Package ‘DTRreg’

October 12, 2022

Type Package

Title DTR Estimation and Inference via G-Estimation, Dynamic WOLS, Q-Learning, and Dynamic Weighted Survival Modeling (DWSurv)

Version 1.7

Date 2020-09-11

Author
Michael Wallace, Erica E M Moodie, David A Stephens, Gabrielle Simoneau and Juliana Schulz

Maintainer Michael Wallace <michael.wallace@uwaterloo.ca>

Description Dynamic treatment regime estimation and inference via G-estimation, dynamic weighted ordinary least squares (dWOLS) and Q-learning. Inference via bootstrap and (for G-estimation) recursive sandwich estimation. Estimation and inference for survival outcomes via Dynamic Weighted Survival Modeling (DWSurv). Extension to continuous treatment variables (gdwols). Wallace et al. (2017) <DOI:10.18637/jss.v080.i02>; Simoneau et al. (2020) <DOI:10.1080/00949655.2020.1793341>.

License GPL-2

Imports graphics, stats, utils, dplyr, MASS

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2020-09-13 16:10:03 UTC

R topics documented:

chooseM *Adaptive Choice of the Bootstrap Resample Size M for the m-out-of-n Bootstrap with for DTR Estimation*

Description

Implementation of a double-bootstrap alogrithm for choosing the bootstrap resample size m in a data-adaptive manner. The function returns a resample size m to be used to apply the m-out-of-n bootstrap with DTRreg.

Usage

```
chooseM(outcome, blip.mod, treat.mod, tf.mod, data = NULL,
        method = "gest", weight = "default", missing = "default",
        treat.mod.man = NULL, B1 = 500, B2 = 500)
```

Arguments

`outcome`	The outcome variable.
`blip.mod`	A list of formula objects specifying covariates of a (linear) blip function for each stage in order. No dependent variable should be specified.
`treat.mod`	A list of formula objects specifying the treatment model for each stage in order. Treatment variable should be included as the dependent variable. If treatment is binary a logistic regression model will be used, otherwise a linear regression model will be used.
`tf.mod`	A list of formula objects specifying covariates of a (linear) treatment-free model for each stage in order. No dependent variable should be specified.
`data`	A data frame containing all necessary covariates contained in the above models.
`method`	The DTR method to be used, choose "dwols" for dynamic WOLS, "gest" for G-estimation, or "qlearn" for Q-learning.
`weight`	If using dynamic WOLS the option for the weights used. Default is the form |A - E[A|...]|, "iptw" gives inverse probability of treatment style weights.
`missing`	If set to "ipcw" and data are missing then inverse probability of censored weights is used with the probability of censoring estimated via logistic regression on the full covariate history up to that point.
`treat.mod.man`	A list of vectors of known treatment weights can be specified to be used instead of those estimated by the routine.
`B1`	Number of first-level boostrap resamples.
`B2`	Number of second-level boostrap resamples.

Details

The m-out-of-n bootstrap is an adequate tool for constructing valid confidence intervals for the first stage parameters in DTRreg. The resample size m is: $m = n^{\frac{1+alpha(1-pHat)}{1+alpha}}$. The estimated non-regularity level is computed by `DTRreg`. The double-bootstrap algorithm is a cross-validation tool for choosing the tuning parameter alpha in a data-driven way.

The current implementation is valid for a two-stage DTR. Moreover, the current implementation may be unstable when there are many missing data.

Value

`m`	Resample size for using in the m-out-of-n bootstrap.

Author(s)

Gabrielle Simoneau

References

Chakraborty, B., Moodie, E. E. M. (2013) *Statistical Methods for Dynamic Treatment Regimes*. New York: Springer.

Efron B., Tibshirani R. J. (1994) An Introduction to the Bootstrap. *CRC press*.

Wallace, M. P., Moodie, E. M. (2015) Doubly-Robust Dynamic Treatment Regimen Estimation Via Weighted Least Squares. *Biometrics* **71**(3), 636–644 (doi:10.1111/biom.12306.)

Examples

```
##################
# example single run of a 2-stage g-estimation analysis
set.seed(1)
# expit function
expit <- function(x) {1 / (1 + exp(-x))}
# sample size
n <- 100
# variables (X = patient information, A = treatment)
X1 <- rnorm(n)
A1 <- rbinom(n, 1, expit(X1))
X2 <- rnorm(n)
A2 <- rbinom(n, 1, expit(X2))
# blip functions
gamma1 <- A1 * (1 + X1)
gamma2 <- A2 * (1 + X2)
# observed outcome: treatment-free outcome plus blip functions
Y <- exp(X1) + exp(X2) + gamma1 + gamma2 + rnorm(n)
# models to be passed to DTRreg
# blip model
blip.mod <- list(~X1, ~X2)
# treatment model (correctly specified)
treat.mod <- list(A1~X1, A2~X2)
# treatment-free model (incorrectly specified)
tf.mod <- list(~X1, ~X2)
```

```
# perform dWOLS without calculating confidence intervals
mod1 <- DTRreg(Y, blip.mod, treat.mod, tf.mod, method = "dwols")

# choose m adaptively for that model
# m <- chooseM(Y, blip.mod, treat.mod, tf.mod, method = "dwols",
#  B1 = 200, B2 = 200)$m
m <- 94

# dWOLS with confidence intervals from the m-out-of-n bootstrap
mod2 <- DTRreg(Y, blip.mod, treat.mod, tf.mod, method = "dwols",
  var.estim = "bootstrap", M = m)

##################
```

`confint` *Flexible Confidence Interval Calculations for DTRs*

Description

Confidence intervals for dWOLS or DWSurv parameters, with the possibility of deriving constructing the confidence intervals using the percentile method when bootstrap is used (DWSurv only).

Usage

```
## S3 method for class 'DTRreg'
confint(object, parm = NULL, level = 0.95, type = "se", ...)
```

Arguments

`object`	A model object generated by the function DTRreg.
`type`	Typical Wald-type confidence interval "se" (default) or confidence intervals derived with the percentile method "percentile" (currently available with dWSurv only).
`parm`	Not available for DTRreg objects.
`level`	the confidence level required.
`...`	Space for additional arguments (not currently used by DTRreg).

Value

A list with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2 in

Author(s)

Gabrielle Simoneau

References

Simoneau G, Moodie EE, Wallace MP, Platt RW. Optimal dynamic treatment regimes with survival endpoints: introducing DWSurv in the R package DTRreg. Journal of Statistical Computation and Simulation. 2020 Jul 16:1-8.

Examples

```
##################
# simulate data
expit <- function(x) exp(x) / (1 + exp(x))
theta1 <- c(4.7, 1.5, -0.8, 0.1, 0.1)
n <- 100
X1 <- runif(n, 0.1, 1.29)
X12 <- rbinom(n, 1, 0.4)
A1 <- rbinom(n, 1, expit(2*X1 - 1))
delta <- rbinom(n, 1, expit(3*X12 + 0.1))
logT <- theta1[1] + theta1[2]*X1[delta == 1] + theta1[3]*X12[delta == 1] +
theta1[4]*A1[delta == 1] + theta1[5]*A1[delta == 1]*X1[delta == 1] +
rnorm(sum(delta), sd = 0.3)

C <- rexp(n - sum(delta), rate = 1/300)
Y <- rep(NA, n)
Y[delta == 1] <- exp(logT)
Y[delta == 0] <- C

dataset <- data.frame(X1, X12, A1, delta, Y)

model <- DWSurv(time = list(~Y), blip.mod = list(~X1), treat.mod = list(A1~X1),
tf.mod = list(~X1 + X12), cens.mod = list(delta~X12), data = dataset, var.estim = "bootstrap",
boot.opt = "standard", B = 200)
confint(model, type = "percentile")
#################
```

DTRreg *DTR Estimation and Inference via G-estimation and Dynamic WOLS*

Description

Dynamic treatment regimen estimation and inference via G-estimation and dynamic WOLS. Estimation of blip model parameters for multi-stage data.

Usage

```
DTRreg(outcome, blip.mod, treat.mod, tf.mod, data = NULL,
        method = "gest", weight = "default", var.estim = "none",
        B = 200, M = 0, truncate = 0, verbose = "FALSE",
        interrupt = "FALSE", treat.range = NULL, missing = "default",
        interactive = FALSE, treat.mod.man = NULL, type = "DTR")
```

Arguments

`outcome`	The outcome variable.
`blip.mod`	A list of formula objects specifying covariates of a (linear) blip function for each stage in order. No dependent variable should be specified.
`treat.mod`	A list of formula objects specifying the treatment model for each stage in order. Treatment variable should be included as the dependent variable. If treatment is binary a logistic regression model will be used, otherwise a linear regression model will be used.
`tf.mod`	A list of formula objects specifying covariates of a (linear) treatment-free model for each stage in order. No dependent variable should be specified.
`data`	A data frame containing all necessary covariates contained in the above models.
`method`	The DTR method to be used, choose "dwols" for dynamic WOLS, "gest" for G-estimation, or "qlearn" for Q-learning.
`weight`	If using dynamic WOLS the option for the weights used. Default is the form |A - E[A|...]|, "iptw" gives inverse probability of treatment style weights.
`var.estim`	Covariance matrix estimation method, either "bootstrap" (for either dWOLS or G-estimation) or "sandwich" for recursive sandwich estimation in the G-estimation context.
`B`	Number of bootstrap samples.
`M`	Subsample size for m out of n bootstrap. If unspecified this is set to the sample size (i.e. n)
`truncate`	Bootstrap option. Truncate (a number between 0 and 0.5) will replace the lowest and highest specified proportion of parameter estimates with the relevant quantiles affording some robustness to extreme values when estimating covariance.
`verbose`	Bootstrap option. If TRUE then estimated time to completion will be printed approximately every 30 seconds.
`interrupt`	Bootstrap option. If TRUE then user will be given the option to abort if estimated time to completion exceeds 10 minutes.
`treat.range`	For continuous treatments. Specify the maximum/minimum value that treatments can be take. If unspecified then the minimum/maximum value of observed treatments is used. If you wish to have unrestricted treatments set this option to c(-Inf,+Inf).
`missing`	If set to "ipcw" and data are missing then inverse probability of censored weights is used with the probability of censoring estimated via logistic regression on the full covariate history up to that point.
`interactive`	If TRUE on-screen prompts will guide the user through the specification of blip, treatment and treatment-free models.
`treat.mod.man`	A list of vectors of known treatment weights can be specified to be used instead of those estimated by the routine.
`type`	If specified as something other than "DTR", DTRreg will take an ‘effect estimation’ (as opposed to a DTR estimation) approach, treating the observed outcome as being equal to an outcome assuming no treatment is received at any stage, plus a blip component at each stage. The main difference is that each stage’s

pseudo-outcome is generated by subtracting a blip function, rather than adding a regret function as in the DTR framework. Note that most of the DTR-specific output will either be suppressed or irrelevant.

Details

DTRreg allows the estimation of optimal dynamic treatment regimens (DTRs, also known as adaptive treatment strategies) from multi-stage trials using G-estimation and dynamic weighted ordinary least squares (dWOLS). Both methods focus on estimating the parameters of the blip: a model of the difference in expected outcome under the observed treatment and some reference treatment (usually a control) at a given stage, assuming identical histories and optimal treatment thereafter. The reader is referred to Chakraborty and Moodie (2013) for a thorough introduction and review of DTR methods. The dWOLS method may be used to obtain parameter estimates identical to those from Q-learning (by setting method = "qlearn"). This option is intended primarily for exploratory purposes; the authors note that there is a dedicated R package for Q-learning (qLearn), although it is limited to the 2-stage setting.

Both of these methods require the specification of three models for each stage of the analysis: a treatment model (conditional mean of the treatment variable), a treatment-free model (conditional mean of outcome assuming only reference treatments are used), and a blip model. Only the blip model must be correctly specified (or over-specified), with consistent parameter estimates obtainable if at least one of the other two models is correctly specified. Note that all of these must be specified as lists of formula objects, even if only one stage of treatment is considered.

Note that as is conventional, it is assumed a larger value of the outcome is preferred (which can be easily achieved via transformation of your data if necessary).

When treatment is binary, if confidence intervals are computed (via specification of var.estim other than 'none'), then DTRreg will calculate the proportion of subjects at each stage for whom optimal treatment is non-unique. If this proportion exceeds 0.05 a non-regularity warning will be displayed, along with the proportion of subjects for whom this is the case. Note that this warning is only displayed if a variance estimation option is selected.

Value

An object of class `DTR`, a list including elements

`psi`	Blip parameter estimates for each stage of treatment.
`opt.treat`	Optimal treatment decisions for each subject at each stage of treatment.
`covmat`	Covariance matrix of blip parameter estimates.
`regret`	Estimates of the regret for each subject based on observed treatment and blip parameter estimates.
`beta`	Treatment-free model parameter estimates (note that these may not be consistent).
`opt.Y`	Predicted optimal outcome under recommended regimen.
`nonreg`	Non-regularity estimates.

The functions coef, predict and confint may be used with such model objects. The first two have specific help files for their implementation, while confint is used in the same way as the standard confint command, with the exception of the parm option, which is not available.

Author(s)

Michael Wallace

References

Chakraborty, B., Moodie, E. E. M. (2013) *Statistical Methods for Dynamic Treatment Regimes*. New York: Springer.

Robins, J. M. (2004) *Optimal structural nested models for optimal sequential decisions*. In Proceedings of the Second Seattle Symposium on Biostatistics, D. Y. Lin and P. J. Heagerty (eds), 189–326. New York: Springer.

Wallace, M. P., Moodie, E. M. (2015) Doubly-Robust Dynamic Treatment Regimen Estimation Via Weighted Least Squares. *Biometrics* **71**(3), 636–644 (doi:10.1111/biom.12306.)

Examples

```
##################
# example single run of a 2-stage g-estimation analysis
set.seed(1)
# expit function
expit <- function(x) {1 / (1 + exp(-x))}
# sample size
n <- 10000
# variables (X = patient information, A = treatment)
X1 <- rnorm(n)
A1 <- rbinom(n, 1, expit(X1))
X2 <- rnorm(n)
A2 <- rbinom(n, 1, expit(X2))
# blip functions
gamma1 <- A1 * (1 + X1)
gamma2 <- A2 * (1 + X2)
# observed outcome: treatment-free outcome plus blip functions
Y <- exp(X1) + exp(X2) + gamma1 + gamma2 + rnorm(n)
# models to be passed to DTRreg
# blip model
blip.mod <- list(~X1, ~X2)
# treatment model (correctly specified)
treat.mod <- list(A1~X1, A2~X2)
# treatment-free model (incorrectly specified)
tf.mod <- list(~X1, ~X2)

# perform G-estimation
mod1 <- DTRreg(Y, blip.mod, treat.mod, tf.mod, method = "gest")
mod1
##################
```

DWSurv *DTR estimation and inference for time-to-event data using DWSurv*

Description

Dynamic treatment regimen estimation and inference via dynamic weighted survival modeling (DWSurv). Inference for the blip estimators with single- and multi-stage data.

Usage

```
DWSurv(time, blip.mod, treat.mod, tf.mod, cens.mod, data = NULL, weight = "default",
       var.estim = "none", asymp.opt = "adjusted", boot.opt = "standard", B = 500,
         optimization = "max", quiet = FALSE)
```

Arguments

`time`	A list of formula specifying the survival time variable for each stage in order. The time variable should be specified on the right hand side of the formula. No dependent variable should be specified. The list should be as long as the maximum number of stages.
`blip.mod`	A list of formula objects specifying covariates of a (linear) blip function for each stage in order. No dependent variable should be specified.
`treat.mod`	A list of formula objects specifying the treatment model for each stage in order. The treatment variable should be binary and included as the dependent variable. Logistic regression models are used.
`tf.mod`	A list of formula objects specifying covariates of a (linear) treatment-free model for each stage in order. No dependent variable should be specified.
`cens.mod`	A list of formula objects specifying the censoring model for each stage in order. The event indicator, which takes value 1 if an event was observed and 0 otherwise, should be included as the dependent variable and should be the same across stages. In the absence of censoring, one still needs to specify an event indicator with 1s on the right-hand side of the formula and leave the left-hand side empty (see example below).
`data`	A data frame containing all necessary covariates contained in the above models.
`weight`	A user-supplied function for the weights to be used in DWSurv. The function must have the four following arguments: treatment received A, probability of receiving treatment A=1, status, probability of being observed status = 1. Default is the inverse probability of censoring weights combined with |A - E[A|...]|.
`var.estim`	Covariance matrix estimation method, either "asymptotic", "bootstrap" or "none" (default).
`asymp.opt`	If the asymptotic variance estimation is used, specify either the "adjusted" (default) or "naive" version.
`boot.opt`	If bootstrap is used for variance estimation, specify either the "standard" (default), "empirical" or "normal". The last two are parametric bootstraps.

`B`	Number of bootstrap resamples, if applicable.
`optimization`	If "max" (default), it is assumed that larger values/longer survival times are preferred. Set to "min" if the sequence of optimal decision rules should minimize survival time.
`quiet`	To suppress warnings when bootstrapping.

Details

The function `DWSurv()` allows estimating an optimal dynamic treatment regime from multi-stage trials or observational data when the outcome of interest is survival time subject to right-censoring. The dynamic weighted survival modeling (DWSurv) algorithm is implemented. The method focuses on estimating the parameters of the blip: a model of the difference in expected outcome under the observed treatment and some reference treatment (usually a control) at a given stage, assuming identical histories and optimal treatment thereafter.

The method requires the specification of four models for each stage of the analysis: a treatment model (conditional mean of the treatment variable), a censoring model, a treatment-free model (conditional mean of outcome assuming only reference treatments are used), and a blip model. Only the blip model must be correctly specified (or over-specified), with consistent parameter estimates obtainable if at least one of the treatment-free or the treatment and censoring models are correctly specified. Note that all of these must be specified as lists of formula objects, even if only one stage of treatment is considered.

Note that as is conventional, it is assumed a larger survival time is preferred (which can be easily achieved via transformation of your data if necessary).

Value

An object of class `DTR`, a list including elements

`psi`	Blip parameter estimates for each stage of treatment.
`opt.treat`	Optimal treatment decisions for each subject at each stage of treatment.
`covmat`	Covariance matrix of blip parameter estimates.
`log.regret`	Estimates of the log-transformed regret for each subject based on observed treatment and blip parameter estimates.
`beta`	Treatment-free model parameter estimates (note that these may not be consistent).
`opt.Y`	Predicted optimal survival time under recommended regimen.
`nonreg`	Non-regularity estimates.
`psi.boot`	If applicable, the B bootstrap estimates of the blip parameters across stages.

The functions coef and confint may be used with such model objects. The first has specific help files for their implementation, while confint is used in the same way as the standard confint command, with an additional `type` options which can be set to "percentile" when bootstrap is used to derive confidence intervals. The parm option is not available.

Author(s)

Gabrielle Simoneau

References

Simoneau, G., Moodie, E. E. M., Wallace, M.P., Platt, R. W. (2020) Optimal Dynamic Treatment Regimes with Survival Endpoints: Introducing DWSurv in the R package DTRreg. *Journal of Statistical Computation and Simulation (in press)*.

Simoneau, G., Moodie, E. E. M., Nijjar, J. S., Platt, R. W. (2019) Estimating Optimal Dynamic Treatment with Survival Outcomes. *Journal of the American Statistical Association*, pp.1-9 (doi:10.1080/01621459.2019.162

Wallace, M. P., Moodie, E. E. M., Stephens, D. A. (2017) Dynamic Treatment Regimen Estimation via Regression-Based Techniques: Introducing R Package DTRreg. *Journal of Statistical Software* **80**(2), 1–20 (doi:10.18637/jss.v080.i02).

Examples

```
##################
#### example single run of a 2-stage DWSurv analysis
set.seed(1)
# expit function
expit <- function(x) {1 / (1 + exp(-x))}
# sample size and parameters
n <- 1000
theta1 <- c(6.3, 1.5, -0.8, 0.1, 0.1)
theta2 <- c(4, 1.1, -0.2, -0.9, 0.6, -0.1)
lambda <- 1/300
p <- 0.9
beta <- 2
# covariates and treatment (X = patient information, A = treatment)
X1 <- runif(n, 0.1, 1.29)
X14 <- X1^4
A1 <- rbinom(n, size = 1, prob = expit(2*X1 - 1))
X2 <- runif(n, 0.9, 2)
X23 <- X2^3
A2 <- rbinom(n, size = 1, prob = expit(-2*X2 + 2.8))
delta <- rbinom(n, size = 1, prob = expit(2*X1 - 0.4))
eta2 <- rbinom(n, 1, prob = 0.8)
delta2 <- delta[eta2 == 1]
# survival time
logY2 <- logT2 <- theta2[1] + theta2[2]*X2[eta2 == 1] + theta2[3]*X23[eta2 == 1]
  + theta2[4]*A2[eta2 == 1] + theta2[5]*A2[eta2 == 1]*X2[eta2 == 1]
  + theta2[6]*X1[eta2 == 1] + rnorm(sum(eta2), sd = 0.3)
trueA2opt <- ifelse(theta2[4]*A2[eta2 == 1]
  + theta2[5]*A2[eta2 == 1]*X2[eta2 == 1] > 0, 1, 0)
logT2opt <- logT2 + (trueA2opt - A2[eta2 == 1])*(theta2[4]*A2[eta2 == 1]
  + theta2[5]*A2[eta2 == 1]*X2[eta2 == 1])
logT <- theta1[1] + theta1[2]*X1 + theta1[3]*X14 + theta1[4]*A1
  + theta1[5]*A1*X1 + rnorm(n, sd = 0.3)
T1 <- exp(logT[eta2 == 1 & delta == 1]) - exp(logT2opt[delta2 == 1])
logT[eta2 == 1 & delta == 1] <- log(T1 + exp(logT2[delta2 == 1]))
# censoring time
C <- (- log(runif(n - sum(delta), 0, 1))/(lambda * exp(beta * X1[delta == 0])))^(1/p)
eta2d0 <- eta2[delta == 0]
C1 <- rep(NA, length(C))
C2 <- rep(NA, length(C))
```

```
for(i in 1:length(C))
{
  if(eta2d0[i] == 0){
    C1[i] <-  C[i]
    C2[i] <- 0
  }else{
    C1[i] <- runif(1, 0, C[i])
    C2[i] <- C[i] - C1[i]
  }
}
# observed survival time
Y2 <- rep(NA, n)
Y1 <- rep(NA, n)
Y2[delta == 0] <- C2
Y1[delta == 0] <- C1
Y1[delta == 1 & eta2 == 1] <- T1
Y1[delta == 1 & eta2 == 0] <- exp(logT[delta == 1 & eta2 == 0])
Y2[delta == 1 & eta2 == 0] <- 0
Y2[delta == 1 & eta2 == 1] <- exp(logT2[delta2 == 1])
logY <- log(Y1 + Y2)
logY2 <- log(Y2[eta2 == 1])
# data and run DWSurv
mydata <- data.frame(X1,X14,A1,X2,X23,A2,delta,Y1,Y2)
mod <- DWSurv(time = list(~Y1, ~Y2), blip.mod = list(~X1, ~X2),
  treat.mod = list(A1~X1, A2~X2), tf.mod = list(~X1 + X14, ~X2 + X23 + X1),
  cens.mod = list(delta~X1, delta~X1), var.estim = "asymptotic", data = mydata)
mod

#### example in the absence of censoring
# create an event indicator
delta <- rep(1,n)
delta2 <- delta[eta2 == 1]
T1 <- exp(logT[eta2 == 1 & delta == 1]) - exp(logT2opt[delta2 == 1])
logT[eta2 == 1 & delta == 1] <- log(T1 + exp(logT2[delta2 == 1]))
# observed survival time
Y2 <- rep(NA, n)
Y1 <- rep(NA, n)
Y1[delta == 1 & eta2 == 1] <- T1
Y1[delta == 1 & eta2 == 0] <- exp(logT[delta == 1 & eta2 == 0])
Y2[delta == 1 & eta2 == 0] <- 0
Y2[delta == 1 & eta2 == 1] <- exp(logT2[delta2 == 1])
logY <- log(Y1 + Y2)
logY2 <- log(Y2[eta2 == 1])
# data and run DWSurv
mydata <- data.frame(X1,X14,A1,X2,X23,A2,delta,Y1,Y2)
mod_nocensoring <- DWSurv(time = list(~Y1, ~Y2), blip.mod = list(~X1, ~X2),
  treat.mod = list(A1~X1, A2~X2), tf.mod = list(~X1 + X14, ~X2 + X23 + X1),
  cens.mod = list(~delta, ~delta), var.estim = "asymptotic", data = mydata)
mod_nocensoring
##################
```

gdwols *DTR Estimation and Inference via Generalized dWOLS*

Description

Dynamic treatment regimen estimation via generalized dynamic weighted ordinary least squares for continuous treatment. Estimation of blip model parameters for multi-stage data.

Usage

```
gdwols(outcome, Xpsi1, Xpsi2, treat.mod, treat.fam, tf.mod,
        weight.fcn, data = NULL, m, k, treat.range)
```

Arguments

`outcome`	The outcome variable.
`Xpsi1, Xpsi2`	A list of formula objects specifying the tailoring variables of a (quadratic) blip function for each stage in order. Xpsi1 specifies the covariates interacting with the treatment (A); Xpsi2 specifies the covariates interacting with squared-treatment (A^2). No dependent variable should be specified.
`treat.mod`	A list of formula objects specifying the treatment model for each stage in order. The treatment variable should be included as the dependent variable.
`treat.fam`	A description of the dose distribution along with the link function to be used in the treatment model for computing weights; should be specified in a similar format as that used in the standard glm function. Options include the gaussian and gamma distributions; ex: gaussian(link = "identity"), Gamma(link = "log"). If unspecified, default will be gaussian(link = "identity").
`tf.mod`	A list of formula objects specifying covariates of a (linear) treatment-free model for each stage in order. No dependent variable should be specified.
`weight.fcn`	The weight function to be used in calculating the balancing weights. Options include "ipw", "cipw", "qpom", "wo"; see details below.
`data`	The data (across all stages), including the outcome, stage-specific covariates and doses.
`m`	The number of bins (levels) to be used for categorizing the continuous doses. (This argument is only pertinent in the case where the weight function used is based on a categorization of the continuous doses.)
`k`	The number of treatment stages.
`treat.range`	The range of permissible dose values for continuous treatments; the optimal treatment will be restricted to lie withinin the specified treatment range. If unspecified then the minimum/maximum value of the observed treatments is used. For unrestricted treatments, set this option to c(-Inf,+Inf). In the case of an unrestricted treatment range, a warning message will be shown whenever the optimal treatment is in fact a minimum for at least one observation.

Details

GdWOLS (generalized dynamic weighted ordinaly least squares) allows for the estimation of optimal dynamic treatment regimes (DTRs, also known as adaptive treatment strategies) for multi-stage trials in the case where treatment is measured on a continuous scale. This method allows for the estimation of the blip model parameters. The GdWOLS approach requires the specification of three models at each stage: a treatment model (conditional mean of the treatment variable), a treatment-free model (conditional mean of the outcome assuming the reference treatment levels are used, in this case, 0), and a blip model (the difference in expected outcome under the observed treatment and the reference treatment at a given stage, assuming identical histories and optimal treatment thereafter). In the GdWOLS framework, only the blip model must be correctly specified (or over-specified); consistent parameter estimates are obtained as long as at least one of the other two models is correctly specified.

Several weight function options have been implemented within the package, including the IPW, capped-IPW, Q-POM and W-O weights:

"ipw": (IPW) weights based on the inverse probability of treatment, probabilities are calculated using a GLM with the specified family and link funciton provided in the treat.fam argument.

"cipw": (capped-IPW) IPW weights capped at the 99th percentile of the observed weights

"qpom": (Q-POM) weights based on the stabilized inverse probability of treatment applied to the categorized (into m bins) doses, probabilities are calculated using a proportional odds model

"wo": (W-O) overlap weights for the categorized continuous doses (Li and Li, 2019)

Note: - All the model components (Xpsi1, Xpsi2, treatment and treatment-free models) must be specified as lists of formula objects, even if only one stage of treatment is considered. - The implementation in the gdwols function only considers quadratic blip functions of the form psi1*Xpsi1*A + psi2*Xpsi2*(A^2). This ensures that the optimal treatment is given a.opt=-(psi1*X1)/2(psi2*X2), provided that psi2*X2<0, see paper for more details. - As is conventional, it is assumed that a larger value of the outcome is preferred (which can be easily achieved via transformation of the data if necessary).

Value

An object of class `DTR`, a list including elements

`psi`	Blip parameter estimates for each stage of treatment.
`beta`	Treatment-free model parameter estimates for each stage of treatment.
`opt.treat`	Optimal treatment value for each stage of treatment.

Author(s)

Juliana Schulz

References

Schulz, J., and Moodie, E. E. M. (2020) Doubly Robust Estimation of Optimal Dosing Strategies. Journal of the American Statistical Association. (DOI: 10.1080/01621459.2020.1753521)

Li, F., and Li, F. (2019) "Propensity Score Weighting for Causal Inference With Multiple Treatments," The Annals of Applied Statistics, 13, 2389-2415.

Examples

```
##################
# example single run of a 2-stage G-dWOLS analysis

set.seed(1)
# number of stages
k<-2
# number of bins for categorized weight functions
m<-5
# sample size
n<-1000

# parameter values
psi.mat<-matrix(rep(c(1,1,-1),2),byrow=TRUE,nrow=2)
alpha.mat<-matrix(rep(c(-1,1),2),byrow=TRUE,nrow=2)

### generate data
# stage 1
x1<-abs(rnorm(n,10,1))
a1<-rnorm(n,alpha.mat[1,1]+alpha.mat[1,2]*x1,1)
# stage 2
x2<-abs(rnorm(n,10,1))
a2<-rnorm(n,alpha.mat[2,1]+alpha.mat[2,2]*x2,1)
# blips
gamma1<-as.matrix(cbind(a1,a1*x1,a1^2))
gamma2<-as.matrix(cbind(a2,a2*x2,a2^2))
# y: outcome
# y <- trmt free + blip
y<-log(x1)+sin(x1)+log(x2)+sin(x2)+gamma1+gamma2 + rnorm(n,0,1)
# convert to a vector for formatting into data frame
y <- as.vector(y)
# data
data<-data.frame(cbind(y,x1,x2,a1,a2))

# models to be passed to gdwols function
# tailoring variables that interact with a1
Xpsi1<-list(~x1,~x2)
# tailoring variables that interact with a2
Xpsi2<-list(~1,~1)
# treatment model at each stage
treat.mod<-list(a1~x1,a2~x2)
# treatment-free model at each stage (misspecified)
tf.mod<-list(~x1,~log(x2)+sin(x2))


out1 <- gdwols(y, Xpsi1, Xpsi2, treat.mod, treat.fam = gaussian(link = "identity"),
               tf.mod, weight.fcn="ipw", data, m, k)
out1$psi

out2 <- gdwols(y, Xpsi1, Xpsi2, treat.mod, treat.fam = gaussian(link = "identity"),
               tf.mod, weight.fcn="cipw", data, m, k)
```

```
out2$psi

out3 <- gdwols(y, Xpsi1, Xpsi2, treat.mod, treat.fam = gaussian(link = "identity"),
               tf.mod, weight.fcn="qpom", data, m, k)
out3$psi

out4 <- gdwols(y, Xpsi1, Xpsi2, treat.mod, treat.fam = gaussian(link = "identity"),
               tf.mod, weight.fcn="wo", data, m, k)
out4$psi


##################
```

`plot` *Diagnostic Plots for DTR Estimation*

Description

Diagnostic plots for assessment of treatment, treatment-free and blip models following DTR estimation using DTRreg and DWSurv.

Usage

```
## S3 method for class 'DTRreg'
plot(x, method = "DTRreg", ...)
```

Arguments

`x`	A model object generated by the functions DTRreg and dWSurv.
`method`	"DTRreg" if the function DTRreg was used (default) and "dWSurv" if the function dWSurv was used.
`...`	Space for additional arguments (not currently used by DTRreg)

Details

DTR estimation using G-estimation and dWOLS requires the specification of three models: the treatment, treatment-free and blip. The treatment model may be assessed via standard diagnostics, whereas the treatment-free and blip models may be simultaneously assessed using diagnostic plots introduced by Rich et al. The plot() function first presents diagnostic plots that assess the latter, plotting fitted values against residuals and covariates following DTR estimation. If there is any evidence of a relationship between the variables in these plots, this is evidence that at least one of the blip or treatment-free models is mis-specified.

Following these plots, the plot() function will present standard diagnostic plots for the treatment model. These are produced directly by the standard plot() command applied to the models that were fit. For example, if treatment is binary, the resulting plots are the same as those that are generated by the plot() command applied to a glm object for logistic regression.

Author(s)

Michael Wallace

References

Chakraborty, B., Moodie, E. E. M. (2013) *Statistical Methods for Dynamic Treatment Regimes*. New York: Springer.

Rich B., Moodie E. E. M., Stephens D. A., Platt R. W. (2010) Model Checking with Residuals for G-estimation of Optimal Dynamic Treatment Regimes. *International Journal of Biostatistics* **6**(2), Article 12.

Robins, J. M. (2004) *Optimal structural nested models for optimal sequential decisions*. In Proceedings of the Second Seattle Symposium on Biostatistics, D. Y. Lin and P. J. Heagerty (eds), 189–326. New York: Springer.

Wallace, M. P., Moodie, E. M. (2015) Doubly-Robust Dynamic Treatment Regimen Estimation Via Weighted Least Squares. *Biometrics* **71**(3), 636–644 (doi:10.1111/biom.12306.)

Examples

```
##################
# example single run of a 2-stage g-estimation analysis
set.seed(1)
# expit function
expit <- function(x) {1 / (1 + exp(-x))}
# sample size
n <- 10000
# variables (X = patient information, A = treatment)
X1 <- rnorm(n)
A1 <- rbinom(n, 1, expit(X1))
X2 <- rnorm(n)
A2 <- rbinom(n, 1, expit(X2))
# blip functions
gamma1 <- A1 * (1 + X1)
gamma2 <- A2 * (1 + X2)
# observed outcome: treatment-free outcome plus blip functions
Y <- exp(X1) + exp(X2) + gamma1 + gamma2 + rnorm(n)
# models to be passed to DTRreg
# blip model
blip.mod <- list(~X1, ~X2)
# treatment model (correctly specified)
treat.mod <- list(A1~X1, A2~X2)
# treatment-free model (incorrectly specified)
tf.mod <- list(~X1, ~X2)

# perform G-estimation
mod1 <- DTRreg(Y, blip.mod, treat.mod, tf.mod, method = "gest")

# model diagnostics: note treatment-free model is mis-specified
plot(mod1)
##################
```

predict *Optimal Outcome Prediction for DTRs*

Description

Predicted outcome assuming optimal treatment (according to analysis via G-estimation or dWOLS) was followed. Assumes blip and treatment-free models correctly specified.

Usage

```
## S3 method for class 'DTRreg'
predict(object, newdata, treat.range = NULL, ...)
```

Arguments

`object`	A model object generated by the function DTRreg.
`newdata`	A dataset (usually the data analyzed by DTRreg for which predicted outcomes are desired. If a new dataset is provided, variable names should correspond to those presented to DTRreg
`treat.range`	If treatment is continuous (rather than binary), a list of vectors of the form c(min,max) which specify the minimum and maximum value the treatment may take. If unspecified, this will be inferred from the treat.range provided with use of the original DTRreg command. As such, if no treatment range was specified there either, treat.range will be the minimum and maximum observed treatment value at each stage.
`...`	Space for additional arguments (not currently used by DTRreg)

Details

This function may be used in a similar fashion to more traditional modelling commands (such as lm). Users are referred to the primary DTRreg help command (and associated literature) for information concerning model specification. In particular, we note that the predict function assumes that the treatment-free model has been correctly specified, as the treatment-free parameters are used in the prediction process.

Value

An n x 1 matrix of predicted outcome values.

Author(s)

Michael Wallace

References

Chakraborty, B., Moodie, E. E. M. (2013) *Statistical Methods for Dynamic Treatment Regimes*. New York: Springer.

Robins, J. M. (2004) *Optimal structural nested models for optimal sequential decisions*. In Proceedings of the Second Seattle Symposium on Biostatistics, D. Y. Lin and P. J. Heagerty (eds), 189–326. New York: Springer.

Wallace, M. P., Moodie, E. M. (2015) Doubly-Robust Dynamic Treatment Regimen Estimation Via Weighted Least Squares. *Biometrics* **71**(3), 636–644 (doi:10.1111/biom.12306.)

Examples

```
##################
# example single run of a 2-stage g-estimation analysis
set.seed(1)
# expit function
expit <- function(x) {1 / (1 + exp(-x))}
# sample size
n <- 10000
# variables (X = patient information, A = treatment)
X1 <- rnorm(n)
A1 <- rbinom(n, 1, expit(X1))
X2 <- rnorm(n)
A2 <- rbinom(n, 1, expit(X2))
# blip functions
gamma1 <- A1 * (1 + X1)
gamma2 <- A2 * (1 + X2)
# observed outcome: treatment-free outcome plus blip functions
Y <- exp(X1) + exp(X2) + gamma1 + gamma2 + rnorm(n)
# models to be passed to DTRreg
# blip model
blip.mod <- list(~X1, ~X2)
# treatment model (correctly specified)
treat.mod <- list(A1~X1, A2~X2)
# treatment-free model (incorrectly specified)
tf.mod <- list(~X1, ~X2)

# perform G-estimation
mod1 <- DTRreg(Y, blip.mod, treat.mod, tf.mod, method = "gest")

# predicted Y for optimal treatment
dat <- data.frame(X1,X2,A1,A2)
predict(mod1, newdata = dat)
##################
```

Index