Package 'flexsurv'

July 22, 2025

Type Package

Title Flexible Parametric Survival and Multi-State Models

Version 2.3.2

Date 2024-08-16

Description Flexible parametric models for time-to-event data,
including the Royston-Parmar spline model, generalized gamma and
generalized F distributions. Any user-defined parametric
distribution can be fitted, given at least an R function defining
the probability density or hazard. There are also tools for
fitting and predicting from fully parametric multi-state models,
based on either cause-specific hazards or mixture models.

License GPL (>= 2)

Depends survival, R (>= 2.15.0)

Imports assertthat, deSolve, generics, magrittr, mstate (>= 0.2.10),
Matrix, muhaz, mvtnorm, numDeriv, quadprog, Rcpp (>= 0.11.5),
rlang, rstpm2, purrr, statmod, tibble, tidyr, dplyr,
tidyselect, ggplot2

Encoding UTF-8

Suggests splines2, flexsurvcure, survminer, lubridate, rmarkdown,
colorspace, eha, knitr, msm, testthat, TH.data, broom, covr

URL https://github.com/chjackson/flexsurv,
http://chjackson.github.io/flexsurv/

BugReports https://github.com/chjackson/flexsurv/issues

VignetteBuilder knitr

LazyData yes

LinkingTo Rcpp

RoxygenNote 7.3.1

NeedsCompilation yes

Author Christopher Jackson [aut, cre],
      Paul Metcalfe [ctb],
      Jordan Amdahl [ctb],
      Matthew T. Warkentin [ctb],
      Michael Sweeting [ctb],
      Kevin Kunzmann [ctb]

Maintainer Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

Repository CRAN

Date/Publication 2024-08-17 05:50:02 UTC

Contents

---

flexsurv-package   *flexsurv: Flexible parametric survival and multi-state models*

Description

flexsurv: Flexible parametric models for time-to-event data, including the generalized gamma, the generalized F and the Royston-Parmar spline model, and extensible to user-defined distributions.

Details

flexsurvreg fits parametric models for time-to-event (survival) data. Data may be right-censored, and/or left-censored, and/or left-truncated. Several built-in parametric distributions are available. Any user-defined parametric model can also be employed by supplying a list with basic information about the distribution, including the density or hazard and ideally also the cumulative distribution or hazard.

Covariates can be included using a linear model on any parameter of the distribution, log-transformed to the real line if necessary. This typically defines an accelerated failure time or proportional hazards model, depending on the distribution and parameter.

flexsurvspline fits the flexible survival model of Royston and Parmar (2002) in which the log cumulative hazard is modelled as a natural cubic spline function of log time. Covariates can be included on any of the spline parameters, giving either a proportional hazards model or an arbitrarily-flexible time-dependent effect. Alternative proportional odds or probit parameterisations are available.

Output from the models can be presented as survivor, cumulative hazard and hazard functions (summary.flexsurvreg). These can be plotted against nonparametric estimates (plot.flexsurvreg) to assess goodness-of-fit. Any other user-defined function of the parameters may be summarised in the same way.

Multi-state models for time-to-event data can also be fitted with the same functions. Predictions from those models can then be made using the functions pmatrix.fs, pmatrix.simfs, totlos.fs, totlos.simfs, or sim.fmsm, or alternatively by msfit.flexsurvreg followed by mssample or probtrans from the package **mstate**.

Distribution ("dpqr") functions for the generalized gamma and F distributions are given in GenGamma, GenF (preferred parameterisations) and GenGamma.orig, GenF.orig (original parameterisations). flexsurv also includes the standard Gompertz distribution with unrestricted shape parameter, see Gompertz.

User guide

The **flexsurv user guide** vignette explains the methods in detail, and gives several worked examples. A further vignette **flexsurv-examples** gives a few more complicated examples, and users are encouraged to submit their own.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

References

Jackson, C. (2016). flexsurv: A Platform for Parametric Survival Modeling in R. Journal of Statistical Software, 70(8), 1-33. doi:10.18637/jss.v070.i08

Royston, P. and Parmar, M. (2002). Flexible parametric proportional-hazards and proportional-odds models for censored survival data, with application to prognostic modelling and estimation of treatment effects. Statistics in Medicine 21(1):2175-2197.

Cox, C. (2008). The generalized F distribution: An umbrella for parametric survival analysis. Statistics in Medicine 27:4301-4312.

Cox, C., Chu, H., Schneider, M. F. and Muñoz, A. (2007). Parametric survival analysis and taxonomy of hazard functions for the generalized gamma distribution. Statistics in Medicine 26:4252-4374

See Also

Useful links:

- <https://github.com/chjackson/flexsurv>
- <http://chjackson.github.io/flexsurv/>
- Report bugs at <https://github.com/chjackson/flexsurv/issues>

AIC.fmsm	*Akaike's information criterion from a flexible parametric multistate model*

Description

Defined as the sum of the AICs of the transition-specific models.

Usage

```
## S3 method for class 'fmsm'
AIC(object, ..., k = 2)
```

Arguments

object	Object returned by `fmsm` representing a multistate model.
...	Further arguments (currently unused).
k	Penalty applied to number of parameters (defaults to the standard 2).

Value

The sum of the AICs of the transition-specific models.

AICc   *Second-order Akaike information criterion*

Description

Generic function for the second-order Akaike information criterion. The only current implementation of this in **flexsurv** is in `AICc.flexsurvreg`, see there for more details.

Usage

```
AICc(object, ...)

AICC(object, ...)
```

Arguments

object	Fitted model object.
...	Other arguments (currently unused).

Details

This can be spelt either as `AICC` or `AICc`.

AICc.flexsurvreg   *Second-order Akaike information criterion*

Description

Second-order or "corrected" Akaike information criterion, often known as AICc (see, e.g. Burnham and Anderson 2002). This is defined as -2 log-likelihood + $(2*p*n)/(n - p -1)$, whereas the standard AIC is defined as -2 log-likelihood + $2*p$, where p is the number of parameters and n is the sample size. The correction is intended to adjust AIC for small-sample bias, hence it only makes a difference to the result for small n.

Usage

```
## S3 method for class 'flexsurvreg'
AICc(object, cens = TRUE, ...)

## S3 method for class 'flexsurvreg'
AICC(object, cens = TRUE, ...)
```

Arguments

object	Fitted model returned by [flexsurvreg](#) (or [flexsurvspline](#)).
cens	Include censored observations in the sample size term (n) used in this calculation. See [BIC.flexsurvreg](#) for a discussion of the issues with defining the sample size.
...	Other arguments (currently unused).

Details

This can be spelt either as AICC or AICc.

Value

The second-order AIC of the fitted model.

References

Burnham, K. P., Anderson, D. R. (2002) Model Selection and Multimodel Inference: a practical information-theoretic approach. Second edition. Springer: New York.

See Also

[BIC](#), [AIC](#), [BIC.flexsurvreg](#), [nobs.flexsurvreg](#)

---

ajfit	*Aalen-Johansen nonparametric estimates comparable to a fitted flexsurvmix model*

Description

Given a fitted flexsurvmix model, return the Aalen-Johansen estimates of the probability of occupying each state at a series of times covering the observed data. State 1 represents not having experienced any of the competing events, while state 2 and any further states correspond to having experienced each of the competing events respectively. These estimates can be compared with the fitted probabilities returned by [p_flexsurvmix](#) to check the fit of a flexsurvmix model.

Usage

```
ajfit(x, newdata = NULL, tidy = TRUE)
```

Arguments

x	Fitted model returned by [flexsurvmix](#).
newdata	Data frame of alternative covariate values to check fit for. Only factor covariates are supported.
tidy	If TRUE then a single tidy data frame is returned. Otherwise the function returns the object returned by survfit, or a list of these objects if we are computing subset-specific estimates.

Details

This is only supported for models with no covariates or models containing only factor covariates.

For models with factor covariates, the Aalen-Johansen estimates are computed for the subsets of the data defined in newdata. If newdata is not supplied, then this function returns state occupancy probabilities for all possible combinations of the factor levels.

The Aalen-Johansen estimates are computed using `survfit` from the `survival` package (Therneau 2020).

References

Therneau T (2020). _A Package for Survival Analysis in R_. R package version 3.2-3, <URL: https://CRAN.R-project.org/package=survival>.

| ajfit_flexsurvmix | *Forms a tidy data frame for plotting the fit of parametric mixture multi-state models against nonparametric estimates* |

Description

This computes Aalen-Johansen estimates of the probability of occupying each state at a series of times, using `ajfit`. The equivalent estimates from the parametric model are then produced using `p_flexsurvmix`, and concatenated with the nonparametric estimates to form a tidy data frame. This data frame can then simply be plotted using `ggplot`.

Usage

```
ajfit_flexsurvmix(x, maxt = NULL, startname = "Start", B = NULL)
```

Arguments

x	Fitted model returned by `flexsurvmix`.
maxt	Maximum time to produce parametric estimates for. By default this is the maximum event time in the data, the maximum time we have nonparametric estimates for.
startname	Label to give the state corresponding to "no event happened yet". By default this is `"Start"`.
B	Number of simulation replications to use to calculate a confidence interval for the parametric estimates in `p_flexsurvmix`. Comparable intervals for the Aalen-Johansen estimates are returned if this is set. Otherwise if B=NULL then no intervals are returned.

---

ajfit_fmsm | *Check the fit of Markov flexible parametric multi-state models against nonparametric estimates.*

Description

Computes both parametric and comparable Aalen-Johansen nonparametric estimates from a flexible parametric multi-state model, and returns them together in a tidy data frame. Only models with no covariates, or only factor covariates, are supported. If there are factor covariates, then the nonparametric estimates are computed for subgroups defined by combinations of the covariates. Another restriction of this function is that all transitions must have the same covariates on them.

Usage

```
ajfit_fmsm(x, maxt = NULL, newdata = NULL)
```

Arguments

x
: Object returned by [fmsm](#) representing a flexible parametric multi-state model. This must be Markov, rather than semi-Markov, and no check is performed for this. Note that all "competing risks" style models, with just one source state and multiple destination states, are Markov, so those are fine here.

maxt
: Maximum time to compute parametric estimates to.

newdata
: Data frame defining the subgroups to consider. This must have a column for each covariate in the model. If omitted, then all potential subgroups defined by combinations of factor covariates are included. Continuous covariates are not supported.

Value

Tidy data frame containing both Aalen-Johansen and parametric estimates of state occupancy over time, and by subgroup if subgroups are included.

---

augment.flexsurvreg | *Augment data with information from a flexsurv model object*

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Usage

```
## S3 method for class 'flexsurvreg'
augment(
  x,
  data = NULL,
  newdata = NULL,
  type.predict = "response",
  type.residuals = "response",
  ...
)
```

Arguments

x	Output from [flexsurvreg](#) or [flexsurvspline](#), representing a fitted survival model object.
data	A [data.frame](#) or [tibble](#) containing the original data that was used to produce the object x.
newdata	A [data.frame](#) or [tibble](#) containing all the original predictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.
type.predict	Character indicating type of prediction to use. Passed to the type argument of the [predict](#) generic. Allowed arguments vary with model class, so be sure to read the predict.my_class documentation.
type.residuals	Character indicating type of residuals to use. Passed to the type argument of [residuals](#) generic. Allowed arguments vary with model class, so be sure to read the residuals.my_class documentation.
...	Additional arguments. Not currently used.

Details

If neither of data or newdata are specified, then model.frame(x) will be used. It is worth noting that model.frame(x) will include a [Surv](#) object and not the original time-to-event variables used when fitting the flexsurvreg object. If the original data is desired, specify data.

Value

A [tibble](#) containing data or newdata and possible additional columns:

- .fitted Fitted values of model
- .se.fit Standard errors of fitted values
- .resid Residuals (not present if newdata specified)

Examples

```
fit <- flexsurvreg(formula = Surv(futime, fustat) ~ age, data = ovarian, dist = "exp")
augment(fit, data = ovarian)
```

| basis | *Natural cubic spline basis* |

Description

Compute a basis for a natural cubic spline, by default using the parameterisation described by Royston and Parmar (2002). Used for flexible parametric survival models.

Usage

```
basis(knots, x, spline = "rp")
```

Arguments

knots	Vector of knot locations in increasing order, including the boundary knots at the beginning and end.
x	Vector of ordinates to compute the basis for.
spline	`"rp"` to use the natural cubic spline basis described in Royston and Parmar. `"splines2ns"` to use the alternative natural cubic spline basis from the splines2 package (Wang and Yan 2021), which may be better behaved due to the basis being orthogonal.

Details

The exact formula for the basis is given in `flexsurvspline`.

Value

A matrix with one row for each ordinate and one column for each knot.

basis returns the basis, and dbasis returns its derivative with respect to x.

fss and dfss are the same, but with the order of the arguments swapped around for consistency with similar functions in other R packages.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

References

Royston, P. and Parmar, M. (2002). Flexible parametric proportional-hazards and proportional-odds models for censored survival data, with application to prognostic modelling and estimation of treatment effects. Statistics in Medicine 21(1):2175-2197.

Wang W, Yan J (2021). Shape-Restricted Regression Splines with R Package splines2. Journal of Data Science, 19(3), 498-517.

See Also

flexsurvspline.

---

bc   *Breast cancer survival data*

Description

Survival times of 686 patients with primary node positive breast cancer.

Usage

    bc

Format

A data frame with 686 rows.

censrec	(numeric)	1=dead, 0=censored
rectime	(numeric)	Time of death or censoring in days
group	(numeric)	Prognostic group: "Good","Medium" or "Poor",
		from a regression model developed by Sauerbrei and Royston (1999).

Source

German Breast Cancer Study Group, 1984-1989. Used as a reference dataset for the spline-based survival model of Royston and Parmar (2002), implemented here in flexsurvspline. Originally provided with the stpm (Royston 2001, 2004) and stpm2 (Lambert 2009, 2010) Stata modules.

References

Royston, P. and Parmar, M. (2002). Flexible parametric proportional-hazards and proportional-odds models for censored survival data, with application to prognostic modelling and estimation of treatment effects. Statistics in Medicine 21(1):2175-2197.

Sauerbrei, W. and Royston, P. (1999). Building multivariable prognostic and diagnostic models: transformation of the predictors using fractional polynomials. Journal of the Royal Statistical Society, Series A 162:71-94.

See Also

flexsurvspline

BIC.flexsurvreg	*Bayesian Information Criterion (BIC) for comparison of flexsurvreg models*

Description

Bayesian Information Criterion (BIC) for comparison of flexsurvreg models

Usage

```
## S3 method for class 'flexsurvreg'
BIC(object, cens = TRUE, ...)
```

Arguments

object	Fitted model returned by [flexsurvreg](or [flexsurvspline]).
cens	Include censored observations in the sample size term (n) used in the calculation of BIC.
...	Other arguments (currently unused).

Details

There is no "official" definition of what the sample size should be for the use of BIC in censored survival analysis. BIC is based on an approximation to Bayesian model comparison using Bayes factors and an implicit vague prior. Informally, the sample size describes the number of "units" giving rise to a distinct piece of information (Kass and Raftery 1995). However censored observations provide less information than observed events, in principle. The default used here is the number of individuals, for consistency with more familiar kinds of statistical modelling. However if censoring is heavy, then the number of events may be a better represent the amount of information. Following these principles, the best approximation would be expected to be somewere in between.

AIC and BIC are intended to measure different things. Briefly, AIC measures predictive ability, whereas BIC is expected to choose the true model from a set of models where one of them is the truth. Therefore BIC chooses simpler models for all but the tiniest sample sizes ($log(n) > 2$, $n > 7$). AIC might be preferred in the typical situation where "all models are wrong but some are useful". AIC also gives similar results to cross-validation (Stone 1977).

Value

The BIC of the fitted model. This is minus twice the log likelihood plus p*log(n), where p is the number of parameters and n is the sample size of the data. If weights was supplied to flexsurv, the sample size is defined as the sum of the weights.

References

Kass, R. E., & Raftery, A. E. (1995). Bayes factors. Journal of the American Statistical Association, 90(430), 773-795.

Stone, M. (1977). An asymptotic equivalence of choice of model by cross-validation and Akaike's criterion. Journal of the Royal Statistical Society: Series B (Methodological), 39(1), 44-47.

See Also

BIC, AIC, AICC.flexsurvreg, nobs.flexsurvreg

---

bootci.fmsm	*Bootstrap confidence intervals for flexsurv output functions*

Description

Calculate a confidence interval for a model output by repeatedly replacing the parameters in a fitted model object with a draw from the multivariate normal distribution of the maximum likelihood estimates, then recalculating the output function.

Usage

```
bootci.fmsm(
  x,
  B,
  fn,
  cl = 0.95,
  attrs = NULL,
  cores = NULL,
  sample = FALSE,
  ...
)
```

Arguments

x	Output from flexsurvreg or flexsurvspline, representing a fitted survival model object. Or a list of such objects, defining a multi-state model.
B	Number of parameter draws to use
fn	Function to bootstrap the results of. It must have an argument named x giving a fitted flexsurv model object. This may return a value with any format, e.g. list, matrix or vector, as long as it can be converted to a numeric vector with unlist. See the example below.
cl	Width of symmetric confidence interval, by default 0.95
attrs	Any attributes of the value returned from fn which we want confidence intervals for. These will be unlisted, if possible, and appended to the result vector.
cores	Number of cores to use for parallel processing.
sample	If TRUE then the bootstrap sample itself is returned. If FALSE then the quantiles of the sample are returned giving a confidence interval.
...	Additional arguments to pass to fn.

Value

A matrix with two rows, giving the upper and lower confidence limits respectively. Each row is a vector of the same length as the unlisted result of the function corresponding to fncall.

Examples

```
## How to use bootci.fmsm

## Write a function with one argument called x giving a fitted model,
## and returning some results of the model.  The results may be in any form.

tmat <- rbind(c(NA,1,2),c(NA,NA,3),c(NA,NA,NA))
bexp <- flexsurvreg(Surv(Tstart, Tstop, status) ~ trans, data=bosms3, dist="exp")

summfn <- function(x, t){
 resp <-  flexsurv::pmatrix.fs(x, trans=tmat, t=t)
 rest <- flexsurv::totlos.fs(x, trans=tmat, t=t)
 list(resp, rest)
}

## Use bootci.fmsm to obtain the confidence interval
## The matrix columns are in the order of the unlisted results of the original
## summfn.  You will have to rearrange them into the format that you want.
## If summfn has any extra arguments, in this case \code{t}, make sure they are
## passed through via the ... argument to bootci.fmsm

bootci.fmsm(bexp, B=3, fn=summfn, t=10)
bootci.fmsm(bexp, B=3, fn=summfn, t=5)
```

---

bos | *Bronchiolitis obliterans syndrome after lung transplants*

Description

A dataset containing histories of bronchiolitis obliterans syndrome (BOS) from lung transplant recipients. BOS is a chronic decline in lung function, often observed after lung transplantation.

Format

A data frame containing a sequence of observed or censored transitions to the next stage of severity or death. It is grouped by patient and includes histories of 204 patients. All patients start in state 1 (no BOS) at six months after transplant, and may subsequently develop BOS or die.

bosms3 contains the data for a three-state model: no BOS, BOS or death. bosms4 uses a four-state representation: no BOS, mild BOS, moderate/severe BOS or death.

id	(numeric)	Patient identification number
from	(numeric)	Observed starting state of the transition
to	(numeric)	Observed or potential ending state of the transition
Tstart	(numeric)	Time at the start of the interval
Tstop	(numeric)	Time at the end of the interval
time	(numeric)	Time difference Tstart-Tstop
status	(numeric)	1 if the transition to state to was observed, or 0 if the transition to state to was censored (for example,

trans   (factor)   Number of the transition from-to in the set of all ntrans allowed transitions, numbered from 1 to ntra

Details

The entry time of each patient into each stage of BOS was estimated by clinicians, based on their history of lung function measurements and acute rejection and infection episodes. BOS is only assumed to occur beyond six months after transplant. In the first six months the function of each patient's new lung stabilises. Subsequently BOS is diagnosed by comparing the lung function against the "baseline" value.

The same data are provided in the **msm** package, but in the native format of **msm** to allow Markov models to be fitted. In **flexsurv**, much more flexible models can be fitted.

Source

Papworth Hospital, U.K.

References

Heng. D. et al. (1998). Bronchiolitis Obliterans Syndrome: Incidence, Natural History, Prognosis, and Risk Factors. Journal of Heart and Lung Transplantation 17(12)1255–1263.

---

coef.flexsurvreg   *Extract model coefficients from fitted flexible survival models*

Description

Extract model coefficients from fitted flexible survival models. This presents all parameter estimates, transformed to the real line if necessary. For example, shape or scale parameters, which are constrained to be positive, are returned on the log scale.

Usage

```
## S3 method for class 'flexsurvreg'
coef(object, ...)
```

Arguments

object   Output from flexsurvreg or flexsurvspline, representing a fitted survival model object.

...   Further arguments passed to or from other methods. Currently unused.

Details

This matches the behaviour of coef.default for standard R model families such as [glm](glm), where intercepts in regression models are presented on the same scale as the covariate effects. Note that any parameter in a distribution fitted by flexsurvreg or flexsurvreg may be an intercept in a regression model.

Value

This returns the mod$res.t[,"est"] component of the fitted model object mod. See flexsurvreg, flexsurvspline for full documentation of all components.

Author(s)

C. H. Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

See Also

flexsurvreg, flexsurvspline.

---

coxsnell_flexsurvreg   *Cox-Snell residuals from a parametric survival model*

Description

Cox-Snell residuals from a parametric survival model

Usage

```
coxsnell_flexsurvreg(x)
```

Arguments

x   Object returned by flexsurvreg or flexsurvspline representing a fitted sur-
                  vival model

Value

A data frame with a column called est giving the Cox-Snell residual, defined as the fitted cumula-
tive hazard at each data point. fitted cumulative hazard at the given observed data point, and other
columns indicating the observation time, observed event status, and covariate values defining the
data at this point.

The cumulative hazards est should form a censored sample from an Exponential(1). Therefore to
check the fit of the model, plot a nonparametric estimate of the cumulative hazard curve against a
diagonal line through the origin, which is the theoretical cumulative hazard trajectory of the Expo-
nential(1).

Examples

```
 fitg <- flexsurvreg(formula = Surv(futime, fustat) ~ age, data = ovarian, dist = "gengamma")
 cs <- coxsnell_flexsurvreg(fitg)

 ## Model appears to fit well, with some small sample noise
 surv <- survfit(Surv(cs$est, ovarian$fustat) ~ 1)
 plot(surv, fun="cumhaz")
 abline(0, 1, col="red")
```

---

flexsurvmix   *Flexible parametric mixture models for times to competing events*

Description

In a mixture model for competing events, an individual can experience one of a set of different events. We specify a model for the probability that they will experience each event before the others, and a model for the time to the event conditionally on that event occurring first.

Usage

```
flexsurvmix(
  formula,
  data,
  event,
  dists,
  pformula = NULL,
  anc = NULL,
  partial_events = NULL,
  initp = NULL,
  inits = NULL,
  fixedpars = NULL,
  dfns = NULL,
  method = "direct",
  em.control = NULL,
  optim.control = NULL,
  aux = NULL,
  sr.control = survreg.control(),
  integ.opts,
  hess.control = NULL,
  ...
)
```

Arguments

formula   Survival model formula. The left hand side is a Surv object specified as in
                   [flexsurvreg](). This may define various kinds of censoring, as described in [Surv]().
                   Any covariates on the right hand side of this formula will be placed on the lo-
                   cation parameter for every component-specific distribution. Covariates on other
                   parameters of the component-specific distributions may be supplied using the
                   anc argument.

                   Alternatively, formula may be a list of formulae, with one component for each
                   alternative event. This may be used to specify different covariates on the location
                   parameter for different components.

                   A list of formulae may also be used to indicate that for particular individuals,
                   different events may be observed in different ways, with different censoring

mechanisms. Each list component specifies the data and censoring scheme for that mixture component.

For example, suppose we are studying people admitted to hospital,and the competing states are death in hospital and discharge from hospital. At time t we know that a particular individual is still alive, but we do not know whether they are still in hospital, or have been discharged. In this case, if the individual were to die in hospital, their death time would be right censored at t. If the individual will be (or has been) discharged before death, their discharge time is completely unknown, thus interval-censored on (0,Inf). Therefore, we need to store different event time and status variables in the data for different alternative events. This is specified here as

formula = list("discharge" = Surv(t1di, t2di, type="interval2"), "death" = Surv(t1de, status_de))

where for this individual, (t1di, t2di) = (0, Inf) and (t1de, status_de) = (t, 0).

The "dot" notation commonly used to indicate "all remaining variables" in a formula is not supported in flexsurvmix.

data	Data frame containing variables mentioned in formula, event and anc.
event	Variable in the data that specifies which of the alternative events is observed for which individual. If the individual's follow-up is right-censored, or if the event is otherwise unknown, this variable must have the value NA.
	Ideally this should be a factor, since the mixture components can then be easily identified in the results with a name instead of a number. If this is not already a factor, it is coerced to one. Then the levels of the factor define the required order for the components of the list arguments dists, anc, inits and dfns. Alternatively, if the components of the list arguments are named according to the levels of event, then the components can be arranged in any order.
dists	Vector specifying the parametric distribution to use for each component. The same distributions are supported as in [flexsurvreg](#).
pformula	Formula describing covariates to include on the component membership proabilities by multinomial logistic regression. The first component is treated as the baseline.
	The "dot" notation commonly used to indicate "all remaining variables" in a formula is not supported.
anc	List of component-specific lists, of length equal to the number of components. Each component-specific list is a list of formulae representing covariate effects on parameters of the distribution.
	If there are covariates for one component but not others, then a list containing one null formula on the location parameter should be supplied for the component with no covariates, e.g list(rate=~1) if the location parameter is called rate.
	Covariates on the location parameter may also be supplied here instead of in formula. Supplying them in anc allows some components but not others to have covariates on their location parameter. If a covariate on the location parameter was provided in formula, and there are covariates on other parameters, then a null formula should be included for the location parameter in anc, e.g list(rate=~1)

partial_events   List specifying the factor levels of event which indicate knowledge that an individual will not experience particular events, but may experience others. The names of the list indicate codes that indicate partial knowledge for some individuals. The list component is a vector, which must be a subset of levels(event) defining the events that a person with the corresponding event code may experience.

    For example, suppose there are three alternative events called "disease1","disease2" and "disease3", and for some individuals we know that they will not experience "disease2", but they may experience the other two events. In that case we must create a new factor level, called, for example "disease1or3", and set the value of event to be "disease1or3" for those individuals. Then we use the "partial_events" argument to tell flexsurvmix what the potential events are for individuals with this new factor level.

    partial_events = list("disease1or3" = c("disease1","disease3"))

initp   Initial values for component membership probabilities. By default, these are assumed to be equal for each component.

inits   List of component-specific vectors. Each component-specific vector contains the initial values for the parameters of the component-specific model, as would be supplied as the inits argument of flexsurvreg. By default, a heuristic is used to obtain initial values, which depends on the parametric distribution being used, but is usually based on the empirical mean and/or variance of the survival times.

fixedpars   Indexes of parameters to fix at their initial values and not optimise. Arranged in the order: baseline mixing probabilities, covariates on mixing probabilities, time-to-event parameters by mixing component. Within mixing components, time-to-event parameters are ordered in the same way as in flexsurvreg.

    If fixedpars=TRUE then all parameters will be fixed and the function simply calculates the log-likelihood at the initial values.

    Not currently supported when using the EM algorithm.

dfns   List of lists of user-defined distribution functions, one for each mixture component. Each list component is specified as the dfns argument of flexsurvreg.

method   Method for maximising the likelihood. Either "em" for the EM algorithm, or "direct" for direct maximisation.

em.control   List of settings to control EM algorithm fitting. The only options currently available are

    trace set to 1 to print the parameter estimates at each iteration of the EM algorithm

    reltol convergence criterion. The algorithm stops if the log likelihood changes by a relative amount less than reltol. The default is the same as in optim, that is, sqrt(.Machine$double.eps).

    var.method method to compute the covariance matrix. "louis" for the method of Louis (1982), or "direct"for direct numerical calculation of the Hessian of the log likelihood.

    optim.p.control A list that is passed as the control argument to optim in the M step for the component membership probability parameters. The optimisation in the M step for the time-to-event parameters can be controlled by the optim.control argument to flexsurvmix.

	For example, em.control = list(trace=1, reltol=1e-12).
optim.control	List of options to pass as the control argument to optim, which is used by method="direct" or in the M step for the time-to-event parameters in method="em". By default, this uses fnscale=10000 and ndeps=rep(1e-06,p) where p is the number of parameters being estimated, unless the user specifies these options explicitly.
aux	A named list of other arguments to pass to custom distribution functions. This is used, for example, by flexsurvspline to supply the knot locations and modelling scale (e.g. hazard or odds). This cannot be used to fix parameters of a distribution — use fixedpars for that.
sr.control	For the models which use survreg to find the maximum likelihood estimates (Weibull, exponential, log-normal), this list is passed as the control argument to survreg.
integ.opts	List of named arguments to pass to integrate, if a custom density or hazard is provided without its cumulative version. For example, integ.opts = list(rel.tol=1e-12)
hess.control	List of options to control covariance matrix computation. Available options are: numeric. If TRUE then numerical methods are used to compute the Hessian for models where an analytic Hessian is available. These models include the Weibull (both versions), exponential, Gompertz and spline models with hazard or odds scale. The default is to use the analytic Hessian for these models. For all other models, numerical methods are always used to compute the Hessian, whether or not this option is set. tol.solve. The tolerance used for solve when inverting the Hessian (default .Machine$double.eps) tol.evalues The accepted tolerance for negative eigenvalues in the covariance matrix (default 1e-05). The Hessian is positive definite, thus invertible, at the maximum likelihood. If the Hessian computed after optimisation convergence can't be inverted, this is either because the converged result is not the maximum likelihood (e.g. it could be a "saddle point"), or because the numerical methods used to obtain the Hessian were inaccurate. If you suspect that the Hessian was computed wrongly enough that it is not invertible, but not wrongly enough that the nearest valid inverse would be an inaccurate estimate of the covariance matrix, then these tolerance values can be modified (reducing tol.solve or increasing tol.evalues) to allow the inverse to be computed.
...	Optional arguments to the general-purpose optimisation routine optim. For example, the BFGS optimisation algorithm is the default in flexsurvreg, but this can be changed, for example to method="Nelder-Mead" which can be more robust to poor initial values. If the optimisation fails to converge, consider normalising the problem using, for example, control=list(fnscale = 2500), for example, replacing 2500 by a number of the order of magnitude of the likelihood. If 'false' convergence is reported with a non-positive-definite Hessian, then consider tightening the tolerance criteria for convergence. If the optimisation takes a long time, intermediate steps can be printed using the trace argument of the control list. See optim for details.

Details

This differs from the more usual "competing risks" models, where we specify "cause-specific hazards" describing the time to each competing event. This time will not be observed for an individual if one of the competing events happens first. The event that happens first is defined by the minimum of the times to the alternative events.

The flexsurvmix function fits a mixture model to data consisting of a single time to an event for each individual, and an indicator for what type of event occurs for that individual. The time to event may be observed or censored, just as in flexsurvreg, and the type of event may be known or unknown. In a typical application, where we follow up a set of individuals until they experience an event or a maximum follow-up time is reached, the event type is known if the time is observed, and the event type is unknown when follow-up ends and the time is right-censored.

The model is fitted by maximum likelihood, either directly or by using an expectation-maximisation (EM) algorithm, by wrapping flexsurvreg to compute the likelihood or to implement the E and M steps.

Some worked examples are given in the package vignette about multi-state modelling, which can be viewed by running vignette("multistate", package="flexsurv").

Value

List of objects containing information about the fitted model. The important one is res, a data frame containing the parameter estimates and associated information.

References

Jackson, C. H. and Tom, B. D. M. and Kirwan, P. D. and Mandal, S. and Seaman, S. R. and Kunzmann, K. and Presanis, A. M. and De Angelis, D. (2022) A comparison of two frameworks for multi-state modelling, applied to outcomes after hospital admissions with COVID-19. Statistical Methods in Medical Research 31(9) 1656-1674.

Larson, M. G., & Dinse, G. E. (1985). A mixture model for the regression analysis of competing risks data. Journal of the Royal Statistical Society: Series C (Applied Statistics), 34(3), 201-211.

Lau, B., Cole, S. R., & Gange, S. J. (2009). Competing risk regression models for epidemiologic data. American Journal of Epidemiology, 170(2), 244-256.

---

flexsurvreg   *Flexible parametric regression for time-to-event data*

Description

Parametric modelling or regression for time-to-event data. Several built-in distributions are available, and users may supply their own.

Usage

```
flexsurvreg(
  formula,
  anc = NULL,
  data,
  weights,
  bhazard,
  rtrunc,
  subset,
  na.action,
  dist,
  inits,
  fixedpars = NULL,
  dfns = NULL,
  aux = NULL,
  cl = 0.95,
  integ.opts = NULL,
  sr.control = survreg.control(),
  hessian = TRUE,
  hess.control = NULL,
  ...
)
```

Arguments

formula　　　　　A formula expression in conventional R linear modelling syntax. The response must be a survival object as returned by the [Surv](#) function, and any covariates are given on the right-hand side. For example,

`Surv(time, dead) ~ age + sex`

Surv objects of type="right","counting", "interval1" or "interval2" are supported, corresponding to right-censored, left-truncated or interval-censored observations.

If there are no covariates, specify 1 on the right hand side, for example `Surv(time, dead) ~ 1`.

If the right hand side is specified as . all remaining variables are included as covariates. For example, `Surv(time, dead) ~ .` corresponds to `Surv(time, dead) ~ age + sex` if data contains the variables time, dead, age, and sex.

By default, covariates are placed on the "location" parameter of the distribution, typically the "scale" or "rate" parameter, through a linear model, or a log-linear model if this parameter must be positive. This gives an accelerated failure time model or a proportional hazards model (see dist below) depending on how the distribution is parameterised.

Covariates can be placed on other ("ancillary") parameters by using the name of the parameter as a "function" in the formula. For example, in a Weibull model, the following expresses the scale parameter in terms of age and a treatment variable treat, and the shape parameter in terms of sex and treatment.

`Surv(time, dead) ~ age + treat + shape(sex) + shape(treat)`

However, if the names of the ancillary parameters clash with any real functions that might be used in formulae (such as I(), or factor()), then those functions will not work in the formula. A safer way to model covariates on ancillary parameters is through the anc argument to flexsurvreg.

survreg users should also note that the function strata() is ignored, so that any covariates surrounded by strata() are applied to the location parameter. Likewise the function frailty() is not handled.

anc   An alternative and safer way to model covariates on ancillary parameters, that is, parameters other than the main location parameter of the distribution. This is a named list of formulae, with the name of each component giving the parameter to be modelled. The model above can also be defined as:

Surv(time, dead) ~ age + treat, anc = list(shape = ~ sex + treat)

data   A data frame in which to find variables supplied in formula. If not given, the variables should be in the working environment.

weights   Optional numeric variable giving weights for each individual in the data. The fitted model is then defined by maximising the weighted sum of the individual-specific log-likelihoods.

bhazard   Optional variable giving expected hazards for relative survival models. The model is described by Nelson et al. (2007).

bhazard should contain a vector of values for each person in the data.

- For people with observed events, bhazard refers to the hazard at the observed event time.
- For people whose event time is left-censored or interval-censored, bhazard should contain the probability of dying by the end of the corresponding interval, conditionally on being alive at the start.
- For people whose event time is right-censored, the value of bhazard is ignored and does not need to be specified.

If bhazard is supplied, then the parameter estimates returned by flexsurvreg and the outputs returned by summary.flexsurvreg describe the parametric model for relative survival.

For relative survival models, the log-likelihood returned by flexsurvreg is a partial log-likelihood, which omits a constant term defined by the sum of the cumulative hazards at the event or censoring time for each individual. Hence this constant must be added if a full likelihood is needed.

rtrunc   Optional variable giving individual-specific right-truncation times. Used for analysing data with "retrospective ascertainment". For example, suppose we want to estimate the distribution of the time from onset of a disease to death, but have only observed cases known to have died by the current date. In this case, times from onset to death for individuals in the data are right-truncated by the current date minus the onset date. Predicted survival times for new cases can then be described by an un-truncated version of the fitted distribution.

These models can suffer from weakly identifiable parameters and badly-behaved likelihood functions, and it is advised to compare convergence for different initial values by supplying different inits arguments to flexsurvreg.

subset   Vector of integers or logicals specifying the subset of the observations to be used in the fit.

na.action   a missing-data filter function, applied after any 'subset' argument has been used. Default is options()$na.action.

dist   Typically, one of the strings in the first column of the following table, identifying a built-in distribution. This table also identifies the location parameters, and whether covariates on these parameters represent a proportional hazards (PH) or accelerated failure time (AFT) model. In an accelerated failure time model, the covariate speeds up or slows down the passage of time. So if the coefficient (presented on the log scale) is log(2), then doubling the covariate value would give half the expected survival time.

"gengamma"	Generalized gamma (stable)	mu	AFT
"gengamma.orig"	Generalized gamma (original)	scale	AFT
"genf"	Generalized F (stable)	mu	AFT
"genf.orig"	Generalized F (original)	mu	AFT
"weibull"	Weibull	scale	AFT
"gamma"	Gamma	rate	AFT
"exp"	Exponential	rate	PH
"llogis"	Log-logistic	scale	AFT
"lnorm"	Log-normal	meanlog	AFT
"gompertz"	Gompertz	rate	PH

"exponential" and "lognormal" can be used as aliases for "exp" and "lnorm", for compatibility with survreg.

Alternatively, dist can be a list specifying a custom distribution. See section "Custom distributions" below for how to construct this list.

Very flexible spline-based distributions can also be fitted with flexsurvspline.

The parameterisations of the built-in distributions used here are the same as in their built-in distribution functions: dgengamma, dgengamma.orig, dgenf, dgenf.orig, dweibull, dgamma, dexp, dlnorm, dgompertz, respectively. The functions in base R are used where available, otherwise, they are provided in this package.

A package vignette "Distributions reference" lists the survivor functions and covariate effect parameterisations used by each built-in distribution.

For the Weibull, exponential and log-normal distributions, flexsurvreg simply works by calling survreg to obtain the maximum likelihood estimates, then calling optim to double-check convergence and obtain the covariance matrix for flexsurvreg's preferred parameterisation.

The Weibull parameterisation is different from that in survreg, instead it is consistent with dweibull. The "scale" reported by survreg is equivalent to 1/shape as defined by dweibull and hence flexsurvreg. The first coefficient (Intercept) reported by survreg is equivalent to log(scale) in dweibull and flexsurvreg.

Similarly in the exponential distribution, the rate, rather than the mean, is modelled on covariates.

The object flexsurv.dists lists the names of the built-in distributions, their parameters, location parameter, functions used to transform the parameter ranges

to and from the real line, and the functions used to generate initial values of each parameter for estimation.

inits   An optional numeric vector giving initial values for each unknown parameter. These are numbered in the order: baseline parameters (in the order they appear in the distribution function, e.g. shape before scale in the Weibull), covariate effects on the location parameter, covariate effects on the remaining parameters. This is the same order as the printed estimates in the fitted model.

                If not specified, default initial values are chosen from a simple summary of the survival or censoring times, for example the mean is often used to initialize scale parameters. See the object flexsurv.dists for the exact methods used. If the likelihood surface may be uneven, it is advised to run the optimisation starting from various different initial values to ensure convergence to the true global maximum.

fixedpars   Vector of indices of parameters whose values will be fixed at their initial values during the optimisation. The indices are ordered as in inits. For example, in a stable generalized Gamma model with two covariates, to fix the third of three generalized gamma parameters (the shape Q, see the help for [GenGamma](#)) and the second covariate, specify fixedpars = c(3, 5)

dfns   An alternative way to define a custom survival distribution (see section "Custom distributions" below). A list whose components may include ″d″, ″p″, ″h″, or ″H″ containing the probability density, cumulative distribution, hazard, or cumulative hazard functions of the distribution. For example,

                list(d=dllogis, p=pllogis).

                If dfns is used, a custom dlist must still be provided, but dllogis and pllogis need not be visible from the global environment. This is useful if flexsurvreg is called within other functions or environments where the distribution functions are also defined dynamically.

aux   A named list of other arguments to pass to custom distribution functions. This is used, for example, by [flexsurvspline](#) to supply the knot locations and modelling scale (e.g. hazard or odds). This cannot be used to fix parameters of a distribution — use fixedpars for that.

cl   Width of symmetric confidence intervals for maximum likelihood estimates, by default 0.95.

integ.opts   List of named arguments to pass to [integrate](#), if a custom density or hazard is provided without its cumulative version. For example,

                integ.opts = list(rel.tol=1e-12)

sr.control   For the models which use [survreg](#) to find the maximum likelihood estimates (Weibull, exponential, log-normal), this list is passed as the control argument to [survreg](#).

hessian   Calculate the covariances and confidence intervals for the parameters. Defaults to TRUE.

hess.control   List of options to control covariance matrix computation. Available options are:

                numeric. If TRUE then numerical methods are used to compute the Hessian for models where an analytic Hessian is available. These models include the Weibull (both versions), exponential, Gompertz and spline models with hazard

or odds scale. The default is to use the analytic Hessian for these models. For all other models, numerical methods are always used to compute the Hessian, whether or not this option is set.

tol.solve. The tolerance used for [solve](#) when inverting the Hessian (default .Machine$double.eps)

tol.evalues The accepted tolerance for negative eigenvalues in the covariance matrix (default 1e-05).

The Hessian is positive definite, thus invertible, at the maximum likelihood. If the Hessian computed after optimisation convergence can't be inverted, this is either because the converged result is not the maximum likelihood (e.g. it could be a "saddle point"), or because the numerical methods used to obtain the Hessian were inaccurate. If you suspect that the Hessian was computed wrongly enough that it is not invertible, but not wrongly enough that the nearest valid inverse would be an inaccurate estimate of the covariance matrix, then these tolerance values can be modified (reducing tol.solve or increasing tol.evalues) to allow the inverse to be computed.

...   Optional arguments to the general-purpose optimisation routine [optim](#). For example, the BFGS optimisation algorithm is the default in [flexsurvreg](#), but this can be changed, for example to method="Nelder-Mead" which can be more robust to poor initial values. If the optimisation fails to converge, consider normalising the problem using, for example, control=list(fnscale = 2500), for example, replacing 2500 by a number of the order of magnitude of the likelihood. If 'false' convergence is reported with a non-positive-definite Hessian, then consider tightening the tolerance criteria for convergence. If the optimisation takes a long time, intermediate steps can be printed using the trace argument of the control list. See [optim](#) for details.

Details

Parameters are estimated by maximum likelihood using the algorithms available in the standard R [optim](#) function. Parameters defined to be positive are estimated on the log scale. Confidence intervals are estimated from the Hessian at the maximum, and transformed back to the original scale of the parameters.

The usage of [flexsurvreg](#) is intended to be similar to [survreg](#) in the **survival** package.

Value

A list of class "flexsurvreg" containing information about the fitted model. Components of interest to users may include:

call	A copy of the function call, for use in post-processing.
dlist	List defining the survival distribution used.
res	Matrix of maximum likelihood estimates and confidence limits, with parameters on their natural scales.
res.t	Matrix of maximum likelihood estimates and confidence limits, with parameters all transformed to the real line (using a log transform for all built-in models where this is necessary). The [coef](#), [vcov](#) and [confint](#) methods for flexsurvreg objects work on this scale.

coefficients   The transformed maximum likelihood estimates, as in res.t. Calling coef()
               on a [flexsurvreg](#) object simply returns this component.

loglik   Log-likelihood. This will differ from Stata, where the sum of the log uncen-
               sored survival times is added to the log-likelihood in survival models, to remove
               dependency on the time scale.

               For relative survival models specified with bhazard, this is a partial log-likelihood
               which omits a constant term defined by the sum of the cumulative hazards over
               all event or censoring times.

logliki   Vector of individual contributions to the log-likelihood

AIC   Akaike's information criterion (-2*log likelihood + 2*number of estimated pa-
               rameters)

cov   Covariance matrix of the parameters, on the real-line scale (e.g. log scale),
               which can be extracted with [vcov](#).

data   Data used in the model fit. To extract this in the standard R formats, use use
               [model.frame.flexsurvreg](#) or [model.matrix.flexsurvreg](#).

Custom distributions

[flexsurvreg](#) is intended to be easy to extend to handle new distributions. To define a new distri-
bution for use in [flexsurvreg](#), construct a list with the following elements:

"name" A string naming the distribution. If this is called "dist", for example, then there must be
       visible in the working environment, at least, either

       a) a function called ddist which defines the probability density,

       or

       b) a function called hdist which defines the hazard.

       Ideally, in case a) there should also be a function called pdist which defines the probability
       distribution or cumulative density, and in case b) there should be a function called Hdist defin-
       ing the cumulative hazard. If these additional functions are not provided, **flexsurv** attempts to
       automatically create them by numerically integrating the density or hazard function. However,
       model fitting will be much slower, or may not even work at all, if the analytic versions of these
       functions are not available.

       The functions must accept vector arguments (representing different times, or alternative val-
       ues for each parameter) and return the results as a vector. The function [Vectorize](#) may be
       helpful for doing this: see the example below. These functions may be in an add-on pack-
       age (see below for an example) or may be user-written. If they are user-written they must
       be defined in the global environment, or supplied explicitly through the dfns argument to
       flexsurvreg. The latter may be useful if the functions are created dynamically (as in the
       source of flexsurvspline) and thus not visible through R's scoping rules.

       Arguments other than parameters must be named in the conventional way – for example x for
       the first argument of the density function or hazard, as in [dnorm](#)(x, ...) and q for the first
       argument of the probability function. Density functions should also have an argument log,
       after the parameters, which when TRUE, computes the log density, using a numerically stable
       additive formula if possible.

       Additional functions with names beginning with "DLd" and "DLS" may be defined to calculate
       the derivatives of the log density and log survival probability, with respect to the parameters of

the distribution. The parameters are expressed on the real line, for example after log transformation if they are defined as positive. The first argument must be named `t`, representing the time, and the remaining arguments must be named as the parameters of the density function. The function must return a matrix with rows corresponding to times, and columns corresponding to the parameters of the distribution. The derivatives are used, if available, to speed up the model fitting with `optim`.

`"pars"` Vector of strings naming the parameters of the distribution. These must be the same names as the arguments of the density and probability functions.

`"location"` Name of the main parameter governing the mean of the distribution. This is the default parameter on which covariates are placed in the `formula` supplied to `flexsurvreg`.

`"transforms"` List of R functions which transform the range of values taken by each parameter onto the real line. For example, `c(log, log)` for a distribution with two positive parameters.

`"inv.transforms"` List of R functions defining the corresponding inverse transformations. Note these must be lists, even for single parameter distributions they should be supplied as, e.g. `c(exp)` or `list(exp)`.

`"inits"` A function of the observed survival times `t` (including right-censoring times, and using the halfway point for interval-censored times) which returns a vector of reasonable initial values for maximum likelihood estimation of each parameter. For example, `function(t){ c(1, mean(t)) }` will always initialize the first of two parameters at 1, and the second (a scale parameter, for instance) at the mean of `t`.

For example, suppose we want to use an extreme value survival distribution. This is available in the CRAN package **eha**, which provides conventionally-defined density and probability functions called `eha::dEV` and `eha::pEV`. See the Examples below for the custom list in this case, and the subsequent command to fit the model.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

References

Jackson, C. (2016). flexsurv: A Platform for Parametric Survival Modeling in R. Journal of Statistical Software, 70(8), 1-33. doi:10.18637/jss.v070.i08

Cox, C. (2008) The generalized *F* distribution: An umbrella for parametric survival analysis. Statistics in Medicine 27:4301-4312.

Cox, C., Chu, H., Schneider, M. F. and Muñoz, A. (2007) Parametric survival analysis and taxonomy of hazard functions for the generalized gamma distribution. Statistics in Medicine 26:4252-4374

Jackson, C. H. and Sharples, L. D. and Thompson, S. G. (2010) Survival models in health economic evaluations: balancing fit and parsimony to improve prediction. International Journal of Biostatistics 6(1):Article 34.

Nelson, C. P., Lambert, P. C., Squire, I. B., & Jones, D. R. (2007). Flexible parametric models for relative survival, with application in coronary heart disease. Statistics in medicine, 26(30), 5486-5498.

See Also

flexsurvspline for flexible survival modelling using the spline model of Royston and Parmar.

plot.flexsurvreg and lines.flexsurvreg to plot fitted survival, hazards and cumulative hazards from models fitted by flexsurvreg and flexsurvspline.

Examples

```
## Compare generalized gamma fit with Weibull
fitg <- flexsurvreg(formula = Surv(futime, fustat) ~ 1, data = ovarian, dist="gengamma")
fitg
fitw <- flexsurvreg(formula = Surv(futime, fustat) ~ 1, data = ovarian, dist="weibull")
fitw
plot(fitg)
lines(fitw, col="blue", lwd.ci=1, lty.ci=1)
## Identical AIC, probably not enough data in this simple example for a
## very flexible model to be worthwhile.

## Custom distribution
## make "dEV" and "pEV" from eha package (if installed)
##    available to the working environment
if (require("eha")) {
custom.ev <- list(name="EV",
                     pars=c("shape","scale"),
                     location="scale",
                     transforms=c(log, log),
                     inv.transforms=c(exp, exp),
                     inits=function(t){ c(1, median(t)) })
fitev <- flexsurvreg(formula = Surv(futime, fustat) ~ 1, data = ovarian,
                     dist=custom.ev)
fitev
lines(fitev, col="purple", col.ci="purple")
}


## Custom distribution: supply the hazard function only
hexp2 <- function(x, rate=1){ rate } # exponential distribution
hexp2 <- Vectorize(hexp2)
custom.exp2 <- list(name="exp2", pars=c("rate"), location="rate",
                     transforms=c(log), inv.transforms=c(exp),
                     inits=function(t)1/mean(t))
flexsurvreg(Surv(futime, fustat) ~ 1, data = ovarian, dist=custom.exp2)
flexsurvreg(Surv(futime, fustat) ~ 1, data = ovarian, dist="exp")
## should give same answer
```

---

flexsurvrtrunc   *Flexible parametric models for right-truncated, uncensored data defined by times of initial and final events.*

Description

This function estimates the distribution of the time between an initial and final event, in situations where individuals are only observed if they have experienced both events before a certain time, thus they are right-truncated at this time. The time of the initial event provides information about the time from initial to final event, given the truncated observation scheme, and initial events are assumed to occur with an exponential growth rate.

Usage

```
flexsurvrtrunc(
  t,
  tinit,
  rtrunc,
  tmax,
  data = NULL,
  method = "joint",
  dist,
  theta = NULL,
  fixed.theta = TRUE,
  inits = NULL,
  fixedpars = NULL,
  dfns = NULL,
  integ.opts = NULL,
  cl = 0.95,
  optim_control = list()
)
```

Arguments

t	Vector of time differences between an initial and final event for a set of individuals.
tinit	Absolute time of the initial event for each individual.
rtrunc	Individual-specific right truncation points on the same scale as t, so that each individual's survival time t would not have been observed if it was greater than the corresponding element of rtrunc. Only used in method="joint". In method="final", the right-truncation is implicit.
tmax	Maximum possible time between initial and final events that could have been observed. This is only used in method="joint", and is ignored in method="final".
data	Data frame containing t, rtrunc and tinit.
method	If "joint" then the "joint-conditional" method is used. If "final" then the "conditional-on-final" method is used. The "conditional-on-initial" method can be implemented by using [flexsurvreg](#) with a rtrunc argument. These methods are all described in Seaman et al. (2020).
dist	Typically, one of the strings in the first column of the following table, identifying a built-in distribution. This table also identifies the location parameters, and whether covariates on these parameters represent a proportional hazards (PH) or accelerated failure time (AFT) model. In an accelerated failure time model,

the covariate speeds up or slows down the passage of time. So if the coefficient (presented on the log scale) is log(2), then doubling the covariate value would give half the expected survival time.

`"gengamma"`	Generalized gamma (stable)	mu	AFT
`"gengamma.orig"`	Generalized gamma (original)	scale	AFT
`"genf"`	Generalized F (stable)	mu	AFT
`"genf.orig"`	Generalized F (original)	mu	AFT
`"weibull"`	Weibull	scale	AFT
`"gamma"`	Gamma	rate	AFT
`"exp"`	Exponential	rate	PH
`"llogis"`	Log-logistic	scale	AFT
`"lnorm"`	Log-normal	meanlog	AFT
`"gompertz"`	Gompertz	rate	PH

`"exponential"` and `"lognormal"` can be used as aliases for `"exp"` and `"lnorm"`, for compatibility with survreg.

Alternatively, `dist` can be a list specifying a custom distribution. See section "Custom distributions" below for how to construct this list.

Very flexible spline-based distributions can also be fitted with flexsurvspline.

The parameterisations of the built-in distributions used here are the same as in their built-in distribution functions: dgengamma, dgengamma.orig, dgenf, dgenf.orig, dweibull, dgamma, dexp, dlnorm, dgompertz, respectively. The functions in base R are used where available, otherwise, they are provided in this package.

A package vignette "Distributions reference" lists the survivor functions and covariate effect parameterisations used by each built-in distribution.

For the Weibull, exponential and log-normal distributions, flexsurvreg simply works by calling survreg to obtain the maximum likelihood estimates, then calling optim to double-check convergence and obtain the covariance matrix for flexsurvreg's preferred parameterisation.

The Weibull parameterisation is different from that in survreg, instead it is consistent with dweibull. The `"scale"` reported by survreg is equivalent to 1/shape as defined by dweibull and hence flexsurvreg. The first coefficient (Intercept) reported by survreg is equivalent to `log(scale)` in dweibull and flexsurvreg.

Similarly in the exponential distribution, the rate, rather than the mean, is modelled on covariates.

The object flexsurv.dists lists the names of the built-in distributions, their parameters, location parameter, functions used to transform the parameter ranges to and from the real line, and the functions used to generate initial values of each parameter for estimation.

theta	Initial value (or fixed value) for the exponential growth rate theta. Defaults to 1.
fixed.theta	Should theta be fixed at its initial value or estimated. This only applies to method=`"joint"`. For method=`"final"`, theta must be fixed.

inits	Initial values for the parameters of the parametric survival distributon. If not supplied, a heuristic is used. as is done in `flexsurvreg`.
fixedpars	Integer indices of the parameters of the survival distribution that should be fixed to their values supplied in `inits`. Same length as `inits`.
dfns	An alternative way to define a custom survival distribution (see section "Custom distributions" below). A list whose components may include "d", "p", "h", or "H" containing the probability density, cumulative distribution, hazard, or cumulative hazard functions of the distribution. For example,
	`list(d=dllogis, p=pllogis)`.
	If `dfns` is used, a custom `dlist` must still be provided, but `dllogis` and `pllogis` need not be visible from the global environment. This is useful if `flexsurvreg` is called within other functions or environments where the distribution functions are also defined dynamically.
integ.opts	List of named arguments to pass to `integrate`, if a custom density or hazard is provided without its cumulative version. For example,
	`integ.opts = list(rel.tol=1e-12)`
cl	Width of symmetric confidence intervals for maximum likelihood estimates, by default 0.95.
optim_control	List to supply as the `control` argument to `optim` to control the likelihood maximisation.

Details

Covariates are not currently supported.

Note that `flexsurvreg`, with an `rtrunc` argument, can fit models for a similar kind of data, but those models ignore the information provided by the time of the initial event.

A nonparametric estimator of survival under right-truncation is also provided in `survrtrunc`. See Seaman et al. (2020) for a full comparison of the alternative models.

References

Seaman, S., Presanis, A. and Jackson, C. (2020) Estimating a Time-to-Event Distribution from Right-Truncated Data in an Epidemic: a Review of Methods

See Also

`flexsurvreg`, `survrtrunc`.

Examples

```
set.seed(1)
## simulate time to initial event
X <- rexp(1000, 0.2)
## simulate time between initial and final event
T <- rgamma(1000, 2, 10)

## right-truncate to keep only those with final event time
```

```
## before tmax
tmax <- 40
obs <- X + T < tmax
rtrunc <- tmax - X
dat <- data.frame(X, T, rtrunc)[obs,]

flexsurvrtrunc(t=T, rtrunc=rtrunc, tinit=X, tmax=40, data=dat,
                dist="gamma", theta=0.2)

flexsurvrtrunc(t=T, rtrunc=rtrunc, tinit=X, tmax=40, data=dat,
                dist="gamma", theta=0.2, fixed.theta=FALSE)

flexsurvrtrunc(t=T, rtrunc=rtrunc, tinit=X, tmax=40, data=dat,
                dist="gamma", theta=0.2, inits=c(1, 8))

flexsurvrtrunc(t=T, rtrunc=rtrunc, tinit=X, tmax=40, data=dat,
                dist="gamma", theta=0.2, method="final")

flexsurvrtrunc(t=T, rtrunc=rtrunc, tinit=X, tmax=40, data=dat,
                dist="gamma", fixed.theta=TRUE)

flexsurvrtrunc(t=T, rtrunc=rtrunc, tinit=X, tmax=40, data=dat,
                dist="weibull", fixed.theta=TRUE)

flexsurvrtrunc(t=T, rtrunc=rtrunc, tinit=X, tmax=40, data=dat,
                dist="lnorm", fixed.theta=TRUE)

flexsurvrtrunc(t=T, rtrunc=rtrunc, tinit=X, tmax=40, data=dat,
                dist="gengamma", fixed.theta=TRUE)

flexsurvrtrunc(t=T, rtrunc=rtrunc, tinit=X, tmax=40, data=dat,
                dist="gompertz", fixed.theta=TRUE)
```

---

flexsurvspline   *Flexible survival regression using the Royston/Parmar spline model.*

Description

Flexible parametric modelling of time-to-event data using the spline model of Royston and Parmar (2002).

Usage

```
flexsurvspline(
  formula,
  data,
  weights,
  bhazard,
  rtrunc,
```

```
        subset,
        k = 0,
        knots = NULL,
        bknots = NULL,
        scale = "hazard",
        timescale = "log",
        spline = "rp",
        ...
    )
```

Arguments

formula   A formula expression in conventional R linear modelling syntax. The response
            must be a survival object as returned by the [Surv](#) function, and any covariates
            are given on the right-hand side. For example,

            Surv(time, dead) ~ age + sex

            specifies a model where the log cumulative hazard (by default, see scale) is a
            linear function of the covariates age and sex.

            If there are no covariates, specify 1 on the right hand side, for example Surv(time,
            dead) ~ 1.

            Time-varying covariate effects can be specified using the method described in
            [flexsurvreg](#) for placing covariates on ancillary parameters. The ancillary pa-
            rameters here are named gamma1, ..., gammar where r is the number of knots k
            plus one (the "degrees of freedom" as defined by Royston and Parmar). So for
            the default Weibull model, there is just one ancillary parameter gamma1.

            Therefore a model with one internal spline knot, where the equivalents of the
            Weibull shape and scale parameters, but not the higher-order term gamma2, vary
            with age and sex, can be specified as:

            Surv(time, dead) ~ age + sex + gamma1(age) + gamma1(sex)

            or alternatively (and more safely, see flexsurvreg)

            Surv(time, dead) ~ age + sex, anc=list(gamma1=~age + sex)

            Surv objects of type="right","counting", "interval1" or "interval2" are
            supported, corresponding to right-censored, left-truncated or interval-censored
            observations.

data   A data frame in which to find variables supplied in formula. If not given, the
            variables should be in the working environment.

weights   Optional variable giving case weights.

bhazard   Optional variable giving expected hazards for relative survival models.

rtrunc   Optional variable giving individual right-truncation times (see [flexsurvreg](#)).
            Note that these models can suffer from weakly identifiable parameters and badly-
            behaved likelihood functions, and it is advised to compare convergence for dif-
            ferent initial values by supplying different inits arguments to flexsurvspline.

subset   Vector of integers or logicals specifying the subset of the observations to be used
            in the fit.

k   Number of knots in the spline. The default k=0 gives a Weibull, log-logistic or
            lognormal model, if "scale" is "hazard", "odds" or "normal" respectively. k

is equivalent to df-1 in the notation of stpm for Stata. The knots are then chosen as equally-spaced quantiles of the log uncensored survival times, for example, at the median with one knot, or at the 33% and 67% quantiles of log time (or time, see "timescale") with two knots. To override this default knot placement, specify knots instead.

knots   Locations of knots on the axis of log time (or time, see "timescale"). If not specified, knot locations are chosen as described in k above. Either k or knots must be specified. If both are specified, knots overrides k.

bknots   Locations of boundary knots, on the axis of log time (or time, see "timescale"). If not supplied, these are are chosen as the minimum and maximum log death time.

scale   If "hazard", the log cumulative hazard is modelled as a spline function.

If "odds", the log cumulative odds is modelled as a spline function.

If "normal", $-\Phi^{-1}(S(t))$ is modelled as a spline function, where $\Phi^{-1}()$ is the inverse normal distribution function qnorm.

timescale   If "log" (the default) the log cumulative hazard (or alternative) is modelled as a spline function of log time. If "identity", it is modelled as a spline function of time, however this model would not satisfy the desirable property that the cumulative hazard (or alternative) should approach 0 at time zero.

spline   "rp" to use the natural cubic spline basis described in Royston and Parmar.

"splines2ns" to use the alternative natural cubic spline basis from the splines2 package (Wang and Yan 2021), which may be better behaved due to the basis being orthogonal.

...   Any other arguments to be passed to or through flexsurvreg, for example, anc, inits, fixedpars, weights, subset, na.action, and any options to control optimisation. See flexsurvreg.

Details

This function works as a wrapper around flexsurvreg by dynamically constructing a custom distribution using dsurvspline, psurvspline and unroll.function.

In the spline-based survival model of Royston and Parmar (2002), a transformation $g(S(t, z))$ of the survival function is modelled as a natural cubic spline function of log time $x = \log(t)$ plus linear effects of covariates z.

$$g(S(t, z)) = s(x, \boldsymbol{\gamma}) + \boldsymbol{\beta}^T \mathbf{z}$$

The proportional hazards model (scale="hazard") defines $g(S(t, \mathbf{z})) = \log(-\log(S(t, \mathbf{z}))) = \log(H(t, \mathbf{z}))$, the log cumulative hazard.

The proportional odds model (scale="odds") defines $g(S(t, \mathbf{z})) = \log(S(t, \mathbf{z})^{-1} - 1)$, the log cumulative odds.

The probit model (scale="normal") defines $g(S(t, \mathbf{z})) = -\Phi^{-1}(S(t, \mathbf{z}))$, where $\Phi^{-1}()$ is the inverse normal distribution function qnorm.

With no knots, the spline reduces to a linear function, and these models are equivalent to Weibull, log-logistic and lognormal models respectively.

The spline coefficients $\gamma_j : j = 1, 2 \ldots$, which are called the "ancillary parameters" above, may also be modelled as linear functions of covariates $\mathbf{z}$, as

$$\gamma_j(\mathbf{z}) = \gamma_{j0} + \gamma_{j1} z_1 + \gamma_{j2} z_2 + \ldots$$

giving a model where the effects of covariates are arbitrarily flexible functions of time: a non-proportional hazards or odds model.

Natural cubic splines are cubic splines constrained to be linear beyond boundary knots k_{min}, k_{max}. The spline function is defined as

$$s(x, \boldsymbol{\gamma}) = \gamma_0 + \gamma_1 x + \gamma_2 v_1(x) + \ldots +$$
$$\gamma_{m+1} v_m(x)$$

where $v_j(x)$ is the jth basis function

$$v_j(x) = (x - k_j)_+^3 - \lambda_j (x - k_{min})_+^3 - (1 -$$
$$\lambda_j)(x - k_{max})_+^3$$

$$\lambda_j = \frac{k_{max} - k_j}{k_{max} - k_{min}}$$

and $(x - a)_+ = max(0, x - a)$.

Value

A list of class "flexsurvreg" with the same elements as described in flexsurvreg, and including extra components describing the spline model. See in particular:

k	Number of knots.
knots	Location of knots on the log time axis.
scale	The scale of the model, hazard, odds or normal.
res	Matrix of maximum likelihood estimates and confidence limits. Spline coefficients are labelled "gamma...", and covariate effects are labelled with the names of the covariates.
	Coefficients gamma1,gamma2,... here are the equivalent of s0,s1,... in Stata streg, and gamma0 is the equivalent of the xb constant term. To reproduce results, use the noorthog option in Stata, since no orthogonalisation is performed on the spline basis here.
	In the Weibull model, for example, gamma0,gamma1 are -shape*log(scale), shape respectively in dweibull or flexsurvreg notation, or (-Intercept/scale, 1/scale) in survreg notation.
	In the log-logistic model with shape a and scale b (as in eha::dllogis from the **eha** package), 1/b^a is equivalent to exp(gamma0), and a is equivalent to gamma1.
	In the log-normal model with log-scale mean mu and standard deviation sigma, -mu/sigma is equivalent to gamma0 and 1/sigma is equivalent to gamma1.

loglik	The maximised log-likelihood. This will differ from Stata, where the sum of the log uncensored survival times is added to the log-likelihood in survival models, to remove dependency on the time scale.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

References

Royston, P. and Parmar, M. (2002). Flexible parametric proportional-hazards and proportional-odds models for censored survival data, with application to prognostic modelling and estimation of treatment effects. Statistics in Medicine 21(1):2175-2197.

Wang W, Yan J (2021). Shape-Restricted Regression Splines with R Package splines2. Journal of Data Science, 19(3), 498-517.

Jackson, C. (2016). flexsurv: A Platform for Parametric Survival Modeling in R. Journal of Statistical Software, 70(8), 1-33. doi:10.18637/jss.v070.i08

See Also

[flexsurvreg](#) for flexible survival modelling using general parametric distributions.

[plot.flexsurvreg](#) and [lines.flexsurvreg](#) to plot fitted survival, hazards and cumulative hazards from models fitted by [flexsurvspline](#) and [flexsurvreg](#).

Examples

```
## Best-fitting model to breast cancer data from Royston and Parmar (2002)
## One internal knot (2 df) and cumulative odds scale

spl <- flexsurvspline(Surv(recyrs, censrec) ~ group, data=bc, k=1, scale="odds")

## Fitted survival

plot(spl, lwd=3, ci=FALSE)

## Simple Weibull model fits much less well

splw <- flexsurvspline(Surv(recyrs, censrec) ~ group, data=bc, k=0, scale="hazard")
lines(splw, col="blue", ci=FALSE)

## Alternative way of fitting the Weibull

## Not run:
splw2 <- flexsurvreg(Surv(recyrs, censrec) ~ group, data=bc, dist="weibull")

## End(Not run)
```

---

fmixmsm   *Constructor for a mixture multi-state model based on flexsurvmix*

Description

Constructor for a mixture multi-state model based on flexsurvmix

Usage

```
fmixmsm(...)
```

Arguments

`...`	Named arguments. Each argument should be a fitted model as returned by [flexsurvmix](). The name of each argument names the starting state for that model.

Value

A list of [flexsurvmix]() objects, with the following attribute(s):

pathways A list of all potential pathways until absorption, for models without cycles. For models with cycles this will have an element has_cycle=TRUE, plus the pathways discovered before the function found the cycle and gave up.

---

fmsm   *Construct a multi-state model from a set of parametric survival models*

Description

Construct a multi-state model from a set of parametric survival models

Usage

```
fmsm(..., trans)
```

Arguments

`...`	Objects returned by [flexsurvreg]() or [flexsurvspline]() representing fitted survival models.
`trans`	A matrix of integers specifying which models correspond to which transitions. The r, s entry is i if the ith argument specified in `...` is the model for the state r to state s transition. The entry should be NA if the transition is disallowed.

Value

A list containing the objects given in . . . , and with attributes "trans" and "statenames" defining the transition structure matrix and state names, and with list components named to describe the transitions they correspond to. If any of the arguments in . . . are named, then these are used to define the transition names, otherwise default names are chosen based on the state names.

GenF	*Generalized F distribution*

Description

Density, distribution function, hazards, quantile function and random generation for the generalized F distribution, using the reparameterisation by Prentice (1975).

Usage

```
dgenf(x, mu = 0, sigma = 1, Q, P, log = FALSE)

pgenf(q, mu = 0, sigma = 1, Q, P, lower.tail = TRUE, log.p = FALSE)

Hgenf(x, mu = 0, sigma = 1, Q, P)

hgenf(x, mu = 0, sigma = 1, Q, P)

qgenf(p, mu = 0, sigma = 1, Q, P, lower.tail = TRUE, log.p = FALSE)

rgenf(n, mu = 0, sigma = 1, Q, P)
```

Arguments

x, q	Vector of quantiles.
mu	Vector of location parameters.
sigma	Vector of scale parameters.
Q	Vector of first shape parameters.
P	Vector of second shape parameters.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are $P(X \leq x)$, otherwise, $P(X > x)$.
p	Vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required.

Details

If $y \sim F(2s_1, 2s_2)$, and $w = \log(y)$ then $x = \exp(w\sigma + \mu)$ has the original generalized F distribution with location parameter μ, scale parameter $\sigma > 0$ and shape parameters s_1, s_2.

In this more stable version described by Prentice (1975), s_1, s_2 are replaced by shape parameters Q, P, with $P > 0$, and

$$s_1 = 2(Q^2 + 2P + Q\delta)^{-1}, \quad s_2 = 2(Q^2 + 2P - Q\delta)^{-1}$$

equivalently

$$Q = \left(\frac{1}{s_1} - \frac{1}{s_2}\right)\left(\frac{1}{s_1} + \frac{1}{s_2}\right)^{-1/2}, \quad P = \frac{2}{s_1 + s_2}$$

Define $\delta = (Q^2 + 2P)^{1/2}$, and $w = (\log(x) - \mu)\delta/\sigma$, then the probability density function of x is

$$f(x) = \frac{\delta(s_1/s_2)^{s_1} e^{s_1 w}}{\sigma x (1 + s_1 e^w/s_2)^{(s_1+s_2)} B(s_1, s_2)}$$

The original parameterisation is available in this package as `dgenf.orig`, for the sake of completion / compatibility. With the above definitions,

`dgenf(x, mu=mu, sigma=sigma, Q=Q, P=P) = dgenf.orig(x, mu=mu, sigma=sigma/delta, s1=s1, s2=s2)`

The generalized F distribution with `P=0` is equivalent to the generalized gamma distribution `dgengamma`, so that `dgenf(x, mu, sigma, Q, P=0)` equals `dgengamma(x, mu, sigma, Q)`. The generalized gamma reduces further to several common distributions, as described in the `GenGamma` help page.

The generalized F distribution includes the log-logistic distribution (see `eha::dllogis`) as a further special case:

`dgenf(x, mu=mu, sigma=sigma, Q=0, P=1) = eha::dllogis(x, shape=sqrt(2)/sigma, scale=exp(mu))`

The range of hazard trajectories available under this distribution are discussed in detail by Cox (2008). Jackson et al. (2010) give an application to modelling oral cancer survival for use in a health economic evaluation of screening.

Value

dgenf gives the density, pgenf gives the distribution function, qgenf gives the quantile function, rgenf generates random deviates, Hgenf retuns the cumulative hazard and hgenf the hazard.

Note

The parameters `Q` and `P` are usually called q and p in the literature - they were made upper-case in these R functions to avoid clashing with the conventional arguments q in the probability function and p in the quantile function.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

References

R. L. Prentice (1975). Discrimination among some parametric models. Biometrika 62(3):607-614.

Cox, C. (2008). The generalized F distribution: An umbrella for parametric survival analysis. Statistics in Medicine 27:4301-4312.

Jackson, C. H. and Sharples, L. D. and Thompson, S. G. (2010). Survival models in health economic evaluations: balancing fit and parsimony to improve prediction. International Journal of Biostatistics 6(1):Article 34.

See Also

GenF.orig, GenGamma

GenF.orig *Generalized F distribution (original parameterisation)*

Description

Density, distribution function, quantile function and random generation for the generalized F distribution, using the less flexible original parameterisation described by Prentice (1975).

Usage

```
dgenf.orig(x, mu = 0, sigma = 1, s1, s2, log = FALSE)

pgenf.orig(q, mu = 0, sigma = 1, s1, s2, lower.tail = TRUE, log.p = FALSE)

Hgenf.orig(x, mu = 0, sigma = 1, s1, s2)

hgenf.orig(x, mu = 0, sigma = 1, s1, s2)

qgenf.orig(p, mu = 0, sigma = 1, s1, s2, lower.tail = TRUE, log.p = FALSE)

rgenf.orig(n, mu = 0, sigma = 1, s1, s2)
```

Arguments

x, q	vector of quantiles.
mu	Vector of location parameters.
sigma	Vector of scale parameters.
s1	Vector of first F shape parameters.
s2	vector of second F shape parameters.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are $P(X \leq x)$, otherwise, $P(X > x)$.
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required.

Details

If $y \sim F(2s_1, 2s_2)$, and $w = \log(y)$ then $x = \exp(w\sigma + \mu)$ has the original generalized F distribution with location parameter μ, scale parameter $\sigma > 0$ and shape parameters $s_1 > 0, s_2 > 0$. The probability density function of x is

$$f(x|\mu, \sigma, s_1, s_2) = \frac{(s_1/s_2)^{s_1} e^{s_1 w}}{\sigma x (1 + s_1 e^w / s_2)^{(s_1+s_2)} B(s_1, s_2)}$$

where $w = (\log(x) - \mu)/\sigma$, and $B(s_1, s_2) = \Gamma(s_1)\Gamma(s_2)/\Gamma(s_1 + s_2)$ is the beta function.

As $s_2 \to \infty$, the distribution of x tends towards an original generalized gamma distribution with the following parameters:

[dgengamma.orig](x, shape=1/sigma, scale=exp(mu) / s1^sigma, k=s1)

See `GenGamma.orig` for how this includes several other common distributions as special cases.

The alternative parameterisation of the generalized F distribution, originating from Prentice (1975) and given in this package as `GenF`, is preferred for statistical modelling, since it is more stable as s_1 tends to infinity, and includes a further new class of distributions with negative first shape parameter. The original is provided here for the sake of completion and compatibility.

Value

dgenf.orig gives the density, pgenf.orig gives the distribution function, qgenf.orig gives the quantile function, rgenf.orig generates random deviates, Hgenf.orig retuns the cumulative hazard and hgenf.orig the hazard.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

References

R. L. Prentice (1975). Discrimination among some parametric models. Biometrika 62(3):607-614.

See Also

`GenF`, `GenGamma.orig`, `GenGamma`

| GenGamma | *Generalized gamma distribution* |

Description

Density, distribution function, hazards, quantile function and random generation for the generalized gamma distribution, using the parameterisation originating from Prentice (1974). Also known as the (generalized) log-gamma distribution.

Usage

```
dgengamma(x, mu = 0, sigma = 1, Q, log = FALSE)

pgengamma(q, mu = 0, sigma = 1, Q, lower.tail = TRUE, log.p = FALSE)

Hgengamma(x, mu = 0, sigma = 1, Q)

hgengamma(x, mu = 0, sigma = 1, Q)

qgengamma(p, mu = 0, sigma = 1, Q, lower.tail = TRUE, log.p = FALSE)

rgengamma(n, mu = 0, sigma = 1, Q)
```

Arguments

x, q	vector of quantiles.
mu	Vector of "location" parameters.
sigma	Vector of "scale" parameters. Note the inconsistent meanings of the term "scale" - this parameter is analogous to the (log-scale) standard deviation of the log-normal distribution, "sdlog" in [dlnorm], rather than the "scale" parameter of the gamma distribution [dgamma]. Constrained to be positive.
Q	Vector of shape parameters.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are $P(X \leq x)$, otherwise, $P(X > x)$.
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required.

Details

If $\gamma \sim Gamma(Q^{-2}, 1)$, and $w = log(Q^2\gamma)/Q$, then $x = \exp(\mu + \sigma w)$ follows the generalized gamma distribution with probability density function

$$f(x|\mu, \sigma, Q) = \frac{|Q|(Q^{-2})^{Q^{-2}}}{\sigma x \Gamma(Q^{-2})} \exp(Q^{-2}(Qw - \exp(Qw)))$$

This parameterisation is preferred to the original parameterisation of the generalized gamma by Stacy (1962) since it is more numerically stable near to $Q = 0$ (the log-normal distribution), and allows $Q <= 0$. The original is available in this package as [dgengamma.orig], for the sake of completion and compatibility with other software - this is implicitly restricted to Q>0 (or k>0 in the original notation). The parameters of [dgengamma] and [dgengamma.orig] are related as follows.

dgengamma.orig(x, shape=shape, scale=scale, k=k) =

dgengamma(x, mu=log(scale) + log(k)/shape, sigma=1/(shape*sqrt(k)),Q=1/sqrt(k))

The generalized gamma distribution simplifies to the gamma, log-normal and Weibull distributions with the following parameterisations:

```
dgengamma(x, mu, sigma, Q=0)       =   dlnorm(x, mu, sigma)
dgengamma(x, mu, sigma, Q=1)       =   dweibull(x, shape=1/sigma, scale=exp(mu))
dgengamma(x, mu, sigma, Q=sigma)   =   dgamma(x, shape=1/sigma^2, rate=exp(-mu) / sigma^2)
```

The properties of the generalized gamma and its applications to survival analysis are discussed in detail by Cox (2007).

The generalized F distribution GenF extends the generalized gamma to four parameters.

Value

dgengamma gives the density, pgengamma gives the distribution function, qgengamma gives the quantile function, rgengamma generates random deviates, Hgengamma retuns the cumulative hazard and hgengamma the hazard.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

References

Prentice, R. L. (1974). A log gamma model and its maximum likelihood estimation. Biometrika 61(3):539-544.

Farewell, V. T. and Prentice, R. L. (1977). A study of distributional shape in life testing. Technometrics 19(1):69-75.

Lawless, J. F. (1980). Inference in the generalized gamma and log gamma distributions. Technometrics 22(3):409-419.

Cox, C., Chu, H., Schneider, M. F. and Muñoz, A. (2007). Parametric survival analysis and taxonomy of hazard functions for the generalized gamma distribution. Statistics in Medicine 26:4252-4374

Stacy, E. W. (1962). A generalization of the gamma distribution. Annals of Mathematical Statistics 33:1187-92

See Also

GenGamma.orig, GenF, Lognormal, GammaDist, Weibull.

GenGamma.orig   *Generalized gamma distribution (original parameterisation)*

Description

Density, distribution function, hazards, quantile function and random generation for the generalized gamma distribution, using the original parameterisation from Stacy (1962).

Usage

```
dgengamma.orig(x, shape, scale = 1, k, log = FALSE)

pgengamma.orig(q, shape, scale = 1, k, lower.tail = TRUE, log.p = FALSE)

Hgengamma.orig(x, shape, scale = 1, k)

hgengamma.orig(x, shape, scale = 1, k)

qgengamma.orig(p, shape, scale = 1, k, lower.tail = TRUE, log.p = FALSE)

rgengamma.orig(n, shape, scale = 1, k)
```

Arguments

x, q	vector of quantiles.
shape	vector of "Weibull" shape parameters.
scale	vector of scale parameters.
k	vector of "Gamma" shape parameters.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are $P(X \leq x)$, otherwise, $P(X > x)$.
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required.

Details

If $w \sim Gamma(k, 1)$, then $x = \exp(w/shape + \log(scale))$ follows the original generalised gamma distribution with the parameterisation given here (Stacy 1962). Defining shape= $b > 0$, scale= $a > 0$, x has probability density

$$f(x|a, b, k) = \frac{b}{\Gamma(k)} \frac{x^{bk-1}}{a^{bk}}$$

$$\exp(-(x/a)^b)$$

The original generalized gamma distribution simplifies to the gamma, exponential and Weibull distributions with the following parameterisations:

```
dgengamma.orig(x, shape, scale, k=1)      =   dweibull(x, shape, scale)
dgengamma.orig(x, shape=1, scale, k)      =   dgamma(x, shape=k, scale)
dgengamma.orig(x, shape=1, scale, k=1)    =   dexp(x, rate=1/scale)
```

Also as k tends to infinity, it tends to the log normal (as in dlnorm) with the following parameters (Lawless, 1980):

```
dlnorm(x, meanlog=log(scale) + log(k)/shape,sdlog=1/(shape*sqrt(k)))
```

For more stable behaviour as the distribution tends to the log-normal, an alternative parameterisation was developed by Prentice (1974). This is given in `dgengamma`, and is now preferred for statistical modelling. It is also more flexible, including a further new class of distributions with negative shape k.

The generalized F distribution `GenF.orig`, and its similar alternative parameterisation `GenF`, extend the generalized gamma to four parameters.

Value

`dgengamma.orig` gives the density, `pgengamma.orig` gives the distribution function, `qgengamma.orig` gives the quantile function, `rgengamma.orig` generates random deviates, `Hgengamma.orig` retuns the cumulative hazard and `hgengamma.orig` the hazard.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

References

Stacy, E. W. (1962). A generalization of the gamma distribution. Annals of Mathematical Statistics 33:1187-92.

Prentice, R. L. (1974). A log gamma model and its maximum likelihood estimation. Biometrika 61(3):539-544.

Lawless, J. F. (1980). Inference in the generalized gamma and log gamma distributions. Technometrics 22(3):409-419.

See Also

`GenGamma`, `GenF.orig`, `GenF`, `Lognormal`, `GammaDist`, `Weibull`.

---

get_basepars	*Evaluate baseline time-to-event distribution parameters given covariate values in a flexsurvmix model*

Description

Evaluate baseline time-to-event distribution parameters given covariate values in a flexsurvmix model

Usage

```
get_basepars(x, newdata, event)
```

Arguments

x	Fitted model object
newdata	Data frame of alternative covariate values
event	Event

---

glance.flexsurvreg   _Glance at a flexsurv model object_

Description

Glance accepts a model object and returns a tibble with exactly one row of model summaries.

Usage

```
## S3 method for class 'flexsurvreg'
glance(x, ...)
```

Arguments

x	Output from [flexsurvreg](#) or [flexsurvspline](#), representing a fitted survival model object.
...	Not currently used.

Value

A one-row [tibble](#) containing columns:

- N Number of observations used in fitting
- events Number of events
- censored Number of censored events
- trisk Total length of time-at-risk (i.e. follow-up)
- df Degrees of freedom (i.e. number of estimated parameters)
- logLik Log-likelihood
- AIC Akaike's "An Information Criteria"
- BIC Bayesian Information Criteria

Examples

```
fitg <- flexsurvreg(formula = Surv(futime, fustat) ~ age, data = ovarian, dist = "gengamma")
glance(fitg)
```

Gompertz   *The Gompertz distribution*

Description

Density, distribution function, hazards, quantile function and random generation for the Gompertz distribution with unrestricted shape.

Usage

```
dgompertz(x, shape, rate = 1, log = FALSE)

pgompertz(q, shape, rate = 1, lower.tail = TRUE, log.p = FALSE)

qgompertz(p, shape, rate = 1, lower.tail = TRUE, log.p = FALSE)

rgompertz(n, shape = 1, rate = 1)

hgompertz(x, shape, rate = 1, log = FALSE)

Hgompertz(x, shape, rate = 1, log = FALSE)
```

Arguments

x, q	vector of quantiles.
shape, rate	vector of shape and rate parameters.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are $P(X \leq x)$, otherwise, $P(X > x)$.
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required.

Details

The Gompertz distribution with shape parameter a and rate parameter b has probability density function

$$f(x|a,b) = be^{ax} \exp(-b/a(e^{ax} - 1))$$

and hazard

$$h(x|a,b) = be^{ax}$$

The hazard is increasing for shape $a > 0$ and decreasing for $a < 0$. For $a = 0$ the Gompertz is equivalent to the exponential distribution with constant hazard and rate b.

The probability distribution function is

$$F(x|a,b) = 1 - \exp(-b/a(e^{ax} - 1))$$

Thus if a is negative, letting x tend to infinity shows that there is a non-zero probability $\exp(b/a)$ of living forever. On these occasions qgompertz and rgompertz will return Inf.

Value

dgompertz gives the density, pgompertz gives the distribution function, qgompertz gives the quantile function, hgompertz gives the hazard function, Hgompertz gives the cumulative hazard function, and rgompertz generates random deviates.

Note

Some implementations of the Gompertz restrict a to be strictly positive, which ensures that the probability of survival decreases to zero as x increases to infinity. The more flexible implementation given here is consistent with streg in Stata.

The functions eha::dgompertz and similar available in the package **eha** label the parameters the other way round, so that what is called the shape there is called the rate here, and what is called 1 / scale there is called the shape here. The terminology here is consistent with the exponential dexp and Weibull dweibull distributions in R.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

References

Stata Press (2007) Stata release 10 manual: Survival analysis and epidemiological tables.

See Also

dexp

---

hexp   *Hazard and cumulative hazard functions*

Description

Hazard and cumulative hazard functions for distributions which are built into flexsurv, and whose distribution functions are in base R.

Usage

```
hexp(x, rate = 1, log = FALSE)

Hexp(x, rate = 1, log = FALSE)

hgamma(x, shape, rate = 1, log = FALSE)

Hgamma(x, shape, rate = 1, log = FALSE)

hlnorm(x, meanlog = 0, sdlog = 1, log = FALSE)

Hlnorm(x, meanlog = 0, sdlog = 1, log = FALSE)

hweibull(x, shape, scale = 1, log = FALSE)

Hweibull(x, shape, scale = 1, log = FALSE)
```

Arguments

x	Vector of quantiles
rate	Rate parameter (exponential and gamma)
log	Compute log hazard or log cumulative hazard
shape	Shape parameter (Weibull and gamma)
meanlog	Mean on the log scale (log normal)
sdlog	Standard deviation on the log scale (log normal)
scale	Scale parameter (Weibull)

Details

For the exponential and the Weibull these are available analytically, and so are programmed here in numerically stable and efficient forms.

For the gamma and log-normal, these are simply computed as minus the log of the survivor function (cumulative hazard) or the ratio of the density and survivor function (hazard), so are not expected to be robust to extreme values or quick to compute.

Value

Hazard (functions beginning 'h') or cumulative hazard (functions beginning 'H').

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

See Also

[dexp](),[dweibull](),[dgamma](),[dlnorm](),[dgompertz](),[dgengamma](),[dgenf]()

---

hr_flexsurvreg   *Hazard ratio as a function of time from a parametric survival model*

Description

Hazard ratio as a function of time from a parametric survival model

Usage

```
hr_flexsurvreg(
  x,
  newdata = NULL,
  t = NULL,
  start = 0,
  ci = TRUE,
  B = 1000,
  cl = 0.95,
  na.action = na.pass
)
```

Arguments

x	Object returned by [flexsurvreg](#) or [flexsurvspline](#).
newdata	A data frame with two rows, each specifying a set of covariate values. The hazard ratio is calculated as hazard(z2)/hazard(z1), where z1 is the first row of newdata and z2 is the second row.
	newdata must be supplied unless the model x includes just one covariate. With one covariate, a default is constructed, which defines the hazard ratio between the second and first level of the factor (if the covariate is a factor), or between a value of 1 and a value of 0 (if the covariate is numeric).
t	Times to calculate fitted values for. By default, these are the sorted unique observation (including censoring) times in the data - for left-truncated datasets these are the "stop" times.
start	Optional left-truncation time or times. The returned survival, hazard or cumulative hazard will be conditioned on survival up to this time. Predicted times returned with "rmst", "mean", "median" or "quantile" will be times since time zero, not times since the start time.
	A vector of the same length as t can be supplied to allow different truncation times for each prediction time, though this doesn't make sense in the usual case where this function is used to calculate a predicted trajectory for a single individual. This is why the default start time was changed for version 0.4 of **flexsurv** - this was previously a vector of the start times observed in the data.
ci	Set to FALSE to omit confidence intervals.

B	Number of simulations from the normal asymptotic distribution of the estimates used to calculate confidence intervals or standard errors. Decrease for greater speed at the expense of accuracy, or set B=0 to turn off calculation of CIs and SEs.
cl	Width of symmetric confidence intervals, relative to 1.
na.action	Function determining what should be done with missing values in newdata. If na.pass (the default) then summaries of NA are produced for missing covariate values. If na.omit, then missing values are dropped, the behaviour of summary.flexsurvreg before flexsurv version 1.2.

Value

A data frame with estimate and confidence limits for the hazard ratio, and one row for each of the times requested in t.

---

lines.flexsurvreg   *Add fitted flexible survival curves to a plot*

Description

Add fitted survival (or hazard or cumulative hazard) curves from a flexsurvreg model fit to an existing plot.

Usage

```
## S3 method for class 'flexsurvreg'
lines(
  x,
  newdata = NULL,
  X = NULL,
  type = "survival",
  t = NULL,
  est = TRUE,
  ci = NULL,
  B = 1000,
  cl = 0.95,
  col = "red",
  lty = 1,
  lwd = 2,
  col.ci = NULL,
  lty.ci = 2,
  lwd.ci = 1,
  ...
)
```

Arguments

x	Output from [flexsurvreg](), representing a fitted survival model object.
newdata	Covariate values to produce fitted curves for, as a data frame, as described in [plot.flexsurvreg]().
X	Covariate values to produce fitted curves for, as a matrix, as described in [plot.flexsurvreg]().
type	"survival" for survival, "cumhaz" for cumulative hazard, or "hazard" for hazard, as in [plot.flexsurvreg]().
t	Vector of times to plot fitted values for.
est	Plot fitted curves (TRUE or FALSE.)
ci	Plot confidence intervals for fitted curves.
B	Number of simulations controlling accuracy of confidence intervals, as used in [summary]().
cl	Width of confidence intervals, by default 0.95 for 95% intervals.
col	Colour of the fitted curve(s).
lty	Line type of the fitted curve(s).
lwd	Line width of the fitted curve(s).
col.ci	Colour of the confidence limits, defaulting to the same as for the fitted curve.
lty.ci	Line type of the confidence limits.
lwd.ci	Line width of the confidence limits, defaulting to the same as for the fitted curve.
...	Other arguments to be passed to the generic [plot]() and [lines]() functions.

Details

Equivalent to [plot.flexsurvreg](...,add=TRUE).

Author(s)

C. H. Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

See Also

[flexsurvreg]()

Llogis	*The log-logistic distribution*

Description

Density, distribution function, hazards, quantile function and random generation for the log-logistic distribution.

Usage

```
dllogis(x, shape = 1, scale = 1, log = FALSE)

pllogis(q, shape = 1, scale = 1, lower.tail = TRUE, log.p = FALSE)

qllogis(p, shape = 1, scale = 1, lower.tail = TRUE, log.p = FALSE)

rllogis(n, shape = 1, scale = 1)

hllogis(x, shape = 1, scale = 1, log = FALSE)

Hllogis(x, shape = 1, scale = 1, log = FALSE)
```

Arguments

x, q	vector of quantiles.
shape, scale	vector of shape and scale parameters.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are $P(X \leq x)$, otherwise, $P(X > x)$.
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required.

Details

The log-logistic distribution with shape parameter $a > 0$ and scale parameter $b > 0$ has probability density function

$$f(x|a,b) = (a/b)(x/b)^{a-1}/(1 + (x/b)^a)^2$$

and hazard

$$h(x|a,b) = (a/b)(x/b)^{a-1}/(1 + (x/b)^a)$$

for $x > 0$. The hazard is decreasing for shape $a \leq 1$, and unimodal for $a > 1$.

The probability distribution function is

$$F(x|a, b) = 1 - 1/(1 + (x/b)^a)$$

If $a > 1$, the mean is $bc/sin(c)$, and if $a > 2$ the variance is $b^2 * (2 * c/sin(2 * c) - c^2/sin(c)^2)$, where $c = \pi/a$, otherwise these are undefined.

Value

dllogis gives the density, pllogis gives the distribution function, qllogis gives the quantile function, hllogis gives the hazard function, Hllogis gives the cumulative hazard function, and rllogis generates random deviates.

Note

Various different parameterisations of this distribution are used. In the one used here, the interpretation of the parameters is the same as in the standard Weibull distribution ([dweibull](#)). Like the Weibull, the survivor function is a transformation of $(x/b)^a$ from the non-negative real line to [0,1], but with a different link function. Covariates on b represent time acceleration factors, or ratios of expected survival.

The same parameterisation is also used in [eha::dllogis](#) in the **eha** package.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

References

Stata Press (2007) Stata release 10 manual: Survival analysis and epidemiological tables.

See Also

[dweibull](#)

---

logLik.flexsurvreg   *Log likelihood from a flexsurvreg model*

Description

Log likelihood from a flexsurvreg model

Usage

```
## S3 method for class 'flexsurvreg'
logLik(object, ...)
```

Arguments

object	A fitted model object of class [flexsurvreg](#), e.g. as returned by flexsurvreg or flexsurvspline.
...	Other arguments (currently unused).

Value

Log-likelihood (numeric) with additional attributes df (degrees of freedom, or number of parameters that were estimated), and number of observations nobs (including observed events and censored observations).

---

meanfinal_fmixmsm   *Mean time to final state in a mixture multi-state model*

Description

Calculate the mean time from the start of the process to a final (or "absorbing") state in a mixture multi-state model. Models with cycles are not supported.

Usage

```
meanfinal_fmixmsm(x, newdata = NULL, final = FALSE, B = NULL)
```

Arguments

x	Object returned by [fmixmsm](#), representing a multi-state model built from piecing together mixture models fitted by [flexsurvmix](#).
newdata	Data frame or list of covariate values. If omitted for a model with covariates, a default is used, defined by all combinations of factors if the only covariates in the model are factors, or all covariate values of zero if there are any non-factor covariates in the model.
final	If TRUE then the mean time to the final state is calculated for each final state, by taking a weighted average of the mean time to travel each pathway ending in that final state, weighted by the probability of the pathway. If FALSE (the default) then a separate mean is calculated for each pathway.
B	Number of simulations to use to compute 95% confidence intervals, based on the asymptotic multivariate normal distribution of the basic parameter estimates. If B=NULL then intervals are not computed.

Value

A data frame of mean times to absorption, by covariate values and pathway (or by final state)

---

mean_exp   *Mean and restricted mean survival functions*

Description

Mean and restricted mean survival time functions for distributions which are built into flexsurv.

Usage

```
mean_exp(rate = 1)

rmst_exp(t, rate = 1, start = 0)

mean_gamma(shape, rate = 1)

rmst_gamma(t, shape, rate = 1, start = 0)

rmst_genf(t, mu, sigma, Q, P, start = 0)

mean_genf(mu, sigma, Q, P)

rmst_genf.orig(t, mu, sigma, s1, s2, start = 0)

mean_genf.orig(mu, sigma, s1, s2)

rmst_gengamma(t, mu = 0, sigma = 1, Q, start = 0)

mean_gengamma(mu = 0, sigma = 1, Q)

rmst_gengamma.orig(t, shape, scale = 1, k, start = 0)

mean_gengamma.orig(shape, scale = 1, k)

rmst_gompertz(t, shape, rate = 1, start = 0)

mean_gompertz(shape, rate = 1)

mean_llogis(shape = 1, scale = 1)

rmst_llogis(t, shape = 1, scale = 1, start = 0)

mean_lnorm(meanlog = 0, sdlog = 1)

rmst_lnorm(t, meanlog = 0, sdlog = 1, start = 0)

mean_weibull(shape, scale = 1)
```

```
rmst_weibull(t, shape, scale = 1, start = 0)

rmst_weibullPH(t, shape, scale = 1, start = 0)

mean_weibullPH(shape, scale = 1)
```

Arguments

rate	Rate parameter (exponential and gamma)
t	Vector of times to which restricted mean survival time is evaluated
start	Optional left-truncation time or times. The returned restricted mean survival will be conditioned on survival up to this time.
shape	Shape parameter (Weibull, gamma, log-logistic, generalized gamma [orig], generalized F [orig])
mu	Mean on the log scale (generalized gamma, generalized F)
sigma	Standard deviation on the log scale (generalized gamma, generalized F)
Q	Vector of first shape parameters (generalized gamma, generalized F)
P	Vector of second shape parameters (generalized F)
s1	Vector of first F shape parameters (generalized F [orig])
s2	vector of second F shape parameters (generalized F [orig])
scale	Scale parameter (Weibull, log-logistic, generalized gamma [orig], generalized F [orig])
k	vector of shape parameters (generalized gamma [orig]).
meanlog	Mean on the log scale (log normal)
sdlog	Standard deviation on the log scale (log normal)

Details

For the exponential, Weibull, log-logistic, lognormal, and gamma, mean survival is provided analytically. Restricted mean survival for the exponential distribution is also provided analytically. Mean and restricted means for other distributions are calculated via numeric integration.

Value

mean survival (functions beginning 'mean') or restricted mean survival (functions beginning 'rmst_').

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

See Also

dexp,dweibull,dgamma,dlnorm,dgompertz,dgengamma,dgenf

---

mean_flexsurvmix   *Mean times to events from a flexsurvmix model*

Description

This returns the mean of each event-specific parametric time-to-event distribution in the mixture model, which is the mean time to event conditionally on that event being the one that happens.

Usage

```
mean_flexsurvmix(x, newdata = NULL, B = NULL)
```

Arguments

x	Fitted model object returned from [flexsurvmix](#).
newdata	Data frame or list of covariate values. If omitted for a model with covariates, a default is used, defined by all combinations of factors if the only covariates in the model are factors, or all covariate values of zero if there are any non-factor covariates in the model.
B	Number of simulations to use to compute 95% confidence intervals, based on the asymptotic multivariate normal distribution of the basic parameter estimates. If B=NULL then intervals are not computed.

Value

Mean times to next event conditionally on each alternative event, given the specified covariate values.

---

model.frame.flexsurvmix
                        Model frame from a flexsurvmix object

---

Description

Returns a list of data frames, with each component containing the data that were used for the model fit for that mixture component.

Usage

```
## S3 method for class 'flexsurvmix'
model.frame(formula, ...)
```

Arguments

formula	Fitted model object from [flexsurvmix](#).
...	Further arguments (currently unused).

Value

A list of data frames

---

model.frame.flexsurvreg

Extract original data from flexsurvreg *objects.*

Description

Extract the data from a model fitted with flexsurvreg.

Usage

```
## S3 method for class 'flexsurvreg'
model.frame(formula, ...)

## S3 method for class 'flexsurvreg'
model.matrix(object, par = NULL, ...)
```

Arguments

formula	A fitted model object, as returned by [flexsurvreg]().
...	Further arguments (not used).
object	A fitted model object, as returned by [flexsurvreg]().
par	String naming the parameter whose linear model matrix is desired.
	The default value of par=NULL returns a matrix consisting of the model matrices for all models in the object cbinded together, with the intercepts excluded. This is not really a "model matrix" in the usual sense, however, the columns directly correspond to the covariate coefficients in the matrix of estimates from the fitted model.

Value

model.frame returns a data frame with all the original variables used for the model fit.

model.matrix returns a design matrix for a part of the model that includes covariates. The required part is indicated by the "par" argument (see above).

Author(s)

C. H. Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

See Also

[flexsurvreg](), [model.frame](), [model.matrix]().

---

msfit.flexsurvreg   *Cumulative intensity function for parametric multi-state models*

Description

Cumulative transition-specific intensity/hazard functions for fully-parametric multi-state or competing risks models, using a piecewise-constant approximation that will allow prediction using the functions in the **mstate** package.

Usage

```
msfit.flexsurvreg(
  object,
  t,
  newdata = NULL,
  variance = TRUE,
  tvar = "trans",
  trans,
  B = 1000
)
```

Arguments

object   Output from [flexsurvreg](#) or [flexsurvspline](#), representing a fitted survival model object.

The model should have been fitted to data consisting of one row for each observed transition and additional rows corresponding to censored times to competing transitions. This is the "long" format, or counting process format, as explained in the **flexsurv** vignette.

The model should contain a categorical covariate indicating the transition. In flexsurv this variable can have any name, indicated here by the tvar argument. In the Cox models demonstrated by **mstate** it is usually included in model formulae as strata(trans), but note that the strata function does not do anything in **flexsurv**. The formula supplied to [flexsurvreg](#) should be precise about which parameters are assumed to vary with the transition type.

Alternatively, if the parameters (including covariate effects) are assumed to be different between different transitions, then a list of transition-specific models can be formed. This list has one component for each permitted transition in the multi-state model. This is more computationally efficient, particularly for larger models and datasets. See the example below, and the vignette.

t   Vector of times. These do not need to be the same as the observed event times, and since the model is parametric, they can be outside the range of the data. A grid of more frequent times will provide a better approximation to the cumulative hazard trajectory for prediction with [probtrans](#) or [mssample](#), at the cost of greater computational expense.

newdata	A data frame specifying the values of covariates in the fitted model, other than the transition number. This must be specified if there are other covariates. The variable names should be the same as those in the fitted model formula. There must be either one value per covariate (the typical situation) or n values per covariate, a different one for each of the n allowed transitions.

variance	Calculate the variances and covariances of the transition cumulative hazards (TRUE or FALSE). This is based on simulation from the normal asymptotic distribution of the estimates, which is computationally-expensive.

tvar	Name of the categorical variable in the model formula that represents the transition number. This should have been defined as a factor, with factor levels that correspond to elements of trans, conventionally a sequence of integers starting from 1. Not required if x is a list of transition-specific models.

trans	Matrix indicating allowed transitions in the multi-state model, in the format understood by **mstate**: a matrix of integers whose r, s entry is i if the ith transition type (reading across rows) is r, s, and has NAs on the diagonal and where the r, s transition is disallowed.

B	Number of simulations from the normal asymptotic distribution used to calculate variances. Decrease for greater speed at the expense of accuracy.

Value

An object of class "msfit", in the same form as the objects used in the **mstate** package. The msfit method from **mstate** returns the equivalent cumulative intensities for Cox regression models fitted with coxph.

Author(s)

C. H. Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

References

Liesbeth C. de Wreede, Marta Fiocco, Hein Putter (2011). **mstate**: An R Package for the Analysis of Competing Risks and Multi-State Models. *Journal of Statistical Software*, 38(7), 1-30. doi:10.18637/jss.v038.i07

Mandel, M. (2013). "Simulation based confidence intervals for functions with complicated derivatives." The American Statistician 67(2):76-81

See Also

flexsurv provides alternative functions designed specifically for predicting from parametric multi-state models without calling **mstate**. These include pmatrix.fs and pmatrix.simfs for the transition probability matrix, and totlos.fs and totlos.simfs for expected total lengths of stay in states. These are generally more efficient than going via **mstate**.

Examples

```
## 3 state illness-death model for bronchiolitis obliterans
## Compare clock-reset / semi-Markov multi-state models
```

```
## Simple exponential model (reduces to Markov)

bexp <- flexsurvreg(Surv(years, status) ~ trans,
                    data=bosms3, dist="exp")
tmat <- rbind(c(NA,1,2),c(NA,NA,3),c(NA,NA,NA))
mexp <- msfit.flexsurvreg(bexp, t=seq(0,12,by=0.1),
                          trans=tmat, tvar="trans", variance=FALSE)


## Cox semi-parametric model within each transition

bcox <- coxph(Surv(years, status) ~ strata(trans), data=bosms3)

if (require("mstate")){

mcox <- mstate::msfit(bcox, trans=tmat)

## Flexible parametric spline-based model

bspl <- flexsurvspline(Surv(years, status) ~ trans + gamma1(trans),
                       data=bosms3, k=3)
mspl <- msfit.flexsurvreg(bspl, t=seq(0,12,by=0.1),
                          trans=tmat, tvar="trans", variance=FALSE)

## Compare fit: exponential model is OK but the spline is better

plot(mcox, lwd=1, xlim=c(0, 12), ylim=c(0,4))
cols <- c("black","red","green")
for (i in 1:3){
    lines(mexp$Haz$time[mexp$Haz$trans==i], mexp$Haz$Haz[mexp$Haz$trans==i],
          col=cols[i], lwd=2, lty=2)
    lines(mspl$Haz$time[mspl$Haz$trans==i], mspl$Haz$Haz[mspl$Haz$trans==i],
          col=cols[i], lwd=3)
}
legend("topright", lwd=c(1,2,3), lty=c(1,2,1),
   c("Cox", "Exponential", "Flexible parametric"), bty="n")


}

## Fit a list of models, one for each transition
## More computationally efficient, but only valid if parameters
## are different between transitions.

## Not run:
bexp.list <- vector(3, mode="list")
for (i in 1:3) {
  bexp.list[[i]] <- flexsurvreg(Surv(years, status) ~ 1, subset=(trans==i),
                                data=bosms3, dist="exp")
}

## The list of models can be passed to this and other functions,
## as if it were a single multi-state model.
```

```
msfit.flexsurvreg(bexp.list, t=seq(0,12,by=0.1), trans=tmat)

## End(Not run)
```

---

nobs.flexsurvreg   *Number of observations contributing to a fitted flexible survival model*

Description

Number of observations contributing to a fitted flexible survival model

Usage

```
## S3 method for class 'flexsurvreg'
nobs(object, cens = TRUE, ...)
```

Arguments

object	Output from [flexsurvreg](#) or [flexsurvspline](#), representing a fitted survival model object.
cens	Include censored observations in the number. TRUE by default. If FALSE then the number of observed events is returned. See [BIC.flexsurvreg](#) for a discussion of the issues with defining the sample size for censored data.
...	Further arguments passed to or from other methods. Currently unused.

Details

By default, this matches the behaviour of the nobs method for [survreg](#) objects, including both censored and uncensored observations.

If a weighted flexsurvreg analysis was done, then this function returns the sum of the weights.

Value

This returns the mod$N component of the fitted model object mod. See [flexsurvreg](#), [flexsurvspline](#) for full documentation of all components.

Author(s)

C. H. Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

See Also

[flexsurvreg](#), [flexsurvspline](#).

---

normboot.flexsurvreg   *Simulate from the asymptotic normal distribution of parameter esti-*
                            mates.

Description

Produce a matrix of alternative parameter estimates under sampling uncertainty, at covariate values
supplied by the user. Used by [summary.flexsurvreg](#) for obtaining confidence intervals around
functions of parameters.

Usage

```
normboot.flexsurvreg(
  x,
  B,
  newdata = NULL,
  X = NULL,
  transform = FALSE,
  raw = FALSE,
  tidy = FALSE,
  rawsim = NULL
)
```

Arguments

x	A fitted model from [flexsurvreg](#) (or [flexsurvspline](#)).
B	Number of samples.
newdata	Data frame or list containing the covariate values to evaluate the parameters at. If there are covariates in the model, at least one of newdata or X must be supplied, unless raw=TRUE.
X	Alternative (less convenient) format for covariate values: a matrix with one row, with one column for each covariate or factor contrast. Formed from all the "model matrices", one for each named parameter of the distribution, with intercepts excluded, cbinded together.
transform	TRUE if the results should be transformed to the real-line scale, typically by log if the parameter is defined as positive. The default FALSE returns parameters on the natural scale.
raw	Return samples of the baseline parameters and the covariate effects, rather than the default of adjusting the baseline parameters for covariates.
tidy	If FALSE (the default) then a list is returned. If TRUE a data frame is returned, consisting of the list elements rbinded together, with integer variables labelling the covariate number and simulation replicate number.
rawsim	allows input of raw samples from a previous run of normboot.flexsurvreg. This is useful if running normboot.flexsurvreg multiple time on the same dataset but with counterfactual contrasts, e.g. treat =0 vs. treat =1. Used in standsurv.flexsurvreg.

Value

If newdata includes only one covariate combination, a matrix will be returned with B rows, and one column for each named parameter of the survival distribution.

If more than one covariate combination is requested (e.g. newdata is a data frame with more than one row), then a list of matrices will be returned, one for each covariate combination.

Author(s)

C. H. Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

References

Mandel, M. (2013). "Simulation based confidence intervals for functions with complicated derivatives." The American Statistician (in press).

See Also

summary.flexsurvreg

Examples

```
fite <- flexsurvreg(Surv(futime, fustat) ~ age, data = ovarian, dist="exp")
normboot.flexsurvreg(fite, B=10, newdata=list(age=50))
normboot.flexsurvreg(fite, B=10, X=matrix(50,nrow=1))
normboot.flexsurvreg(fite, B=10, newdata=list(age=0))  ## closer to...
fite$res
```

---

pars.fmsm	*Transition-specific parameters in a flexible parametric multi-state model*

Description

List of maximum likelihood estimates of transition-specific parameters in a flexible parametric multi-state model, at given covariate values.

Usage

```
pars.fmsm(x, trans, newdata = NULL, tvar = "trans")
```

Arguments

x	A multi-state model fitted with flexsurvreg. See msfit.flexsurvreg for the required form of the model and the data.
	x can also be a list of flexsurvreg models, with one component for each permitted transition in the multi-state model, as illustrated in msfit.flexsurvreg.
trans	Matrix indicating allowed transitions. See msfit.flexsurvreg.

newdata   A data frame specifying the values of covariates in the fitted model, other than
                 the transition number. See `msfit.flexsurvreg`.

tvar   Variable in the data representing the transition type. Not required if x is a list of
                 models.

Value

A list with one component for each permitted transition. Each component has one element for each
parameter of the parametric distribution that generates the corresponding event in the multi-state
model.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>.

---

pdf_flexsurvmix   *Fitted densities for times to events in a flexsurvmix model*

Description

This returns an estimate of the probability density for the time to each competing event, at a vector
of times supplied by the user.

Usage

```
pdf_flexsurvmix(x, newdata = NULL, t = NULL)
```

Arguments

x   Fitted model object returned from `flexsurvmix`.

newdata   Data frame or list of covariate values. If omitted for a model with covariates, a
                 default is used, defined by all combinations of factors if the only covariates in
                 the model are factors, or all covariate values of zero if there are any non-factor
                 covariates in the model.

t   Vector of times at which to evaluate the probability density

Value

A data frame with each row giving the fitted density dens for a combination of covariate values,
time and competing event.

---

pfinal_fmsm	*Probabilities of final states in a flexible parametric competing risks model*

Description

This requires the model to be Markov, and is not valid for semi-Markov models, as it works by wrapping [pmatrix.fs](#) to calculate the transition probability over a very large time. As it also works on a fmsm object formed from transition-specific time-to-event models, it therefore only works on competing risks models, defined by just one starting state with multiple destination states representing competing events. For these models, this function returns the probability governing which competing event happens next. However this function simply wraps pmatrix.fs, so for other models, pmatrix.fs or pmatrix.simfs can be used with a large forecast time t.

Usage

```
pfinal_fmsm(x, newdata = NULL, fromstate, maxt = 1e+05, B = 0, cores = NULL)
```

Arguments

x	Object returned by [fmsm](#), representing a multi-state model formed from transition-specific time-to-event models fitted by [flexsurvreg](#).
newdata	Data frame of covariate values, with one column per covariate, and one row per alternative value.
fromstate	State from which to calculate the transition probability state. This should refer to the name of a row of the transition matrix attr(x,trans).
maxt	Large time to use for forecasting final state probabilities. The transition probability from zero to this time is used. Note Inf will not work. The default is 100000.
B	Number of simulations to use to calculate 95% confidence intervals based on the asymptotic normal distribution of the basic parameter estimates. If B=0 then no intervals are calculated.
cores	Number of processor cores to use. If NULL (the default) then a single core is used.

Value

A data frame with one row per covariate value and destination state, giving the state in column state, and probability in column val. Additional columns lower and upper for the confidence limits are returned if B=0.

---

plot.flexsurvreg   *Plots of fitted flexible survival models*

Description

Plot fitted survival, cumulative hazard or hazard from a parametric model against nonparametric estimates to diagnose goodness-of-fit. Alternatively plot a user-defined function of the model parameters against time.

Usage

```
## S3 method for class 'flexsurvreg'
plot(
  x,
  newdata = NULL,
  X = NULL,
  type = "survival",
  fn = NULL,
  t = NULL,
  start = 0,
  est = TRUE,
  ci = NULL,
  B = 1000,
  cl = 0.95,
  col.obs = "black",
  lty.obs = 1,
  lwd.obs = 1,
  col = "red",
  lty = 1,
  lwd = 2,
  col.ci = NULL,
  lty.ci = 2,
  lwd.ci = 1,
  ylim = NULL,
  add = FALSE,
  ...
)
```

Arguments

x	Output from [flexsurvreg](#) or [flexsurvspline](#), representing a fitted survival model object.
newdata	Data frame containing covariate values to produce fitted values for. See [summary.flexsurvreg](#).
	If there are only factor covariates in the model, then Kaplan-Meier (or nonparametric hazard...) curves are plotted for all distinct groups, and by default, fitted curves are also plotted for these groups. To plot Kaplan-Meier and fitted curves for only a subset of groups, use `plot(survfit())` followed by `lines.flexsurvreg()`.

	If there are any continuous covariates, then a single population Kaplan-Meier curve is drawn. By default, a single fitted curve is drawn with the covariates set to their mean values in the data - for categorical covariates, the means of the 0/1 indicator variables are taken.
X	Alternative way to supply covariate values, as a model matrix. See `summary.flexsurvreg`. newdata is an easier way.
type	`"survival"` for survival, to be plotted against Kaplan-Meier estimates from `plot.survfit`.
	`"cumhaz"` for cumulative hazard, plotted against transformed Kaplan-Meier estimates from `plot.survfit`.
	`"hazard"` for hazard, to be plotted against smooth nonparametric estimates from `muhaz`. The nonparametric estimates tend to be unstable, and these plots are intended just to roughly indicate the shape of the hazards through time. The `min.time` and `max.time` options to `muhaz` may sometimes need to be passed as arguments to `plot.flexsurvreg` to avoid an error here.
	Ignored if `"fn"` is specified.
fn	Custom function of the parameters to summarise against time. The first two arguments of the function must be `t` representing time, and `start` representing left-truncation points, and any remaining arguments must be parameters of the distribution. It should return a vector of the same length as `t`.
t	Vector of times to plot fitted values for, see `summary.flexsurvreg`.
start	Left-truncation points, see `summary.flexsurvreg`.
est	Plot fitted curves (TRUE or FALSE.)
ci	Plot confidence intervals for fitted curves. By default, this is TRUE if one observed/fitted curve is plotted, and FALSE if multiple curves are plotted.
B	Number of simulations controlling accuracy of confidence intervals, as used in `summary`. Decrease for greater speed at the expense of accuracy, or set B=0 to turn off calculation of CIs.
cl	Width of confidence intervals, by default 0.95 for 95% intervals.
col.obs	Colour of the nonparametric curve.
lty.obs	Line type of the nonparametric curve.
lwd.obs	Line width of the nonparametric curve.
col	Colour of the fitted parametric curve(s).
lty	Line type of the fitted parametric curve(s).
lwd	Line width of the fitted parametric curve(s).
col.ci	Colour of the fitted confidence limits, defaulting to the same as for the fitted curve.
lty.ci	Line type of the fitted confidence limits.
lwd.ci	Line width of the fitted confidence limits.
ylim	y-axis limits: vector of two elements.
add	If TRUE, add lines to an existing plot, otherwise new axes are drawn.
...	Other options to be passed to `plot.survfit` or `muhaz`, for example, to control the smoothness of the nonparametric hazard estimates. The `min.time` and `max.time` options to `muhaz` may sometimes need to be changed from the defaults.

Note

Some standard plot arguments such as "xlim","xlab" may not work. This function was designed as a quick check of model fit. Users wanting publication-quality graphs are advised to set up an empty plot with the desired axes first (e.g. with plot(...,type="n",...)), then use suitable [lines](#) functions to add lines.

If case weights were used to fit the model, these are used when producing nonparametric estimates of survival and cumulative hazard, but not for the hazard estimates.

Author(s)

C. H. Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

See Also

[flexsurvreg](#)

---

plot.standsurv   *Plot standardized metrics from a fitted flexsurv model*

Description

Plot standardized metrics such as the marginal survival, restricted mean survival and hazard, based on a fitted flexsurv model.

Usage

```
## S3 method for class 'standsurv'
plot(x, contrast = FALSE, ci = FALSE, expected = FALSE, ...)
```

Arguments

x	A standsurv object returned by standsurv
contrast	Should contrasts of standardized metrics be plotted. Defaults to FALSE
ci	Should confidence intervals be plotted (if calculated in standsurv)?
expected	Should the marginal expected survival / hazard also be plotted? This can only be invoked if rmap and ratetable have been passed to standsurv
...	Not currently used

Value

A ggplot showing the standardized metric calculated by standsurv over time. Modification of the plot is possible by adding further ggplot objects, see Examples.

Examples

```
## Use bc dataset, with an age variable appended
## mean age is higher in those with smaller observed survival times
newbc <- bc
newbc$age <- rnorm(dim(bc)[1], mean = 65-scale(newbc$recyrs, scale=FALSE),
sd = 5)

## Fit a Weibull flexsurv model with group and age as covariates
weib_age <- flexsurvreg(Surv(recyrs, censrec) ~ group+age, data=newbc,
                        dist="weibull")
## Calculate standardized survival and the difference in standardized survival
## for the three levels of group across a grid of survival times
standsurv_weib_age <- standsurv(weib_age,
                                      at = list(list(group="Good"),
                                                list(group="Medium"),
                                                list(group="Poor")),
                                      t=seq(0,7, length=100),
                                      contrast = "difference", ci=TRUE,
                                      boot = TRUE, B=10, seed=123)
plot(standsurv_weib_age)
plot(standsurv_weib_age) + ggplot2::theme_bw() + ggplot2::ylab("Survival") +
 ggplot2::xlab("Time (years)") +
 ggplot2::guides(color=ggplot2::guide_legend(title="Prognosis"),
                             fill=ggplot2::guide_legend(title="Prognosis"))
plot(standsurv_weib_age, contrast=TRUE, ci=TRUE) +
 ggplot2::ylab("Difference in survival")
```

---

plot.survrtrunc	*Plot nonparametric estimates of survival from right-truncated data.*

Description

`plot.survrtrunc` creates a new plot, while `lines.survrtrunc` adds lines to an exising plot.

Usage

```
## S3 method for class 'survrtrunc'
plot(x, ...)

## S3 method for class 'survrtrunc'
lines(x, ...)
```

Arguments

x	Object of class "survrtrunc" as returned by survrtrunc.
...	Other arguments to be passed to plot.survfit or lines.survfit.

---

pmatrix.fs	*Transition probability matrix from a fully-parametric, time-inhomogeneous Markov multi-state model*

Description

The transition probability matrix for time-inhomogeneous Markov multi-state models fitted to time-to-event data with [flexsurvreg](#). This has r, s entry giving the probability that an individual is in state s at time t, given they are in state r at time 0.

Usage

```
pmatrix.fs(
  x,
  trans = NULL,
  t = 1,
  newdata = NULL,
  condstates = NULL,
  ci = FALSE,
  tvar = "trans",
  sing.inf = 1e+10,
  B = 1000,
  cl = 0.95,
  tidy = FALSE,
  ...
)
```

Arguments

x	A model fitted with [flexsurvreg](#). See [msfit.flexsurvreg](#) for the required form of the model and the data. Additionally, this must be a Markov / clock-forward model, but can be time-inhomogeneous. See the package vignette for further explanation.
	x can also be a list of models, with one component for each permitted transition, as illustrated in [msfit.flexsurvreg](#).
trans	Matrix indicating allowed transitions. See [msfit.flexsurvreg](#).
t	Time or vector of times to predict state occupancy probabilities for.
newdata	A data frame specifying the values of covariates in the fitted model, other than the transition number. See [msfit.flexsurvreg](#).
condstates	xInstead of the unconditional probability of being in state s at time t given state r at time 0, return the probability conditional on being in a particular subset of states at time t. This subset is specified in the condstates argument, as a vector of character labels or integers.
	This is used, for example, in competing risks situations, e.g. if the competing states are death or recovery from a disease, and we want to compute the probability a patient has died, given they have died or recovered. If these are absorbing

states, then as t increases, this converges to the case fatality ratio. To compute this, set t to a very large number, Inf will not work.

ci   Return a confidence interval calculated by simulating from the asymptotic normal distribution of the maximum likelihood estimates. Turned off by default, since this is computationally intensive. If turned on, users should increase B until the results reach the desired precision.

tvar   Variable in the data representing the transition type. Not required if x is a list of models.

sing.inf   If there is a singularity in the observed hazard, for example a Weibull distribution with shape < 1 has infinite hazard at t=0, then as a workaround, the hazard is assumed to be a large finite number, sing.inf, at this time. The results should not be sensitive to the exact value assumed, but users should make sure by adjusting this parameter in these cases.

B   Number of simulations from the normal asymptotic distribution used to calculate variances. Decrease for greater speed at the expense of accuracy.

cl   Width of symmetric confidence intervals, relative to 1.

tidy   If TRUE then return the results as a tidy data frame

...   Arguments passed to [ode](#) in **deSolve**.

Details

This is computed by solving the Kolmogorov forward differential equation numerically, using the methods in the **deSolve** package. The equation is

$$\frac{dP(t)}{dt} = P(t)Q(t)$$

where $P(t)$ is the transition probability matrix for time t, and $Q(t)$ is the transition hazard or intensity as a function of t. The initial condition is $P(0) = I$.

Note that the package **msm** has a similar method pmatrix.msm. pmatrix.fs should give the same results as pmatrix.msm when both of these conditions hold:

- the time-to-event distribution is exponential for all transitions, thus the flexsurvreg model was fitted with dist="exp" and the model is time-homogeneous.

- the **msm** model was fitted with exacttimes=TRUE, thus all the event times are known, and there are no time-dependent covariates.

msm only allows exponential or piecewise-exponential time-to-event distributions, while **flexsurvreg** allows more flexible models. **msm** however was designed in particular for panel data, where the process is observed only at arbitrary times, thus the times of transition are unknown, which makes flexible models difficult.

This function is only valid for Markov ("clock-forward") multi-state models, though no warning or error is currently given if the model is not Markov. See [pmatrix.simfs](#) for the equivalent for semi-Markov ("clock-reset") models.

Value

The transition probability matrix, if t is of length 1. If t is longer, return a list of matrices, or a data frame if tidy is TRUE.

If ci=TRUE, each element has attributes "lower" and "upper" giving matrices of the corresponding confidence limits. These are formatted for printing but may be extracted using attr().

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>.

See Also

pmatrix.simfs, totlos.fs, msfit.flexsurvreg.

Examples

```
# BOS example in vignette, and in msfit.flexsurvreg
bexp <- flexsurvreg(Surv(Tstart, Tstop, status) ~ trans,
                    data=bosms3, dist="exp")
tmat <- rbind(c(NA,1,2),c(NA,NA,3),c(NA,NA,NA))
# more likely to be dead (state 3) as time moves on, or if start with
# BOS (state 2)
pmatrix.fs(bexp, t=c(5,10), trans=tmat)
```

---

pmatrix.simfs   *Transition probability matrix from a fully-parametric, semi-Markov multi-state model*

Description

The transition probability matrix for semi-Markov multi-state models fitted to time-to-event data with flexsurvreg. This has r, s entry giving the probability that an individual is in state s at time t, given they are in state r at time 0.

Usage

```
pmatrix.simfs(
  x,
  trans,
  t = 1,
  newdata = NULL,
  ci = FALSE,
  tvar = "trans",
  tcovs = NULL,
  M = 1e+05,
  B = 1000,
  cl = 0.95,
```

```
    cores = NULL,
    tidy = FALSE
)
```

Arguments

x	A model fitted with [flexsurvreg](). See [msfit.flexsurvreg]() for the required form of the model and the data. Additionally this should be semi-Markov, so that the time variable represents the time since the last transition. In other words the response should be of the form Surv(time,status). See the package vignette for further explanation.
	x can also be a list of [flexsurvreg]() models, with one component for each permitted transition, as illustrated in [msfit.flexsurvreg](). This can be constructed by [fmsm]().
trans	Matrix indicating allowed transitions. See [msfit.flexsurvreg](). This is not required if x is a list constructed by [fmsm]().
t	Time to predict state occupancy probabilities for. This can be a single number or a vector of different numbers.
newdata	A data frame specifying the values of covariates in the fitted model, other than the transition number. See [msfit.flexsurvreg]().
ci	Return a confidence interval calculated by simulating from the asymptotic normal distribution of the maximum likelihood estimates. This is turned off by default, since two levels of simulation are required. If turned on, users should adjust B and/or M until the results reach the desired precision. The simulation over M is generally vectorised, therefore increasing B is usually more expensive than increasing M.
tvar	Variable in the data representing the transition type. Not required if x is a list of models.
tcovs	Predictable time-dependent covariates such as age, see [sim.fmsm]().
M	Number of individuals to simulate in order to approximate the transition probabilities. Users should adjust this to obtain the required precision.
B	Number of simulations from the normal asymptotic distribution used to calculate confidence limits. Decrease for greater speed at the expense of accuracy.
cl	Width of symmetric confidence intervals, relative to 1.
cores	Number of processor cores used when calculating confidence limits by repeated simulation. The default uses single-core processing.
tidy	If TRUE then the results are returned as a tidy data frame with columns for the estimate and confidence limits, and rows per state transition and time interval.

Details

This is computed by simulating a large number of individuals M using the maximum likelihood estimates of the fitted model and the function [sim.fmsm](). Therefore this requires a random sampling function for the parametric survival model to be available: see the "Details" section of [sim.fmsm](). This will be available for all built-in distributions, though users may need to write this for custom models.

Note the random sampling method for `flexsurvspline` models is currently very inefficient, so that looping over the `M` individuals will be very slow.

`pmatrix.fs` is a more efficient method based on solving the Kolmogorov forward equation numerically, which requires the multi-state model to be Markov. No error or warning is given if running `pmatrix.simfs` with a Markov model, but this is still invalid.

Value

The transition probability matrix. If `ci=TRUE`, there are attributes `"lower"` and `"upper"` giving matrices of the corresponding confidence limits. These are formatted for printing but may be extracted using `attr()`.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>.

See Also

`pmatrix.fs`,`sim.fmsm`,`totlos.simfs`, `msfit.flexsurvreg`.

Examples

```
# BOS example in vignette, and in msfit.flexsurvreg

bexp <- flexsurvreg(Surv(years, status) ~ trans, data=bosms3, dist="exp")
tmat <- rbind(c(NA,1,2),c(NA,NA,3),c(NA,NA,NA))

# more likely to be dead (state 3) as time moves on, or if start with
# BOS (state 2)

pmatrix.simfs(bexp, t=5, trans=tmat)
pmatrix.simfs(bexp, t=10, trans=tmat)

# these results should converge to those in help(pmatrix.fs), as M
# increases here and ODE solving precision increases there, since with
# an exponential distribution, the semi-Markov model is the same as the
# Markov model.
```

---

ppath_fmixmsm   *Probability of each pathway taken through a mixture multi-state model*

Description

Probability of each pathway taken through a mixture multi-state model

Usage

```
ppath_fmixmsm(x, newdata = NULL, final = FALSE, B = NULL)
```

Arguments

x	Object returned by [fmixmsm](#), representing a multi-state model built from piecing together mixture models fitted by [flexsurvmix](#).
newdata	Data frame or list of covariate values. If omitted for a model with covariates, a default is used, defined by all combinations of factors if the only covariates in the model are factors, or all covariate values of zero if there are any non-factor covariates in the model.
final	If TRUE then the probabilities of pathways with the same final state are added together, to produce the probability of each ultimate outcome or absorbing state from the multi-state model.
B	Number of simulations to use to compute 95% confidence intervals, based on the asymptotic multivariate normal distribution of the basic parameter estimates. If B=NULL then intervals are not computed.

Value

Data frame of pathway probabilities by covariate value and pathway.

---

predict.flexsurvreg   *Predictions from flexible survival models*

Description

Predict outcomes from flexible survival models at the covariate values specified in newdata.

Usage

```
## S3 method for class 'flexsurvreg'
predict(
  object,
  newdata,
  type = "response",
  times,
  start = 0,
  conf.int = FALSE,
  conf.level = 0.95,
  se.fit = FALSE,
  p = c(0.1, 0.9),
  ...
)
```

Arguments

object	Output from [flexsurvreg](#) or [flexsurvspline](#), representing a fitted survival model object.

newdata   Data frame containing covariate values at which to produce fitted values. There
                 must be a column for every covariate in the model formula used to fit object,
                 and one row for every combination of covariate values at which to obtain the
                 fitted predictions.

                 If newdata is omitted, then the original data used to fit the model are used, as
                 extracted by model.frame(object). However this will currently not work if
                 the model formula contains functions, e.g. ~ factor(x). The names of the
                 model frame must correspond to variables in the original data.

type   Character vector for the type of predictions desired.

                   • "response" for mean survival time (the default). "mean" is an acceptable
                     synonym
                   • "quantile" for quantiles of the survival distribution as specified by p
                   • "rmst" for restricted mean survival time
                   • "survival" for survival probabilities
                   • "cumhaz" for cumulative hazards
                   • "hazard" for hazards
                   • "link" for fitted values of the location parameter, analogous to the linear
                     predictor in generalized linear models (type = "lp" and type = "linear"
                     are acceptable synonyms). This is on the natural scale of the parameter, not
                     the log scale.

times   Vector of time horizons at which to compute fitted values. Only applies when
                 type is "survival", "cumhaz", "hazard", or "rmst". Will be silently ignored
                 for all other types.

                 If not specified, predictions for "survival", "cumhaz", and "hazard" will be
                 made at each observed event time in model.frame(object).

                 For "rmst", when times is not specified predictions will be made at the maxi-
                 mum observed event time from the data used to fit object. Specifying times =
                 Inf is valid, and will return mean survival (equal to type = "response").

start   Optional left-truncation time or times. The returned survival, hazard, or cumula-
                 tive hazard will be conditioned on survival up to this time. start must be length
                 1 or the same length as times. Predicted times returned with type "rmst" or
                 "quantile" will be times since time zero, not times since the start time.

conf.int   Logical. Should confidence intervals be returned? Default is FALSE.

conf.level   Width of symmetric confidence intervals, relative to 1.

se.fit   Logical. Should standard errors of fitted values be returned? Default is FALSE.

p   Vector of quantiles at which to return fitted values when type = "quantile".
                 Default is c(0.1, 0.9).

...   Not currently used.

Value

A tibble with same number of rows as newdata and in the same order. If multiple predictions are
requested, a tibble containing a single list-column of data frames.

For the list-column of data frames - the dimensions of each data frame will be identical. Rows are
added for each value of times or p requested.

This function is a wrapper around `summary.flexsurvreg`, designed to help **flexsurv** to integrate with the "tidymodels" ecosystem, in particular the **censored** package. `summary.flexsurvreg` returns the same results but in a more conventional format.

Author(s)

Matthew T. Warkentin (<https://github.com/mattwarkentin>)

See Also

`summary.flexsurvreg`, `residuals.flexsurvreg`

Examples

```
fitg <- flexsurvreg(formula = Surv(futime, fustat) ~ age, data = ovarian, dist = "gengamma")

## Simplest prediction: mean or median, for covariates defined by original dataset
predict(fitg)
predict(fitg, type = "quantile", p = 0.5)

## Simple prediction for user-defined covariate values
predict(fitg, newdata = data.frame(age = c(40, 50, 60)))
predict(fitg, type = "quantile", p = 0.5, newdata = data.frame(age = c(40,50,60)))

## Predict multiple quantiles and unnest
require(tidyr)
pr <- predict(fitg, type = "survival", times = c(600, 800))
tidyr::unnest(pr, .pred)
```

---

probs_flexsurvmix   *Probabilities of competing events from a flexsurvmix model*

Description

Probabilities of competing events from a flexsurvmix model

Usage

```
probs_flexsurvmix(x, newdata = NULL, B = NULL)
```

Arguments

x   Fitted model object returned from `flexsurvmix`.

newdata   Data frame or list of covariate values. If omitted for a model with covariates, a default is used, defined by all combinations of factors if the only covariates in the model are factors, or all covariate values of zero if there are any non-factor covariates in the model.

| B | Number of simulations to use to compute 95% confidence intervals, based on the asymptotic multivariate normal distribution of the basic parameter estimates. If B=NULL then intervals are not computed. |

Value

A data frame containing the probability that each of the competing events will occur next, by event and by any covariate values specified in newdata.

---

p_flexsurvmix   *Transition probabilities from a flexsurvmix model*

Description

These quantities are variously known as transition probabilities, or state occupancy probabilities, or values of the "cumulative incidence" function, or values of the "subdistribution" function. They are the probabilities that an individual has experienced an event of a particular kind by time t.

Usage

```
p_flexsurvmix(x, newdata = NULL, startname = "start", t = 1, B = NULL)
```

Arguments

x	Fitted model object returned from [flexsurvmix]().
newdata	Data frame or list of covariate values. If omitted for a model with covariates, a default is used, defined by all combinations of factors if the only covariates in the model are factors, or all covariate values of zero if there are any non-factor covariates in the model.
startname	Name of the state where individuals start. This considers the model as a multistate model where people start in this state, and may transition to one of the competing events.
t	Vector of times t to calculate the probabilities of transition by.
B	Number of simulations to use to compute 95% confidence intervals, based on the asymptotic multivariate normal distribution of the basic parameter estimates. If B=NULL then intervals are not computed.

Details

Note that "cumulative incidence" is a misnomer, as "incidence" typically means a hazard, and the quantities computed here are not cumulative hazards, but probabilities.

Value

A data frame with transition probabilities by time, covariate value and destination state.

---

qfinal_fmixmsm	*Quantiles of the distribution of the time until reaching a final state in a mixture multi-state model*

Description

Calculate the quantiles of the time from the start of the process to each possible final (or "absorbing") state in a mixture multi-state model. Models with cycles are not supported.

Usage

```
qfinal_fmixmsm(
  x,
  newdata = NULL,
  final = FALSE,
  B = NULL,
  n = 10000,
  probs = c(0.025, 0.5, 0.975)
)
```

Arguments

x	Object returned by [fmixmsm](#), representing a multi-state model built from piecing together mixture models fitted by [flexsurvmix](#).
newdata	Data frame or list of covariate values. If omitted for a model with covariates, a default is used, defined by all combinations of factors if the only covariates in the model are factors, or all covariate values of zero if there are any non-factor covariates in the model.
final	If `TRUE` then the mean time to the final state is calculated for each final state, by taking a weighted average of the mean time to travel each pathway ending in that final state, weighted by the probability of the pathway. If `FALSE` (the default) then a separate mean is calculated for each pathway.
B	Number of simulations to use to compute 95% confidence intervals, based on the asymptotic multivariate normal distribution of the basic parameter estimates. If B=NULL then intervals are not computed.
n	Number of individual-level simulations to use to characterise the time-to-event distributions
probs	Quantiles to calculate, by default, c(0.025, 0.5, 0.975)

Value

Data frame of quantiles of the time to final state by pathway and covariate value, or by final state and covariate value.

---

qgeneric   *Generic function to find quantiles of a distribution*

Description

Generic function to find the quantiles of a distribution, given the equivalent probability distribution function.

Usage

```
qgeneric(pdist, p, matargs = NULL, scalarargs = NULL, ...)
```

Arguments

pdist   Probability distribution function, for example, [pnorm](#) for the normal distribution, which must be defined in the current workspace. This should accept and return vectorised parameters and values. It should also return the correct values for the entire real line, for example a positive distribution should have pdist(x)==0 for $x < 0$.

p   Vector of probabilities to find the quantiles for.

matargs   Character vector giving the elements of ... which represent vector parameters of the distribution. Empty by default. When vectorised, these will become matrices. This is used for the arguments gamma and knots in [qsurvspline](#).

scalarargs   Character vector naming scalar arguments of the distribution function that cannot be vectorised. This is used for the arguments scale and timescale in [qsurvspline](#).

...   The remaining arguments define parameters of the distribution pdist. These MUST be named explicitly.

                This may also contain the standard arguments log.p (logical; default FALSE, if TRUE, probabilities p are given as log(p)), and lower.tail (logical; if TRUE (default), probabilities are P[X <= x] otherwise, P[X > x].).

                If the distribution is bounded above or below, then this should contain arguments lbound and ubound respectively, and these will be returned if p is 0 or 1 respectively. Defaults to -Inf and Inf respectively.

Details

This function is used by default for custom distributions for which a quantile function is not provided.

It works by finding the root of the equation $h(q) = pdist(q) - p = 0$. Starting from the interval $(-1, 1)$, the interval width is expanded by 50% until $h()$ is of opposite sign at either end. The root is then found using [uniroot](#).

This assumes a suitably smooth, continuous distribution.

Value

Vector of quantiles of the distribution at p.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

Examples

```
qnorm(c(0.025, 0.975), 0, 1)
qgeneric(pnorm, c(0.025, 0.975), mean=0, sd=1) # must name the arguments
```

---

quantile_flexsurvmix   *Quantiles of time-to-event distributions in a flexsurvmix model*

Description

This returns the quantiles of each event-specific parametric time-to-event distribution in the mixture model, which describes the time to the event conditionally on that event being the one that happens.

Usage

```
quantile_flexsurvmix(x, newdata = NULL, B = NULL, probs = c(0.025, 0.5, 0.975))
```

Arguments

x	Fitted model object returned from [flexsurvmix]().
newdata	Data frame or list of covariate values. If omitted for a model with covariates, a default is used, defined by all combinations of factors if the only covariates in the model are factors, or all covariate values of zero if there are any non-factor covariates in the model.
B	Number of simulations to use to compute 95% confidence intervals, based on the asymptotic multivariate normal distribution of the basic parameter estimates. If B=NULL then intervals are not computed.
probs	Vector of alternative quantiles, by default c(0.025, 0.95, 0.975) giving the median and a 95% interval.

---

residuals.flexsurvreg   *Calculate residuals for flexible survival models*

Description

Calculates residuals for flexsurvreg or flexsurvspline model fits.

Usage

```
## S3 method for class 'flexsurvreg'
residuals(object, type = "response", ...)
```

Arguments

object	Output from flexsurvreg or flexsurvspline, representing a fitted survival model object.
type	Character string for the type of residual desired. Currently only "response" and "coxsnell" are supported. More residual types may become available in future versions.
...	Not currently used.

Details

Residuals of type = "response" are calculated as the naive difference between the observed survival and the covariate-specific predicted mean survival from predict.flexsurvreg, ignoring whether the event time is observed or censored.

type="coxsnell" returns the Cox-Snell residual, defined as the estimated cumulative hazard at each data point. To check the fit of the A more fully featured utility for this is provided in the function coxsnell_flexsurvreg.

Value

Numeric vector with the same length as nobs(object).

See Also

predict.flexsurvreg

Examples

```
fitg <- flexsurvreg(formula = Surv(futime, fustat) ~ age, data = ovarian, dist = "gengamma")
residuals(fitg, type="response")
```

---

rmst_flexsurvmix   *Restricted mean times to events from a flexsurvmix model*

Description

This returns the restricted mean of each event-specific parametric time-to-event distribution in the mixture model, which is the mean time to event conditionally on that event being the one that happens, and conditionally on the event time being less than some time horizon `tot`.

Usage

```
rmst_flexsurvmix(x, newdata = NULL, tot = Inf, B = NULL)
```

Arguments

x   Fitted model object returned from [flexsurvmix](#).

newdata   Data frame or list of covariate values. If omitted for a model with covariates, a default is used, defined by all combinations of factors if the only covariates in the model are factors, or all covariate values of zero if there are any non-factor covariates in the model.

tot   Time horizon to compute the restricted mean until.

B   Number of simulations to use to compute 95% confidence intervals, based on the asymptotic multivariate normal distribution of the basic parameter estimates. If B=NULL then intervals are not computed.

Value

Restricted mean times to next event conditionally on each alternative event, given the specified covariate values.

---

rmst_generic   *Generic function to find restricted mean survival time for some distribution*

Description

Generic function to find the restricted mean of a distribution, given the equivalent probability distribution function, using numeric integration.

Usage

```
rmst_generic(pdist, t, start = 0, matargs = NULL, scalarargs = NULL, ...)
```

Arguments

pdist	Probability distribution function, for example, [pnorm](#) for the normal distribution, which must be defined in the current workspace. This should accept and return vectorised parameters and values. It should also return the correct values for the entire real line, for example a positive distribution should have pdist(x)==0 for $x < 0$.
t	Vector of times at which rmst is evaluated
start	Optional left-truncation time or times. The returned restricted mean survival will be conditioned on survival up to this time.
matargs	Character vector giving the elements of ... which represent vector parameters of the distribution. Empty by default. When vectorised, these will become matrices. This is used for the arguments gamma and knots in [psurvspline](#).
scalarargs	Character vector naming scalar arguments of the distribution function that cannot be vectorised. This is used, for example, for the arguments scale and timescale in [psurvspline](#).
...	The remaining arguments define parameters of the distribution pdist. These MUST be named explicitly.

Details

This function is used by default for custom distributions for which an rmst function is not provided.

This assumes a suitably smooth, continuous distribution.

Value

Vector of restricted mean survival times of the distribution at p.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

Examples

```
rmst_lnorm(500, start=250, meanlog=7.4225, sdlog = 1.1138)
rmst_generic(plnorm, 500, start=250, meanlog=7.4225, sdlog = 1.1138)
# must name the arguments
```

---

sim.fmsm	*Simulate paths through a fully parametric semi-Markov multi-state model*

Description

Simulate changes of state and transition times from a semi-Markov multi-state model fitted using [flexsurvreg](#).

Usage

```
sim.fmsm(
  x,
  trans = NULL,
  t,
  newdata = NULL,
  start = 1,
  M = 10,
  tvar = "trans",
  tcovs = NULL,
  tidy = FALSE
)
```

Arguments

x	A model fitted with [flexsurvreg](). See [msfit.flexsurvreg]() for the required form of the model and the data.
	Alternatively x can be a list of fitted [flexsurvreg]() model objects. The ith element of this list is the model corresponding to the ith transition in trans. This is a more efficient way to fit a multi-state model, but only valid if the parameters are different between different transitions.
trans	Matrix indicating allowed transitions. See [msfit.flexsurvreg]().
t	Time, or vector of times for each of the M individuals, to simulate trajectories until.
newdata	A data frame specifying the values of covariates in the fitted model, other than the transition number. See [msfit.flexsurvreg]().
start	Starting state, or vector of starting states for each of the M individuals.
M	Number of individual trajectories to simulate.
tvar	Variable in the data representing the transition type. Not required if x is a list of models.
tcovs	Names of "predictable" time-dependent covariates in newdata, i.e. those whose values change at the same rate as time. Age is a typical example. During simulation, their values will be updated after each transition time, by adding the current time to the value supplied in newdata. This assumes the covariate is measured in the same unit as time. tcovs is supplied as a character vector.
tidy	If TRUE then the simulated data are returned as a tidy data frame with one row per simulated transition. See [simfs_bytrans]() for a function to rearrange the data into this format if it was simulated in non-tidy format. Currently this includes only event times, and excludes any times of censoring that are reported when tidy=FALSE.

Details

sim.fmsm relies on the presence of a function to sample random numbers from the parametric survival distribution used in the fitted model x, for example [rweibull]() for Weibull models. If x was fitted using a custom distribution, called dist say, then there must be a function called

(something like) rdist either in the working environment, or supplied through the dfns argument to [flexsurvreg](). This must be in the same format as standard R functions such as [rweibull](), with first argument n, and remaining arguments giving the parameters of the distribution. It must be vectorised with respect to the parameter arguments.

This function is only valid for semi-Markov ("clock-reset") models, though no warning or error is currently given if the model is not of this type. An equivalent for time-inhomogeneous Markov ("clock-forward") models has currently not been implemented.

Value

If tidy=TRUE, a data frame with one row for each simulated transition, giving the individual ID id, start state start, end state end, transition label trans, time of the transition since the start of the process (time), and time since the previous transition (delay).

If tidy=FALSE, a list of two matrices named st and t. The rows of each matrix represent simulated individuals. The columns of t contain the times when the individual changes state, to the corresponding states in st.

The first columns will always contain the starting states and the starting times. The last column of t represents either the time when the individual moves to an absorbing state, or right-censoring in a transient state at the time given in the t argument to sim.fmsm.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>.

See Also

[pmatrix.simfs](),[totlos.simfs]()

Examples

```
bexp <- flexsurvreg(Surv(years, status) ~ trans, data=bosms3, dist="exp")
tmat <- rbind(c(NA,1,2),c(NA,NA,3),c(NA,NA,NA))
sim.fmsm(bexp, M=10, t=5, trans=tmat)
```

---

simfinal_fmsm   *Simulate and summarise final outcomes from a flexible parametric multi-state model*

Description

Estimates the probability of each final outcome ("absorbing" state), and the mean and quantiles of the time to that outcome for people who experience it, by simulating a large sample of individuals from the model. This can be used for both Markov and semi-Markov models.

Usage

```
simfinal_fmsm(
  x,
  newdata = NULL,
  probs = c(0.025, 0.5, 0.975),
  t = 1000,
  M = 1e+05,
  B = 0,
  cores = NULL
)
```

Arguments

x	Object returned by [fmsm](), representing a multi-state model formed from transition-specific time-to-event models fitted by [flexsurvreg]().
newdata	Data frame of covariate values, with one column per covariate, and one row per alternative value.
probs	Quantiles to calculate, by default, c(0.025, 0.5, 0.975) for a median and 95% interval.
t	Maximum time to simulate to, passed to [sim.fmsm](), so that the summaries are taken from the subset of individuals in the simulated data who are in the absorbing state at this time.
M	Number of individuals to simulate.
B	Number of simulations to use to calculate 95% confidence intervals based on the asymptotic normal distribution of the basic parameter estimates. If B=0 then no intervals are calculated.
cores	Number of processor cores to use. If NULL (the default) then a single core is used.

Details

For a competing risks model, i.e. a model defined by just one starting state and multiple destination states representing competing events, this returns the probability governing the next event that happens, and the distribution of the time to each event conditionally on that event happening.

Value

A tidy data frame with rows for each combination of covariate values and quantity of interest. The quantity of interest is identified in the column quantity, and the value of the quantity is in val, with additional columns lower and upper giving 95% confidence intervals for the quantity, if B>0.

simfs_bytrans	*Reformat simulated multi-state data with one row per simulated transition*

Description

Reformat simulated multi-state data with one row per simulated transition

Usage

```
simfs_bytrans(simfs)
```

Arguments

simfs   Output from `sim.fmsm` representing simulated histories from a multi-state model.

Value

Data frame with four columns giving transition start state, transition end state, transition name and the time taken by the transition.

simt_flexsurvmix	*Simulate times to competing events from a mixture multi-state model*

Description

Simulate times to competing events from a mixture multi-state model

Usage

```
simt_flexsurvmix(x, newdata = NULL, n)
```

Arguments

x   Fitted model object returned from `flexsurvmix`.

newdata   Data frame or list of covariate values. If omitted for a model with covariates, a default is used, defined by all combinations of factors if the only covariates in the model are factors, or all covariate values of zero if there are any non-factor covariates in the model.

n   Number of simulations

Value

Data frame with n*m rows and a column for each competing event, where m is the number of alternative covariate values, that is the number of rows of newdata. The simulated time represents the time to that event conditionally on that event being the one that occurs. This function doesn't simulate which event occurs.

---

simulate.flexsurvreg   *Simulate censored time-to-event data from a fitted flexsurvreg model*

Description

Simulate censored time-to-event data from a fitted flexsurvreg model

Usage

```
## S3 method for class 'flexsurvreg'
simulate(
  object,
  nsim = 1,
  seed = NULL,
  newdata = NULL,
  start = NULL,
  censtime = NULL,
  tidy = FALSE,
  ...
)
```

Arguments

object	Object returned by [flexsurvreg]().
nsim	Number of simulations per row in `newdata`.
seed	Random number seed. This is returned with the result of this function, as described in [simulate]() for the `lm` method.
newdata	Data frame defining alternative sets of covariate values to simulate with. If omitted, this defaults to the data originally used to fit the model.
start	Optional left-truncation time or times. The returned survival, hazard or cumulative hazard will be conditioned on survival up to this time. Predicted times returned with `"rmst"`, `"mean"`, `"median"` or `"quantile"` will be times since time zero, not times since the `start` time.
	A vector of the same length as `t` can be supplied to allow different truncation times for each prediction time, though this doesn't make sense in the usual case where this function is used to calculate a predicted trajectory for a single individual. This is why the default `start` time was changed for version 0.4 of **flexsurv** - this was previously a vector of the start times observed in the data.
censtime	A right-censoring time, or vector of times matching the rows of `newdata`. If `NULL` (the default) then uncensored times to events are simulated.
tidy	If `TRUE` then a "tidy" or "long"-format data frame is returned, with rows defined by combinations of covariates and simulation replicates. The simulation replicate is indicated in the column named `i`.

If `FALSE`, then a data frame is returned with one row per set of covariate values, and different columns for different simulation replicates. This is the traditional format for 'simulate' methods in base R.

In either case, the simulated time and indicator for whether the time is an event time (rather than a time of right-censoring) are returned in different columns.

`...`   Other arguments (not currently used).

Value

A data frame, with format determined by whether `tidy` was specified.

Examples

```
fit <- flexsurvreg(formula = Surv(futime, fustat) ~ rx, data = ovarian, dist="weibull")
fit2 <- flexsurvspline(formula = Surv(futime, fustat) ~ rx, data = ovarian, k=3)
nd = data.frame(rx=1:2)
simulate(fit, seed=1002, newdata=nd)
simulate(fit, seed=1002, newdata=nd, start=500)
simulate(fit2, nsim=3, seed=1002, newdata=nd)
simulate(fit2, nsim=3, seed=1002, newdata=nd, start=c(500,1000))
```

---

standsurv   *Marginal survival and hazards of fitted flexsurvreg models*

Description

Returns a tidy data.frame of marginal survival probabilities, or hazards, restricted mean survival, or quantiles of the marginal survival function at user-defined time points and covariate patterns. Standardization is performed over any undefined covariates in the model. The user provides the data to standardize over. Contrasts can be calculated resulting in estimates of the average treatment effect or the average treatment effect in the treated if a treated subset of the data are supplied.

Usage

```
standsurv(
  object,
  newdata = NULL,
  at = list(list()),
  atreference = 1,
  type = "survival",
  t = NULL,
  ci = FALSE,
  se = FALSE,
  boot = FALSE,
  B = NULL,
  cl = 0.95,
  trans = "log",
```

```
    contrast = NULL,
    trans.contrast = NULL,
    seed = NULL,
    rmap,
    ratetable,
    scale.ratetable = 365.25,
    n.gauss.quad = 100,
    quantiles = 0.5,
    interval = c(1e-08, 500)
)
```

Arguments

object	Output from [flexsurvreg](#) or [flexsurvspline](#), representing a fitted survival model object.
newdata	Data frame containing covariate values to produce marginal values for. If not specified then the fitted model data.frame is used. There must be a column for every covariate in the model formula for which the user wishes to standardize over. These are in the same format as the original data, with factors as a single variable, not 0/1 contrasts. Any covariates that are to be fixed should be specified in at. There should be one row for every combination of covariates in which to standardize over. If newdata contains a variable named '(weights)' then a weighted mean will be used to create the standardized estimates. This is the default behaviour if the fitted model contains case weights, which are stored in the fitted model data.frame.
at	A list of scenarios in which specific covariates are fixed to certain values. Each element of at must itself be a list. For example, for a covariate group with levels "Good", "Medium" and "Poor", the standardized survival plots for each group averaging over all other covariates is specified using at=list(list(group="Good"), list(group="Medium"), list(group="Poor")).
atreference	The reference scenario for making contrasts. Default is 1 (i.e. the first element of at).
type	"survival" for marginal survival probabilities. In a relative survival framework this returns the marginal all-cause survival (see details).
	"hazard" for the hazard of the marginal survival probability. In a relative survival framework this returns the marginal all-cause hazard (see details).
	"rmst" for standardized restricted mean survival.
	"relsurvival" for marginal relative survival (can only be specified if a relative survival model has been fitted in flexsurv).
	"excesshazard" for marginal excess hazards (can only be specified if a relative survival model has been fitted in flexsurv).
	"quantile" for quantiles of the marginal all-cause survival distribution. The quantiles option also needs to be provided.
t	Times to calculate marginal values at.
ci	Should confidence intervals be calculated? Defaults to FALSE
se	Should standard errors be calculated? Defaults to FALSE

boot	Should bootstrapping be used to calculate standard error and confidence intervals? Defaults to FALSE, in which case the delta method is used
B	Number of bootstrap simulations from the normal asymptotic distribution of the estimates used to calculate confidence intervals or standard errors. Decrease for greater speed at the expense of accuracy. Only specify if boot = TRUE
cl	Width of symmetric confidence intervals, relative to 1.
trans	Transformation to apply when calculating standard errors via the delta method to obtain confidence intervals. The default transformation is "log". Other possible names are "none", "loglog", "logit".
contrast	Contrasts between standardized measures defined by at scenarios. Options are "difference" and "ratio". There will be n-1 new columns created where n is the number of at scenarios. Default is NULL (i.e. no contrasts are calculated).
trans.contrast	Transformation to apply when calculating standard errors for contrasts via the delta method to obtain confidence intervals. The default transformation is "none" for differences in survival, hazard, quantiles, or RMST, and "log" for ratios of survival, hazard, quantiles or RMST.
seed	The random seed to use (for bootstrapping confidence intervals)
rmap	An list that maps data set names to expected ratetable names. This must be specified if all-cause survival and hazards are required after fitting a relative survival model. This can also be specified if expected rates are required for plotting purposes. See the details section below.
ratetable	A table of expected event rates (see ratetable)
scale.ratetable	
	Transformation from the time scale of the fitted flexsurv model to the time scale in ratetable. For example, if the analysis time of the fitted model is in years and the ratetable is in units/day then we should use scale.ratetable = 365.25. This is the default as often the ratetable will be in units/day (see example).
n.gauss.quad	Number of Gaussian quadrature points used for integrating the all-cause survival function when calculating RMST in a relative survival framework (default = 100)
quantiles	If type="quantile", this specifies the quantiles of the survival time distribution to return estimates for.
interval	Interval of survival times for quantile root finding. Default is c(1e-08, 500).

Details

The syntax of standsurv follows closely that of Stata's standsurv command written by Paul Lambert and Michael Crowther. The function calculates standardized (marginal) measures including standardized survival functions, standardized restricted mean survival times, quantiles and the hazard of standardized survival. The standardized survival is defined as

$$S_s(t|X=x) = E(S(t|X=x,Z)) = \frac{1}{N}\sum_{i=1}^{N} S(t|X=x, Z=z_i)$$

The hazard of the standardized survival is a weighted average of individual hazard functions at time t, weighted by the survival function at this time:

$$h_s(t|X = x) = \frac{\sum_{i=1}^{N} S(t|X = x, Z = z_i)h(t|X = x, Z = z_i)}{\sum_{i=1}^{N} S(t|X = x, Z = z_i)}$$

Marginal expected survival and hazards can be calculated by providing a population-based lifetable of class ratetable in `ratetable` and a mapping between stratification factors in the lifetable and the user dataset using `rmap`. If these stratification factors are not in the fitted survival model then the user must specify them in `newdata` along with the covariates of the model. The marginal expected survival is calculated using the "Ederer" method that assumes no censoring as this is most relevant approach for forecasting (see `survexp`). A worked example is given below.

Marginal all-cause survival and hazards can be calculated after fitting a relative survival model, which utilise the expected survival from a population ratetable. See Rutherford et al. (Chapter 6) for further details.

Value

A `tibble` containing one row for each time-point. The column naming convention is `at{i}` for the ith scenario with corresponding confidence intervals (if specified) named `at{i}_lci` and `at{i}_uci`. Contrasts are named `contrast{k}_{j}` for the comparison of the kth versus the jth `at` scenario.

In addition tidy long-format data.frames are returned in the attributes `standsurv_at` and `standsurv_contrast`. These can be passed to `ggplot` for plotting purposes (see `plot.standsurv`).

Author(s)

Michael Sweeting <mikesweeting79@gmail.com>

References

Paul Lambert, 2021. "STANDSURV: Stata module to compute standardized (marginal) survival and related functions," Statistical Software Components S458991, Boston College Department of Economics. https://ideas.repec.org/c/boc/bocode/s458991.html

Rutherford, MJ, Lambert PC, Sweeting MJ, Pennington B, Crowther MJ, Abrams KR, Latimer NR. 2020. "NICE DSU Technical Support Document 21: Flexible Methods for Survival Analysis" https://nicedsu.sites.sheffield.ac.uk/tsds/flexible-methods-for-survival-analysis-tsd

Examples

```
## mean age is higher in those with smaller observed survival times
newbc <- bc
set.seed(1)
newbc$age <- rnorm(dim(bc)[1], mean = 65-scale(newbc$recyrs, scale=FALSE),
 sd = 5)

## Fit a Weibull flexsurv model with group and age as covariates
weib_age <- flexsurvreg(Surv(recyrs, censrec) ~ group+age, data=newbc,
                    dist="weibull")
```

```
## Calculate standardized survival and the difference in standardized survival
## for the three levels of group across a grid of survival times
standsurv_weib_age <- standsurv(weib_age,
                                        at = list(list(group="Good"),
                                                  list(group="Medium"),
                                                  list(group="Poor")),
                                        t=seq(0,7, length.out=100),
                                        contrast = "difference", ci=FALSE)
standsurv_weib_age

## Calculate hazard of standardized survival and the marginal hazard ratio
## for the three levels of group across a grid of survival times
## 10 bootstraps for confidence intervals (this should be larger)
## Not run:
haz_standsurv_weib_age <- standsurv(weib_age,
                                        at = list(list(group="Good"),
                                                  list(group="Medium"),
                                                  list(group="Poor")),
                                        t=seq(0,7, length.out=100),
                                        type="hazard",
                                        contrast = "ratio", boot = TRUE,
                                        B=10, ci=TRUE)
haz_standsurv_weib_age
plot(haz_standsurv_weib_age, ci=TRUE)
## Hazard ratio plot shows a decreasing marginal HR
## Whereas the conditional HR is constant (model is a PH model)
plot(haz_standsurv_weib_age, contrast=TRUE, ci=TRUE)

## Calculate standardized survival from a Weibull model together with expected
## survival matching to US lifetables

# age at diagnosis in days. This is required to match to US ratetable, whose
# timescale is measured in days
newbc$agedays <- floor(newbc$age * 365.25)
## Create some random diagnosis dates centred on 01/01/2010 with SD=1 year
## These will be used to match to expected rates in the lifetable
newbc$diag <- as.Date(floor(rnorm(dim(newbc)[1],
                      mean = as.Date("01/01/2010", "%d/%m/%Y"), sd=365)),
                      origin="1970-01-01")
## Create sex (assume all are female)
newbc$sex <- factor("female")
standsurv_weib_expected <- standsurv(weib_age,
                                        at = list(list(group="Good"),
                                                  list(group="Medium"),
                                                  list(group="Poor")),
                                        t=seq(0,7, length.out=100),
                                        rmap=list(sex = sex,
                                                  year = diag,
                                                  age = agedays),
                                        ratetable = survival::survexp.us,
                                        scale.ratetable = 365.25,
                                        newdata = newbc)
```

```
## Plot marginal survival with expected survival superimposed
plot(standsurv_weib_expected, expected=TRUE)

## End(Not run)
```

---

summary.flexsurvreg   *Summaries of fitted flexible survival models*

Description

Return fitted survival, cumulative hazard or hazard at a series of times from a fitted flexsurvreg
or flexsurvspline model.

Usage

```
## S3 method for class 'flexsurvreg'
summary(
  object,
  newdata = NULL,
  X = NULL,
  type = "survival",
  fn = NULL,
  t = NULL,
  quantiles = 0.5,
  start = 0,
  cross = TRUE,
  ci = TRUE,
  se = FALSE,
  B = 1000,
  cl = 0.95,
  tidy = FALSE,
  na.action = na.pass,
  ...
)
```

Arguments

object   Output from flexsurvreg or flexsurvspline, representing a fitted survival
              model object.

newdata   Data frame containing covariate values to produce fitted values for. Or a list that
              can be coerced to such a data frame. There must be a column for every covariate
              in the model formula, and one row for every combination of covariates the fitted
              values are wanted for. These are in the same format as the original data, with
              factors as a single variable, not 0/1 contrasts.

              If this is omitted, if there are any continuous covariates, then a single summary is
              provided with all covariates set to their mean values in the data - for categorical
              covariates, the means of the 0/1 indicator variables are taken. If there are only
              factor covariates in the model, then all distinct groups are used by default.

X   Alternative way of defining covariate values to produce fitted values for. Since
                           version 0.4, newdata is an easier way that doesn't require the user to create
                           factor contrasts, but X has been kept for backwards compatibility.

                           Columns of X represent different covariates, and rows represent multiple com-
                           binations of covariate values. For example matrix(c(1,2),nrow=2) if there is
                           only one covariate in the model, and we want survival for covariate values of
                           1 and 2. A vector can also be supplied if just one combination of covariates is
                           needed.

                           For "factor" (categorical) covariates, the values of the contrasts representing fac-
                           tor levels (as returned by the contrasts function) should be used. For example,
                           for a covariate agegroup specified as an unordered factor with levels 20-29,
                           30-39, 40-49,50-59, and baseline level 20-29, there are three contrasts. To
                           return summaries for groups 20-29 and 40-49, supply X = rbind(c(0,0,0),
                           c(0,1,0)), since all contrasts are zero for the baseline level, and the second
                           contrast is "turned on" for the third level 40-49.

type   "survival" for survival probabilities.

                           "cumhaz" for cumulative hazards.

                           "hazard" for hazards.

                           "rmst" for restricted mean survival.

                           "mean" for mean survival.

                           "median" for median survival (alternative to type="quantile" with quantiles=0.5).

                           "quantile" for quantiles of the survival time distribution.

                           "link" for the fitted value of the location parameter (i.e. the "linear predictor"
                           but on the natural scale of the parameter, not on the log scale)

                           Ignored if "fn" is specified.

fn   Custom function of the parameters to summarise against time. This has optional
                           first two arguments t representing time, and start representing left-truncation
                           points, and any remaining arguments must be parameters of the distribution. It
                           should be vectorised, and return a vector corresponding to the vectors given by
                           t, start and the parameter vectors.

t   Times to calculate fitted values for. By default, these are the sorted unique
                           observation (including censoring) times in the data - for left-truncated datasets
                           these are the "stop" times.

quantiles   If type="quantile", this specifies the quantiles of the survival time distribution
                           to return estimates for.

start   Optional left-truncation time or times. The returned survival, hazard or cumu-
                           lative hazard will be conditioned on survival up to this time. Predicted times
                           returned with "rmst", "mean", "median" or "quantile" will be times since
                           time zero, not times since the start time.

                           A vector of the same length as t can be supplied to allow different truncation
                           times for each prediction time, though this doesn't make sense in the usual case
                           where this function is used to calculate a predicted trajectory for a single in-
                           dividual. This is why the default start time was changed for version 0.4 of
                           flexsurv - this was previously a vector of the start times observed in the data.

cross   If TRUE (the default) then summaries are calculated for all combinations of times specified in t and covariate vectors specifed in newdata.

If FALSE, then the times t should be of length equal to the number of rows in newdata, and one summary is produced for each row of newdata paired with the corresponding element of t. This is used, e.g. when determining Cox-Snell residuals.

ci   Set to FALSE to omit confidence intervals.

se   Set to TRUE to include standard errors.

B   Number of simulations from the normal asymptotic distribution of the estimates used to calculate confidence intervals or standard errors. Decrease for greater speed at the expense of accuracy, or set B=0 to turn off calculation of CIs and SEs.

cl   Width of symmetric confidence intervals, relative to 1.

tidy   If TRUE, then the results are returned as a tidy data frame instead of a list. This can help with using the **ggplot2** package to compare summaries for different covariate values.

na.action   Function determining what should be done with missing values in newdata. If na.pass (the default) then summaries of NA are produced for missing covariate values. If na.omit, then missing values are dropped, the behaviour of summary.flexsurvreg before flexsurv version 1.2.

...   Further arguments passed to or from other methods. Currently unused.

Details

Time-dependent covariates are not currently supported. The covariate values are assumed to be constant through time for each fitted curve.

Value

If tidy=FALSE, a list with one component for each unique covariate value (if there are only categorical covariates) or one component (if there are no covariates or any continuous covariates). Each of these components is a matrix with one row for each time in t, giving the estimated survival (or cumulative hazard, or hazard) and 95% confidence limits. These list components are named with the covariate names and values which define them.

If tidy=TRUE, a data frame is returned instead. This is formed by stacking the above list components, with additional columns to identify the covariate values that each block corresponds to.

If there are multiple summaries, an additional list component named X contains a matrix with the exact values of contrasts (dummy covariates) defining each summary.

The [plot.flexsurvreg](#) function can be used to quickly plot these model-based summaries against empirical summaries such as Kaplan-Meier curves, to diagnose model fit.

Confidence intervals are obtained by sampling randomly from the asymptotic normal distribution of the maximum likelihood estimates and then taking quantiles (see, e.g. Mandel (2013)).

Author(s)

C. H. Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

References

Mandel, M. (2013). "Simulation based confidence intervals for functions with complicated derivatives." The American Statistician (in press).

See Also

[flexsurvreg](), [flexsurvspline]().

---

summary.flexsurvrtrunc

Summarise quantities of interest from fitted flexsurvrtrunc models

---

Description

This function extracts quantities of interest from the untruncated version of a model with individual-specific right truncation points fitted by [flexsurvrtrunc](). Note that covariates are currently not supported by [flexsurvrtrunc]().

Usage

```
## S3 method for class 'flexsurvrtrunc'
summary(
  object,
  type = "survival",
  fn = NULL,
  t = NULL,
  quantiles = 0.5,
  ci = TRUE,
  se = FALSE,
  B = 1000,
  cl = 0.95,
  ...
)
```

Arguments

object	Output from [flexsurvreg]() or [flexsurvspline](), representing a fitted survival model object.
type	`"survival"` for survival probabilities.
	`"cumhaz"` for cumulative hazards.
	`"hazard"` for hazards.
	`"rmst"` for restricted mean survival.
	`"mean"` for mean survival.
	`"median"` for median survival (alternative to `type="quantile"` with `quantiles=0.5`).
	`"quantile"` for quantiles of the survival time distribution.
	Ignored if `"fn"` is specified.

fn	Custom function of the parameters to summarise against time. This has optional first argument t representing time, and any remaining arguments must be parameters of the distribution. It should return a vector of the same length as t.
t	Times to calculate fitted values for. By default, these are the sorted unique observation (including censoring) times in the data - for left-truncated datasets these are the "stop" times.
quantiles	If type="quantile", this specifies the quantiles of the survival time distribution to return estimates for.
ci	Set to FALSE to omit confidence intervals.
se	Set to TRUE to include standard errors.
B	Number of simulations from the normal asymptotic distribution of the estimates used to calculate confidence intervals or standard errors. Decrease for greater speed at the expense of accuracy, or set B=0 to turn off calculation of CIs and SEs.
cl	Width of symmetric confidence intervals, relative to 1.
...	Further arguments passed to or from other methods. Currently unused.

---

survrtrunc	*Nonparametric estimator of survival from right-truncated, uncensored data*

Description

Estimates the survivor function from right-truncated, uncensored data by reversing time, interpreting the data as left-truncated, applying the Kaplan-Meier / Lynden-Bell estimator and transforming back.

Usage

```
survrtrunc(t, rtrunc, tmax, data = NULL, eps = 0.001, conf.int = 0.95)
```

Arguments

t	Vector of observed times from an initial event to a final event.
rtrunc	Individual-specific right truncation points, so that each individual's survival time t would not have been observed if it was greater than the corresponding element of rtrunc. If any of these are greater than tmax, then the actual individual-level truncation point for these individuals is taken to be tmax.
tmax	Maximum possible time to event that could have been observed.
data	Data frame to find t and rtrunc in. If not supplied, these should be in the working environment.
eps	Small number that is added to t before implementing the time-reversed estimator, to ensure the risk set is consistent between forward and reverse time scales. It should be just large enough that t+eps is not ==t. This should not need changing from the default of 0.001, unless t are extremely large or small and the data are rounded to integer.
conf.int	Confidence level, defaulting to 0.95.

Details

Note that this does not estimate the untruncated survivor function - instead it estimates the survivor function truncated above at a time defined by the maximum possible time that might have been observed in the data.

Define X as the time of the initial event, Y as the time of the final event, then we wish to determine the distribution of $T = Y - X$.

Observations are only recorded if $Y \leq t_{max}$. Then the distribution of T in the resulting sample is right-truncated by $\texttt{rtrunc} = t_{max} - X$.

Equivalently, the distribution of $t_{max} - T$ is left-truncated, since it is only observed if $t_{max} - T \geq X$. Then the standard Kaplan-Meier type estimator as implemented in survfit is used (as described by Lynden-Bell, 1971) and the results transformed back.

This situation might happen in a disease epidemic, where X is the date of disease onset for an individual, Y is the date of death, and we wish to estimate the distribution of the time T from onset to death, given we have only observed people who have died by the date t_{max}.

If the estimated survival is unstable at the highest times, then consider replacing tmax by a slightly lower value, then if necessary, removing individuals with t > tmax, so that the estimand is changed to the survivor function truncated over a slightly narrower interval.

Value

A list with components:

time Time points where the estimated survival changes.

surv Estimated survival at time, truncated above at tmax.

se.surv Standard error of survival.

std.err Standard error of -log(survival). Named this way for consistency with survfit.

lower Lower confidence limits for survival.

upper Upper confidence limits for survival.

References

D. Lynden-Bell (1971) A method of allowing for known observational selection in small samples applied to 3CR quasars. Monthly Notices of the Royal Astronomical Society, 155:95–118.

Seaman, S., Presanis, A. and Jackson, C. (2020) Review of methods for estimating distribution of time to event from right-truncated data.

Examples

```
## simulate some event time data
set.seed(1)
X <- rweibull(100, 2, 10)
T <- rweibull(100, 2, 10)

## truncate above
tmax <- 20
obs <- X + T < tmax
```

```
rtrunc <- tmax - X
dat <- data.frame(X, T, rtrunc)[obs,]
sf <-    survrtrunc(T, rtrunc, data=dat, tmax=tmax)
plot(sf, conf.int=TRUE)
## Kaplan-Meier estimate ignoring truncation is biased
sfnaive <- survfit(Surv(T) ~ 1, data=dat)
lines(sfnaive, conf.int=TRUE, lty=2, col="red")

## truncate above the maximum observed time
tmax <- max(X + T) + 10
obs <- X + T < tmax
rtrunc <- tmax - X
dat <- data.frame(X, T, rtrunc)[obs,]
sf <-    survrtrunc(T, rtrunc, data=dat, tmax=tmax)
plot(sf, conf.int=TRUE)
## estimates identical to the standard Kaplan-Meier
sfnaive <- survfit(Surv(T) ~ 1, data=dat)
lines(sfnaive, conf.int=TRUE, lty=2, col="red")
```

Survspline   *Royston/Parmar spline survival distribution*

Description

Probability density, distribution, quantile, random generation, hazard, cumulative hazard, mean and restricted mean functions for the Royston/Parmar spline model. These functions have all parameters of the distribution collected together in a single argument gamma. For the equivalent functions with one argument per parameter, see Survsplinek.

Usage

```
dsurvspline(
  x,
  gamma,
  beta = 0,
  X = 0,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  offset = 0,
  log = FALSE
)

psurvspline(
  q,
  gamma,
```

```
  beta = 0,
  X = 0,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  offset = 0,
  lower.tail = TRUE,
  log.p = FALSE
)

qsurvspline(
  p,
  gamma,
  beta = 0,
  X = 0,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  offset = 0,
  lower.tail = TRUE,
  log.p = FALSE
)

rsurvspline(
  n,
  gamma,
  beta = 0,
  X = 0,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  offset = 0
)

Hsurvspline(
  x,
  gamma,
  beta = 0,
  X = 0,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  offset = 0
)
```

```
hsurvspline(
  x,
  gamma,
  beta = 0,
  X = 0,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  offset = 0
)

rmst_survspline(
  t,
  gamma,
  beta = 0,
  X = 0,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  offset = 0,
  start = 0
)

mean_survspline(
  gamma,
  beta = 0,
  X = 0,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  offset = 0
)
```

Arguments

x, q, t	Vector of times.
gamma	Parameters describing the baseline spline function, as described in [flexsurvspline](). This may be supplied as a vector with number of elements equal to the length of knots, in which case the parameters are common to all times. Alternatively a matrix may be supplied, with rows corresponding to different times, and columns corresponding to knots.
beta	Vector of covariate effects. Not supported and ignored since version 2.3, and this argument will be removed in 2.4.

X	Matrix of covariate values. Not supported and ignored since version 2.3, and this argument will be removed in 2.4.
knots	Locations of knots on the axis of log time, supplied in increasing order. Unlike in flexsurvspline, these include the two boundary knots. If there are no additional knots, the boundary locations are not used. If there are one or more additional knots, the boundary knots should be at or beyond the minimum and maximum values of the log times. In flexsurvspline these are exactly at the minimum and maximum values.
	This may in principle be supplied as a matrix, in the same way as for gamma, but in most applications the knots will be fixed.
scale	"hazard", "odds", or "normal", as described in flexsurvspline. With the default of no knots in addition to the boundaries, this model reduces to the Weibull, log-logistic and log-normal respectively. The scale must be common to all times.
timescale	"log" or "identity" as described in flexsurvspline.
spline	"rp" to use the natural cubic spline basis described in Royston and Parmar. "splines2ns" to use the alternative natural cubic spline basis from the splines2 package (Wang and Yan 2021), which may be better behaved due to the basis being orthogonal.
offset	An extra constant to add to the linear predictor η. Not supported and ignored since version 2.3, and this argument will be removed in 2.4.
log, log.p	Return log density or probability.
lower.tail	logical; if TRUE (default), probabilities are $P(X \leq x)$, otherwise, $P(X > x)$.
p	Vector of probabilities.
n	Number of random numbers to simulate.
start	Optional left-truncation time or times. The returned restricted mean survival will be conditioned on survival up to this time.

Value

dsurvspline gives the density, psurvspline gives the distribution function, hsurvspline gives the hazard and Hsurvspline gives the cumulative hazard, as described in flexsurvspline.

qsurvspline gives the quantile function, which is computed by crude numerical inversion (using qgeneric).

rsurvspline generates random survival times by using qsurvspline on a sample of uniform random numbers. Due to the numerical root-finding involved in qsurvspline, it is slow compared to typical random number generation functions.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

References

Royston, P. and Parmar, M. (2002). Flexible parametric proportional-hazards and proportional-odds models for censored survival data, with application to prognostic modelling and estimation of treatment effects. Statistics in Medicine 21(1):2175-2197.

Wang W, Yan J (2021). Shape-Restricted Regression Splines with R Package splines2. Journal of Data Science, 19(3), 498-517.

See Also

flexsurvspline.

Examples

```
## reduces to the weibull
regscale <- 0.786; cf <- 1.82
a <- 1/regscale; b <- exp(cf)
dweibull(1, shape=a, scale=b)
dsurvspline(1, gamma=c(log(1 / b^a), a)) # should be the same

## reduces to the log-normal
meanlog <- 1.52; sdlog <- 1.11
dlnorm(1, meanlog, sdlog)
dsurvspline(1, gamma = c(-meanlog/sdlog, 1/sdlog), scale="normal")
# should be the same
```

Survsplinek	*Royston/Parmar spline survival distribution functions with one argument per parameter*

Description

Probability density, distribution, quantile, random generation, hazard, cumulative hazard, mean and restricted mean functions for the Royston/Parmar spline model, with one argument per parameter. For the equivalent functions with all parameters collected together in a single argument, see Survspline.

Usage

```
mean_survspline0(
  gamma0,
  gamma1,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

mean_survspline1(
  gamma0,
  gamma1,
  gamma2,
  knots = c(-10, 10),
```

```
    scale = "hazard",
    timescale = "log",
    spline = "rp"
)

mean_survspline2(
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

mean_survspline3(
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

mean_survspline4(
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

mean_survspline5(
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
```

```
  gamma6,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

mean_survspline6(
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  gamma6,
  gamma7,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

mean_survspline7(
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  gamma6,
  gamma7,
  gamma8,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

rmst_survspline0(
  t,
  gamma0,
  gamma1,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  start = 0
)
```

```
rmst_survspline1(
  t,
  gamma0,
  gamma1,
  gamma2,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  start = 0
)

rmst_survspline2(
  t,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  start = 0
)

rmst_survspline3(
  t,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  knots = c(-10, 10),
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  start = 0
)

rmst_survspline4(
  t,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
```

```
    knots = c(-10, 10),
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    start = 0
)

rmst_survspline5(
    t,
    gamma0,
    gamma1,
    gamma2,
    gamma3,
    gamma4,
    gamma5,
    gamma6,
    knots = c(-10, 10),
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    start = 0
)

rmst_survspline6(
    t,
    gamma0,
    gamma1,
    gamma2,
    gamma3,
    gamma4,
    gamma5,
    gamma6,
    gamma7,
    knots = c(-10, 10),
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    start = 0
)

rmst_survspline7(
    t,
    gamma0,
    gamma1,
    gamma2,
    gamma3,
    gamma4,
    gamma5,
```

```
    gamma6,
    gamma7,
    gamma8,
    knots = c(-10, 10),
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    start = 0
)

dsurvspline0(
    x,
    gamma0,
    gamma1,
    knots,
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    log = FALSE
)

dsurvspline1(
    x,
    gamma0,
    gamma1,
    gamma2,
    knots,
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    log = FALSE
)

dsurvspline2(
    x,
    gamma0,
    gamma1,
    gamma2,
    gamma3,
    knots,
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    log = FALSE
)

dsurvspline3(
    x,
```

```
      gamma0,
      gamma1,
      gamma2,
      gamma3,
      gamma4,
      knots,
      scale = "hazard",
      timescale = "log",
      spline = "rp",
      log = FALSE
    )

    dsurvspline4(
      x,
      gamma0,
      gamma1,
      gamma2,
      gamma3,
      gamma4,
      gamma5,
      knots,
      scale = "hazard",
      timescale = "log",
      spline = "rp",
      log = FALSE
    )

    dsurvspline5(
      x,
      gamma0,
      gamma1,
      gamma2,
      gamma3,
      gamma4,
      gamma5,
      gamma6,
      knots,
      scale = "hazard",
      timescale = "log",
      spline = "rp",
      log = FALSE
    )

    dsurvspline6(
      x,
      gamma0,
      gamma1,
      gamma2,
```

```
    gamma3,
    gamma4,
    gamma5,
    gamma6,
    gamma7,
    knots,
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    log = FALSE
)

dsurvspline7(
    x,
    gamma0,
    gamma1,
    gamma2,
    gamma3,
    gamma4,
    gamma5,
    gamma6,
    gamma7,
    gamma8,
    knots,
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    log = FALSE
)

psurvspline0(
    q,
    gamma0,
    gamma1,
    knots,
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    lower.tail = TRUE,
    log.p = FALSE
)

psurvspline1(
    q,
    gamma0,
    gamma1,
    gamma2,
    knots,
```

```
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    lower.tail = TRUE,
    log.p = FALSE
)

psurvspline2(
    q,
    gamma0,
    gamma1,
    gamma2,
    gamma3,
    knots,
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    lower.tail = TRUE,
    log.p = FALSE
)

psurvspline3(
    q,
    gamma0,
    gamma1,
    gamma2,
    gamma3,
    gamma4,
    knots,
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    lower.tail = TRUE,
    log.p = FALSE
)

psurvspline4(
    q,
    gamma0,
    gamma1,
    gamma2,
    gamma3,
    gamma4,
    gamma5,
    knots,
    scale = "hazard",
    timescale = "log",
    spline = "rp",
```

```
    lower.tail = TRUE,
    log.p = FALSE
  )

  psurvspline5(
    q,
    gamma0,
    gamma1,
    gamma2,
    gamma3,
    gamma4,
    gamma5,
    gamma6,
    knots,
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    lower.tail = TRUE,
    log.p = FALSE
  )

  psurvspline6(
    q,
    gamma0,
    gamma1,
    gamma2,
    gamma3,
    gamma4,
    gamma5,
    gamma6,
    gamma7,
    knots,
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    lower.tail = TRUE,
    log.p = FALSE
  )

  psurvspline7(
    q,
    gamma0,
    gamma1,
    gamma2,
    gamma3,
    gamma4,
    gamma5,
    gamma6,
```

```
    gamma7,
    gamma8,
    knots,
    scale = "hazard",
    timescale = "log",
    spline = "rp",
    lower.tail = TRUE,
    log.p = FALSE
)

qsurvspline0(
  p,
  gamma0,
  gamma1,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  lower.tail = TRUE,
  log.p = FALSE
)

qsurvspline1(
  p,
  gamma0,
  gamma1,
  gamma2,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  lower.tail = TRUE,
  log.p = FALSE
)

qsurvspline2(
  p,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  lower.tail = TRUE,
  log.p = FALSE
)
```

```
qsurvspline3(
  p,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  lower.tail = TRUE,
  log.p = FALSE
)

qsurvspline4(
  p,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  lower.tail = TRUE,
  log.p = FALSE
)

qsurvspline5(
  p,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  gamma6,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  lower.tail = TRUE,
  log.p = FALSE
)
```

```
qsurvspline6(
  p,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  gamma6,
  gamma7,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  lower.tail = TRUE,
  log.p = FALSE
)

qsurvspline7(
  p,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  gamma6,
  gamma7,
  gamma8,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp",
  lower.tail = TRUE,
  log.p = FALSE
)

rsurvspline0(
  n,
  gamma0,
  gamma1,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)
```

```
rsurvspline1(
  n,
  gamma0,
  gamma1,
  gamma2,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

rsurvspline2(
  n,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

rsurvspline3(
  n,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

rsurvspline4(
  n,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
```

```
)

rsurvspline5(
  n,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  gamma6,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

rsurvspline6(
  n,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  gamma6,
  gamma7,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

rsurvspline7(
  n,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  gamma6,
  gamma7,
  gamma8,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
```

```
)

hsurvspline0(
  x,
  gamma0,
  gamma1,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

hsurvspline1(
  x,
  gamma0,
  gamma1,
  gamma2,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

hsurvspline2(
  x,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

hsurvspline3(
  x,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)
```

```
hsurvspline4(
  x,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

hsurvspline5(
  x,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  gamma6,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

hsurvspline6(
  x,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  gamma6,
  gamma7,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

hsurvspline7(
  x,
  gamma0,
```

```
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  gamma6,
  gamma7,
  gamma8,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

Hsurvspline0(
  x,
  gamma0,
  gamma1,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

Hsurvspline1(
  x,
  gamma0,
  gamma1,
  gamma2,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

Hsurvspline2(
  x,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

Hsurvspline3(
```

```
  x,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

Hsurvspline4(
  x,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

Hsurvspline5(
  x,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  gamma6,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

Hsurvspline6(
  x,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
```

```
  gamma5,
  gamma6,
  gamma7,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)

Hsurvspline7(
  x,
  gamma0,
  gamma1,
  gamma2,
  gamma3,
  gamma4,
  gamma5,
  gamma6,
  gamma7,
  gamma8,
  knots,
  scale = "hazard",
  timescale = "log",
  spline = "rp"
)
```

Arguments

gamma0, gamma1, gamma2, gamma3, gamma4, gamma5, gamma6, gamma7, gamma8

Parameters describing the baseline spline function, as described in [flexsurvspline](#).

knots   Locations of knots on the axis of log time, supplied in increasing order. Un-
                 like in [flexsurvspline](#), these include the two boundary knots. If there are no
                 additional knots, the boundary locations are not used. If there are one or more
                 additional knots, the boundary knots should be at or beyond the minimum and
                 maximum values of the log times. In [flexsurvspline](#) these are exactly at the
                 minimum and maximum values.

                 This may in principle be supplied as a matrix, in the same way as for gamma, but
                 in most applications the knots will be fixed.

scale   "hazard", "odds", or "normal", as described in [flexsurvspline](#). With the de-
                 fault of no knots in addition to the boundaries, this model reduces to the Weibull,
                 log-logistic and log-normal respectively. The scale must be common to all times.

timescale   "log" or "identity" as described in [flexsurvspline](#).

spline   "rp" to use the natural cubic spline basis described in Royston and Parmar.
                 "splines2ns" to use the alternative natural cubic spline basis from the splines2
                 package (Wang and Yan 2021), which may be better behaved due to the basis
                 being orthogonal.

start   Optional left-truncation time or times. The returned restricted mean survival
                 will be conditioned on survival up to this time.
```

| x, q, t | Vector of times. |
|---|---|
| log, log.p | Return log density or probability. |
| lower.tail | logical; if TRUE (default), probabilities are $P(X \leq x)$, otherwise, $P(X > x)$. |
| p | Vector of probabilities. |
| n | Number of random numbers to simulate. |

### Details

These functions go up to 7 spline knots, or 9 parameters.

### Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

---

| tidy.flexsurvreg | *Tidy a flexsurv model object* |
|---|---|

---

### Description

Tidy summarizes information about the components of the model into a tidy data frame.

### Usage

```
## S3 method for class 'flexsurvreg'
tidy(
  x,
  conf.int = FALSE,
  conf.level = 0.95,
  pars = "all",
  transform = "none",
  ...
)
```

Arguments

x	Output from [flexsurvreg](#) or [flexsurvspline](#), representing a fitted survival model object.
conf.int	Logical. Should confidence intervals be returned? Default is FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Default is 0.95.
pars	One of "all", "baseline", or "coefs" for all parameters, baseline distribution parameters, or covariate effects (i.e. regression betas), respectively. Default is "all".
transform	Character vector of transformations to apply to requested pars. Default is "none", which returns pars as-is.
	Users can specify one or both types of transformations:

- • ″baseline.real″ which transforms the baseline distribution parameters to the real number line used for estimation.
- • ″coefs.exp″ which exponentiates the covariate effects.

See Details for a more complete explanation.

...   Not currently used.

Details

flexsurvreg models estimate two types of coefficients, baseline distribution parameters, and co-variate effects which act on the baseline distribution. By design, flexsurvreg returns distribution parameters on the same scale as is found in the relevant d/p/q/r functions. Covariate effects are returned on the log-scale, which represents either log-time ratios (accelerated failure time mod-els) or log-hazard ratios for proportional hazard models. By default, tidy() will return baseline distribution parameters on their natural scale and covariate effects on the log-scale.

To transform the baseline distribution parameters to the real-value number line (the scale used for estimation), pass the character argument ″baseline.real″ to transform. To get time ratios or hazard ratios, pass ″coefs.exp″ to transform. These transformations may be done together by submitting both arguments as a character vector.

Value

A tibble containing the columns: term, estimate, std.error, statistic, p.value, conf.low, and conf.high, by default.

statistic and p.value are only provided for covariate effects (NA for baseline distribution pa-rameters). These are computed as Wald-type test statistics with p-values from a standard normal distribution.

Examples

```
fitg <- flexsurvreg(formula = Surv(futime, fustat) ~ age, data = ovarian, dist = ″gengamma″)
tidy(fitg)
tidy(fitg, pars = ″coefs″, transform = ″coefs.exp″)
```

---

tidy.standsurv   *Tidy a standsurv object.*

Description

This function is used internally by standsurv and tidy data.frames are automatically returned by the function.

Usage

```
## S3 method for class 'standsurv'
tidy(x, ...)
```

Arguments

x	A standsurv object.
...	Not currently used.

Value

Returns additional tidy data.frames (tibbles) stored as attributes named standpred_at and stand-pred_contrast.

---

totlos.fs	*Total length of stay in particular states for a fully-parametric, time-inhomogeneous Markov multi-state model*

Description

The matrix whose r, s entry is the expected amount of time spent in state s for a time-inhomogeneous, continuous-time Markov multi-state process that starts in state r, up to a maximum time t. This is defined as the integral of the corresponding transition probability up to that time.

Usage

```
totlos.fs(
  x,
  trans = NULL,
  t = 1,
  newdata = NULL,
  ci = FALSE,
  tvar = "trans",
  sing.inf = 1e+10,
  B = 1000,
  cl = 0.95,
  ...
)
```

Arguments

x	A model fitted with `flexsurvreg`. See `msfit.flexsurvreg` for the required form of the model and the data. Additionally, this must be a Markov / clock-forward model, but can be time-inhomogeneous. See the package vignette for further explanation.
	x can also be a list of models, with one component for each permitted transition, as illustrated in `msfit.flexsurvreg`.
trans	Matrix indicating allowed transitions. See `msfit.flexsurvreg`.
t	Time or vector of times to predict up to. Must be finite.
newdata	A data frame specifying the values of covariates in the fitted model, other than the transition number. See `msfit.flexsurvreg`.

ci   Return a confidence interval calculated by simulating from the asymptotic nor-
                mal distribution of the maximum likelihood estimates. Turned off by default,
                since this is computationally intensive. If turned on, users should increase B
                until the results reach the desired precision.

tvar   Variable in the data representing the transition type. Not required if x is a list of
                models.

sing.inf   If there is a singularity in the observed hazard, for example a Weibull distribution
                with shape < 1 has infinite hazard at t=0, then as a workaround, the hazard
                is assumed to be a large finite number, sing.inf, at this time. The results
                should not be sensitive to the exact value assumed, but users should make sure
                by adjusting this parameter in these cases.

B   Number of simulations from the normal asymptotic distribution used to calculate
                variances. Decrease for greater speed at the expense of accuracy.

cl   Width of symmetric confidence intervals, relative to 1.

...   Arguments passed to ode in **deSolve**.

Details

This is computed by solving a second order extension of the Kolmogorov forward differential equa-
tion numerically, using the methods in the **deSolve** package. The equation is expressed as a linear
system

$$\frac{dT(t)}{dt} = P(t)$$

$$\frac{dP(t)}{dt} = P(t)Q(t)$$

and solved for $T(t)$ and $P(t)$ simultaneously, where $T(t)$ is the matrix of total lengths of stay, $P(t)$
is the transition probability matrix for time t, and $Q(t)$ is the transition hazard or intensity as a
function of t. The initial conditions are $T(0) = 0$ and $P(0) = I$.

Note that the package **msm** has a similar method totlos.msm. totlos.fs should give the same
results as totlos.msm when both of these conditions hold:

  • the time-to-event distribution is exponential for all transitions, thus the flexsurvreg model
    was fitted with dist="exp", and is time-homogeneous.

  • the **msm** model was fitted with exacttimes=TRUE, thus all the event times are known, and
    there are no time-dependent covariates.

msm only allows exponential or piecewise-exponential time-to-event distributions, while **flexsurvreg**
allows more flexible models. **msm** however was designed in particular for panel data, where the
process is observed only at arbitrary times, thus the times of transition are unknown, which makes
flexible models difficult.

This function is only valid for Markov ("clock-forward") multi-state models, though no warning
or error is currently given if the model is not Markov. See totlos.simfs for the equivalent for
semi-Markov ("clock-reset") models.

Value

The matrix of lengths of stay $T(t)$, if t is of length 1, or a list of matrices if t is longer.

If ci=TRUE, each element has attributes "lower" and "upper" giving matrices of the corresponding confidence limits. These are formatted for printing but may be extracted using attr().

The result also has an attribute P giving the transition probability matrices, since these are unavoidably computed as a side effect. These are suppressed for printing, but can be extracted with attr(...,"P").

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>.

See Also

totlos.simfs, pmatrix.fs, msfit.flexsurvreg.

Examples

```
# BOS example in vignette, and in msfit.flexsurvreg
bexp <- flexsurvreg(Surv(Tstart, Tstop, status) ~ trans,
                    data=bosms3, dist="exp")
tmat <- rbind(c(NA,1,2),c(NA,NA,3),c(NA,NA,NA))

# predict 4 years spent without BOS, 3 years with BOS, before death
# As t increases, this should converge

totlos.fs(bexp, t=10, trans=tmat)
totlos.fs(bexp, t=1000, trans=tmat)
totlos.fs(bexp, t=c(5,10), trans=tmat)

# Answers should match results in help(totlos.simfs) up to Monte Carlo
# error there / ODE solving precision here, since with an exponential
# distribution, the "semi-Markov" model there is the same as the Markov
# model here
```

---

totlos.simfs	*Expected total length of stay in specific states, from a fully-parametric, semi-Markov multi-state model*

Description

The expected total time spent in each state for semi-Markov multi-state models fitted to time-to-event data with flexsurvreg. This is defined by the integral of the transition probability matrix, though this is not analytically possible and is computed by simulation.

Usage

```
totlos.simfs(
  x,
  trans,
  t = 1,
  start = 1,
  newdata = NULL,
  ci = FALSE,
  tvar = "trans",
  tcovs = NULL,
  group = NULL,
  M = 1e+05,
  B = 1000,
  cl = 0.95,
  cores = NULL
)
```

Arguments

x	A model fitted with [flexsurvreg](). See [msfit.flexsurvreg]() for the required form of the model and the data. Additionally this should be semi-Markov, so that the time variable represents the time since the last transition. In other words the response should be of the form Surv(time,status). See the package vignette for further explanation.
	x can also be a list of [flexsurvreg]() models, with one component for each permitted transition, as illustrated in [msfit.flexsurvreg](). This can be constructed by [fmsm]().
trans	Matrix indicating allowed transitions. See [msfit.flexsurvreg](). This is not required if x is a list constructed by [fmsm]().
t	Maximum time to predict to.
start	Starting state.
newdata	A data frame specifying the values of covariates in the fitted model, other than the transition number. See [msfit.flexsurvreg]().
ci	Return a confidence interval calculated by simulating from the asymptotic normal distribution of the maximum likelihood estimates. This is turned off by default, since two levels of simulation are required. If turned on, users should adjust B and/or M until the results reach the desired precision. The simulation over M is generally vectorised, therefore increasing B is usually more expensive than increasing M.
tvar	Variable in the data representing the transition type. Not required if x is a list of models.
tcovs	Predictable time-dependent covariates such as age, see [sim.fmsm]().
group	Optional grouping for the states. For example, if there are four states, and group=c(1,1,2,2), then [totlos.simfs]() returns the expected total time in states 1 and 2 combined, and states 3 and 4 combined.

M	Number of individuals to simulate in order to approximate the transition probabilities. Users should adjust this to obtain the required precision.
B	Number of simulations from the normal asymptotic distribution used to calculate confidence limits. Decrease for greater speed at the expense of accuracy.
cl	Width of symmetric confidence intervals, relative to 1.
cores	Number of processor cores used when calculating confidence limits by repeated simulation. The default uses single-core processing.

Details

This is computed by simulating a large number of individuals M using the maximum likelihood estimates of the fitted model and the function sim.fmsm. Therefore this requires a random sampling function for the parametric survival model to be available: see the "Details" section of sim.fmsm. This will be available for all built-in distributions, though users may need to write this for custom models.

Note the random sampling method for flexsurvspline models is currently very inefficient, so that looping over M will be very slow.

The equivalent function for time-inhomogeneous Markov models is totlos.fs. Note neither of these functions give errors or warnings if used with the wrong type of model, but the results will be invalid.

Value

The expected total time spent in each state (or group of states given by group) up to time t, and corresponding confidence intervals if requested.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>.

See Also

pmatrix.simfs, sim.fmsm, msfit.flexsurvreg.

Examples

```
# BOS example in vignette, and in msfit.flexsurvreg
bexp <- flexsurvreg(Surv(years, status) ~ trans, data=bosms3, dist="exp")
tmat <- rbind(c(NA,1,2),c(NA,NA,3),c(NA,NA,NA))

# predict 4 years spent without BOS, 3 years with BOS, before death
# As t increases, this should converge
totlos.simfs(bexp, t=10, trans=tmat)
totlos.simfs(bexp, t=1000, trans=tmat)
```

---

unroll.function   *Convert a function with matrix arguments to a function with vector*
                         arguments.

Description

Given a function with matrix arguments, construct an equivalent function which takes vector arguments defined by the columns of the matrix. The new function simply uses cbind on the vector arguments to make a matrix, and calls the old one.

Usage

```
unroll.function(mat.fn, ...)
```

Arguments

mat.fn   A function with any number of arguments, some of which are matrices.

...   A series of other arguments. Their names define which arguments of mat.fn are matrices. Their values define a vector of strings to be appended to the names of the arguments in the new function. For example

                ```
                fn <- unroll.function(oldfn, gamma=1:3, alpha=0:1)
                ```

                will make a new function fn with arguments gamma1,gamma2,gamma3,alpha0,alpha1.

                Calling

                ```
                fn(gamma1=a,gamma2=b,gamma3=c,alpha0=d,alpha1=e)
                ```

                should give the same answer as

                ```
                oldfn(gamma=cbind(a,b,c),alpha=cbind(d,e))
                ```

Value

The new function, with vector arguments.

Usage in flexsurv

This is used by [flexsurvspline](#) to allow spline models, which have an arbitrary number of parameters, to be fitted using [flexsurvreg](#).

The "custom distributions" facility of [flexsurvreg](#) expects the user-supplied probability density and distribution functions to have one explicitly named argument for each scalar parameter, and given R vectorisation, each of those arguments could be supplied as a vector of alternative parameter values.

However, spline models have a varying number of scalar parameters, determined by the number of knots in the spline. [dsurvspline](#) and [psurvspline](#) have an argument called gamma. This can be supplied as a matrix, with number of columns n determined by the number of knots (plus 2), and rows referring to alternative parameter values. The following statements are used in the source of flexsurvspline:

```
 dfn <-
unroll.function(dsurvspline, gamma=0:(nk-1)) pfn <-
unroll.function(psurvspline, gamma=0:(nk-1))
```

to convert these into functions with arguments gamma0, gamma1,...,gamman, corresponding to the columns of gamma, where n = nk-1, and with other arguments in the same format.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

See Also

[flexsurvspline,flexsurvreg](#)

Examples

```
fn <- unroll.function(ncol, x=1:3)
fn(1:3, 1:3, 1:3) # equivalent to...
ncol(cbind(1:3,1:3,1:3))
```

---

vcov.flexsurvreg   *Variance-covariance matrix from a flexsurvreg model*

Description

Variance-covariance matrix from a flexsurvreg model

Usage

```
## S3 method for class 'flexsurvreg'
vcov(object, ...)
```

Arguments

object   A fitted model object of class [flexsurvreg](#), e.g. as returned by flexsurvreg
              or flexsurvspline.

...   Other arguments (currently unused).

Value

Variance-covariance matrix of the estimated parameters, on the scale that they were estimated on (for positive parameters this is the log scale).

WeibullPH   *Weibull distribution in proportional hazards parameterisation*

Description

Density, distribution function, hazards, quantile function and random generation for the Weibull distribution in its proportional hazards parameterisation.

Usage

```
dweibullPH(x, shape, scale = 1, log = FALSE)

pweibullPH(q, shape, scale = 1, lower.tail = TRUE, log.p = FALSE)

qweibullPH(p, shape, scale = 1, lower.tail = TRUE, log.p = FALSE)

hweibullPH(x, shape, scale = 1, log = FALSE)

HweibullPH(x, shape, scale = 1, log = FALSE)

rweibullPH(n, shape, scale = 1)
```

Arguments

x, q	Vector of quantiles.
shape	Vector of shape parameters.
scale	Vector of scale parameters.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are $P(X \leq x)$, otherwise, $P(X > x)$.
p	Vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required.

Details

The Weibull distribution in proportional hazards parameterisation with 'shape' parameter a and 'scale' parameter m has density given by

$$f(x) = amx^{a-1}exp(-mx^a)$$

cumulative distribution function $F(x) = 1 - exp(-mx^a)$, survivor function $S(x) = exp(-mx^a)$, cumulative hazard mx^a and hazard amx^{a-1}.

[dweibull](#) in base R has the alternative 'accelerated failure time' (AFT) parameterisation with shape a and scale b. The shape parameter a is the same in both versions. The scale parameters are related as $b = m^{-1/a}$, equivalently m = b^-a.

In survival modelling, covariates are typically included through a linear model on the log scale parameter. Thus, in the proportional hazards model, the coefficients in such a model on m are interpreted as log hazard ratios.

In the AFT model, covariates on b are interpreted as time acceleration factors. For example, doubling the value of a covariate with coefficient $beta = log(2)$ would give half the expected survival time. These coefficients are related to the log hazard ratios γ as $\beta = -\gamma/a$.

Value

dweibullPH gives the density, pweibullPH gives the distribution function, qweibullPH gives the quantile function, rweibullPH generates random deviates, HweibullPH retuns the cumulative hazard and hweibullPH the hazard.

Author(s)

Christopher Jackson <chris.jackson@mrc-bsu.cam.ac.uk>

See Also

dweibull

Index