Package 'flexrsurv'

May 8, 2026

Type Package

Title Flexible Relative Survival Analysis

Version 2.0.18

Date 2024-02-01

Author Isabelle Clerc-Urmès [aut],
Michel Grzebyk [aut, cre],
Guy Hédelin [ctb],
CENSUR working survival group [ctb]

Maintainer Michel Grzebyk <michel.grzebyk@inrs.fr>

Description Package for parametric relative survival analyses. It allows to model non-linear and non-proportional effects and both non proportional and non linear effects, using splines (B-spline and truncated power basis), Weighted Cumulative Index of Exposure effect, with correction model for
the life table. Both non proportional and non linear effects are described in
Remontet, L. et al. (2007) <doi:10.1002/sim.2656> and
Mahboubi, A. et al. (2011) <doi:10.1002/sim.4208>.

License GPL (>= 2.0)

Depends survival

Suggests relsurv, mexhaz, ggplot2, date, lubridate

Imports utils, orthogonalsplinebasis, methods, stats, Epi, Formula, formula.tools, splines, statmod, numDeriv, R.utils, Matrix

Encoding UTF-8

NeedsCompilation yes

Repository CRAN

Date/Publication 2024-02-09 16:10:02 UTC

Contents

`flexrsurv-package` *Package for flexible relative survival analyses*

Description

`flexrsurv` is a package for parametric relative survival analyses. The package implements non-linear, non-proportional effects and both non proportional and non linear effects, using splines (B-spline and truncated power basis), Weighted Cumulative Index of Exposure effect, with correction model for the life table. Both non proportional and non linear effects are described in Remontet et al. (2007) doi:10.1002/sim.2656 and Mahboubi et al. (2011) doi:10.1002/sim.4208.

The main function is `flexrsurv()`

Author(s)

Michel Grzebyk and Isabelle Clerc-Urmès, with contributions from the CENSUR working survival group.

Maintainer: <michel.grzebyk@inrs.fr>

References

Mahboubi, A., M. Abrahamowicz, et al. (2011). "Flexible modeling of the effects of continuous prognostic factors in relative survival." Stat Med 30(12): 1351-1365. doi:10.1002/sim.4208

Remontet, L., N. Bossard, et al. (2007). "An overall strategy based on regression models to estimate relative survival and model the effects of prognostic factors in cancer survival studies." Stat Med 26(10): 2214-2228. doi:10.1002/sim.2656

See Also

`flexrsurv`

flexrsurv *Fit Relative Survival Model*

Description

flexrsurv is used to fit relative survival regression model. Time dependent variables, non-proportionnal (time dependent) effects, non-linear effects are implemented using Splines (B-spline and truncated power basis). Simultaneously non linear and non proportional effects are implemented using approaches developed by Remontet et al.(2007) and Mahboubi et al. (2011).

Usage

```
flexrsurv(formula=formula(data),
   data=parent.frame(),
   knots.Bh,
   degree.Bh=3,
   Spline=c("b-spline", "tp-spline", "tpi-spline"),
   log.Bh=FALSE,
   bhlink=c("log", "identity"),
   Min_T=0,
   Max_T=NULL,
   model=c("additive","multiplicative"),
   rate=NULL,
   weights=NULL,
   na.action=NULL,
   int_meth=c("GL", "CAV_SIM", "SIM_3_8", "BOOLE", "BANDS"),
   npoints=20,
   stept=NULL,
   bands=NULL,
   init=NULL,
   initbyglm=TRUE,
   initbands=bands,
   optim.control=list(trace=100, REPORT=1, fnscale=-1, maxit=25),
   optim_meth=c("BFGS", "CG", "Nelder-Mead", "L-BFGS-B", "SANN", "Brent"),
  control.glm=list(epsilon=1e-8, maxit=100, trace=FALSE, epsilon.glm=1e-1, maxit.glm=25),
   vartype =  c("oim", "opg", "none"),
   debug=FALSE
   )


flexrsurv.ll(formula=formula(data),
   data=parent.frame(),
   knots.Bh=NULL,
   degree.Bh=3,
   Spline=c("b-spline", "tp-spline", "tpi-spline"),
   log.Bh=FALSE,
   bhlink=c("log", "identity"),
```

```
  Min_T=0,
  Max_T=NULL,
  model=c("additive","multiplicative"),
  rate=NULL,
  weights=NULL,
  na.action=NULL,
  int_meth=c("GL", "CAV_SIM", "SIM_3_8", "BOOLE", "GLM", "BANDS"),
  npoints=20,
  stept=NULL,
  bands=NULL,
  init=NULL,
  optim.control=list(trace=100, REPORT=1, fnscale=-1, maxit=25),
  optim_meth=c("BFGS", "CG", "Nelder-Mead", "L-BFGS-B", "SANN", "Brent"),
  vartype =  c("oim", "opg", "none"),
  debug=FALSE
  )
```

Arguments

formula	a formula object, with the response on the left of a ~ operator, and the terms on the right. The response must be a survival object as returned by the Surv function.
data	a data.frame in which to interpret the variables named in the formula.
knots.Bh	the internal breakpoints that define the spline used to estimate the baseline hazard. Typical values are the mean or median for one knot, quantiles for more knots.
degree.Bh	degree of the piecewise polynomial of the baseline hazard. Default is 3 for cubic splines.
Spline	a character string specifying the type of spline basis. "b-spline" for B-spline basis, "tp-spline" for truncated power basis and "tpi-spline" for monotone (increasing) truncated power basis.
log.Bh	logical value: if TRUE, an additional basis equal to log(time) is added to the spline bases of time.
bhlink	logical value: if TRUE, log of baseline hazard is modelled, if FALSE, the baseline hazard is out of the log.
Min_T	minimum of time period which is analysed. Default is max(0.0, min(bands)).
Max_T	maximum of time period which is analysed. Default is max(c(bands, timevar))
model	character string specifying the type of model for both non-proportionnal and non linear effects. The model method=="additive" assumes effects as explained in Remontet et al.(2007), the model method=="multiplicative" assumes effects as explained in Mahboubi et al. (2011).
rate	an optional vector of the background rate for a relevant comparative population to be used in the fitting process. Should be a numeric vector (for relative survival model). rate is evaluated in the same way as variables in formula, that is first in data and then in the environment of formula.

weights	an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector. If not null, the total likelihood is the weighted sum of individual likelihood.
na.action	a missing-data filter function, applied to the model.frame, after any subset argument has been used. Default is options()$na.action.
int_meth	character string specifying the the numerical integration method. Possible values are "GL" for Gauss-Legendre quadrature, "CAV_SIM" for Cavalieri-Simpson's rule, "SIM_3_8" for the Simpson's 3/8 rule, "BOOLE" for the Boole's rule, or "BANDS" for the midpoint rule with specified bands.
npoints	number of points used in the Gauss-Legendre quadrature (when int_meth="GL").
stept	scalar value of the time-step in numerical integration. It is required only when int_meth="CAV_SIM" or "SIM_3_8" or "BOOLE". If no value is supplied, Max_T/500 is used.
bands	bands used to split data in the numerical integration when int_meth="BANDS".
init	starting values of the parameters.
initbyglm	a logical value indicating indicating how are found or refined init values. If TRUE, the fitting method described in Remontet et al.(2007) is ued to find or refine starting values. This may speedup the fit. If FALSE, the maximisation of the likelihood starts at values given in init. If init=NULL, the starting values correspond to a constant net hazard equal to the ratio of the number of event over the total number of person-time.
initbands	bands used to split data when initbyglm=TRUE.
optim.control	a list of control parameters passed to the optim() function.
optim_meth	method to be used to optimize the likelihood. See optim.
control.glm	a list of control parameters passed to the glm() function when method="glm".
vartype	character string specifying the type of variance matrix computed by flexrsurv: the inverse of the hessian matrix computed at the MLE estimate (ie. the inverse of the observed information matrix) if vartype="oim", the inverse of the outer product of the gradients if vartype="opg". The variance is not computed when vartype="none".
debug	control the volum of intermediate output

Details

A full description of the additive and the multiplicative both non-linear and non-proportional models is given respectively in Remontet (2007) and Mahboubi (2011).

flexrsurv.ll is the workhorse function: it is not normally called directly.

Value

flexrsurv returns an object of class "flexrsurv". An object of class "flexrsurv" is a list containing at least the following components:

coefficients	a named vector of coefficients
loglik	the log-likelihood

`var`	estimated covariance matrix for the estimated coefficients
`informationMatrix`	estimated information matrix
`bhlink`	the linkk of baseline hazard: if `"identity"` baseline = sum g0_i b_i(t); if `"log"` log(baseline) = sum g0_i b_i(t);
`init`	vector of the starting values supplied
`converged`	logical, Was the optimlizer algorithm judged to have converged?
`linear.predictors`	the linear fit on link scale (not including the baseline hazard term if `bhlink = "identity"`)
`fitted.values`	the estimated value of the hazard rate at each event time, obtained by transforming the linear predictors by the inverse of the link function
`cumulative.hazard`	the estimated value of the cumulative hazard in the time interval
`call`	the matched call
`formula`	the formula supplied
`terms`	the `terms` object used
`data`	the `data` argument
`rate`	the rate vector used
`time`	the time vector used
`workingformula`	the formula used by the fitter
`optim.control`	the value of the `optim.control` argument supplied
`control.glm`	the value of the `control.glm` argument supplied
`method`	the name of the fitter function used

References

Mahboubi, A., M. Abrahamowicz, et al. (2011). "Flexible modeling of the effects of continuous prognostic factors in relative survival." Stat Med 30(12): 1351-1365. doi:10.1002/sim.4208

Remontet, L., N. Bossard, et al. (2007). "An overall strategy based on regression models to estimate relative survival and model the effects of prognostic factors in cancer survival studies." Stat Med 26(10): 2214-2228. doi:10.1002/sim.2656

See Also

`print.flexrsurv`, `summary.flexrsurv`, `logLik.flexrsurv`, `predict.flexrsurv`, `NPH`, `NLL`, and `NPHNLL`.

Examples

```
if (requireNamespace("relsurv", quietly = TRUE)) {
```

```
# data from package relsurv
data(rdata, package="relsurv")

# rate table from package relsurv
data(slopop, package="relsurv")


# get the death rate at event (or end of followup) from slopop for rdata
rdata$iage <- findInterval(rdata$age*365.24+rdata$time, attr(slopop, "cutpoints")[[1]])
rdata$iyear <- findInterval(rdata$year+rdata$time, attr(slopop, "cutpoints")[[2]])
therate <- rep(-1, dim(rdata)[1])
for( i in 1:dim(rdata)[1]){
  therate[i] <- slopop[rdata$iage[i], rdata$iyear[i], rdata$sex[i]]
}

rdata$slorate <- therate

# change sex coding
rdata$sex01 <- rdata$sex -1

# fit a relative survival model with a non linear effect of age
fit <- flexrsurv(Surv(time,cens)~sex01+NLL(age, Knots=60, Degree=3,
                                           Boundary.knots = c(24, 95)),
                 rate=slorate, data=rdata,
                 knots.Bh=1850,  # one interior knot at 5 years
                 degree.Bh=3,
                 Max_T=5400,
                 Spline = "b-spline",
                 initbyglm=TRUE,
                 initbands=seq(0, 5400, 100),
                 int_meth= "BANDS",
                 bands=seq(0, 5400, 50)
                 )
summary(fit)

# fit a relative survival model with a non linear & non proportional effect of age
fit2 <- flexrsurv(Surv(time,cens)~sex01+NPHNLL(age, time, Knots=60,
                                               Degree=3,
                                               Knots.t = 1850, Degree.t = 3),
                 rate=slorate, data=rdata,
                 knots.Bh=1850,  # one interior knot at 5 years
                 degree.Bh=3,
                 Spline = "b-spline",
                 initbyglm=TRUE,
                 int_meth= "BOOLE",
                 step=50
                 )
summary(fit2, correlation=TRUE)

}
```

`flexrsurvclt` *Fit Relative Survival Model and Correct Life Tables*

Description

`flexrsurvclt` is used to fit relative survival regression model. transition package.

Usage

```
flexrsurvclt(formula=formula(data),
formula.table=NULL,
data=parent.frame(),
Id,
baselinehazard=TRUE,
firstWCEIadditive=FALSE,
knots.Bh,
degree.Bh=3,
intercept.Bh=TRUE,
Spline=c("b-spline", "tp-spline", "tpi-spline"),
log.Bh=FALSE,
bhlink=c("log", "identity"),
Min_T=0,
Max_T=NULL,
model=c("additive","multiplicative"),
rate,
logit_start,
logit_end,
logit_start_byperiod = NULL,
logit_end_byperiod = NULL,
weights_byperiod = NULL,
Id_byperiod = NULL,
contrasts.table = NULL,
knots.table=c(-2.5,0,2.5),
degree.table=3,
Spline.table=c("restricted B-splines"),
Spline.CLT=R2bBSplineBasis(knots=c(-2.5,0,2.5), degree=3),
model_correction = c("cohort", "period"),
weights=NULL,
na.action=NULL,
datacontrol=NULL,
Idcontrol,
ratecontrol,
logit_startcontrol,
logit_endcontrol,
logit_start_byperiodcontrol = NULL,
logit_end_byperiodcontrol = NULL,
weights_byperiodcontrol = NULL,
```

```
    Id_byperiodcontrol = NULL,
    weightscontrol=NULL,
    int_meth=c("GL", "CAV_SIM", "SIM_3_8", "BOOLE", "GLM", "BANDS"),
    bands=NULL,
    npoints=20,
    stept=NULL,
    init=NULL,
    initbyglm=TRUE,
    initbands=bands,
    optim.control=list(trace=100, REPORT=1, fnscale=-1, maxit=25),
    optim_meth=c("BFGS", "CG", "Nelder-Mead", "L-BFGS-B", "SANN", "Brent"),
    Coptim.control=list(),
    lower = -Inf,
    upper = Inf,
    control.glm=list(epsilon=1e-8, maxit=100, trace=FALSE,
    epsilon.glm=.1, maxit.glm=25),
    vartype =  c("oim", "opg", "none"),
    varmethod = c("optim", "numDeriv.hessian", "numDeriv.jacobian"),
    numDeriv.method.args=list(eps=5e-7, d=0.001,
    zero.tol=sqrt(.Machine$double.eps/7e-4), r=4, v=2),
    debug=FALSE
       )

    flexrsurvclt.ll(formula=formula(data),
    formula.table=NULL,
    data=parent.frame(),
    Id,
    baselinehazard=TRUE,
    firstWCEIadditive=FALSE,
    knots.Bh,
    degree.Bh=3,
    Spline=c("b-spline", "tp-spline", "tpi-spline"),
    log.Bh=FALSE,
    bhlink=c("log", "identity"),
    intercept.Bh=TRUE,
    Min_T=0,
    Max_T=NULL,
    model=c("additive","multiplicative"),
    rate,
    logit_start,
    logit_end,
    logit_start_byperiod = NULL,
    logit_end_byperiod = NULL,
    weights_byperiod = NULL,
    Id_byperiod = NULL,
    contrasts.table = NULL,
    knots.table=c(-2.5,0,2.5),
    degree.table=3,
```

```
Spline.table=c("restricted B-splines"),
Spline.CLT=R2bBSplineBasis(knots=c(-2.5,0,2.5), degree=3),
model_correction = c("cohort", "period"),
weights=NULL,
na.action=NULL,
datacontrol=NULL,
Idcontrol,
ratecontrol,
logit_startcontrol,
logit_endcontrol,
logit_start_byperiodcontrol = NULL,
logit_end_byperiodcontrol = NULL,
weights_byperiodcontrol = NULL,
Id_byperiodcontrol = NULL,
weightscontrol=NULL,
int_meth=c("GL", "CAV_SIM", "SIM_3_8", "BOOLE", "GLM", "BANDS"),
bands=NULL,
npoints=20,
stept=NULL,
init=NULL,
optim.control=list(trace=100, REPORT=1, fnscale=-1, maxit=25),
Coptim.control= list(),
optim_meth=c("BFGS", "CG", "Nelder-Mead", "L-BFGS-B", "SANN", "Brent"),
lower = -Inf,
upper = Inf,
vartype =  c("oim", "opg", "none"),
varmethod = c("optim", "numDeriv.hessian", "numDeriv.jacobian"),
numDeriv.method.args=list(eps=5e-7, d=0.001,
  zero.tol=sqrt(.Machine$double.eps/7e-4), r=4, v=2),
debug=FALSE
   )
```

Arguments

formula	a formula object, with the response on the left of a ~ operator, and the terms on the right. The response must be a survival object as returned by the Surv function.
formula.table	a formula object, with empty left hand side, and the terms on the right. This is the formula of the proportional part of the correction model for the table table
data	a data.frame in which to interpret the variables named in the formulas.
Id	vector whose unique values defines the Ids of the subjects.
baselinehazard	if FALSE, no baseline hazard in the model
firstWCEIadditive	if TRUE, the first WCEI term in the formula is considered as the baseline
knots.Bh	the internal breakpoints that define the spline used to estimate the baseline hazard. Typical values are the mean or median for one knot, quantiles for more knots.

`degree.Bh` degree of the piecewise polynomial of the baseline hazard. Default is 3 for cubic splines.

`intercept.Bh` TRUE if the first bases is included in the baseline hazard. Default is TRUE.

`Spline` a character string specifying the type of spline basis. "b-spline" for B-spline basis, "tp-spline" for truncated power basis and "tpi-spline" for monotone (increasing) truncated power basis.

`log.Bh` logical value: if TRUE, an additional basis equal to log(time) is added to the spline bases of time.

`bhlink` character string specifying the link function of the baseline hazard: Default is `bhlink="log"` for including the baseline in the exponential; if `bhlink="identity"`, the baseline hazard is out of the exponential.

`Min_T` minimum of time period which is analysed. Default is `max(0.0, min(bands))`.

`Max_T` maximum of time period which is analysed. Default is `max(c(bands, timevar))`

`model` character string specifying the type of model for both non-proportional and non linear effects. The model `method=="additive"` assumes effects as explained in Remontet et al.(2007), the model `method=="multiplicative"` assumes effects as explained in Mahboubi et al. (2011).

`rate` a vector of the background rate for a relevant comparative population to be used in the fitting process. Should be a numeric vector (for relative survival model). `rate` is evaluated in the same way as variables in `formula`, that is first in `data` and then in the environment of `formula`.

`logit_start` a vector of the logit of the cumulative hazard at the start of the interval in the life table. `logit_start` is evaluated in the same way as variables in `formula`, that is first in `data` and then in the environment of `formula`.

`logit_end` a vector of the logit of the cumulative hazard at the end of the interval in the life table. `logit_end` is evaluated in the same way as variables in `formula`, that is first in `data` and then in the environment of `formula`.

`logit_start_byperiod`, `logit_end_byperiod`, `weights_byperiod`, `Id_byperiod`

A REMPLIR

`knots.table` the internal breakpoints on the logit scale that define the knots of the spline used to estimate the correction model of the life table.

`degree.table` degree of the piecewise polynomial of the spline used to estimate the correction model of the life table. Default is 3 for cubic splines.

`contrasts.table`

an optional list. See the `contrasts.arg` of `model.matrix()`.

`Spline.table` a character string specifying the type of spline basis of the the correction model of the life table. In this version, only "restricted B-splines" is available. "restricted B-splines" are B-spline basis with linear extrapolation + 2nd derivative at boundaries == 0.

`Spline.CLT` a S4 object with method deriv() and evaluate(). The spline basis of the correction of the life table can be specified either by the parameters (`knots.table`, `degree.table`) or an S4 object that ca be used for this purpose. IMPORTANT : the coef of the first basis is constraints to one and `evaluate(deriv(spline_B), left_boundary_knots) == 1`

model_correction
: character string specifying A COMPLETER. method_correction="cohort" when the provided logit are those of the survival of individuals; method_correction="period" when the provided logit are those of the survival fuction of age distribution by period.

weights
: an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector. If not null, the total likelihood is the weighted sum of individual likelihood.

na.action
: a missing-data filter function, applied to the model.frame. If NULL, default is options()$na.action.

datacontrol
: a data.frame in which to interpret the variables named in the formula for the control group.

Idcontrol, ratecontrol, logit_startcontrol, logit_endcontrol, weightscontrol
: Id, rate, logit of the cumulative hazard at the start and the end of the intervalle in the life table, and weights for the control group

logit_start_byperiodcontrol, logit_end_byperiodcontrol, weights_byperiodcontrol, Id_byperiodcontrol
: A REMPLIR

int_meth
: character string specifying the the numerical integration method. Possible values are "GL" for Gauss-Legendre method, "CAV_SIM" for Cavalieri-Simpson's rule, "SIM_3_8" for the Simpson's 3/8 rule, "BOOLE" for the Boole's rule, or "BANDS" for the midpoint rule with specified bands.

bands
: bands used to split data in the numerical integration when int_meth="BANDS").

npoints
: number of points used in the numerical integration when int_meth="GL").

stept
: scalar value of the time-step in numerical integration. It is required only when int_meth="CAV_SIM" or "SIM_3_8" or "BOOLE". If no value is supplied, Max_T/500 is used.

init
: starting values of the parameters.

initbyglm
: a logical value indicating indicating how are found or refined init values. If TRUE, the fitting method described in Remontet et al.(2007) is ued to find or refine starting values. This may speedup the fit. If FALSE, the maximisation of the likelihood starts at values given in init. If init=NULL, the starting values correspond to a constant net hazard equal to the ratio of the number of event over the total number of person-time.

initbands
: bands used to split data when initbyglm=TRUE.

optim.control
: a list of control parameters passed to the optim() function.

optim_meth
: method to be used to optimize the likelihood. See optim.

Coptim.control
: a list of control parameters passed to the constrOptim() function See constrOptim.

lower, upper
: Bounds on the variables for the "L-BFGS-B" method, or bounds in which to search for method "Brent". See optim.

control.glm
: a list of control parameters passed to the glm() function when method="glm".

vartype — character string specifying the type of variance matrix computed by flexrsurv: the inverse of the hessian matrix computed at the MLE estimate (ie. the inverse of the observed information matrix) if vartype="oim", the inverse of the outer product of the gradients if vartype="opg". The variance is not computed when vartype="none".

varmethod — character string specifying the method to compute the hessian matrix when vartype="oim". If varmethod="oim", the hessian matrixe is computed by optim. If varmethod="numDeriv.hessian", the hessian matrix is computed by numDeriv:hessian with method="Richardson". If varmethod="numDeriv.jacobian", the hessian matrixe is computed by numDeriv:jacobian with method="Richardson".

numDeriv.method.args — arguments passed to numDeriv:hessian or numDeriv:jacobian when varmethod="numDeriv.hessian" or varmethod="numDeriv.jacobian". Arguments not specified remain with their default values as specified in details. See numDeriv:grad for details about these parameters.

debug — control the volum of intermediate output

Details

A full description of the additive and the multiplicative both non-linear and non-proportional models is given respectively in Remontet (2007) and Mahboubi (2011).

flexrsurv.ll is the workhorse function: it is not normally called directly.

Value

flexrsurv returns an object of class "flexrsurv". An object of class "flexrsurv" is a list containing at least the following components:

coefficients — a named vector of coefficients

loglik — the log-likelihood

var — estimated covariance matrix for the estimated coefficients

informationMatrix — estimated information matrix

bhlink — the linkk of baseline hazard: if "identity" baseline = sum g0_i b_i(t); if "log" log(baseline) = sum g0_i b_i(t);

init — vector of the starting values supplied

converged — logical, Was the optimlizer algorithm judged to have converged?

linear.predictors — the linear fit on link scale (not including the baseline hazard term if bhlink = "identity")

fitted.values — the estimated value of the hazard rate at each event time, obtained by transforming the linear predictors by the inverse of the link function

cumulative.hazard — the estimated value of the cumulative hazard in the time interval

call — the matched call

formula	the formula supplied
terms	the terms object used
data	the data argument
rate	the rate vector used
time	the time vector used
workingformula	the formula used by the fitter
optim.control	the value of the optim.control argument supplied
control.glm	the value of the control.glm argument supplied
method	the name of the fitter function used

References

Mahboubi, A., M. Abrahamowicz, et al. (2011). "Flexible modeling of the effects of continuous prognostic factors in relative survival." Stat Med 30(12): 1351-1365. doi:10.1002/sim.4208

Remontet, L., N. Bossard, et al. (2007). "An overall strategy based on regression models to estimate relative survival and model the effects of prognostic factors in cancer survival studies." Stat Med 26(10): 2214-2228. doi:10.1002/sim.2656

See Also

print.flexrsurv, summary.flexrsurv, logLik.flexrsurv, predict.flexrsurv, NPH, NLL, and NPHNLL.

Examples

```
if (requireNamespace("relsurv", quietly = TRUE) & requireNamespace("date", quietly = TRUE)) {

library(date)
# data from package relsurv
data(rdata, package="relsurv")

# rate table from package relsurv
data(slopop, package="relsurv")

# get the death rate at event (or end of followup) from slopop for rdata
rdata$iage <- findInterval(rdata$age*365.24+rdata$time, attr(slopop, "cutpoints")[[1]])
rdata$iyear <- findInterval(rdata$year+rdata$time, attr(slopop, "cutpoints")[[2]])
therate <- rep(-1, dim(rdata)[1])
for( i in 1:dim(rdata)[1]){
  therate[i] <- slopop[rdata$iage[i], rdata$iyear[i], rdata$sex[i]]
}

rdata$slorate <- therate

# get the logit_start and logit_end
# logit start at age 18
```

```
tmpsurv <- Surv(rep(0, length(rdata$time)), rdata$time, rdata$cens)


HH <- getHazardFromTable(tmpsurv, startdate=rdata$year,
       startage=rdata$age*365.25 , matchdata=rdata, ratetable=slopop,
       age="age", year="year",
       rmap=list(sex=sex),
       agemin=18,
     ratename = "poprate", cumrateendname ="cumrateend", cumrateentername ="cumrateenter"
      )

rdata$slorate <- HH$poprate
rdata$logit_start <- log(exp(HH$cumrateenter)-1)
rdata$logit_end <- log(exp(HH$cumrateend)-1)

rdata$Id <- 1:dim(rdata)[1]




# change sex coding
rdata$sex01 <- rdata$sex -1

# fit a relative survival model with a non linear effect of age
#   without correction of life table
#   partial likelihood
fit00 <- flexrsurvclt(Surv(time,cens)~sex01+NLL(age, Knots=60, Degree=3,
                                          Boundary.knots = c(24, 95)),
                      rate=slorate,
      data=rdata,
      knots.Bh=1850,  # one interior knot at 5 years
                 degree.Bh=3,
                 Max_T=5400,
                 Spline = "b-spline",
                 initbyglm=TRUE,
                 initbands=seq(0, 5400, 100),
                 int_meth= "BANDS",
                 bands=seq(0, 5400, 50)
                 )
summary(fit00)

# fit a relative survival model with a non linear effect of age
#   without correction of life table
#   full likelihood
fit0 <- flexrsurvclt(Surv(time,cens)~sex01+NLL(age, Knots=60, Degree=3,
                                          Boundary.knots = c(24, 95)),
                                          rate=slorate,
           logit_start=logit_start,
                    logit_end=logit_end,
data=rdata,
Id=Id,
                 knots.Bh=1850,  # one interior knot at 5 years
```

```
                  degree.Bh=3,
                  Max_T=5400,
                  Spline = "b-spline",
                  initbyglm=TRUE,
                  initbands=seq(0, 5400, 100),
                  int_meth= "BANDS",
                  bands=seq(0, 5400, 50)
                  )
summary(fit0)

# fit a relative survival model with a non linear effect of age
#   with correction of life table
#   full likelihood
fit1 <- flexrsurvclt(Surv(time,cens)~sex01+NLL(age, Knots=60, Degree=3,
                                          Boundary.knots = c(24, 95)),
                  rate=slorate,
                      logit_start=logit_start,
                     logit_end=logit_end,
data=rdata,
Id=Id,
                  knots.Bh=1850,  # one interior knot at 5 years
                  degree.Bh=3,
                  Max_T=5400,
                  Spline = "b-spline",
              Spline.CLT=flexrsurv:::R2bBSplineBasis(knots=c(-2.5,0,2.5), degree=3),
                  initbyglm=TRUE,
                  initbands=seq(0, 5400, 100),
                  int_meth= "BANDS",
                  bands=seq(0, 5400, 50)
                  )
summary(fit1)

print(coef(fit1))

# fit a relative survival model with a non linear effect of age
#   with correction of life table, strabified by sex
#   full likelihood
fit2 <- flexrsurvclt(Surv(time,cens)~sex01+NLL(age, Knots=60, Degree=3,
                                          Boundary.knots = c(24, 95)),
    formula.table= ~sex,
                  rate=slorate,
                      logit_start=logit_start,
                     logit_end=logit_end,
data=rdata,
Id=Id,
                  knots.Bh=1850,  # one interior knot at 5 years
                  degree.Bh=3,
                  Max_T=5400,
                  Spline = "b-spline",
              Spline.CLT=flexrsurv:::R2bBSplineBasis(knots=c(-2.5,0,2.5), degree=3),
                  initbyglm=TRUE,
                  initbands=seq(0, 5400, 100),
                  int_meth= "BANDS",
```

```
                bands=seq(0, 5400, 50)
                )
summary(fit2)

AIC(fit0, fit1, fit2)
}
```

---

getBrassHazardFromTable

Compute expected hazards with respect to a corrected reference life table

---

Description

returns the cumulative hazard and the hazard rate of subjects in a reference life table

Usage

```
getBrassPseudoHazardFromTable(Y, startdate, startage, matchdata = NULL,
        ratetable = survival::survexp.us,
age = 1, year = 2, rmap,
agemin = 16, scale = 365.25,
ratename = "rateend",
cumrateendname = "cumrateend",
cumrateentername = "cumrateenter",
idname = "Id_byperiod",
origin = "01/01/1970", format="%d/%m/%Y",
left.open = FALSE,
SplineBrass=R2bBSplineBasis(knots=c(-2.5,0,2.5), degree=3)*c(1, 0, 1),
verbose=FALSE)
```

Arguments

Y	An object with interval data. It can be an object of class Surv with arguments time, time2 and event or a two-column matrix with starting time in the first column and ending time in the second column.
startdate	a numeric vector such that as.Date(startadate) is interpreted as the date of the start (when Y[,]==0).
startage	a numeric vector of age in days the start (when Y[,]==0).
matchdata	an optional data.frame in which to interpret the additional variables to be mapped to the ratetable variables.
ratetable	an object of class ratetable, ie a table of event rates.
age, year	character values of the names of the age and period variables in the rate table.
rmap	an optional list that maps data set names to the ratetable names. See survexp an example bellow.

agemin	numeric value of the age at which the cumulative hazard starts.
scale	numeric value to scale agemin.
ratename, cumrateendname, cumrateentername, idname	names of the returned variables
origin, format	passed to as.Date
left.open	logical, passed to findInterval
SplineBrass	Spline basis used to transform the rates
verbose	logical, if true the progression of the computation is output.

Details

The cumulative rates are computed using survexp.

Value

A data.frame with 3 columns with the rate at the ending time, the cumulative rate from agemin up to the starting time and upt to the ending time.

See Also

getHazardFromTable for the cumulative hazard and the hazard rate of subjects in a reference life table. survexp

Examples

```
if (requireNamespace("relsurv", quietly = TRUE) & requireNamespace("date", quietly = TRUE)) {

library(date)
# data from package relsurv
data(rdata, package="relsurv")

# rate table from package relsurv
data(slopop, package="relsurv")

tmpsurv <- Surv(rep(0, length(rdata$time)), rdata$time, rdata$cens)

HH <- getPseudoHazardFromTable(tmpsurv, startdate=rdata$year,
               startage=rdata$age*365.24 , matchdata=rdata, ratetable=slopop,
               age="age", year="year",
               rmap=list(sex=sex),
               agemin=18, scale=365.24,
               ratename = "poprate",
               cumrateendname ="cumrateend",
               cumrateentername ="cumrateenter",
               idname="Id_byperiod"
               )
summary(HH)
}
```

getHazardFromTable *computes expected hazards with respect to a reference life table*

Description

returns the cumulative hazard and the hazard rate of subjects in a reference life table

Usage

```
getHazardFromTable(Y, startdate, startage, matchdata = NULL,
        ratetable = survival::survexp.us,
age = 1, year = 2, rmap, agemin = 16, scale = 365.25,
ratename = "rateend",
cumrateendname = "cumrateend",
cumrateentername = "cumrateenter",
origin = "01/01/1970", format = "%d/%m/%Y",
left.open = FALSE, verbose=FALSE)
```

Arguments

Y	An object with interval data. It can be an object of class Surv with arguments time, time2 and event or a two-column matrix with starting time in the first column and ending time in the second column.
startdate	a numeric vector such that as.Date(startadate) is interpreted as the date of the start (when Y[,]==0).
startage	a numeric vector of age in days the start (when Y[,]==0).
matchdata	an optional data.frame in which to interpret the additional variables to be mapped to the ratetable variables.
ratetable	an object of class ratetable, ie a table of event rates.
age, year	character values of the names of the age and period variables in the rate table.
rmap	an optional list that maps data set names to the ratetable names. See survexp an example bellow.
agemin	numeric value of the age at which the cumulative hazard starts.
scale	numeric value to scale agemin.
ratename, cumrateendname, cumrateentername	names of the returned variables
origin, format	arguments passed as.Date
left.open	logical, passed to findInterval
verbose	logical, if true the progression of the computation is output.

Details

The cumulative rates are computed using survexp.

Value

A `data.frame` with 3 columns with the rate at the ending time, the cumulative rate from agemin up to the starting time and upt to the ending time.

See Also

`getPseudoHazardFromTable` for the cumulative hazard in each period of a reference life table. `getBrassHazardFromTable` for the cumulative hazard in a corrected reference life table. `survexp`

Examples

```
if (requireNamespace("relsurv", quietly = TRUE) & requireNamespace("date", quietly = TRUE)) {

library(date)
# data from package relsurv
data(rdata, package="relsurv")

# rate table from package relsurv
data(slopop, package="relsurv")

tmpsurv <- Surv(rep(0, length(rdata$time)), rdata$time, rdata$cens)

HH <- getHazardFromTable(tmpsurv, startdate=rdata$year,
               startage=rdata$age*365.24 , matchdata=rdata, ratetable=slopop,
               age="age", year="year",
               rmap=list(sex=sex),
               agemin=18, scale=365.24,
               ratename = "poprate",
               cumrateendname ="cumrateend",
               cumrateentername ="cumrateenter"
               )
summary(HH)
}
```

`getPseudoHazardFromTable` *Computes expected hazards with respect to a reference life table*

Description

return the cumulative pseudo hazard and the hazard rate of subjects in a reference life table

Usage

```
getPseudoHazardFromTable(Y, startdate, startage, matchdata = NULL,
        ratetable = survival::survexp.us,
age = 1, year = 2, rmap,
```

```
agemin = 16, scale = 365.25,
ratename = "rateend",
cumrateendname = "cumrateend",
cumrateentername = "cumrateenter",
idname = "Id_byperiod",
origin = "01/01/1970", format="%d/%m/%Y",
left.open = FALSE, verbose=FALSE)
```

Arguments

Y	An object with interval data. It can be an object of class Surv with arguments time, time2 and event or a two-column matrix with starting time in the first column and ending time in the second column.
startdate	a numeric vector such that as.Date(startadate) is interpreted as the date of the start (when Y[,]==0).
startage	a numeric vector of age in days the start (when Y[,]==0).
matchdata	an optional data.frame in which to interpret the additional variables to be mapped to the ratetable variables.
ratetable	an object of class ratetable, ie a table of event rates.
age, year	character values of the names of the age and period variables in the rate table.
rmap	an optional list that maps data set names to the ratetable names. See survexp an example bellow.
agemin	numeric value of the age at which the cumulative hazard starts.
scale	numeric value to scale agemin.
ratename, cumrateendname, cumrateentername, idname	names of the returned variables
origin, format	passed to as.Date
left.open	logical, passed to findInterval
verbose	logical, if true the progression of the computation is output.

Details

The cumulative rates are computed using survexp.

Value

A data.frame with 3 columns with the rate at the ending time, the cumulative rate from agemin up to the starting time and upt to the ending time.

See Also

getHazardFromTable for the cumulative hazard and the hazard rate of subjects in a reference life table. survexp

Examples

```
if (requireNamespace("relsurv", quietly = TRUE) & requireNamespace("date", quietly = TRUE)) {

library(date)
# data from package relsurv
data(rdata, package="relsurv")

# rate table from package relsurv
data(slopop, package="relsurv")

tmpsurv <- Surv(rep(0, length(rdata$time)), rdata$time, rdata$cens)

HH <- getPseudoHazardFromTable(tmpsurv, startdate=rdata$year,
               startage=rdata$age*365.24 , matchdata=rdata, ratetable=slopop,
               age="age", year="year",
               rmap=list(sex=sex),
               agemin=18, scale=365.24,
               ratename = "poprate",
               cumrateendname ="cumrateend",
               cumrateentername ="cumrateenter",
               idname="Id_byperiod"
               )
summary(HH)
}
```

---

logLik.flexrsurv *Log-Likelihood and the number of observations for a* flexrsuv *fit.*

Description

Function to extract Log-Likelihood and the number of observations from a flexrsuv or flexrsuvclt fit.

Usage

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

## S3 method for class 'flexrsurv'
nobs(object, ...)
```

Arguments

object	any object of class flexrsuv results of a flexrsurv fit.
...	not used

Value

logLik returns a standard logLik object (see logLik)

nobs returns a single number, normally an integer.

See Also

logLik, nobs.

NLL *Non Log-Linear effect*

Description

Generate the spline basis matrix for non log-linear effect.

Usage

```
NLL(x,
    Spline = c("b-spline", "tp-spline", "tpi-spline"),
    Knots = NULL,
    Degree = 3,
    Intercept = FALSE,
    Boundary.knots = range(x),
    Keep.duplicates = TRUE,
    outer.ok = TRUE,
    ...)
```

Arguments

x	the predictor variable.
Spline	a character string specifying the type of spline basis. "b-spline" for B-spline basis, "tp-spline" for truncated power basis and "tpi-spline" for monotone (increasing) truncated power basis.
Knots	the internal breakpoints that define the spline used to estimate the NLL effect. By default there are none.
Degree	degree of splines which are considered.
Intercept	a logical value indicating whether intercept/first basis of spline should be considered.
Boundary.knots	range of variable which is analysed.
Keep.duplicates	Should duplicate interior knots be kept or removed. Defaults is FALSE, which removes duplicate knots with a warning if duplicate interior knots are found.
outer.ok	logical indicating how are managed x values outside the knots. If FALSE, return NA, if TRUE, return 0 for the corresponding x values.
...	not used

Details

NLL is based on package orthogonalsplinebasis

See Also

NPH and NPHNLL.

NPH *Non Proportional Hazard effect*

Description

Generate the design matrix of spline basis for non proportional effect.

Usage

```
NPH(x,
    timevar,
    Spline = c("b-spline", "tp-spline", "tpi-spline"),
    Knots.t = NULL,
    Degree.t = 3,
    Intercept.t = TRUE,
    Boundary.knots.t = c(0, max(timevar)),
    Keep.duplicates.t = TRUE,
    outer.ok = TRUE,
    ...)
```

Arguments

x	the predictor variable.
timevar	the time variable.
Spline	a character string specifying the type of spline basis. "b-spline" for B-spline basis, "tp-spline" for truncated power basis and "tpi-spline" for monotone (increasing) truncated power basis.
Knots.t	the internal breakpoints that define the spline used to estimate the NPH effect. By default there are none.
Degree.t	degree of splines which are considered.
Intercept.t	a logical value indicating whether intercept/first basis of spline should be considered.
Boundary.knots.t	range of time period which is analysed. By default it is c(0, max(timevar)).
Keep.duplicates.t	Should duplicate interior knots be kept or removed. Defaults is FALSE, which removes duplicate knots with a warning if duplicate interior knots are found.

outer.ok	logical indicating how are managed timevar values outside the knots. If FALSE, return NA, if TRUE, return 0 for the corresponding timevar values.
...	not used

Details

NPH is based on package orthogonalsplinebasis

See Also

NLL, and NPHNLL.

NPHNLL *Non Proportional Hazard and Non Log-Linear effect*

Description

Generate the design matrix of spline basis for both non log-linear and non proportional effect.

Usage

```
NPHNLL(x,
       timevar,
       model = c("additive", "multiplicative"),
       Spline = c("b-spline", "tp-spline", "tpi-spline"),
       Knots = NULL,
       Degree = 3,
       Intercept = FALSE,
       Boundary.knots = range(x),
       Knots.t = NULL,
       Degree.t = 3,
       Intercept.t = (model == "multiplicative"),
       Boundary.knots.t = c(0, max(timevar)),
       outer.ok = TRUE,
       Keep.duplicates = TRUE,
       xdimnames = ":XxXxXXxXxX ",
       tdimnames = ":TtTtTTtTtT ")
```

Arguments

x	the predictor variable.
timevar	the time variable.
model	character string specifying the type of model for both non-proportionnal and non linear effects. The model method=="additive" assumes effects as explained in Remontet et al.(2007), the model method=="multiplicative" assumes effects as explained in Mahboubi et al. (2011).

Spline	a character string specifying the type of spline basis. "b-spline" for B-spline basis, "tp-spline" for truncated power basis and "tpi-spline" for monotone (increasing) truncated power basis.
Knots	the internal breakpoints that define the spline used to estimate the NLL part of effect. By default there are none.
Degree	degree of splines of variable which are considered.
Intercept	a logical value indicating whether intercept/first basis of spline should be considered.
Boundary.knots	range of variable which is analysed.
Knots.t	the internal breakpoints that define the spline used to estimate the NPH part of effect. By default there are none.
Degree.t	degree of splines of time variable which are considered.
Intercept.t	a logical value indicating whether intercept/first basis of spline should be considered.
Boundary.knots.t	range of time period which is analysed.
Keep.duplicates	Should duplicate interior knots be kept or removed. Defaults is FALSE, which removes duplicate knots with a warning if duplicate interior knots are found.
outer.ok	logical indicating how are managed timevar or x values outside the knots. If FALSE, return NA, if TRUE, return 0 for the corresponding timevar or x values.
xdimnames	string to build dimnames of x bases
tdimnames	string to build dimnames of timevar bases

Details

NPHNLL is based on package orthogonalsplinebasis

References

Mahboubi, A., M. Abrahamowicz, et al. (2011). "Flexible modeling of the effects of continuous prognostic factors in relative survival." Stat Med 30(12): 1351-1365. doi:10.1002/sim.4208

Remontet, L., N. Bossard, et al. (2007). "An overall strategy based on regression models to estimate relative survival and model the effects of prognostic factors in cancer survival studies." Stat Med 26(10): 2214-2228. doi:10.1002/sim.2656

See Also

NPH and NLL.

predict.flexrsurv *Predictions for a relative survival model*

Description

Predict linear predictors, hazard and cumulative hazard for a model fitted by flexrsuv

Usage

```
  ## S3 method for class 'flexrsurv'
predict(object, newdata= NULL,
   type = c("lp", "link", "risk", "hazard", "hazardrate",
             "rate", "loghazard", "log", "lograte",
             "cumulative.rate", "cumulative.hazard", "cumulative", "cum",
             "survival", "surv", "netsurv"),
   se.fit=FALSE,
   ci.fit = FALSE,
   level = .95,
   na.action=na.pass, ...)
```

Arguments

object	the results of a flexrsurv fit.
newdata	Optional new data at which to do predictions. If absent predictions are for the data frame used in the original fit.
type	the type of predicted value. Choices are the linear predictor ("lp", "log", "loghazard", "lograte"), the hazard ("rate", "hazard", "hazardrate", "risk") or the cumulative hazard ("cum", "cumulative.hazard", "cumulative").
se.fit	if TRUE, pointwise standard errors are produced for the predictions (not available for cumulative hazard).
ci.fit	if TRUE, confidence intervale are produced for the predictions.
level	Confidence level.
na.action	function determining what should be done with missing values in newdata. The default is to predict NA.
...	For future methods

Details

For cumulative hazard, the cumulative hazard is computed from 0 until the given end time. The cumulative hazard is computed using the same numerical integration method as the one used to fit the model.

Value

a vector or a list containing the predictions (element "fit") and their standard errors (element "se.fit") if the se.fit option is TRUE.

Note

To work correctly, arguments `Boundary.knots` and `Boundary.knots.t` must be included in the call to `NPH()`, `NLL()` and `NPHNLL()` in the formula of `flexrsurv`

See Also

predict, flexrsurv, flexrsurvclt

Examples

```
if (requireNamespace("relsurv", quietly = TRUE)) {

# data from package relsurv
data(rdata, package="relsurv")

# rate table from package relsurv
data(slopop, package="relsurv")


# get the death rate at event (or end of followup) from slopop for rdata
rdata$iage <- findInterval(rdata$age*365.24+rdata$time, attr(slopop, "cutpoints")[[1]])
rdata$iyear <- findInterval(rdata$year+rdata$time, attr(slopop, "cutpoints")[[2]])
therate <- rep(-1, dim(rdata)[1])
for( i in 1:dim(rdata)[1]){
  therate[i] <- slopop[rdata$iage[i], rdata$iyear[i], rdata$sex[i]]
}

rdata$slorate <- therate

# change sex coding
rdata$sex01 <- rdata$sex -1
# centering age
rdata$agec <- rdata$age- 60

# fit a relative survival model with a non linear effect of age
fit <- flexrsurv(Surv(time,cens)~sex01+NLL(age, Knots=60, Degree=3,
                                           Boundary.knots = c(24, 95)),
                 rate=slorate, data=rdata,
                 knots.Bh=1850,  # one interior knot at 5 years
                 degree.Bh=3,
                 Spline = "b-spline",
                 initbyglm=TRUE,
                 int_meth= "BOOLE",
                 step=50
                 )
summary(fit, correlation=TRUE)



newrdata <- rdata
```

```
newrdata$age <- rep(60, length(rdata$age))
newrdata$sex <- factor(newrdata$sex, labels=c("m", "f"))

linpred <- predict(fit, newdata=newrdata, type="lp", ci.fit=TRUE)
predhazard <- predict(fit, newdata=newrdata, type="hazard" , ci.fit=TRUE)
predcumhazard <- predict(fit, newdata=newrdata, type="cum", ci.fit=TRUE)

require(ggplot2)
tmp <- cbind(newrdata, linpred)
glp <- ggplot(tmp, aes(time, colour=sex))
glp + geom_ribbon(aes(ymin = lwr, ymax = upr, fill=sex)) +
   geom_line(aes(y=fit)) +
   scale_fill_manual(values = alpha(c("blue", "red"), .3))


tmp <- cbind(newrdata, predhazard)
glp <- ggplot(tmp, aes(time, colour=sex))
glp + geom_ribbon(aes(ymin = lwr, ymax = upr, fill=sex)) +
   geom_line(aes(y=fit))  +
   scale_fill_manual(values = alpha(c("blue", "red"), .3))


tmp <- cbind(newrdata, predcumhazard)
glp <- ggplot(tmp, aes(time, colour=sex))
glp + geom_ribbon(aes(ymin = lwr, ymax = upr, fill=sex)) +
   geom_line(aes(y=fit)) +
   scale_fill_manual(values = alpha(c("blue", "red"), .3))
}
```

`predictCLT` *Predictions for relational life table model*

Description

Predict the relational model for a life table correction model fitted by `flexrsuvclt` or for specified knots, degree and coefficients

Usage

```
  predictCLT (...)

  ## S3 method for class 'flexrsurvclt'
predictCLT(object, newdata= NULL,
   type = c("clt", "correction"),
   se.fit=FALSE, na.action=na.pass, newcoef = NULL, ...)

   ## Default S3 method:
predictCLT(knots, degree, newdata, newcoef, ...)
```

Arguments

object	the results of a flexrsurvclt fit.
newdata	Optional new vector of logarithm of the cumulative distribution odds (LCDO) at which to do predictions. If absent predictions are for values of the LCDO used in the original fit (logit_end parameter in the call to flexrsuvclt)).
newcoef	Optional new coefficients for which to do predictions. If absent predictions are for the coefficients of the fitted model in object.
type	the type of predicted value. Choices are "clt" or "correction" to compute the corrected logarithm of the cumulative distribution odds.
se.fit	if TRUE, pointwise standard errors are produced for the predictions (not yet implemented).
knots, degree	knots and degree of the relational model.
na.action	function determining what should be done with missing values in newdata. The default is to predict NA.
...	For future methods

Details

predictCLT with knots and degree arguments computes corrected values of .

Value

a vector or a list containing the predicted relational model (element "fit") and their standard errors (element "se.fit") if the se.fit option is TRUE.

See Also

predict.flexrsurvclt, flexrsurv, flexrsurvclt

Examples

```
if (requireNamespace("relsurv", quietly = TRUE) & requireNamespace("date", quietly = TRUE)) {

library(date)
# data from package relsurv
data(rdata, package="relsurv")

class(rdata$year)<-"integer"


# rate table from package relsurv
data(slopop, package="relsurv")

# get the death rate at event (or end of followup) from slopop for rdata
rdata$iage <- findInterval(rdata$age*365.24+rdata$time, attr(slopop, "cutpoints")[[1]])
rdata$iyear <- findInterval(rdata$year+rdata$time, attr(slopop, "cutpoints")[[2]])
```

```
therate <- rep(-1, dim(rdata)[1])
for( i in 1:dim(rdata)[1]){
  therate[i] <- slopop[rdata$iage[i], rdata$iyear[i], rdata$sex[i]]
}

rdata$slorate <- therate

# get the logit_start and logit_end
# logit start at age 18


tmpsurv <- Surv(rep(0, length(rdata$time)), rdata$time, rdata$cens)


HH <- getHazardFromTable(tmpsurv, startdate=rdata$year,
       startage=rdata$age*365.25 , matchdata=rdata, ratetable=slopop,
       age="age", year="year",
       rmap=list(sex=sex),
       agemin=18,
     ratename = "poprate", cumrateendname ="cumrateend", cumrateentername ="cumrateenter"
      )

rdata$slorate <- HH$poprate
rdata$logit_start <- log(exp(HH$cumrateenter)-1)
rdata$logit_end <- log(exp(HH$cumrateend)-1)

rdata$Id <- 1:dim(rdata)[1]

# change sex coding
rdata$sex01 <- rdata$sex -1

# fit a relative survival model with a non linear effect of age
#   with correction of life table
#   full likelihood
fit1 <- flexrsurvclt(Surv(time,cens)~sex01+NLL(age, Knots=60, Degree=3,
                                          Boundary.knots = c(24, 95)),
                rate=slorate,
                    logit_start=logit_start,
                   logit_end=logit_end,
data=rdata,
Id=Id,
                knots.Bh=1850,  # one interior knot at 5 years
                degree.Bh=3,
                Max_T=5400,
                Spline = "b-spline",
             Spline.CLT=flexrsurv:::R2bBSplineBasis(knots=c(-2.5,0,2.5), degree=3),
                initbyglm=TRUE,
                initbands=seq(0, 5400, 100),
                int_meth= "BANDS",
                bands=seq(0, 5400, 50)
                )
```

```
corrected_logit_end <- predictCLT(fit1)



try_logit_end <- predictCLT(knots=c(-2.5,0,2.5), degree=3, newcoef = c(0.5, 2),
newdata = rdata$logit_end  )

plot(rdata$logit_end, corrected_logit_end)
points(rdata$logit_end, try_logit_end, col = 2)


}
```

---

predictSpline *Generic method for prediction of spline function*

Description

Predict a spline function by specifying its type, knots, degree and coefficients

Usage

```
  predictSpline (object, x, ...)

   ## Default S3 method:
predictSpline(object=c("b-spline", "tp-spline"),
x, knots, degree, keep.duplicates = FALSE, coef=1, ... )
```

Arguments

object	the type of spline to be predicted ("b-spline", the default, or "tp-spline")
x	Vector of values at wich to predict the spline function.
knots, degree	knots and degree of the relational model.
keep.duplicates	Should duplicate interior knots be kept or removed. Defaults is FALSE, which removes duplicate knots with a warning if duplicate interior knots are found.
coef	vector of coefficient of the spline function.
...	not used

Details

predictSpline .

Value

A vector the evaluated spline function with same length as x.

See Also

predict.flexrsurvclt, flexrsurv, flexrsurvclt

Examples

```
predspline <- predictSpline("b-spline",
x= seq(from=-3, to = 3, by=.1),
coef = .5 * 1:5,
knots=c(-3,0,3), degree=3)
plot(seq(from=-3, to = 3, by=.1), predspline)
```

---

print.flexrsurv *Print a Short Summary of a Relative Survival Model*

Description

Print number of observations, number of events, the formula, the estimated coeficients and the log likelihood.

Usage

```
  ## S3 method for class 'flexrsurv'
print(x,
      digits = max(3, getOption("digits") - 3), ...)
```

Arguments

x	the result of a call to the flexrsuv function.
digits	the minimum number of significant digits to be printed in values, see print.default.
...	other options

See Also

The default method print.default, and help for the function flexrsurv, flexrsurvclt.

summary.flexrsurv *Summarizing Flexible Relative Survival Model Fits*

Description

summary methods for class flexrsurv. Produces and prints summaries of the results of a fitted Relative Survival Model

Usage

```
## S3 method for class 'flexrsurv'
summary(object, correlation = FALSE, symbolic.cor = FALSE, ...)

## S3 method for class 'summary.flexrsurv'
print(x, digits = max(3L, getOption("digits") - 3L),
      symbolic.cor = x$symbolic.cor,
      signif.stars = getOption("show.signif.stars"), ...)
```

Arguments

object	an object of class "flexrsurv", usually, a result of a call to flexrsurv.
x	an object of class "summary.flexrsurv", usually, a result of a call to summary.flexrsurv.
correlation	logical; if TRUE, the correlation matrix of the estimated parameters is returned and printed.
symbolic.cor	logical. If TRUE, print the correlations in a symbolic form (see symnum) rather than as numbers.
digits	the number of significant digits to use when printing.
signif.stars	logical. If TRUE,'significance stars' are printed for each coefficient.
...	further arguments passed to or from other methods.

Details

print.summary.glm tries to be smart about formatting the coefficients, standard errors, etc. and additionally gives 'significance stars' if signif.stars is TRUE.

Correlations are printed to two decimal places (or symbolically): to see the actual correlations print summary(object)$correlation directly.

The dispersion of a GLM is not used in the fitting process, but it is needed to find standard errors. If dispersion is not supplied or NULL, the dispersion is taken as 1 for the binomial and Poisson families, and otherwise estimated by the residual Chisquared statistic (calculated from cases with non-zero weights) divided by the residual degrees of freedom.

Value

The function summary.flexrsurv computes and returns a list of summary statistics of the fitted flexible relative survival model given in `object`. The returned value is an object of class "`summary.flexrsurv`", which a list with components:

`call`	the "`call`" component from `object`.
`terms`	the "`terms`" component from `object`.
`coefficients`	the matrix of coefficients, standard errors, z-values and p-values.
`cov`	the estimated covariance matrix of the estimated coefficients.
`correlation`	(only if `correlation` is true.) the estimated correlations of the estimated coefficients.
`symbolic.cor`	(only if `correlation` is true.) the value of the argument symbolic.cor.
`loglik`	the "`loglik`" component from object.
`df.residual`	the "`df.residual`" component from object.

See Also

`summary`, `flexrsurv`, `flexrsurvclt`.

Examples

```
if (requireNamespace("relsurv", quietly = TRUE)) {

# data from package relsurv
data(rdata, package="relsurv")
# rate table from package relsurv
data(slopop, package="relsurv")

# get the death rate at event (or end of followup) from slopop for rdata
rdata$iage <- findInterval(rdata$age*365.24+rdata$time, attr(slopop, "cutpoints")[[1]])
rdata$iyear <- findInterval(rdata$year+rdata$time, attr(slopop, "cutpoints")[[2]])
therate <- rep(-1, dim(rdata)[1])
for( i in 1:dim(rdata)[1]){
  therate[i] <- slopop[rdata$iage[i], rdata$iyear[i], rdata$sex[i]]
}

rdata$slorate <- therate


# change sex coding
rdata$sex01 <- rdata$sex -1

# fit a relative survival model with a non linear effetc of age

fit <- flexrsurv(Surv(time,cens)~sex01+NLL(age, Knots=60, Degree=3),
                 rate=slorate, data=rdata,
                 knots.Bh=1850,  # one interior knot at 5 years
                 degree.Bh=3,
```

```
                  Spline = "b-spline",
                  initbyglm=TRUE,
                  initbands=seq(from=0, to=5400, by=200),
                  int_meth= "CAV_SIM",
                  step=50
                  )

summary(fit)
}
```

WCEI *Weighted cumulative exposure index*

Description

Generate the spline basis matrix for Weighted cumulative exposure index.

Usage

```
WCEI(x,
    timevar,
    fromT=0,
    Spline.WCEI=NULL,
    Spline = c("m-spline", "b-spline", "tp-spline", "tpi-spline"),
    Knots.t = NULL,
    Degree.t = 3,
    Intercept.t = TRUE,
    Boundary.knots.t = range(c(timevar, fromT)),
    Keep.duplicates.t = TRUE,
    outer.ok = TRUE,
    ...)
```

Arguments

x	the exposure variable.
timevar	the time variable.
fromT	Time at which starts exposure
Spline.WCEI	a S4 object with method deriv(), evaluate() and predict().
Spline	a character string specifying the type of spline basis. "b-spline" for B-spline basis, "tp-spline" for truncated power basis and "tpi-spline" for monotone (increasing) truncated power basis.
Knots.t	the internal breakpoints that define the spline used to estimate the WCEI effect. By default there are none.

Degree.t	degree of splines which are considered.
Intercept.t	a logical value indicating whether intercept/first basis of spline should be considered.
Boundary.knots.t	range of variable which is analysed.
Keep.duplicates.t	Should duplicate interior knots be kept or removed. Defaults is FALSE, which removes duplicate knots with a warning if duplicate interior knots are found.
outer.ok	logical indicating how are managed x values outside the knots. If FALSE, return NA, if TRUE, return 0 for the corresponding x values.
...	not used

Details

WCEI is based on package orthogonalsplinebasis

See Also

NLL NPH and NPHNLL.

Index