Package ‘cuRe’

May 8, 2026

Type Package

Title Parametric Cure Model Estimation

Version 1.1.1

Date 2023-07-03

Maintainer Lasse Hjort Jakobsen <`hjort9230@hotmail.com`>

Description Contains functions for estimating generalized parametric mixture and non-mixture cure models <doi:10.1016/j.cmpb.2022.107125>, loss of lifetime, mean residual lifetime, and crude event probabilities.

URL `https://github.com/LasseHjort/cuRe`

BugReports `https://github.com/LasseHjort/cuRe/issues`

License GPL (>= 2)

Encoding UTF-8

Depends R (>= 3.5.0), survival, rstpm2, date

Imports numDeriv, statmod, relsurv, reshape2

LazyData Yes

RoxygenNote 7.2.3

NeedsCompilation no

Author Lasse Hjort Jakobsen [aut, cre],
Mark Clements [cph],
Rasmus Kuhr Jensen [ctb],
Lars Klingen Gjærde [ctb]

Repository CRAN

Date/Publication 2023-07-03 06:40:02 UTC

Contents

`AIC.cm` *Akaike information criterion*

Description

This function computes the AIC for a fitted parametric cure model.

Usage

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

Arguments

`object`	Object of class cm or gfcm.
`...`	Additionally arguments. Not currently used
`k`	Number to control if either AIC or BIC is to be computed (default is 2 equal to AIC).

`AIC.gfcm` *Akaike information criterion*

Description

This function computes the AIC for a fitted parametric cure model.

Usage

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

Arguments

`object`	Object of class cm or gfcm.
`...`	Additionally arguments. Not currently used
`k`	Number to control if either AIC or BIC is to be computed (default is 2 equal to AIC).

`BIC.cm` *Bayesian information criterion*

Description

This function computes the BIC for a fitted parametric cure model.

Usage

```
## S3 method for class 'cm'
BIC(object, ...)
```

Arguments

`object`	Object of class cm or gfcm.
`...`	Some methods for this generic function may take additional, optional arguments. At present none do.

`BIC.gfcm` *Bayesian information criterion*

Description

This function computes the BIC for a fitted parametric cure model.

Usage

```
## S3 method for class 'gfcm'
BIC(object, ...)
```

Arguments

`object`	Object of class cm or gfcm.
`...`	Some methods for this generic function may take additional, optional arguments. At present none do.

`bsx` *Polynomial B-splines with eXtensions*

Description

Generate the B-spline basis matrix for a polynomial spline with derivative restrictions at the boundary knots.

Usage

```
bsx(
  x,
  df = NULL,
  knots = NULL,
  degree = 3,
  intercept = FALSE,
  Boundary.knots = range(x),
  deriv = NULL
)
```

Arguments

`x`	the predictor variable. Missing values are allowed.
`df`	degrees of freedom; one can specify `df` rather than knots; `bs()` then chooses `df-degree` (minus one if there is an intercept) knots at suitable quantiles of `x` (which will ignore missing values). The default, `NULL`, corresponds to no inner knots, i.e., `degree-intercept`.

`knots`	the internal breakpoints that define the spline. The default is `NULL`, which results in a basis for ordinary polynomial regression. Typical values are the mean or median for one knot, quantiles for more knots. See also `Boundary.knots`.
`degree`	degree of the piecewise polynomial—default is `3` for cubic splines.
`intercept`	if `TRUE`, an intercept is included in the basis; default is `FALSE`.
`Boundary.knots`	boundary points at which to anchor the B-spline basis (default the range of the non-NA data). If both `knots` and `Boundary.knots` are supplied, the basis parameters do not depend on `x`. Data can extend beyond `Boundary.knots`.
`deriv`	an integer vector of length 2 with values between 0 and `degree + 1` giving the derivative constraint order at the left and right boundary knots; an order of 2 constrains the second derivative to zero (f''(x)=0); an order of 1 constrains the first and second derivatives to zero (f'(x)=f''(x)=0); an order of 0 constrains the zero, first and second derivatives to zero (f(x)=f'(x)=f''(x)=0) An order of `degree + 1` computes the basis matrix similarly to `bs`.

Value

A matrix with containing the basis functions evaluated in `x`.

`calc.Crude` *Crude event probabilities*

Description

Function for computing crude event probabilties based on relative survival models.

Usage

```
calc.Crude(
  object,
  newdata = NULL,
  type = c("disease", "other", "condother"),
  time = NULL,
  tau = 100,
  reverse = FALSE,
  var.type = c("ci", "se", "n"),
  exp.fun = NULL,
  ratetable = cuRe::survexp.dk,
  rmap,
  scale = ayear,
  smooth.exp = FALSE,
  pars = NULL,
  link = "loglog",
  n = 100
)
```

Arguments

`object`	Fitted model to do predictions from. Possible classes are `gfcm`, `cm`, `stpm2`, and `pstpm2`.
`newdata`	Data frame from which to compute predictions. If empty, predictions are made on the the data which the model was fitted on.
`type`	Probability to compute. Possible values are `disease` (default), `other`, and `condother` (see details).
`time`	Time points at which to compute predictions. If empty, a grid of 100 time points between 0 and `tau` is selected.
`tau`	Upper bound of the integral used to compute the probability of disease-related death (see details). The argument is only used for `type = condother`. Default is 100.
`reverse`	Logical. If `TRUE`, 1 - probability is provided (default is `FALSE`). Only applicable for `type = condother`.
`var.type`	Character. Possible values are `"ci"` (default) for confidence intervals, `"se"` for standard errors, and `"n"` for neither.
`exp.fun`	Object of class `list` containing functions for the expected survival of each row in `newdata`. If not specified, the function computes the expected survival based on `newdata` using the `survival::survexp` function. If `newdata` is not provided, the expected survival is based on the data which the model was fitted on.
`ratetable`	Object of class `ratetable` used to compute the general population survival. Default is `survexp.dk`.
`rmap`	List to be passed to `survexp` from the `survival` package if `exp.fun = NULL`. Detailed documentation on this argument can be found by `?survexp`.
`scale`	Numeric. Passed to the `survival::survexp` function and defaults to 365.24. That is, the time scale is assumed to be in years.
`smooth.exp`	Logical. If `TRUE`, the general population survival function is smoothed by the function `smooth.spline` using the the argument `all.knots = TRUE`.
`pars`	A vector of parameter values for the model given in `object`. Currently not used.
`link`	Link function for computing variance in order to restrict confidence intervals to [0, 1]. Default is `loglog`.
`n`	Number of knots used for the Gauss-Legendre quadrature.

Details

The function estimates crude probabilities by using the relative survival, expected survival, and the cause-specific hazard function. The crude cumulative incidence of disease-related death (`type = "disease"`) is

$$P(T \leq t, D = disease) = \int_0^t S^*(u)R(u)\lambda(u)du.$$

The crude cumulative incidence of death from other causes (`type = "other"`) is

$$P(T \leq t, D = other) = \int_0^t S^*(u)R(u)h^*(u)du.$$

The conditional probability of eventually dying from other causes than disease (`type = "condother"`) is

$$P(D = other|T > t) = \frac{P(D = disease) - P(T \leq t, D = disease)}{P(T > t)}.$$

The probability of disease-related death, P(D = disease), can be computed by using `type = "disease"` and choosing a sufficiently large time point. For P(D = other| T>t), the argument `tau` controls this time point (default is 100).

Value

A list containing the crude probability estimates of each individual in `newdata`.

Examples

```
##Use data cleaned version of the colon disease data from the rstpm2 package
data("colonDC")
set.seed(2)
colonDC <- colonDC[sample(1:nrow(colonDC), 1000), ]

##Extract general population hazards
colonDC$bhaz <- general.haz(time = "FU", rmap = list(age = "agedays", sex = "sex", year= "dx"),
                            data = colonDC, ratetable = survexp.dk)

##Spline-base cure model
#Fit cure model
fit <- rstpm2::stpm2(Surv(FUyear, status) ~ 1, data = colonDC, df = 6,
                     bhazard = colonDC$bhaz, cure = TRUE)

#Compute the probability of disease-related death
res <- calc.Crude(fit, time = seq(0, 20, length.out = 50),
                  rmap = list(age = agedays, sex = sex, year = dx),
                  var.type = "n")
plot(res)

#Compute the conditional probability of dying from other causes than disease
res <- calc.Crude(fit, time = seq(0, 20, length.out = 50), type = "condother",
                  rmap = list(age = agedays, sex = sex, year = dx), var.type = "n")
plot(res)


#Simple parametric cure model
#Fit cure model
fit <- fit.cure.model(Surv(FUyear, status) ~ 1, data = colonDC, bhazard = "bhaz",
                      type = "mixture", dist = "weibull", link = "logit")

#Compute the probability of disease-related death
res <- calc.Crude(fit, time = seq(0, 20, length.out = 50),
                  rmap = list(age = agedays, sex = sex, year = dx),
                  var.type = "n")
plot(res)

#Compute the conditional probability of disease-related death
```

```
res2 <- calc.Crude(fit, time = seq(0, 20, length.out = 50), type = "condother",
                   rmap = list(age = agedays, sex = sex, year = dx), reverse = TRUE,
                   var.type = "n")
plot(res2)
```

`calc.Crude.quantile` *Compute the time to statistical cure using the conditional probability of disease-related death*

Description

The following function estimates the time to statistical cure using the conditional probability of disease-related death.

Usage

```
calc.Crude.quantile(
  fit,
  q = 0.05,
  newdata = NULL,
  max.time = 20,
  exp.fun = NULL,
  var.type = c("ci", "se", "n"),
  rmap,
  ratetable = cuRe::survexp.dk,
  tau = 100,
  reverse = TRUE,
  scale = ayear
)
```

Arguments

`fit`	Fitted model to do predictions from. Possible classes are `gfmc`, `cm`, `stpm2`, and `pstpm2`.
`q`	Threshold to estimate statistical cure according to.
`newdata`	Data frame from which to compute predictions. If empty, predictions are made on the the data which the model was fitted on.
`max.time`	Upper boundary of the interval [0, `max.time`] in which to search for solution (see details). Default is 20.
`exp.fun`	Object of class `list` containing functions for the expected survival of each row in `newdata`. If not specified, the function computes the expected survival based on `newdata` using the `survival::survexp` function. If `newdata` is not provided, the expected survival is based on the data which the model was fitted on.
`var.type`	Character. Possible values are "`ci`" (default) for confidence intervals, "`se`" for standard errors, and "`n`" for neither.

`rmap`	List to be passed to `survexp` from the `survival` package if `exp.fun = NULL`. Detailed documentation on this argument can be found by `?survexp`.
`ratetable`	Object of class `ratetable` used to compute the general population survival. Default is `survexp.dk`
`tau`	Upper bound of integral (see ?calc.Crude). Default is 100.
`reverse`	Logical passed on to `calc.Crude`. If `TRUE` (default), 1 - probability is provided. Only applicable for `type = condother`.
`scale`	Numeric. Passed to the `survival::survexp` function and defaults to 365.24. That is, the time scale is assumed to be in years.

Details

The cure point is calculated as the time point at which the conditional probability of disease-related death reaches the threshold, `q`. If `q` is not reached within `max.time`, no solution is reported.

Value

The estimated cure point.

Examples

```
##Use data cleaned version of the colon cancer data from the rstpm2 package
data("colonDC")
set.seed(2)
colonDC <- colonDC[sample(1:nrow(colonDC), 500), ]

##Extract general population hazards
colonDC$bhaz <- general.haz(time = "FU", rmap = list(age = "agedays", sex = "sex", year= "dx"),
                            data = colonDC, ratetable = survexp.dk)

#Fit cure model and estimate cure point
fit <- rstpm2::stpm2(Surv(FUyear, status) ~ 1, data = colonDC, df = 6,
                     bhazard = colonDC$bhaz, cure = TRUE)
cp <- calc.Crude.quantile(fit, q = 0.05,
                          rmap = list(age = agedays, sex = sex, year = dx))

#Compare the result with the trajectory of the conditional probability of disease-related death
res <- calc.Crude(fit, type = "condother", time = seq(0, 20, length.out = 100),
                  var.type = "n",
                  rmap = list(age = agedays, sex = sex, year = dx), reverse = TRUE)
plot(res)
abline(h = 0.05, v = cp$Estimate)
```

`calc.cure.quantile` *Compute the time to statistical cure using the conditional probability of cure*

Description

The following function estimates the time to statistical cure using the conditional probability of cure.

Usage

```
calc.cure.quantile(
  fit,
  q = 0.05,
  newdata = NULL,
  max.time = 20,
  var.type = c(”ci”, ”n”),
  reverse = TRUE,
  bdr.knot = NULL
)
```

Arguments

`fit`	Fitted model to do predictions from. Possible classes are `gfcm`, `cm`, and `stpm2`.
`q`	Threshold to estimate statistical cure according to.
`newdata`	Data frame from which to compute predictions. If empty, predictions are made on the the data which the model was fitted on.
`max.time`	Upper boundary of the interval [0, `max.time`] in which to search for solution (see details). Default is 20.
`var.type`	Character. Possible values are "`ci`" (default) for confidence intervals, "`se`" for standard errors, and "`n`" for neither.
`reverse`	Logical. Whether to use the conditional probability of not being cured (default) or the conditional probability of cure.
`bdr.knot`	Time point from which cure is assumed. Only relevant for class `stpm2`.

Details

The cure point is calculated as the time point at which the conditional probability of disease-related death reaches the threshold, `q`. If `q` is not reached within `max.time`, no solution is reported.

Value

The estimated cure point.

Examples

```
##Use data cleaned version of the colon cancer data from the rstpm2 package
data("colonDC")
set.seed(2)
colonDC <- colonDC[sample(1:nrow(colonDC), 1000), ]

##Extract general population hazards
colonDC$bhaz <- general.haz(time = "FU", rmap = list(age = "agedays", sex = "sex", year= "dx"),
                            data = colonDC, ratetable = survexp.dk)

#Fit cure model and estimate cure point
fit <- GenFlexCureModel(Surv(FUyear, status) ~ 1, data = colonDC,
                        df = 5, bhazard = "bhaz")
calc.cure.quantile(fit, q = 0.05)
```

`calc.LL` *Loss of lifetime estimation*

Description

Function for computing mean residual lifetime and loss of lifetime estimates based on relative survival models.

Usage

```
calc.LL(
  object,
  newdata = NULL,
  type = c("ll", "mrl"),
  time = NULL,
  tau = 100,
  var.type = c("ci", "se", "n"),
  exp.fun = NULL,
  ratetable = cuRe::survexp.dk,
  rmap,
  smooth.exp = FALSE,
  scale = ayear,
  pars = NULL,
  n = 100
)
```

Arguments

`object`	Fitted model to do predictions from. Possible classes are `gfcm`, `cm`, `stpm2`, and `pstpm2`.
`newdata`	Data frame from which to compute predictions. If empty, predictions are made on the the data which the model was fitted on.

`type`	Type of life expectation estimate. Possible values are `ll` (default) which gives the loss of lifetime, and `mrl` which gives the mean residual lifetime.
`time`	Time points at which to compute predictions. If empty, a grid of 100 time points between 0 and `tau` is selected.
`tau`	The upper limit of the integral (see details). Default is 100.
`var.type`	Character. Possible values are "`ci`" (default) for confidence intervals, "`se`" for standard errors, and "`n`" for neither.
`exp.fun`	Object of class `list` containing functions for the expected survival of each row in `newdata`. If not specified, the function computes the expected survival based on `newdata` using the `survival::survexp` function. If `newdata` is not provided, the expected survival is based on the data which the model was fitted on.
`ratetable`	Object of class `ratetable` used to compute the general population survival. Default is `survexp.dk`.
`rmap`	List to be passed to `survexp` from the `survival` package if `exp.fun = NULL`. Detailed documentation on this argument can be found by `?survexp`.
`smooth.exp`	Logical. If `TRUE`, the general population survival function is smoothed by the function `smooth.spline` using the the argument `all.knots = TRUE`.
`scale`	Numeric. Passed to the `survival::survexp` function and defaults to 365.24. That is, the time scale is assumed to be in years.
`pars`	A vector of parameter values for the model given in `object`. Currently not used.
`n`	Number of knots used for the Gauss-Legendre quadrature.

Details

The mean residual lifetime function and loss of lifetime function are based on numerical integration of the area under the observed and expected conditional survival functions. If `type = "ll"`, the function computes

$$\frac{\int_t^\infty S^*(u)}{S^*(t)} - \frac{\int_t^\infty S(u)}{S(t)}.$$

If `type = "mrl"`, the function computes

$$\frac{\int_t^\infty S(u)}{S(t)},$$

for a given t. The function $S^*(t)$ is the general population survival function and $S(t)$ is the observed survival function. Integration to infinity is not required in studies of human mortality, so an upper limit, `tau`, is chosen instead. As most humans die before they 100 years, this is the default setting of the function. The integral is computed by Gauss-Legendre quadrature and the point wise variance is estimated using the delta method and numerical differentiation.

Value

An object of class `le` containing the life expectancy estimates of each individual in `newdata`.

Examples

```
##Use data cleaned version of the colon cancer data from the rstpm2 package
data("colonDC")
set.seed(2)
colonDC <- colonDC[sample(1:nrow(colonDC), 1000), ]

##Extract general population hazards
colonDC$bhaz <- general.haz(time = "FU", rmap = list(age = "agedays", sex = "sex", year= "dx"),
                            data = colonDC, ratetable = survexp.dk)

##Spline-base cure model
#Fit cure model
fit <- rstpm2::stpm2(Surv(FUyear, status) ~ 1, data = colonDC,
                     df = 6, bhazard = colonDC$bhaz, cure = TRUE)

#Compute and plot the loss of lifetime function
res <- calc.LL(fit, time = seq(0, 20, length.out = 50),
               rmap = list(age = agedays, sex = sex, year = dx),
               var.type = "n")
plot(res)

#Compute and plot the mean residual lifetime
res <- calc.LL(fit, time = seq(0, 20, length.out = 50), type = "mrl",
               rmap = list(age = agedays, sex = sex, year = dx),
               var.type = "n")
plot(res)


#Simple parametric cure model
#Fit cure model
fit <- fit.cure.model(Surv(FUyear, status) ~ 1, data = colonDC, bhazard = "bhaz",
                      type = "mixture", dist = "weibull", link = "logit")

#Compute and plot the loss of lifetime function
res <- calc.LL(fit, time = seq(0, 20, length.out = 50),
               rmap = list(age = agedays, sex = sex, year = dx),
               var.type = "n")
plot(res)

#Compute and plot the mean residual lifetime
res <- calc.LL(fit, time = seq(0, 20, length.out = 50), type = "mrl",
               rmap = list(age = agedays, sex = sex, year = dx),
               var.type = "n")
plot(res)
```

`calc.LL.quantile` *Compute the time to statistical cure using the loss of lifetime function*

Description

The following function estimates the time to statistical cure using the loss of lifetime function.

Usage

```
calc.LL.quantile(
  fit,
  q = 1,
  newdata = NULL,
  max.time = 20,
  var.type = c("ci", "n"),
  exp.fun = NULL,
  rmap = NULL,
  ratetable = cuRe::survexp.dk,
  tau = 100,
  type = "ll",
  scale = ayear
)
```

Arguments

`fit`	Fitted model to do predictions from. Possible classes are `gfcm`, `cm`, `stpm2`, and `pstpm2`.
`q`	Threshold to estimate statistical cure according to.
`newdata`	Data frame from which to compute predictions. If empty, predictions are made on the the data which the model was fitted on.
`max.time`	Upper boundary of the interval [0, `max.time`] in which to search for solution (see details). Default is 20.
`var.type`	Character. Possible values are "`ci`" (default) for confidence intervals, "`se`" for standard errors, and "`n`" for neither.
`exp.fun`	Object of class `list` containing functions for the expected survival of each row in `newdata`. If not specified, the function computes the expected survival based on `newdata` using the `survival::survexp` function. If `newdata` is not provided, the expected survival is based on the data which the model was fitted on.
`rmap`	List to be passed to `survexp` from the `survival` package if `exp.fun = NULL`. Detailed documentation on this argument can be found by `?survexp`.
`ratetable`	Object of class `ratetable` used to compute the general population survival. Default is `survexp.dk`
`tau`	Upper bound of integral (see ?calc.LL). Default is 100.
`type`	Type of life expectancy measure. Possible values are `type = "ll"` for the loss of lifetime and `type = "mrl"` for the mean residual lifetime.
`scale`	Numeric. Passed to the `survival::survexp` function and defaults to 365.24. That is, the time scale is assumed to be in years.

Value

The estimated cure point.

Examples

```
##Use data cleaned version of the colon cancer data from the rstpm2 package
data("colonDC")
set.seed(2)
colonDC <- colonDC[sample(1:nrow(colonDC), 1000), ]

##Extract general population hazards
colonDC$bhaz <- general.haz(time = "FU", rmap = list(age = "agedays", sex = "sex", year= "dx"),
                            data = colonDC, ratetable = survexp.dk)

#Fit cure model and estimate cure point
fit <- rstpm2::stpm2(Surv(FUyear, status) ~ 1, data = colonDC, df = 6,
                     bhazard = colonDC$bhaz, cure = TRUE)
calc.LL.quantile(fit, q = 1,
                 rmap = list(age = agedays, sex = sex, year = dx))
```

`cb` *Restricted cubic splines*

Description

Function for computing the basis matrix for restricted cubic splines

Usage

```
cb(x, df = NULL, knots = NULL, ortho = FALSE, R.inv = NULL, intercept = FALSE)
```

Arguments

`x`	Values to evaluate the basis functions in.
`df`	Degrees of freedom. One can supply `df` rather than knots; `cb` then chooses `df` + 1 knots at suitably chosen quantiles of `x` (which will ignore missing values).
`knots`	Chosen knots for the spline.
`ortho`	Logical. If `TRUE` orthogonalization of the basis matrix is carried out.
`R.inv`	Matrix or vector containing the values of the R matrix from the QR decomposition of the basis matrix. This is used for making new predictions based on the initial orthogonalization. Therefore the default is `NULL`.
`intercept`	Logical. If `FALSE`, the intercept of the restricted cubic spline is removed.

Value

A matrix with containing the basis functions evaluated in `x`.

References

Royston P. and Parmar M.K. (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*, 21:15.

`cbc` *Restricted cubic splines with cure*

Description

Function for computing the basis matrix for restricted cubic splines which are constant beyond the last knot

Usage

```
cbc(x, df = NULL, knots = NULL, ortho = FALSE, R.inv = NULL, intercept = FALSE)
```

Arguments

`x`	Values to evaluate the basis functions in.
`df`	Degrees of freedom. One can supply `df` rather than knots; `cbc` then chooses `df + 1` knots at suitably chosen quantiles of `x` (which will ignore missing values) and adds an additional knot at the 95th quantile of `x`.
`knots`	Chosen knots for the spline.
`ortho`	Logical. If `TRUE` orthogonalization of the basis matrix is carried out.
`R.inv`	Matrix or vector containing the values of the R matrix from the QR decomposition of the basis matrix. This is used for making new predictions based on the initial orthogonalization. Therefore the default is `NULL`.
`intercept`	Logical. If `FALSE`, the intercept of the restricted cubic spline is removed.

Value

A matrix with containing the basis functions evaluated in `x`.

References

Andersson T.M.-L., et al. (2011) Estimating and modelling cure in population-based cancer studies within the framework of flexible parametric survival models. *BMC Medical Research Methodology*, 11:96.

`colonDC` *Colon cancer data*

Description

A dataset containing the individual baseline and follow-up data on more than 15,000 colon cancer patient. The data is a cleaned version of the `rstpm2::colon` data set.

Usage

```
data(colonDC)
```

Format

A data frame with 15564 rows and 11 variables

Details

- sex. Either female or male
- age. Diagnostic age
- stage. Clinical stage at diagnosis (either unknown, localised, regional, or distant)
- statusDC. Alive indicator with cause of death
- subsite. Anatomical subsite of tumour (either coecum and ascending, transverse, descending and sigmoid, or other and NOS)
- dx. Date of diagnosis
- exit. Date of study exit
- status. Alive indicator (0 = alive, 1 = dead)
- FU. Follow-up time measured in days
- FUyear. Follow-up time measured in years
- agedays. Diagnostic age in days.

`fit.cure.model` *Parametric cure model*

Description

This function fits parametric cure models using simple parametric distributions.

Usage

```
fit.cure.model(
  formula,
  data,
  formula.surv = NULL,
  type = c("mixture", "nmixture"),
  dist = c("weibull", "exponential", "lognormal", "weiwei", "weiexp", "gmw"),
  link = c("logit", "loglog", "identity", "probit"),
  bhazard = NULL,
  covariance = TRUE,
  link.mix = c("logit", "loglog", "identity", "probit"),
  control = list(maxit = 10000),
  method = "Nelder-Mead",
  init = NULL
)
```

Arguments

`formula`	Formula for modelling the cure proportion. The left hand side has to be of the form `Surv(time, status)`.
`data`	Data frame in which to interpret the variable names in `formula` and `formula.surv`.
`formula.surv`	List of formulas for each parameter in the parametric distribution (see details).
`type`	A character indicating the type of cure model. Possible values are `mixture` (default) and `nmixture`.
`dist`	The parametric distribution of the survival of the uncured.
`link`	Character. Specifies the link function of the cure proportion.
`bhazard`	Background hazard.
`covariance`	Logical. If `TRUE` (default), the covariance matrix is computed.
`link.mix`	Character. Specifies the link function for the mixture parameter in a weibull-weibull mixture model and weibull-exponential model. Only used when `dist = "weiwei"` and `dist = "weiexp"`.
`control`	List of control parameters passed to `optim`.
`method`	Optimization method passed to `optim`.
`init`	Initial values for the maximum likelihood optimization. If not provided, the optimization will start in 0.

Details

If `type = "mixture"`, the function fits the model,

$$S(t|z) = \pi(z) + [1 - \pi(z)]S_u(t|z),$$

and if `type = "nmixture"`, the function fits the model,

$$S(t|z) = \pi(z)^{\widetilde{F}(t)},$$

where z is a vector of covariates. The `formula.surv` argument is used to model $S_u(t)$ (1 - $\widetilde{F}(t)$). It is a `list` of formulas with as many entries as there are parameters in the chosen parametric distribution. If not specified, all formulas are assumed to be `~1`. The ith formula, i.e., `formula.surv[[i]]` refers to θ_i in the below survival functions.

Exponential model:

$$S_u(t) = \exp\left(-t\theta_1\right).$$

Weibull model:

$$S_u(t) = \exp\left(-\theta_1 t^{\theta_2}\right).$$

Log-normal model:

$$S_u(t) = 1 - \Phi\left(\frac{\log(t) - \theta_1}{\theta_2}\right).$$

Weibull-exponential mixture model:

$$S_u(t) = \theta_1 \exp\left(-\theta_2 t^{\theta_3}\right) + (1 - \theta_1)\exp\left(-\theta_4 t\right).$$

Weibull-Weibull mixture model:

$$S_u(t) = \theta_1 \exp\left(-\theta_2 t^{\theta_3}\right) + (1 - \theta_1)\exp\left(-\theta_4 t^{\theta_5}\right).$$

Generalized modified Weibull distribution:

$$S_u(t) = 1 - \left(1 - \exp\left(-\theta_1 t_2^{\theta} \exp(\theta_3 t)\right)\right)_4^{\theta}.$$

In the Weibull-exponential and Weibull-Weibull mixture models, the link function for the mixture component is controlled by `link.mix`. The remaining parameters are modelled using an exponential link function except θ_1 in the log-normal model, which is modelled using the identity. Parameters are not transformed back to the original scale in the outputted object and related `print.cm` and `summary.cm` functions

Value

An object of class `cm` containing the estimated parameters of the cure model. The appropriate link functions taken on π and the θ_i's are linear in the covariates corresponding to their respective parameter estimates.

Examples

```
##Use data cleaned version of the colon cancer data from the rstpm2 package
data("colonDC")
set.seed(2)
colonDC <- colonDC[sample(1:nrow(colonDC), 500), ]

##Extract general population hazards
colonDC$bhaz <- general.haz(time = "FU", rmap = list(age = "agedays", sex = "sex", year= "dx"),
                            data = colonDC, ratetable = survexp.dk)


###Without covariates
##Fit weibull mixture cure model
```

```
fit.wei <- fit.cure.model(Surv(FUyear, status) ~ 1, data = colonDC, bhazard = "bhaz",
                          type = "mixture", dist = "weibull", link = "logit")

##Plot various summaries of the model (see ?predict.cm)
plot(fit.wei)
plot(fit.wei, time = seq(0, 40, length.out = 100))
plot(fit.wei, type = "hazard")
plot(fit.wei, type = "survuncured")
plot(fit.wei, type = "probcure")

#Fit a weibull-weibull mixture cure model
fit.weiwei <- fit.cure.model(Surv(FUyear, status) ~ 1, data = colonDC, bhazard = "bhaz",
                          type = "mixture", dist = "weiwei", link = "logit")

#Compare to the weibull model
plot(fit.wei, ci = FALSE)
plot(fit.weiwei, add = TRUE, col = 2, ci = FALSE)

###With covariates
##Fit weibull mixture cure model with age effect for both components of the Weibull model
fit <- fit.cure.model(Surv(FUyear, status) ~ age, data = colonDC, bhazard = "bhaz",
                      formula.surv = list(~ age, ~ age),
                      type = "mixture", dist = "weibull", link = "logit")

##Plot model for age 50 and 60
plot(fit, newdata = data.frame(age = 60),
     time = seq(0, 15, length.out = 100), ci = FALSE)
plot(fit, newdata = data.frame(age = 50),
     time = seq(0, 15, length.out = 100), ci = FALSE, add = TRUE, col = 2)

plot(fit, newdata = data.frame(age = 60),
     time = seq(0, 15, length.out = 100), ci = FALSE, type = "hazard")
plot(fit, newdata = data.frame(age = 50),
    time = seq(0, 15, length.out = 100), ci = FALSE, type = "hazard", add = TRUE, col = 2)
```

`general.haz` *Extract general population hazard*

Description

Yearly general population hazards matched on age, gender, and calendar year is extracted from a ratetable.

Usage

```
general.haz(
  time,
  rmap,
  data = NULL,
  ratetable = cuRe::survexp.dk,
```

```
  scale = 365.24
)
```

Arguments

`time`	Either a numeric vector of follow-up times (in days) or a character indicating the variable containing the follow-up times in the data.
`rmap`	A named list. The names must match the dimension names of the ratetable input. The elements should be given as vectors containing the relevant variables in the data or a character indicating the name of the variable in the data. Elements in the list named 'age' and 'year' are transformed such that the age and year of the event/censoring time is used. 'age' must be a numeric vector given as age in days or a character indicating the variable containing the patient ages in the data. 'year' must be of class `Date` with the calendar time points or a character indicating the variable containing the calendar times in the data. Other variables should be coded as characters in the data.
`data`	The data from which to extract variables from. If `time`, `age`, `sex`, or `year` are not characters, this will not be used.
`ratetable`	Object of class `ratetable` to extract background hazards from. Defaults to `survexp.dk`.
`scale`	Numeric to adjust the scale of the outputted hazard values. If the ratetable provides daily hazards and `scale = 365.24` (default), the outputted hazard values are yearly.

Value

An object of class `numeric` containing the yearly expected hazards.

Examples

```
##Use data cleaned version of the colon cancer data from the rstpm2 package
data("colonDC")
set.seed(2)
colonDC <- colonDC[sample(1:nrow(colonDC), 1000), ]

##Extract general population hazards
bhaz1 <- general.haz(time = "FU",
                     rmap = list(age = "agedays", sex = "sex", year= "dx"),
                     data = colonDC,
                     ratetable = survexp.dk)

bhaz2 <- general.haz(time = colonDC$FU,
                 rmap = list(age = colonDC$agedays, sex = colonDC$sex, year = colonDC$dx),
                     data = colonDC,
                     ratetable = survexp.dk)

all(bhaz2 == bhaz1)
```

`GenFlexCureModel` *Fit generalized mixture cure model*

Description

The following function fits a generalized mixture or non-mixture cure model using a link function for the cure rate and for the survival of the uncured. For a mixture cure model, the model is specified by

$$S(t|z) = \pi(z) + [1 - \pi(z)]S_u(t|z),$$

where

$$g_1[S_u(t|z)] = \eta_1(t, z) \qquad and \qquad g_2[\pi(z)] = \eta_2(z).$$

The function implements multiple link functions for both g_1 and g_2. The default time-effect is natural cubic splines, but the function allows for the use of other smoothers.

Usage

```
GenFlexCureModel(
  formula,
  data,
  smooth.formula = NULL,
  smooth.args = NULL,
  df = 3,
  tvc = NULL,
  tvc.formula = NULL,
  bhazard = NULL,
  cr.formula = ~1,
  type = "mixture",
  link.type.cr = c("logit", "loglog", "identity", "probit"),
  link.type = c("PH", "PO", "probit"),
  init = NULL,
  baseoff = FALSE,
  timeVar = "",
  time0Var = "",
  covariance = T,
  verbose = T,
  control = list(maxit = 10000),
  method = "Nelder-Mead",
  constraint = TRUE,
  ini.types = c("cure", "flexpara")
)
```

Arguments

`formula`	Formula for modelling the survival of the uncured. Reponse has to be of the form `Surv(time, status)`.

`data`	Data frame in which to interpret the variables names in `formula`, `smooth.formula`, `tvc.formula`, and `cr.formula`.
`smooth.formula`	Formula for describing the time-effect of the survival of the uncured. If `NULL` (default), the function uses the natural cubic splines modelled on the log-time scale.
`smooth.args`	List. Optional arguments to the time-effect of the survival of the uncured (default is `NULL`).
`df`	Integer. Degrees of freedom (default is 3) for the time-effect of the survival of the uncured. Is not used if `smooth.formula` is provided.
`tvc`	Named list of integers. Specifies the degrees of freedom for time-varying covariate effects. For instance, `tvc = list(x = 3)` creates a time-varying spline-effect of the covariate "x" with 3 degrees of freedom using the `rstpm2::nsx` function.
`tvc.formula`	Formula for the time-varying covariate effects. For time-varying effects, a linear term of the covariate has to be included in `formula`.
`bhazard`	Background hazard.
`cr.formula`	Formula for the cure proportion. The left hand side of the formula is not used and should therefore not be specified.
`type`	A character indicating the type of cure model. Possible values are `mixture` for mixture cure models (default) and `nmixture` for non-mixture cure models.
`link.type.cr`	Character providing the link function for the cure proportion. Possible values are `logit` (default), `loglog`, `identity`, and `probit`.
`link.type`	Character providing the link function for the survival of the uncured. Possible values are `PH` for a proportional hazards model (default), `PO` for a proportion odds model, and `probit` for a probit model.
`init`	Initial values for the optimization procedure. If not specified, the function will create initial values internally.
`baseoff`	Logical. If `TRUE`, the time-effect is modelled only using `tvc.formula` rather than merging with `smooth.formula`.
`timeVar`	Optional character giving the name of the variable specifying the time component of the `Surv` object. Should currently not be used.
`time0Var`	Optional character giving the name of the variable specifying the time start time component used for delayed entry. Should currently not be used.
`covariance`	Logical. If `TRUE` (default), the covariance matrix is computed.
`verbose`	Logical. If `TRUE` status messages of the function is outputted.
`control`	Named list with control arguments passed to `optim`.
`method`	Character passed to `optim` indicating the method for optimization. See `?optim` for details.
`constraint`	Logical. Indicates whether non-negativity constraints should be forced upon the hazard of the uncured patients (see details).
`ini.types`	Character vector denoting the executed schemes for computing initial values (see details).

Details

The default smoother is natural cubic splines established by the `rstpm2::nsx` function. Functions such as `ns`, `bs` are readily available for usage. Also the `cb` function in this package can be used. Initial values are calculated by two procedures and the model is fitted under each set of initial values. The model producing the highest likelihood is selected.

Using `link.type = 'PH'`, the link function $g_1(x) = \log(-\log(x))$ is used. Using `link.type = 'PO'`, the link function $g_1(x) = \log(\frac{x}{1-x})$ is used.

If `constraint = TRUE`, a non-negative hazard of the uncured is ensured by a general penalization scheme. If `constraint = FALSE`, penalization is still employed, but on the all-cause hazard instead.

Value

An object of class `gfcm`.

Examples

```
##Use data cleaned version of the colon cancer data from the rstpm2 package
data("colonDC")
set.seed(2)
colonDC <- colonDC[sample(1:nrow(colonDC), 400), ]

##Extract general population hazards
colonDC$bhaz <- general.haz(time = "FU", rmap = list(age = "agedays", sex = "sex", year= "dx"),
                            data = colonDC, ratetable = survexp.dk)

###Without covariates
##Fit mixture cure model
fit <- GenFlexCureModel(Surv(FUyear, status) ~ 1, data = colonDC, df = 4, bhazard = "bhaz")

##Plot model
plot(fit)
plot(fit, time = seq(0.001, 40, length.out = 100))
plot(fit, type = "hazard")
plot(fit, type = "survuncured")
plot(fit, type = "probcure")

##Predict cure proportion
predict(fit, type = "curerate")


##Fit non-mixture cure model
fit <- GenFlexCureModel(Surv(FUyear, status) ~ 1, data = colonDC, df = 4,
                        bhazard = "bhaz", type = "nmixture")

##Plot relative survival
plot(fit)

##Predict cure proportion
```

```
predict(fit, type = "curerate")

###With covariates
##Fit mixture cure model
fit <- GenFlexCureModel(Surv(FUyear, status) ~ sex, data = colonDC, df = 4,
                        bhazard = "bhaz", cr.formula = ~ sex)

##Plot model
plot(fit, newdata = data.frame(sex = factor("female", levels = c("male", "female"))),
     time = seq(0.001, 15, length.out = 100), ci = FALSE)
plot(fit, newdata = data.frame(sex = factor("male", levels = c("male", "female"))),
     time = seq(0.001, 15, length.out = 100), col = 2, ci = FALSE, add = TRUE)


plot(fit, newdata = data.frame(sex = factor("female", levels = c("male", "female"))),
     time = seq(0.001, 15, length.out = 100), ci = FALSE, type = "survuncured")
plot(fit, newdata = data.frame(sex = factor("male", levels = c("male", "female"))),
     time = seq(0.001, 15, length.out = 100), col = 2, ci = FALSE,
     add = TRUE, type = "survuncured")

predict(fit, type = "curerate",
        data.frame(sex = factor(c("male", "female"),
                                levels = c("male", "female"))))


##Fit mixture cure model with time-varying covariates
colonDC$gender <- as.numeric(colonDC$sex) - 1
fit <- GenFlexCureModel(Surv(FUyear, status) ~ gender, data = colonDC, df = 6,
                        bhazard = "bhaz", cr.formula = ~ gender, tvc = list(gender = 2))

##Plot model
plot(fit, newdata = data.frame(gender = 0))
plot(fit, newdata = data.frame(gender = 1), add = TRUE, col = 2)

plot(fit, type = "hazard", newdata = data.frame(gender = 0), ci = FALSE)
plot(fit, type = "hazard", newdata = data.frame(gender = 1),
     add = TRUE, col = 2, ci = FALSE)

#Predict cure proportions for a male and female patients
predict(fit, type = "curerate", newdata = data.frame(gender = 0))
predict(fit, type = "curerate", newdata = data.frame(gender = 1))
```

`lrtest` *Likelihood ratio tests for comparing nested cure models*

Description

Function for doing likelihood ratio tests for class `cuRe`.

Usage

```
lrtest(model1, model2)
```

Arguments

`model1`	A model fitted by the fit.cure.model or the GenFlexCureModel function
`model2`	Another model fitted by the fit.cure.model or the GenFlexCureModel function for comparison.

Details

One of the models must be nested within the other model, and both models must be fitted to the same data set.

Value

An object of class "anova" containing the degrees of freedom and log-likelihoods of the two models, the difference in degrees of freedom, the likelihood ratio Chi-squared statistic (twice the difference in log-likelihoods) , and the p-value for the asymptotic likelihood ratio test.

`lts` *Long term survival predictions*

Description

Function for computing survival estimates using a relative survival model and the expected general population survival.

Usage

```
lts(
  fit,
  type = c("surv", "hazard", "cumhaz", "loghaz", "fail"),
  newdata = NULL,
  time = NULL,
  var.type = c("ci", "se", "n"),
  exp.fun = NULL,
  ratetable = cuRe::survexp.dk,
  rmap,
  scale = 365.24,
  smooth.exp = FALSE,
  link = NULL,
  mean = FALSE
)
```

Arguments

`fit`	Fitted model to do predictions from. Possible classes are `gfcm`, `stpm2`, `pstpm2`, and `cm`.
`type`	Prediction type (see details). The default is `surv`.
`newdata`	Data frame from which to compute predictions. If empty, predictions are made on the the data which the model was fitted on.
`time`	Optional time points at which to compute predictions. If empty, a grid of 100 time points between 0 and the maximum follow-up time is selected.
`var.type`	Character. Possible values are "`ci`" (default) for confidence intervals, "`se`" for standard errors, and "`n`" for neither.
`exp.fun`	Object of class `list` containing functions for the expected survival of each row in `newdata`. If not specified, the function computes the expected survival based on `newdata` using the `survival::survexp` function. If `newdata` is not provided, the expected survival is based on the data which the model was fitted on.
`ratetable`	Object of class `ratetable` used to compute the general population survival. Default is `survexp.dk`.
`rmap`	List to be passed to `survexp` from the `survival` package if `exp.fun = NULL`. Detailed documentation on this argument can be found by `?survexp`.
`scale`	Numeric. Passed to the `survival::survexp` function and defaults to 365.24. That is, the time scale is assumed to be in years.
`smooth.exp`	Logical. If `TRUE`, the general population survival function is smoothed by the function `smooth.spline` using the the argument `all.knots = TRUE`.
`link`	Character, indicating the link function for the variance calculations. Possible values are "`log`", "`cloglog`" for $log(-log(x))$, "`mlog`" for -log(x), and "`I`" for the indentity.
`mean`	Logical. If `TRUE`, the function outputs the average estimate across the rows in `newdata`. If `newdata = NULL`, the argument is not used. The default is `FALSE`.

Details

Possible values for argument `type` are:
`surv`: Survival function computed by $S(t) = R(t)S^*(t)$
`hazard`: Hazard function computed by $h(t) = \lambda(t) + h^*(t)$
`cumhaz`: The cumulative hazard function computed by $H(t) = \Lambda(t) + H^*(t)$
`loghazard`: The log-hazard function computed by $\log(\lambda(t) + h^*(t))$
`fail`: The distribution function computed by $1 - R(t)S^*(t)$

Value

An object of class `lts` containing the predictions of each individual in `newdata`.

Examples

```
##Use data cleaned version of the colon cancer data from the rstpm2 package
data("colonDC")
set.seed(2)
colonDC <- colonDC[sample(1:nrow(colonDC), 1000), ]

##Extract general population hazards
colonDC$bhaz <- general.haz(time = "FU", rmap = list(age = "agedays", sex = "sex", year= "dx"),
                            data = colonDC, ratetable = survexp.dk)

##Fit flexible parametric relative survival model
fit <- stpm2(Surv(FUyear, status) ~ 1, data = colonDC, df = 6, bhazard = colonDC$bhaz)

##Compute survival probabilities from 0 to 20 years
pred <- lts(fit, rmap = list(age = agedays, sex = sex, year = dx))

##Plot the survival function
plot(pred)
```

`plot.crude` *Plot crude event probabilities*

Description

Plot function for the computed crude event probabilties.

Usage

```
## S3 method for class 'crude'
plot(x, ylim = c(0, 1), xlim = NULL, ci = T, col = 1,
        ylab = NULL, xlab = "Time", add = F, ...)
```

Arguments

`x`	Object of class `crude` in which crude probabilities are stored.
`ylim`	Limits of y-axis.
`xlim`	Limits of x-axis.
`ci`	Logical. If `TRUE` (default), confidence intervals are added to the plot.
`col`	Colour of each curve.
`ylab`	Label of the y-axis. If `NULL`, the function uses its default labels depending on `object$type`.
`xlab`	Label of the x-axis (default is "Time").
`add`	Logical indicating wether the cruves should be added to the current plot window (default is `FALSE`).
`...`	Further arguments passed to `plot` and `lines`.

plot.cuRe *Plot function for Flexible mixture cure model*

Description

Plot function associated with the classes `gfcm` and `cm`

Usage

```
## S3 method for class 'cuRe'
plot(x, newdata = NULL,
     type = c("surv", "probcure", "survuncured", "hazarduncured", "cumhazuncured",
     "densityuncured", "failuncured", "oddsuncured", "loghazarduncured", "hazard",
      "density", "fail", "loghazard", "odds", "cumhaz"), time = NULL, xlim = NULL,
        ylim = c(0, 1), xlab = "Time", ylab = NULL, col = 1, ci = T, add = F, ...)
```

Arguments

`x`	Object of class `cuRe`.
`newdata`	Data frame from which to compute predictions. If empty, predictions are made on the the data which the model was fitted on.
`type`	Character. Defines the desired scale to plot. See ?predict.gfcm for possible values.
`time`	Optional time points at which to compute predictions. This argument is not used if type is `curerate`.
`xlim`	Limits of the x-axis.
`ylim`	Limits of the y-axis.
`xlab`	Label of the x-axis. Default is "Time".
`ylab`	Label of the y-axis. If `NULL`, the function uses its default y-labels depending on `object$type` and `object$excess`.
`col`	Colour of each line.
`ci`	Logical. If `TRUE` (default), confidence intervals are added to the plot.
`add`	Loglca. If `TRUE` the curve is added to the existing plot.
`...`	Further arguments passed to `plot` and `lines`.

Value

A plot containing the predictions of each observation in `newdata`.

`plot.le` *Plot the loss of lifetime function*

Description

Plot function for the class `le`

Usage

```
## S3 method for class 'le'
plot(x, ylim = NULL, xlim = NULL, ci = T, col = 1,
        ylab = NULL, xlab = "Time", add = F, ...)
```

Arguments

`x`	Object of class `le`.
`ylim`	Limit of the y-axis.
`xlim`	Limit of x-axis.
`ci`	Logical. If `TRUE` (default), confidence intervals are added to the plot.
`col`	Numeric or character indicating the colours of the curves.
`ylab`	Label to be written on the y-axis.
`xlab`	Label to be written on the x-axis.
`add`	Logical. If `TRUE`, the curve is added to the current plot window.
`...`	Further argument passed to `plot` and `lines`.

`plot.lts` *Plot the long term survival*

Description

Plot function for the class `lts`.

Usage

```
## S3 method for class 'lts'
plot(x, ylim = NULL, xlim = NULL, ci = T, col = 1,
        ylab = NULL, xlab = "Time", add = F, ...)
```

Arguments

`x`	Object of class `lts`.
`ylim`	Limit of the y-axis.
`xlim`	Limit of x-axis.
`ci`	Logical. If `TRUE` (default), confidence intervals are added to the plot.
`col`	Numeric or character indicating the colours of the curves.
`ylab`	Label to be written on the y-axis. If `NULL`, this is based on `type`.
`xlab`	Label to be written on the x-axis.
`add`	Logical indicating whether to add to current plot window (default is FALSE).
`...`	Further argument passed to `plot` and `lines`.

`predict.cm` *Predict function for flexible mixture cure model*

Description

Function for doing predictions for class `cm`.

Usage

```
## S3 method for class 'cm'
predict(object, newdata = NULL,
       type = c("surv", "curerate", "probcure", "survuncured", "hazarduncured",
        "cumhazuncured","densityuncured", "failuncured", "oddsuncured",
        "loghazarduncured","hazard", "density", "fail", "loghazard",
        "odds", "cumhaz"), time = NULL, var.type = c("ci", "se", "n"),
        pars = NULL, link = NULL, keep.attributes = F, ...)
```

Arguments

`object`	Object of class `cm` to do predictions from.
`newdata`	Data frame from which to compute predictions. If empty, predictions are made on the data which the model was fitted on.
`type`	Prediction type (see details). The default is `surv`.
`time`	Optional time points at which to compute predictions. This argument is not used if type is `curerate`.
`var.type`	Character. Possible values are "`ci`" (default) for confidence intervals, "`se`" for standard errors, and "`n`" for neither.
`pars`	Numerical vector containing the parameters values of the model. In general, this argument can be ignored by the user.
`link`	Character, indicating the link function for the variance calculations. Possible values are "`log`", "`cloglog`" for $log(-log(x))$, "`mlog`" for -log(x), and "`I`" for the indentity. If `NULL` (default), the function will determine `link` from `type`.

`keep.attributes`	Logical. If `TRUE`, `newdata` will be added to the attributes of the output.
`...`	Additional arguments. Currently not used.

Details

Possible values for argument `type` are:
`surv`: Survival function
`curerate`: The cure fraction
`probcure`: The conditional probability of being cured
`survuncured`: The survival of the uncured
`hazarduncured`: The hazard function of the uncured
`cumhazuncured`: The cumulative hazard of the uncured
`densityuncured`: The density function of the uncured
`failuncured`: The distribution function of the uncured, i.e., 1 - `survuncured`
`oddsuncured`: Odds of the uncured, i.e., (1 - `survuncured`) / `survuncured`
`loghazarduncured`: The log-hazard of the uncured
`hazard`: The hazard function
`density`: The density function
`fail`: The distribution function
`loghazard`: The log-hazard function
`odds`: The odds, i.e., (1 - `surv`) / `surv`
`cumhaz`: The cumulative hazard function

Value

A list containing the predictions of each individual in `newdata`.

`predict.gfcm` *Predict function for flexible mixture cure model*

Description

Function for doing predictions for class `gfcm`.

Usage

```
## S3 method for class 'gfcm'
predict(object, newdata = NULL,
        type = c("surv", "curerate", "probcure", "survuncured",
        "hazarduncured", "cumhazuncured", "densityuncured",
        "failuncured", "oddsuncured", "loghazarduncured","hazard",
        "density", "fail", "loghazard", "odds", "cumhaz"), indi = TRUE,
        time = NULL, var.type = c("ci", "se", "n"), pars = NULL,
        link = NULL, keep.attributes = FALSE, ...)
```

Arguments

`object`	Object of class `gfcm` to do predictions from.
`newdata`	Data frame from which to compute predictions. If empty, predictions are made on the data which the model was fitted on.
`type`	Prediction type (see details). The default is `surv`.
`indi`	Logical. If `TRUE`, each line in `newdata` is treated as an individual observations. This functionality allows predictions for each observation at more than one time point.
`time`	Optional time points at which to compute predictions. This argument is not used if type is `curerate`.
`var.type`	Character. Possible values are "`ci`" (default) for confidence intervals, "`se`" for standard errors, and "`n`" for neither.
`pars`	Numerical vector containing the parameters values of the model. In general, this argument can be ignored by the user.
`link`	Character, indicating the link function for the variance calculations. Possible values are "`log`", "`cloglog`" for $log(-log(x))$, "`mlog`" for -log(x), and "`I`" for the indentity. If `NULL` (default), the function will determine `link` from `type`.
`keep.attributes`	Logical. If `TRUE`, `newdata` will be added to the attributes of the output.
`...`	Additional arguments. Currently not used.

Details

Possible values for argument `type` are:
`surv`: Survival function
`curerate`: The cure fraction
`probcure`: The conditional probability of being cured
`survuncured`: The survival of the uncured
`hazarduncured`: The hazard function of the uncured
`cumhazuncured`: The cumulative hazard of the uncured
`densityuncured`: The density function of the uncured
`failuncured`: The distribution function of the uncured, i.e., 1 - `survuncured`
`oddsuncured`: Odds of the uncured, i.e., (1 - `survuncured`) / `survuncured`
`loghazarduncured`: The log-hazard of the uncured
`hazard`: The hazard function
`density`: The density function
`fail`: The distribution function
`loghazard`: The log-hazard function
`odds`: The odds, i.e., (1 - `surv`) / `surv`
`cumhaz`: The cumulative hazard function

Value

A list containing the predictions of each individual in `newdata`.

`survexp.dk` *Ratetable of the Danish general population*

Description

Object of class `ratetable` containing the daily hazards in the Danish general population as reported by the Human Mortality Database (www.mortality.org).

Usage

```
survexp.dk
```

Format

An object of class `ratetable` of dimension 111 x 180 x 2.

Details

The ratetable was generated by using the `relsurv::transrate.hmd` function. The data were downloaded on 15-09-2017 seperately for male and female Danish citizens.
The data can be accessed through:
Female: http://www.mortality.org/hmd/DNK/STATS/fltper_1x1.txt
Male: http://www.mortality.org/hmd/DNK/STATS/mltper_1x1.txt

Index