005; Vaart et al., 2006), with the number of cross-validation folds increasing with decreased sample size. The default number of folds ranges from leave-one-out cross validation (LOOCV) for datasets with fewer than 30 independent observations to 2-fold cross validation for datasets with over 10000 independent observations. Default Super Learner libraries are provided and will be detailed in the following sections, but should be amended to suit the data at hand and to incorporate subject matter knowledge.

Estimating Treatment Propensity

Let π_0 be the true conditional distribution of A given W (i.e. the treatment propensity), let $\mathcal{M}_\pi = \{\hat{\pi} : P_n \to \hat{\pi}(P_n)\}$ be the library of candidate propensity score estimators, and let L_π be a loss function such that the risk $\mathbb{P}_0 L_\pi(\pi) \equiv \mathbb{E}_0 [L_\pi(\pi, O)]$ is minimized by π_0. The discrete Super Learner estimator is then the candidate propensity estimator with minimal cross validated risk,

$$\hat{\pi}^{SL} = \underset{\hat{\pi} \in \mathcal{M}_\pi}{\operatorname{argmin}} \sum_{v=1}^{V} \mathbb{P}_{Q_v^{\mathcal{V}}} L_\pi(\hat{\pi}(P_v^{\mathcal{T}})) \tag{7}$$

where $\hat{\pi}(P_v^{\mathcal{T}})$ are candidate propensity score estimators trained on data $Q_v^{\mathcal{T}}$. Currently **concrete** uses default **SuperLearner** loss functions (non-negative least squares) and specifies a default library consisting of glmnet (Friedman et al., 2010) and xgboost (Chen and Guestrin, 2016).

Estimating Conditional Hazards

For $\delta = 0, \ldots, J$ where ($\delta = 0$) is censoring and ($\delta \in \{1, \ldots, J\}$) are outcomes of interest, let $\lambda_\delta : \delta = 0, \ldots, J$ be the true conditional hazards, let $\mathcal{M}_\delta = \{\hat{\lambda}_\delta : P_n \to \hat{\lambda}_\delta(P_n)\}$ be the libraries of candidate estimators, and let L_δ be loss functions such that the risks $\mathbb{P}_0 L_\delta(\cdot)$ are minimized by the true conditional hazards λ_δ. The discrete Super Learner selectors for each δ then chooses the candidate which has minimal cross validated risk

$$\hat{\lambda}_\delta^{SL} = \underset{\hat{\lambda}_\delta \in \mathcal{M}_\delta}{\operatorname{argmin}} \sum_{v=1}^{V} \mathbb{P}_{Q_v^\mathcal{V}} L_\delta(\hat{\lambda}_\delta(P_v^\mathcal{T})) \ : \ \delta = 0, \ldots, J \tag{8}$$

where $\hat{\lambda}_\delta(P_v^\mathcal{T})$ are candidate event δ conditional hazard estimators trained on data $Q_v^\mathcal{T}$. The current **concrete** default is a library of two Cox models, treatment-only and main-terms, with cross-validated risk computed using negative log Cox partial-likelihood loss (Cox, 1975; Rytgaard and van der Laan, 2022)

$$\mathbb{P}_{Q_v^\mathcal{V}} L_\delta(\hat{\lambda}_\delta(P_v^\mathcal{T})) = \mathbb{P}_{Q_v^\mathcal{V}} L_\delta(\hat{\beta}_{\delta,Q_v^\mathcal{T}}) = - \sum_{i:\, O_i \in Q_v^\mathcal{V}} \left[\hat{\beta}'_{\delta,Q_v^\mathcal{T}} W_i - \log \left[\sum_{i:\, O_i \in Q_v^\mathcal{V}} \mathbb{1}\left(\widetilde{T}_l \geq \widetilde{T}_i\right) \exp(\hat{\beta}'_{\delta,Q_v^\mathcal{T}} W_h) \right] \right]$$

where $\mathcal{R}(t)$ is the risk set at time t, $\{h : \widetilde{T}_h \geq t\}$ and $\hat{\beta}_{\delta,Q_v^\mathcal{T}}$ are the coefficients of an event δ candidate Cox regression trained on data $Q_v^\mathcal{T}$.

Solving the Efficient Influence Curve Equation

For parameters such as risk ratios which are derived from cause-specific absolute risks, we solve a vector of absolute risk efficient influence curve (EIC) equations with one element for each combination of target event, target time, and intervention. That is, the EIC for a target parameter involving J competing events, K target times, and M interventions is a $J \times K \times M$ dimensional vector where the component corresponding to the cause-specific risk of event j, at time t_k, and under intervention propensity $\pi^*{}_m$ is:

$$\begin{aligned} D^*_{m,j,k}(\lambda, \pi, S_c)(O) = \sum_{l=1}^{J} \int h_{m,j,k,l,s}(\lambda, \pi, S_c)(O) \left(N_l(s) - \mathbf{1}(\widetilde{T} \geq s)\, \lambda_l(s|\ A,\ W) \right) ds \\ + \int_\mathcal{A} F_j(t_k \mid A = a, W)\, \pi^*{}_m(a \mid W)\, da - \Psi_{\pi^*,j,t}(P_0) \end{aligned} \tag{9}$$

where $N_l : l = 0, \ldots, J$ are the cause-specific counting processes

$$N_l(s) = \mathbf{1}\left\{ \widetilde{T} \leq s, \tilde{\Delta} = l \right\}$$

and $h_{m,j,k,l,s}(\lambda, \pi, S_c)(O)$ is the TMLE "clever covariate"

$$h_{m,j,k,l,s}(\lambda, \pi, S_c)(O) = \frac{\pi^*{}_m(A \mid W)\, \mathbf{1}(s \leq t_k)}{\pi(A \mid W)\ S_c(s\text{-}|\ A,\ W)} \left(\mathbf{1}(l = j) - \frac{F_j(t_k|\ A,\ W) - F_j(s|\ A,\ W)}{S(s|\ A,\ W)} \right) \tag{10}$$

We highlight here that the clever covariate is a function of the {intervention-defined treatment propensity}, the {observed intervention-related densities} (i.e. the observed treatment propensity and cumulative conditional probability of remaining uncensored) which are unaffected by TMLE targeting, and the {observed outcome-related densities} which will be updated by TMLE targeting. Note also that our notation for the EIC ($D^*_{m,j,k}(\lambda, \pi, S_c)(O)$) and clever covariate ($h_{m,j,k,l,s}(\lambda, \pi, S_c)(O)$) reflects the dependence on P through the cause-j conditional hazards $\lambda = (\lambda_l \ : \ l = 1, \ldots, J)$ and the treatment propensity π and conditional censoring survival $S_c(t|\ a,\ w) = \exp\left(- \int_0^t \lambda_0(s|\ a,\ w)\, ds \right)$.

The one-step continuous-time survival TMLE (Rytgaard and van der Laan, 2021) involves updating the cause-specific hazards λ along the universally least favorable submodel, which is implemented as recursive limited updates along a sequence of locally least favorable submodels. To describe this procedure, let us first introduce the following vectorized notation:

$$D^* = \left(D^*_{m,j,k} : m = 1, \ldots, M,\ j = 1, \ldots, J,\ k = 1, \ldots, K \right)$$
$$h_{l,s} = \left(h_{m,j,k,l,s} : m = 1, \ldots, M,\ j = 1, \ldots, J,\ k = 1, \ldots, K \right)$$

The one-step continuous-time survival TMLE recursively updates the cause-specific hazards in the

following manner: starting from $b = 0$, with $\lambda_j^0 = \hat{\lambda}_j^{SL}$, and $\lambda^b = \left(\lambda_l^b \ : \ l = 1, \ldots, J\right)$

$$\lambda_l^{b+1} = \lambda_l^b \exp\left(\frac{\left\langle \mathbb{P}_n D^*(\lambda^b, \pi, S_c)(O),\ h_{j,s}(\lambda^b, \pi, S_c)(O)\right\rangle}{||\mathbb{P}_n D^*(\lambda^b, \pi, S_c)(O)||}\epsilon_b\right), \quad l = 1, \ldots, J \tag{11}$$

where

$$\langle x, y\rangle = x^\top y \quad , \quad ||x|| = \sqrt{x^\top x}$$

and the step sizes ϵ_b are chosen such that

$$||\mathbb{P}_n D^*(\lambda^{b+1}, \pi, S_c)(O)|| < ||\mathbb{P}_n D^*(\lambda^b, \pi, S_c)(O)||$$

The recursive update following Eq (11) is completed at the iteration B where

$$\mathbb{P}_n D^*(\lambda^B, \pi, S_c)(O) \leq \frac{\sqrt{\mathbb{P}_n D^*(\lambda^B, \pi, S_c)(O)^2}}{\sqrt{n}\ \log(n)} \tag{12}$$

This updated vector of conditional hazards λ^B is then used to compute a plug-in estimate of the statistical estimand simultaneously across causes, target times, and interventions.

Estimating TMLE variance

In **concrete**, the variance of the TMLE estimates is estimated based on the plug-in estimate of the sample variance of the EIC, $\frac{\mathbb{P}_n\ D^*(\hat{\lambda}^B, \hat{\pi}, \hat{S}_c)(O)^2}{n}$, which is a consistent estimator for the variance of the TMLE when all nuisance parameter estimators are consistent. In the presence of practical positivity violations arising from sparsity in finite samples (discussed further in the section on the [ConcreteEst object][sec-concreteest]), the EIC-based variance estimator can be anti-conservative and variance estimation by bootstrap may be more reliable (Tran et al., 2018). However, bias resulting from positivity violations cannot be remedied in this way, and so other methods of addressing positivity violations are recommended instead (Petersen et al., 2012). For multidimensional estimands, simultaneous confidence intervals can be computed by simulating the $1 - \alpha$ quantiles of a multivariate normal distribution with the covariance structure of the estimand EICs.

Using concrete

The basic **concrete** workflow consists of sequentially calling the following three functions:

- `formatArguments()`
- `doConcrete()`
- `getOutput()`.

Users specify their estimation problem and desired analysis through `formatArguments()`, which checks the specified analysis for red flags and then produces a `"ConcreteArgs"` environment object. The `"ConcreteArgs"` object is then passed into `doConcrete()` which performs the specified continuous-time one-step survival TMLE and produces a `"ConcreteEst"` object which can be interrogated for diagnostics and estimation details. The `"ConcreteEst"` object can then be passed into `getOutput()` to produce tables and plots of cause-specific absolute risk derived estimands such as risk differences and relative risks.

formatArguments()

The arguments of `formatArguments()` are primarily involved in specifying the observed data structure, the target estimand, and the TMLE estimator. The output, a `"ConcreteArgs"` object, contains the necessary elements of a continuous-time TMLE analysis.

```
ConcreteArgs <- formatArguments(
    DataTable = data,
    EventTime = "time", # name of event time variable
    EventType = "status", # name of event status variable
    Treatment = "trt", # name of treatment variable
```

```
    Intervention = 0:1, # 2 different static interventions
    TargetTime = 365.25/2 * (6:16), # 11 target times: 3-8 years biannually
    TargetEvent = 1:2, # 2 different competing risks
    CVArg = list(V = 10)
)
```

Data

The observed data are passed into the `DataTable` argument as either a `data.frame` or `data.table` object, which must contain columns corresponding to the observed time-to-event $\widetilde{T}$, the indicator of which event occured Δ, and the treatment variable A. Treatment values in A must be numeric, with binary treatments encoded as 0 and 1, and if the dataset contains a column with uniquely identifying subject IDs, its name should be passed into the `ID` argument. Any number of columns containing baseline covariates W can also be included, but the data must be without missingness; imputation of missing covariates should be done prior to passing data into **concrete** while data with missing treatment or outcome values is not supported by **concrete**. % ID is for compatibility with planned future functionality for clustered and longitudinal data.

In the PBC example, the observed data is the `data` object, $\widetilde{T}$ is the column `"time"`, Δ is the column `"status"`, A is the column `"trt"`, and covariates L are the remaining columns: (`"age"`, `"sex"`, and `"albumin"`).

Target estimand: intervention, target events, and target times

Static interventions on a binary treatment A setting all observations to $A = 0$ or $A = 1$ can specified with 0, 1, or c(0, 1) if both interventions are of interest, i.e. for contrasts such as risk ratios and risk differences. More complex interventions can be specified with a list containing a pair of functions: an "intervention" function which outputs desired treatment assignments and a "g.star" function which outputs desired treatment probabilities. Interventions depending on baseline covariates can be passed in as "intervention" functions without an accompanying "g.star" function. These functions can take treatment and covariates as arguments and must produce treatment assignments and probabilities respectively, each with the same dimensions as the observed treatment. The function `makeITT()` creates list of functions corresponding to the binary treat-all and treat-none static interventions, which can be used as a template for specifying more complex interventions.

The `TargetEvent` argument specifies the event types of interest. Event types must be be coded as integers, with non-negative integers reserved for censoring. If `TargetEvent` is left `NULL`, then all positive integer event types in the observed data will be jointly targeted. In the `pbc` dataset, there are 3 event values encoded by the`status` column: 0 for censored, 1 for transplant, and 2 for death. To analyze `pbc` with transplants treated as right censoring, `TargetEvent` should be set to 2, whereas for a competing risks analysis one could either leave `TargetEvent = NULL` or set `TargetEvent = 1:2` as in the above example.

The `TargetTime` argument specifies the times at which the cause-specific absolute risks or event-free survival are estimated. Target times should be restricted to the time range in which target events are observed and `formatArguments()` will return an error if target time is after the last observed failure event time. If no `TargetTime` is provided, then **concrete** will target the last observed event time, though this is likely to result in a highly variable estimate if prior censoring is substantial. The `TargetTime` argument can either be a single number or a vector, as one-step TMLE can target cause-specific risks at multiple times simultaneously. % For estimands involving full curves, `TargetTime=` should be set to a fine grid covering the desired interval (Rytgaard and van der Laan, 2021).

Estimator specification

The `formatArguments()` function can be used to modify the estimation procedure. The arguments of `formatArguments()` involved in estimation are the cross-validation setup `CVArg`, the Superlearner candidate libraries `Model`, the software backends `PropScoreBackend` and `HazEstBackend`, and the practical TMLE implementation choices `MaxUpdateIter`, `OneStepEps`, and `MinNuisance`. Note that `Model` is used in this section in line with common usage in statistical software, rather than to refer to formal statistical or causal models as in preceding sections.

Cross-validation is implemented using `origami::make_folds()` and using the input of the `CVArg` argument. If no input is provided into `CVArg`, the default cross-validation setup follows the recommendations in (Phillips et al., 2022). Cross-validation folds are stratified by event type and the number of folds ranges from 2 for datasets with greater than 10000 independent observations to LOOCV for

datasets with fewer than 30 independent observations. Chapter 5 of the online Targeted Learning Handbook (Malenica et al.) demonstrates the specification of several other cross-validation schemes.

Super Learner libraries for estimating nuisance parameters are specified through the `Model` argument. The input should be a named list with an element for the treatment variable and one for each event type including censoring as illustrated in the following code example. The list element corresponding to treatment must be named with the column name of the treatment variable, and the list elements corresponding to each event type must be named by the character which corresponds to the numeric value of the event type (e.g. "0" for censoring). Any missing specifications will be filled in with defaults, and the resulting list of libraries can be accessed in the output `.[["Model"]]` and further edited by the user, as shown below.

```
ConcreteArgs$Model <- list(
    "trt" = c("SL.glmnet", "SL.bayesglm", "SL.xgboost", "SL.polymars"),
    "0" = NULL, # will use the default library
    "1" = list(Surv(time, status == 1) ~ trt, Surv(time, status == 1) ~ .),
    "2" = list("Surv(time, status == 2) ~ trt", "Surv(time, status == 2) ~ .")
)
```

In **concrete**, propensity scores are by default estimated using the **SuperLearner** package, obtained by setting `PropScoreBackend = "Superlearner"`, with candidate algorithms `c("xgboost","glmnet")` implemented by packages **xgboost** and **glmnet**. Alternatively the **sl3** package can be used by specifying learners . For further details about these packages, see their respective package documentations at the [Guide to SuperLearner][https://cran.r-project.org/web/packages/SuperLearner/vignettes/Guide-to-SuperLearner.html] and the [Targeted Learning Handbook][https://tlverse.org/tlverse-handbook/sl3.html].

For estimating the necessary conditional hazards, **concrete** currently relies on a discrete Superlearner consisting of a library of Cox models implemented by `survival::coxph()` evaluated on cross-validated partial-likelihood loss as detailed in the [Conditional Hazard][sec-haz] subsection. Support for estimation of hazards using coxnet, Poisson-HAL and other methods may be added in the future, but currently the `HazEstBackend` argument must be "coxph". The default Cox specifications are a treatment-only model and a main-terms model with treatment and all covariates. These models can be specified as strings or formulas as can be seen in the above example.

As detailed by Eq. (11) and (12), the one-step TMLE update step involves recursively updating cause-specific hazards, summing along small steps ϵ_b. At default the maximum step size is 0.1, and is halved persistently whenever a step would increase the mean estimated EIC. The `MaxUpdateIter` argument is used to provide a definite stop to the recursive TMLE update. The default of 500 steps should be sufficient for most applications, but may need to be increased when targeting estimands with many components or for rare events. The presence of practical positivity sparsity can also result in slow TMLE convergence, but increasing `MaxUpdateIter` would not be not an adequate solution there as the resulting TMLE estimates and inference may still be unreliable. The `MinNuisance` argument can be used to specify a lower bound for the product of the propensity score and lagged survival probability for remaining uncensored; this term is present in the denominator of the efficient influence function and enforcing a lower bound decreases estimator variance at the cost of introducing bias but improving stability.

ConcreteArgs

The `"ConcreteArgs"` output of `formatArguments()` is an environment containing the estimation specification as objects that can be modified by the user. The modified `"ConcreteArgs"` object should then be passed back through `formatArguments()` to check the modified estimation specification.

```
# decrease the maximum tmle update number to 50
ConcreteArgs$MaxUpdateIter <- 50

# add a candidate regression with treatment interactions
ConcreteArgs[["Model"]][["2"]][[3]] <- "Surv(time, status == 2) ~ trt*."

# validate new estimation specification
ConcreteArgs <- formatArguments(ConcreteArgs)
```

`"ConcreteArgs"` objects can be printed to display summary information about the specified estimation problem,

```
print(ConcreteArgs, Verbose = TRUE)
```

```
Observed Data (312 rows x 7 cols)
Unique IDs: "ID" (n=312),  Time-to-Event: "time",  Event Type: "status",  Treatment: "trt"

Cens. 0 : n=168 (0.54),  [min,max] = [788, 4556]
Event 1 : n=19 (0.06),  [min,max] = [533, 3092]
Event 2 : n=125 (0.4),  [min,max] = [41, 4191]

Treatment: 0: n=158 (0.51)   1: n=154 (0.49)

3 Baseline Covariates
   ColName CovName CovVal
1:      L1     age      .
2:      L2 albumin      .
3:      L3     sex      f


Target Events: 1, 2
Target Times (n at risk): 1095.75 (240/312), 1278.375 (224/312), 1461 (194/312),  ... , 2556.75 (93/312), 2739.375 (76/312), 2922 (67/312)
Intervention "A=1":  Trt Assignments = (1,1,1,1,1,1,1,1,1,1 ... ),  Observed Prevalence = 0.49
Intervention "A=0":  Trt Assignments = (0,0,0,0,0,0,0,0,0,0 ... ),  Observed Prevalence = 0.51

Stratified 10-Fold Cross Validation
Trt Pr Estimation (SuperLearner): Default SL Selector, Default Loss Fn, 4 candidates - SL.glmnet, SL.ranger, SL.xgboost, SL.glm
Cens. 0 Estimation (coxph): Discrete SL Selector, Log Partial-LL Loss, 2 candidates - TrtOnly, MainTerms
Event 1 Estimation (coxph): Discrete SL Selector, Log Partial-LL Loss, 2 candidates - model1, model2
Event 2 Estimation (coxph): Discrete SL Selector, Log Partial-LL Loss, 3 candidates - model1, model2, model3

One-step TMLE (finite sum approx.) simultaneously targeting all cause-specific Absolute Risks
g nuisance bounds = [0.04929, 1],  max update steps = 50,  starting one-step epsilon = 0.1

****
Cox model specifications have been renamed where necessary to reflect changed covariate names. Model specifications in .[['Model']] can be
checked against the covariate names in attr(.[['DataTable']], 'CovNames')
****
```

In particular, we can see that the specified analysis is for two competing risks (Target Events: 1, 2) under interventions "A=1" and "A=0" assigning all subjects to treated and control arms, respectively. Objects in the `"ConcreteArgs"` environment can be interrogated directly for details about any particular aspect of the estimation specification.

doConcrete()

Adequately specified `"ConcreteArgs"` objects can then be passed into the `doConcrete()` function which will then perform the specified TMLE analysis. The output is an object of class `"ConcreteEst"` which contains TMLE point estimates and corresponding estimated influence curves for the cause-specific absolute risks for each targeted event at each targeted time under each intervention. If the `GComp` argument is set to `TRUE`, then a Super Learner-based g-formula plugin estimate of the targeted risks will be included in the output.

```
ConcreteEst <- doConcrete(ConcreteArgs)
```

We have reviewed the one-step continuous-time TMLE implementation in [Estimation][sec-estimation], so here we will name the non-exported functions in `doConcrete()` which perform each of the steps of the one-step continuous-time survival TMLE procedure, in case users wish to explore the implementation in depth.

The cross-validation ([Cross-Validation][sec-cv]) is checked and evaluated in `formatArguments()`, returning fold assignments as the `.[["CVFolds"]]` element of the `"ConcreteArgs"` object.

The initial estimation of nuisance parameters and is performed by the function `getInitialEstimate()` which depends on `getPropScore()` for propensity scores ([Estimating Propensities][sec-trt-ps]) and `getHazEstimate()` for the conditional hazards ([Estimating Conditional Hazards][sec-haz]).

Computing of EICs is done by `getEIC()` which is used within the `doTmleUpdate()` function which performs the one-step TMLE update procedure (Solving the EIC equation][sec-eic]).

ConcreteEst objects

The print method for `"ConcreteEst"` objects summarizes the estimation target and displays diagnostic information about TMLE update convergence, intervention-related nuisance parameter truncation, and the nuisance parameter Super Learners.

```
print(ConcreteEst, Verbose  = TRUE)
```

```
Continuous-Time One-Step TMLE targeting the Cause-Specific Absolute Risks for:
Interventions: "A=1", "A=0"  |  Target Events: 1, 2  |  Target Times: 1095.75, 1278.375, 1461, ... ,2556.75, 2739.375, 2922

**TMLE did not converge!!**

   Intervention Time Event  Pt Est      se     PnEIC abs(PnEIC / Stop Criteria)
1:          A=1 1461     1 0.03097 0.01161 -0.003864                     1.911
2:          A=0 1461     1 0.04107 0.01614  0.004071                     1.448

For Intervention "A=1", no subjects had G-related nuisance weights falling below 0.0493
For Intervention "A=0", no subjects had G-related nuisance weights falling below 0.0493

Initial Estimators:
Treatment:
                    Risk     Coef
SL.glmnet_All  0.2506075 0.715341
SL.ranger_All  0.2526866 0.144194
SL.xgboost_All 0.2855064 0.140465
SL.glm_All     0.2530794 0.000000

Cens. 0:
             Risk Coef
TrtOnly   375.2720    0
MainTerms 374.6215    1

Event 1:
           Risk Coef
model1 57.00548    0
model2 51.18322    1

Event 2:
           Risk Coef
model1 360.3758    0
model2 322.4561    1
model3 324.6004    0
```

If TMLE has not converged, the mean EICs that have not attained the desired cutoff, i.e. Eq (11), will be displayed in a table. Increasing the the maximum number of TMLE update iterations via `MaxUpdateIter` can allow TMLE to finish updating nuisance parameters, though at target time points when few events have yet occurred even small mean EIC values may not meet the convergence criteria and adequate convergence may require many iterations.

The extent to which the intervention-related nuisance parameters (i.e. propensity scores and probabilities of remaining uncensored) have been lower-bounded is also reported for each intervention both in terms of the percentage of nuisance weights that have been truncated and the percentage of subjects with truncated nuisance weights. If users suspect possible positivity issues, the plot method for `"ConcreteEst"` objects can be used to visualize the distribution of estimated propensity scores for each intervention, with the red vertical line marking the cutoff for truncation.

```
plot(ConcreteEst, ask  = FALSE)
```

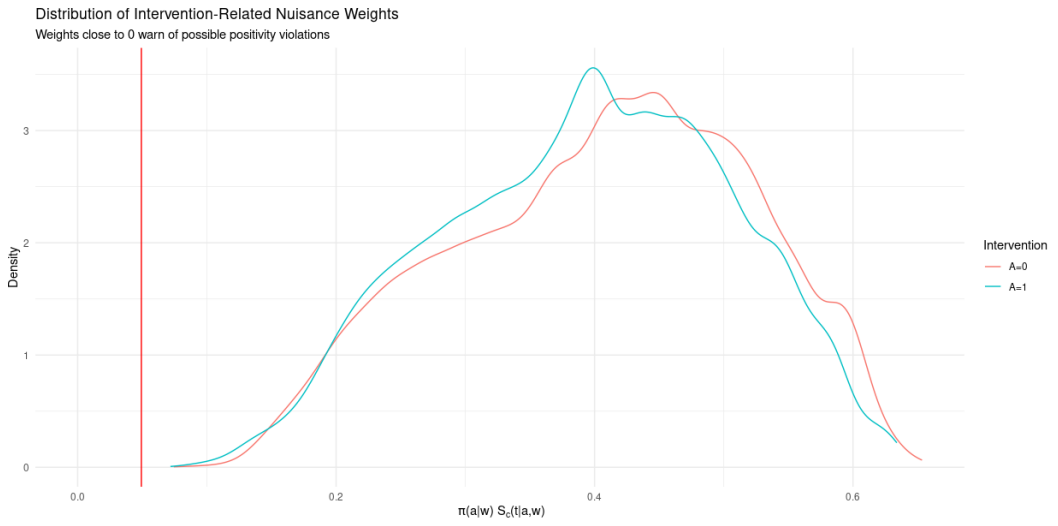


Figure 1

Intervention-related nuisance parameters with values close to 0 indicate the possibility of positivity violations and may warrant re-examining the target time(s), interventions, and covariate adjustment sets. In typical survival applications, positivity issues may arise when targeting times at which some subjects are highly likely to have been censored, or if certain subjects are unlikely to have received a desired treatment intervention. As positivity violations not only impact causal interpretability, but also

estimator behavior, we urge users to re-consider their target analyses; (Petersen et al., 2012) provides guidance on reacting to positivity issues.

The last three tables ("Cens. 0", "Event 1", and "Event 2") in the above code output show the candidate estimators of nuisance parameters, summarized with the cross-validated risk of each candidate estimator followed by their weight in the corresponding Super Learners.

getOutput()

`getOutput()` takes as an argument the `"ConcreteEst"` object returned by `doConcrete()` and can be used to produce tables and plots of the cause-specific risks, risk differences, and relative risks. By default `getOutput()` returns a `data.table` with point estimates and pointwise standard errors for cause-specific absolute risks, risk differences, and risk ratios. By default, the first listed intervention is used as the "treated" group while the second is considered "control"; other contrasts can be specified via the `Intervention` argument. Below we show a subset of the relative risk estimates produced by the "nutshell" estimation specification for the pbc dataset.

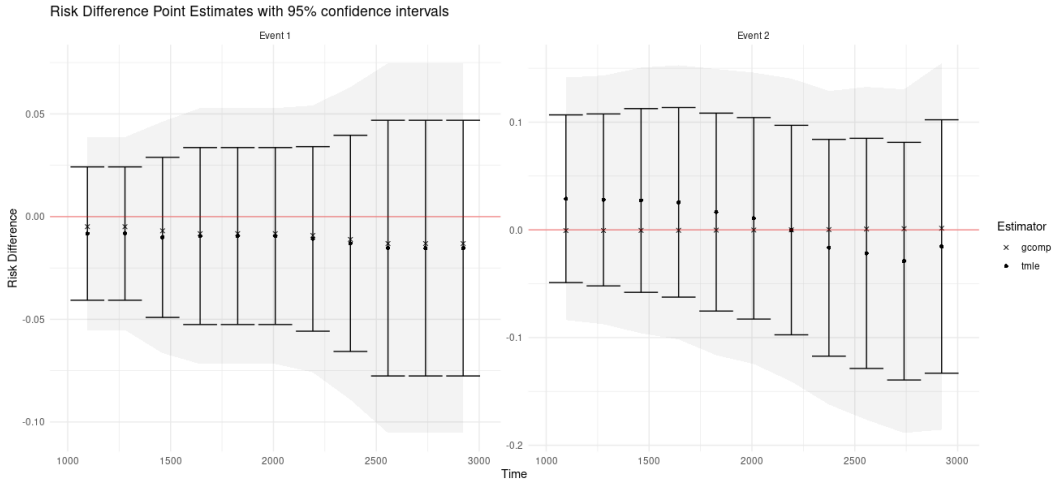
```
ConcreteOut <- getOutput(ConcreteEst = ConcreteEst, Estimand = "RD",
             Intervention = 1:2, GComp = TRUE, Simultaneous = TRUE, Signif = 0.05)
head(ConcreteOut, 12)
```

```
        Time Event  Estimand  Intervention Estimator   Pt Est    se CI Low CI Hi SimCI Low SimCI Hi
 1: 1095.750     1 Risk Diff [A=1] - [A=0]      tmle -0.00820 0.017 -0.041 0.024    -0.055    0.039
 2: 1095.750     1 Risk Diff [A=1] - [A=0]     gcomp -0.00490    NA     NA    NA        NA       NA
 3: 1095.750     2 Risk Diff [A=1] - [A=0]      tmle  0.02900 0.040 -0.049 0.110    -0.084    0.140
 4: 1095.750     2 Risk Diff [A=1] - [A=0]     gcomp -0.00063    NA     NA    NA        NA       NA
 5: 1278.375     1 Risk Diff [A=1] - [A=0]      tmle -0.00820 0.017 -0.041 0.024    -0.055    0.039
 6: 1278.375     1 Risk Diff [A=1] - [A=0]     gcomp -0.00490    NA     NA    NA        NA       NA
 7: 1278.375     2 Risk Diff [A=1] - [A=0]      tmle  0.02800 0.041 -0.052 0.110    -0.088    0.140
 8: 1278.375     2 Risk Diff [A=1] - [A=0]     gcomp -0.00059    NA     NA    NA        NA       NA
 9: 1461.000     1 Risk Diff [A=1] - [A=0]      tmle -0.01000 0.020 -0.049 0.029    -0.066    0.046
10: 1461.000     1 Risk Diff [A=1] - [A=0]     gcomp -0.00690    NA     NA    NA        NA       NA
11: 1461.000     2 Risk Diff [A=1] - [A=0]      tmle  0.02700 0.044 -0.058 0.110    -0.096    0.150
12: 1461.000     2 Risk Diff [A=1] - [A=0]     gcomp -0.00051    NA     NA    NA        NA       NA
```

From left to right, the first five columns show the target times, target events, estimands, interventions, and estimators. The following columns show the point estimates, estimated standard error, confidence intervals and simultaneous confidence bands. Desired level of CI coverage is controlled by the `Signif` argument which is set to a default alpha = 0.05, and whether or not to compute a simultaneous confidence band is controlled by the `Simultaneous` argument.

However, as can often be the case when estimands involve many time points or multiple events, it can be difficult to quickly read treatment effects from a table. Instead plotting can make treatment effects and trends visible at a glance.

```
plot(ConcreteOut, NullLine = TRUE, ask = FALSE)
```



Here 95% confidence bands for the cause-specific risk differences across the target times is shown in gray. The `plot` method for `"ConcreteOut"` object invisibly returns a list of `"ggplot"` objects, which

can be useful for personalizing these graphs. Currently these plots will not signal whether or not TMLE has converged and whether positivity may be an issue, so users should take care not to ignore the diagnostic output of the `"ConcreteEst"` object prior to obtaining effect estimates using `getOutput()`.

Summary

This paper introduces the **concrete** R package implementation of continuous-time estimation for absolute risks of right censored time-to-event outcomes. The package fits into the principled causal-inference workflow laid out by the targeted learning roadmap and allows fully compatible estimation of cause-specific absolute risk estimands for multiple events and at multiple times. The `formatArguments()` function is used to specify desired analyses, `doConcrete()` performs the specified analysis, and `getOutput()` is used to produce formatted output of the target estimands. Cause-specific hazards can be estimated using ensembles of proportional hazards regressions and flexible options are available for estimating treatment propensities. Confidence intervals and confidence bands can be computed for TMLEs, relying on the asymptotic linearity of the TMLEs. We are currently looking into adding support for estimating cause-specific risks using coxnet and HAL-based regressions, as well as supporting stochastic interventions with multinomial or continuous treatment variables.

References

Bibliography

P. C. Austin and J. P. Fine. Accounting for competing risks in randomized controlled trials: a review and recommendations for improvement. *Statistics in Medicine*, 36(8):1203–1209, 2017. ISSN 1097-0258. doi: 10.1002/sim.7215. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.7215. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/sim.7215. [p1]

J. Benichou and M. H. Gail. Estimates of absolute cause-specific risk in cohort studies. *Biometrics*, 46 (3):813–826, Sept. 1990. ISSN 0006-341X. [p1]

D. Benkeser and N. Hejazi. survtmle: Compute Targeted Minimum Loss-Based Estimates in Right-Censored Survival Settings, Apr. 2019. URL https://CRAN.R-project.org/package=survtmle. [p2]

A. F. Bibaut and M. J. van der Laan. Fast rates for empirical risk minimization over c\'adl\'ag functions with bounded sectional variation norm. *arXiv:1907.09244 [math, stat]*, Aug. 2019. URL http://arxiv.org/abs/1907.09244. arXiv: 1907.09244 version: 2. [p5]

T. Chen and C. Guestrin. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, Aug. 2016. Association for Computing Machinery. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939785. URL https://dl.acm.org/doi/10.1145/2939672.2939785. [p5]

D. R. Cox. Partial likelihood. *Biometrika*, 62(2):269–276, Aug. 1975. ISSN 0006-3444. doi: 10.1093/biomet/62.2.269. URL https://doi.org/10.1093/biomet/62.2.269. [p6]

R. Denz, R. Klaaßen-Mielke, and N. Timmesfeld. A Comparison of Different Methods to Adjust Survival Curves for Confounders, Mar. 2022. URL http://arxiv.org/abs/2203.10002. Number: arXiv:2203.10002 arXiv:2203.10002 [stat]. [p3]

S. Dudoit and M. J. van der Laan. Asymptotics of cross-validated risk estimation in estimator selection and performance assessment. *Statistical Methodology*, 2(2):131–154, July 2005. ISSN 1572-3127. doi: 10.1016/j.stamet.2005.02.003. URL https://www.sciencedirect.com/science/article/pii/S1572312705000158. [p5]

S. Ferreira Guerra, M. E. Schnitzer, A. Forget, and L. Blais. Impact of discretization of the timeline for longitudinal causal inference methods. *Statistics in Medicine*, 39(27):4069–4085, 2020. ISSN 1097-0258. doi: 10.1002/sim.8710. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.8710. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/sim.8710. [p3]

J. P. Fine and R. J. Gray. A Proportional Hazards Model for the Subdistribution of a Competing Risk. *Journal of the American Statistical Association*, 94(446):496–509, June 1999. ISSN 0162-1459. doi: 10.1080/01621459.1999.10474144. URL https://www.

tandfonline.com/doi/abs/10.1080/01621459.1999.10474144. Publisher: Taylor & Francis _eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1999.10474144. [p1]

T. R. Fleming and D. P. Harrington. *Counting Processes and Survival Analysis*. Wiley Series in Probability and Mathematical Statistics: Applied Probability and Statistics. John Wiley & Sons, Inc., New York, 1991. ISBN 978-1-118-15067-2. [p2]

J. H. Friedman, T. Hastie, and R. Tibshirani. Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, 33:1–22, Feb. 2010. ISSN 1548-7660. doi: 10.18637/jss.v033.i01. URL https://doi.org/10.18637/jss.v033.i01. [p5]

T. A. Gerds, T. H. Scheike, and P. K. Andersen. Absolute risk regression for competing risks: interpretation, link functions, and prediction. *Statistics in Medicine*, 31(29):3921–3930, 2012. ISSN 1097-0258. doi: 10.1002/sim.5459. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.5459. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/sim.5459. [p1]

T. A. Gerds, J. S. Ohlendorff, P. Blanche, R. Mortensen, M. Wright, N. Tollenaar, J. Muschelli, U. B. Mogensen, and B. Ozenne. riskRegression: Risk Regression Models and Prediction Scores for Survival Analysis with Competing Risks, Sept. 2022. URL https://CRAN.R-project.org/package=riskRegression. [p3]

E. H. Kennedy. Semiparametric Theory and Empirical Processes in Causal Inference. In H. He, P. Wu, and D.-G. D. Chen, editors, *Statistical Causal Inferences and Their Applications in Public Health Research*, ICSA Book Series in Statistics, pages 141–167. Springer International Publishing, Cham, 2016. ISBN 978-3-319-41259-7. doi: 10.1007/978-3-319-41259-7_8. URL https://doi.org/10.1007/978-3-319-41259-7_8. [p1]

J. P. Klein and P. K. Andersen. Regression Modeling of Competing Risks Data Based on Pseudovalues of the Cumulative Incidence Function. *Biometrics*, 61(1):223–229, 2005. ISSN 1541-0420. doi: 10.1111/j.0006-341X.2005.031209.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.0006-341X.2005.031209.x. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.0006-341X.2005.031209.x. [p1]

M. T. Koller, H. Raatz, E. W. Steyerberg, and M. Wolbers. Competing risks and the clinical community: irrelevance or ignorance? *Statistics in Medicine*, 31(11-12):1089–1097, May 2012. ISSN 1097-0258. doi: 10.1002/sim.4384. [p1]

M. J. v. d. Laan. Statistical Inference for Variable Importance. *The International Journal of Biostatistics*, 2 (1), Feb. 2006. ISSN 1557-4679. doi: 10.2202/1557-4679.1008. URL https://www.degruyter.com/document/doi/10.2202/1557-4679.1008/html?lang=en. Publisher: De Gruyter. [p5]

M. J. v. d. Laan. A Generally Efficient Targeted Minimum Loss Based Estimator based on the Highly Adaptive Lasso. *The International Journal of Biostatistics*, 13(2), Oct. 2017. doi: 10.1515/ijb-2015-0097. [p5]

M. J. v. d. Laan and S. Dudoit. Unified Cross-Validation Methodology For Selection Among Estimators and a General Cross-Validated Adaptive Epsilon-Net Estimator: Finite Sample Oracle Inequalities and Examples. *U.C. Berkeley Division of Biostatistics Working Paper Series*, Nov. 2003. [p1]

M. J. v. d. Laan and S. Rose. *Targeted Learning: Causal Inference for Observational and Experimental Data*. Springer Series in Statistics. Springer-Verlag, New York, 2011. ISBN 978-1-4419-9781-4. doi: 10.1007/978-1-4419-9782-1. [p1, 5]

M. J. v. d. Laan and S. Rose. *Targeted Learning in Data Science: Causal Inference for Complex Longitudinal Studies*. Springer Series in Statistics. Springer International Publishing, 2018. ISBN 978-3-319-65303-7. doi: 10.1007/978-3-319-65304-4. [p1]

M. J. v. d. Laan, E. C. Polley, and A. E. Hubbard. Super Learner. *Statistical Applications in Genetics and Molecular Biology*, 6(1), Sept. 2007. ISSN 1544-6115, 2194-6302. doi: 10.2202/1544-6115.1309. [p1, 5]

I. Malenica, M. van der Laan, J. Coyle, N. Hejazi, R. Phillips, and A. Hubbard. Chapter 5 Cross-validation. In *Targeted Learning in R: Causal Data Science with the tlverse Software Ecosystem*. URL https://tlverse.org/tlverse-handbook/origami.html. [p9]

J. Pearl, M. Glymour, and N. P. Jewell. *Causal Inference in Statistics - A Primer*. Wiley, Chichester, West Sussex, 1st edition edition, Mar. 2016. ISBN 978-1-119-18684-7. [p1]

M. L. Petersen and M. J. van der Laan. Causal models and learning from data: integrating causal modeling and statistical estimation. *Epidemiology (Cambridge, Mass.)*, 25(3):418–426, May 2014. ISSN 1531-5487. doi: 10.1097/EDE.0000000000000078. [p1]

M. L. Petersen, K. E. Porter, S. Gruber, Y. Wang, and M. J. van der Laan. Diagnosing and responding to violations in the positivity assumption. *Statistical Methods in Medical Research*, 21(1):31–54, Feb. 2012. ISSN 0962-2802, 1477-0334. doi: 10.1177/0962280210386207. URL http://journals.sagepub.com/doi/10.1177/0962280210386207. [p7, 12]

R. V. Phillips, M. J. van der Laan, H. Lee, and S. Gruber. Practical considerations for specifying a super learner, Apr. 2022. URL http://arxiv.org/abs/2204.06139. arXiv:2204.06139 [stat]. [p1, 5, 8]

E. Polley, E. LeDell, C. Kennedy, S. Lendle, and M. v. d. Laan. SuperLearner: Super Learner Prediction, May 2021. URL https://CRAN.R-project.org/package=SuperLearner. [p1]

J. Robins. 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: 1393–1512, 1986. doi: 10.1016/0270-0255(86)90088-6. URL https://linkinghub.elsevier.com/retrieve/pii/0270025586900886. [p4]

D. B. Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology*, 66(5):688–701, 1974. ISSN 0022-0663. doi: 10.1037/h0037350. URL http://content.apa.org/journals/edu/66/5/688. [p1]

H. C. W. Rytgaard and M. J. van der Laan. One-step TMLE for targeting cause-specific absolute risks and survival curves. *arXiv:2107.01537 [stat]*, Sept. 2021. URL http://arxiv.org/abs/2107.01537. arXiv: 2107.01537 version: 2. [p1, 2, 3, 6, 8]

H. C. W. Rytgaard and M. J. van der Laan. Targeted maximum likelihood estimation for causal inference in survival and competing risks analysis. *Lifetime Data Analysis*, Nov. 2022. ISSN 1572-9249. doi: 10.1007/s10985-022-09576-2. URL https://doi.org/10.1007/s10985-022-09576-2. [p1, 4, 6]

H. C. W. Rytgaard, F. Eriksson, and M. J. van der Laan. Estimation of time-specific intervention effects on continuously distributed time-to-event outcomes by targeted maximum likelihood estimation. *Biometrics*, n/a(00):1–12, 2023. ISSN 1541-0420. doi: 10.1111/biom.13856. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/biom.13856. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/biom.13856. [p1, 5]

T. H. Scheike, M.-J. Zhang, and T. A. Gerds. Predicting cumulative incidence probability by direct binomial regression. *Biometrika*, 95(1):205–220, Mar. 2008. ISSN 0006-3444. doi: 10.1093/biomet/asm096. URL https://doi.org/10.1093/biomet/asm096. [p1]

J. Schwab, S. Lendle, M. Petersen, M. v. d. Laan, and S. Gruber. ltmle: Longitudinal Targeted Maximum Likelihood Estimation, Mar. 2020. URL https://CRAN.R-project.org/package=ltmle. [p2]

M. Sloma, F. J. Syed, M. Nemati, and K. S. Xu. Empirical Comparison of Continuous and Discrete-time Representations for Survival Prediction. *Proceedings of machine learning research*, 146:118–131, Mar. 2021. ISSN 2640-3498. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8232898/. [p3]

O. Sofrygin, M. J. v. d. Laan, and R. Neugebauer. stremr: Streamlined Estimation of Survival for Static, Dynamic and Stochastic Treatment and Monitoring Regimes, Jan. 2017. URL https://CRAN.R-project.org/package=stremr. [p2]

O. Sofrygin, Z. Zhu, J. A. Schmittdiel, A. S. Adams, R. W. Grant, M. J. Laan, and R. Neugebauer. Targeted learning with daily EHR data. *Statistics in Medicine*, 38(16):3073–3090, July 2019. ISSN 0277-6715, 1097-0258. doi: 10.1002/sim.8164. URL https://onlinelibrary.wiley.com/doi/10.1002/sim.8164. [p3]

T. M. Therneau and P. M. Grambsch. *Modeling Survival Data: Extending the Cox Model*. Statistics for Biology and Health. Springer, New York, NY, 2000. ISBN 978-1-4419-3161-0 978-1-4757-3294-8. doi: 10.1007/978-1-4757-3294-8. URL http://link.springer.com/10.1007/978-1-4757-3294-8. [p2]

L. Tran, M. Petersen, J. Schwab, and M. J. van der Laan. Robust variance estimation and inference for causal effect estimation, Oct. 2018. URL http://arxiv.org/abs/1810.03030. arXiv:1810.03030 [math, stat]. [p7]

A. W. v. d. Vaart, S. Dudoit, and M. J. v. d. Laan. Oracle inequalities for multi-fold cross validation. *Statistics & Decisions*, 24(3), Jan. 2006. ISSN 0721-2631. doi: 10.1524/stnd.2006.24.3.351. [p5]

T. Westling, A. Luedtke, P. Gilbert, and M. Carone. Inference for treatment-specific survival curves using machine learning, June 2021. URL http://arxiv.org/abs/2106.06602. Number: arXiv:2106.06602 arXiv:2106.06602 [stat]. [p3]

David Chen
University of California, Berkeley

Helene C. W. Rytgaard
University of Copenhagen

Edwin Fong
Novo Nordisk

Jens M. Tarp
Novo Nordisk

Maya L. Petersen
University of California, Berkeley

Mark J. van der Laan
University of California

Thomas A. Gerds
University of Copenhagen