Package ‘smoothSurv’

July 23, 2025

Version 2.6

Date 2023-12-13

Title Survival Regression with Smoothed Error Distribution

Depends R (>= 3.0.0), survival

Imports graphics, stats

Description Contains, as a main contribution, a function to fit a regression model with possibly right, left or interval censored observations and with the error distribution expressed as a mixture of G-splines. Core part of the computation is done in compiled 'C++' written using the 'Scythe' Statistical Library Version 0.3.

Encoding UTF-8

License GPL (>= 2)

URL https://msekce.karlin.mff.cuni.cz/~komarek/

NeedsCompilation yes

Author Arnošt Komárek [aut, cre] (ORCID: <https://orcid.org/0000-0001-8778-3762>),
Kevin M. Quinn [ctb, cph] (Scythe_*.[h,cpp] files in the /src subdirectory),
Andrew D. Martin [ctb, cph] (Scythe_*.[h,cpp] files in the /src subdirectory),
Daniel B. Pemstein [ctb, cph] (Scythe_*.[h,cpp] files in the /src subdirectory),
Berwin A. Turlach [ctb] (Basis of the code in /src/solve.QP.compact.cpp)

Maintainer Arnošt Komárek <arnost.komarek@mff.cuni.cz>

Repository CRAN

Date/Publication 2023-12-13 15:30:02 UTC

Contents

`confint.smoothSurvReg` *Confidence Intervals for Regression Parameters of 'smoothSurvReg' Model*

Description

Computes confidence intervals for one or more regression related parameters (regression coefficients, scale parameter or regression coefficients in a model for scale) for a 'smoothSurvReg' model.

Usage

```
## S3 method for class 'smoothSurvReg'
confint(object, parm, level = 0.95,
    method = c("pseudo-variance", "asymptotic"), ...)
```

Arguments

`object`	Object of class smoothSurvReg.
`parm`	A specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	The confidence level required.

`method`	Type of confidence intervals to b calculated. Option "pseudo-variance" provides confidence intervals derived from inverted minus second derivatives of the penalized log-likelihood (pseudo-variance matrix), option "asymptotic" provides confidence intervals derived from the asyptotic covariance matrix of the parameter estimates.
`...`	Argument included in the function parameters for the compatibility with the generic function.

Value

A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1 - level)/2 and 1 - (1 - level)/2 in % (by default 2.5 % and 97.5 %).

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

See Also

`smoothSurvReg`

`estimTdiff` *Estimate expectation of survival times and their difference from the results given by survival regression function*

Description

Estimate expectation of survival times and their difference from the results given by survival regression function

Usage

```
estimTdiff(x, ...)
## S3 method for class 'smoothSurvReg'
estimTdiff(x, cov1, cov2, logscale.cov1, logscale.cov2,
    time0 = 0, conf.level=0.95, ...)
```

Arguments

`x`	Object of an appropriate class.
`cov1`	Vector or matrix with covariates values for which the expectations of the first survival time are to be computed. It must be a matrix with as many columns as is the number of covariates (interactions included, intercept excluded) or the vector of length equal to the number of covariates (interactions included, intercept excluded). If matrix is supplied then is assumed that each row of this matrix gives one covariate combination for the first survival time. Intercept is not to be included in `cov1`. If `cov1` is missing an expectation of a survivor time for the value of a covariate vector equal to zero is computed. If there is only intercept in the model, this parameter must be always missing.

`cov2`	Vector or matrix with covariate values for which the expectations of the second survival time are to be computed. It must be of same size as `cov1`.
`logscale.cov1`	Vector or matrix with covariate values for the expression of log-scale (if this depended on covariates) for the first survival time. It can be omitted in the case that log-scale was common for all observations.
`logscale.cov2`	Vector or matrix with covariate values for the expression of log-scale (if this depended on covariates) for the second survival time. It can be omitted in the case that log-scale was common for all observations.
`time0`	Starting time of the follow-up as used in the model. I.e. the model is assumed to be $\log(T - time0) = x'\beta + \sigma\varepsilon$
`conf.level`	confidence level of produced confidence intervals.
`...`	who knows

Value

A `data.frame` with columns named “ET1”, “sd.ET1”, “ET1.lower”, “ET1.upper”, ‘ET2”, “sd.ET2”, “ET2.lower”, “ET2.upper”, “diffT”, “sd.diffT”, “diffT.lower”, “diffT.upper” giving the estimates of expected values of the survival times for covariate values given in rows of `cov1` and `logscale.cov1`, their standard errors, estimates of expected values of survival times for covariate values given in rows of `cov2` and `logscale.cov2`, their standard errors and estimates of a difference of expected values of survival times for covariate values given in rows of `cov1`, `logscale.cov1` and `cov2`, `logscale.cov2`, their standard errors and confidence intervals.

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

See Also

`smoothSurvReg`

`eval.Gspline` *Evaluate a G-spline in a grid of values*

Description

This function computes values of

$$f(x) = \sum_{j=1}^{g} c_j \varphi_{\mu_j,\sigma_j^2}(x)$$

in a grid of x values.

In above expression, $\varphi_{\mu_j,\sigma_j^2}(x)$ denotes a density of $N(\mu_j, \sigma_j^2)$.

Usage

```
eval.Gspline(Gspline, grid)
```

Arguments

`Gspline`	A data frame with at least three columns named “Knot”, “SD basis” and “c coef.” which determine $\mu_1, \ldots, \mu_g$, $\sigma_1, \ldots, \sigma_g$ and $c_1, \ldots, c_g$. Data.frame with such properties can be found e.g. as `spline` component of the resulting object returned by functions `smoothSurvReg` and `minPenalty`.
`grid`	A numeric vector giving the grid of x values at which the G-spline is to be evaluated.

Value

A data.frame with columns named “x” (grid) and “y” (G-spline values).

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

Examples

```
spline <- minPenalty(knots=seq(-4.2, 4.2, by=0.3), sdspline=0.2, difforder=3)$spline
values <- eval.Gspline(spline, seq(-4.5, 4.5, by=0.05))
plot(values, type="l", bty="n", lwd=3)
```

`extreme value` *Density of the Extreme Value Distribution of a Minimum.*

Description

Density function of the extreme value distribution of a minimum with location α and scale β and the density of the standardized version (with zero mean and unit variance).

Usage

```
dextreme(x, alpha=0, beta=1)
dstextreme(x)
```

Arguments

`x`	Vector of quantiles.
`alpha`	Vector of location parameters.
`beta`	Vector of scale parameters.

Details

Extreme value distribution of a minimum with the location α and the scale β has a density

$$f(x) = \frac{1}{\beta}\exp\left[\frac{x-\alpha}{\beta} - \exp\left(\frac{x-\alpha}{\beta}\right)\right]$$

the mean equal to $\alpha - \beta\ e$, where e is approximately 0.5772 and the variance equal to $\beta^2\frac{\pi}{6}$. Its standardized version is obtained with $\alpha = \frac{\sqrt{6}}{\pi}\ e$ and $\beta = \frac{\sqrt{6}}{\pi}$

Value

The value of the density.

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

Examples

```
dextreme(1, (sqrt(6)/pi)*0.5772, sqrt(6)/pi)
dstextreme(1)       ## approximately same result as on the previous row
```

`fdensity.smoothSurvReg`

Density for Objects of Class 'smoothSurvReg'

Description

Compute and plot density function for given combinations of covariates based on the fitted model.

Usage

```
## S3 method for class 'smoothSurvReg'
fdensity(x, cov, logscale.cov, time0 = 0, plot = TRUE,
    by, xlim, ylim, xlab = "t", ylab = "f(t)",
    type = "l", lty, main, sub, legend, bty = "n", cex.legend = 1, ...)
```

Arguments

`x`	Object of class smoothSurvReg.
`cov`	Vector or matrix with covariates values for which the survivor curve/cdf is to be computed and plotted. It must be a matrix with as many columns as is the number of covariates (interactions included) or the vector of length equal to the number of covariates (interactions included). Intercept is not to be included in `cov`. If `cov` is missing a survivor curve for the value of a covariate vector equal to zero is plotted. If there is only intercept in the model the survivor curve based on the fitted error distribution is always plotted.

`logscale.cov`	Vector or matrix with covariate values for the expression of log-scale (if this depended on covariates). It can be omitted in the case that log-scale was common for all observations.
`time0`	Starting time of the follow-up as used in the model. I.e. the model is assumed to be $\log(T - time0) = x'\beta + \sigma\varepsilon$
`plot`	If `TRUE` the plot is directly produced, otherwise only a `data.frame` with information used for later plotting is returned.
`by`	Step for a ploting grid. If `missing` it is automatically computed.
`xlim`, `ylim`	Arguments passed to the `plot` function.
`xlab`, `ylab`	Arguments passed to the `plot` function.
`type`, `lty`	Arguments passed to the `plot` function.
`main`, `sub`	Arguments passed to the `plot` function.
`legend`, `bty`	Argument passed to the `plot` function.
`cex.legend`	argument passed to `cex` argument of the `legend` function.
`...`	Arguments passed to the `plot` function.

Value

A dataframe with columns named `x` and `y` where `x` gives the grid and `y` the values of the density function at that grid.

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

See Also

`smoothSurvReg`, `plot`

`hazard.smoothSurvReg` *Hazard Curves for Objects of Class 'smoothSurvReg'*

Description

Compute and plot hazard function for given combinations of covariates based on the fitted model.

Usage

```
## S3 method for class 'smoothSurvReg'
hazard(x, cov, logscale.cov, time0 = 0, plot = TRUE,
    by, xlim, ylim, xlab = "t", ylab = "h(t)",
    type = "l", lty, main, sub, legend, bty = "n", cex.legend = 1, ...)
```

Arguments

`x`	Object of class smoothSurvReg.
`cov`	Vector or matrix with covariates values for which the survivor curve/cdf is to be computed and plotted. It must be a matrix with as many columns as is the number of covariates (interactions included) or the vector of length equal to the number of covariates (interactions included). Intercept is not to be included in `cov`. If `cov` is missing a survivor curve for the value of a covariate vector equal to zero is plotted. If there is only intercept in the model the survivor curve based on the fitted error distribution is always plotted.
`logscale.cov`	Vector or matrix with covariate values for the expression of log-scale (if this depended on covariates). It can be omitted in the case that log-scale was common for all observations.
`time0`	Starting time of the follow-up as used in the model. I.e. the model is assumed to be $\log(T - time0) = x'\beta + \sigma\varepsilon$
`plot`	If `TRUE` the plot is directly produced, otherwise only a `data.frame` with information used for later plotting is returned.
`by`	Step for a ploting grid. If `missing` it is automatically computed.
`xlim`, `ylim`	Arguments passed to the `plot` function.
`xlab`, `ylab`	Arguments passed to the `plot` function.
`type`, `lty`	Arguments passed to the `plot` function.
`main`, `sub`	Arguments passed to the `plot` function.
`legend`, `bty`	Argument passed to the `plot` function.
`cex.legend`	argument passed to `cex` argument of the `legend` function.
`...`	Arguments passed to the `plot` function.

Value

A dataframe with columns named `x` and `y` where `x` gives the grid and `y` the values of the hazard function at that grid.

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

See Also

`smoothSurvReg`, `plot`

`minPenalty` *Minimize the penalty term under the two (mean and variance) constraints*

Description

This function minimizes

$$\frac{1}{2}\sum_{j=m+1}^{g}\Bigl(\Delta^m a_j\Bigr)^2$$

with respect to $a_1, \dots, a_g$ under the constraints

$$\sum_{j=1}^{g} c_j \mu_j = 0$$

and

$$\sum_{j=1}^{g} c_j(\mu_j^2 + \sigma_0^2) = 1,$$

where

$$c_j = \frac{\exp(a_j)}{\sum_{l=1}^{g}} \exp(a_l)$$

with one of a's fixed to zero.

Note that the minimum is always zero. We are thus mainly interested in the point where the minimum is reached.

Usage

```
minPenalty(knots = NULL, dist.range = c(-6, 6), by.knots = 0.3, sdspline = NULL,
    difforder = 3, init.c,
   maxiter = 200, rel.tolerance = 1e-10, toler.chol = 1e-15, toler.eigen = 1e-3,
    maxhalf = 10, debug = 0, info = TRUE)
```

Arguments

`knots`	A vector of knots $\mu_1, \dots, \mu_g$.
`dist.range`	Approximate minimal and maximal knot. If not given by `knots` the knots are determined as `c(seq(0, dist.range[2], by = by.knots), seq(0, dist.range[1], by = -by.knots))`. The sequence of knots is sorted and multiple entries are removed.
`by.knots`	The distance between the two knots used when building a vector of knots if these are not given by `knots`.
`sdspline`	Standard deviation σ_0^2 of the basis G-spline (here it appeares only in the variance constraint). If not given it is determined as 2/3 times the maximal distance between the two knots. If `sdspline` >= 1 it is changed to 0.9 to be able to satisfy the constraints.

`difforder`	The order of the finite difference used in the penalty term.
`init.c`	Optional vector of the initial values for the G-spline coefficients c, all values must lie between 0 and 1 and must sum up to 1.
`maxiter`	Maximum number of Newton-Raphson iterations.
`rel.tolerance`	(Relative) tolerance to declare the convergence. For this function, the convergence is declared if absolute value of the penalty is lower than `rel.tolerance` and if both constraints are satisfied up to `rel.tolerance`.
`toler.chol`	Tolerance to declare Cholesky decomposition singular.
`toler.eigen`	Tolerance to declare an eigen value of a matrix to be zero.
`maxhalf`	Maximum number of step-halving steps if updated estimate leads to a decrease of the objective function.
`debug`	If non-zero print debugging information.
`info`	If TRUE information concerning the iteration process is printed during the computation to the standard output.

Value

A list with the components “spline”, “penalty”, “warning”, “fail”.

`spline`	A data frame with columns named “Knot”, “SD basis”, “c coef.” and “a coef.” which gives the optimal values of $c_1, \ldots, c_g$ and $a_1, \ldots, a_g$ in the latter two columns. This data.frame can be further worked out using the function `eval.Gspline`.
`penalty`	The value of the penalty term when declaring convergence.
`warning`	Possible warnings concerning the convergence.
`fail`	Failure indicator. It is zero if everything went OK.

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

Examples

```
optimum <- minPenalty(knots=seq(-4.2, 4.2, by = 0.3), sdspline=0.2, difforder=3)
where <- optimum$spline
print(where)
show <- eval.Gspline(where, seq(-4.2, 4.2, by=0.05))
plot(show, type="l", bty="n", lwd=2)
```

`piece` *Left Continuous Piecewise Constant Function with a Finite Support.*

Description

Function to evaluate a left continuous piecewise constant function with a finite support.

Usage

```
piece(x, breaks, values)
```

Arguments

`x`	Vector of values where the piecewise constant function should be evaluated.
`breaks`	Vector of sorted breakpoints of the piecewise constant function.
`values`	Values of the piecewise constant function. It takes the value `value[i]` on the interval `(breaks[i], breaks[i+1]]`. The function is assumed to be zero outside its range specified as `(breaks[1], breaks[length(breaks)]]`. The length of the vector `values` must be equal to `length(breaks) - 1`

Value

The value of the piecewise constant function.

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

Examples

```
my.breaks <- c(-2, 1.5, 4, 7)
my.values <- c(0.5, 0.9, -2)
grid <- seq(-3, 8, by = 0.25)
piece(grid, my.breaks, my.values)
```

`plot.smoothSurvReg` *Plot Objects of Class 'smoothSurvReg'*

Description

Plot the fitted error distribution.

Usage

```
## S3 method for class 'smoothSurvReg'
plot(x, plot = TRUE, resid = TRUE, knots = TRUE,
   compare = TRUE, components = FALSE, standard = TRUE,
   by, toler.c = 1e-5,
   xlim, ylim,
  xlab = expression(epsilon), ylab = expression(paste("f(",epsilon,")", sep = "")),
   type = "l", lty = 1, main, sub, bty = "n", ...)
```

Arguments

`x`	Object of class smoothSurvReg.
`plot`	If `TRUE` the plot is directly produced, otherwise only a `data.frame` with information used for later plotting is returned.
`resid`	If `resid` & `plot` residuals are added to the plot in the following way. Residuals based on exact observations are plotted by `pch = 3` (+), residuals based on right censored observations using `pch = 4` (x), residuals based on left censored observations using `pch = 2` (traingle) and midpoints of residuals based on interval censored observations using `pch = 5` (diamond).
`knots`	If `knots` & `plot` bullets indicating (transformed) knots corresponding to 'c' G-spline coefficients higher than `toler.c` are added to the x-axis of the plot.
`compare`	If `compare` & `!components` & `standard` & `plot` plots of the three parametric distributions (st. normal, st. logistic and st. minimum extreme value) are added to the plot for comparison.
`components`	If `components` & `standard` & `plot` dashed lines with weighted G-spline components are added to the plot.
`standard`	If `standard` data for plotting the fitted error distribution are created, otherwise data for plotting the distribution of $\alpha + \sigma\varepsilon$ are created. In the case that $\log(\sigma)$ depends on covariates and `standard` is FALSE, all covariates equal to zero are used to compute $\log(\sigma)$.
`by`	Step for a ploting grid. If `NULL` it is automatically computed.
`toler.c`	Tolerance to indicate zero 'c' G-spline coefficients used if `knots == TRUE`.
`xlim, ylim`	Arguments passed to the `plot` function.
`xlab, ylab`	Arguments passed to the `plot` function.
`type, lty`	Arguments passed to the `plot` function.
`main, sub`	Arguments passed to the `plot` function.
`bty`	Argument passed to the `plot` function.
`...`	Arguments passed to the `plot` function.

Value

A dataframe with columns named `x` and `y` where `x` gives the grid and `y` the values of the density at that grid.

Author(s)

Arnošt Komárek `<arnost.komarek@mff.cuni.cz>`

See Also

`smoothSurvReg`, `plot`

`print.estimTdiff` *Print for Objects of Class 'estimTdiff'*

Description

Print a summary information of the estimates and tests for expected values of survival times based on a regression.

Usage

```
## S3 method for class 'estimTdiff'
print(x, digits = min(options()$digits, 4), ...)
```

Arguments

`x`	Object of class estimTdiff.
`digits`	Controls the number of digits to print when printing numeric values. It is a suggestion only. Valid values are 1...22.
`...`	Further arguments passed to or from other methods.

Value

No return value, called to print the object.

Author(s)

Arnošt Komárek `<arnost.komarek@mff.cuni.cz>`

See Also

`smoothSurvReg`, `print`

`print.smoothSurvReg` *Summary and Print for Objects of Class 'smoothSurvReg'*

Description

Print a summary information of the fitted model.

For **regression coefficients** the following information is given:

'`Value`'	-	estimate of the coefficient
'`Std.Error`'	-	estimated standard error based on the pseudo-variance estimate (3.1) in Komárek, Lesaffre and Hilton (2005)
'`Std.Error2`'	-	estimated standard error based on the asymptotic variance estimate (3.2) in Komárek, Lesaffre and Hilton (2005)
'`Z`'	-	the Wald statistic obtained as '`Value`' divided by '`Std.Error`'
'`Z2`'	-	the Wald statistic obtained as '`Value`' divided by '`Std.Error2`'
'`p`'	-	the two-sided P-value based on normality of the statistic '`Z`'
'`p2`'	-	the two-sided P-value based on normality of the statistic '`Z2`'

Further, we print:

'`Lambda`'	-	the optimal value of the smoothing hyperparameter divided by the sample size, i.e., λ/n in the notation of Komárek, Lesaffre and Hilton (2005)
'`Log(Lambda)`'	-	logarithm of the above
'`df`'	-	effective degrees of freedom of the model, see Section 2.2.3 of Komárek, Lesaffre and Hilton (2005)
'`AIC`'	-	Akaike's information criterion of the model, see Section 2.2.3 of Komárek, Lesaffre and Hilton (2005)

With argument **spline** set to `TRUE`, analogous table like that for the regression coefficients is printed also for the weights of the penalized Gaussian mixture (G-spline).

Usage

```
## S3 method for class 'smoothSurvReg'
print(x, spline, digits = min(options()$digits, 4), ...)
## S3 method for class 'smoothSurvReg'
summary(object, spline, digits = min(options()$digits, 4), ...)
```

Arguments

`x`	Object of class smoothSurvReg.
`object`	Object of class smoothSurvReg.
`spline`	TRUE/FALSE. If TRUE an information on fitted G-spline is printed.

`digits`	Controls the number of digits to print when printing numeric values. It is a suggestion only. Valid values are 1...22.
`...`	Further arguments passed to or from other methods.

Value

No return value, called to print the object.

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

References

Komárek, A., Lesaffre, E., and Hilton, J. F. (2005). Accelerated failure time model for arbitrarily censored data with smoothed error distribution. *Journal of Computational and Graphical Statistics*, **14**, 726–745.

Lesaffre, E., Komárek, A., and Declerck, D. (2005). An overview of methods for interval-censored data with an emphasis on applications in dentistry. *Statistical Methods in Medical Research*, **14**, 539–552.

See Also

`smoothSurvReg`, `print`, `summary`

`residuals.smoothSurvReg`

Residuals for Objects of Class 'smoothSurvReg'

Description

Compute residuals for the fitted model.

Usage

```
## S3 method for class 'smoothSurvReg'
residuals(object, ...)
```

Arguments

`object`	Object of class smoothSurvReg.
`...`	Argument included in the function parameters for the compatibility with the generic function.

Value

A dataframe with columns named `res`, `res2` and `censor` where the column `res2` is included only if there are any interval censored observations. Column `res` contains all residuals, column `res2` is applicable only for interval censored observations. Column `censor` gives the type of censoring (0 for right censoring, 1 for exact observations, 2 for left censoring and 3 for interval censoring).

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

See Also

`smoothSurvReg`

`smoothSurvReg` *Regression for a Survival Model with Smoothed Error Distribution*

Description

Regression for a survival model. These are all time-transformed location models, with the most useful case being the accelerated failure models that use a log transformation. Error distribution is assumed to be a mixture of G-splines. Parameters are estimated by the penalized maximum likelihood method.

Usage

```
smoothSurvReg(formula = formula(data), logscale = ~1,
   data = parent.frame(), subset, na.action = na.fail,
   init.beta, init.logscale, init.c, init.dist = "best",
   update.init = TRUE, aic = TRUE, lambda = exp(2:(-9)),
   model = FALSE, control = smoothSurvReg.control(), ...)
```

Arguments

`formula`	A formula expression as for other regression models. See the documentation for `lm` and `formula` for details. Use `Surv` on the left hand side of the formula.
`logscale`	A formula expression to determine a possible dependence of the log-scale on covariates.
`data`	Optional data frame in which to interpret the variables occurring in the formula.
`subset`	Subset of the observations to be used in the fit.
`na.action`	Function to be used to handle any NAs in the data. It's default value is `na.fail`. It is not recommended to change it in the case when `logscale` depends on covariates.
`init.beta`	Optional vector of the initial values of the regression parameter β (intercept and regression itself).

`init.logscale`	Optional value of the initial value of the parameters that determines the log-scale parameter $\log(\sigma)$.
`init.c`	Optional vector of the initial values for the G-spline coefficients c, all values must lie between 0 and 1 and must sum up to 1.
`init.dist`	A character string specifying the distribution used by `survreg` to find the initial values for parameters (if not given by the user). It is assumed to name `"best"` or an element from `survreg.distributions`. These include `"weibull"`, `"exponential"`, `"gaussian"`, `"logistic"`, `"lognormal"` and `"loglogistic"`. If "best" is specified one of `"lognormal"`, `"weibull"` and `"loglogistic"` giving the highest likelihood is used.
`update.init`	If TRUE, the initial values are updated during the grid search for the lambda parameter giving the optimal AIC. Otherwise, fits with all lambdas during the grid search start with same initials determine at the beginning either from the values of `init.beta`, `init.scale`, `init.c` or from the initial `survreg` fit as determined by the parameter `init.dist`.
`aic`	If TRUE the optimal value of the tuning parameter λ is determined via a grid search through the values specified by the parameter `lambda`. If FALSE, only the model with $\lambda =$ `lambda[1]` is fitted.
`lambda`	A grid of values of the tuning parameter λ searched for the optimal value if `aic` = TRUE.
`model`	If TRUE, the model frame is returned.
`control`	A list of control values, in the format productced by `smoothSurvReg.control`.
`...`	Other arguments which will be passed to `smoothSurvReg.control`. See its help page for more options to control the fit and for the possibility to fix some values and not to estimate them.

Details

Read the papers referred below.

There is a slight difference in the definition of the penalty used by the R function compared to what is written in the paper. The penalized log-likelihood given in the paper has a form

$$\ell_P(\theta) = \ell(\theta) - \frac{\lambda}{2}\sum_{j=m+1}^{g}(\Delta^m a_j)^2,$$

while the penalized log-likelihood used in the R function multiplies the tuning parameter λ given by `lambda` by a sample size n to keep default values more or less useful for samples of different sizes. So that the penalized log-likelihood which is maximized by the R function has the form

$$\ell_P(\theta) = \ell(\theta) - \frac{\lambda \cdot n}{2}\sum_{j=m+1}^{g}(\Delta^m a_j)^2.$$

Value

An object of class `smoothSurvReg` is returned. See `smoothSurvReg.object` for details.

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

References

Komárek, A., Lesaffre, E., and Hilton, J. F. (2005). Accelerated failure time model for arbitrarily censored data with smoothed error distribution. *Journal of Computational and Graphical Statistics*, **14**, 726–745.

Lesaffre, E., Komárek, A., and Declerck, D. (2005). An overview of methods for interval-censored data with an emphasis on applications in dentistry. *Statistical Methods in Medical Research*, **14**, 539–552.

Examples

```
##### EXAMPLE 1:  Common scale
##### ========================
### We generate interval censored data and fit a model with few artificial covariates
set.seed(221913282)
x1 <- rbinom(50, 1, 0.4)                                         ## binary covariate
x2 <- rnorm(50, 180, 10)                                         ## continuous covariate
y1 <- 0.5*x1 - 0.01*x2 + 0.005 *x1*x2 + 1.5*rnorm(50, 0, 1)    ## generate log(T), left limit
t1 <- exp(y1)                                          ## left limit of the survival time
t2 <- t1 + rgamma(50, 1, 1)                            ## right limit of the survival time
surv <- Surv(t1, t2, type = "interval2")                         ## survival object

## Fit the model with an interaction
fit1 <- smoothSurvReg(surv ~ x1 * x2, logscale = ~1, info = FALSE, lambda = exp(2:(-1)))

## Print the summary information
summary(fit1, spline = TRUE)

## Plot the fitted error distribution
plot(fit1)

## Plot the fitted error distribution with its components
plot(fit1, components = TRUE)

## Plot the cumulative distribution function corresponding to the error density
survfit(fit1, cdf = TRUE)

## Plot survivor curves for persons with (x1, x2) = (0, 180) and (1, 180)
cov <- matrix(c(0, 180, 0,   1, 180, 180), ncol = 3, byrow = TRUE)
survfit(fit1, cov = cov)

## Plot hazard curves for persons with (x1, x2) = (0, 180) and (1, 180)
cov <- matrix(c(0, 180, 0,   1, 180, 180), ncol = 3, byrow = TRUE)
hazard(fit1, cov = cov)

## Plot densities for persons with (x1, x2) = (0, 180) and (1, 180)
cov <- matrix(c(0, 180, 0,   1, 180, 180), ncol = 3, byrow = TRUE)
fdensity(fit1, cov = cov)
```

```
## Compute estimates expectations of survival times for persons with
## (x1, x2) = (0, 180), (1, 180), (0, 190), (1, 190), (0, 200), (1, 200)
## and estimates of a difference of these expectations:
## T(0, 180) - T(1, 180), T(0, 190) - T(1, 190), T(0, 200) - T(1, 200),
cov1 <- matrix(c(0, 180, 0,   0, 190, 0,   0, 200, 0), ncol = 3, byrow = TRUE)
cov2 <- matrix(c(1, 180, 180,   1, 190, 190,   1, 200, 200), ncol = 3, byrow = TRUE)
print(estimTdiff(fit1, cov1 = cov1, cov2 = cov2))


##### EXAMPLE 2:  Scale depends on covariates
##### ======================================
### We generate interval censored data and fit a model with few artificial covariates
set.seed(221913282)
x1 <- rbinom(50, 1, 0.4)                                        ## binary covariate
x2 <- rnorm(50, 180, 10)                                        ## continuous covariate
x3 <- runif(50, 0, 1)                                ## covariate for the scale parameter
logscale <- 1 + x3
scale <- exp(logscale)
y1 <- 0.5*x1 - 0.01*x2 + 0.005 *x1*x2 + scale*rnorm(50, 0, 1)   ## generate log(T), left limit
t1 <- exp(y1)                                          ## left limit of the survival time
t2 <- t1 + rgamma(50, 1, 1)                           ## right limit of the survival time
surv <- Surv(t1, t2, type = ”interval2”)                         ## survival object

## Fit the model with an interaction
fit2 <- smoothSurvReg(surv ~ x1 * x2, logscale = ~x3, info = FALSE, lambda = exp(2:(-1)))

## Print the summary information
summary(fit2, spline = TRUE)

## Plot the fitted error distribution
plot(fit2)

## Plot the fitted error distribution with its components
plot(fit2, components = TRUE)

## Plot survivor curves for persons with (x1, x2) = (0, 180) and (1, 180)
## x3 = 0.8 and 0.9
cov <- matrix(c(0, 180, 0,   1, 180, 180), ncol = 3, byrow = TRUE)
logscale.cov <- c(0.8, 0.9)
survfit(fit2, cov = cov, logscale.cov = logscale.cov)

## Plot hazard curves for persons with (x1, x2) = (0, 180) and (1, 180)
## x3 = 0.8 and 0.9
cov <- matrix(c(0, 180, 0,   1, 180, 180), ncol = 3, byrow = TRUE)
logscale.cov <- c(0.8, 0.9)
hazard(fit2, cov = cov, logscale.cov=c(0.8, 0.9))

## Plot densities for persons with (x1, x2) = (0, 180) and (1, 180)
## x3 = 0.8 and 0.9
cov <- matrix(c(0, 180, 0,   1, 180, 180), ncol = 3, byrow = TRUE)
logscale.cov <- c(0.8, 0.9)
fdensity(fit2, cov = cov, logscale.cov = logscale.cov)
```

```
## More involved examples can be found in script files
## used to perform analyses  and draw pictures
## presented in above mentioned references.
## These scripts and some additional files can be found as *.tar.gz files
## in the /inst/doc directory of this package.
##
```

`smoothSurvReg.control` *More Options for 'smoothSurvReg'*

Description

This function checks and sets the fitting options for smoothSurvReg. Its arguments can be used instead of ... in a call to smoothSurvReg.

Usage

```
smoothSurvReg.control(est.c = TRUE, est.scale = TRUE,
   maxiter = 200, firstiter = 0, rel.tolerance = 5e-5,
   toler.chol = 1e-15, toler.eigen = 1e-3,
   maxhalf = 10, debug = 0, info = TRUE, lambda.use = 1.0, sdspline = NULL,
   difforder = 3, dist.range = c(-6, 6), by.knots = 0.3,
   knots = NULL, nsplines = NULL, last.three = NULL)
```

Arguments

`est.c`	If TRUE the G-spline coefficients are estimated. Otherwise, they are fixed to the values given by `init.c` parameter of `smoothSurvReg`.
`est.scale`	If TRUE the scale parameter σ is estimated. Otherwise, it is fixed to the value given by `init.scale` parameter of `smoothSurvReg`.
`maxiter`	Maximum number of Newton-Raphson iterations.
`firstiter`	The index of the first iteration. This option comes from older versions of this function.
`rel.tolerance`	(Relative) tolerance to declare the convergence. In this version of the function, the convergence is declared if the relative difference between two consecutive values of the penalized log-likelihood are smaller than rel.tolerance.
`toler.chol`	Tolerance to declare Cholesky decomposition singular.
`toler.eigen`	Tolerance to declare an eigen value of a matrix to be zero.
`maxhalf`	Maximum number of step-halving steps if updated estimate leads to a decrease of the objective function.
`debug`	If non-zero print debugging information.
`info`	If TRUE information concerning the iteration process is printed during the computation to the standard output.

`lambda.use`	The value of the tuning (penalty) parameter λ used in a current fit by the `smoothSurvReg.fit` function. Value of this option is not interesting for the user. The parameter `lambda` of the function `smoothSurvReg` is more important for the user.
`sdspline`	Standard deviation of the basis G-spline. If not given it is determined as 2/3 times the maximal distance between the two knots. If `est.c = TRUE` and `sdspline` >= 1 it is changed to 0.9 to be able to satisfy the constraints imposed to the fitted error distribution.
`difforder`	The order of the finite difference used in the penalty term.
`dist.range`	Approximate minimal and maximal knot. If not given by `knots` the knots are determined as `c(seq(0, dist.range[2], by = by.knots), seq(0, dist.range[1], by = -by.knots))`. The sequence of knots is sorted and multiple entries are removed.
`by.knots`	The distance between the two knots used when building a vector of knots if these are not given by `knots`. This option is ignored if `nsplines` is not `NULL`.
`knots`	A vector of knots.
`nsplines`	This option is ignored at this moment. It is used to give the number of G-splines to the function `smoothSurvReg.fit`.
`last.three`	A vector of length 3 with indeces of reference knots. The 'a' coefficient of the `knot[last.three[1]]` is then equal to zero, 'a' coefficients with indeces `last.three[2:3]` are expressed as a function of remaining 'a' coefficients such that resulting error distribution has zero mean and unit variance. If `maxiter > 0` `last.three` is determined after the convergence is reached. If `maxiter == 0` `last.three` is used to compute variance matrices.

Value

A list with the same elements as the input except `dist.range` and `by.knots` is returned.

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

`smoothSurvReg.fit` *Work Function to Fit the Model Using 'smoothSurvReg'*

Description

Fit the survival regression model with smoothed error distribution. This function is not to be called by the user.

Usage

```
smoothSurvReg.fit(x, z, y, offset = NULL, correctlik, init, controlvals, common.logscale)
```

Arguments

`x`	A covariate matrix.
`z`	A covariate matrix for log(scale).
`y`	A two- or three- column matrix with response. The last column indicate the type of censoring. See `Surv` for details.
`offset`	A vector with possible offset term.
`correctlik`	Correction term to the likelihood due to the log transformation of the response.
`init`	A list with components `beta`, `scale`, `ccoef` giving the initial values for the fitting process.
`controlvals`	A list returned by `smoothSurvReg.control`.
`common.logscale`	Indicator (TRUE/FALSE) indicating whether the log-scale is same for all observations or whether it depends on covariates.

Value

A list with components `regres`, `spline`, `loglik`, `aic`, `degree.smooth`, `var`, `var2`, `dCdC`, `iter`, `estimated`, `warning`, `fail`, `H`, `I`, `G`, `U`.

Note

WARNING: Most users will call the higher level routine smoothSurvReg. Consequently, this function has very few error checks on its input arguments.

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

`smoothSurvReg.object` *Smoothed Survival Regression Object*

Description

This class of objects is returned by the `smoothSurvReg` class of functions to represent a fitted smoothed survival regression model.

Objects of this class have methods for the functions `print`, `summary`, `plot`, `residuals`, `survfit`.

COMPONENTS I

The following components must be included in a legitimate `smoothSurvReg` object.

fail Indicator of the failure of the fitting procedure. Possible values are 0 for no problems, 3 if the iteration process was stopped because of non-positive definite minus Hessian, 4 if the eiteration process was stopped because too many halving steps were performed, 5 if it was not possible to find the three reference knots (it was not then possible to perform optimization with respect to the full parameter vector), 6 if the maximal number of iterations was performed without reaching a convergence. The `fail` component is increased by 10 if the final minus Hessian of the penalized log-likelihood was not positive definite. The `fail` component is further increased by 20 if the computed effective degrees of freedom were non-positive. The `fail` component is further increased by 40 if there are negative estimates of standard errors for some regression parameters. The `fail` component is 99 or higher if the fitting procedure failed at all and there is no fit produced.

COMPONENTS II

The following components must be included in a legitimate `smoothSurvReg` object if `fail` is lower than 99.

regres Estimates of the regression parameters α, β, σ if these have been estimated with their standard errors stored in a data frame with `colnames` “Value”, “Std.Error”, “Std.Error2” and `rownames` derived from the names of the design matrix with “(Intercept)” for the intercept, “Scale” for the scale and “Log(scale)” for the log-scale. If the log-scale depends on covariates then rows named “LScale.(Intercept)”, “LScale.cov1” etc. give estimates of regression parameters for log-scale. The two standard errors are computed using either `var` or `var2` described below.

spline Description of the fitted error density. A data frame with `colnames` “Knot”, “SD basis”, “c coef.”, “Std.Error.c”, “Std.Error2.c”, “a coef.”, “Std.Error.a” and “Std.Error2.a” and `rownames` knot[1], ..., knot[g] where g stands for the number of basis G-splines. The column “Knot” contains the knots in ascending order, “SD basis” the standard deviation of an appropriate basis G-spline, “c coef.” estimates of the G-spline coefficients and “Std.Error.c” and “Std.Error2.c” the estimates of their standard errors based either on `var` or `var2`. The column “a coef.” contains the estimates of transformed c coefficients where

$$c_j = \frac{\exp(a_j)}{\sum_{l=1}^{g} \exp(a_l)}, j = 1, \dots, g.$$

If the error distribution is estimated, one of the a coefficients is set to zero and two other a’s are expressed as a function of the remaining a coefficients (to avoid equality constraints concerning the mean and the variance of the error distribution). The standard error for these three a coefficients is then not available (it is equal to `NA`). Standard error is set to `NaN` is a diagonal element of the appropriate covariance matrix was negative.

loglik Maximized penalized log-likelihood, log-likelihood and the penalty term. A data frame with one row and three columns named “Log Likelihood”, “Penalty” and “Penalized Log Likelihood”.

aic Akaike’s information criterion of the fitted model computed as a maximized value of the penalized log-likelihood minus the effective degrees of freedom.

degree.smooth Effective degrees of freedom, number of parameters and related information. A data frame with one row and columns named “Lambda”, “Log(Lambda)”, “df”, “Number of parameters”, “Mean param.”, “Scale param.”, “Spline param.” where “Lambda” gives the value of the tunning parameter used in the final (optimal) fit, “df” the effective degrees of freedom, “Number of parameters” the real number of parameters and “Mean param.”, “Scale param.” and “Spline param.” its decomposition. Note that if G-spline coefficients are estimated “Spline param.” is equal to the number of basis G-spline with non-zero coefficients minus three.

var The estimate of the covariance matrix of the estimates based on the Bayesian approximation. It is equal to the inverse of the converged minus Hessian of the penalized log-likelihood. Note that there are no columns and rows corresponding to the three transformed G-spline coefficients since these are functions of the remaining transformed G-spline coefficients (to avoid equality constraints).

var2 The estimate of the covariance matrix of the estimates based on the asymptotic theory for penalized models. It is equal to $H^{-1}\,I\,H^{-1}$ where H is converged minus Hessian of the penalized log-likelihood and I is converged minus Hessian of the log-likelihood component of the penalized log-likelihood.

dCdD A matrix with derivatives of c spline coefficients with respect to d spline coefficients (these are a coefficients with three of them omitted). This matrix can be used later to compute estimates and standard errors of functions of original parameters using a Delta method. For closer definition of d coefficients see an enclosed document.

iter Used number of iterations to fit the model with the optimal λ.

estimated Indicator of what has really been estimated and not fixed. A four-component vector with component names “(Intercept)”, “Scale”, “ccoef”, “common.logscale”. The first component is TRUE if the intercept was included in the regression model. The second component is TRUE if the scale parameter was not fixed, the third component is TRUE is the G-spline coefficients were not fixed. The fourth component is TRUE if the log-scale does not depend on covariates.

warning A data frame with one column called “warnings” and three rows called “Convergence”, “Final minus Hessian” and “df” containing a string information corresponding to the value of the `fail` component of the object. It contains a string “OK” if there are no problems with the appropriate part of the fitting process.

H Converged minus Hessian of the penalized log-likelihood.

I Converged minus Hessian of the log-likelihood component of the penalized log-likelihood. $I = H - G$.

G Converged minus Hessian of the penalty term of the penalty term of the penalized log-likelihood. $G = H - I$.

U Converged score vector based on the penalized log-likelihood.

na.action The `na.action` attribute, if any, that was returned by the `na.action` routine.

terms The `terms` object used.

formula A symbolic description of the model to be fit.

call The matched call.

init.dist A string indicating the error distribution of the untransformed response to find the initial values. Possible values are “lognormal”, “loglogistic”, “weibull”.

model If requested, the model frame used.

x The model matrix used.

y The response matrix used (two columns if there were no interval censored observations, three columns if there were some interval censored observations). The last column indicates the death status.

z The model matrix used for the expression of log-scale.

init.spline A data frame describing the initial error density. It has columns named “Knot”, “SD basis”, “c coef.” and rows named “knot[1]”, . . . , “knot[g]”.

init.regres Initial estimates of the regression parameters. A data frame with one column named “Value” and rows named as in the `regres` component of the `smoothSurvReg` object.

adjust Adjusted intercept and scale. A data frame with a column named “Value” and rows named “(Intercept)” and “Scale”. “(Intercept)” gives the overall intercept taking into account the mean of the fitted error distribution, “Scale” gives the overall scale taking into account the variance of the fitted error distribution. If the error distribution is standardized (always when G-spline coefficients are estimated) then the “(Intercept)” is equal to the “(Intercept)” from the `regres` component and “Scale” is equal to the “Scale” of either `regres` or `init.regres` component. `NA`’s appeare in this `data.frame` in the case that log-scale depends on covariates.

error.dist A data frame with columns named “Mean”, “Var” and “SD” and a row named “Error distribution: ” giving the mean, variance and the standard deviation of the fitted error distribution. These are equal to 0, 1 and 1 if the G-spline coefficients were estimated.

searched Information concerning the searched values of the tunning paramater λ when looking for the best AIC. A data frame with columns named “Lambda”, “Log(Lambda)”, “AIC”, “df”, “PenalLogLik”, “LogLik”, “nOfParm”, “fail”.

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

`standardized logistic` *Density of Standardized Logistic Distribution.*

Description

Density function of the logistic distribution with zero mean and unit variance.

Usage

```
dstlogis(x)
```

Arguments

`x`	Vector of quantiles.

Details

`dstlogis(x)` = `dlogis(x, 0, sqrt(3)/pi)`

Value

The value of the density.

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

See Also

`dlogis` for the logistic distribution.

Examples

```
dstlogis(0)
dstlogis(seq(-3, 3, 0.2))
```

`std.data` *Standardization of the Data*

Description

Chosen columns of a `data.frame` are standardized by using a sample mean and sample standard deviation.

Usage

```
std.data(datain, cols)
```

Arguments

`datain`	Input data frame.
`cols`	A character vector with names of the columns to be standardized.

Details

For each chosen column the sample mean and the sample standard deviation is computed and then used to standardize the column. Missing values are ignored when computing the mean and the standard deviation.

Value

A `data.frame` with same variables as the input `data.frame`. Chosen columns are standardized.

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

See Also

`scale`

Examples

```
variable1 <- rnorm(30)
variable2 <- rbinom(30, 1, 0.4)
variable3 <- runif(30)
data.example <- data.frame(variable1, variable2, variable3)
## We standardize only the first and the third column.
data.std <- std.data(data.example, c("variable1", "variable3"))
print(data.std)
print(c(mean(data.std$variable1), sd(data.std$variable1)))
print(c(mean(data.std$variable3), sd(data.std$variable3)))
```

`survfit.smoothSurvReg` *Survivor Curves for Objects of Class 'smoothSurvReg'*

Description

Compute and plot survivor function/cumulative distribution function for given combinations of covariates based on the fitted model.

Usage

```
## S3 method for class 'smoothSurvReg'
survfit(formula, cov, logscale.cov, time0 = 0, plot = TRUE, cdf = FALSE,
   by, xlim, ylim = c(0, 1), xlab = "t", ylab,
   type = "l", lty, main, sub, legend, bty = "n", cex.legend = 1, ...)
```

Arguments

`formula`	Object of class smoothSurvReg.
`cov`	Vector or matrix with covariates values for which the survivor curve/cdf is to be computed and plotted. It must be a matrix with as many columns as is the number of covariates (interactions included) or the vector of length equal to the number of covariates (interactions included). Intercept is not to be included in `cov`. If `cov` is missing a survivor curve for the value of a covariate vector equal to zero is plotted. If there is only intercept in the model the survivor curve based on the fitted error distribution is always plotted.
`logscale.cov`	Vector or matrix with covariate values for the expression of log-scale (if this depended on covariates). It can be omitted in the case that log-scale was common for all observations.
`time0`	Starting time of the follow-up as used in the model. I.e. the model is assumed to be $\log(T - time0) = x'\beta + \sigma\varepsilon$
`plot`	If `TRUE` the plot is directly produced, otherwise only a `data.frame` with information used for later plotting is returned.
`cdf`	If `TRUE` cumulative distribution function is plotted instead of the survivor function.
`by`	Step for a ploting grid. If `NULL` it is automatically computed.

`xlim`, `ylim`	Arguments passed to the `plot` function.
`xlab`, `ylab`	Arguments passed to the `plot` function.
`type`, `lty`	Arguments passed to the `plot` function.
`main`, `sub`	Arguments passed to the `plot` function.
`legend`, `bty`	Argument passed to the `plot` function.
`cex.legend`	argument passed to `cex` argument of the `legend` function.
`...`	Arguments passed to the `plot` function.

Value

A dataframe with columns named `x` and `y` where `x` gives the grid and `y` the values of the survivor/cum. distribution function at that grid.

Author(s)

Arnošt Komárek <`arnost.komarek@mff.cuni.cz`>

See Also

`smoothSurvReg`, `plot`

Index