Package 'survMisc'

April 15, 2015

Imports graphics, utils, stats, KMsurv, combinat, Hmisc, zoo, km.ci, ggplot2, data.table, gridExtra, rpart, gam

Depends survival

Maintainer Chris Dardis <christopherdardis@gmail.com>

License GPL-2

Title Miscellaneous Functions for Survival Data

Type Package

Author Chris Dardis

Description A collection of functions for analysis of survival data. These extend the methods available in the package survival.

Version 0.4.6

Date 2015-04-01

Collate 'air.R' 'asSurv.R' 'genSurv.R' 'autoplotRpart.R' 'autoplotSurvfit.R' 'autoplotTAP.R' 'btumors.R' 'tne.R' 'sf.R' 'ci.R' 'gastric.R' 'covMatSurv.R' 'comp.R' 'cutp.R' 'd1.R' 'gamTerms.R' 'plotTerm.R' 'dx.R' 'gof.R' 'ic.R' 'local.R' 'lrSS.R' 'meanSurv.R' 'mpip.R' 'multiCoxph.R' 'plotMultiCoxph.R' 'plotSurv.R' 'profLik.R' 'quantile.R' 'rsq.R' 'sig.R' 'survMisc_package.R' 'tableRhs.R' 'tneBMT.R' 'tneKidney.R' 'whas100.R' 'whas500.R'

NeedsCompilation yes

Repository CRAN

Date/Publication 2015-04-15 14:01:05

R topics documented:

`survMisc-package` *Miscellaneous functions for survival analysis.*

Description

Miscellaneous functions for survival analysis.

Package:	survMisc
Type:	Package
Version:	0.4.6
Date:	2014-12-21
License:	GPL (>= 2)
LazyLoad:	yes

A collection of functions for the analysis of survival data. These extend the methods already available in `package:survival`.
The intent is to generate a workspace for some of the common tasks arising in survival analysis.
The package should be regarded as 'in development' until release 1.0, meaning that there may be changes to certain function names and parameters, although I will try to keep this to a minimum.

There are references in many of the functions to the textbook:
Klein J, Moeschberger M 2003. *Survival Analysis*, 2nd edition. New York: Springer. Springer (paywall).
which is herein referred to as **K&M**.

Notes for developers: The package follows the **camelCase** naming convention.
It is recommended that other packages do *not* depend on this until at least version 1.0.

For bug reports, feature requests or suggestions for improvement, please try to submit to github. Otherwise email me at the address below.

Author(s)

Chris Dardis <christopherdardis@gmail.com>

`air` *Environmental data timeseries*

Description

Environmental data timeseries

Format

A `data.frame` with 111 observations (rows) and 4 variables (columns).

Details

Data was taken from an environmental study that measured the four variables on 111 consecutive days.

Columns are:

ozone surface concentration of ozone in New York, in parts per million

radiation solar radiation

temperature observed temperature, in degrees Fahrenheit

wind wind speed, in miles per hour

Source

Chambers J, Hastie T (eds). 1992. *Statistical Models in S*. pg 348. Wadsworth and Brooks, Pacific Grove, CA. Springer (paywall)

See Also

gamTerms

`as.Surv` *Convert* `time` *and* `status` *to a right-censored survival object, ordered by time*

Description

Convert `time` and `status` to a right-censored survival object, ordered by time

Usage

```
as.Surv(ti, st)
```

Arguments

`ti`	Vector of time points
`st`	Vector of status (e.g. death) at time points

Details

A traditional `Surv` object will only allow discrete events, i.e. status $s \in N$. Typically, $s \in \{0, 1\}$. There may be times when allowing non-binary values is of interest, e.g. in constructing *expected* survival.
Caution is required when using this function with functions that assume binary values for `status`.

Value

An object of class `Surv`.

See Also

dx
`?survival::Surv`

Examples

```
c1 <- coxph(formula = Surv(time, status == 2) ~ age + log(bili), data=pbc)
E <- predict(c1, type="expected")
as.Surv(pbc$time, E)
## Not run:
summary(coxph(as.Surv(pbc$time, E) ~ log(pbc$bili)))
### Warning:
### In Surv(stime, sstat) : Invalid status value, converted to NA
## End(Not run)
```

`autoplot.rpart` *Generate a* `ggplot` *for an* `rpart` *object*

Description

Uses `ggplot2` to render a recursive partitioning tree

Usage

```
autoplot(object, ...)

## S3 method for class 'rpart'
autoplot(object, ...,
  title = "Recursive partitioning tree \n Terminal nodes show fitted response",
  titSize = 20, uniform = FALSE, minbranch = 0.3, compress = FALSE,
  nspace, branch = 1, all = TRUE, nodeLabels = NULL, lineSize = 1,
  vArrows = FALSE, nodeSize = 5, nodeColor = "darkblue",
  leaf = c("fitR", "en", "both"), leafSize = 5, leafColor = "darkblue",
  digits = NULL, yNU = 0.02, yND = 0.02, yLD = 0.02)
```

Arguments

`object`	An object of class `rpart`, as returned by `rpart::rpart()`
`...`	Additional arguments (not implemented)
`title`	Title for plot
`titSize`	Title text size
`uniform`	The default is to use a non-uniform spacing *proportional to the error in the fit*. If `uniform=TRUE`, uniform vertical spacing of the nodes is used. This may be less cluttered when fitting a large plot onto a page.
`minbranch`	This parameter is ignored if `uniform=TRUE`. The usual tree shows branch lengths *in proportion to improvement*. Where improvement is minimal, there may be insuffcient room for node labels. `minbranch` sets the minimum length for a branch to `minbranch` times the average branch length.

`compress` If `compress=FALSE` (the default), the leaves (terminal nodes) will be at the horizontal plot co-ordinates of $1 : l$, the number of **l**eaves.
If `compress=TRUE`, the tree is arranged in a more compact form.
(The compaction algorithm assumes `uniform=TRUE`. The result is usually an improvement even when this is not the case.)

`nspace` Applies only when `compress=TRUE`.
The amount of extra space between a node with children and a leaf, as compared to the minimal space between leaves.
The default is the value of `branch`, below.

`branch` Controls the shape of the branches from parent to child node.
Needs to be in the range $0 - 1$.
A value of 1 gives square shouldered branches.
A value of 0 gives 'V' shaped branches, with other values being intermediate.

`all` If `all="FALSE"` only terminal nodes (i.e. leaves) will be labelled

`nodeLabels` These can be used to replace the names of the default labels from the fit.
Should be given in the same order as those names

`lineSize` Line size connecting nodes

`vArrows` If `vArrows=TRUE`, adds vertical arrows for descending lines

`nodeSize` Node text size

`nodeColor` Node text color

`leaf` If `leaf="fitR"` (the default) leaves are labelled with the fitted response. If this is a `factor`, the `labels` of the `factor` are used.
The following apply only when the object is fit with `rpart(..., method="exp")`:

- If `leaf="en"`, the leaves are labelled with number of **e**vents and the **n**umber at risk.
- If `leaf="both"` labels show both fitted responses and number of events/ number at risk.

`leafSize` Leaf (terminal node) text size

`leafColor` Leaf text color

`digits` Number of significant digits for fitted response.
Default is `getOption("digits")`

`yNU` **y** value for **N**odes moved **U**p. Used to prevent text from overlapping.
This multiplier is applied to the difference of the largest and smallest y values plotted.
May need to be increased if larger text sizes are used in labelling nodes or node labels span > 1 line.
Typically is < 0.1.

`yND` **y** value for **N**odes moved **D**own.
As above, but applies to text appearing below the node.

`yLD` **y** value for **L**eaves moved **D**own.
As above, but applies to text appearing below the leaves.

Details

The plot shows a division at each node. This is read as *right=TRUE*.
Thus for a node reading **x > 0.5** the line descending to the right is that where **x > 0.5**.

Value

A `list` with the following elmenents:

plot A plot, as generated by `ggplot`

segments A `data.table` with the co-ordinates used to plot the lines

nodes A `data.table` with the co-ordinates used to plot the nodes. Columns are labelled as follows:

- **x, y** x and y co-ordinates
- **node** Node name
- **n** Number of observations (number at risk) for this node
- **isP** Is node a **p**arent?
- **lC, rC** Left and right **c**hildren of node
- **isL** Is node a **l**eaf (terminal node)?
- **resp** Predicted **resp**onse
- **yNU, yND, yLD** adjusted y values for **n**odes (**u**p and **d**down) and **l**eaves (**d**own)

And where applicable:

- **e** Number of events
- **en** Number of events / number of observations

Author(s)

Chris Dardis. Adapted from work by Andrie de Vries and Terry Therneau.

Examples

```
data("cu.summary", package="rpart")
fit <- rpart::rpart(Price ~ Mileage + Type + Country, cu.summary)
autoplot(fit)
data("stagec", package="rpart")
progstat <- factor(stagec$pgstat, levels = 0:1, labels = c("No", "Prog"))
cfit <- rpart::rpart(progstat ~ age + eet + g2 + grade + gleason + ploidy,
                     data = stagec, method = 'class')
autoplot(cfit)
set.seed(1)
df1 <- genSurvDf(model=FALSE)
r1 <- rpart::rpart(Surv(t1, e) ~ ., data=df1, method="exp")
autoplot(r1, leaf="en", title="Nodes show events / no. at risk")
autoplot(r1, compress=TRUE, branch=0.5, nspace=0.1,
         title="Nodes show events / no. at risk")
### oversize text; need to adjust 'y' values for text to compensate
a1 <- autoplot(r1, compress=TRUE, digits=5,
               nodeSize=10, yNU=0.05, yND=0.03,
               leafSize=10, , yLD=0.08, nodeLabels=seq(17))$plot
### can use expand_limits if text is cut off at margins
a1 + ggplot2::expand_limits(x=c(0.75, 7.25))
```

`autoplot.survfit` *Generate a* `ggplot` *for a* `survfit` *object*

Description

Uses ggplot2 to render a table showing the number of subjects at risk per time period and survival curves (Kaplan-Meier plot) and to render

Usage

```
## S3 method for class 'survfit'
autoplot(object, ..., xLab = "Time", yLab = "Survival",
  title = "Marks show times with censoring", titleSize = 15,
  axisTitleSize = 15, axisLabSize = 10, survLineSize = 0.5,
  type = c("single", "CI", "fill"), palette = c("Dark2", "Set2", "Accent",
  "Paired", "Pastel1", "Pastel2", "Set1", "Set3"), jitter = c("none",
  "noEvents", "all"), censShape = 3, censSize = 5, legend = TRUE,
  legLabs = NULL, legTitle = "Strata", legTitleSize = 10,
  legLabSize = 10, alpha = 0.05, CIline = 10, fillLineSize = 0.05,
  pVal = FALSE, sigP = 1, pX = 0.1, pY = 0.1, timeTicks = c("major",
  "minor"), tabTitle = "Number at risk by time", tabTitleSize = 15,
  tabLabSize = 5, nRiskSize = 5)
```

Arguments

`object`	An object of class `survfit`
`...`	Additional arguments (not implemented)
`xLab`	Label for x axis on survival plot
`yLab`	Label for y axis on survival plot
`title`	Title for survival plot
`titleSize`	Title size for survival plot
`axisTitleSize`	Title size for axes
`axisLabSize`	Title size for label axes
`survLineSize`	Survival line size
`type`	Type of plot. The default, `type="single"`, plots single lines. • If `type="CI"` will add lines indicating confidence intervals (taken from `upper` and `lower` values of `survfit` object). Higher values of `alpha` (transparency) are recommended for this, e.g. `alpha=0.8`. • If `type="fill"` will add filled rectangles from the survival lines to the confidence intervals above.
`palette`	Options are taken from color_brewer. • `palette="Dark2"` (the default) is recommended for `single` or `CI` plots. • `palette="Set2"` is recommended for `fill` plots.

jitter	By default, jitter="none". • If jitter="noEvents", adds some random, positive noise to survival lines with no events (i.e. all observations censored). This will bring them just above 1 on the y-axis, making them easier to see separately. • If jitter="all" add some vertical noise to all survival lines.
legend	If legend=FALSE, no legends will be produced for the plot or table
legLabs	These can be used to replace the names of the strata from the fit. Should be given in the same order as those strata
legTitle	Title for legend
legTitleSize	Title size for legend
legLabSize	Legend labels width and height
alpha	Alpha, transparency of lines indicating confidence intervals or filled rectangles. Should be in range $0 - 1$. Larger values e.g. alpha=0.7 are recommended for confidence intervals
censShape	Shape of marks to indicate censored onservations. Default is 3 which gives vertical ticks. Use censShape=10 for circular marks.
censSize	Size of marks to indicate censored onservations
CIline	Confidence interval line type
fillLineSize	Line size surrouding filled boxes
pVal	If pVal=TRUE, adds p value from log-rank test to plot
sigP	No. of significant digits to display in p value. Typically 1 to 3.
pX	Location of p value on x axis. Should be in the range of $0 - 1$, where value is to be placed relative to the maximum observed time. E.g. pX = 0.5 will place it half-way along x-axis
pY	Location of p value on y axis. Should be in the range of $0 - 1$, as above
timeTicks	Numbers to mark on the survival plot and table. • If timeTicks="major" (the default) only the major x-axis (time) marks from the survival plot are are labelled on the plot and table. • If timeTicks="minor", minor axis marks are labelled instead.
tabTitle	Table title
tabTitleSize	Table title text size
tabLabSize	Table legend text size
nRiskSize	Number at risk - text size

Value

A list of ggplot objects, with elements:

plot	the survial plot
table	the table of events per time

This list has the additional class of tableAndPlot, allowing methods from autoplot.tableAndPlot.

Note

The returned `list` contains standard `ggplot2` objects. These can be modified further, as in the last example, which changes to colors to a user-defined sequence. The default color scheme has been chosen for ease of display and accessibility.

Size arguments are passed to `ggplot2`'s `x=element_text(size=)`.

Author(s)

Chris Dardis. Based on existing work by R. Saccilotto, Abhijit Dasgupta, Gil Tomas and Mark Cowley.

Examples

```
data(kidney, package="KMsurv")
s1 <- survfit(Surv(time, delta) ~ type, data=kidney)
autoplot(s1, type="fill", survLineSize=2)
autoplot(s1, type="CI", pVal=TRUE, pX=0.3,
 legLabs=c("surgical", "percutaneous"),
 title="Time to infection following catheter placement \n
   by type of catheter, for dialysis patients")$plot
s1 <- survfit(Surv(time=time, event=delta) ~ 1, data=kidney)
autoplot(s1, legLabs="")$plot
autoplot(s1, legend=FALSE)$plot
### load all datasets from package:km.ci
d1 <- data(package="km.ci")$results[, "Item"]
data(list=d1, package="km.ci")
(s1 <- survfit(Surv(time, status) ~ 1, data=rectum.dat))
### change confidence intervals to log Equal-Precision confidence bands
suppressWarnings(km.ci::km.ci(s1, method="logep"))
autoplot(s1, type="fill", legend=FALSE)$plot
###
### manually changing the output
###
s1 <- survfit(Surv(time, delta) ~ type, data=kidney)
g1 <- autoplot(s1, type="CI", alpha=0.8, survLineSize=2)$plot
### change default colors
g1 + ggplot2::scale_colour_manual(values=c("red", "blue")) +
    ggplot2::scale_fill_manual(values=c("red", "blue"))
### change limits of y-axis
g1 + ggplot2::scale_y_continuous(limits=c(0, 1))
```

---

autoplot.tableAndPlot *Arrange and plot a survival plot, it's legend and a table.*

Description

Uses `gridExtra::gridArrange` to arrange a plot, it's legend and a table.

Usage

```
## S3 method for class 'tableAndPlot'
autoplot(object, ..., hideTabLeg = TRUE,
  plotHeight = 0.75, tabHeight = 0.25)
```

Arguments

object	An object of class tableAndPlot, as returned by autoplot.survfit
...	Additional arguments (not implemented)
hideTabLeg	Hide table legend. If supTabLeg = TRUE (the default), the table legend will not appear.
plotHeight	Plot height.
tabHeight	Table height.

Details

Arguments to plotHeigth and tabHeight are best specified as fractions adding to 1,
e.g. $0.85 + 0.15 = 1$.

Value

A graph, as plotted by gridExtra::grid.arrange

Note

Other ggplot2 objects may be plotted using this method.
They need to be stored in a list of length 2.
The class of this list should be modified with
class(list1) <- c("tableAndPlot", "list")

Author(s)

Chris Dardis. Based on existing work by R. Saccilotto, Abhijit Dasgupta, Gil Tomas and Mark Cowley.

Examples

```
data(kidney, package="KMsurv")
a1 <- autoplot(survfit(Surv(time, delta) ~ type, data=kidney), type="fill")
autoplot(a1)
a1 <- autoplot(survfit(Surv(time, delta) ~ type, data=kidney), type="fill")
data(bmt, package="KMsurv")
s2 <- survfit(Surv(time=t2, event=d3) ~ group, data=bmt)
autoplot(autoplot(s2))
```

`btumors` *Brain tumors trial data*

Description

Brain tumors trial data

Format

A `data.frame` with 6 rows and 4 columns.

Details

Data from a trial of primary brain tumors performed by the Radiation Therapy Oncology Group in 1978. 272 patients in total were enrolled in a trial comparing chemotherapy to chemotherapy + radiotherapy. Prognostic factors are illustrated.

Columns are:

age Age

- **1** < 40
- **2** 40 - 60
- **3** > 60

nec Necrosis

- **0** absent
- **1** present

n Number of patients

ms Median survival (months)

Source

Schoenfeld D. Sample-size formula for the proportional-hazards regression model. Biometrics 1983 June; 39:499-503. JSTOR

See Also

`lrSS`

ci *Confidence intervals for survival curves*

Description

Confidence intervals for survival curves

Usage

```
ci(x, ...)

## S3 method for class 'survfit'
ci(x, ..., CI = c("0.95", "0.9", "0.99"), how = c("point",
  "nair", "hall"), trans = c("log", "lin", "asin"), tL = NULL, tU = NULL)
```

Arguments

x	An object of class survfit
...	Additional arguments (not implemented)
CI	Confidence intervals. As the function currently relies on lookup tables, currently only 95% (the default), 90% and 99% are supported.
how	Method to use for confidence interval. point (the default) uses pointwise confirence intervals. The alternatives use confidence *bands* (see details).
trans	Transformation to use. The default is trans="log". Also supported are linear and arcsine-square root transformations.
tL	**L**ower time point. Used in construction of confidence bands.
tU	**U**pper time point. Used in construction of confidence bands.

Details

In the equations below

$$\sigma_s^2(t) = \frac{\hat{V}[\hat{S}(t)]}{\hat{S}^2(t)}$$

Where $\hat{S}(t)$ is the Kaplan-Meier survival estimate and $\hat{V}[\hat{S}(t)]$ is Greenwood's estimate of its variance.

The **pointwise** confidence intervals are valid for *individual* times, e.g. median and quantile values. When plotted and joined for multiple points they tend to be narrower than the *bands* described below. Thus they tend to exaggerate the impression of certainty when used to plot confidence intervals for a time range. They should not be interpreted as giving the intervals within which the *entire* survival function lies.

For a given significance level α, they are calculated using the standard normal distribution Z as follows:

- linear
$$\hat{S}(t) \pm Z_{1-\alpha}\sigma(t)\hat{S}(t)$$
- log transform
$$[\hat{S}(t)^{\frac{1}{\theta}}, \hat{S}(t)^{\theta}]$$
where
$$\theta = \exp \frac{Z_{1-\alpha}\sigma(t)}{\log \hat{S}(t)}$$
- arcsine-square root transform
upper:
$$\sin^2(\max[0, \arcsin \sqrt{\hat{S}(t)} - \frac{Z_{1-\alpha}\sigma(t)}{2}\sqrt{\frac{\hat{S}(t)}{1-\hat{S}(t)}}])$$
lower:
$$\sin^2(\min[\frac{\pi}{2}, \arcsin \sqrt{\hat{S}(t)} + \frac{Z_{1-\alpha}\sigma(t)}{2}\sqrt{\frac{\hat{S}(t)}{1-\hat{S}(t)}}])$$

Confidence **bands** give the values within which the survival function falls within a *range* of time-points.

The time range under consideration is given so that $t_l \geq t_{min}$, the minimum or lowest event time and $t_u \leq t_{max}$, the maximum or largest event time.
For a sample size n and $0 < a_l < a_u < 1$:

$$a_l = \frac{n\sigma_s^2(t_l)}{1 + n\sigma_s^2(t_l)}$$

$$a_u = \frac{n\sigma_s^2(t_u)}{1 + n\sigma_s^2(t_u)}$$

For the **Nair** or **equal precision** (**EP**) confidence bands, we begin by obtaining the relevant confidence coefficient c_α. This is obtained from the upper α-th fractile of the random variable

$$U = \sup |W^o(x)\sqrt{[x(1-x)]}|, \quad a_l \leq x \leq a_u$$

Where W^o is a standard Brownian bridge.
The intervals are:

- linear
$$\hat{S}(t) \pm c_\alpha \sigma_s(t)\hat{S}(t)$$
- log transform (the default)
$$[\hat{S}(t)^{\frac{1}{\theta}}, \hat{S}(t)^{\theta}]$$
where
$$\theta = \exp \frac{c_\alpha \sigma_s(t)}{\log \hat{S}(t)}$$

- arcsine-square root transform

  upper:

$$\sin^2(\max[0, \arcsin\sqrt{\hat{S}(t)} - \frac{c_\alpha \sigma_s(t)}{2}\sqrt{\frac{\hat{S}(t)}{1-\hat{S}(t)}}])$$

  lower:

$$\sin^2(\min[\frac{\pi}{2}, \arcsin\sqrt{\hat{S}(t)} + \frac{c_\alpha \sigma_s(t)}{2}\sqrt{\frac{\hat{S}(t)}{1-\hat{S}(t)}}])$$

For the **Hall-Wellner** bands the confidence coefficient k_α is obtained from the upper α-th fractile of a Brownian bridge.
In this case t_l can be $= 0$.
The intervals are:

- linear

$$\hat{S}(t) \pm k_\alpha \frac{1 + n\sigma_s^2(t)}{\sqrt{n}}\hat{S}(t)$$

- log transform

$$[\hat{S}(t)^{\frac{1}{\theta}}, \hat{S}(t)^\theta]$$

  where

$$\theta = \exp\frac{k_\alpha[1 + n\sigma_s^2(t)]}{\sqrt{n}\log\hat{S}(t)}$$

- arcsine-square root transform
  upper:

$$\sin^2(\max[0, \arcsin\sqrt{\hat{S}(t)} - \frac{k_\alpha[1 + n\sigma_s(t)]}{2\sqrt{n}}\sqrt{\frac{\hat{S}(t)}{1-\hat{S}(t)}}])$$

  lower:

$$\sin^2(\min[\frac{\pi}{2}, \arcsin\sqrt{\hat{S}(t)} + \frac{k_\alpha[1 + n\sigma_s^2(t)]}{2\sqrt{n}}\sqrt{\frac{\hat{S}(t)}{1-\hat{S}(t)}}])$$

Value

A `survfit` object. The `upper` and `lower` elements in the list (representing confidence intervals) are modified from the original.
Other elements will also be shortened if the time range under consideration has been reduced from the original.

Note

- For the Nair and Hall-Wellner bands, the function currently relies on the lookup tables in `package:km.ci`.
- Generally, the arcsin-square root transform has the best coverage properties.
- All bands have good coverage properties for samples as small as $n = 20$, except for the **Nair / EP** bands with a linear transformation, which perform poorly when $n < 200$.

Source

The function is loosely based on `km.ci::km.ci`.

References

Nair V, 1984. Confidence bands for survival functions with censored data: a comparative study. *Technometrics*. **26**(3):265-75. JSTOR.

Hall WJ, Wellner JA, 1980. Confidence bands for a survival curve from censored data. *Biometrika*. **67**(1):133-43. JSTOR.

Examples are from: **K&M**. Section 4.4, pg 111.

See Also

`sf`

`quantile`

Examples

```
data(bmt, package="KMsurv")
b1 <- bmt[bmt$group==1, ] # ALL patients
s1 <- survfit(Surv(t2, d3) ~ 1, data=bmt[bmt$group==1, ])
ci(s1, how="nair", trans="lin", tL=100, tU=600)
s2 <- survfit(Surv(t2, d3) ~ group, data=bmt)
ci(s2, CI="0.99", how="point", trans="asin", tL=100, tU=600)
```

---

comp *Compare survival curves*

Description

Compare survival curves

Usage

```
comp(x, ...)

## S3 method for class 'survfit'
comp(x, ..., FHp = 1, FHq = 1, lim = 10000,
  scores = NULL)

## S3 method for class 'coxph'
comp(x, ..., FHp = 1, FHq = 1, scores = NULL,
  lim = 10000)
```

Arguments

x	A survfit or coxph object
...	Additional arguments
FHp	p for Fleming-Harrington test
FHq	q for Fleming-Harrington test
lim	limit used for Renyi tests when generating supremum of absolute value of Brownian motion
scores	scores for tests for trend

Details

The **log-rank** tests are given by the general expression:

$$Q = \sum W_i(e_i - \hat{e}_i)^T \sum W_i \hat{V}_i W_i^{-1} \sum W_i(e_i - \hat{e}_i)$$

Where W is the weight, given below, e is the no. of events, $\hat{e}$ is the no. of expected events for that time and $\hat{V}$ is the variance-covariance matrix given by covMatSurv.

The sum is taken to the largest observed survival time (i.e. censored observations are excluded). If there are K groups, then $K-1$ are selected (arbitrary). Likewise the corresponding variance-covariance matrix is reduced to the appropriate $K-1 \times K-1$ dimensions. Q is distributed as chi-square with $K-1$ degrees of freedom.

For 2 strata this simplifies to:

$$Q = \frac{\sum W_i[e1_i - n1_i(\frac{e_i}{n_i})]}{\sqrt{\sum W_i^2 \frac{n1_i}{n_i}(1 - \frac{n1_i}{n_i})(\frac{n_i - e_i}{n_i - 1})e_i}}$$

Here e and n refer to the no. events and no. at risk overall and $e1$ and $n1$ refer to the no. events and no. at risk in group 1.

The weights are given as follows:

Log-rank weight = 1

Gehan-Breslow generalized Wilcoxon weight = n, the no. at risk

Tarone-Ware weight = $\sqrt{n}$

Peto-Peto weight = $\bar{S}(t)$, a modified estimator of survival function given by

$$\bar{S}(t) = \prod 1 - \frac{e_i}{n_i + 1}$$

modified Peto-Peto (by Andersen) weight = $\bar{S}(t)\frac{n}{n+1}$

Fleming-Harrington weight at $t_0 = 1$ and thereafter is:

$$\hat{S}(t_{i-1})^p[1 - \hat{S}(t_{i-1})^q]$$

Here $\hat{S}$ is the Kaplan-Meier (product-limit) estimator. Note that both p and q need to be ≥ 0

The **Supremum (Renyi)** family of tests are designed to detect differences in survival curves which cross.
That is, an early difference in survival in favor of one group is balanced by a later reversal.
The same weights as above are used.
They are calculated by finding

$$Z(t_i) = \sum_{t_k \le t_i} W(t_k)[e1_k - n1_k \frac{e_k}{n_k}], \quad i = 1, 2, ..., k$$

(which is similar to the numerator used to find Q in the log-rank test for 2 groups above).
and it's variance:

$$\sigma^2(\tau) = \sum_{t_k \le \tau} W(t_k)^2 \frac{n1_k n2_k (n_k - e_k) e_k}{n_k^2 (n_k - 1)}$$

where τ is the largest t where both groups have at least one subject at risk.

Then calculate:

$$Q = \frac{\sup |Z(t)|}{\sigma(\tau)}, \quad t < \tau$$

When the null hypothesis is true, the distribution of Q is approximately

$$Q \sim \sup |B(x)|, \quad 0 \le x \le 1$$

And for a standard Brownian motion (Wiener) process:

$$Pr[\sup |B(t)| > x] = 1 - \frac{4}{\pi} \sum_{k=0}^{\infty} \frac{(-1)^k}{2k+1} \exp \frac{-\pi^2 (2k+1)^2}{8x^2}$$

Tests for trend are designed to detect ordered differences in survival curves.
That is, for at least one group:

$$S_1(t) \ge S_2(t) \ge ... \ge S_K(t) \quad t \le \tau$$

where τ is the largest t where all groups have at least one subject at risk. The null hypothesis is that

$$S_1(t) = S_2(t) = ... = S_K(t) \quad t \le \tau$$

Scores used to construct the test are typically $s = 1, 2, ..., K$, but may be given as a vector representing a numeric characteristic of the group.
They are calculated by finding

$$Z_j(t_i) = \sum_{t_i \le \tau} W(t_i)[e_{ji} - n_{ji} \frac{e_i}{n_i}], \quad j = 1, 2, ..., K$$

The test statistic is

$$Z = \frac{\sum_{j=1}^{K} s_j Z_j(\tau)}{\sqrt{\sum_{j=1}^{K} \sum_{g=1}^{K} s_j s_g \sigma_{jg}}}$$

where σ is the the appropriate element in the variance-covariance matrix (as in `covMatSurv`).
If ordering is present, the statistic Z will be greater than the upper αth percentile of a standard normal distribution.

Value

A `list` with two elements, `tne` and `tests`.
The first is a `data.table` with one row for each time at which an event occurred.
Columns show time, no. at risk and no. events (by stratum and overall).

The second contains the tests, as a `list`.

The first element is the log-rank family of tests.
The following additional tests depend on the no. of strata:
For a `survfit` or a `coxph` object with 2 strata, these are the Supremum (Renyi) family of tests.
For a `survfit` or `coxph` object with at least 3 strata, there are tests for trend.

Note

Regarding the Fleming-Harrington weights:

- $p = q = 0$ gives the log-rank test, i.e. $W = 1$
- $p = 1, q = 0$ gives a version of the Mann-Whitney-Wilcoxon test (tests if populations distributions are identical)
- $p = 0, q > 0$ gives more weight to differences later on
- $p > 0, q = 0$ gives more weight to differences early on

The example using `alloauto` data illustrates this. Here the log-rank statistic has a p-value of around 0.5 as the late advantage of allogenic transplants is offset by the high early mortality. However using Fleming-Harrington weights of $p = 0, q = 0.5$, emphasising differences later in time, gives a p-value of 0.04.

References

Gehan A. A Generalized Wilcoxon Test for Comparing Arbitrarily Singly-Censored Samples. Biometrika 1965 Jun. 52(1/2):203–23. JSTOR

Tarone RE, Ware J 1977 On Distribution-Free Tests for Equality of Survival Distributions. *Biometrika*;**64**(1):156–60. JSTOR

Peto R, Peto J 1972 Asymptotically Efficient Rank Invariant Test Procedures. *J Royal Statistical Society* **135**(2):186–207. JSTOR

Fleming TR, Harrington DP, O'Sullivan M 1987 Supremum Versions of the Log-Rank and Generalized Wilcoxon Statistics. *J American Statistical Association* **82**(397):312–20. JSTOR

Billingsly P 1999 *Convergence of Probability Measures.* New York: John Wiley & Sons. Wiley (paywall)

Examples are from Klein J, Moeschberger M 2003 *Survival Analysis*, 2nd edition. New York: Springer. Examples 7.2, 7.4, 7.5, 7.6, 7.9, pp 210-225.

Examples

```
### 2 curves
data(kidney,package="KMsurv")
s1 <- survfit(Surv(time=time, event=delta) ~ type, data=kidney)
```

```
comp(s1)
### 3 curves
data(bmt, package="KMsurv")
comp(survfit(Surv(time=t2, event=d3) ~ group, data=bmt))
### see effect of F-H test
data(alloauto, package="KMsurv")
s3 <- survfit(Surv(time, delta) ~ type, data=alloauto)
comp(s3, FHp=0, FHq=1)
### see trend tests
data(larynx, package="KMsurv")
s4 <- survfit(Surv(time, delta) ~ stage, data=larynx)
comp(s4)
### Renyi tests
data("gastric", package="survMisc")
s5 <- survfit(Surv(time, event) ~ group, data=gastric)
comp(s5)
c1 <- coxph(Surv(time=time, event=delta) ~ type, data=kidney )
comp(c1)
```

`covMatSurv` *Covariance matrix for survival data*

Description

Gives variance-covariance matrix for comparing survival data for two or more groups.
Inputs are vectors corresponding to observations at a set of discrete time points for right censored data, except for $n1$, the no. at risk by predictor.
This should be specified as a vector for one group, otherwise as a matrix with each column corresponding to a group.

Usage

```
covMatSurv(t, n, e, n1)
```

Arguments

`t`	time
`n`	number at risk
`e`	number of events
`n1`	number at risk (by predictor). If 2 groups, should be given as a `vector` with the number at risk for group 1. If ≥ 2 groups, a `matrix` with one column for each group.

Value

An `array`. The first two dimensions = number of groups. This is the square matrix below.
The third dimension is the number of observations (time points).

Where there are 2 groups, the resulting sparse square matrix (i.e. the non-diagonal elements are 0) at time i has diagonal elements:

$$v_i = -\frac{n_{0i} n_{1i} e_i (n_i - e_i)}{n_i^2 (n_i - 1)}$$

where n_1 is the number at risk in group 1.

For ≥ 2 groups, the resulting square matrix has diagonal elements:

$$v_{kki} = \frac{n_{ki}(n_i - n_{ki}) e_i (n_i - e_i)}{n_i^2 (n_i - 1)}$$

and off diagonal elements:

$$v_{kli} = \frac{-n_{ki} n_{li} e_i (n_i - e_i)}{n_i^2 (n_i - 1)}$$

See Also

Called by `comp`

Examples

```
data(tneKidney)
covMatSurv(t=tneKidney$t, n=tneKidney$n, e=tneKidney$e, n1=tneKidney$n_1)
```

---

cutp *Cutpoint for a continuous variable in a* `coxph` *or* `survfit` *model*

Description

Determine the optimal cutpoint for a continuous variable in a `coxph` or `survfit` model

Usage

```
cutp(x, ...)

## S3 method for class 'coxph'
cutp(x, ..., var = "", plot = FALSE)

## S3 method for class 'survfit'
cutp(x, ..., var = "", plot = FALSE)
```

Arguments

x	A survfit or coxph object
...	Additional arguments. Passed to graphics::plot.
var	Variable to test. Must be continuous (i.e. > 2 unique values)
plot	If plot=TRUE will plot cut points against the test statistic Q.

Details

The statistic is based on the score test from the Cox model. For the cut point μ, of a predictor K, the data is split into two groups, those $\geq \mu$ and those $< \mu$.

The log-rank statistic LR is calculated for each unique element k in K:

$$LR_k = \sum_{i=1}^{D}(e_i^+ - n_i^+ \frac{e_i}{n_i})$$

Where e_i^+ and n_i^+ refer to the number of events and number at risk in those above the cutpoint, respectively.
The sum is taken to across distinct times with observed events, to D, the largest of these.
It is normalized (standardized), in the case of censoring, by finding σ^2 which is:

$$\sigma^2 = \frac{1}{D-1}\sum_{i}^{D}(1 - \sum_{j=1}^{i}\frac{1}{D+1-j})^2$$

The test statistic is then

$$Q = \frac{\max|LR_k|}{\sigma\sqrt{D-1}}$$

Under the null hypothesis that the chosen cut-point does *not* predict survival, the distribution of Q has a limiting distibution which is the supremum of the absolute value of a Brownian bridge:

$$p = Pr(\sup Q \geq q) = 2\sum_{i=1}^{\infty}(-1)^{i+1}\exp(-2i^2q^2)$$

Value

A data.frame with columns:

cp	The **c**ut **p**oint. The optimum value at which to divide the groups into those $\geq$ the cutpoint and those below.
Q	The test statistic
p	p-value

If plot=TRUE a plot of cut points against values of the log-rank test statistic LR.

References

Examples are from Klein J, Moeschberger M 2003 *Survival Analysis*, 2nd edition. New York: Springer. Example 8.3, pp 273-274.

Contal C, O'Quigley J, 1999 An application of changepoint methods in studying the effect of age on survival in breast cancer. *Computational Statistics & Data Analysis* **30**(3):253–70. ScienceDirect

Examples

```
data(kidtran, package="KMsurv")
k1 <- kidtran
k2 <- k1[k1$gender==1 & k1$race==2, ]
c1 <- coxph(Surv(time, delta) ~ age, data = k2)
cutp(c1, var="age", plot=TRUE)
k2 <- k1[k1$gender==2 & k1$race==2, ]
c1 <- coxph(Surv(time, delta) ~ age, data = k2)
cutp(c1, var="age")
```

---

dx *Diagnostics for* coxph *models*

Description

Diagnostics for coxph models

Usage

```
dx(x, ...)

## S3 method for class 'coxph'
dx(x, ..., what = c("all", "ph", "lin", "inf"),
  toPdf = TRUE, file = "dxPlots.pdf", maxStrata = 5, defCont = 2,
  noQuantiles = 3, maxFact = 4, identify = FALSE, degfP = 3,
  degfRS = 4, degfSS = 4, timeTrans = c("km", "log", "rank", "identity"),
  ties)
```

Arguments

x	An object of class coxph.
...	Additional arguments. Can be passed to graphics::plot or graphics::matplot. See ?par for details.
what	Which plots to make. See **Value** below.
toPdf	Print plots to pdf. This is usually recommended as each plot is created on a new device and 'R' can typically only have 61 devices open simultaneously. • If toPdf=TRUE, each plot is created on a new page. • If toPdf=FALSE, each plot is created on a new screen device.

`file`	Filename to store plots. Default is `"dxPlots.pdf"`.
`maxStrata`	Used for time vs. log-log survival plot. If there are $>$ `maxStrata` strata, no plot is shown for this. Recommended is ≤ 5 to prevent the plot from becoming visually cluttered.
`defCont`	Definition of continuous variable. Variables with more than `defCont` unique values will be split into quantiles to facilitate graphs. (This does *not* apply to `factor` variables).
`noQuantiles`	No. of quantiles into which to split continuous variables
`maxFact`	Maximum number of levels in a factor. Used in plotting differences in log-hazard curves.
`identify`	Identify outliers manually. Cannot be used with `toPdf=TRUE`.
`degfP`	Degrees of freedom for smoothing spline in Poisson model.
`degfRS`	Degrees of freedom for regression splines.
`degfSS`	Degrees of freedom for smoothing splines. If `degFSS=0`, the 'optimal' degrees of freedom is chosen according to the AIC.
`timeTrans`	Type of time transformation to use when refitting model with time-transformed covariate. See `?survival::cox.zph` for details.
`ties`	Method of handling ties when refitting model (for stratified plots). Default is the same as the original model, `x`. Usually one of `"breslow"` or `"efron"`.

Value

Plots with base graphics.
If `what="ph"`:

$\pm$	Time vs. $-\log-\log$ survival. If not too many strata to plot, as per argument `maxStrata`.
*	Quantile-quantile plot. Unit exponential distribution vs. expected events (or Cox-Snell residuals)
*	Observed vs. expected hazard
*	Expected events vs. hazard based on sorted expected events
*	Time vs. hazard, per predictor. Continuous variables are split into quantiles.
*	Time vs. difference in log hazards, per predictor. Continuous variables are split into quantiles.
*	Reference hazard vs. hazard for predictor. Continuous variables are split into quantiles.

If `what="lin"` (only applies to continuous variables):

$\pm$	Predictor vs. residuals from a Poisson model with smoothing spline.

±	Predictor vs. partial residual for predictor (with regression spline). For predictors with > `degfRS`.
±	Predictor vs. partial residual for predictor (with smoothing spline). For predictors with > `degfSS`.
*	Time vs. scaled Schoenfeld residuals, per predictor.

If `what="inf"`:

*	Observation vs. jacknife influence.
*	Observation vs. jacknife influence (scaled by standard error of coefficients).
*	Observation vs. leverage (=scaled score residual).
*	Martingale residuals vs. likelihood displacement residuals.

If `what="lin"`, a `list` of `data.tables` is also returned to the console:

`Poisson`	Results from anova for a Poisson fit (via `gam`) with nonparametric effects. The model is re-fit with smoothing splines for continuous variables. Needs at least one predictor to have > 3 unique values.
`tt`	Results from the time-transformed fit, using `survival::cox.zph`.

Note

TESTS OF PROPORTIONAL HAZARDS

A simple graphical test of proportional hazards, applicable to time-fixed variables with a small number of levels, is a plot of time vs. $-\log(-\log[\hat{S}(t)])$.
The Kaplan-Meier curves should be parallel as:

$$\hat{S}_i(t) = \exp -H_i(t) = \exp[-(H_0(t)\exp[\hat{\beta}X_i(t)])]$$

where $H_0(t)$ is the Breslow's estimator of the baseline hazard (i.e. all co-variates are zero), often represented as $\lambda_0(t)$. Thus

$$-\log(-\log[\hat{S}(t)]) = -\log(H_0(t)) - \hat{\beta}X_i(t)$$

A note on Cox-Snell residuals: Given n observations, the residuals are:

$$\hat{M}_i = Y_i - E(Y_i), \quad i = 1, ..., n$$

where Y_i are the observed events, $E(Y_i)$ are the expected events and $\hat{M}_i$ is the vector of residuals, known as **martingale** residuals.
The expected events $E(Y_i)$ are generated for each observation as

$$E(Y_i) = H_0(t)\exp\sum\hat{\beta}X_i(t)$$

The equation for these residuals may be rewritten as:

$$E(Y_i) = Y_i - \hat{M}_i, \quad i = 1, ..., n$$

Somewhat unintuitively, these predicted values $E(Y_i)$, are also known as the **Cox-Snell** residuals.

These Cox-Snell residuals are used to assess the fit of a proportional hazards model. More formally, they are are generated from the (non time-dependent) covariates X, a matrix with one row per observation (total n) and additional indicators of time t and status δ. The estimated coefficients are $\hat{\beta}$, where $\hat{\beta}$ is a vector of length p (the number of predictors).
The residuals are:

$$r_i = H_0(t_i) \exp \sum_{k=1}^{p} \hat{\beta}_k X_{ik}(t_i), \quad i = 1, ..., n, \quad k = 1, ..., p$$

If the coefficients are close to their true values, then r_i should follow a unit-exponential distribution, i.e. $H_0(t) \approx t$.

Thus a `qqplot` of r_i against a unit-exponential distribution is given. This is of limited value, as the *null* model (no coefficients) will be closer to the true exponential distribution than that with *any* coefficients.

Another simple graphical test is a plot of observed vs. expected values for the cumulative hazard.
The expected values of the hazard are generated using the expected events (using `as.Surv`). This should follow a straight line through the origin with a slope of 1.

To check if the coefficients are close to their true values, we can compute the Nelson-Aalen estimator of the cumulative hazard rate of the r_i's. A plot of this estimator against r_i should be a straight line through the origin with a slope of 1.

Continuous predictors are split into quantiles to facilitate the following plots:

- Plots of time vs. cumulative hazard, per predictor, should be a constant multiples of a baseline hazard, i.e. parallel.
- Plots of time vs. difference in log hazards, per predictor, should be constant over time i.e. parallel.
  The difference should be close to 0.
- Plots of hazard vs. reference group, per predictor, should be linear with a slope of 45 degrees.

Discretizing a continuous variable

These methods work by stratifying a covariate K into q disjoint quantiles or groups g. A stratified `coxph` model is fitted to these quantiles and one is selected as a reference.
The **cumulative hazard** $\hat{H}_g(t)$ is plotted for each group g. These should be a constant multiple of the reference stratum $\hat{H}_1(t)$ over time.

A simpler way to compare these is to plot the **differences in log cumulative hazard**, that is:

$$\log \hat{H}_g(t) - \log \hat{H}_1(t), \quad g = 2, ..., q$$

Each curve should be horizontal and constant over time.
Curves above zero indicate an increase in hazard in group g vs. the reference at that time.

Andersen plots show $\log \hat{H}_1(t)$ vs. $\log \hat{H}_g(t), \quad g = 2, ..., q$.

If proportional hazards are present, these should be straight lines through the origin.
If the curve is convex (towards the upper left of the plot) this shows that $\hat{H}_g(t) \div \hat{H}_1(t)$ is an increasing function of t. Thus if convex, the hazard rate in group g is increased vs. the reference, group 1.

A model with **time-dependent coefficients** should not vary from one without such coefficients if the assumption of proportional-hazards is met. That is, making the coefficient a function of time, $\hat{\beta}_k \rightarrow f(\hat{\beta}_k, t)$ and plotting this against time t should give a horizontal line.
To test this we plot the *scaled Schoenfeld residuals* against time. These are

$$s_i^* = V^{-1}(\hat{\beta}, t_i)s_i$$

A note on generating Schoenfeld residuals: These are based on the contribution of each observation to the derivative of the log partial likelihood.
They are defined for each time where an event occurs and have a value for each coefficient $\hat{\beta}_k$.
They are given by:

$$s_{ik} = X_{ik} - \bar{x}_k, \quad i = 1, ..., n \quad k = 1, ..., p$$

Here, $\bar{x}_k$ is the mean of those still at risk for covariate k.
This is a weighted mean of the values of X_k (for cofficient k):

$$\bar{x}_k = \frac{\sum WX_i}{\sum W}$$

and the weights are:

$$W = \exp \hat{\beta} X_i$$

where X_i refers to those still at risk at time t_i.
Now the inverse of the variance of s_{ik} is approximately:

$$V^{-1} \approx YV\hat{(}\beta)$$

where Y is the number of events and $V\hat{(}\beta)$ is the covariance matrix of the estimated coefficients.

Given the above

$$E(s_i^*) + \hat{\beta}_k \approx \hat{\beta}_k(t_i)$$

so that a plot of time vs. s_i^* should be horizontal.

TESTS OF LINEARITY FOR CONTINUOUS VARIABLES

The **martingale** residual is used to help determine the best functional form of a covariate in a `coxph` model. As above, the Cox model assumes that the hazard function satisfies:

$$H_i(t) = H_0(t) \exp X_i\hat{\beta}$$

That is, for a continuous variable, a unit increase in the variable produces the same change in risk across the value of the variable. (E.g. an increase in age of 5 years leads to the same change in hazard, no matter what the increase is from or to).
To verify this is the case, a null model is fitted (i.e no coefficients, similar to intercept-only model in linear regression). Martingale residuals are calcuated for this.
Plots of these residuals against the values of each of the predictors in the model are shown. If

the correct model for covariate k is based on a smooth function $f()$, i.e. $\exp(f(X_k)\hat{\beta_k})$ then the following should hold:

$$E(M_i|X_{ik} = X_k) \approx c.f(X_k)$$

Where M_i is the martingale residual and the constant c depends on the amount of **c**ensoring and is roughly independent of X_k.
A `lowess` smoothed line is added to the plot. This should be approximately linear if the assumption of proportional hazards is met. If the plot shows a sharp threshold, a discretised version of the covariate may be preferable.

Poisson regression models *are also* proportional hazards models. The Cox model may thus be rewritten in Poisson form to allow for application of residual methods applicable to Poisson regression.
The Cox model can be written as:

$$H_i(t) = \exp(f(x)\hat{\beta}H_0(t))$$

And the standard Poisson model can be written as:

$$E(Y_i|X) = \exp X_i T$$

where Y_i are the observed events and T is the observation time (often referred to as θ). This is thus:

$$E(Y_i|X) = \exp((f(X_i)\hat{\beta}) \int_0 Y_i(t)H_0(t)dt)$$

Where $T = \int_0 Y_i(t)H_0(t)dt$. Once expressed as a Poisson model, this can be analysed using tools available for generalized additive models (`gam`).
To do this the `coxph` model is refit with `gam`. The outcome (left-hand side) is those times in which an event occurred. The predictors are the same. For continuous terms, an attempt is made to fit a non-linear function to improve the fit, with a default of 4 degrees of freedom (`degfP=4`).
The Poisson model fit with `gam` has an additional `offset` term. This is

$$\texttt{offset} = \log[\exp(-X_i\hat{\beta})\exp(X_i\hat{\beta}H_0(t_i))] = H_0(t_i) \approx \int_0 Y_i(t)H_0(t)dt$$

See `?predict.coxph` for details. Plots and anova are generated (see `?anova.gam` for details). Plots show the residuals by the values of the predictor. Ideally these should be horizontal with the intercept at zero.

Regression splines may be used to replace continuous terms directly in the `coxph` function. These are fit by connecting number of knots with locally fitting curves. The degrees of freedom (by default `degfRS=4`) is the number of knots plus one. `degfRS` -1 dummy variables are generated to try to improve the fit of the variable. Plots of the original variable vs. the fitted splines should be linear. The function uses B-splines.

Penalized smoothing splines are an alternative to regression splines which, for small degrees of freedom, have better properties regarding their locality of influence. They are chosen to minimize β for the basis functions, in:

$$\theta \sum_{i=1}^{n} [y_i - f(x_i, \beta)]^2 + (1 - \theta) \int [f''(x, \beta)]^2 dx$$

Here the first term is the residual sum of squares and the second is the integral of the second derivative of the function f with respect to x.
For a straight line $f''(x) = 0$ and the term will increase in proportion to the degree of curvature. θ is a **t**uning parameter based on the degrees of freedom (by default `degfSS=4`). As $\theta \to 0$ (2 degrees of freedom, including intercept), the solution converges to the least-squares line. As $\theta \to 1$, (n degrees of freedom), the solution approaches a curve that passes through each point. Plots of the fitted splines vs. the original variable should be linear.

TESTS OF INFLUENCE

The simplest measure of influence is the **jackknife** value

$$J_i = \hat{\beta} - \hat{\beta}_{-i}$$

where β_{-i} is the result of a fit that includes all observations except i. This can be computed as

- Converge to $\hat{\beta}$ as usual e.g. via Newton-Raphson.
- Delete observation i.
- Perform one additional iteration.

This may be expressed as:

$$\delta\beta = 1'(U\chi^{-1}) = 1'D$$

Here D, the matrix of **dfbeta residuals**, is comprised of the score residuals U scaled by the variance of β, $var(B)$. Each row of D is the change in $\hat{\beta}$ if observation i is removed.

Caution - for plots to verify proportional hazards: the variance of the curves is not constant over time. Increasing departures from model assumptions are likely to be found as time increases. `package:surv2sample` may need to be installed from source to allow one of the examples to run.

References

Examples are from **K&M** Example 11.1 - 11.7, pg 355–66.

Last example is from: Therneau T, Grambsch P 2000. *Modeling Survival Data*, 1st edition. New York: Springer. Section 5.1.2, pg 91. Springer (paywall)

Andersen PK, Borgan O, Gill R, Keiding N 1982. Linear Nonparametric Tests for Comparison of Counting Processes, with Applications to Censored Survival Data, Correspondent Paper. *International Statistical Review* **50**(3):219–44. JSTOR

Examples

```
## Not run:
### running these examples with toPdf=FALSE will
### open too many devices to be compatible with R CMD check
### results from these examples can be found in the package source
### under /survMisc/inst/doc/
###
### for log-log plot
if(require(devtools)){
### this package is now archived, so need to install from url
```

```
 install_url("http://cran.r-project.org/src/contrib/Archive/surv2sample/surv2sample_0.1-2.tar.gz")
     library(surv2sample)
     data(gastric, package="surv2sample")
     dx(coxph(Surv(time/365, event) ~ treatment, data=gastric), file="gasDx.pdf")
 }
data(bmt, package="KMsurv")
bmt <- within(bmt, {
z1 <- z1 -28
z2 <- z2- 28
z3 <- z1*z2
z4 <- as.double(group == 2)
z5 <- as.double(group == 3)
z6 <- z8
z7 <- (z7 / 30) - 9
z8 <- z10
})
c1 <- coxph(Surv(t2, d3) ~ z1 + z2 + z3 + z4 + z5 + z6 + z7 + z8,
         method="breslow", data=bmt)
dx(c1, file="bmtDx.pdf")
###
data(alloauto, package="KMsurv")
c1 <- coxph(Surv(time,delta) ~ factor(type),
            method="breslow", data=alloauto)
dx(c1, file="alloDx.pdf")
### GAM model. Therneau 5.3
data(pbc, package="survival")
w1 <- which(is.na(pbc$protime))
pbc <- pbc[-w1, ]
c1 <- coxph(Surv(time, status==2) ~ age + edema + bili + protime + albumin,
            data=pbc, method="breslow")
dx(c1, file="pbcDx.pdf")
### Time dependent covariate. Therneau 6.3
data(veteran, package="survival")
veteran$celltype <- relevel(veteran$celltype, ref="adeno")
c1 <- coxph(Surv(time, status) ~ trt * celltype + karno + diagtime + log(age) + prior,
            data=veteran[-1, ])
dx(c1, what="ph", file="vetDx.pdf")

## End(Not run)
### simple example which doesn't take up too many devices
c1 <- coxph(formula = Surv(time, status == 2) ~ age + log(bili), data=pbc)
dx(c1)
```

---

gamTerms *Individual terms of a Generalized Additive or Cox Proportional Hazards Model*

Description

Returns the individual terms of a gam or coxph object, along with the standard errors, in a way useful for plotting.

Usage

```
gamTerms(fit, se = TRUE, link = FALSE, weights, data)
```

Arguments

fit	The result of a Generalized Additive Model (gam) or a Cox proportional hazards model (coxph)
se	If TRUE, also return the standard errors
link	If TRUE, then the individual terms are centered so that the average of the inverse-link of the data, i.e., the data on the original scale, has mean zero
weights	A vector of case weights. If not supplied (the default), the data is centered so that the weighted mean is zero.
data	A data.frame in which to evaluate the model. If missing, eval(fit$call$data) is used.

Value

A list with one element per term.
Each element is a matrix whose columns are x, y, and (optionally) se(y).
There is one row per unique x value, and the matrix is sorted by these values. (This makes it easy to plot the results).
The first element of the list, constant, contains an overall mean for the decomposition.

Author(s)

Terry Therneau, Dirk Larson. Updated/adapted from S-plus by Chris Dardis.

See Also

air
?gam::gam
?gam::plot.gam

Examples

```
data(air, package="survMisc")
gfit <- gam::gam(ozone ~ gam::s(temperature) + gam::s(wind), data=air)
temp <- gamTerms(gfit)
identical(names(temp), c("constant", "temperature", "wind"))
### air has 111 rows, but only 28 unique wind speeds:
dim(temp$wind)
### plot the fit versus square root of wind speed
yy <- cbind(temp$wind[, 2],
            temp$wind[, 2] - 1.96 * temp$wind[, 3],
            temp$wind[, 2] + 1.96 * temp$wind[, 3])
### Adding the constant makes this a plot of
### actual y (ozone) at the mean temp
yy <- yy + temp$constant
```

```
graphics::matplot(sqrt(temp$wind[, 1]), yy, lty=c(1, 2, 2),
                  type='l', col=1, xaxt='n', xlab='Wind Speed', ylab='Ozone')
temp <- seq(3, 19, 2)
graphics::axis(1, sqrt(temp), format(temp))
```

`gastric` *gastric cancer trial data*

Description

gastric cancer trial data

Format

A `data.frame` with 90 rows (observations) and 3 columns (variables).

Details

Data from a trial of locally unresectable gastic cancer.
Patients (45 in each group) were randomized to one of two groups: chemotheapy vs. chemotherapy + radiotherapy.
Columns are:

time Time in days

event Death

group Treatment

- **0** chemotherapy
- **1** chemotherapy + radiotherapy

Source

Klein J, Moeschberger. Survival Analysis, 2nd edition. Springer 2003. Example 7.9, pg 224.

References

Gastrointestinal Tumor Study Group, 1982. A comparison of combination chemotherapy and combined modality therapy for locally advanced gastric carcinoma. *Cancer*. **49**(9):1771-7. Pubmed.

Stablein DM, Koutrouvelis IA, 1985. A two-sample test sensitive to crossing hazards in uncensored and singly censored data. *Biometrics*. **41**(3):643-52. JSTOR.

See Also

comp

genSurv *Generate survival data*

Description

Generate survival data

Usage

```
genSurvDf(b = 2L, f = 2L, c = 1L, n = 100L, pb = 0.5, nlf = 3L,
  rc = 0.8, pe = 0.5, t0 = 1L, tMax = 100L, asFactor = TRUE,
  model = FALSE, timelim = 5)

genSurvDt(b = 2L, f = 2L, c = 1L, n = 100L, pb = 0.5, nlf = 3L,
  rc = 0.8, pe = 0.5, t0 = 1L, tMax = 100L, asFactor = TRUE,
  model = TRUE, timelim = 5)
```

Arguments

b	*binomial predictors*, the number of predictors which are binary, i.e. limited to 0 or 1
f	*factors*, the number of predictors which are factors
c	*continuous predictors*, the number of predictors which are continuous
n	number of observations (rows) in the data
nlf	the number of levels in a factor
pb	*probability for binomnial predictors*: the probability of binomial predictors being $= 1$ e.g. if `pb=0.3`, 30% will be 1s, 70% will be 0s
rc	*ratio for continuous variables*: the ratio of levels of continuous variables to the total number of observations *n* e.g. if `rc=0.8` and `n=100`, it will be in the range $1 - 80$
pe	*probability of event* the probability of events (typically death/failure) occurring, i.e. $P(e = 1)$. e.g. if `pe=0.6`, 60% will be 1s, 40% will be 0s
t0	Lowest (starting) time
tMax	Highest (final) time
asFactor	if `asFactor=TRUE` (the default), predictors given as `factors` will be converted to `factors` in the data frame before the model is fit
timelim	function will timeout after `timelim` secs. This is present to prevent duplication of rows.
model	If `model=TRUE` will also return a model fitted with `survival::coxph`.

Value

If `model=FALSE` (the default) a `data.frame` or `data.table` as above.
If `model=TRUE`: a list with the following values:

`df or dt`	A `data.frame` (for `genSurvDf`) or `data.table` (for `genSurvDt`). Predictors are labelled $x1, x2, ..., xn$. Outcome is $t1$ (time) and e event (e.g. death). Rows represent to n observations
`model`	A model fit with `survival::coxph`

Note

`genSurvDt` is faster and more efficient for larger datasets.

Using `asFactor=TRUE` with factors which have a large number of levels (e.g. `nlf >30`) on large datasets (e.g. $n > 1000$) can cause fitting to be slow.

Examples

```
set.seed(1)
genSurvDf(model=TRUE)
genSurvDf(b=0, c=2, n=100, pe=0.7)
genSurvDf(b=1, c=0, n=1000)
genSurvDf(f=1, nlf=4, b=1, c=0, asFactor=FALSE)
set.seed(1)
genSurvDt()
genSurvDt(b=0, f=0, c=1, n=20L, pe=0.7)
```

---

gof *Goodness of fit test for* coxph *models*

Description

Goodness of fit test for coxph models

Usage

```
gof(x, ...)

## S3 method for class 'coxph'
gof(x, ..., G = NULL)
```

Arguments

`x`	An object of class coxph
`...`	Additional arguments (not implemented)

G — Number of **g**roups into which to divide risk score. If `G=NULL` (the default), uses closest integer to

$$G = \max(2, \quad \min(10, \quad \frac{ne}{40}))$$

where ne is the number of events overall.

Details

In order to verify the overall goodness of fit, the risk score r_i for each observation i is given by

$$r_i = \hat{\beta} X_i$$

where $\hat{\beta}$ is the vector of fitted coefficients and X_i is the vector of predictor variables for observation i.
This risk score is then sorted and 'lumped' into a grouping variable with G groups, (containing approximately equal numbers of observations).
The number of observed (e) and expected (exp) events in each group are used to generate a Z statistic for each group, which is assumed to follow a normal distribution with $Z \sim N(0, 1)$.
The indicator variable `indicG` is added to the original model and the two models are compared to determine the improvement in fit via the likelihood ratio test.

Value

A `list` with elements:

`groups` — A `data.table` with one row per group G. The columns are

n Number of observations

e Number of events

exp Number of events expected. This is

$$exp = \sum e_i - M_i$$

where e_i are the events and M_i are the martingale residuals for each observation i

z Z score, calculated as

$$Z = \frac{e - exp}{\sqrt{exp}}$$

p p-value for Z, which is

$$p = 2.\texttt{pnorm}(-|z|)$$

where `pnorm` is the normal distribution function with mean $\mu = 0$ and standard deviation $\sigma = 1$ and $|z|$ is the absolute value.

`lrTest` — Likelihood-ratio test. Tests the improvement in log-likelihood with addition of an indicator variable with $G-1$ groups. This is done with `survival:::anova.coxph`. The test is distributed as chi-square with $G - 1$ degrees of freedom

Note

The choice of G is somewhat arbitrary but rarely should be > 10.
As illustrated in the example, a larger value for G makes the Z test for each group more likely to be significant. This does *not* affect the significance of adding the indicator variable `indicG` to the original model.

The Z score is chosen for simplicity, as for large sample sizes the Poisson distribution approaches the normal. Strictly speaking, the Poisson would be more appropriate for e and exp as per Counting Theory.
The Z score may be somewhat conservative as the expected events are calculated using the martingale residuals from the overall model, rather than by group. This is likely to bring the expected events closer to the observed events.

This test is similar to the Hosmer-Lemeshow test for logistic regression.

Source

Method and example are from:
May S, Hosmer DW 1998. A simplified method of calculating an overall goodness-of-fit test for the Cox proportional hazards model. *Lifetime Data Analysis* **4**(2):109–20. Springer (paywall)

References

Default value for G as per:
May S, Hosmer DW 2004. A cautionary note on the use of the Gronnesby and Borgan goodness-of-fit test for the Cox proportional hazards model. *Lifetime Data Analysis* **10**(3):283–91. Springer (paywall)

Changes to the `pbc` dataset in the example are as detailed in:
Fleming T, Harrington D 2005. *Counting Processes and Survival Analysis*. New Jersey: Wiley and Sons. Chapter 4, section 4.6, pp 188. Wiley (paywall)

Examples

```
data("pbc", package="survival")
pbc <- pbc[!is.na(pbc$trt), ]
### make corrections as per Fleming
pbc[pbc$id==253, "age"] <-  54.4
pbc[pbc$id==107, "protime"] <-  10.7
### misspecified; should be log(bili) and log(protime) instead
c1 <- coxph(Surv(time, status==2) ~
            age + log(albumin) + bili + edema + protime,
            data=pbc)
gof(c1, G=10)
gof(c1)
```

ic *Information criterion*

Description

Information **C**riterion for a fitted model.

Usage

```
BIC(object, ...)

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

AIC(object, ..., k = 2)

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

AICc(object, ...)

## S3 method for class 'coxph'
AICc(object, ..., k = 2)
```

Arguments

object	An object of class coxph
...	Not implemented
k	The weight of the equivalent degrees of freedom (edf) of the AIC formula

Details

Given a set of candidate models for the same data, the preferred model is the one with the minimum IC value.
The Akaike information criterion, AIC, is given by

$$AIC = k.edf - 2\ln L$$

Where edf is the equivalent degrees of freedom (i.e., equivalent to the number of free parameters in the model) and L is the model likelihood.
k is a constant, which is $= 2$ for the traditional AIC.

AIC corrected for finite sample size n, AICc, is

$$AICc = AIC + \frac{k.edf(edf+1)}{n - edf - 1}$$

where n is the sample size. Thus there is a greater penalty for more parameters.

The Bayesian information criterion is

$$BIC = \ln n.edf - 2 \ln L$$

This penalises models with more parameters to a greater extent.

Value

A named vector with

edf the equivalent degrees of freedom for the fitted model fit

IC the information criterion, either AIC, AICc or BIC

Note

For survival models the **effective** n is the number of events rather than the number of observations. This is used in computing the criteria above.

`local` *Local tests for a model*

Description

Local tests for a model

Usage

```
locScore(x, ...)

## S3 method for class 'coxph'
locScore(x, ..., all = FALSE, hypo = NULL,
  ties = c("breslow", "efron", "exact"))

locLR(x, ...)

## S3 method for class 'coxph'
locLR(x, ..., all = FALSE, hypo = NULL,
  ties = c("breslow", "efron", "exact"))

locWald(x, ...)

## S3 method for class 'coxph'
locWald(x, ..., all = FALSE, hypo = NULL)
```

Arguments

x	A model of class coxph
...	Additional arguments (not implemented)
all	Fit *all* combinations of predictors
hypo	Hypothesis to test. There should be at least one coefficient to exclude and one to keep. This is a specified as vector of the same length as the number of coefficients in the model. This should be a logical vector (i.e. composed of TRUE and FALSE or a vector of 0s and 1s. • FALSE or zeros indicate coefficients to exclude • TRUE or ones indicate coefficients to keep.
ties	Method of handling ties when refitting model. Must be one of breslow, efron or exact.

Details

The null hypothesis is that some of the coefficients in the model are zero ($H_0 : \hat{B}_i = 0, \quad i \geq 1$) vs. the alternative that at least one of them is nonzero.

All of these tests are distributed as chi-square with degrees of freedom = number of excluded coefficients.

For the **score** test, the model is fitted again with the coefficients of interest excluded.
A value for the remaining coefficients is obtained. Then the complete model is fit again using these new values as initial values for those remaining coefficients and using zero as the initial value for the excluded coefficients.
Values for the excluded coefficients are generated without iteration. (I.e. the first values calculated, with no convergence towards maximum likelihood estimators).
The test is:

$$\chi^2_{SC} = U^T I^{-1} U$$

where U is the score vector and I^{-1} is the covariance or inverse of the information matrix. (These are given by `colSums(survival::coxph.detail(x)$score)` and `x$var` respectively).

For the **likelihood ratio** test, the model is also refit with the coefficients of interest excluded. The likelihood ratios from the full model and those with coefficients excluded are used to construct the test:

$$\chi^2_{LR} = 2(LR_{full} - LR_{excluded})$$

The **Wald** chi-squared statistic is given by:

$$\chi^2_W = \hat{B}^T I^{-1} \hat{B}$$

Where $\hat{B}$ is the vector of fitted coefficients (from the complete model) thought to be $= 0$.
I^{-1} is composed of the corresponding elements from the covariance matrix of the model.

Value

For `locScore` a `list` with the following elements, which are `data.tables`:

`coef`	coefficients from refitted model(s)
`score`	hypothesis and chi-square test

For `locLR` and `locWald`, a `data.table` showing the hypothesis being tested and the results of the test.

References

Examples are from: **K&M** Example 8.2, pp 264-6.

Examples

```
data(larynx, package="KMsurv")
c1 <- coxph(Surv(time, delta) ~ factor(stage) + age, data=larynx)
locScore(c1, all=TRUE)
locScore(c1, hypo=c(0, 0, 0, 1))
locScore(coxph(Surv(time, delta) ~ stage + age, data=larynx))
###
data(larynx, package="KMsurv")
c1 <- coxph(Surv(time, delta) ~ factor(stage) + age, data=larynx, method="breslow")
locLR(c1, all=TRUE)
locLR(c1, hypo=c(FALSE, FALSE, FALSE, TRUE))
###
data(larynx, package="KMsurv")
c1 <- coxph(Surv(time, delta) ~ factor(stage) + age, data=larynx, method="breslow")
locWald(c1, all=TRUE)
locWald(c1, hypo=c(0, 0, 0, 1))
```

---

lrSS *Sample size required to show difference in survival by log-rank test given prior information about Kaplan-Meier estimate*

Description

No. of events required in a two-group trial (with one binary covariate) for a two-sided log-rank test to detect a given hazard ratio.
This is calculated by:

$$n = \frac{(Z_{\frac{\alpha}{2}} + Z_{\beta})^2}{p(1-p)\log^2 HR}$$

Where Z refers to the corresponding Z-value from the standard normal distribution.
This default calculation requires that the subjects be followed until *all* have experienced the event of interest (typically death). If this is not likely to be the case, then a more informed estimate may be generated by dividing n by the overall probability of death occurring by the end of the study.
This may be generated with prior information about $\hat{S}$ and median survival times (for the *control* group B; group A is the *experimental* group).

Given accrual time a and follow-up time f, Simpsons rule can be used to give an estimate of the proportion of patients that will die in group B:

$$d_B = 1 - \frac{1}{6}[\hat{S}_B(f) + 4\hat{S}_B(f + 0.5a) + \hat{S}_B(f + a)]$$

Given median survival time t, the proportion of patients expected to die in group B is:

$$d_B = [1 - \frac{e^{\frac{-0.69f}{t}}(1 - e^{\frac{-0.69f}{t}})}{\frac{0.69a}{t}}]$$

Usage

```
lrSS(HR, alpha = 0.1, beta = 0.2, p = 0.5, Sp, tp, mtp, a, f)
```

Arguments

`HR`	Hazard Ratio. Ratio of hazard with treatment to that without.
`alpha`	Significance level α, two-tailed
`beta`	Power is $1 - \beta$
`p`	Proportion of subjects allocated to one group. Needs to be in range $0 - 1$. Arbitrary - can be either of the two groups.
`Sp`	Prior Kaplan-Meier estimate of survival (given no intervention)
`tp`	Prior times corresponding to survival estimates. There must be one time corresponding to each of: $f, 0.5 * a + f, a + f$.
`mtp`	Median time, prior. (Prior median survival time).
`a`	Accrue. Time period for which subjects accrued.
`f`	Follow-up. Time period for which subjects followed-up.

Value

If any of `Sp`, `tp`, `mtp` `a` or `f` are missing, will return the number of subjects required (with *no* prior information). Otherwise, returns a `list` with the following values:

n number of subjects required (with no prior information)

pS with prior Kaplan-Meier estimates:

- **dB** probability death in group B (that with prior information)
- **dA** probability death in group A (new treatment)
- **p** overall probability of death
- **n** number of subjects required

pM with prior median survival time estimates:

- **dB** probability death in group B (that with prior information)
- **dA** probability death in group A (new treatment)
- **p** overall probability of death
- **n** number of subjects required

Note

Assumes there are two groups and one intervention (covariate) which is present or absent. The values in second example are taken from Schoenfelds paper, except for `mtp`.

Source

Schoenfeld D, 1983. Sample-size formula for the proportional-hazards regression model. *Biometrics*. (39):499-503. JSTOR

See Also

`btumors`

Examples

```
lrSS(HR=1.3, alpha=0.05)
data(btumors)
m1 <- mean(rep(btumors[,"ms"],btumors[,"n"]))
lrSS(HR=1.5, Sp=c(0.43, 0.2, 0.11), tp=c(1, 2, 3), mtp=m1, a=2, f=1)
```

---

mean.Surv *Mean for* Surv *object*

Description

Mean for `Surv` object

Usage

```
## S3 method for class 'Surv'
mean(x, alpha = 0.05, method = c("Efron", "Gill", "Brown"),
  tMax = NULL, by = 1, dfm = FALSE, ...)
```

Arguments

`x`	A `Surv` object
`alpha`	Significance level α
`method`	If the last observation is censored at time t_k, one of the following values for $\hat{S}$, the Kaplan-Meier estimate of survival time from then until `tMax` is used: **Effron** $\hat{S} = 0$ **Gill** $\hat{S} = \hat{S}(t_k)$ i.e. $\hat{S}$ is equal to the last recorded value of $\hat{S}$. **Brown** $\hat{S} = e^{\frac{t_i}{t_k} \log \hat{S}(t_k)}$ for $t_k \leq t_i \leq$ `tMax`
`tMax`	If the last observation is censored at time t_k, an estimate of $\hat{S}$ will be generated from t_k to `tMax`. If `tMax=NULL` a value of $2 \times t_{max}$, twice the longest time recorded, is used.

by	Increments (units of time) between t_k and tMax
dfm	If TRUE, will return the dataframe used to calculate the statistics
...	Additional arguments

Value

A list with the following elements:

mean	Mean of the Surv object
variance	The variance
CI	The confidence level (from $alpha$ above)
upper	Upper value for confidence interval
lower	Lower value for the confidence interval

If the last observation is censored at time t_k, two values are returned, one calculated up to t_k, the other to tMax.

Examples

```
data(bmt, package="KMsurv")
b1 <- bmt[bmt$group==1, ] # ALL patients
s1 <- Surv(time=b1$t2, event=b1$d3)
mean(s1)
mean(Surv(time=c(6, 14, 21, 44, 62), event=c(1, 1, 0, 1, 1)))
mean(Surv(time=c(6, 14, 21, 44, 62), event=c(1, 1, 0, 1, 0)))
```

---

mpip *Multi-center Post-Infarction Project*

Description

Multi-center Post-Infarction Project

Details

A data frame with 866 observations (rows) and 13 variables (columns), taken from an environmental study that followed patients admitted to hospital with a myocardial infarction. The goal of the study was to identify factors which would be important in predicting the clinical course of the patients.

Columns are:

ved Ventricular etopic depolarizations/hour, from a 24-hour Holter (ECG) monitor. A large number of these irregular heartbeats may predict a high risk of fatal arrythmia. The variable is highly skewed.

angina Angina

0 None

1 With physical activity or emotion

2 At rest

educat Education level. 1=postgraduate. 8=less than grade 7

mi Myocardial infarction

0 No

1 Yes

nyha New York Heart Association Class

1 Cardiac disease, but no symptoms and no limitation in ordinary physical activity

2 Mild symptoms (Mild shortness of breath and/or angina) and slight limitation during ordinary activity.

3 Marked limitation in activity due to symptoms, even during less-than-ordinary activity, e.g. walking short distances (20-100 m). Comfortable only at rest.

4 Severe limitations. Experiences symptoms even while at rest. Mostly bedbound patients.

rales Pulmonary rales on initial examination.

0 No

1 Yes

ef Ejection fraction: percentage of blood cleared from the heart on each contraction.

ecg ECG classification. Anterior vs. inferior vs. other was the main grouping of interest.

11 or 12 Anterior

14 Inferior

anyAngina Any angina. Binary classification based on variable angina above.

0 No

1 Yes

futime Follow-up time (days)

status **0** Alive

1 Died

date Date of enrollment, with reference to day 0, 1/1/1960

bb Use of beta-blockers.

0 No

1 Yes

lved Log ved. Derived from ved above, as $lved = log(ved + 0.02)$. This is done to overcome skewing.

Source

Survival package - Revision 6780

References

The Multicenter Postinfarction Research Group 1983. Risk stratification and survival after myocardial infarction. *N Engl J Med.* **309**(6):331-6. NEJM (paywall)

See Also

plotTerm

---

multi *Multiple* coxph *models*

Description

Multiple coxph models

Usage

```
multi(x, ...)

## S3 method for class 'coxph'
multi(x, ..., maxCoef = 5L, crit = c("aic", "aicc", "bic"),
  how = c("all", "evolve"), confSetSize = 100L, maxiter = 100L,
  bunch = 1000L, mutRate = 0.1, sexRate = 0.2, immRate = 0.3,
  deltaM = 1, deltaB = 1, conseq = 10L, report = TRUE)
```

Arguments

x	An object of class coxph
...	Not implemented
maxCoef	Maximum no. of coefficients
crit	Information criterion *IC*
how	Method used to fit models. If how="all" (the default), all subsets of the given model will be fit
confSetSize	Size of returned confidence size. Number represents a row in the set. (Columns represent parameters/coefficients in the models).
maxiter	Maximum no. of iterations to use (for cox fitter). Needs to be integer and should not normally need to be > 100.
bunch	When using how="evolve": no. of models to screen per generation
mutRate	Mutation rate for new models (both asexual and sexual selection). Should be in range $0 - 1$.
sexRate	Sexual reproduction rate. Should be in range $0 - 1$.
immRate	Immigration rate. Should be in range $0 - 1$. Also sexRate + immRate should not be > 1.
deltaM	Target for change in mean IC determining convergence when how="evolve". The last mean IC (from the best confSetSize models screened) is compared with that from the most recently fitted bunch.
deltaB	Change in best IC determining convergence of evolution. This typically converges faster than deltaB.

conseq	Consecutive generations allowed which are 'divergent' by both of the above criteria. Algorithm will stop after this no. is reached.
report	If report=TRUE (the default), print report to screen during fitting. Gives current change in best and mean IC as well as object size of fitted models.

Details

This is based loosely on package:glmulti (although is admittedly less efficient). A more detailed discussion of the issues involved in multiple model fitting is presented in the reference paper describing that package's implementation.

It is designed for cases where there a large no. of candidate models for a given dataset (currently only right-censored survival data). First, the model.matrix for the given formula is constructed. For those unfamiliar with model.matrix, a predictor given as a factor is expanded to it's design matrix, so that e.g. for 4 original levels there will be 3 binary $(0/1)$ columns. Currently all levels of a factor are considered independently when fitting models.
Thus there is one column for each coefficient in the original model.

The original formula can include the following terms: offset, weight and strata. Other *special* terms such as cluster are not currently supported. The formula may contain interaction terms and other transformations.

If how="all", all possible combinations of these coefficients are fitted (or up to maxCoef predictors if this is less).

If how="evolve" the algorithm proceeds as follows:

1. Fit bunch random models and sort by IC
2. Generate another bunch candidate models based on these. immRate gives the proportion that will be completely random new models. sexRate gives the proportion that will be the products of existing models. These are a random combination of the first elements from model 1 and the last elements from model 2. The sum of immRate and sexRate should thus be $<= 1$.
3. Other models (asexual) will be selected from the existing pool of fitted models with a likelihood inversely proportional to their IC (i.e. lower IC - more likely). Both these and those generated by sexual reproduction have a chance of mutation (elements changing from 1 to 0 or vice versa) given by mutRate.
4. Fit new models (not already fitted).
5. Proceed until model fitting is 'divergent' conseq times then stop. Divergent is here taken to mean that the targets for *both* deltaM and deltaB have not been met.
   deltaM typically converges more slowly. Thus a large value of deltaM will require new bunches of models to be signifiantly better than the best (size = confSetSize) existing candidates. Negative values of deltaM (not typically recommended) are more permissive; i.e. new models can be somewhat worse than those existing.

The models are returned in a data.table, with one row per model giving the fitted coefficients, the IC for the model and the relative evidence weights.

Value

A `data.table` with one row per model. This is of `class` `multi.coxph` which has it's own `plot` method. Columns show the coefficients from the fitted model. Values of 0 indicate coefficient was not included. The `data.table` is sorted by IC and also gives a column for relative evidence weights. These are generated from:

$$w_i = \exp(\frac{-IC_i - IC_{best}}{2})$$

Where IC_i is the information criterion for the given model, and IC_{best} is that for the best model yet fitted. They are then scaled to sum to 1.

Note

The algorithm will tend to slow as the population of fitted models expands.

References

Calgano V, de Mazancourt C, 2010. glmulti: An R Package for Easy Automated Model Selection with (Generalized) Linear Models. *Journal of Statistical Software*. **34**(12):1-29. Available at JSS.

See Also

`ic`

`plot.MultiCoxph`

Examples

```
set.seed(1)
df1 <- genSurvDf(b=1, c=5, f=0, model=FALSE)
multi(coxph(Surv(t1, e) ~ ., data=df1), crit="aic")
## Not run:
### longer example
dt1 <- genSurvDt(b=1, c=30, f=0, model=FALSE)
multi(coxph(Surv(t1, e) ~ ., data=dt1),
maxCoef=8, crit="bic", how="evolve", deltaM=1, deltaB=1, conseq=10)
## End(Not run)
```

---

plot.MultiCoxph *Plot an* `multi.coxph` *object*

Description

Plot an `multi.coxph` object

Usage

```
## S3 method for class 'multi.coxph'
plot(x, type = c("p", "w", "s"), ...)
```

Arguments

x	An object of class multi.coxph
...	Additional arguments. These are passed to graphics::plot.default or (if type="s") graphics::barplot.
type	Type of plot

Details

One of three types of graph is possible.

- If type="p" then **p**oints representing the information criterion (IC) for each model are plotted. A line is also drawn 2 units above the minimum IC. Models below this are typically worth considering. If it is not visible on the plot, then important models have have been overlooked, suggesting a larger value for confSetSize may be appropriate.
- If type="w" then the **w**eights (relative evidence weights) of the models are plotted. These can be interpreted as the probability that each model is the best in the set. A red vertical line is shown where the cumulated evidence weight reaches 95
- If type="s" then the **s**um of the relative evidence weights for each term/ coefficient is plotted. The sum is taken across all models in which the term appears.

Value

A graph (base graphics).

See Also

multi

Examples

```
set.seed(1)
dt1 <- genSurvDt(b=2, c=5, f=0, model=FALSE)
m1 <- multi(coxph(Surv(t1, e) ~ ., data=dt1), crit="bic")
plot(m1, type="w")
```

---

plot.Surv *Plot Survival object*

Description

Plots an object of class Surv.
Different methods apply to different types of Surv objects.

Usage

```
## S3 method for class 'Surv'
plot(x, l = 3, ...)
```

Arguments

x	A Surv object
l	Length of arrow. Length is l/nrow(x)
...	Additional arguments. These are passed to graphics::arrows when drawing right- or left-censored observations.

Value

A graph (base graphics). The type of graph depends on the type of the Surv object. This is given by attr(s, which="type") :

counting	Lines with an arrow pointing right if right censored
right	Lines with an arrow pointing right if right censored
left	Lines with an arrow pointing left if left censored
interval	If censored: • Lines with an arrow pointing right if right censored. • Lines with an arrow pointing left if left censored. If not censored: • Lines if observations of more than one time point • Points if observation of one time only (i.e. start and end times are the same)

Examples

```
df0 <- data.frame(t1=c(0, 2, 4, 6, NA, NA, 12, 14),
                  t2=c(NA, NA, 4, 6, 8, 10, 16, 18))
s5 <- Surv(df0$t1, df0$t2, type="interval2")
plot(s5)
```

---

plotTerm — *Plot individual terms of a Generalized Additive* gam *or Cox Proportional Hazards* coxph *Model*

Description

Plot individual terms of a Generalized Additive gam or Cox Proportional Hazards coxph Model

Usage

```
plotTerm(x, term = 1, se = TRUE, p = 0.95, rug = TRUE, const = 0,
  col = 1, xlab = NULL, data, ...)
```

Arguments

x	The result of a Generalized Additive Model (gam) or a Cox proportional hazards model (coxph)
term	The term to be plotted. An integer, based on the position of the term on the right-hand side of the model formula e.g. the first term is 1.
se	If se=TRUE (the default), also plot confidence intervals based on the standard errors
p	P-value used to plot the confidence intervals from standard errors. Based on the normal distribution
rug	Add rug (1-dimensional plot) to x-axis. See ?graphics::rug
const	Value for constant term. If const=TRUE, add the overall mean for the decomposition as returned by gamTerms
col	Color of line(s) on plot. If se=TRUE, use a vector of 3 colors: the first is the main line, the second is lower the CI, the third is the upper CI.
data	A data.frame in which to evaluate the model. If missing, eval(x$call$data) is used.
xlab	Label for x-axis.
...	Additional arguments; passed to graphics::matplot (with standard errors) or graphics::plot (without).

Value

A plot (base graphics) of the term in question. If se=TRUE, this is done using graphics::matplot otherwise graphics::plot.default is used.

Author(s)

Terry Therneau. Updated from S-plus by Chris Dardis

See Also

gamTerms, mpip

Examples

```
fit1 <- coxph(Surv(time, status) ~ sex + pspline(age), data=lung)
plotTerm(fit1, term=2, rug=FALSE, ylab="Log-hazard",
         col=c("blue", "red", "red"))
### smoothing splines
data(mpip)
c1 <- coxph(Surv(futime, status) ~
            pspline(lved) + factor(nyha) + rales + pspline(ef),
            data=mpip)
plotTerm(c1, 4, ylab="Log-hazard", xlab="Ejection fraction (%)",
       main="Log-hazard by ejection fraction \n Line fitted by penalized smoothing splines")
```

profLik *Profile likelihood for coefficients in Coxph model*

Description

Profile likelihood for coefficients in Coxph model

Usage

```
profLik(x, CI = 0.95, interval = 50, mult = c(0.1, 2), ...)
```

Arguments

x	A coxph model
CI	Confidence Interval
interval	Number of points over which to evaluate coefficient
mult	Multiplier. Coefficent will be multiplied by lower and upper value and evaluated across this range
...	Additional parameters passed to graphics::plot.default.

Details

Plots of range of values for coefficient in model with log-likelihoods for the model with the coefficient fixed at these values.

For each coefficient a range of possible values is chosen, given by $\hat{B} * mult_{lower} - \hat{B} * mult_{upper}$. A series of model are fit (given by interval). The coefficient is included in the model as a *fixed* term and the partial log-likelihood for the model is calculated.

A curve is plotted which gives the partial log-likelihood for each of these candidate values. An appropriate confidence interval (CI) is given by subtracting 1/2 the value of the approapriate quantile of a chi-squared distribution with 1 degree of freedom.

Two circles are also plotted giving the 95

Value

One plot for each coefficient in the model.

References

Example is from: Therneau T, Grambsch P 2000. *Modeling Survival Data*, 1st edition. New York: Springer. Section 3.4.1, pg 57.

Examples

```
c1 <- coxph(formula = Surv(time, status == 2) ~ age + edema + log(bili) +
                      log(albumin) + log(protime), data = pbc)
profLik(c1, col="red")
```

---

quantile *Quantiles and median for* Surv, survfit *and* coxph *objects*

Description

Extends stats::quantile and stats::quantile to work with Surv, survfit and coxph objects.

Usage

```
quantile(x, ...)

## S3 method for class 'Surv'
quantile(x, ..., q = c(25, 50, 75), CI = TRUE,
  alpha = 0.05, ci = c("log", "lin", "asr"))

## S3 method for class 'survfit'
quantile(x, ..., q = c(25, 50, 75), CI = TRUE,
  alpha = 0.05, ci = c("log", "lin", "asr"))

## S3 method for class 'coxph'
quantile(x, ..., q = c(25, 50, 75), CI = TRUE,
  alpha = 0.05, ci = c("log", "lin", "asr"))

median(x, ...)

## S3 method for class 'Surv'
median(x, ..., CI = FALSE, alpha = 0.05, ci = c("log",
  "lin", "asr"))

## S3 method for class 'survfit'
median(x, ..., CI = FALSE, alpha = 0.05, ci = c("log",
  "lin", "asr"))

## S3 method for class 'coxph'
median(x, ..., CI = FALSE, alpha = 0.05, ci = c("log",
  "lin", "asr"))
```

Arguments

x	A Surv, survfit or coxph object.
...	Additional arguments (not implemented).

q	(for quantile) Vector of quantiles (expressed as percentage). For the median, q=50.
CI	Include confidence interval. Defaults are CI=TRUE for quantile and CI=FALSE for median.
alpha	Significance level α.
ci	**C**onfidence **i**nterval. One of: **log** (the default), **lin**ear or **a**rcsine-**s**quare **r**oot.

Value

For quantile: A data.table (or a list of data.tables, one per stratum), with columns:

q	quantile
t	time

If CI = TRUE then upper and lower confidence intervals, as per argument ci).

l	lower confidence limit
u	upper confidence limit

For median: A data.table with columns:

t	time
s	stratum

If CI = TRUE then a list of data.tables, one per stratum, as above.

Note

If a time cannot be calculated, NaN is returned.

References

Examples for quantiles are from: Klein J, Moeschberger M 2003 *Survival Analysis*, 2nd edition. New York: Springer. Example 4.2, pg 121.

See Also

Confidence intervals are calculated as shown in the pointwise confidence intervals in ci.

Examples

```
data(bmt, package="KMsurv")
b1 <- bmt[bmt$group==1, ] # ALL patients
s1 <- Surv(time=b1$t2, event=b1$d3)
quantile(s1)
b1 <- bmt[bmt$group==2, ] # AML low-risk patients
s1 <- Surv(time=b1$t2, event=b1$d3)
quantile(s1)
b1 <- bmt[bmt$group==3, ] # AML high-risk patients
s1 <- Surv(time=b1$t2, event=b1$d3)
```

```
quantile(s1)
###
s1 <- survfit(Surv(t2, d3) ~ group, data=bmt)
quantile(s1)
c1 <- coxph(Surv(t2, d3)~ group, data=bmt)
quantile(c1)
b1 <- bmt[bmt$group==1, ] # ALL patients
s1 <- Surv(time=b1$t2, event=b1$d3)
median(s1)
median(s1, CI=TRUE)
data(bmt, package="KMsurv")
b1 <- bmt[bmt$group==1, ] # ALL patients
s1 <- survfit(Surv(t2, d3)~ group, data=bmt)
median(s1)
median(s1, ci="asr", CI=TRUE)
c1 <- coxph(Surv(t2, d3) ~ group, data=bmt)
median(c1)
```

---

rsq *r^2 measures for a a* coxph *or* survfit *model*

Description

r^2 measures for a a coxph or survfit model

Usage

```
rsq(x, ...)

## S3 method for class 'coxph'
rsq(x, ..., sigD = 2)

## S3 method for class 'survfit'
rsq(x, ..., sigD = 2)
```

Arguments

x	A survfit or coxph object
...	Additional arguments (not implemented)
sigD	**Sig**nificant **d**igits (for ease of display). If sigD=NULL will return the original numbers.

Value

A list with the following elements:

`cod` The **c**oefficient **o**f **d**etermination, which is

$$R^2 = 1 - \exp(\frac{2}{n}L_0 - L_1)$$

where L_0 and L_1 are the log partial likelihoods for the *null* and *full* models respectively and n is the number of observations in the data set.

`mer` The **m**easure of **e**xplained **r**andomness, which is:

$$R^2_{mer} = 1 - \exp(\frac{2}{m}L_0 - L_1)$$

where m is the number of observed *events*.

`mev` The **m**easure of **e**xplained **v**ariation (similar to that for linear regression), which is:

$$R^2 = \frac{R^2_{mer}}{R^2_{mer} + \frac{\pi}{6}(1 - R^2_{mer})}$$

References

Nagelkerke NJD, 1991. A Note on a General Definition of the Coefficient of Determination. *Biometrika* **78**(3):691–92. JSTOR

O'Quigley J, Xu R, Stare J, 2005. Explained randomness in proportional hazards models. *Stat Med* **24**(3):479–89. Wiley (paywall) Available at UCSD

Royston P, 2006. Explained variation for survival models. *The Stata Journal* **6**(1):83–96. The Stata Journal

---

`sf` *Estimates of **s**urvival (or hazard) **f**unction based on n and e*

Description

Estimates of **s**urvival (or hazard) **f**unction based on n and e

Usage

```
sf(n, e, what = c("all", "s", "sv", "h", "hv"))
```

Arguments

`n`	**N**umber at risk per time point (a `vector`)
`e`	Number of **e**vents per time point (a `vector`)
`what`	See return, below

Value

The return value will be a `vector`, unless `what="all"` (the default), in which case it will be a `data.table`.
If `what="s"`, the **s**urvival is returned, based on the Kaplan-Meier or product-limit estimator. This is 1 at $t = 0$ and thereafter is given by:

$$\hat{S}(t) = \prod_{t \leq t_i} (1 - \frac{e_i}{n_i})$$

If `what="sv"`, the **s**urvival **v**ariance is returned.
Greenwoods estimtor of the variance of the Kaplan-Meier (product-limit) estimator is:

$$Var[\hat{S}(t)] = [\hat{S}(t)]^2 \sum_{t_i \leq t} \frac{e_i}{n_i(n_i - e_i)}$$

If `what="h"`, the **h**azard is returned, based on the the Nelson-Aalen estimator. This has a value of $\hat{H} = 0$ at $t = 0$ and thereafter is given by:

$$\hat{H}(t) = \sum_{t \leq t_i} \frac{e_i}{n_i}$$

If `what="hv"`, the **h**azard **v**ariance is returned.
The variance of the Nelson-Aalen estimator is given by:

$$Var[\hat{H}(t)] = \sum_{t_i \leq t} \frac{e_i}{n_i^2}$$

If `what="all"` (the default), *all* of the above are returned in a `data.table`, along with:
Survival, based on the Nelson-Aalen estimator. Given by

$$\hat{S_{na}} = e^H$$

where H is hazard. HKM Hazard, based on the Kaplan-Meier estimator. Given by

$$\hat{H_{km}} = -\log S$$

where S is survival.

References

Examples for are from: **K&M**.
`what="sv"`: Table 4.1A, pg 93.
`what="hv"`: Table 4.2, pg 94.
`what="all"`: Table 4.3, pg 97.

Examples

```
data(bmt, package="KMsurv")
b1 <- bmt[bmt$group==1, ] # ALL patients
t1 <- tne(Surv(time=b1$t2, event=b1$d3))
sf(n=t1$n, e=t1$e, what="all")
###
data(drug6mp, package="KMsurv")
s1 <- Surv(time=drug6mp$t2, event=drug6mp$relapse) # 6MP patients
t1 <- tne(s1)
sf(n=t1$n, e=t1$e, what="sv")
sf(n=t1$n, e=t1$e, what="hv")
```

---

sig *Significiance tests of coefficients in a* coxph *model*

Description

Significiance tests of coefficients in a coxph model

Usage

```
sig(x, ...)

## S3 method for class 'coxph'
sig(x, ...)
```

Arguments

x	A model of class coxph
...	Additional arguments (not implemented)

Value

A data.frame with one row for each coefficient in the original model. There are three columns, one for each of the tests:

Wald	the statistic is: $$\frac{\hat{B}}{\hat{SE}}$$ where $\hat{B}$ is the estimate of the coefficient and $\hat{SE}$ is its standard error.
plr	**P**artial **l**ikelihood **r**atio test. The statistic is the difference in the likelihood ratio of the original model and that with the coefficient omitted.
lrt	Aka the **score** test. The Null hypothesis is that $\hat{B} = 0$. The statistic is cacluated by refitting the model with the coefficient omitted, to generate initial values for the other $\hat{B}$s. It is then fitted again with all covariates, using these values and setting $\hat{B} = 0$.

All statistics are distributed as χ-square, with degrees of freedom = no. of coefficients -1.

`tableRhs` *Table the outcome against all predictors in a formula*

Description

Table the outcome against all predictors in a formula

Usage

```
tableRhs(formula = y ~ ., data = parent.frame(), return = c("summary",
  "zeros", "zEq", "counts", "all"), nlf = 2)
```

Arguments

`formula`	A formula. Works with formulas where the left-hand side is a `Surv` object describing right-censored data.
`data`	A `data.frame`.
`return`	See **Value** below.
`nlf`	Number of levels defining a factor. Predictors with $> nlf$ levels are considered continuous and are not tabulated. Needs to be less than the number of observations (rows) in the model specified by the `formula`.

Details

Cross-tabulation of outcomes against levels of a predictor.
This is a useful step prior to fitting survival models where the outcome has limited values.

Value

- If `return="summary"` (the default), a `table` with one row per predictor and three columns:

  zeros at least one zero present
  someEq outcomes equal for least *some* levels of the predictor
  allEq outcomes equal for *all* levels of the predictor

Other values return a `list` of `tables`. Each element is named after the predictor.

- If `return="zeros"`, one `table` for each predictor with a least one zero present. Each `table` shows only those levels of the predictor for which one level of the outcome is zero.
- If `return="zEq"`, one `table` for each predictor with a least one **z**ero present or one level which has **eq**ual outcomes. Each `table` shows only those levels where one of the above apply.
- If `return="counts"`, each `table` gives the total number of levels where zeros and equal outcomes are present and absent.
- If `return="all"`, a list of `tables` of outcomes for *all* levels of each predictor.

Examples

```
## Not run:
set.seed(1)
d1 <- genSurvDf(c=3, rc=0.5, model=FALSE)
tableRhs(Surv(t1, e) ~ ., data=d1, return="summary", nlf=2)
t1 <- tableRhs(Surv(t1, e) ~ ., data=d1, return="c", nlf=99)
### simple graph
p <- par()
par( mfrow=c(2, 2))
for (i in 1:length(t1)){
    graphics::mosaicplot(t1[[i]], main="", cex=1.5)
    title(main=list(names(t1[i]), cex=3))
}
par <- p
set.seed(2)
d1 <- genSurvDf(f=1, n=10, model=FALSE)
t1 <- tableRhs(Surv(t1, e) ~ x1, nlf=9, data=d1)
tableRhs(e ~ x1, nlf=9, r="zEq", data=d1)
tableRhs(e ~ ., nlf=3, r="c", data=d1)

## End(Not run)
```

---

tne *Time, No. at risk, No. events*

Description

Time, No. at risk, No. events

Usage

```
tne(x, ...)

## S3 method for class 'Surv'
tne(x, ..., eventsOnly = FALSE)

## S3 method for class 'survfit'
tne(x, ..., eventsOnly = FALSE, what = c("table", "list",
  "all"), nameStrata = TRUE)

## S3 method for class 'coxph'
tne(x, ..., eventsOnly = FALSE, what = c("table", "list",
  "all"), nameStrata = TRUE)

## S3 method for class 'formula'
tne(x, ..., eventsOnly = FALSE, what = c("table", "list",
  "all"), nameStrata = TRUE)
```

Arguments

x	A object of class Surv, survfit, coxph or formula.
...	Additional arguments (not implemented)
eventsOnly	If eventsOnly=TRUE shows only times at which at least one event occurred. Otherwise shows *all* times recorded (i.e. including those censored)
what	See **Value** below
nameStrata	Applies only if what=="list" or what=="all". The default is to name the elements of the list after each stratum. As the names for each stratum are made by concatenating the predictor names, this can become unwieldly. If nameStrata="FALSE" they are instead numbered. A list is returned with the numbered list or data.table and a vector giving the names of the strata.

Value

For a Surv object: A data.table with columns:

t	time
n	no. at risk
e	no. events

For a survfit, coxph or formula: If what="table" (the default), a data.table with columns as above. In addition:

s	stratum; predictor names are separated with an underscore '_'
ns	no. at risk (by strata)
Es	no. events expected (by strata)
e_Es	no. events minus no. events expected

Additional columns returned match those of the predictors in the model.frame (for survfit objects) or model.matrix (in other cases). If what="list" = then instead a list with one element for each stratum, where each elements is a data.table with columns **t**, **n** and **e** as for a Surv object. If what="all", a data.table with a columns **t**, **n** and **e** as above. There are additional columns for **n** and **e** for each stratum.

Note

The number of events expected (per stratum) is given by:

$$E = \frac{e_i(n[s]_i)}{n_i}$$

where $n[s]_i$ is the no. at risk for the stratum.

If the formula is 'intercept-only', the stratum I=1 is returned.

Interaction terms are not currently supported by survfit objects.

References

Example using kidney data is from: **K&M**. Example 7.2, pg 210.

Examples

```
### Surv object
df0 <- data.frame(t=c(1,1,2,3,5,8,13,21),
                  e=rep(c(0,1),4))
s1 <- Surv(df0$t, df0$e, type="right")
tne(s1)
tne(s1, eventsOnly=TRUE)
### survfit object
data(kidney, package="KMsurv")
s1 <- survfit(Surv(time=time, event=delta) ~ type, data=kidney)
tne(s1)
tne(s1, what="all")
tne(s1, what="all", eventsOnly=TRUE)
tne(survfit(Surv(time=time, event=delta) ~ 1, data=kidney))
data(larynx, package="KMsurv")
tne(survfit(Surv(time, delta) ~ factor(stage) + age, data=larynx))
data(bmt, package="KMsurv")
tne(survfit(Surv(t2, d3) ~ z3 +z10, data=bmt), what="all")
tne(survfit(Surv(t2, d3) ~ 1, data=bmt))
### coxph object
data(kidney, package="KMsurv")
c1 <- coxph(Surv(time=time, event=delta) ~ type, data=kidney)
tne(c1)
tne(c1, what="list")
tne(coxph(Surv(t2, d3) ~ z3*z10, data=bmt))
### formula object
data(kidney, package="KMsurv")
### this doesn't work
### s1 <- survfit(Surv(t2, d3) ~ z3*z10, data=bmt)
tne(Surv(time=t2, event=d3) ~ z3*z10, data=bmt, what="all")
tne(Surv(time=t2, event=d3) ~ ., data=bmt)
### example where each list element has only one row
### also names are impractical
tne(Surv(time=t2, event=d3) ~ ., data=bmt, what="list", nameStrata=FALSE)
```

---

tneBMT *Time, no. at risk, no. events for BMT data*

Description

Time, no. at risk, no. events for BMT data

Format

A data.frame with 76 rows and 9 columns.

Details

Data on survival time following bone-marrow transplant.

Columns are:

t time

n_1 no. at risk in group 1 (ALL)

e_1 no. events in group 1

n_2 no. at risk in group 2 (AML low-risk)

e_2 no. events in group 2

n_3 no. at risk in group 3 (AML high-risk)

e_3 no. events in group 3

n no. at risk overall

e no. events overall

Source

Generated from `data("bmt", package="KMsurv")`.

- Time (`bmt$t2`) is disease free survival time.
- Event (`bmt$d3`) is death or relapse.

K&M. Example 7.9, pg 224.

References

Copelan EA, Biggs JC, Thompson JM, Crilley P, Szer J, Klein JP, Kapoor N, Avalos BR, Cunningham I, Atkinson K, et al 1991. Treatment for acute myelocytic leukemia with allogeneic bone marrow transplantation following preparation with BuCy2. *Blood*. **78**(3):838-43. .pdf at Blood

See Also

`?KMsurv::bmt`
comp

`tneKidney` *Time, no. at risk, no. events for kidney data.*

Description

Data from Klein J and Moeschberger.
Columns are:

t time

n_1 no. at risk in group 1

e_1 no. events in group 1

n_2 no. at risk in group 2

e_2 no. events in group 2

n no. at risk overall

e no. events overall

Format

A data frame with 16 rows and 7 columns

Source

K&M. Example 7.9, pg 224.

`whas100` *Worcester Heart Attack Study WHAS100 Data*

Description

Worcester Heart Attack Study WHAS100 Data

Format

A data frame with 100 rows and 9 columns. All columns are `integer`, apart from **admitdate** and **foldate** which are `date`, and **bmi** which is `numeric`.

Details

The main goal of this study is to describe factors associated with trends over time in the incidence and survival rates following hospital admission for acute myocardial infarction (MI). Data have been collected during thirteen 1-year periods beginning in 1975 and extending through 2001 on all MI patients admitted to hospitals in the Worcester, Massachusetts Standard Metropolitan Statistical Area.
Columns are:

id ID code

admitdate Admission Date

foldate Follow Up Date

los Length of Hospital Stay (days)

lenfol Follow Up Time (days)

fstat Follow Up Status

1 dead

0 alive

age Age (years)

gender **0** male

0 female

bmi Body Mass Index

Source

Wiley FTP.

References

Hosmer D, Lemeshow S, May S. *Applied Survival Analysis: Regression Modeling of Time to Event Data, 2nd edition*. John Wiley and Sons Inc., New York, NY, 2008. Wiley (paywall)

`whas500` *Worcester Heart Attack Study WHAS500 Data*

Description

Worcester Heart Attack Study WHAS500 Data

Format

A data frame with 500 rows and 22 columns. All columns are `integer`, apart from admitdate, disdate and fdate which are `date` and bmi which is `numeric`.

Details

This is a more complete version of the WHAS100 dataset.
Columns are:

id ID code

age Age at hospital admission (years)

gender Gender

0 male

1 female

hr Initial heart rate (bpm)

sysbp Initial systolic blood pressure (mmHg)

diasbp Initial diastolic blood pressure (mmHg)

bmi Body mass index (kg/m^2)

cvd History of cardiovascular disease

- **0** no
- **1** yes

afib Atrial fibrillation

- **0** no
- **1** yes

sho Cardiogenic shock

- **0** no
- **1** yes

chf Congestive heart failure

- **0** no
- **1** yes

av3 3rd degree AV block (complete heart block) disease

- **0** no
- **1** yes

miord MI order

- **0** first
- **1** recurrent

mitype MI type

- **0** non Q-wave
- **1** Q-wave

year Cohort year

- **1** 1997
- **2** 1999
- **3** 2001

admitdate Hospital admission date

disdate Hospital discharge date

fdate Date of last follow up

los Length of hospital stay. Days between admission and discharge

dstat Discharge status

- **0** alive
- **1** dead

lenfol Length of follow up. Days between admission and last follow-up. ##'

fstat Follow-up status

- **0** alive
- **1** dead

Source

Wiley FTP.

References

Hosmer D, Lemeshow S, May S. *Applied Survival Analysis: Regression Modeling of Time to Event Data, 2nd edition*. John Wiley and Sons Inc., New York, NY, 2008. Wiley (paywall)

See Also

whas100

Index