Package 'flexrsurv'

February 19, 2020

Type Package

Title Flexible Relative Survival Analysis

Version 1.4.5

Date 2020-02-03

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 using splines (B-spline and truncated power basis). It also includes both non proportional and non linear effects of
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 methods, survival, stats, matrixcalc, Epi, formula.tools

Suggests relsurv, ggplot2

Imports utils, orthogonalsplinebasis, statmod

Encoding latin1

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-02-19 07:40:02 UTC

R topics documented:

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

Description

`flexrsurv` is a package for parametric relative survival analyses. The package implements non-linear and non-proportional effects using splines (B-spline and truncated power basis). It also includes both non proportional and non linear effects of Remontet et al. (2007) doi: 10.1002/sim.2656 and Mahboubi et al. (2011) doi: 10.1002/sim.4208.

Details

Package:	flexrsurv
Type:	Package
Version:	1.3.7
Date:	2017-01-12
License:	GPL(>)2.0)

The main function is `flexrsurv()`

Author(s)

Michel Grzebyk and Isabelle Clerc-UrmÃ¨s, 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,

NPH, NLL, and NPHNLL.

Examples

```
# 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)
```

`flexrsurv-internal.Rd`

Internal Objects and functions for the **flexrsurv** *package*

Description

These are not to be called by the user.

Author(s)

Michel Grzebyk <michel.grzebyk@inrs.fr>

`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.

NLLbeta *Non Log-Linear effect and non proportional effect*

Description

Internal functions not inteded for users.

Usage

```
NLLbeta(y, 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,
    ...)

NPHalpha(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.
y	the name of variable for which tests NLL effect.
Spline	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.
`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` or `x` values outside the knots. If `FALSE`, return `NA`, if `TRUE`, return `0` for the corresponding `timevar` or `x` values.
`...`	not used

Details

Internal functions.

Value

`NLLbeta(x,y,...)` returns `y * NLL(x,...)`.

`NPH(x,timevar,...)` is equal to `x * NPHalpha(x,timevar,...)`.

See Also

`NPH`, `NLL`, 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, 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).
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

Examples

```
# 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", se.fit=TRUE )
predhazard <- predict(fit, newdata=newrdata, type="hazard" , se.fit=TRUE )
predcumhazard <- predict(fit, newdata=newrdata, type="cum", se.fit=TRUE)


require(ggplot2)
tmp <- cbind(newrdata, linpred)
glp <- ggplot(tmp, aes(time, colour=sex))
glp + geom_ribbon(aes(ymin = fit-2*se.fit, ymax = fit + 2*se.fit, 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 = fit-2*se.fit, ymax = fit + 2*se.fit, 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 = fit-2*se.fit, ymax = fit + 2*se.fit, fill=sex)) +
   geom_line(aes(y=fit)) +
   scale_fill_manual(values = alpha(c("blue", "red"), .3))
```

`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`.

`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.

Examples

```
# 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)
```

Index