The survival Package

October 16, 2007

**Title**  Survival analysis, including penalised likelihood.

**Maintainer**  Thomas Lumley <tlumley@u.washington.edu>

**Priority**  recommended

**Version**  2.34

**Depends**  stats, utils, graphics, splines, R (>= 2.0.0)

**LazyData**  Yes

**LazyLoad**  Yes

**Author**  S original by Terry Therneau, ported by Thomas Lumley

**Description**  survival analysis: descriptive statistics, two-sample tests, parametric accelerated failure models, Cox model. Delayed entry (truncation) allowed for all models; interval censoring for parametric models. Case-cohort designs.

**License**  GPL2

**R topics documented:**

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Surv</td>
<td>3</td>
</tr>
<tr>
<td>aml</td>
<td>4</td>
</tr>
<tr>
<td>anova.coxph</td>
<td>5</td>
</tr>
<tr>
<td>as.date</td>
<td>6</td>
</tr>
<tr>
<td>attrassign</td>
<td>7</td>
</tr>
<tr>
<td>bladder</td>
<td>8</td>
</tr>
<tr>
<td>cch</td>
<td>8</td>
</tr>
<tr>
<td>clogit</td>
<td>11</td>
</tr>
<tr>
<td>cluster</td>
<td>12</td>
</tr>
<tr>
<td>colon</td>
<td>13</td>
</tr>
<tr>
<td>cox.zph</td>
<td>13</td>
</tr>
<tr>
<td>coxph</td>
<td>15</td>
</tr>
<tr>
<td>coxph.detail</td>
<td>17</td>
</tr>
<tr>
<td>coxph.object</td>
<td>19</td>
</tr>
<tr>
<td>R topics documented</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>date.ddmmyyy</td>
<td>20</td>
</tr>
<tr>
<td>date.mdy</td>
<td>20</td>
</tr>
<tr>
<td>date.mmddyy</td>
<td>21</td>
</tr>
<tr>
<td>date.mmddyyyy</td>
<td>22</td>
</tr>
<tr>
<td>date.object</td>
<td>22</td>
</tr>
<tr>
<td>frailty</td>
<td>23</td>
</tr>
<tr>
<td>heart</td>
<td>24</td>
</tr>
<tr>
<td>is.ratetable</td>
<td>25</td>
</tr>
<tr>
<td>kidney</td>
<td>26</td>
</tr>
<tr>
<td>lines.survfit</td>
<td>27</td>
</tr>
<tr>
<td>lung</td>
<td>29</td>
</tr>
<tr>
<td>mdy.date</td>
<td>30</td>
</tr>
<tr>
<td>nwtco</td>
<td>31</td>
</tr>
<tr>
<td>ovarian</td>
<td>32</td>
</tr>
<tr>
<td>pbc</td>
<td>32</td>
</tr>
<tr>
<td>plot.cox.zph</td>
<td>33</td>
</tr>
<tr>
<td>plot.survfit</td>
<td>34</td>
</tr>
<tr>
<td>predict.coxph</td>
<td>37</td>
</tr>
<tr>
<td>predict.survreg</td>
<td>38</td>
</tr>
<tr>
<td>print.survfit</td>
<td>39</td>
</tr>
<tr>
<td>pspline</td>
<td>40</td>
</tr>
<tr>
<td>pyears</td>
<td>41</td>
</tr>
<tr>
<td>ratetable</td>
<td>44</td>
</tr>
<tr>
<td>rats</td>
<td>44</td>
</tr>
<tr>
<td>residuals.coxph</td>
<td>45</td>
</tr>
<tr>
<td>residuals.survreg</td>
<td>47</td>
</tr>
<tr>
<td>ridge</td>
<td>48</td>
</tr>
<tr>
<td>stanford2</td>
<td>49</td>
</tr>
<tr>
<td>strata</td>
<td>50</td>
</tr>
<tr>
<td>summary.survfit</td>
<td>51</td>
</tr>
<tr>
<td>survSplit</td>
<td>52</td>
</tr>
<tr>
<td>survdiff</td>
<td>53</td>
</tr>
<tr>
<td>survexp</td>
<td>55</td>
</tr>
<tr>
<td>survexp.fit</td>
<td>57</td>
</tr>
<tr>
<td>ratetables</td>
<td>58</td>
</tr>
<tr>
<td>survfit</td>
<td>59</td>
</tr>
<tr>
<td>survfit.object</td>
<td>62</td>
</tr>
<tr>
<td>survival-internal</td>
<td>63</td>
</tr>
<tr>
<td>survobrien</td>
<td>64</td>
</tr>
<tr>
<td>survreg</td>
<td>65</td>
</tr>
<tr>
<td>survreg.control</td>
<td>67</td>
</tr>
<tr>
<td>survreg.distributions</td>
<td>68</td>
</tr>
<tr>
<td>survreg.object</td>
<td>70</td>
</tr>
<tr>
<td>survreg.old</td>
<td>71</td>
</tr>
<tr>
<td>tcut</td>
<td>71</td>
</tr>
<tr>
<td>tobin</td>
<td>72</td>
</tr>
<tr>
<td>untangle.specials</td>
<td>73</td>
</tr>
<tr>
<td>veteran</td>
<td>74</td>
</tr>
</tbody>
</table>
Surv

Index

Surv

Create a Survival Object

Description

Create a survival object, usually used as a response variable in a model formula.

Usage

Surv(time, time2, event, type =, origin = 0)
is.Surv(x)

Arguments

time
  for right censored data, this is the follow up time. For interval data, the first argument is the starting time for the interval.
x
  any R object.
event
  The status indicator, normally 0=alive, 1=dead. Other choices are TRUE/FALSE (TRUE = death) or 1/2 (2=death). For interval censored data, the status indicator is 0=right censored, 1=event at time, 2=left censored, 3=interval censored. Although unusual, the event indicator can be omitted, in which case all subjects are assumed to have an event.
time2
  ending time of the interval for interval censored or counting process data only. Intervals are assumed to be open on the left and closed on the right, [start, end]. For counting process data, event indicates whether an event occurred at the end of the interval.
type
  character string specifying the type of censoring. Possible values are "right", "left", "counting", "interval", or "interval2". The default is "right" or "counting" depending on whether the time2 argument is absent or present, respectively.
origin
  for counting process data, the hazard function origin. This is most often used in conjunction with a model containing time dependent strata in order to align the subjects properly when they cross over from one strata to another.

Details

Typical usages are

Surv(time, event)
Surv(time, time2, event, type =, origin = 0)

In theory it is possible to represent interval censored data without a third column containing the explicit status. Exact, right censored, left censored and interval censored observation would be represented as intervals of (a,a), (a, infinity), (-infinity,b), and (a,b) respectively; each specifying the interval within which the event is known to have occurred.
If \texttt{type = "interval2"} then the representation given above is assumed, with NA taking the place of infinity. If \texttt{type="interval" event} must be given. If \texttt{event} is 0, 1, or 2, the relevant information is assumed to be contained in \texttt{time}, the value in \texttt{time2} is ignored, and the second column of the result will contain a placeholder.

Presently, the only methods allowing interval censored data are the parametric models computed by \texttt{survreg}, so the distinction between open and closed intervals is unimportant. The distinction is important for counting process data and the Cox model.

The function tries to distinguish between the use of 0/1 and 1/2 coding for left and right censored data using \texttt{if (max(status)==2)}. If 1/2 coding is used and all the subjects are censored, it will guess wrong. Use 0/1 coding in this case.

\textbf{Value}

An object of class \texttt{Surv}. There are methods for \texttt{print}, \texttt{is.na}, and subscripting survival objects. \texttt{Surv} objects are implemented as a matrix of 2 or 3 columns.

In the case of \texttt{is.Surv}, a logical value \texttt{TRUE} if \texttt{x} inherits from class \texttt{"Surv"}, otherwise an \texttt{FALSE}.

\textbf{See Also}

\texttt{coxph, survfit, survreg}.

\textbf{Examples}

\begin{verbatim}
with(aml, Surv(time, status))
with(heart, Surv(start,stop,event))
\end{verbatim}

\textbf{aml} \hspace{1cm} \textit{Acute Myelogenous Leukemia survival data}

\textbf{Description}

Survival in patients with Acute Myelogenous Leukemia

\textbf{Usage}

\begin{verbatim}
aml leukemia
\end{verbatim}

\textbf{Format}

\begin{verbatim}
time: survival or censoring time
status: censoring status
x: maintenance chemotherapy given? (factor)
\end{verbatim}
### `anova.coxph`

**Source**


---

### Description

Compute an analysis of deviance table for one Cox model fit.

#### Usage

```r
## S3 method for class 'coxph':
anova(object, ..., test = NULL)
```

#### Arguments

- **object**: An object of class `coxph`
- **...**: Further `coxph` objects
- **test**: A character string, (partially) matching one of "Chisq", "F" or "Cp". See `stat.anova`.

#### Details

Specifying a single object gives a sequential analysis of deviance table for that fit. That is, the reductions in the residual deviance as each term of the formula is added in turn are given in as the rows of a table, plus the residual deviances themselves.

If more than one object is specified, the table has a row for the residual degrees of freedom and deviance for each model. For all but the first model, the change in degrees of freedom and deviance is also given. (This only make statistical sense if the models are nested.) It is conventional to list the models from smallest to largest, but this is up to the user.

The table will optionally contain test statistics (and P values) comparing the reduction in deviance for the row to the residuals. Mallows’ $C_p$ statistic is the residual deviance plus twice the residual degrees of freedom, which is closely related to AIC. Don’t use `test=F`, it doesn’t make sense.

#### Value

An object of class "anova" inheriting from class "data.frame".

#### Warning

The comparison between two or more models by `anova` or will only be valid if they are fitted to the same dataset. This may be a problem if there are missing values.

#### See Also

`coxph, anova`. 
Examples

```r
fit <- coxph(Surv(futime, fustat) ~ resid.ds * rx + ecog.ps, data = ovarian)
anova(fit)
anova(fit, test = "Chisq")
fit2 <- coxph(Surv(futime, fustat) ~ resid.ds + rx + ecog.ps, data = ovarian)
anova(fit2, fit)
```

---

**as.date**

*Coerce Data to Dates*

**Description**

Converts any of the following character forms to a Julian date: 8/31/56, 8-31-1956, 31 8 56, 083156, 31Aug56, or August 31 1956.

**Usage**

```r
as.date(x, order = "mdy", ...)
```

**Arguments**

- `x`: input data vector.
- `order`: if `x` is character, defines the order in which the terms are assumed to appear in a `xx/xx/xx` date. The default is month/day/year; any permutation of mdy is legal.
- `...`: if `x` is character, then any other arguments from `mdy.date()` can be used as well.

**Details**

If `x` is numeric, then `floor(x)` is returned, e.g., `as.date(35)` is the same as `as.date(35.2)` and gives Feb 2, 1960. If `x` is character, the program attempts to parse it.

**Value**

For each date, the number of days between it and January 1, 1960. The date will be missing if the string is not interpretable.

**See Also**

`mdy.date`, `date.mmddyy`, `date.ddmmmyy`

**Examples**

```r
as.date(c("1jan1960", "2jan1960", "31mar1960", "30jul1960"))
```
Description

The "assign" attribute on model matrices describes which columns come from which terms in the model formula. It has two versions. R uses the original version, but the newer version is sometimes useful.

Usage

## Default S3 method:
attrassign(object, tt,...)
## S3 method for class 'lm':
attrassign(object,...)

Arguments

- **object**: model matrix or linear model object
- **tt**: terms object
- **...**: ignored

Value

A list with names corresponding to the term names and elements that are vectors indicating which columns come from which terms

See Also

terms, model.matrix

Examples

```r
formula<-Surv(time,status)~factor(edtrt)
tt<-terms(formula)
mf<-model.frame(tt,data=pbc)
mm<-model.matrix(tt,mf)
## a few rows of data
mm[1:3,]
## old-style assign attribute
attr(mm,"assign")
## new-style assign attribute
attrassign(mm,tt)
```
**bladder**

**Bladder Cancer Recurrences**

**Description**

Data on recurrences of bladder cancer, used by many people to demonstrate methodology for recurrent event modelling.

**Usage**

bladder

**Format**

bladder

id: Patient id
rx: Treatment 1=placebo 2=thiotepa
number: Initial number of tumours (8=8 or more)
size: size (cm) of largest initial tumour
stop: recurrence or censoring time
enum: which recurrence (up to 4)

bladder2

id: Patient id
rx: Treatment 1=placebo 2=thiotepa
number: Initial number of tumours (8=8 or more)
size: size (cm) of largest initial tumour
start: start of interval (0 or previous recurrence time)
stop: recurrence or censoring time
enum: which recurrence (up to 4)

**Source**


**cch**

*Fits proportional hazards regression model to case-cohort data*

**Description**

Returns estimates and standard errors from relative risk regression fit to data from case-cohort studies. A choice is available among the Prentice, Self-Prentice and Lin-Ying methods for unstratified
data. For stratified data the choice is between Borgan I, a generalization of the Self-Prentice estimator for unstratified case-cohort data, and Borgan II, a generalization of the Lin-Ying estimator.

Usage

```r
cch(formula, data = sys.parent(), subcoh, id, stratum=NULL, cohort.size, method =c("Prentice","SelfPrentice","LinYing","I.Borgan","II.Borgan"), robust=FALSE)
```

Arguments

- `formula`: A formula object that must have a `Surv` object as the response. The `Surv` object must be of type "right", or of type "counting".
- `subcoh`: Vector of indicators for subjects sampled as part of the sub-cohort. Code 1 or TRUE for members of the sub-cohort, 0 or FALSE for others. If `data` is a data frame then `subcoh` may be a one-sided formula.
- `id`: Vector of unique identifiers, or formula specifying such a vector.
- `stratum`: A vector of stratum indicators or a formula specifying such a vector.
- `cohort.size`: Vector with size of each stratum original cohort from which subcohort was sampled.
- `data`: An optional data frame in which to interpret the variables occurring in the formula.
- `method`: Three procedures are available. The default method is "Prentice", with options for "SelfPrentice" or "LinYing".
- `robust`: For "LinYing" only, if robust=TRUE, use design-based standard errors even for phase I.

Details

Implements methods for case-cohort data analysis described by Therneau and Li (1999). The three methods differ in the choice of "risk sets" used to compare the covariate values of the failure with those of others at risk at the time of failure. "Prentice" uses the sub-cohort members "at risk" plus the failure if that occurs outside the sub-cohort and is score unbiased. "SelfPren" (Self-Prentice) uses just the sub-cohort members "at risk". These two have the same asymptotic variance-covariance matrix. "LinYing" (Lin-Ying) uses the all members of the sub-cohort and all failures outside the sub-cohort who are "at risk". The methods also differ in the weights given to different score contributions.

The `data` argument must not have missing values for any variables in the model. There must not be any censored observations outside the subcohort.

Value

An object of class "cch" incorporating a list of estimated regression coefficients and two estimates of their asymptotic variance-covariance matrix.

- `coef`: regression coefficients.
- `naive.var`: Self-Prentice model based variance-covariance matrix.
- `var`: Lin-Ying empirical variance-covariance matrix.
Author(s)

Norman Breslow, modified by Thomas Lumley

References


See Also
twophase and svycoxph in the "survey" package for more general two-phase designs. http://faculty.washington.edu/tlumley/survey/

Examples

```r
## The complete Wilms Tumor Data
## (Breslow and Chatterjee, Applied Statistics, 1999)
## subcohort selected by simple random sampling.
##
## subcoh <- nwtco$in.subcohort
## selccoh <- with(nwtco, rel==1|subcoh==1)
## ccoh.data <- nwtco[selccoh,]
## ccoh.data$subcohort <- subcoh[selccoh]
##
## ## central-lab histology
## ccoh.data$histol <- factor(ccoh.data$histol,labels=c("FH","UH"))
## ## tumour stage
## ccoh.data$stage <- factor(ccoh.data$stage,labels=c("I","II","III","IV"))
## ccoh.data$age <- ccoh.data$age/12 # Age in years
##
## ## Standard case-cohort analysis: simple random subcohort
##
## fit.ccP <- cch(Surv(edrel, rel) ~ stage + histol + age, data =ccoh.data, subcoh = ~subcohort, id=~seqno, cohort.size=4028)
##
## fit.ccP
```


fit.ccSP <- cch(Surv(edrel, rel) ~ stage + histol + age, data =ccoh.data,
    subcoh = ~subcohort, id=~seqno, cohort.size=4028, method="SelfPren")

summary(fit.ccSP)

## (post-)stratified on instit
##
## stratsizes<-table(nwtco$instit)

fit.BI<- cch(Surv(edrel, rel) ~ stage + histol + age, data =ccoh.data,
    subcoh = ~subcohort, id=~seqno, stratum=~instit, cohort.size=stratsizes,
    method="I.Borgan")

summary(fit.BI)

clogit

Conditional logistic regression

Description

Estimates a logistic regression model by maximising the conditional likelihood. Uses a model formula of the form case.status~exposure+strata(matched.set). The default is to use the exact conditional likelihood, a commonly used approximate conditional likelihood is provided for compatibility with older software.

Usage

clogit(formula, data, method=c("exact", "approximate"), na.action=getOption("na.action"), subset=NULL,control=coxph.control())

Arguments

  formula  Model formula
  data     data frame
  method   correct or approximate conditional likelihood?
  na.action missing value handling
  subset   subset of records to use
  control  control values

Value

An object of class "clogit", which is a wrapper for a "coxph" object.

Author(s)

Thomas Lumley
See Also

strata, coxph, glm

Examples

clogit(case~spontaneous+induced+strata(stratum),data=infert)

Description

This is a special function used in the context of survival models. It identifies correlated groups of observations, and is used on the right hand side of a formula. Using `cluster()` in a formula implies that robust sandwich variance estimators are desired.

Usage

`cluster(x)`

Arguments

x
  A character, factor, or numeric variable.

Value

x

See Also

coxph, Surv

Examples

frailty.model <- coxph(Surv(time, status)~ age + sex + disease + frailty(id), kidney)
marginal.model <- coxph(Surv(time, status)~ age + sex + disease+cluster(id), kidney)
summary(frailty.model)
summary(marginal.model)

frailty.model <- survreg(Surv(time, status) ~ rx + frailty(litter), rats )
marginal.model <- survreg(Surv(time, status) ~ rx + cluster(litter), rats )
summary(frailty.model)
summary(marginal.model)
Description

These are data from one of the first successful trials of adjuvant chemotherapy for colon cancer. Levamisole is a low-toxicity compound previously used to treat worm infestations in animals; 5-FU is a moderately toxic (as these things go) chemotherapy agent. There are two records per person, one for recurrence and one for death.

Usage

colon

Format

- id: id
- study: 1 for all patients
- rx: Treatment - Obs(ervation), Lev(amisole), Lev(amisole)+5-FU
- sex: 1=male
- age: in years
- obstruct: obstruction of colon by tumour
- perfor: perforation of colon
- adhere: adherence to nearby organs
- nodes: number of lymph nodes with detectable cancer
- status: censoring status
- differ: differentiation of tumour (1=well, 2=moderate, 3=poor)
- extent:Extent of local spread (1=submucosa, 2=muscle, 3=serosa, 4=contiguous structures)
- surg: time from surgery to registration (0=short, 1=long)
- node4: more than 4 positive lymph nodes
- time: days until death
- etype: 1=recurrence, 2=death

Source

Danyu Lin

Description

Test the proportional hazards assumption for a Cox regression model fit (coxph).
Usage

```r
cox.zph(fit, transform="km", global=TRUE)
```

Arguments

- `fit` the result of fitting a Cox regression model, using the `coxph` function.
- `transform` a character string specifying how the survival times should be transformed before the test is performed. Possible values are "km", "rank", "identity" or a function of one argument. The default is "km" for right-censored data and "identity" for counting-processing data.
- `global` should a global chi-square test be done, in addition to the per-variable tests.

Value

- an object of class "cox.zph", with components:
  - `table` a matrix with one row for each variable, and optionally a last row for the global test. Columns of the matrix contain the correlation coefficient between transformed survival time and the scaled Schoenfeld residuals, a chi-square, and the two-sided p-value. For the global test there is no appropriate correlation, so an NA is entered into the matrix as a placeholder.
  - `x` the transformed time axis.
  - `y` the matrix of scaled Schoenfeld residuals. There will be one column per variable and one row per event. The row labels contain the original event times (for the identity transform, these will be the same as `x`).
  - `call` the calling sequence for the routine.

The computations require the original `x` matrix of the Cox model fit. Thus it saves time if the `x=TRUE` option is used in `coxph`. This function would usually be followed by both a plot and a print of the result. The plot gives an estimate of the time-dependent coefficient $\beta(t)$. If the proportional hazards assumption is true, $\beta(t)$ will be a horizontal line. The printout gives a test for $\text{slope}=0$.

References


See Also

`coxph`, `Surv`.

Examples

```r
fit <- coxph( Surv(futime, fustat) ~ age + rx, ovarian)
temp <- cox.zph(fit)
print(temp) #display the results
plot(temp) #plot curves
```
**coxph**  
*Fit Proportional Hazards Regression Model*

**Description**

Fits a Cox proportional hazards regression model. Time dependent variables, time dependent strata, multiple events per subject, and other extensions are incorporated using the counting process formulation of Andersen and Gill.

**Usage**

```r
coxph(formula, data=parent.frame(), weights, subset,  
na.action, init, control, method=c("efron","breslow","exact"),  
singular.ok=TRUE, robust=FALSE,  
model=FALSE, x=FALSE, y=TRUE, ...
```

**Arguments**

- **formula**: a formula object, with the response on the left of a ~ operator, and the terms on the right. The response must be a survival object as returned by the `Surv` function.
- **data**: a data.frame in which to interpret the variables named in the formula, or in the subset and the weights argument.
- **subset**: expression saying that only a subset of the rows of the data should be used in the fit.
- **na.action**: a missing-data filter function, applied to the model.frame, after any subset argument has been used. Default is `options()$na.action`.
- **weights**: case weights.
- **init**: vector of initial values of the iteration. Default initial value is zero for all variables.
- **control**: Object of class `coxph.control` specifying iteration limit and other control options. Default is `coxph.control(...)`.
- **method**: a character string specifying the method for tie handling. If there are no tied death times all the methods are equivalent. Nearly all Cox regression programs use the Breslow method by default, but not this one. The Efron approximation is used as the default here, as it is much more accurate when dealing with tied death times, and is as efficient computationally. The exact method computes the exact partial likelihood, which is equivalent to a conditional logistic model. If there are a large number of ties the computational time will be excessive.
- **singular.ok**: logical value indicating how to handle collinearity in the model matrix. If `TRUE`, the program will automatically skip over columns of the X matrix that are linear combinations of earlier columns. In this case the coefficients for such columns will be NA, and the variance matrix will contain zeros. For ancillary calculations, such as the linear predictor, the missing coefficients are treated as zeros.
robust if TRUE a robust variance estimate is returned. Default is TRUE if the model includes a \texttt{cluster()} operative, FALSE otherwise.

model flags to control what is returned. If these are true, then the model frame, the model matrix, and/or the response is returned as components of the fitted model, with the same names as the flag arguments.

x Return the design matrix in the model object?

y return the response in the model object?

... Other arguments will be passed to \texttt{coxph.control}

Details

The proportional hazards model is usually expressed in terms of a single survival time value for each person, with possible censoring. Andersen and Gill reformulated the same problem as a counting process; as time marches onward we observe the events for a subject, rather like watching a Geiger counter. The data for a subject is presented as multiple rows or "observations", each of which applies to an interval of observation (start, stop).

Value

an object of class "\texttt{coxph}". See \texttt{coxph.object} for details.

Side Effects

Depending on the call, the \texttt{predict}, \texttt{residuals}, and \texttt{survfit} routines may need to reconstruct the x matrix created by \texttt{coxph}. Differences in the environment, such as which data frames are attached or the value of \texttt{options()}$\texttt{contrasts}$, may cause this computation to fail or worse, to be incorrect. See the survival overview document for details.

SPECIAL TERMS

There are two special terms that may be used in the model equation. A 'strata' term identifies a stratified Cox model; separate baseline hazard functions are fit for each strata. The \texttt{cluster} term is used to compute a robust variance for the model. The term + \texttt{cluster(id)}, where \texttt{id == unique(id)}, is equivalent to specifying the \texttt{robust=T} argument, and produces an approximate jackknife estimate of the variance. If the \texttt{id} variable were not unique, but instead identifies clusters of correlated observations, then the variance estimate is based on a grouped jackknife.

CONVERGENCE

In certain data cases the actual MLE estimate of a coefficient is infinity, e.g., a dichotomous variable where one of the groups has no events. When this happens the associated coefficient grows at a steady pace and a race condition will exist in the fitting routine: either the log likelihood converges, the information matrix becomes effectively singular, an argument to \texttt{exp} becomes too large for the computer hardware, or the maximum number of interactions is exceeded. The routine attempts to detect when this has happened, not always successfully.
PENALISED REGRESSION

`coxph` can now maximise a penalised partial likelihood with arbitrary user-defined penalty. Supplied penalty functions include ridge regression (`ridge`), smoothing splines (`pspline`), and frailty models (`frailty`).

References


See Also

`cluster`, `survfit`, `Surv`, `strata`, `ridge`, `pspline`, `frailty`.

Examples

```r
# Create the simplest test data set
#
test1 <- list(time= c(4, 3,1,1,2,2,3),
              status=c(1,NA,1,0,1,1,0),
              x= c(0, 2,1,1,1,0,0),
              sex= c(0, 0,0,0,1,1,1))
coxph( Surv(time, status) ~ x + strata(sex), test1)  # stratified model

# # Create a simple data set for a time-dependent model
#
test2 <- list(start=c(1, 2, 5, 2, 1, 7, 3, 4, 8, 8),
              stop =c(2, 3, 6, 7, 8, 9, 9, 9,14,17),
              event=c(1, 1, 1, 1, 1, 1, 0, 0, 0),
              x =c(1, 0, 0, 1, 1, 1, 0, 0, 0) )
summary( coxph( Surv(start, stop, event) ~ x, test2))
```

Details of a Cox Model Fit

Description

Details of a Cox model fit. Returns the individual contributions to the first and second derivative matrix, at each unique event time.

Usage

`coxph.detail(object)`
Arguments

object a Cox model object, i.e., the result of \texttt{coxph}.

Details

This function may be useful for those who wish to investigate new methods or extensions to the Cox model. The example below shows one way to calculate the Schoenfeld residuals.

Value

a list with components

- **time**: the vector of unique event times
- **nevent**: the number of events at each of these time points.
- **means**: a matrix with one row for each event time and one column for each variable in the Cox model, containing the weighted mean of the variable at that time, over all subjects still at risk at that time. The weights are the risk weights $\exp(x \%\% fit\$coef)$.
- **nrisk**: number of subjects at risk.
- **hazard**: the hazard increment.
- **score**: the contribution to the score vector (first derivative of the log partial likelihood) at each time point.
- **imat**: the contribution to the information matrix (second derivative of the log partial likelihood) at each time point.
- **varhaz**: the variance of the hazard increment.
- **x, y**: copies of the input data.
- **strata**: only present for a stratified Cox model, this is a table giving the number of time points of component \texttt{time} that were contributed by each of the strata.

See Also

\texttt{coxph, residuals.coxph}

Examples

```r
fit <- coxph(Surv(futime,fustat) ~ age + rx + ecog.ps, ovarian, x=TRUE)
fitd <- coxph.detail(fit)
events <- fit$y[,2]==1
etime <- fit$y[events,1]  # the event times --- may have duplicates
indx <- match(etime, fitd$time)
sresid <- fit$x[events,] - fitd$means[indx,]
```
Description

This class of objects is returned by the `coxph` class of functions to represent a fitted proportional hazards model.

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

COMPONENTS

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

- **coefficients** the coefficients of the linear predictor, which multiply the columns of the model matrix. If the model is over-determined there will be missing values in the vector corresponding to the redundant columns in the model matrix.
- **var** the variance matrix of the coefficients. Rows and columns corresponding to any missing coefficients are set to zero.
- **naive.var** this component will be present only if the `robust` option was true. If so, the `var` component will contain the robust estimate of variance, and this component will contain the ordinary estimate.
- **loglik** a vector of length 2 containing the log-likelihood with the initial values and with the final values of the coefficients.
- **score** value of the efficient score test, at the initial value of the coefficients.
- **rscore** the robust log-rank statistic, if a robust variance was requested.
- **wald.test** the Wald test of whether the final coefficients differ from the initial values.
- **iter** number of iterations used.
- **linear.predictors** the vector of linear predictors, one per subject.
- **residuals** the martingale residuals.
- **means** vector of column means of the X matrix. Subsequent survival curves are adjusted to this value.
- **n** the number of observations used in the fit.
- **weights** the vector of case weights, if one was used.
- **method** the computation method used.
- **na.action** the `na.action` attribute, if any, that was returned by the `na.action` routine.

The object will also contain the following, for documentation see the `lm` object: `terms`, `assign`, `formula`, `call`, and, optionally, `x`, `y`, and/or `frame`.

See Also

`coxph`, `coxph.detail`, `cox.zph`, `survfit`, `residuals.coxph`, `survreg`
date.ddmmmyy  
Format a Julian date

Description
Given a vector of Julian dates, this returns them in the form “10Nov89”, “28Jul54”, etc.

Usage
date.ddmmmyy(sdate)

Arguments
sdate  
A vector of Julian dates, e.g., as returned by mdy.date().

Value
A vector of character strings containing the formatted dates.

See Also
  
mdy.date, date.mdy

Examples
date.ddmmmyy(1:10)

date.mdy  
Convert from Julian Dates to Month, Day, and Year

Description
Convert a vector of Julian dates to a list of vectors with the corresponding values of month, day and year, and optionally weekday.

Usage
date.mdy(sdate, weekday = FALSE)

Arguments
sdate  
a Julian date value, as returned by mdy.date(), number of days since 1/1/1960.

weekday  
if TRUE, then the returned list also will contain the day of the week (Sunday=1, Saturday=7).
Value

A list with components `month`, `day`, and `year`.

References


Examples

day <- 7
temp <- date.mdy(mdy.date(month = 7, day = day, year = 1960))
## Check for illegal dates, such as 29 Feb in a non leap year
if (temp$day != day) {
  cat("Some illegal dates\n")
} else {
  cat("All days are legal\n")
}

date.mmddyy

Format a Julian date

Description

Given a vector of Julian dates, this returns them in the form “10/11/89”, “28/7/54”, etc.

Usage

date.mmddyy(sdate, sep = "/")

Arguments

- `sdate`: A vector of Julian dates, e.g., as returned by `mdy.date()`.
- `sep`: Character string used to separate the month, day, and year portions of the returned string.

Value

A vector of character strings containing the formatted dates.

See Also

date.mdy, mdy.date, date.ddmmmyy

Examples

date.mmddyy(as.date(10))
date.mmdyyyyyy       Format a Julian date

Description

Given a vector of Julian dates, this returns them in the form “10/11/1989”, “28/7/1854”, etc.

Usage

date.mmdyyyyyy(sdate, sep = "/")

Arguments

sdate       A vector of Julian dates, e.g., as returned by mdy.date().
sep       Character string used to separate the month, day, and year portions of the returned string.

Value

A vector of character strings containing the formatted dates.

See Also

date.mdy, mdy.date, date.ddmmmyy

Examples

date.mmdyyyyyy(as.date(1:10))

---

date.object       Date Objects

Description

Objects of class "date".

Usage

is.date(x)

Arguments

x       any R object.
frailty

Details

Details are stored as the number of days since 1/1/1960, and are kept in integer format. (This is the same baseline value as is used by SAS). The numerical methods for dates treat \( \text{date} - \text{date} \) as a numeric, and \( \text{date} \pm \text{numeric} \) as a date.

\textit{is.date} returns \( \text{TRUE} \) if \( x \) has class "date", and \( \text{FALSE} \) otherwise. Its behavior is unaffected by any attributes of \( x \); for example, \( x \) could be a date array (in contrast to the behavior of \textit{is.vector}).

\textit{as.date} returns \( x \) if \( x \) is a simple object of class "date", and otherwise a date vector of the same length as \( x \) and with data resulting from coercing the elements of \( x \) to class "date". See the manual page for \textit{as.date()} for details.

Logical operations as well as the numeric functions \textit{exp()}, \textit{log()}, and so on are invalid.

Other methods exist for missing value, \textit{as.character()}, printing, and summarizing.

See Also

date.mdy, mdy.date, date.ddmmmyy, as.date.

frailty

(Approximate) Frailty models

Description

When included in a \texttt{coxph} or \texttt{survreg}, fits by penalised likelihood a random effects (frailty) model.

\texttt{frailty} is generic, with methods for t, Gaussian and Gamma distributions.

Usage

\texttt{frailty(x, distribution="gamma", ...)}

\texttt{frailty.gamma(x, sparse = (nclass > 5), theta, df, eps = 1e-05, method = c("em","aic","df"))}

\texttt{frailty.gaussian(x, sparse = (nclass > 5), theta, df, method = c("reml","aic","df"))}

\texttt{frailty.t(x, sparse = (nclass > 5), theta, df, eps = 1e-05, tdf = 5, method = c("aic","df"))}

Arguments

\texttt{x} group indicator

\texttt{distribution} frailty distribution

\texttt{...} Arguments for specific distribution, including (but not limited to)

\texttt{sparse} Use sparse Newton-Raphson algorithm

\texttt{df} Approximate degrees of freedom

\texttt{theta} Penalty

\texttt{eps} Accuracy of \( df \)

\texttt{method} maximisation algorithm

\texttt{tdf} \( df \) of t-distribution
Details

The penalised likelihood method is equivalent to maximum (partial) likelihood for the gamma frailty but not for the others.

The sparse algorithm uses the diagonal of the information matrix for the random effects, which saves a lot of space.

The frailty distributions are really the log-t and lognormal: t and Gaussian are random effects on the scale of the linear predictor.

Value

An object of class `coxph.penalty` containing a factor with attributes specifying the control functions.

References


See Also

`coxph`, `survreg`, `ridge`, `pspline`
Usage

heart
jsa
jsa1

Format

jsa: original data

birth.dt: birth date
accept.dt: acceptance into program
tax.date: transplant date
fu.date: end of followup
fustat: dead or alive
surgery: prior bypass surgery
age: age (in days)
futime: followup time
wait.time: time before transplant
transplant: transplant indicator
mismatch: mismatch score
hla.a2: particular type of mismatch
mscore: another mismatch score
reject: rejection occurred

jsa1, heart: processed data

start, stop, event: Entry and exit time and status for this interval of time
age: age-48 years
year: year of acceptance (in years after 1 Nov 1967)
surgery: prior bypass surgery 1=yes
transplant: received transplant 1=yes
id: patient id

Source


See Also

stanford2

is.ratetable Verify that an Object is of Class ‘ratetable’
Description

The function verifies not only the class attribute, but the structure of the object.

Usage

\texttt{is.ratetable(x, verbose=FALSE)}

Arguments

- \texttt{x} the object to be verified.
- \texttt{verbose} if \texttt{TRUE} and the object is not a ratetable, then return a character string describing the way(s) in which \texttt{x} fails to be a proper ratetable object.

Details

Rate tables are used by the \texttt{pyears} and \texttt{survexp} functions, and normally contain death rates for some population, categorized by age, sex, or other variables. They have a fairly rigid structure, and the \texttt{verbose} option can help in creating a new rate table.

Value

returns \texttt{TRUE} if \texttt{x} is a ratetable, and \texttt{FALSE} or a description if it is not.

See Also

\texttt{pyears, survexp}

Examples

\begin{verbatim}
  is.ratetable(survexp.us)  ##Yes
  is.ratetable(cancer)  ##No
\end{verbatim}

\begin{table}
\begin{tabular}{ll}
kidney & \textit{Kidney data from survival5} \\
\end{tabular}
\end{table}

Description

Data on the recurrence times to infection, at the point of insertion of the catheter, for kidney patients using portable dialysis equipment. Catheters may be removed for reasons other than infection, in which case the observation is censored. Each patient has exactly 2 observations.
**Format**

- **patient**: id
- **time**: time
- **status**: event status
- **age**: in years
- **sex**: 1=male, 2=female
- **disease**: disease type (0=GN, 1=AN, 2=PKD, 3=Other)
- **frail**: frailty estimate from original paper

**Note**

The original analysis had incorrect handling of ties and so is not exactly reproduced by survival.

**Source**


**Examples**

```r
kfit <- coxph(Surv(time, status)~ age + sex + disease + frailty(id), kidney)
kfit0 <- coxph(Surv(time, status)~ age + sex + disease, kidney)
kfitm1 <- coxph(Surv(time,status) ~ age + sex + disease +
                   frailty(id, dist='gauss'), kidney)
```

---

**Description**

Often used to add the expected survival curve(s) to a Kaplan-Meier plot generated with `plot.survfit`.

**Usage**

```r
## S3 method for class 'survfit':
lines(x, type="s", mark=3, col=1, lty=1,
      lwd=1, mark.time=TRUE , xscale=1, firstx=0, firsty=1, xmax, fun, conf.int=FALSE,
## S3 method for class 'survfit':
points(x, ...)
```
Arguments

- **x**: a survival object, generated from the `survfit` or `survexp` functions.
- **type**: the line type, as described in `lines`. The default is a step function for `survfit` objects, and a connected line for `survexp` objects.
- **mark**: vectors giving the mark symbol, color, line type and line width for the added curves.
- **mark.time**: controls the labeling of the curves. If `FALSE`, no labeling is done. If `TRUE`, then curves are marked at each censoring time. If `mark.time` is a numeric vector, then curves are marked at the specified time points.
- **xscale**: a number used to divide the x values. If time was originally in days, a value of 365.24 would give a plotted scale in years.
- **firstx,firsty**: the starting point for the survival curves. If either of these is set to `NA` or `<blank>`, the plot will start at the first time point of the curve.
- **col,lty,lwd,...**: passed to `lines`
- **xmax**: the maximum horizontal plot coordinate. This shortens the curve before plotting it, so unlike using the `xlim` graphical parameter, warning messages about out of bounds points are not generated.
- **fun**: an arbitrary function defining a transformation of the survival curve. For example, `fun=log` is an alternative way to draw a log-survival curve (but with the axis labeled with log(S) values). Four often used transformations can be specified with a character argument instead: "log" is the same as using the `log=T` option, "event" plots cumulative events (f(y) =1-y), "cumhaz" plots the cumulative hazard function (f(y) = -log(y)) and "cloglog" creates a complementary log-log survival plot (f(y) = log(-log(y)) along with log scale for the x-axis).
- **conf.int**: if `TRUE`, confidence bands for the curves are also plotted. If set to "only", then only the CI bands are plotted, and the curve itself is left off. This can be useful for fine control over the colors or line types of a plot.

Value

A list with components `x` and `y`, containing the coordinates of the last point on each of the curves (but not of the confidence limits). This may be useful for labeling.

Side Effects

One or more curves are added to the current plot.

See Also

`lines`, `par`, `plot.survfit`, `survfit`, `survexp`. 
Examples

```r
fit <- survfit(Surv(time, status) ~ sex, pbc, subset=1:312)
plot(fit, mark.time=FALSE, xscale=365.24,
     xlab='Years', ylab='Survival')
lines(fit[1], lwd=2, xscale=365.24) #darken the first curve and add marks

# add expected survival curves for the two groups,
# based on the US census data

tdata <- data.frame(age=pbc$age*365.24, sex=pbc$sex +1,
                      year= rep(mdy.date(1,1,1976), nrow(pbc)))
tdata<-tdata[1:312,] ## only the randomised people, with no missing data

efit <- survexp(~ sex+ratetable(sex=sex,age=age,year=year), data=tdata, ratetable=survexp.us,
                times=(0:24) *182)
temp <- lines(efit, lty=2, xscale=365.24, lwd=2:1)
text(temp, c("Male", "Female"), adj= -.1) #labels just past the ends

title(main="Primary Biliary Cirrhosis, Observed and Expected")
```

---

**Mayo Clinic Lung Cancer Data**

**Description**

Survival in patients with lung cancer at Mayo Clinic. Performance scores rate how well the patient can perform usual daily activities.

**Usage**

```r
lung
cancer
```

**Format**

- **inst**: Institution code
- **time**: Survival time in days
- **status**: censoring status 1=censored, 2=dead
- **age**: Age in years
- **sex**: Male=1, Female=2
- **ph.ecog**: ECOG performance score (0=good 5=dead)
- **ph.karno**: Karnofsky performance score (bad=0-good=100) rated by physician
- **pat.karno**: Karnofsky performance score rated by patient
- **meal.cal**: Calories consumed at meals
- **wt.loss**: Weight loss in last six months
Description

Given a month, day, and year, returns the number of days since January 1, 1960.

Usage

```r
mdy.date(month, day, year, nineteen = TRUE, fillday = FALSE,
          fillmonth = FALSE)
```

Arguments

- `month`: vector of months.
- `day`: vector of days.
- `year`: vector of years.
- `nineteen`: if `TRUE`, year values between 0 and 99 are assumed to be in the 20th century A.D.; otherwise, if `FALSE`, they are assumed to be in the 1st century A.D.
- `fillday`: if `TRUE`, then missing days are replaced with 15.
- `fillmonth`: if `TRUE`, then a missing month causes the month and day to be set to 7/1.

Details

The date functions are particularly useful in computing time spans, such as number of days on test, and similar functions can be found in other statistical packages. The baseline date of Jan 1, 1960 is, of course, completely arbitrary (it is the same one used by SAS).

The `fillday` and `fillmonth` options are perhaps useful only to the author and a very few others: we sometimes deal with patients whose birth date was in the 1800’s, and only the month or even only the year is known. When the interval is greater than 80 years, a filler seems defensible.

Value

A vector of Julian dates.

References


See Also

date.mmddyy, date.ddmmmyy, date.mmdyyyy
Examples

```r
mdy.date(3, 10, 53)
xzt <- 1:10
xzy <- as.date(xzt)
test <- data.frame(x = xzt, date = xzy)
summary(test)
```

Data from the National Wilm’s Tumor Study

Description

Missing data/measurement error example. Tumor histology predicts survival, but prediction is stronger with central lab histology than with the local institution determination.

Usage

```r
nwtco
```

Format

A data frame with 4028 observations on the following 9 variables.

- `seqno`: id number
- `instit`: Histology from local institution
- `histol`: Histology from central lab
- `stage`: Disease stage
- `study`: study
- `rel`: indicator for relapse
- `edrel`: time to relapse
- `age`: age in months
- `in.subcohort`: Included in the subcohort for the example in the paper

Source

http://faculty.washington.edu/norm/software.html

References


Examples

```r
with(nwtco, table(instit, histol))
anova(coxph(Surv(edrel, rel) ~ histol + instit, data = nwtco))
anova(coxph(Surv(edrel, rel) ~ instit + histol, data = nwtco))
```
**ovarian**  
*Ovarian Cancer Survival Data*

**Description**
Survival in a randomised trial comparing two treatments for ovarian cancer (? from an Eastern Cooperative Oncology Group study)

**Usage**
```
ovarian
```

**Format**
- futime: survival or censoring time
- fustat: censoring status
- age: in years
- resid.ds: residual disease present (1=no, 2=yes)
- rx: treatment group
- ecog.ps: ECOG performance status (1 is better, see reference)

**Source**
Terry Therneau

**References**
for ECOG performance status: [http://ecog.dfci.harvard.edu/~ecogdba/general/perf_stat.html](http://ecog.dfci.harvard.edu/~ecogdba/general/perf_stat.html)

---

**pbc**  
*Mayo Clinic Primary Biliary Cirrhosis Data*

**Description**
Followup of 312 randomised and 108 unrandomised patients with primary biliary cirrhosis, a rare autoimmune liver disease, at Mayo Clinic. Most variables have some missing data, coded as -9.

**Usage**
```
pbc
```

**Format**
age: in years
alb: serum albumin
alkphos: alkaline phosphotase
ascites: presence of ascites
bili: serum bilirunbin
chol: serum cholesterol
edema: presence of edema
edtrt: 0 no edema, 0.5 untreated or successfully treated
       1 unsuccessfully treated edema
hepmeg: enlarged liver
time: survival time
platelet: platelet count
protime: standardised blood clotting time
sex: 1=male
sgot: liver enzyme (now called AST)
spiders: blood vessel malformations in the skin
stage: histologic stage of disease (needs biopsy)
status: censoring status
trt: 1/2/-9 for control, treatment, not randomised
trig: triglycerides
copper: urine copper

Source

---

plot.cox.zph  Graphical Test of Proportional Hazards

Description
Displays a graph of the scaled Schoenfeld residuals, along with a smooth curve.

Usage
```r
## S3 method for class 'cox.zph':
plot(x, resid=TRUE, se=TRUE, df=4, nsmo=40, var, ...)
```

Arguments
- `x`  
  result of the `cox.zph` function.
- `resid`  
a logical value, if TRUE the residuals are included on the plot, as well as the smooth fit.
se
a logical value, if TRUE, confidence bands at two standard errors will be added.
df
the degrees of freedom for the fitted natural spline, df=2 leads to a linear fit.
nsmo
number of points used to plot the fitted spline.
var
the set of variables for which plots are desired. By default, plots are produced
in turn for each variable of a model. Selection of a single variable allows other
features to be added to the plot, e.g., a horizontal line at zero or a main title.
This has been superseded by a subscripting method; see the example below.
additional arguments passed to the plot function.

**Side Effects**
a plot is produced on the current graphics device.

**See Also**
cox.zph, coxph

**Examples**

```r
vfit <- coxph(Surv(time,status) ~ trt + factor(celltype) +
karno + age, data=veteran, x=TRUE)
temp <- cox.zph(vfit)
plot(temp, var=5)       # Look at Karnofsy score, old way of doing plot
plot(temp[5])          # New way with subscripting
abline(0, 0, lty=3)    # Add the linear fit as well
abline(lm(temp$y[,5] ~ temp$x)$coefficients, lty=4, col=3)
title(main="VA Lung Study")
```

---

**plot.survfit**

*Plot Method for 'survfit'*

**Description**
A plot of survival curves is produced, one curve for each strata. The log=T option does extra work
to avoid log(0), and to try to create a pleasing result. If there are zeros, they are plotted by default
at 0.8 times the smallest non-zero value on the curve(s).

If legend.text is supplied a legend is created.

**Usage**
```r
## S3 method for class 'survfit':
plot(x, conf.int=, mark.time=TRUE,
    mark=3, col=1, lty=1, lwd=1, cex=1, log=FALSE, xscale=1, yscale=1,
    firstx=0, firsty=1, xmax, ymin=0, fun,
    xlab="", ylab="", xaxs="S", bty=NULL,
    legend.text=NULL,legend.pos=0,legend.bty="n",main=NULL,...)
```
Arguments

- **x**: an object of class `survfit`, usually returned by the `survfit` function.
- **conf.int**: determines whether confidence intervals will be plotted. The default is to do so if there is only 1 curve, i.e., no strata.
- **mark.time**: controls the labeling of the curves. If set to `FALSE`, no labeling is done. If `TRUE`, then curves are marked at each censoring time which is not also a death time. If `mark.time` is a numeric vector, then curves are marked at the specified time points.
- **mark**: vector of mark parameters, which will be used to label the curves. The `lines` help file contains examples of the possible marks. The vector is reused cyclically if it is shorter than the number of curves.
- **col**: a vector of integers specifying colors for each curve. The default value is 1.
- **lty**: a vector of integers specifying line types for each curve. The default value is 1.
- **lwd**: a vector of numeric values for line widths. The default value is 1.
- **cex**: a numeric value specifying the size of the marks. Not a vector; all marks have the same size.
- **log**: a logical value, if `TRUE` the y axis will be on a log scale. Alternately, one of the standard character strings "x", "y", or "xy" can be given to specific logarithmic horizontal and/or vertical axes.
- **xscale**: scale the x-axis values before plotting. If time were in days, then a value of 365.25 will give labels in years instead of the original days.
- **yscale**: will be used to multiply the labels on the y axis. A value of 100, for instance, would be used to give a percent scale. Only the labels are changed, not the actual plot coordinates, so that adding a curve with `lines(surv.exp(...{}))`, say, will perform as it did without the `yscale` argument.
- **firstx,firsty**: the starting point for the survival curves. If either of these is set to `NA` the plot will start at the first time point of the curve.
- **xmax**: the maximum horizontal plot coordinate. This can be used to shrink the range of a plot. It shortens the curve before plotting it, so that unlike using the `xlim` graphical parameter, warning messages about out of bounds points are not generated.
- **ymin**: lower boundary for y values. Survival curves are most often drawn in the range of 0-1, even if none of the curves approach zero. The parameter is ignored if the `fun` argument is present, or if it has been set to `NA`.
- **fun**: an arbitrary function defining a transformation of the survival curve. For example `fun=log` is an alternative way to draw a log-survival curve (but with the axis labeled with log(S) values), and `fun=sqrt` would generate a curve on square root scale. Four often used transformations can be specified with a character argument instead: "log" is the same as using the `log=T` option, "event" plots cumulative events (f(y) = 1-y), "cumhaz" plots the cumulative hazard function (f(y) = -log(y)), and "cloglog" creates a complimentary log-log survival plot (f(y) = log(-log(y)) along with log scale for the x-axis).
- **xlab**: label given to the x-axis.
ylab

label given to the y-axis.

xaxs
either "S" for a survival curve or a standard x axis style as listed in par. Survival curves are usually displayed with the curve touching the y-axis, but not touching the bounding box of the plot on the other 3 sides. Type "S" accomplishes this by manipulating the plot range and then using the "i" style internally.

bty

see par

legend.pos

position for the legend: a vector of length 2, or 0 to put the legend in the lower left, 1 to put it in the upper right. These may not work well with transformed curves.

legend.bty

Box type, see legend

legend.text

Text for legend, see legend

main

Plot title

... other graphical parameters

Value

a list with components x and y, containing the coordinates of the last point on each of the curves (but not the confidence limits). This may be useful for labeling.

BUGS

Survival curve objects created from a coxph model does not include the censoring times. Therefore, specifying mark.time=T does not work. If you want to mark censoring times on the curve(s) resulting from a coxph fit, provide a vector of times as the mark.time argument in the call to plot.survfit or lines.survfit.

See Also

par, survfit, lines.survfit, legend.

Examples

gleukemia.surv <- survfit(Surv(time, status) ~ x, data = aml)
plot(leukemia.surv, lty = 2:3)
legend(100, .9, c("Maintenance", "No Maintenance"), lty = 2:3)
title("Kaplan-Meier Curves for AML Maintenance Study")

lsurv2 <- survfit(Surv(time, status) ~ x, aml, type='fleming')
plot(lsurv2, lty=2:3, fun="cumhaz",
 xlab="Months", ylab="Cumulative Hazard")

plot(leukemia.surv,lty=1:2,legend.pos=0,col=c("Red","Blue"),legend.text=c("Maintenance", "No Maintenance"))
if (interactive()){
cat("Click on the graphics device to place a legend\n")
plot(leukemia.surv,lty=2:3,legend.pos=locator(1),legend.text=c("Maintenance", "No Maintenance"))
}
predict.coxph

Predictions for Cox model

Description

Compute fitted values and regression terms for a model fitted by coxph

Usage

```r
## S3 method for class 'coxph':
predict(object, newdata, type=c("lp", "risk", "expected", "terms"), se.fit=FALSE, terms=names(object$assign), collapse, safe=FALSE, ...)
```

Arguments

- `object`: A coxph object
- `newdata`: Optional new data to predict at
- `type`: Type of prediction wanted
- `se.fit`: Return standard errors as well?
- `terms`: If type="terms", which terms to return.
- `collapse`: identifier for which rows correspond to different individuals
- `safe`: Use a more intensive ‘safe’ prediction method
- `...`: other unused arguments

Value

A vector or matrix of fitted values. If se.fit=TRUE a list whose first component is the fitted values and second is the standard errors.

See Also

predict.coxph, termplot

Examples

```r
fit<-coxph(Surv(time,status)~x, data=aml)
predict(fit,type="lp")
predict(fit,type="risk")
predict(fit,type="expected")
predict(fit,type="terms")
predict(fit,type="lp", se.fit=TRUE)
predict(fit,type="risk", se.fit=TRUE)
predict(fit,type="expected", se.fit=TRUE)
predict(fit,type="terms", se.fit=TRUE)
```
predict.survreg  Predicted Values for a `survreg` Object

Description

Predicted values for a `survreg` object

Usage

```r
## S3 method for class 'survreg':
predict(object, newdata,
type=c("response", "link", "lp", "linear", "terms", "quantile", "uquantile"),
se.fit=FALSE, terms=NULL, p=c(0.1, 0.9),...)
```

Arguments

- `object`: result of a model fit using the `survreg` function.
- `newdata`: data for prediction. If absent, predictions are for the subjects used in the original fit.
- `type`: the type of predicted value. This can be on the original scale of the data (response), the linear predictor ("linear", with "lp" as an allowed abbreviation), a predicted quantile on the original scale of the data ("quantile"), a quantile on the linear predictor scale ("uquantile"), or the matrix of terms for the linear predictor ("terms"). At this time "link" and linear predictor ("lp") are identical.
- `se.fit`: if TRUE, include the standard errors of the prediction in the result.
- `terms`: subset of terms. The default for residual type "terms" is a matrix with one column for every term (excluding the intercept) in the model.
- `p`: vector of percentiles. This is used only for quantile predictions.
- `...`: other arguments

Value

a vector or matrix of predicted values.

References


See Also

`survreg`, `residuals.survreg`
Examples

# Draw figure 1 from Escobar and Meeker
fit <- survreg(Surv(time, status) ~ age + age^2, data=stanford2,
               dist='lognormal')
plot(stanford2$age, stanford2$time, xlab='Age', ylab='Days',
     xlim=c(0,65), ylim=c(.01, 10^6), log='y')
pred <- predict(fit, newdata=list(age=1:65), type='quantile',
                p=c(.1, .5, .9))
matlines(1:65, pred, lty=c(2,1,2), col=1)

print.survfit

Print a Short Summary of a Survival Curve

Description

Print number of observations, number of events, the restricted mean survival and its standard error, and the median survival with confidence limits for the median.

Usage

## S3 method for class 'survfit':
print(x, scale=1, digits = max(options()$digits - 4,
         3), print.n=getOption("survfit.print.n"), show.rmean=getOption("survfit.print.mean")

Arguments

x the result of a call to the survfit function.

print.n What to use for number of subjects (see below)

digits Number of digits to print

scale a numeric value to rescale the survival time, e.g., if the input data to survfit were
       in days, scale=365 would scale the printout to years.

show.rmean Show the restricted mean survival?

... other unused arguments

Details

The restricted mean (rmean) and its standard error se(rmean) are based on a truncated estimator. If the last observation(s) is not a death, then the survival curve estimate does not go to zero and the mean survival time cannot be estimated. Instead, the quantity reported is the mean of survival restricted to the time before the last censoring. When the last censoring time is not random this quantity is occasionally of interest.

Any randomness in the last censoring time is not taken into account in computing the standard error of the restricted mean. The restricted mean is available mainly for compatibility with S, and is not shown by default.

The median and its confidence interval are defined by drawing a horizontal line at 0.5 on the plot of the survival curve and its confidence bands. The intersection of the line with the lower CI band
defines the lower limit for the median’s interval, and similarly for the upper band. If any of the
intersections is not a point, then we use the smallest point of intersection, e.g., if the survival curve
were exactly equal to 0.5 over an interval.

The "number of observations" is not well-defined for counting process data. Previous versions of
this code used the number at risk at the first time point. This is misleading if many individuals enter
late or change strata. The original S code for the current version uses the number of records, which
is misleading when the counting process data actually represent a fixed cohort with time-dependent
covariates.

Four possibilities are provided, controlled by `print.n` or by `options(survfit.print.n)`:
"none" prints NA, "records" prints the number of records, "start" prints the number at the
first time point and "max" prints the maximum number at risk. The initial default is "start".

Value

`x`, with the invisible flag set.

Side Effects

The number of observations (see Details), the number of events, the median survival with its con-
fidence interval, and optionally the restricted mean survival (rmean) and its standard error, are
printed. If there are multiple curves, there is one line of output for each.

See Also

`summary.survfit`, `survfit.object`, `survfit`

Examples

```r
## effect of print.n and show.rmean
a <- coxph(Surv(start, stop, event) ~ age + strata(transplant), data=heart)
b <- survfit(a)
print(b, print.n="none")
print(b, print.n="records")
print(b, print.n="start")
print(b, print.n="max")
print(b, show.rmean=TRUE)
```

**pspline**

*Penalised smoothing splines*

Description

Specifies a penalised spline basis for the predictor. This is done by fitting a comparatively small set
of splines and penalising the integrated second derivative. Results are similar to smoothing splines
with a knot at each data point but computationally simpler.
Usage

pspline(x, df=4, theta, nterm=2.5 * df, degree=3, eps=0.1, method, ...)

Arguments

x       predictor
df      approximate degrees of freedom. df=0 means use AIC
theta   roughness penalty
nterm   number of splines in the basis
degree  degree of splines
eps     accuracy for df
method  Method for automatic choice of theta
...     I don’t know what this does

Value

Object of class coxph.penalty containing the spline basis with attributes specifying control functions.

See Also

coxph, survreg, ridge, frailty

Examples

lfit6 <- survreg(Surv(time, status) ~ pspline(age, df=2), cancer)
plot(cancer$age, predict(lfit6), xlab='Age', ylab="Spline prediction")
title("Cancer Data")
fit0 <- coxph(Surv(time, status) ~ ph.ecog + age, cancer)
fit1 <- coxph(Surv(time, status) ~ ph.ecog + pspline(age,3), cancer)
fit3 <- coxph(Surv(time, status) ~ ph.ecog + pspline(age,8), cancer)
fit0
fit1
fit3

Description

This function computes the person-years of follow-up time contributed by a cohort of subjects, stratified into subgroups. It also computes the number of subjects who contribute to each cell of the output table, and optionally the number of events and/or expected number of events in each cell.
Usage

```r
pyears(formula, data, weights, subset, na.action, ratetable=survexp.us,
scale=365.25, expect=c('event', 'pyears'), model=FALSE, x=FALSE, y=FALSE)
```

Arguments

- **formula**: a formula object. The response variable will be a vector of follow-up times for each subject, or a `Surv` object containing the follow-up time and an event indicator. The predictors consist of optional grouping variables separated by + operators (exactly as in `survfit`), time-dependent grouping variables such as age (specified with `tcut`), and optionally a `ratetable()` term. This latter matches each subject to his/her expected cohort.

- **data**: a data frame in which to interpret the variables named in the formula, or in the subset and the weights argument.

- **weights**: case weights.

- **subset**: expression saying that only a subset of the rows of the data should be used in the fit.

- **na.action**: a missing-data filter function, applied to the model.frame, after any subset argument has been used. Default is `options()`$na.action.

- **ratetable**: a table of event rates, such as `survexp.uswhite`.

- **scale**: a scaling for the results. As most rate tables are in units/day, the default value of 365.25 causes the output to be reported in years.

- **expect**: should the output table include the expected number of events, or the expected number of person-years of observation. This is only valid with a rate table.

- **model,x,y**: If any of these is true, then the model frame, the model matrix, and/or the vector of response times will be returned as components of the final result.

Details

Because `pyears` may have several time variables, it is necessary that all of them be in the same units. For instance, in the call `py <- pyears(futime + ratetable(age=age, sex=sex, year=entry.dt))` with a `ratetable` whose natural unit is days, it is important that `futime`, `age` and `entry.dt` all be in days. Given the wide range of possible inputs, it is difficult for the routine to do sanity checks of this aspect.

A special function `tcut` is needed to specify time-dependent cutpoints. For instance, assume that age is in years, and that the desired final arrays have as one of their margins the age groups 0-2, 2-10, 10-25, and 25+. A subject who enters the study at age 4 and remains under observation for 10 years will contribute follow-up time to both the 2-10 and 10-25 subsets. If `cut(age, c(0,2,10,25,100))` were used in the formula, the subject would be classified according to his starting age only. The `tcut` function has the same arguments as `cut`, but produces a different output object which allows the `pyyears` function to correctly track the subject.

The results of `pyyears` are normally used as input to further calculations. The example below is from a study of hip fracture rates from 1930 - 1990 in Rochester, Minnesota. Survival post hip fracture has increased over that time, but so has the survival of elderly subjects in the population at large. A model of relative survival helps to clarify what has happened: Poisson regression is used,
but replacing exposure time with expected exposure (for an age and sex matched control). Death rates change with age, of course, so the result is carved into 1 year increments of time. Males and females were done separately.

Value

a list with components

- **pyears**: an array containing the person-years of exposure. (Or other units, depending on the rate table and the scale).
- **n**: an array containing the number of subjects who contribute time to each cell of the **pyears** array.
- **event**: an array containing the observed number of events. This will be present only if the response variable is a **Surv** object.
- **expected**: an array containing the expected number of events (or person years). This will be present only if there was a ratetable term.
- **offtable**: the number of person-years of exposure in the cohort that was not part of any cell in the **pyears** array. This is often useful as an error check; if there is a mismatch of units between two variables, nearly all the person years may be off table.
- **summary**: a summary of the rate-table matching. This is also useful as an error check.
- **call**: an image of the call to the function.
- **na.action**: the **na.action** attribute contributed by an **na.action** routine, if any.

See Also

ratetable, survexp, Surv

Examples

```
# Simple case: a single male subject, born 6/6/36 and entered on study 6/6/55.
#
temp1 <- mdy.date(6,6,36)
temp2 <- mdy.date(6,6,55)
# Now compare the results from person-years
#
temp.age <- tcut(temp2-temp1, floor(c(-1, (18:31 * 365.24))),
  labels=c('0-18', paste(18:30, 19:31, sep='-')))
temp.yr <- tcut(temp2, mdy.date(1,1,1954:1965), labels=1954:1964)
temp.time <- 3700  # total days of fu
py1 <- pyears(temp.time ~ temp.age + temp.yr, scale=1)  # output in days

survexp.uswhite<-survexp.usr[,,'white',]
py2 <- pyears(temp.time ~ temp.age + temp.yr
  + ratetable(age=temp2-temp1, year=temp2, sex=1),
  scale=1, ratetable=survexp.uswhite )  # output in days
```
Ratetable reference in formula

Description

This function matches variable names in data to those in a ratetable for `survexp`.

Usage

```r
ratetable(...)
```

Arguments

... tags matching dimensions of the ratetable and variables in the data frame (see example)

Value

A data frame

See Also

`survexp`, `survexp.us`, `is.ratetable`

Examples

```r
fit <- survfit(Surv(time, status) ~ sex, pbc, subset=1:312)
if (require(date)){
  tdata <- data.frame(agedays=pbc$age *365.24, sex=pbc$sex +1,
                       year= rep(mdy.date(1,1,1976), nrow(pbc)))
  tdata<-tdata[1:312,] ## only the randomised people, with no missing data
  efit <- survexp(~ sex+ratetable(sex=sex, age=agedays, year=year), data=tdata, ratetable=survexp.us, times=(0:24) *182)
}
```

Rat data from survival5

Description

48 rats were injected with a carcinogen, and then randomized to either drug or placebo. The number of tumors ranges from 0 to 13; all rats were censored at 6 months after randomization.
Usage

rats

Format

rat: id
rx: treatment,(1=drug, 0=control)
observation: within rat
start: entry time
stop: exit time
status: event status

Source


---

**residuals.coxph**  
*Calculate Residuals for a ‘coxph’ Fit*

Description

Calculates martingale, deviance, score or Schoenfeld residuals for a Cox proportional hazards model.

Usage

```r
## S3 method for class 'coxph':
residuals(object,
  type=c("martingale", "deviance", "score", "schoenfeld",
         "dfbeta", "dfbetas", "scaledsch","partial"),
  collapse=FALSE, weighted=FALSE, ...)
## S3 method for class 'coxph.null':
residuals(object,
  type=c("martingale", "deviance","score","schoenfeld"),
  collapse=FALSE, weighted=FALSE, ...)
```

Arguments

- **object**  
an object inheriting from class `coxph`, representing a fitted Cox regression model. Typically this is the output from the `coxph` function.

- **type**  
  character string indicating the type of residual desired. Possible values are "martingale","deviance","score","schoenfeld","dfbeta","dfbetas", and "scaledsch". Only enough of the string to determine a unique match is required.
collapse  vector indicating which rows to collapse (sum) over. In time-dependent models more than one row data can pertain to a single individual. If there were 4 individuals represented by 3, 1, 2 and 4 rows of data respectively, then `collapse=c(1,1,1,2,3,3,4,4,4,4)` could be used to obtain per subject rather than per observation residuals.

weighted  if TRUE and the model was fit with case weights, then the weighted residuals are returned.

...  other unused arguments

Value

For martingale and deviance residuals, the returned object is a vector with one element for each subject (without `collapse`). For score residuals it is a matrix with one row per subject and one column per variable. The row order will match the input data for the original fit. For Schoenfeld residuals, the returned object is a matrix with one row for each event and one column per variable. The rows are ordered by time within strata, and an attribute `strata` is attached that contains the number of observations in each strata. The scaled Schoenfeld residuals are used in the `cox.zph` function.

The score residuals are each individual’s contribution to the score vector. Two transformations of this are often more useful: `dfbeta` is the approximate change in the coefficient vector if that observation were dropped, and `dfbetas` is the approximate change in the coefficients, scaled by the standard error for the coefficients.

NOTE

For deviance residuals, the status variable may need to be reconstructed. For score and Schoenfeld residuals, the X matrix will need to be reconstructed.

References


See Also

coxph

Examples

```R
fit <- coxph(Surv(start, stop, event) ~ (age + surgery) * transplant, data=heart)
mresid <- resid(fit, collapse=heart$id)
```
residuals.survreg  Compute Residuals for 'survreg' Objects

Description

This is a method for the function residuals for objects inheriting from class survreg.

Usage

```r
## S3 method for class 'survreg':
residuals(object, type=c("response", "deviance", "dfbeta", "dfbetas", "working", "ldcase", "ldresp", "ldshape", "matrix"), rsigma=TRUE, collapse=FALSE, weighted=FALSE, ...)
```

Arguments

- `object` an object inheriting from class survreg.
- `type` type of residuals, with choices of "response", "deviance", "dfbeta", "dfbetas", "working", "ldcase", "ldresp", "ldshape", and "matrix". See the LaTeX documentation (survival/doc/survival.ps.gz) for more detail.
- `rsigma` include the scale parameters in the variance matrix, when doing computations. (I can think of no good reason not to).
- `collapse` optional vector of subject groups. If given, this must be of the same length as the residuals, and causes the result to be per group residuals.
- `weighted` give weighted residuals? Normally residuals are unweighted.
- `...` other unused arguments

Value

A vector or matrix of residuals is returned. Response residuals are on the scale of the original data, working residuals are on the scale of the linear predictor, and deviance residuals are on log-likelihood scale. The dfbeta residuals are a matrix, where the ith row gives the approximate change in the coefficients due to the addition of subject i. The dfbetas matrix contains the dfbeta residuals, with each column scaled by the standard deviation of that coefficient.

The matrix type produces a matrix based on derivatives of the log-likelihood function. Let L be the log-likelihood, \( p \) be the linear predictor \( \mathbf{X} \beta \), and s be \( \log(\sigma) \). Then the 6 columns of the matrix are \( L \), \( dL/dp \), \( \partial^2 L/\partial p^2 \), \( dL/ds \), \( \partial^2 L/\partial s^2 \) and \( \partial^2 L/\partial p\partial s \). Diagnostics based on these quantities are discussed in an article by Escobar and Meeker. The main ones are the likelihood displacement residuals for perturbation of a case weight (ldcase), the response value (ldresp), and the shape.

References

See Also

predict.survreg

Examples

fit <- survreg(Surv(time,status) ~ x, aml)
rr <- residuals(fit, type='matrix')

ridge

Ridge regression

Description

When used in a coxph or survreg model formula, specifies a ridge regression term. The likelihood is penalised by \( \theta/2 \) time the sum of squared coefficients. If scale=T the penalty is calculated for coefficients based on rescaling the predictors to have unit variance. If df is specified then \( \theta \) is chosen based on an approximate degrees of freedom.

Usage

ridge(..., theta, df=nvar/2, eps=0.1, scale=TRUE)

Arguments

... predictors to be ridged
theta penalty is \( \theta/2 \) time sum of squared coefficients
df Approximate degrees of freedom
eps Accuracy required for df
scale Scale variables before applying penalty?

Value

An object of class coxph.penalty containing the data and control functions.

References

Gray (1992) "Flexible methods of analysing survival data using splines, with applications to breast cancer prognosis" JASA 87:942–951

See Also

coxph,survreg,pspline,frailty
Examples

```r
fit1 <- coxph(Surv(futime, fustat) ~ rx + ridge(age, ecog.ps, theta=1), ovarian)
fit1

lfit0 <- survreg(Surv(time, status) ~1, cancer)
lfit1 <- survreg(Surv(time, status) ~ age + ridge(ph.ecog, theta=5), cancer)
lfit2 <- survreg(Surv(time, status) ~ sex + ridge(age, ph.ecog, theta=1), cancer)
lfit3 <- survreg(Surv(time, status) ~ sex + age + ph.ecog, cancer)
```

stanford2

More Stanford Heart Transplant data

Description

This contains the Stanford Heart Transplant data in a different format. The main data set is in heart.

Usage

stanford2

Format

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>ID number</td>
</tr>
<tr>
<td>time</td>
<td>survival or censoring time</td>
</tr>
<tr>
<td>status</td>
<td>censoring status</td>
</tr>
<tr>
<td>age</td>
<td>in years</td>
</tr>
<tr>
<td>t5</td>
<td>T5 mismatch score</td>
</tr>
</tbody>
</table>

Source


See Also

predict.survreg, heart
Identify Stratum Variables

Description

This is a special function used in the context of the Cox survival model. It identifies stratification variables when they appear on the right hand side of a formula.

Usage

strata(..., na.group=FALSE, shortlabel=FALSE)

Arguments

... any number of variables. All must be the same length.
na.group a logical variable, if TRUE, then missing values are treated as a distinct level of each variable.
shortlabel if TRUE omit variable names from resulting factor labels

Details

The result is identical to the interaction function, but for the labeling of the factors (strata is more verbose).

Value

a new factor, whose levels are all possible combinations of the factors supplied as arguments.

See Also

coxph, interaction

Examples

a<-factor(rep(1:3,4))
b<-factor(rep(1:4,3))
levels(strata(a))
levels(strata(a,b,shortlabel=TRUE))

coxph(Surv(futime, fustat) ~ age + strata(rx), data=ovarian)
Description

Returns a list containing the survival curve, confidence limits for the curve, and other information.

Usage

```r
## S3 method for class 'survfit':
summary(object, times, censored=FALSE, scale=1, ...)
```

Arguments

- `object`: output from a call to `survfit`.
- `times`: vector of times; the returned matrix will contain 1 row for each time. This must be in increasing order and missing values are not allowed. If `censored=T`, the default `times` vector contains all the unique times in `fit`, otherwise the default `times` vector uses only the event (death) times.
- `censored`: logical flag: should the censoring times be included in the output? This is ignored if the `times` argument is present.
- `scale`: rescale the survival time, e.g., if the input data to `survfit` were in days, `scale=365.25` would scale the output to years.
- ... other unused arguments

Value

a list with the following components

- `time`: the timepoint on the curve.
- `surv`: the value of the survival curve at time t+0.
- `n.risk`: the number of subjects at risk at time t-0 (but see the comments on weights in the `survfit` help file).
- `n.event`: if the `times` argument is missing, then this column is the number of events that occurred at time t. Otherwise, it is the cumulative number of events that have occurred since the last time listed until time t+0.
- `std.err`: the standard error of the survival value.
- `conf.int`: level of confidence for the confidence intervals of survival.
- `lower`: lower confidence limits for the curve.
- `upper`: upper confidence limits for the curve.
- `strata`: indicates stratification of curve estimation. If `strata` is not `NULL`, there are multiple curves in the result and the `surv`, `time`, `n.risk`, etc. vectors will contain multiple curves, pasted end to end. The levels of `strata` (a factor) are the labels for the curves.
- `call`: the statement used to create the `fit` object.
- `na.action`: passed through from `fit`, if present.
survSplit

Split a survival data set at specified times

Description

Given a survival data set and a set of specified cut times, split each record into multiple subrecords at each cut time. The new data set will be in ‘counting process’ format, with a start time, stop time, and event status for each record.

Usage

```
survSplit(data, cut, end, event, start, id = NULL, zero = 0, episode=NULL)
```

Arguments

data  data frame
cut   vector of timepoints to cut at
data  character string with name of event time variable
event character string with name of censoring indicator
start character string with name of start time variable (will be created if it does not exist)
id    character string with name of new id variable to create (optional)
zero  If start doesn’t already exist, this is the time that the original records start. May be a vector or single value.
episode character string with name of new episode variable (optional)

Details

The function also works when the original data are in counting-process format, but the id and episode options are of little use in this context.

Value

New, longer, data frame.
survdiff

See Also

Surv, cut, reshape

Examples

aml3 <- survSplit(aml, cut = c(5, 10, 50), end = "time", start = "start",
  event = "status", episode = "i")

summary(aml)
summary(aml3)

coxph(Surv(time, status) ~ x, data = aml)
## the same
coxph(Surv(start, time, status) ~ x, data = aml3)

aml4 <- survSplit(aml3, cut = 20, end = "time", start = "start", event = "status")
coxph(Surv(start, time, status) ~ x, data = aml4)

survdiff

Test Survival Curve Differences

Description

Tests if there is a difference between two or more survival curves using the $G^\rho$ family of tests, or for a single curve against a known alternative.

Usage

survdiff(formula, data, subset, na.action, rho = 0)

Arguments

formula a formula expression as for other survival models, of the form Surv(time, status) ~ predictors. For a one-sample test, the predictors must consist of a single offset(sp) term, where sp is a vector giving the survival probability of each subject. For a k-sample test, each unique combination of predictors defines a subgroup. A strata term may be used to produce a stratified test. To cause missing values in the predictors to be treated as a separate group, rather than being omitted, use the strata function with its na.group=T argument.

data an optional data frame in which to interpret the variables occurring in the formula.

subset expression indicating which subset of the rows of data should be used in the fit. This can be a logical vector (which is replicated to have length equal to the number of observations), a numeric vector indicating which observation numbers are to be included (or excluded if negative), or a character vector of row names to be included. All observations are included by default.
na.action  a missing-data filter function. This is applied to the model.frame after any
subset argument has been used. Default is options()$na.action.

rho       a scalar parameter that controls the type of test.

Value
   a list with components:

   n     the number of subjects in each group.
   obs   the weighted observed number of events in each group. If there are strata, this
          will be a matrix with one column per stratum.
   exp   the weighted expected number of events in each group. If there are strata, this
          will be a matrix with one column per stratum.
   chisq the chisquare statistic for a test of equality.
   var   the variance matrix of the test.
   strata optionally, the number of subjects contained in each stratum.

METHOD
   This function implements the G-rho family of Harrington and Fleming (1982), with weights on each
death of $S(t)^\rho$, where $S(t)$ is the Kaplan-Meier estimate of survival. With $\rho = 0$ this is the log-
rank or Mantel-Haenszel test, and with $\rho = 1$ it is equivalent to the Peto & Peto modification
of the Gehan-Wilcoxon test.

   If the right hand side of the formula consists only of an offset term, then a one sample test is done.
   To cause missing values in the predictors to be treated as a separate group, rather than being omitted,
   use the factor function with its exclude argument.

References
data. Biometrika 69, 553-566.

Examples
   ## Two-sample test
   survdiff(Surv(futime, fustat) ~ rx, data=ovarian)

   ## Stratified 7-sample test
   survdiff(Surv(time, status) ~ pat.karno + strata(inst), data=lung)

   ## Expected survival for heart transplant patients based on
   ## US mortality tables
   expect <- survexp(futime ~ ratetable(age=(accept.dt - birth.dt),
                    sex=1, year=accept.dt, race="white"), jasa, cohort=FALSE,
                    ratetable=survexp.usr)
   ## actual survival is much worse (no surprise)
   survdiff(Surv(jasa$futime, jasa$fustat) ~ offset(expect))
survexp

Compute Expected Survival

Description

Returns either the expected survival of a cohort of subjects, or the individual expected survival for each subject.

Usage

survexp(formula, data, weights, subset, na.action, times, cohort=TRUE, conditional=FALSE, ratetable=survexp.us, scale=1, npoints, se.fit=, model=FALSE, x=FALSE, y=FALSE)

Arguments

formula formula object. The response variable is a vector of follow-up times and is optional. The predictors consist of optional grouping variables separated by the + operator (as in survfit), along with a ratetable term. The ratetable term matches each subject to his/her expected cohort.

data data frame in which to interpret the variables named in the formula, subset and weights arguments.

weights case weights.

subset expression indicating a subset of the rows of data to be used in the fit.

na.action function to filter missing data. This is applied to the model frame after subset has been applied. Default is options()$na.action. A possible value for na.action is na.omit, which deletes observations that contain one or more missing values.

times vector of follow-up times at which the resulting survival curve is evaluated. If absent, the result will be reported for each unique value of the vector of follow-up times supplied in formula.

cohort logical value: if FALSE, each subject is treated as a subgroup of size 1. The default is TRUE.

conditional logical value: if TRUE, the follow-up times supplied in formula are death times and conditional expected survival is computed. If FALSE, the follow-up times are potential censoring times. If follow-up times are missing in formula, this argument is ignored.

ratetable a table of event rates, such as survexp.uswhite, or a fitted Cox model.

scale numeric value to scale the results. If ratetable is in units/day, scale = 365.25 causes the output to be reported in years.

npoints number of points at which to calculate intermediate results, evenly spaced over the range of the follow-up times. The usual (exact) calculation is done at each unique follow-up time. For very large data sets specifying npoints can reduce the amount of memory and computation required. For a prediction from a Cox model npoints is ignored.
survexp

se.fit compute the standard error of the predicted survival. The default is to compute this whenever the routine can, which at this time is only for the Ederer method and a Cox model as the rate table.

model, x, y flags to control what is returned. If any of these is true, then the model frame, the model matrix, and/or the vector of response times will be returned as components of the final result, with the same names as the flag arguments.

Details

Individual expected survival is usually used in models or testing, to 'correct' for the age and sex composition of a group of subjects. For instance, assume that birth date, entry date into the study, sex and actual survival time are all known for a group of subjects. The `survexp.uswhite` population tables contain expected death rates based on calendar year, sex and age. Then `haz <- -log(survexp(death.time ratetable(sex=sex, year=entry.dt, age=(birth.dt-entry.dt)), cohort=F))` gives for each subject the total hazard experienced up to their observed death time or censoring time. This probability can be used as a rescaled time value in models: glm(status 1 + offset(log(haz)), family=poisson) glm(status x + offset(log(haz)), family=poisson) In the first model, a test for intercept=0 is the one sample log-rank test of whether the observed group of subjects has equivalent survival to the baseline population. The second model tests for an effect of variable x after adjustment for age and sex.

Cohort survival is used to produce an overall survival curve. This is then added to the Kaplan-Meier plot of the study group for visual comparison between these subjects and the population at large. There are three common methods of computing cohort survival. In the "exact method" of Ederer the cohort is not censored; this corresponds to having no response variable in the formula. Hakulinen recommends censoring the cohort at the anticipated censoring time of each patient, and Verheul recommends censoring the cohort at the actual observation time of each patient. The last of these is the conditional method. These are obtained by using the respective time values as the follow-up time or response in the formula.

Value

if `cohort=T` an object of class survexp, otherwise a vector of per-subject expected survival values. The former contains the number of subjects at risk and the expected survival for the cohort at each requested time.

References


See Also

`survfit, survexp.us, survexp.fit, pyears, date`
Examples

```r
## compare survival to US population
cancer$year <- rep(as.date("1/1/1980"), nrow(cancer))
efit <- survexp(~ ratetable(sex = sex, year = year, age = age * 365), times = (1:4) * 365, data = cancer)
plot(survfit(Surv(time, status) ~ 1, data = cancer))
lines(efit)

## compare data to Cox model
## fit to randomised patients in Mayo PBC data
m <- coxph(Surv(time, status) ~ edtrt + log(bili) + log(protime) + age + platelet, data = pbc, subset = trt > 0)
## compare Kaplan-Meier to fitted model for 2 edema groups in
## unrandomised patients
plot(survfit(Surv(time, status) ~ edtrt, data = pbc, subset = trt == -9))
lines(survexp(~ edtrt + ratetable(edtrt = edtrt, bili = bili, platelet = platelet, age = age, protime = protime), data = pbc, subset = trt == -9, ratetable = m, cohort = TRUE), col = "purple")
```

survexp.fit

Work Function to Compute Expected Survival

Description

Compute expected survival. This function is not to be called by the user.

Usage

```r
survexp.fit(x, y, times, death, ratetable)
```

Arguments

- `x`: a matrix. The first column contains the group, an integer value that divides the subjects into subsets. Remaining columns must match the dimensions of the `ratetable`, in the correct order.
- `y`: the follow up time for each subject.
- `times`: the vector of times at which a result will be computed.
- `death`: death indicator
- `ratetable`: a rate table, such as `survexp.uswhite`.

Details

For cohort survival it must be the potential censoring time for each subject, ignoring death.

For an exact estimate `times` should be a superset of `y`, so that each subject at risk is at risk for the entire sub-interval of time. For a large data set, however, this can use an inordinate amount of storage and/or compute time. If the `times` spacing is more coarse than this, an actuarial approximation is used which should, however, be extremely accurate as long as all of the returned values are > .99.
For a subgroup of size 1 and \( \text{times} > y \), the conditional method reduces to \( \exp(-h) \) where \( h \) is the expected cumulative hazard for the subject over his/her observation time. This is used to compute individual expected survival.

**Value**

A list containing the number of subjects and the expected survival(s) at each time point. If there are multiple groups, these will be matrices with one column per group.

**WARNING**

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

**See Also**

`survexp`, `survexp.us`

---

### Census Data Sets for the Expected Survival and Person Years Functions

**Description**

Census data sets for the expected survival and person years functions.

**Details**

- **us** total United States population, by age and sex, 1960 to 1980.
- **uswhite** United States white population, by age and sex, 1950 to 1980. This is no longer included, but can be extracted from `survexp.usr` as shown in the examples.
- **usr** United States population, by age, sex and race, 1960 to 1980. Race is white, nonwhite, or black. For 1960 and 1970 the black population values were not reported separately, so the nonwhite values were used.
- **mn** total Minnesota population, by age and sex, 1970 and 1980.
- **mnwhite** Minnesota white population, by age and sex, 1960 to 1980.
- **fl** total Florida population, by age and sex, 1970 and 1980.
- **flr** Florida population, by age, sex and race, 1970-1980. Race is white, nonwhite, or black. For 1970 the black population values were not reported separately, so the nonwhite values were used.
- **az** total Arizona population, by age and sex, 1970 and 1980.
Arizona population, by age, sex and race, 1970-1980. Race is white versus nonwhite. For 1970 the nonwhite population values were not reported separately. In order to make the rate table be a matrix, the 1980 values were repeated. (White and non-white values are quite different).

Each of these tables contains the daily hazard rate for a matched subject from the population, defined as $-\log(1 - q)/365.24$ where $q$ is the 1 year probability of death as reported in the original tables. For age 25 in 1970, for instance, $p = 1 - q$ is the probability that a subject who becomes 25 years of age in 1970 will achieve his/her 26th birthday. The tables are recast in terms of hazard per day entirely for computational convenience. (The fraction .24 in the denominator is based on 24 leap years per century.)

Each table is stored as an array, with additional attributes, and can be subset and manipulated as standard S arrays. Interpolation between calendar years is done “on the fly” by the `survexp` routine.

Some of the deficiencies, e.g., 1970 Arizona non-white, are a result of local (Mayo Clinic) conditions. The data probably exists, but we don’t have a copy it in the library.

The tables have been augmented to contain extrapolated values for 1990 and 2000. The details can be found in Mayo Clinic Biostatistics technical report 63 at [http://www.mayo.edu/hsr/techrpt.html](http://www.mayo.edu/hsr/techrpt.html).

Examples

```r
survexp.uswhite <- survexp.usr[,,"white",]
```

---

## survfit

### Compute a Survival Curve for Censored Data

### Description

Computes an estimate of a survival curve for censored data using either the Kaplan-Meier or the Fleming-Harrington method or computes the predicted survivor function for a Cox proportional hazards model.

### Usage

```r
survfit(formula, data, weights, subset, na.action, newdata, individual=F, conf.int=.95, se.fit=T, type=c("kaplan-meier","fleming-harrington", "fh2"), error=c("greenwood","tsiatis"), conf.type=c("log","log-log", "plain", "none"), conf.lower=c("usual", "peto", "modified"))
```

```r
## S3 method for class 'survfit':
x[,...,drop=FALSE]
basehaz(fit,centered=TRUE)
```
Arguments

formula  A formula object or a \texttt{coxph} object. If a formula object is supplied it must have a \texttt{Surv} object as the response on the left of the \texttt{~} operator and, if desired, terms separated by + operators on the right. One of the terms may be a \texttt{strata} object. For a single survival curve the "\texttt{~ 1}" part of the formula is not required.

data  a data frame in which to interpret the variables named in the formula, or in the \texttt{subset} and the \texttt{weights} argument.

weights  The weights must be nonnegative and it is strongly recommended that they be strictly positive, since zero weights are ambiguous, compared to use of the \texttt{subset} argument.

subset  expression saying that only a subset of the rows of the data should be used in the fit.

na.action  a missing-data filter function, applied to the model frame, after any \texttt{subset} argument has been used. Default is \texttt{options()}$\texttt{na.action}$.

newdata  a data frame with the same variable names as those that appear in the \texttt{coxph} formula. Only applicable when formula is a \texttt{coxph} object. The curve(s) produced will be representative of a cohort who’s covariates correspond to the values in \texttt{newdata}. Default is the mean of the covariates used in the \texttt{coxph} fit.

individual  a logical value indicating whether the data frame represents different time epochs for only one individual (T), or whether multiple rows indicate multiple individuals (F, the default). If the former only one curve will be produced; if the latter there will be one curve per row in \texttt{newdata}.

conf.int  the level for a two-sided confidence interval on the survival curve(s). Default is 0.95.

se.fit  a logical value indicating whether standard errors should be computed. Default is \texttt{TRUE}.

type  a character string specifying the type of survival curve. Possible values are "\texttt{kaplan-meier}", "\texttt{fleming-harrington}" or "\texttt{fh2}" if a formula is given and "\texttt{aalen}" or "\texttt{kaplan-meier}" if the first argument is a \texttt{coxph} object, (only the first two characters are necessary). The default is "\texttt{aalen}" when a \texttt{coxph} object is given, and it is "\texttt{kaplan-meier}" otherwise.

type  either the string "\texttt{greenwood}" for the Greenwood formula or "\texttt{tsiatis}" for the Tsiantis formula, (only the first character is necessary). The default is "\texttt{tsiatis}" when a \texttt{coxph} object is given, and it is "\texttt{greenwood}" otherwise.

conf.type  One of "\texttt{none}", "\texttt{plain}", "\texttt{log}" (the default), or "\texttt{log-log}". Only enough of the string to uniquely identify it is necessary. The first option causes confidence intervals not to be generated. The second causes the standard intervals \texttt{curve} $\pm k \times \texttt{se(curve)}$, where k is determined from \texttt{conf.int}. The log option calculates intervals based on the cumulative hazard or log(survival). The last option bases intervals on the log hazard or log(-log(survival)). These last will never extend past 0 or 1.
**conf.lower**  
controls modified lower limits to the curve, the upper limit remains unchanged.  
The modified lower limit is based on an 'effective n' argument. The confidence  
bands will agree with the usual calculation at each death time, but unlike the  
usual bands the confidence interval becomes wider at each censored observation.  
The extra width is obtained by multiplying the usual variance by a factor $m/n$,  
where $n$ is the number currently at risk and $m$ is the number at risk at the last  
death time. (The bands thus agree with the un-modified bands at each death  
time.) This is especially useful for survival curves with a long flat tail.  
The Peto lower limit is based on the same 'effective n' argument as the mod-
ified limit, but also replaces the usual Greenwood variance term with a simple  
approximation. It is known to be conservative.

- **x**  
a survfit object

- **fit**  
a coxph object

- **centered**  
Compute the baseline hazard at the covariate mean rather than at zero?

- **drop**  
Only FALSE is supported

- **...**  
Other arguments for future expansion

**Details**

Actually, the estimates used are the Kalbfleisch-Prentice (Kalbfleisch and Prentice, 1980, p.86) and  
the Tsiatis/Link/Breslow, which reduce to the Kaplan-Meier and Fleming-Harrington estimates,  
respectively, when the weights are unity. When curves are fit for a Cox model, subject weights of  
$\exp(\sum(\text{coef} \times (x-\text{center})))$ are used, ignoring any value for weights input by the user.  
There is also an extra term in the variance of the curve, due to the variance of the coefficients and  
hence variance in the computed weights.

The Greenwood formula for the variance is a sum of terms $d/(n^a(n-m))$, where $d$ is the number  
of deaths at a given time point, $n$ is the sum of weights for all individuals still at risk at that time,  
and $m$ is the sum of weights for the deaths at that time. The justification is based on a binomial  
argument when weights are all equal to one; extension to the weighted case is ad hoc. Tsiatis  
(1981) proposes a sum of terms $d/(n^a n)$, based on a counting process argument which includes the  
weighted case.

The two variants of the F-H estimate have to do with how ties are handled. If there were 3 deaths  
out of 10 at risk, then the first would increment the hazard by $3/10$ and the second by $1/10 + 1/9  
+ 1/8$. For curves created after a Cox model these correspond to the Breslow and Efron estimates,  
respectively, and the proper choice is made automatically. The fh2 method will give results closer  
to the Kaplan-Meier.

Based on the work of Link (1984), the log transform is expected to produce the most accurate  
confidence intervals. If there is heavy censoring, then based on the work of Dorey and Korn (1987)  
the modified estimate will give a more reliable confidence band for the tails of the curve.

**Value**

- **a survfit object**; see the help on survfit.object for details. Methods defined for survfit  
  objects are provided for print, plot, lines, and points.

- **basehaz**, a dataframe with the baseline hazard, times, and strata.

- The "[" method returns a survfit object giving survival for the selected groups.
References


See Also

print.survfit, plot.survfit, lines.survfit, summary.survfit, survfit.object, coxph, Surv, strata.

Examples

```r
# fit a Kaplan-Meier and plot it
fit <- survfit(Surv(time, status) ~ x, data=aml)
plot(fit)
# plot only 1 of the 2 curves from above
plot(fit[1])
## the other line
lines(fit[2], col="blue")

# fit a cox proportional hazards model and plot the
# predicted survival curve
fit <- coxph(Surv(futime,fustat)~resid.ds+rx+ecog.ps,data=ovarian)
plot( survfit( fit))
```

survfit.object  **Survival Curve Object**

Description

This class of objects is returned by the `survfit` class of functions to represent a fitted survival curve.

Objects of this class have methods for the functions `print`, `summary`, `plot`, `points` and `lines`. The `print.survfit` method does more computation than is typical for a print method and is documented on a separate page.
COMPONENTS

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

- **time** the time points at which the curve has a step.
- **n.risk** the number of subjects at risk at t.
- **n.event** the number of events that occur at time t.
- **surv** the estimate of survival at time t+0. This may be a vector or a matrix.
- **strata** if there are multiple curves, this component gives the number of elements of the `time` etc. vectors corresponding to the first curve, the second curve, and so on. The names of the elements are labels for the curves.
- **std.err** the standard error of the cumulative hazard or -log(survival).
- **upper** upper confidence limit for the survival curve.
- **lower** lower confidence limit for the survival curve.
- **conf.type** the approximation used to compute the confidence limits.
- **conf.int** the level of the confidence limits, e.g. 90 or 95%.
- **na.action** the returned value from the na.action function, if any. It will be used in the printout of the curve, e.g., the number of observations deleted due to missing values.
- **call** the call that produced the object.

SUBSCRIPTS

Survfit objects that contain multiple survival curves can be subscripted. This is most often used to plot a subset of the curves. Usually a single subscript will be used. In one particular case – survival curves for multiple covariate values, from a Cox model that includes a `strata` statement – there is a matrix of curves and 2 subscripts may be used. (In this case `summary.survfit` will also print the data as a matrix).

See Also

- `survfit`, `plot.survfit`, `summary.survfit`, `print.survfit`. 

Description

Internal survival functions
Usage

survreg.fit(x, y, weights, offset, init, controlvals, dist, scale = 0, nstrat = 1, strata, parms = NULL)
survpenal.fit(x, y, weights, offset, init, controlvals, dist, scale = 0, nstrat = 1, strata, pcols, pattr, assign, parms = NULL)
survdiff.fit(y, x, strat, rho = 0)
is.category(x)
mismatch.ratetable(R, ratetable)
## S3 method for class 'ratetable':
as.matrix(x, ...)
is.na.ratetable2(x)
is.na.coxph.penalty(x)
model.newframe(object, newdata, safe = FALSE, response = FALSE, ...)
## S3 method for class 'difftime':
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
coxpenal.df(hmat, hinv, fdiag, assign.list, ptype, nvar, pen1, pen2, sparse)
coxpenal.fit(x, y, strata, offset, init, control, weights, method, rownames, pcols, pattr, assign)
coxph.fit(x, y, strata, offset, init, control, weights, method, rownames)
coxph.wtest(var, b, toler.chol = 1e-09)
agreg.fit(x, y, strata, offset, init, control, weights, method, rownames)
agexact.fit(x, y, strata, offset, init, control, weights, method, rownames)

Details

These are not to be called by the user

survobrien  

O'Brien's Test for Association of a Single Variable with Survival

Description

Peter O'Brien's test for association of a single variable with survival This test is proposed in Biometrics, June 1978.

Usage

survobrien(formula, data)

Arguments

formula  a valid formula for a cox model, without time dependent covariates.
data  a data frame.
Value

A new data frame. The original time and status variables are removed, and have been replaced with `start`, `stop`, and `event`. If a predictor variable is a factor or is protected with `I()`, it is retained as is. Other predictor variables have been replaced with time-dependent logit scores.

Because of the time dependent variables, the new data frame will have many more rows than the original data, approximately #rows * #deaths /2.

METHOD

A time-dependent cox model can now be fit to the new data. The univariate statistic, as originally proposed, is equivalent to single variable score tests from the time-dependent model. This equivalence is the rationale for using the time dependent model as a multivariate extension of the original paper.

In O’Brien’s method, the x variables are re-ranked at each death time. A simpler method, proposed by Prentice, ranks the data only once at the start. The results are usually similar.

References


See Also

`survdiff`

Examples

```r
xx <- survobrien(Surv(futime, fustat) ~ age + factor(rx) + I(ecog.ps),
                 data=ovarian)
coxph(Surv(start, stop, event) ~ age, data=xx)
coxph(Surv(start, stop, event) ~ age + rx + ecog.ps, data=xx)
```

---

**survreg**

Regression for a Parametric Survival Model

Description

Regression for a parametric survival model. These are all time-transformed location models, with the most useful case being the accelerated failure models that use a log transformation.

Usage

```r
survreg(formula=formula(data), data=parent.frame(), weights,
        subset,na.action,dist="weibull", init=NULL, scale=0,
        control=survreg.control(),pars=NULL,model=FALSE, x=FALSE,
        y=TRUE, robust=FALSE, ...)
```
Arguments

- **formula**: a formula expression as for other regression models. See the documentation for `lm` and `formula` for details.
- **data**: optional data frame in which to interpret the variables occurring in the formula.
- **weights**: Optional observation weights
- **subset**: subset of the observations to be used in the fit.
- **na.action**: function to be used to handle any NAs in the data.
- **dist**: assumed distribution for y variable. If the argument is a character string, then it is assumed to name an element from `survreg.distributions`. These include "weibull", "exponential", "gaussian", "logistic", "lognormal" and "loglogistic". Only enough of the name needs to be given to make the choice unique. Otherwise, it is assumed to be a user defined list conforming to the format described in `survreg.distributions`.
- **parms**: a list of fixed parameters. For the t-distribution for instance this is the degrees of freedom; most of the distributions have no parameters.
- **init**: optional vector of initial values for the parameters.
- **scale**: optional fixed value for the scale. If set to <=0 then the scale is estimated.
- **control**: a list of control values, in the format produced by `survreg.control`.
- **model**: if TRUE, the model frame is returned.
- **x**: if TRUE, then the X matrix is returned.
- **y**: if TRUE, then the y vector (or survival times) is returned.
- **robust**: if TRUE, sandwich standard errors are computed. Defaults to TRUE when `formula` contains a `cluster` term.
- **...**: other arguments which will be passed to `survreg.control`.

Value

an object of class `survreg` is returned.

Compatibility note

This routine underwent significant changes from survival4 to survival5. The `survreg.old` function gives a backwards-compatible interface. In S-PLUS the new function is called `survReg` and the old one `survreg`.

See Also

`survreg.object`, `survreg.distributions`, `pspline`, `frailty`, `ridge`, `survreg.old`
survreg.control

Package options for survreg and coxph

Description

These functions check and package the fitting options for `survreg` and `coxph`

Usage

```r
survreg.control(maxiter=30, rel.tolerance=1e-09, failure=1,
                toler.chol=1e-10, iter.max, debug=0, outer.max=10)
coxph.control (eps = 1e-09, toler.chol = .Machine$double.eps^0.75,
               iter.max = 20, toler.inf = sqrt(eps), outer.max = 10)
```

Arguments

- `maxiter` maximum number of iterations
- `rel.tolerance` relative tolerance to declare convergence
- `failure` value of status variable indicating failure
- `toler.chol` Tolerance to declare Cholesky decomposition singular
- `iter.max` same as `maxiter`
- `debug` print debugging information
- `outer.max` maximum number of outer iterations for choosing penalty parameters
- `eps` Tolerance to declare convergence for Cox model
- `toler.inf` An undocumented parameter

Value

A list with the same elements as the input

See Also

`survreg`, `coxph`
Parametric Survival Distributions

Description

List of distributions for accelerated failure models. These are location-scale families for some transformation of time. The entry describes the cdf $F$ and density $f$ of a canonical member of the family.

Usage

survreg.distributions

Format

There are three basic formats; only the first two are used in the built-in distributions

- name: name of distribution
- variance: Variance
- init(x,weights,...): Function returning an initial mean and variance
- deviance(y,scale,parms): Function returning the deviance
- density(x,parms): Function returning $F$, $1 - F, f', f'' / f$
- quantile(p,parms): Quantile function
- scale: Optional fixed value for scale parameter

and for transformations of the time variable

- name: name of distribution
- dist: name of transformed distribution
- trans: transformation (eg log)
- dtrans: derivative of transformation
- itrans: inverse of transformation
- scale: Optional fixed value for scale parameter

For transformations of user-defined families use

- name: name of distribution
- dist: transformed distribution in first format
- trans: transformation (eg log)
- dtrans: derivative of transformation
- itrans: inverse of transformation
- scale: Optional fixed value for scale parameter
Details

There are four basic distributions: `extreme`, `gaussian`, `logistic` and `t`. The last three are parametrised in the same way as the distributions already present in R. The extreme value cdf is

\[ F = 1 - e^{-e^{t}}. \]

When the logarithm of survival time has one of the first three distributions we obtain respectively `weibull`, `lognormal`, and `loglogistic`. The Weibull distribution is not parameterised the same way as in `rweibull`.

The other predefined distributions are defined in terms of these. The `exponential` and `rayleigh` distributions are Weibull distributions with fixed `scale` of 1 and 0.5 respectively, and `loggaussian` is a synonym for `lognormal`.

Parts of the built-in distributions are hardcoded in C, so the elements of `survreg.distributions` in the first format above must not be changed and new ones must not be added. The examples show how to specify user-defined distributions to `survreg`.

See Also

`survreg`, `pnorm`, `plogis`, `pt`

Examples

```r
## not a good fit, but a useful example
survreg(Surv(futime, fustat) ~ ecog.ps + rx, data=ovarian, dist='extreme')

##
my.extreme <- survreg.distributions$extreme
my.extreme$name <- "Xtreme"
survreg(Surv(futime, fustat) ~ ecog.ps + rx, data=ovarian, dist=my.extreme)

## time transformation
survreg(Surv(futime, fustat) ~ ecog.ps + rx, data=ovarian, dist='weibull', scale=1)
my.weibull <- survreg.distributions$weibull
my.weibull$dist <- my.extreme
survreg(Surv(futime, fustat) ~ ecog.ps + rx, data=ovarian, dist=my.weibull, scale=1)

## change the transformation to work in years
## intercept changes by log(365), other coefficients stay the same
my.weibull$trans <- function(y) log(y/365)
my.weibull$itrans <- function(y) exp(365*y)
survreg(Surv(futime, fustat) ~ ecog.ps + rx, data=ovarian, dist=my.weibull, scale=1)

## Weibull parametrisation
y <- rweibull(1000, shape=2, scale=5)
survreg(Surv(y) ~ 1, dist="weibull")
## survreg reports scale=1/2, intercept=log(5)
```

survreg.object  
Parametric Survival Model Object

Description

This class of objects is returned by the \texttt{survreg} function to represent a fitted parametric survival model. Objects of this class have methods for the functions \texttt{print}, \texttt{summary}, \texttt{predict}, and \texttt{residuals}.

COMPONENTS

The following components must be included in a legitimate \texttt{survreg} object.

- **coefficients**: the coefficients of the \texttt{linear.predictors}, which multiply the columns of the model matrix. It does not include the estimate of error (sigma). The names of the coefficients are the names of the single-degree-of-freedom effects (the columns of the model matrix). If the model is over-determined there will be missing values in the coefficients corresponding to non-estimable coefficients.

- **icoef**: coefficients of the baseline model, which will contain the intercept and log(scale), or multiple scale factors for a stratified model.

- **var**: the variance-covariance matrix for the parameters, including the log(scale) parameter(s).

- **loglik**: a vector of length 2, containing the log-likelihood for the baseline and full models.

- **iter**: the number of iterations required

- **linear.predictors**: the linear predictor for each subject.

- **df**: the degrees of freedom for the final model. For a penalized model this will be a vector with one element per term.

- **scale**: the scale factor(s), with length equal to the number of strata.

- **idf**: degrees of freedom for the initial model.

- **means**: a vector of the column means of the coefficient matrix.

- **dist**: the distribution used in the fit.

The object will also have the following components found in other model results (some are optional): \texttt{linear predictors, weights, x, y, model, call, terms} and \texttt{formula}. See \texttt{lm}.

See Also

- \texttt{survreg}, \texttt{lm}
survreg.old

Old (survival4) Regression for a parametric survival model

Description

This routine is a backwards-compatible interface to the improved survreg function, which is better.

Usage

survreg.old(formula, data=sys.frame(sys.parent()), ..., link=c("log", "identity"), dist=c("extreme", "logistic", "gaussian", "exponential", "rayleigh", "weibull"), fixed=list())

Arguments

- **formula**: a formula expression as for other regression models. See the documentation for `lm` and `formula` for details.
- **data**: optional data frame in which to interpret the variables occurring in the formula.
- **...**: other arguments to `survreg`
- **link**: transformation to be used on the y variable.
- **dist**: assumed distribution for the transformed y variable.
- **fixed**: a list of fixed parameters, most often just the scale.

Value

An object of class `survreg` is returned, which inherits from class `glm`.

Examples

```r
survreg.old(Surv(futime, fustat) ~ ecog.ps + rx, ovarian, dist='extreme', link='log', fixed=list(scale=1)) #Fit an exponential
```

tcut

Factors for person-year calculations

Description

Attaches categories for person-year calculations to a variable without losing the underlying continuous representation.

Usage

tcut(x, breaks, labels, scale=1)
Arguments

- `x` numeric/date variable
- `breaks` breaks between categories, which are right-continuous
- `labels` labels for categories
- `scale` Multiply `x` and `breaks` by this.

Value

An object of class `tcut`

See Also

cut, pyears

Examples

```r
temp1 <- mdy.date(6,6,36)
temp2 <- mdy.date(6,6,55)# Now compare the results from person-years
#
temp.age <- tcut(temp2-temp1, floor(c(-1, (18:31 * 365.24))),
    labels=c('0-18', paste(18:30, 19:31, sep='-')))
temp.yr <- tcut(temp2, mdy.date(1,1,1954:1965), labels=1954:1964)
temp.time <- 3700  #total days of fu
py1 <- pyears(temp.time ~ temp.age + temp.yr, scale=1)  #output in days
```

---

tobin

**Tobin’s Tobit data**

Description

Economists fit a parametric censored data model called the ‘tobit’. These data are from Tobin’s original paper.

Usage

tobin

Format

A data frame with 20 observations on the following 3 variables.

- **durable** Durable goods purchase
- **age** Age in years
- **quant** Liquidity ratio (x 1000)
untangle.specials  

Help Process the 'specials' Argument of the 'terms' Function.

Description
Given a terms structure and a desired special name, this returns an index appropriate for subscripting the terms structure and another appropriate for the data frame.

Usage
untangle.specials(tt, special, order=1)

Arguments
- tt: a terms object.
- special: the name of a special function, presumably used in the terms object.
- order: the order of the desired terms. If set to 2, interactions with the special function will be included.

Value
a list with two components:
- vars: a vector of variable names, as would be found in the data frame, of the specials.
- terms: a numeric vector, suitable for subscripting the terms structure, that indexes the terms in the expanded model formula which involve the special.

Examples
```r
formula<-Surv(tt,ss)~x+z+strata(id)
tms<terms(formula,specials="strata")
## the specials attribute
attr(tms,"specials")
## main effects
untangle.specials(tms,"strata")
## and interactions
untangle.specials(tms,"strata",order=1:2)
```
**veteran**

*Veterans’ Administration Lung Cancer study*

---

**Description**

Randomised trial of two treatment regimens for lung cancer. This is a standard survival analysis data set.

**Usage**

**veteran**

**Format**

- **trt**: 1=standard 2=test
- **celltype**: 1=squamous, 2=smallcell, 3=adenoc, 4=large
- **time**: survival time
- **status**: censoring status
- **karno**: Karnofsky performance score (100=good)
- **diagtime**: months from diagnosis to randomisation
- **age**: in years
- **prior**: prior therapy 0=no, 1=yes

**Source**

Index

+Topic **chron**
  - as.date, 5
  - date.ddmmmyy, 19
  - date.mdy, 19
  - date.mmdidayy, 20
  - date.mmdidayyy, 21
  - date.object, 21
  - mdy.date, 29
+Topic **datasets**
  - aml, 3
  - bladder, 7
  - heart, 23
  - lung, 28
  - nwtco, 30
  - ovarian, 31
  - pbc, 31
  - ratetables, 57
  - rats, 43
  - stanford2, 48
  - tobin, 71
  - veteran, 73
+Topic **hplot**
  - plot.survfit, 33
+Topic **internal**
  - survival-internal, 62
+Topic **manip**
  - tcut, 70
+Topic **models**
  - anova.coxph, 3
  - attrassign, 6
  - clogit, 10
+Topic **regression**
  - anova.coxph, 3
  - survreg.object, 69
+Topic **survival**
  - anova.coxph, 3
  - bladder, 7
  - cch, 7
  - clogit, 10
  - cluster, 11
  - colon, 12
  - cox.zph, 12
  - coxph, 14
  - coxph.detail, 16
  - coxph.object, 18
  - frailty, 22
  - heart, 23
  - is.ratetable, 24
  - kidney, 25
  - lines.survfit, 26
  - ovarian, 31
  - plot.cox.zph, 32
  - plot.survfit, 33
  - predict.coxph, 36
  - predict.survreg, 37
  - print.survfit, 38
  - pspline, 39
  - pyears, 40
  - ratetable, 43
  - ratetables, 57
  - rats, 43
  - residuals.coxph, 44
  - residuals.survreg, 46
  - ridge, 47
  - stanford2, 48
  - strata, 49
  - summary.survfit, 50
  - Surv, 1
  - survdiff, 52
  - survexp, 54
  - survexp.fit, 56
  - survfit, 58
  - survfit.object, 61
  - survival-internal, 62
  - survobrien, 63
  - survreg, 64
  - survreg.control, 66
  - survreg.distributions, 67
jasal (heart), 23

kidney, 25

labels.survreg (survreg), 64
legend, 35
leukemia (aml), 3
lines, 27
lines.survfit, 26, 35, 61
lm, 69
lung, 28

match.ratetable
  (survival-internal), 62
Math.date (date.object), 21
Math.ratetable (is.ratetable), 24
Math.Surv (Surv), 1
mdy.date, 5, 19–22, 29
model.frame.coxph (coxph), 14
model.frame.survreg (survreg), 64
model.matrix, 6
model.newframe
  (survival-internal), 62

nwtco, 30

Ops.date (date.object), 21
Ops.ratetable (is.ratetable), 24
Ops.Surv (Surv), 1
ovarian, 31

par, 27, 35
pbc, 31
plogis, 68
plot.cox.zph, 32
plot.date (date.object), 21
plot.survfit, 27, 33, 61, 62
pnorm, 68
points.survfit (lines.survfit), 26
predict, 36
predict.coxph, 36
predict.survreg, 37, 47, 48
print.cox.zph (cox.zph), 12
print.coxph (coxph.object), 18
print.coxph.null (coxph), 14
print.coxph.penal (coxph), 14
print.date (date.object), 21
print.ratetable (ratetable), 43
print.summary.survfit, 51
print.summary.survfit (summary.survfit), 50
print.summary.survreg (survreg), 64
print.Surv (Surv), 1
print.survdiff (survdiff), 52
print.survexp (survexp), 54
print.survfit, 38, 61, 62
print.survreg (survreg.object), 69
print.survreg.penal (survreg), 64
pspline, 16, 23, 39, 47, 65
pt, 68
pyears, 25, 40, 55, 71
ratetable, 42, 43
ratetables, 57
rats, 43
reshape, 52
residuals, 46
residuals.coxph, 17, 18, 44
residuals.survreg, 37, 46
ridge, 16, 23, 40, 47, 65
rweibull, 68

stanford2, 24, 48
stat.anova, 4
strata, 11, 16, 49, 61
summary.coxph (coxph.object), 14
summary.coxph.penal (coxph), 14
Summary.date (date.object), 21
summary.date (date.object), 21
summary.ratetable (ratetable), 43
Summary.Surv (Surv), 1
summary.survfit, 39, 50, 61, 62
summary.survreg (survreg.object), 69
Surv, 1, 8, 11, 13, 16, 42, 52, 61
survdiff, 52, 64
survdiff.fit (survival-internal), 62
survexp, 25, 27, 42, 43, 54, 57, 58
survexp.az (ratetables), 57
survexp.azr (ratetables), 57
survexp.fit, 55, 56
survexp.fl (ratetables), 57
survexp.flr (ratetables), 57
survexp.mn (ratetables), 57
survexp.mnwhite (ratetables), 57
survexp.us, 43, 55, 57

INDEX
survexp.us (ratetables), 57
survexp.usr (ratetables), 57
survexp.wnc (ratetables), 57
survfit, 3, 16, 18, 27, 35, 39, 51, 55, 58, 62
survfit.coxph.null (survfit), 58
survfit.km (survfit), 58
survfit.object, 39, 61, 61
survival-internal, 62
survobrien, 63
survpenal.fit
(survival-internal), 62
survReg (survreg), 64
survreg, 3, 18, 22, 23, 37, 40, 47, 64, 66, 68–70
survreg.control, 66
survreg.distributions, 65, 67
survreg.fit (survival-internal), 62
survreg.object, 65, 69
survreg.old, 65, 70
survSplit, 51
tcut, 70
termplot, 36
terms, 6
tobin, 71
untangle.specials, 72
vcov.coxph (coxph), 14
vcov.survreg (survreg), 64
veteran, 73