Package ‘ebmstate’

July 22, 2025

Type Package

Title Empirical Bayes Multi-State Cox Model

Version 0.1.5

Description
Implements an empirical Bayes, multi-state Cox model for survival analysis. Run ``?'ebmstate-package''' for details. See also Schall (1991) <doi:10.1093/biomet/78.4.719>.

Depends R (>= 3.6.0), survival (>= 2.44-1.1), mstate (>= 0.2.11)

Imports Rcpp, HDInterval, stats, utils, methods

License GPL (>= 3)

LinkingTo Rcpp

RoxygenNote 7.2.3

Encoding UTF-8

LazyData true

NeedsCompilation yes

Author Rui Costa [aut, cre],
Moritz Gerstung [aut],
Terry M Therneau [ctb] (author of 'survival', a package from which code parts were copied),
Thomas Lumley [ctb] (contributor to 'survival', a package from which code parts were copied),
Hein Putter [ctb] (co-author of 'mstate', a package from which code parts were copied),
Liesbeth de Wreede [ctb] (co-author of 'mstate', a package from which code parts were copied),
Marta Fiocco [ctb] (co-author of 'mstate', a package from which code parts were copied),
Ronald Geskus [ctb] (contributor to 'mstate', a package from which code parts were copied)

Maintainer Rui Costa <ruibarrigana@hotmail.com>

Repository CRAN

Date/Publication 2024-10-19 14:20:01 UTC

Contents

`ebmstate-package` *Empirical Bayes multi-state Cox model*

Description

This package implements an empirical Bayes, multi-state Cox model. Different groups of regression coefficients can be defined, with coefficients of the same group sharing the same Gaussian prior. It takes as input a data set in 'long format' and generates estimates of relative hazards, cumulative hazard functions and transition probabilities. It relies on packages `survival` and `mstate` and incorporates some of their functions to reduce upstream dependency.

Details

Package:	ebmstate
Type:	Package
Version:	0.0.73
Date:	2020-01-21
License:	GPL 3

Author(s)

Rui Costa, Moritz Gerstung

`boot_coxrfx` *Bootstrap confidence intervals for regression coefficients*

Description

This function computes 95% highest density bootstrap confidence intervals (non-parametric) for the regression coefficients estimated by CoxRFX.

Usage

```
boot_coxrfx(
  mstate_data_expanded,
  which_group,
  min_nr_samples = 100,
  output = "CIs",
  ...
)
```

Arguments

`mstate_data_expanded`	Data in 'long format', possibly with 'expanded' covariates (as obtained by running mstate::expand.covs).
`which_group`	A character vector with the same meaning as the 'groups' argument of the function `CoxRFX` but named (with the covariate names).
`min_nr_samples`	The confidence interval of any coefficient is based on a number of bootstrap samples at least as high as this argument. See details.
`output`	Determines the sort of output. See value.
`...`	Further arguments to the CoxRFX function.

Details

In a given bootstrap sample there might not be enough information to generate estimates for all coefficients. If a covariate has little or no variation in a given bootstrap sample, no estimate of its coefficient will be computed. The present function will keep taking bootstrap samples until every coefficient has been estimated at least `min_nr_samples` times.

Value

For each regression coefficient, the confidence intervals and the number of bootstrap samples on which they are based, if the ‘output‘ argument is equal to ‘CIs‘; if ‘output‘ is equal to ‘CIs_and_coxrfx_fits‘, also the `CoxRFX` objects for each bootstrap sample.

Author(s)

Rui Costa

`boot_ebmstate` *Bootstrap samples and bootstrap interval estimates*

Description

This function computes bootstrap samples of regression coefficients, cumulative hazard functions, and transition probability functions.

Usage

```
boot_ebmstate(
  mstate_data = NULL,
  which_group = NULL,
  min_nr_samples = NULL,
  patient_data = NULL,
  initial_state = NULL,
  tmat = NULL,
  time_model = NULL,
  backup_file = NULL,
  input_file = NULL,
  coxrfx_args = NULL,
  msfit_args = NULL,
  probtrans_args = NULL
)
```

Arguments

`mstate_data`	A data frame with outcome and covariate data in long format.
`which_group`	A character vector with the same meaning as the ‘groups‘ argument of the function `CoxRFX` but named (with the covariate names).

`min_nr_samples`	The confidence interval of any coefficient is based on a number of bootstrap samples at least as high as this argument. See details.
`patient_data`	The covariate data for which the estimates of cumulative hazards and transition probabilities are computed. Must contain: one row of data for each transition, all the covariate columns in the fitted model, and also the 'strata' column.
`initial_state`	The initial state for which transition probability estimates should be computed
`tmat`	Transition matrix for the multi-state model, as obtained by running `mstate::transMat`
`time_model`	The model of time-dependency: either 'clockforward' or 'clockreset'.
`backup_file`	Path to file. Objects generated while the present function is running are stored in this file. This avoids losing all estimates if and when the algorithm breaks down. See argument `input_file`.
`input_file`	Path to `backup_file` (see argument `backup_file`). If this argument is given, all other arguments should be `NULL`.
`coxrfx_args`	Named list with arguments to the `CoxRFX` function other than `Z`,`surv` and `groups`.
`msfit_args`	Named list with arguments to the `msfit_generic.coxrfx` function other than `object`,`newdata` and `trans`.
`probtrans_args`	Named list with arguments to the `probtrans_ebmstate` function other than `initia_state`,`cumhaz` and `model`.

Details

In a given bootstrap sample there might not be enough information to generate estimates for all coefficients. If a covariate has little or no variation in a given bootstrap sample, no estimate of its coefficient will be computed. The present function will keep taking bootstrap samples until every coefficient has been estimated at least `min_nr_samples` times. `covariate_df` should only contain the covariates of the model one wishes to estimate.

Value

A list with: 95% bootstrap intervals for each regression coefficient and for transition probabilities; bootstrap samples of regression coefficients, cumulative hazards and transition probabilities.

Author(s)

Rui Costa

`boot_probtrans` *Bootstrap confidence intervals for transition probabilities*

Description

Generates 95% highest density bootstrap interval estimates for transition probabilities computed using `probtrans_ebmstate` (clock-reset version).

Usage

```
boot_probtrans(coxrfx_fits_boot, patient_data, tmat, initial_state, max_time)
```

Arguments

`coxrfx_fits_boot`	The list of CoxRFX objects obtained by running `boot_coxrfx`.
`patient_data`	(Single) patient data in 'long format', possibly with 'expanded' covariates (as obtained by running `mstate::expand.covs`).
`tmat`	Transition matrix for the multi-state model, as obtained by running `mstate::transMat`
`initial_state`	The initial state for which transition probability estimates should be computed
`max_time`	The maximum time for which estimates should be computed

Value

Interval estimates for transition probabilities.

Author(s)

Rui Costa

See Also

`probtrans_ebmstate`; `boot_coxrfx`; `transMat`; `expand.covs`

`CIs_for_target_state` *Ancillary function of* `boot_ebmstate`.

Description

Computes 95% highest density bootstrap confidence intervals for the transition probabilities into `target_state`, given a list object with boostrap estimates of transition probabilities into multiple states. This function is not meant to be called by the user.

Usage

```
CIs_for_target_state(target_state, probtrans_objects_boot)
```

Arguments

`target_state`	The target state for whose transition probabilties the confidence intervals are computed.
`probtrans_objects_boot`	A list containing bootstrap estimates of transition probabilities.

Details

Uses function `extract_function`.

Value

95% highest density bootstrap confidence intervals for the transition probabilities into `target_state`.

Author(s)

Rui Costa

See Also

`boot_ebmstate`; `extract_function`.

`convolute_clockforward` *Convolution function for clock-forward models*

Description

Internal function of `probtrans_by_convolution_clockforward`. It is written in C++ and is not meant to be called directly by the user.

Usage

```
convolute_clockforward(
  time_vector,
  diff_vector,
  probtrans_vector_1,
  probtrans_vector_2
)
```

Arguments

`time_vector`, `diff_vector`, `probtrans_vector_1`, `probtrans_vector_2`
Numeric vectors.

Author(s)

Moritz Gerstung & Rui Costa

See Also

`probtrans_by_convolution_clockforward`.

convolute_clockreset *Convolution function for clock-reset models*

Description

Internal function of `probtrans_by_convolution_clockreset`. It is written in C++ and is not meant to be called directly by the user.

Usage

```
convolute_clockreset(time_vector, integrand_1, integrand_2)
```

Arguments

`time_vector`, `integrand_1`, `integrand_2`	Numeric vectors.

Author(s)

Moritz Gerstung & Rui Costa

See Also

`probtrans_by_convolution_clockreset`.

CoxRFX *Empirical Bayes, multi-state Cox model*

Description

This function estimates a multi-state Cox model with one or more Gaussian priors imposed on the regression coefficients (see Therneau et al., 2003). Multiple groups of coefficients can be defined: coefficients within a group share the same (possibly unknown) mean and variance. The parameters and hyperparameters are efficiently estimated by an EM-type algorithm built around the function `survival::coxph`.

Usage

```
CoxRFX(
  Z,
  surv,
  groups = rep(1, ncol(Z)),
  which.mu = unique(groups),
  tol = 0.001,
  max.iter = 50,
  sigma0 = 0.1,
  sigma.hat = c("df", "MLE", "REML", "BLUP"),
  verbose = FALSE,
  ...
)
```

Arguments

`Z`	A data frame consisting of the covariate columns of a data set in 'long format', and two extra columns: one named 'trans', with the transition that each row refers to, and another named 'strata', with the stratum of each transition (transitions belonging to the same stratum are assumed to have the same baseline hazard function).
`surv`	A 'survival' object created with `survival::Surv`.
`groups`	A character or numeric vector whose ith element gives the group of the regression coefficient associated with the ith covariate column of Z (coefficients belonging to the same group share the same Gaussian prior).
`which.mu`	A vector with names or numbers of coefficient groups (see argument `groups`). If the name or number of a group of coefficients is given in this argument, `CoxRFX` will estimate the mean of its Gaussian distribution; otherwise the mean will be fixed at zero.
`tol`	Convergence criterium of the EM algorithm. The algorithm stops unless there is at least one parameter (or hyperparameter) for which it holds that the current estimate differs in absolute terms by more than `tol` from the previous estimate.
`max.iter`	The maximum number of iterations in the EM algorithm.
`sigma0`	A vector with the initial value of the variance hyperparameter for each group of coefficients. Or a single value, in case the initial value of the variance hyperparameter is meant to be the same for all groups.
`sigma.hat`	Which estimator to use for the variance hyperparameters (see details).
`verbose`	Gives more output.
`...`	Further arguments passed to the function `survival::coxph`.

Details

Different estimators exist for the variance hyperparameters: the default is "df", as used by Perperoglou (2014) and introduced by Schall (1991). Alternatives are MLE, REML, and BLUP, as defined by Therneau et al. (2003). Simulations suggest that the 'df' method is the most accurate.

The model can also be fitted using package `coxme`; the `coxme` routine numerically optimises the integrated partial likelihood, which may be more accurate, but is computationally expensive.

Value

An object of class `c(coxrfx,coxph.penal,coxph)`, which is essentially a `coxph` object with a few extra fields [the inputs $groups, $Z and $surv, and the hyperparameters $sigma2 (variances) and $mu (means)]. See `survival::coxph.object`.

Author(s)

Moritz Gerstung & Rui Costa, extending the work of Terry Therneau et al. in the package `survival`.

References

Terry M Therneau, Patricia M Grambsch & V. Shane Pankratz (2003) Penalized Survival Models and Frailty, Journal of Computational and Graphical Statistics, 12:1, 156-175, http://dx.doi.org/10.1198/1061860031365

A. Perperoglou (2014). Cox models with dynamic ridge penalties on time-varying effects of the covariates. Stat Med, 33:170-80. http://dx.doi.org/10.1002/sim.5921

R. Schall (1991). Estimation in generalized linear models with random effects. Biometrika, 78:719-727. http://dx.doi.org/10.1093/biomet/78.4.719

See Also

Package `survival` `survival::coxph.object`; `survival::Surv`; package `coxme`.

Examples

```
# Fit an empirical Bayes Cox model using
# simulated, illness-death data from 250
# patients ('mstate_data_sample').

#load simulated data
data("mstate_data_sample")

# Set class of ‘mstate_data_sample’
class(mstate_data_sample)<-c("data.frame","msdata")

# add transition matrix as attribute
tmat<-mstate::transMat(x=list(c(2,3),c(4),c(),c()),
      names=c("health","illness","death",
     "death_after_illness"))
attr(mstate_data_sample,"trans")<-tmat

# expand covariates by transition:
covariates.expanded<-mstate::expand.covs(
      mstate_data_sample,
      covs=names(mstate_data_sample)
      [!names(mstate_data_sample)%in%c("id","from",
      "to","trans","Tstart","Tstop","time","status",
      "strata")],append=FALSE)


# argument ‘Z’ of coxrfx
Z<-data.frame(covariates.expanded,
   trans=mstate_data_sample$trans,
   strata=mstate_data_sample$trans)

# argument ‘surv’ for a non-homogeneous
# Markov model
surv<-survival::Surv(mstate_data_sample$Tstart,
                mstate_data_sample$Tstop,
                mstate_data_sample$status)

# argument ‘groups’ of coxrfx
groups<-paste0(rep("group", ncol(Z)-2),c("_1","_2","_3"))

#fit random effects model
coxrfx_object<-CoxRFX(Z,surv,groups)

#show point estimates
summary(coxrfx_object)
```

---

coxrfx_object_sample *Example of an empirical Bayes model fit*

Description

An RData object containing the model fit obtained by running `CoxRFX` on the data set `mstate_data_sample` (included in the present package).

Usage

```
coxrfx_object_sample
```

Format

An object of class `c(coxrfx,coxph.penal,coxph)`, which is essentially a coxph object with a few extra fields [the inputs $groups, $Z, and $surv, and the hyperparameters $sigma2 (variances) and $mu (means)].

See Also

`mstate_data_sample`; `CoxRFX`.

---

cumhazCIs_for_target_transition *Ancillary function of* `boot_ebmstate`.

Description

Computes 95% highest density, non-parametric bootstrap confidence intervals for the cumulative hazard rate functions, given a list of `msfit` objects with boostrap estimates of cumulative hazard rate functions for multiple transitions. This function is not meant to be called by the user.

Usage

```
cumhazCIs_for_target_transition(transition, msfit_objects_boot)
```

Arguments

`transition`	The transition for which transition confidence intervals are computed.
`msfit_objects_boot`	List of `msfit` objects with boostrap estimates of cumulative hazard rate functions for multiple transitions.

Value

95% highest density, non-parametric bootstrap confidence intervals for the cumulative hazard rate functions.

Author(s)

Rui Costa

See Also

`boot_ebmstate`.

`cumhaz_splines` *Spline approximations of the cumulative hazard functions*

Description

Creates a spline approximation for the vector of cumulative hazards of each transition.

Usage

```
cumhaz_splines(cumhaz)
```

Arguments

`cumhaz`	An object of class `msfit`, created by `msfit_generic` or `msfit`.

Details

This function is used by the function `probtrans_by_convolution`. It is not meant to be called by the user.

Value

A list of estimated cumulative hazard functions (one for each transition).

Author(s)

Rui Costa

See Also

`msfit_generic`; `msfit`; `probtrans_by_convolution`.

`extract_function` *Ancillary function to* `boot_ebmstate`.

Description

Extracts the bootstrap estimates of transition probabilities for target state 'tstate' from a list with bootstrap estimates of transition probabilities into multiple states. This function is not meant to be called by the user.

Usage

```
extract_function(list_object, tstate)
```

Arguments

`list_object`	A list in which each individual element is a single bootstrap estimate of the probability of transition into different states.
`tstate`	The state whose bootstrap estimates of transition probabilities we wish to extract from `list_object`.

Details

This function is an ancillary function of `CIs_for_target_state`, which in turn is an ancillary function of `boot_ebmstate`.

Value

Bootstrap estimates of transition probabilities into target state 'tstate'.

Author(s)

Rui Costa

See Also

`CIs_for_target_state`; `boot_ebmstate`

joint_cum_hazard_function

Compute the cumulative hazard of leaving a given state

Description

This function is not meant to be called by the user. It is an internal function of `probtrans_by_convolution_clockforward` and `probtrans_by_convolution_clockreset`.

`joint_cum_hazard_function` returns the cumulative hazard of leaving state `i` to any state that can be reached directly from `i`, at each of the time points in `t`. There is no explicit argument `i`: this state is entirely defined by the transitions that can occur when the patient is in it (and these transitions are given in the argument `competing_transitions`).

Usage

```
joint_cum_hazard_function(t, competing_transitions, spline_list)
```

Arguments

`t`	A vector of time points.
`competing_transitions`	The transitions that can occur when the process is in state `i`.
`spline_list`	A list whose elements are spline functions approximating the cumulative hazard of making each possible transition in the process. This is normally a list object created by running `cumhaz_splines`.

Value

A vector with the cumulative hazard of leaving a given state evaluated at given time points.

Author(s)

Rui Costa

See Also

`probtrans_by_convolution_clockforward`; `probtrans_by_convolution_clockreset`; `cumhaz_splines`.

`loo_ebmstate` *Leave-one-out estimation*

Description

This function computes leave-one-out estimation of regression coefficients, cumulative hazard functions, and transition probability functions.

Usage

```
loo_ebmstate(
  mstate_data,
  mstate_data_expanded,
  which_group,
  patient_IDs,
  initial_state,
  tmat,
  time_model,
  backup_file = NULL,
  input_file = NULL,
  coxrfx_args = list(),
  msfit_args = NULL,
  probtrans_args = NULL
)
```

Arguments

`mstate_data`	Data in ‘long format‘.
`mstate_data_expanded`	Data in ‘long format‘, possibly with ‘expanded‘ covariates (as obtained by running mstate::expand.covs).
`which_group`	A character vector with the same meaning as the ‘groups‘ argument of the function `CoxRFX` but named (with the covariate names).
`patient_IDs`	The IDs of the patients whose cumulative hazards and transition probabilities one wishes to estimate.
`initial_state`	The initial state for which transition probability estimates should be computed
`tmat`	Transition matrix for the multi-state model, as obtained by running `mstate::transMat`
`time_model`	The model of time-dependency: either ’clockforward’ or ’clockreset’.
`backup_file`	Path to file. Objects generated while the present function is running are stored in this file. This avoids losing all estimates if and when the algorithm breaks down. See argument `input_file`.
`input_file`	Path to `backup_file` (see argument `backup_file`). If this argument is given, all other arguments should be `NULL`.
`coxrfx_args`	Named list with arguments to the `CoxRFX` function other than `Z`,`surv` and `groups`.

`msfit_args`	Named list with arguments to the `msfit_generic.coxrfx` function other than `object`,`newdata` and `trans`.
`probtrans_args`	Named list with arguments to the `probtrans_ebmstate` function other than `initia_state`,`cumhaz` and `model`.

Details

In a given bootstrap sample there might not be enough information to generate estimates for all coefficients. If a covariate has little or no variation in a given bootstrap sample, no estimate of its coefficient will be computed. The present function will keep taking bootstrap samples until every coefficient has been estimated at least `min_nr_samples` times.

Value

A list with: 95% bootstrap intervals for each regression coefficient and for transition probabilities; bootstrap samples of regression coefficients, cumulative hazards and transition probabilities.

Author(s)

Rui Costa

`MakeInteger` *Convert factor to integer.*

Description

Convert factor to integer.

Usage

```
MakeInteger(v)
```

Arguments

`v`	A factor vector.

Details

An internal function of `CoxRFX`, not meant to called directly by the user.

Value

A data.frame with columns corresponding to levels in the factor.

Author(s)

Moritz Gerstung

See Also

CoxRFX

---

msfit_generic *Compute subject-specific transition hazards.*

Description

This function computes subject-specific or overall cumulative transition hazards for each of the possible transitions in the multi-state model. This help page is an adaptation of the `mstate::msfit` help page.

Usage

```
msfit_generic(object, ...)

## Default S3 method:
msfit_generic(
  object,
  newdata,
  variance = TRUE,
  vartype = c("aalen", "greenwood"),
  trans,
  ...
)

## S3 method for class 'coxrfx'
msfit_generic(object, newdata, trans, ...)
```

Arguments

`object`	An object describing the fit of a multi-state Cox model.
`...`	Further arguments
`newdata`	A data frame in ‘long format’. See details.
`variance`	A logical value indicating whether the (co-)variances of the subject-specific transition hazards should be computed.
`vartype`	A character string specifying the type of variances to be computed (so only needed if variance=TRUE).
`trans`	Transition matrix describing the states and transitions in the multi-state model. See `trans` in `msprep` for more detailed information.

Details

The purpose of `msfit_generic` is to be able to use `mstate::msfit` on model fit objects of class `coxrfx` (i.e. objects generated by `CoxRFX`). This can now be done with `msfit_generic.coxrfx`, which introduces minor modifications to `mstate::msfit`. In particular, it precludes `msfit` from computing the (co-)variances of transition hazard estimators, as this computation relies on asymptotic results for the fixed effects Cox model (see de Wreede et al, 2010, section 2.3.2). The method `msfit_generic.default` corresponds to the original `mstate::msfit` function. The data frame given as `newdata` input needs to have one row for each transition in the multi-state model, and one column for each covariate. An additional column strata (numeric) is needed to describe for each transition to which stratum it belongs. The name has to be `strata`, even if in the original coxph call another variable was used. See `msfit` for more details.

Value

An 'msfit' object. See `msfit` for details. If the S3 method `msfit_generic.coxrfx` is called, the returned object will be of class `c(msfit,coxrfx)`; otherwise, it will be of class `msfit`.

Author(s)

Rui Costa, adapting the work of L. de Wreede, M. Fiocco and H. Putter in the `mstate` package.

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.

See Also

`msfit`; `msprep`; `plot.msfit`.

Examples

```
# Compute cumulative hazard rates
# under a (pre-estimated) empirical Bayes Cox
# model.

#load simulated data (illness-death model,
#500 patients) and estimated empirical
# Bayes Cox model
data("mstate_data_sample")
data("coxrfx_object_sample")

# Make objects 'surv' and 'Z'
# with the data used in the estimation

#outcome data
surv<-coxrfx_object_sample$surv

#covariate data
```

```
Z<-coxrfx_object_sample$Z

# Build a data frame 'patient_data'
# with the covariate values for which
# cumulative hazards are to be computed
# (patient 1 covariate values in this case).
# 'patient_data' must have one row for each
# transition in the model
# and the same columns as 'Z'. The assignment
# of transitions to strata (made in the 'strata'
# column) must follow the original model in
# 'coxrfx_object_sample'.

patient_data<-mstate_data_sample[mstate_data_sample$id==1,
   ,drop=FALSE][rep(1,3),]
patient_data$strata<-patient_data$trans<-1:3
patient_data<-mstate::expand.covs(patient_data,
   covs=names(patient_data)[!names(patient_data)%in%
   c("id","from","to","trans","Tstart","Tstop","time",
   "to","trans","Tstart","Tstop","time","status",
   "strata")],append=TRUE)

# compute cumulative hazards
msfit_object<-msfit_generic(coxrfx_object_sample,
                            patient_data,
                            coxrfx_object_sample$tmat)

# show estimates
print(msfit_object)
```

---

msfit_object_sample *Estimated cumulative hazard rates under an empirical Bayes Cox model (example)*

Description

An RData object containing estimated cumulative hazards, obtained by running `msfit_generic` on the object `coxrfx_object_sample` (also included in the present package).

Usage

```
msfit_object_sample
```

Format

An object of class `c(msfit,coxrfx)`. See `msfit_generic` and `mstate::msfit` for details.

See Also

`coxrfx_object_sample`.

`mstate_data` *An example of long-format multistate data*

Description

An RData object containing disease progression data for a sample of 576 patients with myelodysplastic syndromes (MDS), as an example of long-format multistate data.

Usage

```
mstate_data
```

Format

A data frame.

`mstate_data_sample` *A simulated event-history data set*

Description

A data set generated by simulation from an illness-death Cox model. This is an object of double class 'data.frame' and 'msdata', whose 'trans' attribute is a transition matrix (`attr(mstate_data_sample,"trans")`).

Usage

```
mstate_data_sample
```

Format

A data frame with 649 rows and 18 variables (250 patients):

id patient identification number

from state in which the patient is

to state to which the patient is at risk of going to

trans transition ID number

Tstart when the risk of the transition started

Tstop the time at which the risk of the transition ended or the last follow-up time (whichever happened first)

time Tstop-Tstart

status did the transition occur at Tstop?

Cov1,Cov2,Cov3,Cov4,Cov5,Cov6,Cov7,Cov8,Cov9,Cov10 covariates

`print.coxrfx` *Print method for CoxRFX objects*

Description

This function implicitly calls summary.coxrfx().

Usage

```
## S3 method for class 'coxrfx'
print(x, ...)
```

Arguments

`x`	A `coxrfx` object
`...`	further arguments passed to or from other methods.

Details

Prints two data frames, one with hyperparameter estimates and another with regression coefficient estimates.

Value

Returns an invisible NULL object.

Author(s)

Moritz Gerstung & Rui Costa

`print.msfit` *Print method for* `msfit` *objects generated by* `msfit_generic`

Description

This method is a simple call to `print.default`. Its main purpose is to override `print.coxrfx` when printing an object of double class `msfit` and `coxrfx`.

Usage

```
## S3 method for class 'msfit'
print(x, ...)
```

Arguments

`x`	An object of class `msfit` or double class `msfit` and `coxrfx`.
`...`	Further arguments passed to or from other methods.

Value

The input object (an object of double class `msfit` and `coxrfx`).

Author(s)

Rui Costa

`probtrans_by_convolution` *Compute all transition probabilities from a given state using convolution*

Description

`probtrans_by_convolution` is an internal function of `probtrans_ebmstate` and is not meant to be called directly by the user. It is itself a wrapper for the functions `probtrans_by_convolution_clockforward` and `probtrans_by_convolution_clockreset`, which are the workhorses of the convolution algorithm.

Usage

```
probtrans_by_convolution(tmat, cumhaz, from_state, model, max_time, nr_steps)
```

Arguments

`tmat`	A transition matrix extracted from the cumhaz argument to `probtrans_ebmstate`.
`cumhaz`	`msfit` object (argument passed on from `probtrans_ebmstate`).
`from_state`	Initial state (argument passed on from `probtrans_ebmstate`).
`model`	'clockforward' or 'clockreset' (argument passed on from `probtrans_ebmstate`).
`max_time`	The maximum time for which transition probabilities are estimated.
`nr_steps`	The number of steps in the convolution algorithm (larger increases precision but makes it slower)

Details

For more information on the arguments of this function see `probtrans_ebmstate`.

Author(s)

Rui Costa & Moritz Gerstung

See Also

`probtrans_ebmstate`;`probtrans_by_convolution_clockforward`; `probtrans_by_convolution_clockreset`.

probtrans_by_convolution_clockforward

Compute transition probabilities under a clock-forward model using a convolution algorithm.

Description

Compute transition probabilities for a given starting state and target state under a clock-forward model, using a convolution algorithm.

`probtrans_by_convolution_clockforward` is an internal function of `probtrans_by_convolution` and is not meant to be called directly by the user.

Usage

```
probtrans_by_convolution_clockforward(
  tmat,
  cumhaz,
  from_state,
  to_state,
  spline_list,
  unique_paths_object,
  time
)
```

Arguments

`tmat`	Transition matrix.
`cumhaz`	`msfit` object.
`from_state`	Initial state.
`to_state`	Target state.
`spline_list`	A list whose elements are spline functions approximating the cumulative hazard of making each possible transition in the process. This is normally a list object created by running `cumhaz_splines`.
`unique_paths_object`	An object created by running `unique_paths`.
`time`	A vector of ordered time points.

Author(s)

Rui Costa & Moritz Gerstung

See Also

`probtrans_ebmstate`; `probtrans_by_convolution_clockreset`; `probtrans_by_convolution`; `unique_paths`; `cumhaz_splines`.

probtrans_by_convolution_clockreset

Compute transition probabilities under a clock-reset model using a convolution algorithm.

Description

Compute transition probabilities for a given starting state and target state under a clock-reset model with a single time scale (sojourn time), using a convolution algorithm.

`probtrans_by_convolution_clockreset` is an internal function of `probtrans_by_convolution` and is not meant to be called directly by the user.

Usage

```
probtrans_by_convolution_clockreset(
  tmat,
  cumhaz,
  from_state,
  to_state,
  spline_list,
  unique_paths_object,
  time
)
```

Arguments

`tmat`	Transition matrix.
`cumhaz`	`msfit` object.
`from_state`	Initial state.
`to_state`	Target state.
`spline_list`	A list whose elements are spline functions approximating the cumulative hazard of making each possible transition in the process. This is normally a list object created by running `cumhaz_splines`.
`unique_paths_object`	An object created by running `unique_paths`.
`time`	A vector of ordered time points.

Author(s)

Rui Costa & Moritz Gerstung

See Also

`probtrans_ebmstate`; `probtrans_by_convolution_clockforward`; `probtrans_by_convolution`; `unique_paths`; `cumhaz_splines`.

probtrans_ebmstate   *Compute subject-specific transition probabilities using convolution.*

Description

Compute subject-specific transition probabilities using convolution.

Usage

```
probtrans_ebmstate(
  initial_state,
  cumhaz,
  model,
  max_time = NULL,
  nr_steps = 10000
)
```

Arguments

initial_state	The present function estimates transition probabilities from the state given in this argument.
cumhaz	An `msfit` object created by running `mstate` or `mstate_generic`.
model	Either 'clockforward' or 'clockreset'. See details.
max_time	The maximum time for which transition probabilities are estimated.
nr_steps	The number of steps in the convolution algorithm (larger increases precision but makes it slower)

Details

The clock-forward model is a model in which the transition hazard rates depend only on time since the initiating event. The clock-reset model has a single time scale: the sojourn time in the current state.

The algorithm behind `probtrans_ebmstate` is based on the convolution of density and survival functions and is suitable for processes with a tree-like transition structure only.

Value

An object of class 'probtrans'. See the 'value' section in the help page of `mstate::probtrans`.

Author(s)

Rui Costa & Moritz Gerstung

See Also

`probtrans`;

Examples

```
# Compute transition probabilities
# from an object with (pre-estimated)
# cumulative hazard rates.

#load object with estimated
#cumulative hazard rates
data("msfit_object_sample")

#compute transition probabilities
probtrans_object<-probtrans_ebmstate("health",
   msfit_object_sample,"clockforward")
```

`probtrans_fft` *Compute subject-specific transition probabilities using a convolution algorithm based on the Fast Fourier transform.*

Description

Compute subject-specific transition probabilities using a convolution algorithm based on the Fast Fourier transform.

Usage

```
probtrans_fft(initial_state, cumhaz, max_time, nr_steps = 10000)
```

Arguments

`initial_state`	The present function estimates state occupation probabilities from the state given in this argument.
`cumhaz`	An `msfit` object created by running `mstate` or `mstate_generic`.
`max_time`	The maximum time for which transition probabilities are estimated.
`nr_steps`	The number of steps in the convolution algorithm (larger increases precision but makes it slower)

Details

The `time` argument is crucial for precision. The density of time points and the upper time limit should be increased until the estimated curves become stable. A useful rule of thumb is to set the upper time limit to a time point in which the probability of each transient state is zero and the probability of each absorbing state is constant.

For the same approximation grid, `probtrans_fft` doesn't always yield the same result as `probtrans_ebmstate` (semi-Markov version), even though they are meant to approximate exactly the same convolution. `probtrans_ebmstate` is sensitive to the grid interval size, but not such much to the maximum grid time. `probtrans_fft` is sensitive to both these parameters, as referred above.

The algorithm behind `probtrans_ebmstate` is based on the convolution of density and survival functions and is suitable for processes with a tree-like transition structure only.

Value

An object of class 'probtrans'. See the 'value' section in the help page of `mstate::probtrans`.

Author(s)

Rui Costa

See Also

`probtrans`; `probtrans_ebmstate`

`probtrans_mstate` *Compute subject-specific or overall transition probabilities*

Description

This function is a wrapper for `mstate::probtrans`. Its purpose is to preclude the computation of (co-)variances of the transition probability estimator when the fitted Cox model is empirical Bayes. This help page is an adaptation of the `mstate::probtrans` help page.

Usage

```
probtrans_mstate(object, ...)

## Default S3 method:
probtrans_mstate(
  object,
  predt,
  direction = c("forward", "fixedhorizon"),
  method = c("aalen", "greenwood"),
  variance = TRUE,
  covariance = FALSE,
  ...
)

## S3 method for class 'coxrfx'
probtrans_mstate(object, predt, direction = c("forward", "fixedhorizon"), ...)
```

Arguments

`object`	An msfit object containing estimated cumulative hazards for each of the transitions in the multi-state model and, if standard errors are requested, (co)variances of these cumulative hazards for each pair of transitions.
`...`	other arguments.
`predt`	A positive number indicating the prediction time. This is either the time at which the prediction is made (if direction= "forward") or the time for which the prediction is to be made (if direction="fixedhorizon").

`direction`	One of "forward" (default) or "fixedhorizon", indicating whether prediction is forward or for a fixed horizon.
`method`	A character string specifying the type of variances to be computed (so only needed if either variance or covariance is TRUE). Possible values are "aalen" or "greenwood".
`variance`	Logical value indicating whether standard errors are to be calculated (default is TRUE).
`covariance`	Logical value indicating whether covariances of transition probabilities for different states are to be calculated (default is FALSE).

Details

`probtrans_mstate` computes estimates of transition probabilities for an object generated by `msfit_generic`. It calls the method `probtrans_mstate.coxrfx`, if the msfit object was generated by msfit_generic.coxrfx, or the method probtrans_mstate.default otherwise. Both methods are identical to the function `mstate::probtrans`. The only reserve is that `probtrans_mstate.coxrfx` does not allow the computation of the (co-)variances of the transition probability estimator. In fact, this computation relies on asymptotic results for the *fixed* effects Cox model (see de Wreede et al, 2010, section 2.3.2), and `msfit_generic.coxrfx` produces estimates of cumulative hazards under a random effects/empirical Bayes Cox model.

`probtrans_mstate` should only be used for Markov models, as it relies on product limit calculations.

Value

An object of class `probtrans`. See the 'value' section in the the help page of `probtrans` for details.

Author(s)

Rui Costa, adapting the work of L. de Wreede, M. Fiocco and H. Putter in the `mstate` package.

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.

See Also

`probtrans`; `msfit`; `msfit_generic`.

`successful_transitions` *Find the unique possible path until an absorbing state*

Description

From a `unique_paths` object that shows all possible paths until absorption from an initial state, `successful_transitions` picks the path that finishes in `to_state`, if there is one. The initial state is the one defined in the argument `from_state` to the function `unique_paths`. The process must have a tree-like structure.

Usage

```
successful_transitions(unique_paths_object, to_state, tmat)
```

Arguments

`unique_paths_object`	An object created by running `unique_paths`.
`to_state`	An absorbing state.
`tmat`	Transition matrix.

Details

This function is used by `probtrans_by_convolution_clockforward` and `probtrans_by_convolution_clockreset`. It is not meant to be called by the user.

Value

A vector with the unique sequence of states between two states.

Author(s)

Rui Costa

See Also

`unique_paths`; `probtrans_by_convolution_clockforward`; `probtrans_by_convolution_clockreset`.

summary.coxrfx *A summary method for CoxRFX models*

Description

This function prints the point estimates of parameters and hyperparameters contained in a `coxrfx` object.

Usage

```
## S3 method for class 'coxrfx'
summary(object, ...)
```

Arguments

`object`	A `coxrfx` object (obtained by running the function `CoxRFX`).
`...`	Further arguments passed to or from other methods.

Details

Prints two data frames, one with hyperparameter estimates and another with regression coefficient estimates.

Value

Returns an invisible NULL object.

Author(s)

Rui Costa

unique_paths *Find all possible paths until absorption from a given starting state*

Description

`unique_paths` finds all possible sequences of states until absorption when the process has a tree-like structure.

Usage

```
unique_paths(from_state, tmat)
```

Arguments

`from_state`	Initial state.
`tmat`	A transition matrix describing the states and transitions in the multi-state model, as can be obtained by running `transMat`. See argument `trans` in `msprep` (`mstate` package) for more detailed information.

Details

This function is used by the function `probtrans_by_convolution`. It is not meant to be called by the user.

Value

A matrix where each column is a sequence of states taken by the process until absorption. There are as many columns as the number of possible paths until absorption.

Author(s)

Rui Costa

See Also

`probtrans_by_convolution`; `transMat`.

Index