Package ‘flexsurv’

November 13, 2015

Type Package

Title Flexible Parametric Survival and Multi-State Models

Version 0.7

Date 2015-11-13

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

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

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.

License GPL (>= 2)

Depends survival,R (>= 2.15.0)

Imports mstate,muhaz,mvtnorm,deSolve,quadprog

Suggests eha,numDeriv,testthat,msm,ActuDistns,knitr,TH.data,colorspace

URL https://github.com/chjackson/flexsurv-dev

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

VignetteBuilder knitr

LazyLoad yes

LazyData yes

NeedsCompilation no

Repository CRAN

Date/Publication 2015-11-13 23:56:12

R topics documented:

`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.

Related R packages

flexsurv was written to encourage the use of flexible distributions to account for model uncertainty in survival analysis, initially the three-parameter generalized gamma, four-parameter generalized F and the Royston-Parmar spline model. However it was straightforward to modularise the design of the code to accept any generic parametric distribution.

survreg from the **survival** package, the recommended R package for survival analysis, supports two-parameter location-scale parametric models.

The **eha** package includes functions phreg and aftreg for parametric survival modelling under a variety of distributions and proportional hazards or accelerated failure time parameterisations.

Other facilities for generic maximum likelihood model fitting exist, for example fitdistr in the **MASS** package. flexsurvreg is intended to provide typical outputs and summaries of interest to survival analysts, particularly in medical applications. Feature requests along these lines are welcome.

Note that if an R package provides density and probability functions for a parametric distribution, it can then be used easily in flexsurvreg. For instance, several "reliability" distributions used in industrial statistics are available in the **VGAM** and package, and many other survival distributions are provided in **ActuDistns**. Please report unexplained inconsistencies in results between **flexsurv** and other software.

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.

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

`basis` *Natural cubic spline basis*

Description

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

Usage

```
basis(knots, x)
dbasis(knots, x)
fss(x, knots)
dfss(x, knots)
```

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.

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.

See Also

`flexsurvspline`.

---

bc — *Breast cancer survival data*

Description

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

Usage

```
data(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`

---

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.

Usage

```
bos
```

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.

---

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, subset,
            na.action, dist, inits, fixedpars=NULL, dfns=NULL, aux=NULL,
            cl=0.95, integ.opts, sr.control=survreg.control(), ...)
```

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`.

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.

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 variable giving case weights.

bhazard
: Optional variable giving expected hazards for relative survival models.

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.

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.

`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` in the source 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`.

`...` Optional arguments to the general-purpose R 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. The `coef`, `vcov` and `confint` methods for `flexsurvreg` objects work on this scale.

`loglik` Log-likelihood

`logliki` Vector of individual contributions to the log-likelihood

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

`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 distribution 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 defining 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 values 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 package (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 named "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 `dEV` and `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. 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.

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

`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

```
data(ovarian)
## 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
```

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

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 parameters 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.

`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.

`...` 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

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

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 `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.

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)
```

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)
qgenf(p, mu=0, sigma=1, Q, P, lower.tail = TRUE, log.p = FALSE)
rgenf(n, mu=0, sigma=1, Q, P)
Hgenf(x, mu=0, sigma=1, Q, P)
hgenf(x, mu=0, sigma=1, Q, P)
```

Arguments

`x,q`	Vector of quantiles.
`p`	Vector of probabilities.
`n`	number of observations. If `length(n) > 1`, the length is taken to be the number required.
`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 $P(X \le x)$, otherwise, $P(X > x)$.

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 `Llogis`) as a further special case:

`dgenf(x, mu=mu, sigma=sigma, Q=0, P=1) = 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)
qgenf.orig(p, mu=0, sigma=1, s1, s2, lower.tail = TRUE, log.p = FALSE)
rgenf.orig(n, mu=0, sigma=1, s1, s2)
Hgenf.orig(x, mu=0, sigma=1, s1, s2)
hgenf.orig(x, mu=0, sigma=1, s1, s2)
```

Arguments

`x,q`	vector of quantiles.
`p`	vector of probabilities.
`n`	number of observations. If `length(n) > 1`, the length is taken to be the number required.
`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 $P(X \le x)$, otherwise, $P(X > x)$.

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 $w = (\log(x) - \mu)/\sigma$, $B(s_1, s_2) = \Gamma(s_1)\Gamma(s_2)/\Gamma(s_1 + s_2)$ 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)
qgengamma(p, mu=0, sigma=1, Q, lower.tail = TRUE, log.p = FALSE)
rgengamma(n, mu=0, sigma=1, Q)
Hgengamma(x, mu=0, sigma=1, Q)
hgengamma(x, mu=0, sigma=1, Q)
```

Arguments

x,q	vector of quantiles.
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required.

`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 $P(X \le x)$, otherwise, $P(X > x)$.

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)
qgengamma.orig(p, shape, scale=1, k, lower.tail = TRUE, log.p = FALSE)
rgengamma.orig(n, shape, scale=1, k)
Hgengamma.orig(x, shape, scale=1, k)
hgengamma.orig(x, shape, scale=1, k)
```

Arguments

`x,q`	vector of quantiles.
`p`	vector of probabilities.
`n`	number of observations. If `length(n) > 1`, the length is taken to be the number required.
`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 $P(X \le x)$, otherwise, $P(X > x)$.

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.

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

Arguments

x, q	vector of quantiles.
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required.
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 $P(X \le x)$, otherwise, $P(X > x)$.

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 $1-\exp(b/a)$ 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 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

Hazard and cumulative hazard functions for standard distributions

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

Arguments

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

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

---

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.

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

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.

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.
`p`	vector of probabilities.
`n`	number of observations. If `length(n) > 1`, the length is taken to be the number required.
`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 $P(X \le x)$, otherwise, $P(X > x)$.

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 $a \le 1$, 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 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 `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

---

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.
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.
...	Further arguments (not used).

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. The values of this variable should 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 `NA`s 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. http://www.jstatsoft.org/v38/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 multistate 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)
```

`normboot.flexsurvreg` *Simulate from the asymptotic normal distribution of parameter estimates.*

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)
```

Arguments

`x`	A fitted model from `flexsurvreg`.
`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, `cbind`ed 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.

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

Matrix 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 matrix with one row for each permitted transition, and one column for each parameter of the parametric distribution that generates each event in the multi-state model.

Author(s)

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

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=NULL,
        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.

Author(s)

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

See Also

flexsurvreg

`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, 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 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`.
`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.
`...`	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, 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()`.

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=100000, 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 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 models, with one component for each permitted transition, as illustrated in msfit.flexsurvreg.
- `trans` — Matrix indicating allowed transitions. See msfit.flexsurvreg.
- `t` — Time to predict state occupancy probabilities for. This must be a single number, unlike pmatrix.fs.
- `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 variances. Decrease for greater speed at the expense of accuracy.
- `cl` — Width of symmetric confidence intervals, relative to 1.

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.
```

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, ...)
```

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.
...	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 intended to enable users to define "q" functions for new distributions, in cases where the distribution function pdist is available analytically, but the quantile function is not.

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.

An identical function is provided in the **msm** package.

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
```

---

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, t, newdata=NULL, start=1, M=10, tvar="trans",
         tcovs=NULL, debug=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 `i`th element of this list is the model corresponding to the `i`th 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.
`debug`	Print intermediate outputs: for development use.

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.

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

Value

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)
```

---

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,
                  start=0, ci=TRUE, B=1000, cl=0.95,
                  tidy=FALSE, ...)
```

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 combinations 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 factor 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.

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 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.

`start` Optional left-truncation time or times. The returned survival, hazard or cumulative hazard will be conditioned on survival up to this 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. Decrease for greater speed at the expense of accuracy, or set `B=0` to turn off calculation of CIs.

`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.

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

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 random sampling from the asymptotic normal distribution of the maximum likelihood estimates (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`.

Survspline *Royston/Parmar spline survival distribution*

Description

Probability density, distribution, quantile, hazard and cumulative hazard function for the Royston/Parmar spline model.

Usage

```
dsurvspline(x, gamma, beta=0, X=0, knots=c(-10,10), scale="hazard",
            timescale="log", offset=0, log=FALSE)
psurvspline(q, gamma, beta=0, X=0, knots=c(-10,10), scale="hazard",
            timescale="log", offset=0, lower.tail=TRUE, log.p=FALSE)
qsurvspline(p, gamma, beta=0, X=0, knots=c(-10,10), scale="hazard",
            timescale="log", offset=0, lower.tail=TRUE, log.p=FALSE)
rsurvspline(n, gamma, beta=0, X=0, knots=c(-10,10), scale="hazard",
            timescale="log", offset=0)
hsurvspline(x, gamma, beta=0, X=0, knots=c(-10,10), scale="hazard",
            timescale="log", offset=0)
Hsurvspline(x, gamma, beta=0, X=0, knots=c(-10,10), scale="hazard",
            timescale="log", offset=0)
```

Arguments

`x,q`	Vector of times.
`p`	Vector of probabilities.
`n`	Number of random numbers to simulate.
`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 (deprecated).
`X`	Matrix of covariate values (deprecated).
`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`.
`offset`	An extra constant to add to the linear predictor η.
`log,log.p`	Return log density or probability.
`lower.tail`	logical; if TRUE (default), probabilities are $P(X \le x)$, otherwise, $P(X > x)$.

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.

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
```

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, 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 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.
`...`	Arguments passed to `ode` in **deSolve**.

Details

This is computed by solving a second order extension of the Kolmogorov forward differential equation 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=100000, 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 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 models, with one component for each permitted transition, as illustrated in `msfit.flexsurvreg`.
`trans`	Matrix indicating allowed transitions. See `msfit.flexsurvreg`.
`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 variances. Decrease for greater speed at the expense of accuracy.
cl	Width of symmetric confidence intervals, relative to 1.

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))
```

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)
rweibullPH(n, shape, scale=1)
HweibullPH(x, shape, scale=1, log = FALSE)
hweibullPH(x, shape, scale=1, log = FALSE)
```

Arguments

`x,q`	Vector of quantiles.
`p`	Vector of probabilities.
`n`	number of observations. If `length(n) > 1`, the length is taken to be the number required.
`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 $P(X \le x)$, otherwise, $P(X > x)$.

Details

The Weibull distribution in proportional hazards parameterisation with 'shape' parameter a and 'scale' parameter m has density given by

$$f(x) = mx^{a-1}exp(-mx^a)$$

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