Package 'OptimalTiming'

July 21, 2025

Type Package

Title Optimal Timing Identification

Version 0.1.0

Author Xiao Lin <xlin3@mdanderson.org>, Xuelin Huang<xlhuang@mdanderson.org>

Maintainer Xiao Lin <xlin3@mdanderson.org>

Description Identify the optimal timing for new treatment initiation during multiple state disease transition, including multistate model fitting, simulation of mean residual lifetime for a given transition state, and estimation of confidence interval. The method is referred to de Wreede, L., Fiocco, M., & Putter, H. (2011) <doi:10.18637/jss.v038.i07>.

Depends R (>= 3.0.0)

Imports mstate, survival

License GPL (>= 2)

Encoding UTF-8

LazyData true

LazyLoad true

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-11-14 15:52:34 UTC

Contents

conf.MTL *Confidence interval of mean total lifetime*

Description

This function is used to calculate confidence intervals of mean total lifetime using jackknife resampling.

Usage

```
conf.MTL(obj, state = NULL, nsim = 1000, L = 120)
```

Arguments

obj	An object returned by optim.fit, which contains the transition probabilities and other information used to simulate mean total lifetime.
state	A numeric vector indicating from which state the mean total lifetime is simulated. Default is NULL, where no mean total life for a specific state is output. If obj is returned by optim.fit with treatment=NULL, there is no need to set this argument.
nsim	The times of simulation for mean total life. The default is 1000.
L	The prespecified threshold for blocking the increase of residual lifetime. The default is 120.

Details

This function systematically leaves out each subject from the original datset and simulates mean total lifetimes for each n-1-sized subsample. The jackknife mean and variance are calculated by aggregating n simulated mean total lifetimes. For each jackknife dataset, mean total lifetime is simulated using the algorithm described in sim.MTL.

Value

If the input object comes from optim.fit with treatment=NULL, a list object with elements:

conf.state.MTL	A data frame containing states, corresponding mean total lifetime, standard error and 95% confidence interval. If state=NULL, this element does not exist.
state.table	The correspondence of state number and state label.

If the input object comes from optim.fit with treatment is not NULL, a list object with elements:

conf.strategies	Mean total lifetime for different strategies, along with standard error and 95% confidence interval

See Also

optim.fit

Examples

```
## Not run:
library(OptimalTiming)

###################################
## Example 1: This example shows how to calculate confidence
## intervals for different treatment strategies

## read data
data(SimCml)

## fit multistate model with treatment not equals NULL
fit=optim.fit(data=SimCml,
       transM=matrix(c(0,1,0,0,0,1,0,0,0,1,0,1,1,1,0,0,0,1,1,1,1,
       0,0,0,0,1,1,1,0,0,0,0,0,1,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0),7,byrow=TRUE),
       nstate=7,state_label=c("diagnose","cp1","ap","cp2","bc","sct","death"),
       event_label=c("cp1.s","ap.s","cp2.s","bc.s","sct.s","death.s"),
       treatment=c("sct","sct.s"),absorb=c("death","death.s"),
       cov=c("age"),cov_value=c(0))

## compare different treatment strategies
conf.MTL(obj=fit,nsim=1000,L=120)

###################################
## Example 2: This example shows how to calculate confidence
## intervals for a given state

## read data
data(SimCml)

## delete the information of transplant time
data=SimCml[SimCml$sct.s==0,]
del=which(names(SimCml)%in%c("sct","sct.s"))
data=data[,-del]

## fit multistate model with treatment equals NULL
fit=optim.fit(data=data,
        transM=matrix(c(0,1,0,0,0,0,0,0,1,0,1,1,0,0,0,
        1,1,1,0,0,0,0,1,1,0,0,0,0,0,1,0,0,0,0,0,0),6,byrow=TRUE),
        nstate=6,state_label=c("diagnose","cp1","ap","cp2","bc","death"),
     absorb=c("death","death.s"),event_label=c("cp1.s","ap.s","cp2.s","bc.s","death.s"),
        cov=c("age"),cov_value=c(0))

## calculate mean total lifetime and confidence intervals
## for state 1,2,3,4
conf.MTL(obj=fit,state=c(1,2,3,4),nsim=1000,L=120)
## End(Not run)
```

`optim.fit` *Fit multi-state model for optimization*

Description

This function produces transition probabilities for given covariates values in multi-state models.

Usage

```
optim.fit(data, transM, nstate, state_label, event_label, treatment = NULL,
  absorb, cov, cov_value)
```

Arguments

`data`	Data frame in wide format where each row in the data corresponds to a single subject. Time to a state and the occurrence of the state come in pairs. If a state is not occur, use the time to an absorbing state or censoring time instead. Covariates are added at the end of each row.
`transM`	A $nstate \times nstate$ matrix used to indicate the transitions in multi-state model. If a transition exists between two states, set 1 in a corresponding location, otherwise set 0.
`nstate`	Number of states incorporated in the multi-state model.
`state_label`	A character vector of length `nstate` containing the names of states. The elements in `state_label` are extracted from the column names of `data`, except for the first one, which is a potential state at the initiation of a study for each subject. Assume all subjects have the same initial state.
`event_label`	A character vector of length `nstate-1`, indicating the occurrence of each state. The first state in `state_label` do not need an indiator, as it always exists.
`treatment`	A character vector of length 2, indicating whether there is a treatment variable available. If true, the name and indicator of this treatment extracted from `state_label` and `event_label` consist of `treatment`. If not, `treatment`=NULL. See details. The default value is NULL.
`absorb`	A character vector of length 2, indicating the name and indicator of the absorb state.
`cov`	A character vector containing the names of covariates that have some effect to transition probabilities.
`cov_value`	A numeric vector containing the values of covariates corresponding to `cov`. `cov_value` are used to calculated subject specified transition probabilities.

Details

For optim.fit, transition probabilities are estimated under Markov assumption, which implies that the probability of transition to a future state depends only on the present state, not on the history. Taking covariates at baseline into consideration, transition probabilities can be subject-specific. Cox

proportional hazards model is used to fit transition hazards among multiple states by assuming each transition has its own baseline hazard, and covariates have different effects on different transitions.

Let $\mathbf{S} = 1, 2, \cdots, S$ denote the states in the multi-state model and $X(t)$ be a random process taking values from $\mathbf{S}$. Denote $\alpha_{gh}(t)$ as hazard ratio or transition intensity and Z as baseline covariates. The instantaneous risk of a transition from state g into state h at time t can be fitted by semi-parametric Cox model:

$$\alpha_{gh}(t|Z) = \alpha_{gh,0} exp(\beta^T Z_{gh}).$$

The cumulative hazard ratio is defined as $A_{gh}(t) = \int_0^t \alpha_{gh}(u)du$. Primary interest in this function is to estimate transition probability $P_{gh}(s,t) = P(X(t) = h|X(s) = g)$, indicating the chance of transition from state g at time s to state h at time t. Written in matrix form, transition probability matrix $\mathbf{P}(t)$ can be calculated by means of a product integral: $\mathbf{P}(s,t) = \prod_{(s,t]}(\mathbf{I} + d\mathbf{A}(u))$, where $\mathbf{A}(t)$ is a transition intensity matrix. Both $\mathbf{P}$ and $\mathbf{A}$ are $S \times S$ matrix.

The data format required by this function is wide format, which can be regarded as the augmented data used in single event survival analysis. For example, if there is a "recurrence" state in a multi-state model, two variable are needed to describe this event, namely, "rec" and "rec.s". The former is a time variable, indicating the time from initiation of the study to the occurrence of this state, while the latter is an indicator variable with 1 for occurrence and 0 for censoring. If the event is censored for some patients, use the maximum follow-up instead of the event time. Other states are prepared in the same way. Thus, each row in the augmented data summarize all possible events for a single subject. For covariates, they are located at the end of each row.

If the time of new treatment initiation is provided in data, the argument `treatment` should be assigned as, eg.`treatment=c("sct","sct.s")`. Additionally, the argument `state_lable` and `event_label` shoud be arranged in such order: pre-treatment state, treatment state, post-treatment states and absorbing state. Assume treatment may take place at any pre-treatment states. In this case, `optim.fit` function automatically fit two multistate models, one for post-treatment states if a new treatment is carried out , and the other for pre-treatment states if a new treatment is not carried out. Thus, comparison among strategies of whether and when to initiate new treatment can be performed in `sim.MTL` function. If `treatment=NULL`, a single multistate model will be fitted.

Value

If `treatment` is NULL, a list object called "overall" is output with elements:

`transMat`	A transition matrix describing the states and transitions in multi-state model.
`tranProb`	A list of size `nstate` recording the transition probabilities form each state to another along with standard errors. Element `[[s]]` is a data frame containing transition probabilities from state `s` to state `1,2,...,nstate`.These transition probabilities are time-varying over distinct transition time points.
`coxobj`	An object returned by `coxph()` function in `survival` package.
`ntrans`	The number of available transitions among multiple states.
`...`	Other variables that extracted from the original input.

If `treatment` is not NULL, three list objects called "overall", "treat","no_treat" are output. A list "overall" contains elements:

`transMat`	A transition matrix describing the states and transitions in multi-state model.
`ntrans`	The number of available transitions among multiple states.

...	Other variables that extracted from the original input.

A list "no_treat" contains elements by fitting a model for pre-treatment states:

transMat	A transition matrix describing the states and transitions if the new treatment is not carried out.
tranProb	A list recording the transition probabilities among pre-treatment states.
coxobj	An object returned by coxph() function in survival package.
ntrans	The number of available transitions among pre-treatment states.
data	A data set contaning states if the new treatment is not carried out.
nstate	The number of pre-treatment states.
...	Other variables that extracted from the original input.

A list "treat" contains elements by fitting a model if a new treatment is carried out:

transMat	A transition matrix describing the states and transitions if the new treatment is carried out.
tranProb	A list recording the transition probabilities among post-treatment states.
coxobj	An object returned by coxph() function in survival package.
ntrans	The number of available transitions among post-treatment states.
data	A data set contaning states if the new treatment is carried out.
nstate	The number of post-treatment states.
...	Other variables that extracted from the original input.

References

de Wreede LC, Fiocco M, and Putter H (2010). The mstate package for estimation and prediction in non- and semi-parametric multi-state and competing risks models. Computer Methods and Programs in Biomedicine 99, 261–274.

Examples

```
## Not run:
library(OptimalTiming)
## read data
data(SimCml)

## fit multistate model if the time to new treatment initiation is available in SimCml
fit=optim.fit(data=SimCml,
         transM=matrix(c(0,1,0,0,0,1,0,0,0,1,0,1,1,1,0,0,0,1,1,1,1,
         0,0,0,0,1,1,1,0,0,0,0,0,1,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0),7,byrow=TRUE),
         nstate=7,state_label=c("diagnose","cp1","ap","cp2","bc","sct","death"),
         event_label=c("cp1.s","ap.s","cp2.s","bc.s","sct.s","death.s"),
         treatment=c("sct","sct.s"),absorb=c("death","death.s"),
         cov=c("age"),cov_value=c(0))

## view the content of this object
objects(fit)
```

```
## output transition probabilities
fit$treat$tranProb
fit$no_treat$tranProb
## End(Not run)
```

`sim.MTL` *Simulate mean total lifetime*

Description

This function is used to simulate mean total lifetime for a given initial state according to the estimated transition probabilities.

Usage

```
sim.MTL(obj, state = NULL, nsim = 1000, L = 120)
```

Arguments

`obj`	An object returned by `optim.fit`, which contains the transition probabilities and other information used to simulate mean total lifetime.
`state`	A numeric vector indicating from which state the mean total lifetime is simulated. Default is NULL, where no mean total life for a specific state is output. If obj is returned by optim.fit with treatment=NULL, there is no need to set this argument.
`nsim`	The times of simulation. The default is 1000.
`L`	The prespecified threshold for blocking the increase of residual lifetime. The default is 120. See Details.

Details

This part describes the algorithm used to simulate mean total lifetime in detail. For an initial state, we first extract the transition probability data frame for this state, and cumulate probabilities in row direction. In the transformed data frame, the first column of which impels the time points where transition probabilities are measured, and the last column is accumulated to be 1, indicating the addition of the chance to depart from a state and the chance to remain at the state equals 1. Then, we generate a random value from uniform distribution `Unif(0,1)` to determine the next state by comparing this value to cumulative probabilities. The state, whose cumulative transition probability from initial state firstly surpasses the uniform value, is defined as the next state. The time interval, from the initiation of the study to where the transition take place, is defined as the interim residual life. These two variables (next state and interim residual life) are recorded for late use. Subsequently, we regard the next state as the initial state, and repeat this searching process until the absorbing state is reached or the interim residual lifetime surpasses the prespecified threshold (`L`). Finally, the mean total life is either the last interim residual lifetime from the initiation of study to

the occurrence of absorbing state, or the prespecified threshold (L) if the absorbing has not reached yet.

If a state is given in this function, we set this state as initial state and perform the algorithm mentioned above for nsim times, and average the output to obtain mean total lifetime.

According to different type of input object, this function return different results. If the object comes from optim.fit with treatment=NULL, this function is used to simulate mean total lifetime for a given state. If the object comes from optim.fit with treatment not equals NULL, this function is used to compare mean total lifetimes of subjects who receive the new treatment to those who do not receive the new treatment.

Value

If the input object comes from optim.fit with treatment=NULL, a list object with elements:

state.MTL	A data frame containing states and corresponding mean total lifetime. If state=NULL, this element does not exist.
state.table	The correspondence of state number and state label.

If the input object comes from optim.fit with treatment not equals NULL, a list object with elements:

strategies	Mean total lifetime for different strategies.

See Also

optim.fit

Examples

```
## Not run:
library(OptimalTiming)

###################################
## Example 1: This example shows how to use this package to find
## the optimal timing of new treatment initiation

## read data
data(SimCml)

## fit multistate model with treatment not equals NULL
fit=optim.fit(data=SimCml,
         transM=matrix(c(0,1,0,0,0,1,0,0,0,1,0,1,1,1,0,0,0,1,1,1,1,0,0
         ,0,0,1,1,1,0,0,0,0,0,1,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0),7,byrow=TRUE),
         nstate=7,state_label=c("diagnose","cp1","ap","cp2","bc","sct","death"),
         event_label=c("cp1.s","ap.s","cp2.s","bc.s","sct.s","death.s"),
         treatment=c("sct","sct.s"),absorb=c("death","death.s"),
         cov=c("age"),cov_value=c(0))

## compare different treatment strategies
sim.MTL(obj=fit,nsim=1000,L=120)
```

```
##################################
## Example 2: This example shows how to obtain mean total lifetime
## for a given state

## read data
data(SimCml)

## delete the information of transplant time
data=SimCml[SimCml$sct.s==0,]
del=which(names(SimCml)%in%c("sct","sct.s"))
data=data[,-del]

## fit multistate model with treatment equals NULL
fit=optim.fit(data=data,
        transM=matrix(c(0,1,0,0,0,0,0,0,1,0,1,1,0,0,0,1,
        1,1,0,0,0,0,1,1,0,0,0,0,0,1,0,0,0,0,0,0),6,byrow=TRUE),
        nstate=6,state_label=c("diagnose","cp1","ap","cp2","bc","death"),
        absorb=c("death","death.s"),
        event_label=c("cp1.s","ap.s","cp2.s","bc.s","death.s"),
        cov=c("age"),cov_value=c(0))

## calculate mean total lifetime when the initiate state is cp1 or ap
sim.MTL(obj=fit,state=c(2,3),nsim=1000,L=120)
## End(Not run)
```

SimCml *Simulated data for CML patients*

Description

A dataset containing information of CML patients who have received transplant. Both states before and after transplant are included in this data set. This data set is used for illustration, so the magnitude of event times are out of clinical consideration.

Format

A data frame of 1777 rows (patients) on the following 14 variables:

cp1 Time in months to chronic phase

cp1.s Indicator of the occurrence of chronic phase; 1=occur, 0=censored

ap Time in months to accelerated phase of CML

ap.s Indicator of the occurrence of accelerated phase; 1=occur, 0=censored

cp2 Time in months to chronic phase after progression to advanced stage

cp2.s Indicator whether subject come back to chronic phase after progression to an advanced stage; 1=occur, 0=censored

bc Time in months to blast crisis phase of CML

bc.s Indicator of the occurrence of blast crisis phase; 1=occur, 0=censored

sct Time in months to receive transplant

sct.s Indicator whether patient receive transplant; 1=transplant, 0=no transplant

death Time in months to death

death.s Death indicator; 1=death, 0=censored

age 0=age less than 50; 1=age larger than 50

sex 1=female; 2=male

Examples

```
data(SimCml)
```

Index