Documentation for R functions to estimate marginal models under dependent censoring

Mireille E. Schnitzer

1 Introduction

The following functions produce double robust and locally asymptotically efficient estimation of marginal models under Missing at Random (MAR) censoring. The methods are based on the procedures in the manuscript by Schnitzer et al. [2015] and include Sequential Augmented Regression (SAR) and two implementations of Targeted Minimum Loss-based Estimation (TMLE; van der Laan and Rubin 2006, van der Laan and Gruber 2012). The data may be single time-point or have longitudinal censoring. In the longitudinal setting, the outcome of interest may be a survival-type event outcome, or a binary or continuous variable. The marginal models, which define the target of inference, are allowed to be conditional on baseline variables. If no baseline variables are specified for the marginal model, these functions will estimate the marginal mean (adjusted for censoring).

As described in more detail in Schnitzer et al. [2015], we assume that censoring is independent of future prognosis conditional on all past measured variables and that no subjects are deterministically censored. SAR and TMLE are consistent if either the conditional models for censoring or the nested conditional models for the outcome are correctly specified (double robustness). The methods are asymptotically efficient if all of these "nuisance" models are correctly specified. The marginal model of interest may be interpreted as a prognostic model for the outcome under the counterfactual scenario of no censoring.

The functions `SAR` and `TMLE` are implemented by fitting all of the models for censoring with logistic regressions. If the outcome is binary (including event), the outcome models are fit using logistic regression. If the outcome is continuous, the outcome models are fit using linear regression. The function `TMLE_SL` allows the user to specify a collection of methods, inserted as a library into the Super Learner [Polley and Van der Laan, 2011] algorithm, thereby allowing for less-restrictive specifications of the nuisance models.

For a non-event outcome, the data is assumed to have the form $O = (\mathbf{X}; \{\mathbf{L_t}, C_t\}, t = 1, ..., K; Y_K)$ where $t = 1, ..., K$ denotes the time point. The matrix $\mathbf{X}$ has dimension $n \times r$ and is composed of r column vectors of baseline predictors for Y_K. The matrices $\mathbf{L_t}$ have varying dimension $n \times s_t$ and represent time-dependent covariates (suspected to affect censoring at time t) measured just prior to time t. Vector C_t has length n and indicates the censoring status of each subject at time t. Specifically, if C_t is 1 for a given subject, then $L_k, k \geq t$ and Y_K if $K \geq t$ will be missing. Finally, vector Y_K has length n, occurs at the Kth (final) time-point,

and represents the continuous or binary outcome. Note that previous measures of a longitudinal outcome should be included in $\mathbf{L_t}, t < K$. For a continuous outcome, the functions estimate the marginal model

$$E(Y_K \mid \mathbf{X}) = \mathbf{X}\beta$$

The parameter of interest is β, a vector of length r representing the coefficients of the predictors in the marginal model. $\mathbf{X}$ always includes and may be optionally limited to an intercept term. For a binary non-event outcome, the functions estimate the marginal model

$$logit\{E(Y_K \mid \mathbf{X})\} = \mathbf{X}\beta.$$

For an event outcome, the data is assumed to have the form $O = (\mathbf{X}; \{\mathbf{L_t}, C_t, Y_t\}, t = 1, ..., K)$. Here, the column vectors Y_t are monotonic indicators of a survival-type event. For non-incident events (where the sequence $Y_t, t = 1, ..., K$ is non-monotonic) the previous data format for binary outcomes can instead be used. For an event outcome, the functions estimate the marginal model

$$logit\{E(Y_K \mid \mathbf{X})\} = \mathbf{X}\beta.$$

Details

Initial Publication April 16, 2015
Maintainer Mireille Schnitzer `<mireille.schnitzer@umontreal.ca>`
Depends `SuperLearner`
Copyright Université de Montréal
Also see R packages `SuperLearner` (`http://cran.r-project.org/web/packages/SuperLearner/index.html`), `tmle` (`http://cran.r-project.org/web/packages/tmle/`), and `ltmle` (`http://cran.r-project.org/web/packages/ltmle/`).

Index

SAR

Description

Computes the Sequential Augmented Regression estimate. Variance estimation uses the efficient influence function.

Usage

```
SAR(Y, C, Lout, Lcen=Lout, X, event=T, family="binomial")
```

Arguments

`Y`	For non-event data, this is a column vector of length n , representing the outcome of interest at time K. For event data, this is an $n \times K$ matrix where each column represents an indicator for the incident event at a given time.
`C`	A censoring indicator vector or matrix, of dimension $n \times K$. Missing data in the outcome (and optionally, `Lout` and `Lcen`) occurs at and subsequent to the corresponding time point indicated in `C`. For example, if an entry in column `t` in matrix `C` is equal to 1, the function expects missing data in `Lout, Lcen`, and `Y` for that subject at all time points after and including `t`. Missingness is assumed to be monotonic.
`Lout`	A list with K matrix elements. Each matrix is composed of column vectors used in the prediction of the outcome at the corresponding time point. For example, list element `K` is a matrix of predictors for the outcome at time-point K. List element `K-1` (if it exists) is a matrix containing predictors for the nested outcome model labeled $K-1$.
`Lcen`	A list with K matrix elements. Each matrix is composed of column vectors used in the prediction of censoring at the corresponding time point. For example, list element `s` is a matrix of predictors for censoring at time-point s.
`X`	A vector or matrix of dimension $n \times r$ containing the baseline predictors to be included in the marginal model. If the intercept term is not included, the function will insert it. `X=NA` will specify an intercept-only model.
`event`	Logical: whether or not the outcome can be defined as an incident event.
`family`	Either `"binary"` or `"continuous"`, corresponding to the outcome type. If the outcome is an event, use `"binary"`.

Details

Each censoring model is fit with a logistic regression. For binary outcomes, each nested outcome model is fit with a logistic regression. For continuous outcomes, each nested outcome model is fit with a linear regression.

Value

A list with two elements: `Estimates`, a vector of estimates of β and `VarianceEstimates`, a vector of variance estimates for the estimated values.

Examples

Example with a continuous outcome using `data_cont`.

```
source("SAR.R")
source("TMLE.R")
source("data_cont.R")

DATA<-data_cont(ssize=5000,6363)
```

```
W<-DATA$W; U<-DATA$U; L2<-DATA$L2; L3<-DATA$L3;
Y3<-DATA$Y3;
C3<-DATA$C3; C1<-DATA$C1; C2<-DATA$C2


#If event=F, Y is a column vector (additional information about repeated measures
# can be put into Lout and Lcen)
Y<-(Y3)
C<-cbind(C1,C2,C3)
Lout<-list()
#Covariates for the model for the outcome (Y3):
Lout[[3]]<-cbind(W=W,U=U,WU=W*U,U2=U^2,L3=L3)
#Covariate for the model for H3:
Lout[[2]]<-cbind(W=W,U=U,WU=W*U,U2=U^2,L2=L2)
#Covariate for the model for H2:
Lout[[1]]<-cbind(W=W,U=U,WU=W*U,U2=U^2)

Lcen<-list()
#Covariates for the model for C3:
Lcen[[3]]<-cbind(W=W,U=U,WU=W*U,U2=U^2,UL3=U*L3,L3=L3)
#Covariate for the model for C2:
Lcen[[2]]<-cbind(W=W,U=U,WU=W*U,U2=U^2,UL1=U*L2,L2=L2)
#Covariate for the model for C1:
Lcen[[1]]<-cbind(W=W,U=U,WU=W*U,U2=U^2)

#Covariates for marginal model
X<-W

SAR(Y=Y,C=C,Lout=Lout,Lcen=Lcen,X=X,event=F,family="continuous")
TMLE(Y=Y,C=C,Lout=Lout,Lcen=Lcen,X=X,event=F,family="continuous")
```

TMLE

Description

Computes the Targeted Minimum Loss-based estimate. Variance estimation uses the efficient influence function.

Usage

```
TMLE(Y, C, Lout, Lcen=Lout, X, event=T, family="binomial")
```

Arguments

`Y`	For non-event data, this is a column vector of length n , representing the outcome of interest at time K. For event data, this is an $n \times K$ matrix where each column represents an indicator for the incident event at a given time.
`C`	A censoring indicator vector or matrix, of dimension $n \times K$. Missing data in the outcome (and optionally, `Lout` and `Lcen`) occurs at and subsequent to the corresponding time point indicated in `C`. For example, if an entry in column `t` in matrix `C` is equal to 1, the function expects missing data in `Lout, Lcen`, and `Y` for that subject at all time points after and including `t`. Missingness is assumed to be monotonic.
`Lout`	A list with K matrix elements. Each matrix is composed of column vectors used in the prediction of the outcome at the corresponding time point. For example, list element `K` is a matrix of predictors for the outcome at time-point K. List element `K-1` (if it exists) is a matrix containing predictors for the nested outcome model labeled $K-1$.
`Lcen`	A list with K matrix elements. Each matrix is composed of column vectors used in the prediction of censoring at the corresponding time point. For example, list element `s` is a matrix of predictors for censoring at time-point s.
`X`	A vector or matrix of dimension $n \times r$ containing the baseline predictors to be included in the marginal model. If the intercept term is not included, the function will insert it. `X=NA` will specify an intercept-only model.
`event`	Logical: whether or not the outcome can be defined as an incident event.
`family`	Either `"binary"` or `"continuous"`, corresponding to the outcome type. If the outcome is an event, use `"binary"`.

Details

Each censoring model is fit with a logistic regression. For binary outcomes, each nested outcome model is fit with a logistic regression. For continuous outcomes, each nested outcome model is fit with a linear regression.

Value

A list with two elements: `Estimates`, a vector of estimates of β and `VarianceEstimates`, a vector of variance estimates for the estimated values.

Examples

Example with event data using `data_3`.

```
source("sar.r")
source("tmle.r")
source("data_3.r")

DATA<-data_3(ssize=1000,seed=1515)

W<-DATA$W; U<-DATA$U; L2<-DATA$L2; L3<-DATA$L3;
Y1<-DATA$Y1; Y2<-DATA$Y2; Y3<-DATA$Y3;
C3<-DATA$C3; C1<-DATA$C1; C2<-DATA$C2
```

```
#If event=T, Y should be the same dimension as C
Y<-cbind(Y1,Y2,Y3)
C<-cbind(C1,C2,C3)
Lout<-list()
#Covariates must include all desired interactions, squared terms etc
#Covariates for the model for the outcome (Y3):
Lout[[3]]<-cbind(W=W,U=U,WU=W*U,U2=U^2,L3=L3)
#Covariate for the model for H3:
Lout[[2]]<-cbind(W=W,U=U,WU=W*U,U2=U^2,L2=L2)
#Covariate for the model for H2:
Lout[[1]]<-cbind(W=W,U=U,WU=W*U,U2=U^2)

Lcen<-list()
#Covariates for the model for C3:
Lcen[[3]]<-cbind(W=W,U=U,WU=W*U,U2=U^2,UL3=U*L3,L3=L3)
#Covariate for the model for C2:
Lcen[[2]]<-cbind(W=W,U=U,WU=W*U,U2=U^2,UL2=U*L2,L2=L2)
#Covariate for the model for C1:
Lcen[[1]]<-cbind(W=W,U=U,WU=W*U,U2=U^2)

#Covariates for marginal model
X<-W

SAR(Y=Y,C=C,Lout=Lout,Lcen=Lcen,X=X,event=T,family="binomial")
TMLE(Y=Y,C=C,Lout=Lout,Lcen=Lcen,X=X,event=T,family="binomial")
```

Example with simple crossectional data (continuous outcome).

```
source("sar.r")
source("tmle.r")

W<-runif(n=1000,min=-2,max=2)
A<-rbinom(n=1000,1,prob=plogis(1-0.5*W))
Y<-rnorm(n=1000,mean=(1+0.5*A-0.25*W))

Lout<-list()
Lout[[1]]<-W
Lcen<-list()
Lcen[[1]]<-W


#to estimate E(Y^1)
SAR(Y=Y,C=(1-A),Lout=Lout,Lcen=Lcen,X=NA,event=F,family="continuous")
TMLE(Y=Y,C=(1-A),Lout=Lout,Lcen=Lcen,X=NA,event=F,family="continuous")
#to estimate E(Y^0)
SAR(Y=Y,C=(A),Lout=Lout,Lcen=Lcen,X=NA,event=F,family="continuous")
TMLE(Y=Y,C=(A),Lout=Lout,Lcen=Lcen,X=NA,event=F,family="continuous")
```

TMLE_SL

Description

Computes the Targeted Minimum Loss-based estimate using Super Learner. Variance estimation uses the efficient influence function.

Usage

```
TMLE_SL(Y, C, Lout, Lcen=Lout, X, event=T, family="binomial",
        SL.library.bin=c("SL.glm","SL.glm.interaction","SL.glm.squared"),
SL.library.prob=c("SL.glm", "SL.glm.interaction", "SL.glm.squared"),
SL.library.cont=NA)
```

Arguments

`Y`	For non-event data, this is a column vector of length n , representing the outcome of interest at time K. For event data, this is an $n \times K$ matrix where each column represents an indicator for the incident event at a given time.
`C`	A censoring indicator vector or matrix, of dimension $n \times K$. Missing data in the outcome (and optionally, `Lout` and `Lcen`) occurs at and subsequent to the corresponding time point indicated in `C`. For example, if an entry in column `t` in matrix `C` is equal to 1, the function expects missing data in `Lout, Lcen`, and `Y` for that subject at all time points after and including `t`. Missingness is assumed to be monotonic.
`Lout`	A list with K matrix elements. Each matrix is composed of column vectors used in the prediction of the outcome at the corresponding time point. For example, list element `K` is a matrix of predictors for the outcome at time-point K. List element `K-1` (if it exists) is a matrix containing predictors for the nested outcome model labeled $K-1$.
`Lcen`	A list with K matrix elements. Each matrix is composed of column vectors used in the prediction of censoring at the corresponding time point. For example, list element `s` is a matrix of predictors for censoring at time-point s.
`X`	A vector or matrix of dimension $n \times r$ containing the baseline predictors to be included in the marginal model. If the intercept term is not included, the function will insert it. `X=NA` will specify an intercept-only model.
`event`	Logical: whether or not the outcome can be defined as an incident event.
`family`	Either `"binary"` or `"continuous"`, corresponding to the outcome type. If the outcome is an event, use `"binary"`.
`SL.library.bin`	Vector of strings indicating a list of Super Learner wrapper functions that are used to estimate all models with binary outcomes. This always includes all censoring models and also the first outcome model for binary (including event) outcomes.
`SL.library.bin`	Only necessary when `family="binary"`. The list of functions used to estimate models with probability outcomes, specifically the nested outcome models when the outcome is binary. In particular, this list cannot include classification methods, but `SL.library.bin` can.
`SL.library.cont`	Only necessary when `family="continuous"`. The list of functions used to estimate the outcome models.

Details

More details about Super Learner default wrapper functions can be found in the index in the documentation at `http://cran.r-project.org/web/packages/SuperLearner/SuperLearner.pdf`. We include three wrapper functions `"SL.glm"`,`"SL.glm.interaction"` and `"SL.glm.squared"` that can be used for any outcome type.

Value

A list with two elements: `Estimates`, a vector of estimates of β and `VarianceEstimates`, a

vector of variance estimates for the estimated values.

Examples

Example for event data using function `data_3`.

```
source("tmle_sl.r")
source("data_3.r")

DATA<-data_3(ssize=1000,seed=1515)

W<-DATA$W; U<-DATA$U; L2<-DATA$L2; L3<-DATA$L3;
Y1<-DATA$Y1; Y2<-DATA$Y2; Y3<-DATA$Y3;
C3<-DATA$C3; C1<-DATA$C1; C2<-DATA$C2

#If event=T, Y should be the same dimension as C
Y<-cbind(Y1,Y2,Y3)
C<-cbind(C1,C2,C3)
Lout<-list()
#Covariates for the model for the outcome (Y3):
Lout[[3]]<-cbind(W=W,U=U,L3=L3)
#Covariate for the model for H3:
Lout[[2]]<-cbind(W=W,U=U,L2=L2)
#Covariate for the model for H2:
Lout[[1]]<-cbind(W=W,U=U)

Lcen<-list()
#Covariates for the model for C3:
Lcen[[3]]<-cbind(W=W,U=U,L3=L3)
#Covariate for the model for C2:
Lcen[[2]]<-cbind(W=W,U=U,L2=L2)
#Covariate for the model for C1:
Lcen[[1]]<-cbind(W=W,U=U)

#Covariates for marginal model
X<-W

TMLE_SL(Y=Y,C=C,Lout=Lout,Lcen=Lcen,
    X=X,event=T,family="binomial",
SL.library.bin=c("SL.glm","SL.glm.interaction", "SL.glm.squared"),
SL.library.prob=c("SL.glm","SL.glm.interaction","SL.glm.squared"))
```

Example for continuous outcome data using function `data_cont`.

```
source("tmle_sl.r")
source("data_cont.r")
```

```
DATA<-data_cont(ssize=1000,seed=1515)

W<-DATA$W; U<-DATA$U; L2<-DATA$L2; L3<-DATA$L3;
Y3<-DATA$Y3;
C3<-DATA$C3; C1<-DATA$C1; C2<-DATA$C2

#If event=F, Y is a column vector (additional information about repeated measures
#can be put into Lout and Lcen)
Y<-cbind(Y3)
C<-cbind(C1,C2,C3)
Lout<-list()
#Covariates for the model for the outcome (Y3):
Lout[[3]]<-cbind(W=W,U=U,L3=L3)
#Covariate for the model for H3:
Lout[[2]]<-cbind(W=W,U=U,L2=L2)
#Covariate for the model for H2:
Lout[[1]]<-cbind(W=W,U=U)

Lcen<-list()
#Covariates for the model for C3:
Lcen[[3]]<-cbind(W=W,U=U,L3=L3)
#Covariate for the model for C2:
Lcen[[2]]<-cbind(W=W,U=U,L2=L2)
#Covariate for the model for C1:
Lcen[[1]]<-cbind(W=W,U=U)

#Covariates for marginal model
X<-W

library(randomForest)

TMLE_SL(Y=Y,C=C,Lout=Lout,Lcen=Lcen,
  X=X,event=F,family="continuous",
SL.library.bin=c("SL.glm","SL.glm.interaction","SL.glm.squared"),
SL.library.cont=c("SL.glm","SL.glm.interaction","SL.glm.squared","SL.randomForest"))
```

data_3

Description

A function to generate a longitudinally censored event outcome dataset with three time points.

Usage

```
data_3(ssize, seed)
```

Arguments

`ssize`	The sample size desired.
`seed`	A seed (integer).

Details

Data is generated sequentially, starting at the baseline. Censoring and events are monotone.

Value

A list with the following vector elements.

`W, U`	Two baseline variables. `U` is uniform and `W` is binary.
`C1`	A censoring indicator for time 1.
`Y1`	An event indicator at time 1. Missing if `C1=1`.
`L2`	An intermediate variable (predictive of subsequent censoring and events). Missing if `C1=1`.
`C2`	A censoring indicator for time 2.
`Y2`	An event indicator at time 2. Missing if `C2=1`.
`L3`	An intermediate variable (predictive of subsequent censoring and events). Missing if `C2=1`.
`C3`	A censoring indicator for time 3.
`Y3`	An event indicator at time 3. Missing if `C3=1`

data_cont

Description

A function to generate a longitudinally censored continous outcome dataset with three time points.

Usage

```
data_cont(ssize, seed)
```

Arguments

`ssize`	The sample size desired.
`seed`	A seed (integer).

Details

Data is generated sequentially, starting at the baseline. Censoring is monotone.

Value

A list with the following vector elements.

`W, U`	Two baseline variables. `U` is uniform and `W` is binary.
`C1`	A censoring indicator for time 1.
`L2`	An intermediate variable (predictive of subsequent censoring and events). Missing if `C1=1`.
`C2`	A censoring indicator for time 2.
`L3`	An intermediate variable (predictive of subsequent censoring and events). Missing if `C2=1`.
`C3`	A censoring indicator for time 3.
`Y3`	An event indicator at time 3. Missing if `C3=1`

References

E. C. Polley and M. J. Van der Laan. *Package 'SuperLearner'*. CRAN, 2.0-4 edition, 2011.

M. E. Schnitzer, J. J. Lok, and R. J. Bosch. Double robust and efficient estimation of a prognostic model for events in the presence of dependent censoring. 2015.

M. J. van der Laan and S. Gruber. Targeted minimum loss based estimation of causal effects of multiple time point interventions. *The International Journal of Biostatistics*, 8(1):Article 9, 2012.

M. J. van der Laan and D. Rubin. Targeted maximum likelihood learning. *The International Journal of Biostatistics*, 2(1):Article 11, 2006.