Package 'survival'

February 21, 2011

Title Survival analysis, including penalised likelihood.

Maintainer Terry Therneau <therneau.terry@mayo.edu>

Priority recommended

Version 2.36-5

Date 2011-02-01

Depends stats, utils, graphics, splines, R (>= 2.10.0)

LazyData Yes

LazyLoad Yes

Author Terry Therneau, original Splus->R port 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 GPL (>= 2)

URL http://r-forge.r-project.org

Repository CRAN

Date/Publication 2011-02-21 16:20:01

R topics documented:

1

---

aareg   *Aalen's additive regression model for censored data*

Description

Returns an object of class `"aareg"` that represents an Aalen model.

Usage

```
aareg(formula, data, weights, subset, na.action,
   qrtol=1e-07, nmin, dfbeta=FALSE, taper=1,
   test = c('aalen', 'variance', 'nrisk'),
    model=FALSE, x=FALSE, y=FALSE)
```

Arguments

formula   a formula object, with the response on the left of a '~' operator and the terms,
            separated by + operators, on the right. The response must be a `Surv` object. Due
            to a particular computational approach that is used, the model MUST include an
            intercept term. If "-1" is used in the model formula the program will ignore it.

data   data frame in which to interpret the variables named in the `formula`, `subset`,
            and `weights` arguments. This may also be a single number to handle some
            speci al cases – see below for details. If `data` is missi ng, the variables in the
            model formula should be in the search path.

weights	vector of observation weights. If supplied, the fitting algorithm minimizes the sum of the weights multiplied by the squared residuals (see below for additional technical details). The length of `weights` must be the same as the number of observations. The weights must be nonnegative and it i s recommended that they be strictly positive, since zero weights are ambiguous. To exclude particular observations from the model, use the `subset` argument instead of zero weights.
subset	expression specifying which subset of observations should be used in the fit. Th is can be a logical vector (which is replicated to have length equal to the numb er of observations), a numeric vector indicating the observation numbers to be i ncluded, or a character vector of the observation names that should be included. All observations are included by default.
na.action	a function to filter missing data. This is applied to the `model.fr ame` after any `subset` argument has be en applied. The default is `na.fail`, which returns a n error if any missing values are found. An alternative is `na.excl ude`, which deletes observations that contain one or more missing values.
qrtol	tolerance for detection of singularity in the QR decomposition
nmin	minimum number of observations for an estimate; defaults to 3 times the number of covariates. This essentially truncates the computations near the tail of the data set, when n is small and the calcualtions can become numerically unstable.
dfbeta	should the array of dfbeta residuals be computed. This implies computation of the sandwich variance estimate. The residuals will always be computed if there is a `cluster` term in the model formula.
taper	allows for a smoothed variance estimate. Var(x), where x is the set of covariates, is an important component of the calculations for the Aalen regression model. At any given time point t, it is computed over all subjects who are still at risk at time t. The tape argument allows smoothing these estimates, for example `taper=(1:4)/4` would cause the variance estimate used at any event time to be a weighted average of the estimated variance matrices at the last 4 death times, with a weight of 1 for the current death time and decreasing to 1/4 for prior event times. The default value gives the standard Aalen model.
test	selects the weighting to be used, for computing an overall "average" coefficient vector over time and the subsequent test for equality to zero.
model, x, y	should copies of the model frame, the x matrix of predictors, or the response vector y be included in the saved result.

Details

The Aalen model assumes that the cumulative hazard H(t) for a subject can be expressed as a(t) + X B(t), where a(t) is a time-dependent intercept term, X is the vector of covariates for the subject (possibly time-dependent), and B(t) is a time-dependent matrix of coefficients. The estimates are inheritly non-parametric; a fit of the model will normally be followed by one or more plots of the estimates.

The estimates may become unstable near the tail of a data set, since the increment to B at time t is based on the subjects still at risk at time t. The tolerance and/or nmin parameters may act to truncate the estimate before the last death. The `taper` argument can also be used to smooth out the tail of the curve. In practice, the addition of a taper such as 1:10 appears to have little effect on death times when n is still reasonably large, but can considerably dampen wild occilations in the tail of the plot.

Value

an object of class `"aareg"` representing the fit.

References

Aalen, O.O. (1989). A linear regression model for the analysis of life times. Statistics in Medicine, 8:907-925.

Aalen, O.O (1993). Further results on the non-parametric linear model in survival analysis. Statistics in Medicine. 12:1569-1588.

See Also

print.aareg, summary.aareg, plot.aareg

Examples

```
# Fit a model to the lung cancer data set
lfit <- aareg(Surv(time, status) ~ age + sex + ph.ecog, data=lung,
                     nmin=1)
## Not run:
lfit
Call:
aareg(formula = Surv(time, status) ~ age + sex + ph.ecog, data = lung, nmin = 1
        )

  n=227 (1 observations deleted due to missing values)
    138 out of 138 unique event times used

              slope       coef se(coef)     z          p
Intercept  5.26e-03   5.99e-03 4.74e-03  1.26 0.207000
      age  4.26e-05   7.02e-05 7.23e-05  0.97 0.332000
      sex -3.29e-03  -4.02e-03 1.22e-03 -3.30 0.000976
  ph.ecog  3.14e-03   3.80e-03 1.03e-03  3.70 0.000214

Chisq=26.73 on 3 df, p=6.7e-06; test weights=aalen

plot(lfit[4], ylim=c(-4,4))  # Draw a plot of the function for ph.ecog

## End(Not run)
lfit2 <- aareg(Surv(time, status) ~ age + sex + ph.ecog, data=lung,
                 nmin=1, taper=1:10)
## Not run: lines(lfit2[4], col=2)  # Nearly the same, until the last point

# A fit to the mulitple-infection data set of children with
# Chronic Granuomatous Disease.  See section 8.5 of Therneau and Grambsch.
fita2 <- aareg(Surv(tstart, tstop, status) ~ treat + age + inherit +
                         steroids + cluster(id), data=cgd)
## Not run:
  n= 203
    69 out of 70 unique event times used
```

```
                      slope      coef se(coef) robust se      z         p
Intercept          0.004670  0.017800 0.002780  0.003910   4.55 5.30e-06
treatrIFN-g       -0.002520 -0.010100 0.002290  0.003020  -3.36 7.87e-04
age               -0.000101 -0.000317 0.000115  0.000117  -2.70 6.84e-03
inheritautosomal   0.001330  0.003830 0.002800  0.002420   1.58 1.14e-01
steroids           0.004620  0.013200 0.010600  0.009700   1.36 1.73e-01

Chisq=16.74 on 4 df, p=0.0022; test weights=aalen

## End(Not run)
```

---

aml   *Acute Myelogenous Leukemia survival data*

Description

Survival in patients with Acute Myelogenous Leukemia. The question at the time was whether the standard course of chemotherapy should be extended ('maintainance') for additional cycles.

Usage

```
aml
leukemia
```

Format

time:	survival or censoring time
status:	censoring status
x:	maintenance chemotherapy given? (factor)

Source

Rupert G. Miller (1997), *Survival Analysis*. John Wiley & Sons. ISBN: 0-471-25218-2.

---

anova.coxph   *Analysis of Deviance for a Cox model.*

Description

Compute an analysis of deviance table for one or more Cox model fits.

Usage

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

Arguments

object	An object of class coxph
...	Further coxph objects
test	a character string. The appropriate test is a chisquare, all other choices result in no test being done.

Details

Specifying a single object gives a sequential analysis of deviance table for that fit. That is, the reductions in the model log-likelihood as each term of the formula is added in turn are given in as the rows of a table, plus the log-likelihoods themselves. A robust variance estimate is normally used in situations where the model may be mis-specified, e.g., multiple events per subject. In this case a comparison of partial-likelihood values does not make sense, and anova will refuse to print results.

If more than one object is specified, the table has a row for the degrees of freedom and loglikelihood for each model. For all but the first model, the change in degrees of freedom and loglik 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 loglik for each row.

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

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

attrassign	*Create new-style "assign" attribute*

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 alternate version found in S-plus 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

Details

For instance consider the following

```
survreg(Surv(time, status) ~ age + sex + factor(ph.ecog), lung)
```

R gives the compact for for assign, a vector (0, 1, 2, 3, 3, 3); which can be read as "the first column of the X matrix (intercept) goes with none of the terms, the second column of X goes with term 1 of the model equation, the third column of X with term 2, and columns 4-6 with term 3".

The alternate (S-Plus default) form is a list

```
$(Intercept)      1
$age              2
$sex              3
$factor(ph.ecog)  4 5 6
```

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

```
formula <- Surv(time,status)~factor(edema)
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")
## alternate style assign attribute
attrassign(mm,tt)
```

---

basehaz	*Compute the baseline survival curve for a Cox model*

Description

Compute the baseline survival curve for a Cox model.

Usage

```
basehaz(fit, centered = TRUE)
```

Arguments

fit   The result of a coxph fit.

centered   If TRUE, the resultant curve is for a hypothetical subject whose covariate values
            are the corresponding means from the original data, otherwise for a hypothetical
            subject with a mean vector of zero.

Details

This function exists primarily because users will look for the phrase 'baseline hazard' (often SAS
converts looking for familiar keywords.) The primary function for creating a survival curve is
survfit, which this calls. See that manual page for more options, including confidence limits
and the ability to use other covariate vectors. The result of survfit also has print, plot and
summary methods that make it far more useful.

Value

a data frame with components

time   The sorted vector of unique time points (those at which an event occurred

hazard   The baseline hazard function

strata   If fit was a stratified Cox model, the strata. There will be one survival curve
            per strata.

See Also

survfit

---

bladder   *Bladder Cancer Recurrences*

Description

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

Bladder1 is the full data set from the study. It contains all three treatment arms and all recurrences for 118 subjects; the maximum observed number of recurrences is 9.

Bladder is the data set that appears most commonly in the literature. It uses only the 85 subjects with nonzero follow-up who were assigned to either thiotepa or placebo. The status variable is 1 for recurrence and 0 for everything else (including death for any reason). The data set is laid out in the competing risks format of the paper by Wei, Lin, and Weissfeld.

Bladder2 uses the same subset of subjects as bladder, but formated in the (start, stop] or Anderson-Gill style. Note that in transforming from the WLW to the AG style data set there is a quite common programming mistake that leads to extra follow-up time for 12 subjects (all those with more than 4 recurrences); this includes some earlier releases of the data in R.

Usage

```
bladder1
bladder
bladder2
```

Format

bladder1

id:	Patient id
treatment:	Placebo, pyridoxine (vitamin B6), or thiotepa
number:	Initial number of tumours (8=8 or more)
size:	Size (cm) of largest initial tumour
recur:	Number of recurrences
start,stop:	The start and end time of each time interval
status:	End of interval code, 0=censored, 1=recurrence,
	2=death from bladder disease, 3=death other/unknown cause
rtumor:	Number of tumors found at the time of a recurrence
rsize:	Size of largest tumor at a recurrence
enum:	Event number (observation number within patient)

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

Andrews DF, Hertzberg AM (1985), DATA: A Collection of Problems from Many Fields for the Student and Research Worker, New York: Springer-Verlag.

LJ Wei, DY Lin, L Weissfeld (1989), Regression analysis of multivariate incomplete failure time data by modeling marginal distributions. *Journal of the American Statistical Association*, **84**.

---

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

```
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 indicatorsfor 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

Prentice, RL (1986). A case-cohort design for epidemiologic cohort studies and disease prevention trials. Biometrika 73: 1–11.

Self, S and Prentice, RL (1988). Asymptotic distribution theory and efficiency results for case-cohort studies. Annals of Statistics 16: 64–81.

Lin, DY and Ying, Z (1993). Cox regression with incomplete covariate measurements. Journal of the American Statistical Association 88: 1341–1349.

Barlow, WE (1994). Robust variance estimation for the case-cohort design. Biometrics 50: 1064–1072

Therneau, TM and Li, H (1999). Computing the Cox model for case-cohort designs. Lifetime Data Analysis 5: 99–112.

Borgan, *O*, Langholz, B, Samuelsen, SO, Goldstein, L and Pogoda, J (2000) Exposure stratified case-cohort designs. Lifetime Data Analysis 6, 39-58.

See Also

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

Examples

```
## 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)
```

---

cgd   *Chronic Granulotomous Disease data*

Description

Data are from a placebo controlled trial of gamma interferon in chronic granulotomous disease
(CGD). Uses the complete data on time to first serious infection observed through end of study
for each patient, which includes the initial serious infections observed through the 7/15/89 interim
analysis data cutoff, plus the residual data on occurence of initial serious infections between the
interim analysis cutoff and the final blinded study visit for each patient. Only one patient was taken
off on the day of his last infection.

Usage

```
  cgd
```

Format

id:	subject identifiction number
center:	enrolling center
random:	date of randomization
treatment:	placebo or gamma interferon
sex:	
age:	age in years, at study entry
height:	height in cm at study entry
weight:	weight in kg at study entry
inherit:	pattern of inheritance
steroids:	use of steroids at study entry,1=yes
propylac:	use of prophylactic antibiotics at study entry
hos.cat:	a categorization of the centers into 4 groups
tstart, tstop:	start and end of each time interval
status:	1=the interval ends with an infection
enum:	observation number within subject

Source

Fleming and Harrington, Counting Processes and Survival Analysis, appendix D.2.

| 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.act
```

Arguments

formula	Model formula
data	data frame
method	correct or approximate conditonal 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

```
## Not run: clogit(case~spontaneous+induced+strata(stratum),data=infert)
```

---

cluster   *Identify clusters.*

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.

Details

The function's only action is semantic, to mark a variable as the cluster indicator. The resulting variance is what is known as the "working independence" variance in a GEE model.

Value

x

See Also

coxph, survreg

Examples

```
marginal.model <- coxph(Surv(time, status) ~ rx + cluster(litter), rats)
frailty.model  <- coxph(Surv(time, status) ~ rx + frailty(litter), rats)
```

---

colon   *Chemotherapy for Stage B/C colon cancer*

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
time:	days until event or censoring
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
etype:	event type: 1=recurrence,2=death

Source

Danyu Lin

`cox.zph`	*Test the Proportional Hazards Assumption of a Cox Regression*

Description

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

Usage

```
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.
`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 `slope=0`.

References

P. Grambsch and T. Therneau (1994), Proportional hazards tests and diagnostics based on weighted residuals. *Biometrika,* **81**, 515-26.

See Also

`coxph`, `Surv`.

Examples

```
fit <- coxph(Surv(futime, fustat) ~ age + ecog.ps,
             data=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

```
coxph(formula, data=, 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.
weights	vector of case weights. If weights is a vector of integers, then the estimated coefficients are equivalent to estimating the model from data with the individual cases replicated as many times as indicated by weights.
subset	expression indicating which subset of the rows of data should be used in the fit. 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.
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 cluster operative, FALSE otherwise.
model	logical value: if TRUE, the model frame is returned in component model.
x	logical value: if TRUE, the model frame is returned in component x.
y	logical value: if TRUE, the model frame is returned in component y.
...	Other arguments will be passed to [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 `"coxph"` representing the fit. See `coxph.object` for details.

Side Effects

Depending on the call, the `predict`, `residuals`, and `survfit` routines may need to reconstruct the x matrix created by `coxph`. Differences in the environment, such as which data frames are attached or the value of `options()$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 `cluster` term is used to compute a robust variance for the model. The term `+ cluster(id)` where each value of `id` is unique is equivalent to specifying the `robust=T` argument, and produces an approximate jackknife estimate of the variance. If the `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 exp becomes too large for the computer hardware, or the maximum number of interactions is exceeded. (Nearly always the first occurs.) The routine attempts to detect when this has happened, not always successfully. The primary consequence for he user is that the Wald statistic, coefficient/se(coefficient), is not valid in this case and should be ignored; the likelihood ratio and Wald tests remain valid however.

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

Andersen, P. and Gill, R. (1982). Cox's regression model for counting processes, a large sample study. *Annals of Statistics* **10**, 1100-1120.

Therneau, T., Grambsch, P., Modeling Survival Data: Extending the Cox Model. Springer-Verlag, 2000.

See Also

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

Examples

```
# Create the simplest test data set
test1 <- list(time=c(4,3,1,1,2,2,3),
               status=c(1,1,1,0,1,1,0),
               x=c(0,2,1,1,1,0,0),
               sex=c(0,0,0,0,1,1,1))
# Fit a stratified model
coxph(Surv(time, status) ~ x + strata(sex), test1)
# 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(1,1,1,1,1,1,1,0,0,0),
               x=c(1,0,0,1,0,1,1,1,0,0))
summary(coxph(Surv(start, stop, event) ~ x, test2))


#
# 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, 1, 0, 0, 0),
                 x    =c(1, 0, 0, 1, 0, 1, 1, 1, 0, 0) )


summary( coxph( Surv(start, stop, event) ~ x, test2))

# Fit a stratified model, clustered on patients

bladder1 <- bladder[bladder$enum < 5, ]
coxph(Surv(stop, event) ~ (rx + size + number) * strata(enum) +
      cluster(id), bladder1)
```

---

coxph.control	*Ancillary arguments for controling coxph fits*

Description

This is used to set various numeric parameters controling a Cox model fit. Typically it would only be used in a call to coxph.

Usage

```
coxph.control(eps = 1e-09, toler.chol = .Machine$double.eps^0.75,
iter.max = 20, toler.inf = sqrt(eps), outer.max = 10)
```

Arguments

`eps`	Iteration continues until the relative change in the log partial likelihood is less than eps. Must be positive.
`toler.chol`	Tolerance for detection of singularity during a Cholesky decomposion of the variance matrix, i.e., for detecting a redundant predictor variable.
`iter.max`	Maximum number of iterations to attempt for convergence.
`toler.inf`	Tolerance criteria for the warning message about a possible infinite coefficient value.
`outer.max`	For a penalized coxph model, e.g. with pspline terms, there is an outer loop of iteration to determine the penalty parameters; maximum number of iterations for this outer loop.

Value

a list containing the values of each of the above constants

Author(s)

Terry Therneau

See Also

coxph

`coxph.detail`	*Details of a Cox Model Fit*

Description

Returns the individual contributions to the first and second derivative matrix, at each unique event time.

Usage

```
coxph.detail(object, riskmat=FALSE)
```

Arguments

`object`	a Cox model object, i.e., the result of `coxph`.
`riskmat`	include the at-risk indicator matrix in the output?

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$coef)`.
nrisk	number of subjects at risk.
score	the contribution to the score vector (first derivative of the log partial likelihood) at each time point.
imat	the contribution to the information matrix (second derivative of the log partial likelihood) at each time point.
hazard	the hazard increment. Note that the hazard and variance of the hazard are always for some particular future subject. This routine uses `object$mean` as the future subject.
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 `time` that were contributed by each of the strata.
riskmat	a matrix with one row for each time and one column for each observation containing a 0/1 value to indicate whether that observation was (1) or was not (0) at risk at the given time point.

See Also

coxph, residuals.coxph

Examples

```
fit   <- coxph(Surv(futime,fustat) ~ age + rx + ecog.ps, ovarian, x=TRUE)
fitd <- coxph.detail(fit)
#  There is one Schoenfeld residual for each unique death.  It is a
# vector (covariates for the subject who died) - (weighted mean covariate
# vector at that time).  The weighted mean is defined over the subjects
# still at risk, with exp(X beta) as the weight.

events <- fit$y[,2]==1
etime  <- fit$y[events,1]   #the event times --- may have duplicates
indx   <- match(etime, fitd$time)
schoen <- fit$x[events,] - fitd$means[indx,]
```

| `coxph.object` | *Proportional Hazards Regression Object* |

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`.

Arguments

`coefficients`	the vector of coefficients. 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. Note that this vector has been centered, see `predict.coxph` for more details.
`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.
`nevent`	the number of events (usually deaths) 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`.

Components

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

See Also

`coxph`, `coxph.detail`, `cox.zph`, `residuals.coxph`, `survfit`, `survreg`.

---

dsurvreg   *Distributions available in survreg.*

Description

Density, cumulative probability, and quantiles for the set of distributions supported by the `survreg` function.

Usage

```
dsurvreg(x, mean, scale=1, distribution='weibull', parms)
psurvreg(q, mean, scale=1, distribution='weibull', parms)
qsurvreg(p, mean, scale=1, distribution='weibull', parms)
```

Arguments

x	vector of quantiles. Missing values (`NA`s) are allowed.
q	vector of quantiles. Missing values (`NA`s) are allowed.
p	vector of probabilities. Missing values (`NA`s) are allowed.
mean	vector of means. This is replicated to be the same length as `p` or `q`.
scale	vector of (positive) scale factors. This is replicated to be the same length as `p` or `q`.
distribution	character string giving the name of the distribution. This must be one of the elements of `survreg.distributions`
parms	optional parameters, if any, of the distribution. For the t-distribution this is the degrees of freedom.

Details

Elements of `q` or `p` that are missing will cause the corresponding elements of the result to be missing.

The `mean` and `scale` values are as they would be for `survreg`. In particular, if the distribution is one that involves a transformation, then they are the mean and scale of the transformed distribution. For example, the Weibull distribution is fit using the Extreme value distribution along with a log transformation. Letting $F(t) = 1 - \exp[-(at)^p]$ be the cumulative distribution of the Weibull, the mean corresponds to $-\log(a)$ and the scale to $1/p$ (Kalbfleish and Prentice, section 2.2.2).

Value

density (`dsurvreg`), probability (`psurvreg`), quantile (`qsurvreg`), or for the requested distribution with mean and scale parameters `mean` and `sd`.

References

Kalbfleish, J. D. and Prentice, R. L. (1970). *The Statistical Analysis of Failure Time Data* Wiley, New York.

See Also

survreg, Normal

Examples

```
# List of distributions available
names(survreg.distributions)
## Not run:
 [1] "extreme"    "logistic"   "gaussian"    "weibull"     "exponential"
 [6] "rayleigh"    "loggaussian" "lognormal"   "loglogistic" "t"

## End(Not run)
# Compare results
all.equal(dsurvreg(1:10, 2, 5, dist='lognormal'), dlnorm(1:10, 2, 5))

# Hazard function for a Weibull distribution
x   <- seq(.1, 3, length=30)
haz <- dsurvreg(x, 2, 3)/ (1-psurvreg(x, 2, 3))
## Not run:
plot(x, haz, log='xy', ylab="Hazard") #line with slope (1/scale -1)

## End(Not run)
```

---

frailty | *Random effects terms*

Description

The frailty function allows one to add a simple random effects term to a Cox or survreg model.

Usage

```
frailty(x, distribution="gamma", ...)
frailty.gamma(x, sparse = (nclass > 5), theta, df, eps = 1e-05, method = c("em","ai
frailty.gaussian(x, sparse = (nclass > 5), theta, df, method =c("reml","aic", "df",
frailty.t(x, sparse = (nclass > 5), theta, df, eps = 1e-05, tdf = 5,method = c("aic
```

Arguments

x   the variable to be entered as a random effect. It is always treated as a factor.

distribution either the gamma, gaussian or t distribution may be specified. The rou-
              tines frailty.gamma, frailty.gaussian and frailty.t do the ac-
              tual work.

...   Arguments for specific distribution, including (but not limited to)

sparse   cutoff for using a sparse coding of the data matrix. If the total number of levels
              of x is larger than this value, then a sparse matrix approximation is used. The
              correct cutoff is still a matter of exploration: if the number of levels is very large

(thousands) then the non-sparse calculation may not be feasable in terms of both memory and compute time. Likewise, the accuracy of the sparse approximation appears to be related to the maximum proportion of subjects in any one class, being best when no one class has a large membership.

theta   if specified, this fixes the variance of the random effect. If not, the variance is a parameter, and a best solution is sought. Specifying this implies method='fixed'.

df   if specified, this fixes the degrees of freedom for the random effect. Specifying this implies method='df'. Only one of theta or df should be specified.

method   the method used to select a solution for theta, the variance of the random effect. The fixed corresponds to a user-specified value, and no iteration is done. The df selects the variance such that the degrees of freedom for the random effect matches a user specified value. The aic method seeks to maximize Akiake's information criteria 2*(partial likelihood - df). The em and reml methods are specific to Cox models with gamma and gaussian random effects, respectively. Please see further discussion below.

tdf   the degrees of freedom for the t-distribution.

eps   convergence critera for the iteration on theta.

Details

The frailty plugs into the general penalized modeling framework provided by the coxph and survreg routines. This framework deals with likelihood, penalties, and degrees of freedom; these aspects work well with either parent routine.

Therneau, Grambsch, and Pankratz show how maximum likelihood estimation for the Cox model with a gamma frailty can be accomplished using a general penalized routine, and Ripatti and Palmgren work through a similar argument for the Cox model with a gaussian frailty. Both of these are specific to the Cox model. Use of gamma/ml or gaussian/reml with survreg does not lead to valid results.

The extensible structure of the penalized methods is such that the penalty function, such as frailty or pspine, is completely separate from the modeling routine. The strength of this is that a user can plug in any penalization routine they choose. A weakness is that it is very difficult for the modeling routine to know whether a sensible penalty routine has been supplied.

For Cox models the coxme package has replaced superseded this method.

Value

this function is used in the model statment of either coxph or survreg. It's results are used internally.

References

S Ripatti and J Palmgren, Estimation of multivariate frailty models using penalized partial likelihood, Biometrics, 56:1016-1022, 2000.

T Therneau, P Grambsch and VS Pankratz, Penalized survival models and frailty, J Computational and Graphical Statistics, 12:156-175, 2003.

See Also

coxph, survreg

Examples

```
# Random institutional effect
coxph(Surv(time, status) ~ age + frailty(inst, df=4), lung)

# Litter effects for the rats data
rfit2a <- survreg(Surv(time, status) ~ rx +
                  frailty.gaussian(litter, df=13, sparse=FALSE), rats )
rfit2b <- survreg(Surv(time, status) ~ rx +
                  frailty.gaussian(litter, df=13, sparse=TRUE), rats )
```

---

heart	*Stanford Heart Transplant data*

Description

Survival of patients on the waiting list for the Stanford heart transplant program.

Usage

```
heart
jasa
jasa1
```

Format

jasa: original data

birth.dt:	birth date
accept.dt:	acceptance into program
tx.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

jasa1, 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

J Crowley and M Hu (1977), Covariance analysis of heart transplant survival data. *Journal of the American Statistical Association*, **72**, 27–36.

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

```
is.ratetable(x, verbose=FALSE)
```

Arguments

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

Details

Rate tables are used by the pyears and 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 verbose option can help in creating a new rate table.

Value

returns TRUE if x is a ratetable, and FALSE or a description if it is not.

See Also

pyears, survexp.

Examples

```
is.ratetable(survexp.us)   # True
is.ratetable(cancer)       # False
```

---

    kidney   *Kidney catheter data*

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.

This data has often been used to illustrate the use of random effects (frailty) in a survival model. However, one of the males (id 21) is a large outlier, with much longer survival than his peers. If this observation is removed no evidence remains for a random subject effect.

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 paper ignored the issue of tied times and so is not exactly reproduced by the survival package.

Source

CA McGilchrist, CW Aisbett (1991), Regression with frailty in survival analysis. *Biometrics* **47**, 461–66.

Examples

```
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)
```

---

lines.survfit   *Add Lines or Points to a Survival Plot*

Description

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

Usage

```
## 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, xscale, xmax, fun, ...)
```

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, col, lty, lwd	
	vectors giving the mark symbol, color, line type and line width for the added curves.
...	other graphical parameters
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 365.25 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.
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 cumu-
> lative 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).

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

```
fit <- survfit(Surv(time, status==2) ~ sex, pbc,subset=1:312)
plot(fit, mark.time=FALSE, xscale=365.25,
        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
# The data set does not have entry date, use the midpoint of the study
efit <- survexp(~ ratetable(sex=sex,age=age*365.35,year=as.Date('1979/1/1')) +
            sex, data=pbc,  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")
```

---

  lung   *NCCTG Lung Cancer Data*

Description

Survival in patients with advanced lung cancer from the North Central Cancer Treatment Group.
Performance scores rate how well the patient can perform usual daily activities.

Usage

```
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 as rated by patient
meal.cal:	Calories consumed at meals
wt.loss:	Weight loss in last six months

Source

Terry Therneau

References

Loprinzi CL. Laurie JA. Wieand HS. Krook JE. Novotny PJ. Kugler JW. Bartel J. Law M. Bateman M. Klatt NE. et al. Prospective evaluation of prognostic variables from patient-completed questionnaires. North Central Cancer Treatment Group. Journal of Clinical Oncology. 12(3):601-7, 1994.

---

mgus	*Monoclonal gammapothy data*

Description

Natural history of 241 subjects with monoclonal gammapothy of undetermined significance (MGUS).

Usage

```
mgus
mgus1
mgus2
```

Format

mgus: A data frame with 241 observations on the following 12 variables.

id:	subject id
age:	age in years
sex:	`male` or `female`

dxyr:	year of diagnosis
pcdx:	for subjects who progress to a plasma cell malignancy
	the subtype of malignancy: multiple myeloma (MM) is the
	most common, followed by amyloidosis (AM), macroglobulinemia (MA),
	and other lymphprolifative (LP)
pctime:	days from MGUS until diagnosis of a plasma cell malignancy
futime:	days from diagnosis to last follow-up
death:	1= follow-up is until death
alb:	albumin level at MGUS diagnosis
creat:	creatinine at MGUS diagnosis
hgb:	hemoglobin at MGUS diagnosis
mspike:	size of the monoclonal protien spike at diagnosis

mgus1: The same data set in start,stop format. Contains the id, age, sex, and laboratory variable described above along with

start, stop:	sequential intervals of time for each subject
status:	=1 if the interval ends in an event
event:	the event type

mgus2: The mgus data, but formatted in the competing risks style. Each subject has three observations, one for time to death, one for time to MM, and one for time to a PC malignancy other than MM. Contains the id, age, sex, and laboratory variable described above along with

time:	time to event or censoring
status:	1 if the event occured, 0 otherwise
event:	death, myeloma, or other

Details

Plasma cells are responsible for manufacturing immunoglobulins, an important part of the immune defense. At any given time there are estimated to be about 10^6 different immunoglobulins in the circulation at any one time. When a patient has a plasma cell malignancy the distribuion will become dominated by a single isotype, the product of the malignant clone, visible as a spike on a serum protein electrophoresis. Monoclonal gammapothy of undertermined significance (MGUS) is the presence of such a spike, but in a patient with no evidence of overt malignancy. This data set of 241 sequential subjects at Mayo Clinic was the groundbreaking study defining the natural history of such subjects.

Source

Mayo Clinic data courtesy of Dr. Robert Kyle.

References

R Kyle, Benign monoclonal gammopathy – after 20 to 35 years of follow-up, Mayo Clinic Proc 1993; 68:26-36.

---

model.frame.coxph   *Model.frame method for coxph objects*

Description

Recreate the model frame of a coxph fit.

Usage

```
## S3 method for class 'coxph'
model.frame(formula, ...)
```

Arguments

formula	the result of a coxph fit
...	other arguments to model.frame

Details

For details, see the manual page for the generic function. This function would rarely be called by a user, it is mostly used inside functions like residual that need to recreate the data set from a model in order to do further calculations.

Value

the model frame used in the original fit, or a parallel one for new data.

Author(s)

Terry Therneau

See Also

model.frame

---

model.matrix.coxph *Model.matrix method for coxph models*

Description

Reconstruct the model matrix for a cox model.

Usage

```
## S3 method for class 'coxph'
model.matrix(object, data, contrast.arg =
 object$contrasts, mf, ...)
```

Arguments

object	the result of a coxph model
data	optional, a data frame from which to obtain the data
contrast.arg	optional, a contrasts object describing how factors should be coded
mf	an optional model frame. Used interally when programs have already obtained a model frame.
...	other possible argument to model.frame

Details

When there is a data argument this function differs from most of the other model.matrix methods in that the response variable for the original formual is *not* required to be in the data.

Value

The model matrix for the fit

Author(s)

Terry Therneau

See Also

model.matrix

Examples

```
fit1 <- coxph(Surv(time, status) ~ age + factor(ph.ecog), data=lung)
xfit <- model.matrix(fit1)

fit2 <- coxph(Surv(time, status) ~ age + factor(ph.ecog), data=lung,
                            x=TRUE)
all.equal(model.matrix(fit1), fit2$x)
```

---

nwtco   *Data from the National Wilm's Tumor Study*

Description

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

Usage

```
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

NE Breslow and N Chatterjee (1999), Design and analysis of two-phase studies with binary outcome applied to Wilms tumour prognosis. *Applied Statistics* **48**, 457–68.

Examples

```
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

---

pbc   *Mayo Clinic Primary Biliary Cirrhosis Data*

Description

D This data is from the Mayo Clinic trial in primary biliary cirrhosis (PBC) of the liver conducted between 1974 and 1984. A total of 424 PBC patients, referred to Mayo Clinic during that ten-year interval, met eligibility criteria for the randomized placebo controlled trial of the drug D-penicillamine. The first 312 cases in the data set participated in the randomized trial and contain largely complete data. The additional 112 cases did not participate in the clinical trial, but consented to have basic measurements recorded and to be followed for survival. Six of those cases were lost

to follow-up shortly after diagnosis, so the data here are on an additional 106 cases as well as the 312 randomized participants.

A nearly identical data set found in appendix D of Fleming and Harrington; this version has fewer missing values.

Usage

    pbc

Format

age:	in years
albumin:	serum albumin (mg/dl)
alk.phos:	alkaline phosphotase (U/liter)
ascites:	presence of ascites
ast:	aspartate aminotransferase, once called SGOT (U/ml)
bili:	serum bilirunbin (mg/dl)
chol:	serum cholesterol (mg/dl)
copper:	urine copper (ug/day)
edema:	0 no edema, 0.5 untreated or successfully treated
	1 edema despite diuretic therapy
hepato:	presence of hepatomegaly or enlarged liver
id:	case number
platelet:	platelet count
protime:	standardised blood clotting time
sex:	m/f
spiders:	blood vessel malformations in the skin
stage:	histologic stage of disease (needs biopsy)
status:	status at endpoint, 0/1/2 for censored, transplant, dead
time:	number of days between registration and the earlier of death,
	transplantion, or study analysis in July, 1986
trt:	1/2/NA for D-penicillmain, placebo, not randomised
trig:	triglycerides (mg/dl)

Source

T Therneau and P Grambsch (2000), *Modeling Survival Data: Extending the Cox Model*, Springer-Verlag, New York. ISBN: 0-387-98784-3.

---

pbcseq   *Mayo Clinic Primary Biliary Cirrhosis, sequential data*

Description

This data is a continuation of the PBC data set, and contains the follow-up laboratory data for each study patient. An analysis based on the data can be found in Murtagh, et. al.

The primary PBC data set contains only baseline measurements of the laboratory paramters. This data set contains multiple laboratory results, but only on the 312 randomized patients. Some baseline data values in this file differ from the original PBC file, for instance, the data errors in prothrombin time and age which were discovered after the orignal analysis (see Fleming and Harrington, figure 4.6.7).

One "feature" of the data deserves special comment. The last observation before death or liver transplant often has many more missing covariates than other data rows. The original clinical protocol for these patients specified visits at 6 months, 1 year, and annually thereafter. At these protocol visits lab values were obtained for a large pre-specified battery of tests. "Extra" visits, often undertaken because of worsening medical condition, did not necessarily have all this lab work. The missing values are thus potentially informative.

Usage

```
pbc
```

Format

id:	case number
age:	in years
sex:	m/f
trt:	1/2/NA for D-penicillmain, placebo, not randomised
time:	number of days between registration and the earlier of death, transplantion, or study analysis in July, 1986
status:	status at endpoint, 0/1/2 for censored, transplant, dead
day:	number of days between enrollment and this visit date all measurements below refer to this date
albumin:	serum albumin (mg/dl)
alk.phos:	alkaline phosphotase (U/liter)
ascites:	presence of ascites
ast:	aspartate aminotransferase, once called SGOT (U/ml)
bili:	serum bilirunbin (mg/dl)
chol:	serum cholesterol (mg/dl)
copper:	urine copper (ug/day)
edema:	0 no edema, 0.5 untreated or successfully treated 1 edema despite diuretic therapy
hepato:	presence of hepatomegaly or enlarged liver
platelet:	platelet count
protime:	standardised blood clotting time
spiders:	blood vessel malformations in the skin
stage:	histologic stage of disease (needs biopsy)
trig:	triglycerides (mg/dl)

Source

T Therneau and P Grambsch, "Modeling Survival Data: Extending the Cox Model", Springer-Verlag, New York, 2000. ISBN: 0-387-98784-3.

References

Murtaugh PA. Dickson ER. Van Dam GM. Malinchoc M. Grambsch PM. Langworthy AL. Gips CH. "Primary biliary cirrhosis: prediction of short-term survival based on repeated patient visits." Hepatology. 20(1.1):126-34, 1994.

Fleming T and Harrington D., "Counting Processes and Survival Analysis", Wiley, New York, 1991.

---

plot.aareg	*Plot an aareg object.*

Description

Plot the estimated coefficient function(s) from a fit of Aalen's additive regression model.

Usage

```
## S3 method for class 'aareg'
plot(x, se=TRUE, maxtime, type='s', ...)
```

Arguments

x	the result of a call to the aareg function
se	if TRUE, standard error bands are included on the plot
maxtime	upper limit for the x-axis.
type	graphical parameter for the type of line, default is "steps".
...	other graphical parameters such as line type, color, or axis labels.

Side Effects

A plot is produced on the current graphical device.

References

Aalen, O.O. (1989). A linear regression model for the analysis of life times. Statistics in Medicine, 8:907-925.

See Also

aareg

---

plot.cox.zph   *Graphical Test of Proportional Hazards*

Description

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

Usage

```
## 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 graphical arguments passed to the plot function.

Side Effects

a plot is produced on the current graphics device.

See Also

coxph, cox.zph.

Examples

```
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).

Usage

```
## 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", ...)
```

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. This is not treated as a vector; all marks have the same size.
log	a logical value, if TRUE the y axis wll 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.
yscale	a numeric value 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.
xscale	a numeric value used like `yscale` for labels on the x axis. A value of 365.25 will give labels in years instead of the original days.

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. By default, the plot program obeys tradition by having the plot start at (0,0).
>
> If start.time argument is used in survfit, firstx is set to that value.

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.

...

> for future methods

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.

Examples

```
leukemia.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\nfor AML Maintenance Study")
lsurv2 <- survfit(Surv(time, status) ~ x, aml, type='fleming')
plot(lsurv2, lty=2:3, fun="cumhaz",
xlab="Months", ylab="Cumulative Hazard")
```

---

predict.coxph   *Predictions for a Cox model*

Description

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

Usage

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

Arguments

object	the results of a coxph fit.
newdata	Optional new data at which to do predictions. If absent predictions are for the data frame used in the original fit.
type	the type of predicted value. Choices are the linear predictor ("lp"), the risk score exp(lp) ("risk"), the expected number of events given the covariates and follow-up time ("expected"), and the terms of the linear predictor ("terms").
se.fit	if TRUE, pointwise standard errors are produced for the predictions.
na.action	applies only when the newdata argument is present, and defines the missing value action for the new data. The default is to include all observations.
terms	if type="terms", this argument can be used to specify which terms should be included; the default is all.
collapse	optional vector of subject identifiers. If specified, the output will contain one entry per subject rather than one entry per observation.
...	For future methods

Details

The linear predictors for each strata are centered within each strata of the model. Two practical reasons are for numerical stability (avoid large arguments to the exp function) and reproducability (addition of a constant to the a covariate causes no change). The primary underlying reason is statistical: a Cox model only predicts relative risks between pairs of subjects in the same strata, and hence the addition of a constant to the linear predictor, either overall or only within a particular stratum, has no effect on any downstream computations.

Value

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

See Also

predict,coxph,termplot

Examples

```
fit <- coxph(Surv(time, status) ~ age + ph.ecog + strata(inst), lung)
mresid <- lung$status - predict(fit, type='expected') #Martingale resid
predict(fit,type="lp")
predict(fit,type="expected")
predict(fit,type="risk",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

```
## 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), na.action=na.pass, ...)
```

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.
na.action	applies only when the `newdata` argument is present, and defines the missing value action for the new data. The default is to include all observations.
...	for future methods

Value

a vector or matrix of predicted values.

References

Escobar and Meeker (1992). Assessing influence in regression analysis with censored data. *Biometrics,* 48, 507-528.

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)

# Predicted Weibull survival curve for a lung cancer subject with
#   ECOG score of 2
lfit <- survreg(Surv(time, status) ~ ph.ecog, data=lung)
pct <- 1:98/100   # The 100th percentile of predicted survival is at +infinity
ptime <- predict(lfit, newdata=data.frame(ph.ecog=2), type='quantile',
                p=pct, se=TRUE)
matplot(cbind(ptime$fit, ptime$fit + 2*ptime$se.fit,
                        ptime$fit - 2*ptime$se.fit)/30.5, 1-pct,
        xlab="Months", ylab="Survival", type='l', lty=c(1,2,2), col=1)
```

---

print.aareg   *Print an aareg object*

Description

Print out a fit of Aalen's additive regression model

Usage

```
## S3 method for class 'aareg'
print(x, maxtime, test=c("aalen", "nrisk"),scale=1,...)
```

Arguments

x	the result of a call to the aareg function
maxtime	the upper time point to be used in the test for non-zero slope
test	the weighting to be used in the test for non-zero slope. The default weights are based on the variance of each coefficient, as a function of time. The alternative weight is proportional to the number of subjects still at risk at each time point.
scale	scales the coefficients. For some data sets, the coefficients of the Aalen model will be very small (10-4); this simply multiplies the printed values by a constant, say 1e6, to make the printout easier to read.
...	for future methods

Details

The estimated increments in the coefficient estimates can become quite unstable near the end of follow-up, due to the small number of observations still at risk in a data set. Thus, the test for slope will sometimes be more powerful if this last 'tail' is excluded.

Value

the calling argument is returned.

Side Effects

the results of the fit are displayed.

References

Aalen, O.O. (1989). A linear regression model for the analysis of life times. Statistics in Medicine, 8:907-925.

See Also

aareg

```
print.summary.coxph
```
Print method for summary.coxph objects

Description

Produces a printed summary of a fitted coxph model

Usage

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

Arguments

x	the result of a call to `summary.coxph`
digits	significant digits to print
signif.stars	Show stars to highlight small p-values
...	For future methods

```
print.summary.survfit
```
Print Survfit Summary

Description

Prints the result of `summary.survfit`.

Usage

```
## S3 method for class 'summary.survfit'
print(x, digits = max(options() $digits-4, 3), ...)
```

Arguments

x	an object of class `"summary.survfit"`, which is the result of the `summary.survfit` function.
digits	the number of digits to use in printing the numbers.
...	for future methods

Value

x, with the invisible flag set to prevent printing.

Side Effects

   prints the summary created by `summary.survfit`.

See Also

   `options`, `print`, `summary.survfit`.

   `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.rmean=getOption("survfit.print.rmean"),
    rmean = getOption('survfit.rmean'),...)
```

Arguments

x	the result of a call to the `survfit` function.
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.
digits	Number of digits to print
print.rmean, rmean	
	Options for computation and display of the restricted mean.
...	for future results

Details

   The mean and its variance are based on a truncated estimator. That is, if the last observation(s) is not
   a death, then the survival curve estimate does not go to zero and the mean is undefined. There are
   four possible approaches, which are selected by the `rmean` option. The first is to set the upper limit
   to a constant, e.g.,`rmean=365`. In this case the reported mean would be the expected number of
   days, out of the first 365, that would be experienced by each group. This is useful if interest focuses
   on a fixed period. Other options are `"none"` (no estimate), `"common"` and `"individual"`.
   For the last of these, the mean is computed as the area under the curve, over the range from 0 to the
   maximum observed time for that curve. Since the end point is random, values for different curves
   are not comparable and the printed standard errors are an underestimate as they do not take into
   account this random variation. This option is provided mainly for backwards compatability, as this
   estimate was the default (only) one in earlier releases of the code. The `"common"` option uses the
   maximum time for all curves in the object as a common upper limit for the auc calculation.

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.

Value

x, with the invisible flag set to prevent printing. (The default for all print functions in R is to return the object passed to them; print.survfit follows the pattern. If you want to capture these printed results for further processing, see the `table` component of `summary.survfit`.)

Side Effects

he number of observations, the number of events, the median survival with its confidence 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.

References

Miller, Rupert G., Jr. (1981). *Survival Analysis.* New York:Wiley, p 71.

See Also

summary.survfit.

---

pspline *Smoothing splines using a pspline basis*

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. Traditional smoothing splines use one basis per observation, but several authors have pointed out that the final results of the fit are indistinguishable for any number of basis functions greater than about 2-3 times the degrees of freedom. Eilers and Marx point out that if the basis functions are evenly spaced, this leads to significant computational simplifications.

Usage

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

Arguments

x	predictor. The function does not apply to factor variables.
df	the desired degrees of freedom. One of the arguments df or theta' must be given, but not both. If df=0, then the AIC = (loglik -df) is used to choose an "optimal" degrees of freedom. If AIC is chosen, then an optional argument 'caic=T' can be used to specify the corrected AIC of Hurvich et. al.
theta	roughness penalty for the fit. It is a monotone function of the degrees of freedom, with theta=1 corresponding to a linear fit and theta=0 to an unconstrained fit of nterm degrees of freedom.
nterm	number of splines in the basis
degree	degree of splines
eps	accuracy for df
method	the method for choosing the tuning parameter theta. If theta is given, then 'fixed' is assumed. If the degrees of freedom is given, then 'df' is assumed. If method='aic' then the degrees of freedom is chosen automatically using Akaike's information criterion.
...	optional arguments to the control function

Value

Object of class coxph.penalty containing the spline basis, with the appropriate attributes to be recognized as a penalized term by the coxph or survreg functions.

References

Eilers, Paul H. and Marx, Brian D. (1996). Flexible smoothing with B-splines and penalties. Statistical Science, 11, 89-121.

Hurvich, C.M. and Simonoff, J.S. and Tsai, Chih-Ling (1998). Smoothing parameter selection in nonparametric regression using an improved Akaike information criterion, JRSSB, volume 60, 271–293.

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
```

---

pyears   *Person Years*

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

```
pyears(formula, data, weights, subset, na.action,  rmap,
       ratetable, scale=365.25, expect=c('event', 'pyears'),
       model=FALSE, x=FALSE, y=FALSE, data.frame=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 survival 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`.
rmap	an optional list that maps data set names to the ratetable names. See the details section below.
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.
data.frame	return a data frame rather than a set of arrays.
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 ~ rx, rmap=list(age=age, sex=sex, year=entry.dt),
                  ratetable=survexp.us)
```

the natural unit of the ratetable is hazard per day, 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.

The ratetable being used may have different variable names than the user's data set, this is dealt with by the `rmap` argument. The rate table for the above calculation was `survexp.us`, a call to `summary{survexp.us}` reveals that it expects to have variables `age` = age in days, `sex`, and `year` = the date of study entry, we create them in the `rmap` line. The sex variable is not mapped, therefore the code assumes that it exists in `mydata` in the correct format. (Note: for factors such as sex, the program will match on any unique abbreviation, ignoring case.)

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 `pyears` function to correctly track the subject.

The results of `pyears` are normally used as input to further calculations. The `print` routine, therefore, is designed to give only a summary of the table.

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). The dimension and dimmanes of the array correspond to the variables on the right hand side of the model equation.
`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 if `expect ="pyears"`). This will be present only if there was a `ratetable` term.
`data`	if the `data.frame` option was set, a data frame containing the variables `n`, `event`, `pyears` and `event` that supplants the four arrays listed above, along with variables corresponding to each dimension. There will be one row for each cell in the arrays.
`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

```
# Look at progression rates jointly by calendar date and age
#
temp.yr  <- tcut(mgus$dxyr, 55:92, labels=as.character(55:91))
temp.age <- tcut(mgus$age, 34:101, labels=as.character(34:100))
ptime <- ifelse(is.na(mgus$pctime), mgus$futime, mgus$pctime)
pstat <- ifelse(is.na(mgus$pctime), 0, 1)
pfit <- pyears(Surv(ptime/365.25, pstat) ~ temp.yr + temp.age + sex,  mgus,
     data.frame=TRUE)
# Turn the factor back into numerics for regression
tdata <- pfit$data
tdata$age <- as.numeric(as.character(tdata$temp.age))
tdata$year<- as.numeric(as.character(tdata$temp.yr))
fit1 <- glm(event ~ year + age+ sex +offset(log(pyears)),
             data=tdata, family=poisson)
## Not run:
# fit a gam model
gfit.m <- gam(y ~ s(age) + s(year) + offset(log(time)),
                       family = poisson, data = tdata)

## End(Not run)

# Example #2  Create the hearta data frame:
hearta <- by(heart, heart$id,
             function(x)x[x$stop == max(x$stop),])
hearta <- do.call("rbind", hearta)
# Produce pyears table of death rates on the surgical arm
#  The first is by age at randomization, the second by current age
fit1 <- pyears(Surv(stop/365.25, event) ~ cut(age + 48, c(0,50,60,70,100)) +
       surgery, data = hearta, scale = 1)
fit2 <- pyears(Surv(stop/365.25, event) ~ tcut(age + 48, c(0,50,60,70,100)) +
       surgery, data = hearta, scale = 1)
fit1$event/fit1$pyears  #death rates on the surgery and non-surg arm

fit2$event/fit2$pyears  #death rates on the surgery and non-surg arm
```

---

ratetable	*Ratetable reference in formula*

Description

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

Usage

```
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

```
fit <- survfit(Surv(time, status) ~ sex, pbc,subset=1:312)

# The data set does not have entry date, use the midpoint of the study
efit <- survexp(~ ratetable(sex=sex,age=age*365.35,year=as.Date('1979/1/1')) +
            sex, data=pbc,  times=(0:24)*182)
## Not run:
plot(fit, mark.time=F, xscale=365.25, xlab="Years post diagnosis",
        ylab="Survival")
lines(efit, col=2, xscale=365.25) # Add the expected survival line

## End(Not run)
```

---

ratetableDate	*Convert date objects to ratetable form*

Description

This method converts dates from various forms into the internal form used in ratetable objects.

Usage

```
ratetableDate(x)
```

Arguments

x   a date. The function currently has methods for Date, date, POSIXt, timeDate, and chron objects.

Details

This function is useful for those who create new ratetables, but is normally invisible to users. It is used internally by the survexp and pyears functions to map the various date formats; if a new method is added then those routines will automatically be adapted to the new date type.

Value

a numeric vector, the number of days since 1/1/1960.

Author(s)

Terry Therneau

See Also

pyears, survexp

---

ratetables   *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.

azr 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 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.

Examples

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

---

rats   *Rat data from Gail et al.*

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

MH Gail, TJ Santner, and CC Brown (1980), An analysis of comparative carcinogenesis experiments based on multiple times to tumor. *Biometrics* **36**, 255–266.

---

residuals.coxph   *Calculate Residuals for a 'coxph' Fit*

Description

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

Usage

```
## 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

T. Therneau, P. Grambsch, and T. Fleming. "Martingale based residuals for survival models", *Biometrika*, March 1990.

See Also

coxph

Examples

```
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

```
## 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"`, `"lsresp"`, `"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 $X\beta$, 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

Escobar, L. A. and Meeker, W. Q. (1992). Assessing influence in regression analysis with censored data. *Biometrics* **48**, 507-528.

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

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

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

id:	ID number
time:	survival or censoring time
status:	censoring status
age:	in years
t5:	T5 mismatch score

Source

LA Escobar and WQ Meeker Jr (1992), Assessing influence in regression analysis with censored
data. *Biometrics* **48**, 507–528. Page 519.

See Also

predict.survreg, heart

---

strata   *Identify Stratification 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, sep=', ')
```

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
`sep`	the character used to separate groups, in the created label

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)
```

---

summary.aareg   *Summarize an aareg fit*

Description

Creates the overall test statistics for an Aalen additive regression model

Usage

```
## S3 method for class 'aareg'
summary(object, maxtime, test=c("aalen", "nrisk"), scale=1,...)
```

Arguments

object	the result of a call to the `aareg` function
maxtime	truncate the input to the model at time "maxtime"
test	the relative time weights that will be used to compute the test
scale	scales the coefficients. For some data sets, the coefficients of the Aalen model will be very small (10-4); this simply multiplies the printed values by a constant, say 1e6, to make the printout easier to read.
...	for future methods

Details

It is not uncommon for the very right-hand tail of the plot to have large outlying values, particularly for the standard error. The `maxtime` parameter can then be used to truncate the range so as to avoid these. This gives an updated value for the test statistics, without refitting the model.

Value

a list is returned with the following components

table	a matrix with rows for the intercept and each covariate, and columns giving a slope estimate, the test statistic, it's standard error, the z-score and a p-value
test	the time weighting used for computing the test statistics
test.statistic	the vector of test statistics
test.var	the model based variance matrix for the test statistic
test.var2	optionally, a robust variance matrix for the test statistic
chisq	the overall test (ignoring the intercept term) for significance of any variable
n	a vector containing the number of observations, the number of unique death times used in the computation, and the total number of unique death times

See Also

aareg, plot.aareg

Examples

```
afit <- aareg(Surv(time, status) ~ age + sex + ph.ecog, data=lung,
     dfbeta=TRUE)
summary(afit)
## Not run:
            slope   test se(test) robust se    z        p
Intercept 5.05e-03    1.9    1.54      1.55 1.23 0.219000
      age 4.01e-05  108.0  109.00    106.00 1.02 0.307000
      sex -3.16e-03  -19.5    5.90      5.95 -3.28 0.001030
  ph.ecog 3.01e-03   33.2    9.18      9.17 3.62 0.000299

Chisq=22.84 on 3 df, p=4.4e-05; test weights=aalen
```

```
## End(Not run)

summary(afit, maxtime=600)
## Not run:
              slope     test se(test) robust se      z         p
Intercept  4.16e-03     2.13     1.48      1.47  1.450  0.146000
      age  2.82e-05    85.80   106.00    100.00  0.857  0.392000
      sex -2.54e-03   -20.60     5.61      5.63 -3.660  0.000256
  ph.ecog  2.47e-03    31.60     8.91      8.67  3.640  0.000271

Chisq=27.08 on 3 df, p=5.7e-06; test weights=aalen

## End(Not run)
```

---

summary.coxph   *Summary method for Cox models*

Description

Produces a summary of a fitted coxph model

Usage

```
## S3 method for class 'coxph'
summary(object, conf.int=0.95, scale=1,...)
```

Arguments

object	the result of a coxph fit
conf.int	level for computation of the confidence intervals. If set to FALSE no confidence intervals are printed
scale	vector of scale factors for the coefficients, defaults to 1. The confidence limits are for the risk change associated with one scale unit.
...	for future methods

Value

An object of class `summary.coxph`.

See Also

coxph, print.coxph

Examples

```
fit <- coxph(Surv(time, status) ~ age + sex, lung)
summary(fit)
## Not run:
Call:
coxph(formula = Surv(time, status) ~ age + sex, data = lung)

  n= 228

      coef exp(coef)  se(coef)     z      p
age  0.017     1.017   0.00922  1.85 0.0650
sex -0.513     0.599   0.16745 -3.06 0.0022

    exp(coef) exp(-coef) lower .95 upper .95
age     1.017      0.983     0.999     1.036
sex     0.599      1.670     0.431     0.831

Rsquare= 0.06   (max possible= 0.999 )
Likelihood ratio test= 14.1  on 2 df,   p=0.000857
Wald test            = 13.5  on 2 df,   p=0.00119
Score (logrank) test = 13.7  on 2 df,   p=0.00105

## End(Not run)
```

`summary.survfit`	*Summary of a Survival Curve*

Description

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

Usage

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

Arguments

object	the result of a call to the `survfit` function.
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 value: should the censoring times be included in the output? This is ignored if the `times` argument is present.

scale	numeric value to rescale the survival time, e.g., if the input data to `survfit` were in days, `scale = 365.25` would scale the output to years.
extend	logical value: if TRUE, prints information for all specified `times`, even if there are no subjects left at the end of the specified `times`. This is only valid if the `times` argument is present.
rmean	Show restricted mean: see `print.summary.survfit` for details
...	for future methods

Value

a list with the following components:

surv	the estimate of survival at time t+0.
time	the timepoints on the curve.
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.
n.entered	This is present only for counting process survival data. If the `times` argument is missing, this column is the number of subjects that entered at time t. Otherwise, it is the cumulative number of subjects that have entered since the last time listed until time t.
n.exit.censored	
	if the `times` argument is missing, this column is the number of subjects that left without an event at time t. Otherwise, it is the cumulative number of subjects that have left without an event since the last time listed until time t+0. This is only present for counting process survival data.
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	same as for `fit`, if present.
table	table of information that is returned from `print.survfit` function.
type	type of data censoring. Passed through from the `fit` object.

See Also

`survfit`, `print.summary.survfit`

Examples

```
summary( survfit( Surv(futime, fustat)~1, data=ovarian))
summary( survfit( Surv(futime, fustat)~rx, data=ovarian))
```

Surv	*Create a Survival Object*

Description

Create a survival object, usually used as a response variable in a model formula. Argument matching is special for this function, see Details below.

Usage

```
Surv(time, time2, event,
    type=c('right', 'left', 'interval', 'counting', 'interval2'),
    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.
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 option was intended to be 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, but it has rarely proven useful.
x	any R object.

Details

Typical usages are

```
Surv(time, event)
Surv(time, time2, event, type= )
```

The `time`,`time2` and `event` arguments are matched by position, not by name, so use, eg, `Surv(time, dead)` rather than `Surv(time, event=dead)`

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 interval is a pair of time points within which the event is known to have occurred.

If `type="interval2"` then the representation given above is assumed, with `NA` taking the place of infinity. If `type="interval"` then `event` must be given. If `event` is 0, 1, or 2, the relevant information is assumed to be contained in `time`, the value in `time2` is ignored, and the second column of the internal representation contains a placeholder value.

Presently, the only methods allowing interval censored data are the parametric models computed by `survreg` and survival curves computed by `survfit`; for both of these, 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 `if (max(status)==2)`. If 1/2 coding is used and all the subjects are censored, it will guess wrong. In any questionable case it is safer to use logical coding, e.g., `Surv(time, status==3)` would indicate that a 3 is the code for an event.

Surv objects can be subscripted either as an object, e.g. `x[1:3]` using a single subscript; in which case the `drop` argument is ignored; or as a matrix, using two arguments. If the second subscript is missing and `drop=F` (the default), the result of the subscripting will be a Surv object, e.g., `x[1:3,,drop=F]`, otherwise the result will be a matrix (or vector), in accordance with the default behavior for subscripting matrices.

Value

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

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

See Also

`coxph`, `survfit`, `survreg`.

Examples

```
with(lung, Surv(time, status))
Surv(heart$start, heart$stop, heart$event)
```

---

survConcordance   *Compute a concordance measure.*

Description

This function computes the concordance between a right-censored survival time and a single continuous covariate

Usage

```
survConcordance(formula, data, weights, subset, na.action)
```

Arguments

formula   a formula with a survival time on the left and a single covariate on the right, along with an optional strata() term.

data   a data frame

weights,subset,na.action
                as for coxph

Details

Concordance is defined as Pr(agreement) for any two randomly chosen observations, where in this case agreement means that the observation with the shorter survival time of the two also has the larger risk score. The predictor (or risk score) will often be the result of a Cox model or other regression.

For continuous covariates concordance is equivalent to Kendall's tau, and for logistic regression is is equivalent to the area under the ROC curve. A value of 1 signifies perfect agreement, .6-.7 is a common result for survival data, .5 is an agreement that is no better than chance, and .3-.4 is the performace of some stock market analysts.

The computation involves all n(n-1)/2 pairs of data points in the sample. For survival data, however, some of the pairs are incomparable. For instance a pair of times (5+, 8), the first being a censored value. We do not know whether the first survival time is greater than or less than the second. Among observations that are comparable, pairs may also be tied on survival time (but only if both are uncensored) or on the predictor. The final concordance is (agree + tied/2)/(agree + disagree + tied).

There is, unfortunately, one aspect of the formula above that is unclear. Should the count of ties include observations that are tied on survival time y, tied on the predictor x, or both? By default the concordance only counts ties in x, treating tied survival times as incomparable; this agrees with the AUC calculation used in logistic regression. The Goodman-Kruskal Gamma statistic is (agree-disagree)/(agree + disagree), ignoring ties. It ranges from -1 to +1 similar to a correlation coefficient. Kendall's tau uses ties of both types. All of the components are returned in the result, however, so people can compute other combinations if interested. (If two observations have the same survival and the same x, they are counted in the tied survival time category).

Value

an object containing the concordance, followed by the number of pairs that agree, disagree, are tied, and are not comparable.

See Also

summary.coxph

Examples

```
survConcordance(Surv(time, status) ~age, data=lung)

options(na.action=na.exclude)
fit <- coxph(Surv(time, status) ~ ph.ecog + age + sex, lung)
survConcordance(Surv(time, status) ~predict(fit), lung)
## Not run:
  n=227 (1 observations deleted due to missing values)
Concordance= 0.6371102 , Gamma= 0.2759638
concordant discordant  tied risk  tied time
    12544        7117         126          28

## End(Not run)
```

---

survdiff   *Test Survival Curve Differences*

Description

Tests if there is a difference between two or more survival curves using the G^ρ 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

Harrington, D. P. and Fleming, T. R. (1982). A class of rank test procedures for censored survival 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, rmap, 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), and is often ~1, i.e., expected survival for the entire group.
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.
rmap	an optional list that maps data set names to the ratetable names. See the details section below.
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.
se.fit	compute the standard error of the predicted survival. The default is to compute standard errors whenever possible, 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.us` population tables contain expected death rates based on calendar year, sex and age. Then

```
haz <- -log(survexp(fu.time ~ 1, data=mydata,
                    rmap = list(year=entry.dt, age=(birth.dt-entry.dt)),
                    cohort=FALSE))
```

gives for each subject the total hazard experienced up to their observed death time or last follow-up time (variable fu.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.

The ratetable being used may have different variable names than the user's data set, this is dealt with by the `rmap` argument. The rate table for the above calculation was `survexp.us`, a call to `summary{survexp.us}` reveals that it expects to have variables age = age in days, sex, and year = the date of study entry, we create them in the `rmap` line. The sex variable is not mapped, therefore the code assumes that it exists in `mydata` in the correct format. (Note: for factors such as sex, the program will match on any unique abbreviation, ignoring case.)

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=TRUE` 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

Berry, G. (1983). The analysis of mortality by the subject-years method. *Biometrics*, 39:173-84.

Ederer, F., Axtell, L. and Cutler, S. (1961). The relative survival rate: a statistical methodology. *Natl Cancer Inst Monogr*, 6:101-21.

Hakulinen, T. (1982). Cancer survival corrected for heterogeneity in patient withdrawal. *Biometrics*, 38:933-942.

Verheul, H., Dekker, E., Bossuyt, P., Moulijn, A. and Dunning, A. (1993). Background mortality in clinical survival studies. *Lancet*, 341: 872-875.

See Also

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

Examples

```
#
# Stanford heart transplant data
#  We don't have sex in the data set, but know it to be nearly all males.
# Estimate of conditional survival
survexp(futime ~ 1, rmap=list(sex="male", year=accept.dt,
          age=(accept.dt-birth.dt)), conditional=TRUE, data=jasa)
# Estimate of expected  survival stratified by prior surgery
survexp(futime ~ surgery, rmap= list(sex="male", year=accept.dt,
age=(accept.dt-birth.dt)), conditional=TRUE, data=jasa)

## Compare the survival curves for the Mayo PBC data to Cox model fit
##
pfit <-coxph(Surv(time,status>0) ~ trt + log(bili) + log(protime) + age +
                platelet, data=pbc)
plot(survfit(Surv(time, status>0) ~ trt, data=pbc))
lines(survexp( ~ trt, ratetable=pfit, data=pbc), col='purple')
```

---

survexp.fit	*Compute Expected Survival*

Description

Compute expected survival times.

Usage

```
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	a logical value, if `TRUE` the conditional survival is computed, if `FALSE` the cohort survival is computed. See `survexp` for more details.
ratetable	a rate table, such as `survexp.uswhite`.

Details

For conditional survival `y` must be the time of last follow-up or death for each subject. 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 `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`.

---

survfit	*Create survival curves*

Description

This function creates survival curves from either a formula (e.g. the Kaplan-Meier), a previously fitted Cox model, or a previously fitted accelerated failure time model.

Usage

```
surv fit(formula, ...)
```

Arguments

`formula`	either a formula or a previously fitted model
`...`	other arguments to the specific method

Details

For details see the documentation for the appropriate method, i.e., `?survfit.formula` or `?survfit.coxph`.

Value

An object of class `survfit` containing one or more survival curves.

Note

Older releases of the code also allowed the specification for a single curve to omit the right hand of the formula, i.e., `~ 1`. Handling this case required some non-standard and fairly fragile manipulations, and this case is no longer supported.

Author(s)

Terry Therneau

See Also

`survfit.formula`, `survfit.coxph`, `survfit.object`, `print.survfit`, `plot.survfit`, `summary.survfit`

`survfit.coxph`	*Compute a Survival Curve from a Cox model*

Description

Computes the predicted survivor function for a Cox proportional hazards model.

Usage

```
## S3 method for class 'coxph'
survfit(formula, newdata,
        se.fit=TRUE, conf.int=.95,
        individual=FALSE,
        type,vartype,
        conf.type=c("log","log-log","plain","none"), censor=TRUE, id, ...)
```

Arguments

`formula`	A `coxph` object.
`newdata`	a data frame with the same variable names as those that appear in the `coxph` formula. It is also valid to use a vector, if the data frame would consist of a single row.
	The curve(s) produced will be representative of a cohort whose covariates correspond to the values in `newdata`. Default is the mean of the covariates used in the `coxph` fit.
`individual`	a logical value indicating whether each row of `newdata` represents a distinct individual (FALSE, the default), or if each row of the data frame represents different time epochs for only one individual (TRUE). In the former case the result will have one curve for each row in `newdata`, in the latter only a single curve will be produced.
`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 `TRUE`.
`type,vartype`	a character string specifying the type of survival curve. Possible values are `"aalen"`, `"efron"`, or `"kalbfleish-prentice"` (only the first two characters are necessary). The default is to match the computation used in the Cox model. The Nelson-Aalen-Breslow estimate for `ties='breslow'`, the Efron estimate for `ties='efron'` and the Kalbfleisch-Prentice estimate for a discrete time model `ties='exact'`. Variance estimates are the Aalen-Link-Tsiatis, Efron, and Greenwood. The default will be the Efron estimate for `ties='efron'` and the Aalen estimate otherwise.
`conf.type`	One of `"none"`, `"plain"`, `"log"` (the default), or `"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 `curve +- k *se(curve)`, where k is determined from `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)).
`censor`	if FALSE time points at which there are no events (only censoring) are not included in the result.
`id`	optional vector of subject identifiers, see below.
`...`	for future methods

Details

Serious thought has been given to removing the 'default' for `newdata`, since the resulting curve(s) almost never make sense. It remains due to the unwarranted attachment to the option shown by other packages and by users. Two particularly egregious examples are factor variables and interactions. Suppose one were studying interspecies transmission of a virus, and the data set has a factor variable with levels ("pig", "chicken") and about equal numbers of observations for each. The "mean" covariate level will be 1/2 – is this a flying pig? As to interactions assume data with sex coded as 0/1, ages ranging from 50 to 80, and a model with age*sex. The "mean" value for the age:sex interaction term will be about 30, a value that does not occur in the data.

Users are strongly advised to use the newdata argument. Note that this data set needs to contain values for the main effects but not for any interaction terms.

When the original model contained time-dependent covariates, then the path of that covariate through time needs to be specified in order to obtain a predicted curve. This requires `newdata` to contain multiple lines for each hypothetical subject which gives the covariate values, time interval, and strata for each line (a subject can change strata). If `newdata` consists of a single patient then the `individual=TRUE` argument can be used to signal that all the lines are for one subject. If there are multiple prediction subjects an `id` must be added which demarks which rows go to each. The time interval must have the same (start, stop, status) variables as the original model: although the status variable is not used and thus can be set to a dummy value of 0 or 1, it is necessary for the variables to be recognized as a `Surv` object. Last, although predictions with a time-dependent covariate path can be useful, it is very easy to create a prediction that is senseless. Users are encouraged to seek out a text that discusses the issue in detail.

When all the coefficients are zero, the Kalbfleisch-Prentice estimator reduces to the Kaplan-Meier, the Aalen estimate to the exponential of Nelson's cumulative hazard estimate, and the Efron estimate to the Fleming-Harrington estimate of survival. The variances of the curves from a Cox model are larger, however, due to the variance of the coefficients.

See [survfit](#) for more details about the counts (number of events, number at risk, etc.)

The censor argument was fixed at FALSE in earlier versions of the code and not made available to the user. The default argument is sensible in most instances — and causes the familiar + sign to appear on plots — it is not sensible for time dependent covariates since they lead to a large number of spurious marks.

Value

an object of class `"survfit"`. See `survfit.object` for details. Methods defined for survfit objects are `print`, `plot`, `lines`, and `points`.

References

Fleming, T. H. and Harrington, D. P. (1984). Nonparametric estimation of the survival distribution in censored data. *Comm. in Statistics* **13**, 2469-86.

Kablfleisch, J. D. and Prentice, R. L. (1980). *The Statistical Analysis of Failure Time Data.* New York:Wiley.

Link, C. L. (1984). Confidence intervals for the survival function using Cox's proportional hazards model with covariates. *Biometrics* **40**, 601-610.

Therneau T and Grambsch P (2000), Modeling Survival Data: Extending the Cox Model, Springer-Verlag.

Tsiatis, A. (1981). A large sample study of the estimate for the integrated hazard function in Cox's regression model for survival data. *Annals of Statistics* **9**, 93-108.

See Also

[print](#), [plot](#), [lines](#), [coxph](#), [Surv](#), [strata](#).

Examples

```
#fit a Kaplan-Meier and plot it
fit <- survfit(Surv(time, status) ~ x, data = aml)
plot(fit, lty = 2:3)
legend(100, .8, c("Maintained", "Nonmaintained"), lty = 2:3)

#fit a Cox proportional hazards model and plot the
#predicted survival for a 60 year old
fit <- coxph(Surv(futime, fustat) ~ age, data = ovarian)
plot(survfit(fit, newdata=data.frame(age=60)),
     xscale=365.25, xlab = "Years", ylab="Survival")

# Here is the data set from Turnbull
#  There are no interval censored subjects, only left-censored (status=3),
#  right-censored (status 0) and observed events (status 1)
#
#                           Time
#                       1    2    3    4
# Type of observation
#           death      12    6    2    3
#          losses       3    2    0    3
#      late entry       2    4    2    5
#
tdata <- data.frame(time  =c(1,1,1,2,2,2,3,3,3,4,4,4),
                    status=rep(c(1,0,2),4),
                    n     =c(12,3,2,6,2,4,2,0,2,3,3,5))
fit  <- survfit(Surv(time, time, status, type='interval') ~1,
               data=tdata, weight=n)

#
# Time to progression/death for patients with monoclonal gammopathy
#  Competing risk curves (cumulative incidence)
fit1 <- survfit(Surv(stop, event=='progression') ~1, data=mgus1,
                    subset=(start==0))
fit2 <- survfit(Surv(stop, status) ~1, data=mgus1,
                    subset=(start==0), etype=event) #competing risks
# CI curves are always plotted from 0 upwards, rather than 1 down
plot(fit2, fun='event', xscale=365.25, xmax=7300, mark.time=FALSE,
            col=2:3, xlab="Years post diagnosis of MGUS")
lines(fit1, fun='event', xscale=365.25, xmax=7300, mark.time=FALSE,
            conf.int=FALSE)
text(10, .4, "Competing Risk: death", col=3)
text(16, .15,"Competing Risk: progression", col=2)
text(15, .30,"KM:prog")
```

---

survfit.formula   *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. For competing risks data it computes the cumulative incidence curve.

Usage

```
## S3 method for class 'formula'
survfit(formula, data, weights, subset, na.action,
        etype, id, ...)
```

Arguments

formula   a formula object, which must have a `Surv` object as the response on the left of
               the ~ operator and, if desired, terms separated by + operators on the right. One
               of the terms may be a `strata` object. For a single survival curve the right hand
               side should be ~ 1.

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

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
               `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 `subset`
               argument has been used. Default is `options()$na.action`.

etype   a variable giving the type of event. Presence of this variable signals the program
               to compute the cumulative incidece estimate. For each event `status==1`, the
               etype variable indicates the type of event. For a censored observation the value
               of `etype` is ignored - but do not set it to NA, since that will cause na.action to
               delete the observation.

id   identifies individual subjects, when a given person can have multiple lines of
               data. when used with the `etype` variable, this allows the compuation of a
               cumulative prevalence estimate, i.e., the incidence over time.

...   The following additional arguments are passed to internal functions called by
               `survfit`.

               type a character string specifying the type of survival curve. Possible values are
                   `"kaplan-meier"`, `"fleming-harrington"` or `"fh2"` if a formula
                   is given. This is ignored for competing risks or when the Turnbull estimator
                   is used.

               error a character string specifying the error. Possible values are `"greenwood"`
                   for the Greenwood formula or `"tsiatis"` for the Tsiatis formula, (only
                   the first character is necessary).

               conf.type One of `"none"`, `"plain"`, `"log"` (the default), or `"log-log"`.
                   Only enough of the string to uniquely identify it is necessary. The first op-
                   tion causes confidence intervals not to be generated. The second causes the
                   standard intervals `curve +- k *se(curve)`, where k is determined

from `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)).

conf.lower a character string to specify modified lower limits to the curve, the upper limit remains unchanged. Possible values are `"usual"` (unmodified), `"peto"`, and `"modified"`. T he 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 modified limit, but also replaces the usual Greenwood variance term with a simple approximation. It is known to be conservative.

start.time numeric value specifying a time to start calculating survival information. The resulting curve is the survival conditional on surviving to `start.time`.

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 `TRUE`.

Details

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.

The Greenwood formula for the variance is a sum of terms $d/(n*(n-m))$, 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 $d/(n*n)$, 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 increments the hazard by 3/10 and the second by 1/10 + 1/9 + 1/8. For the first method $S(t) = \exp(H)$, where H is the Nelson-Aalen cumulative hazard estimate, whereas the `fh2` method will give results $S(t)$ results closer to the Kaplan-Meier.

When the data set includes left censored or interval censored data (or both), then the EM approach of Turnbull is used to compute the overall curve. When the baseline method is the Kaplan-Meier, this is known to converge to the maximum likelihood estimate.

The cumulative incidence curve is an alternative to the Kaplan-Meier for competing risks data. For instance, in patients with MGUS, conversion to an overt plasma cell malignancy occurs at a nearly constant rate among those still alive. A Kaplan-Meier estimate, treating death due to other causes as censored, gives a 20 year cumulate rate of 33% for the 241 early patients of Kyle. This estimates the incidence of conversion, if other causes of death were removed.

The CI estimate, on the other hand, estimates the total number of conversions that will actually occur. Because the population is older, this is much smaller than the KM, 22% at 20 years for Kyle's data. If there were no censoring, then CI(t) could very simply be computed as total number of patients with progression by time t divided by the sample size n.

Value

an object of class `"survfit"`. See `survfit.object` for details. Methods defined for survfit objects are `print`, `plot`, `lines`, and `points`.

References

Dorey, F. J. and Korn, E. L. (1987). Effective sample sizes for confidence intervals for survival probabilities. *Statistics in Medicine* **6**, 679-87.

Fleming, T. H. and Harrington, D. P. (1984). Nonparametric estimation of the survival distribution in censored data. *Comm. in Statistics* **13**, 2469-86.

Kablfleisch, J. D. and Prentice, R. L. (1980). *The Statistical Analysis of Failure Time Data.* New York:Wiley.

Kyle, R. A. (1997). Moncolonal gammopathy of undetermined significance and solitary plasmacytoma. Implications for progression to overt multiple myeloma}, *Hematology/Oncology Clinics N. Amer.* **11**, 71-87.

Link, C. L. (1984). Confidence intervals for the survival function using Cox's proportional hazards model with covariates. *Biometrics* **40**, 601-610.

Turnbull, B. W. (1974). Nonparametric estimation of a survivorship function with doubly censored data. *J Am Stat Assoc*, **69**, 169-173.

See Also

`survfit.coxph` for survival curves from Cox models.

`print`, `plot`, `lines`, `coxph`, `Surv`, `strata`.

Examples

```
#fit a Kaplan-Meier and plot it
fit <- survfit(Surv(time, status) ~ x, data = aml)
plot(fit, lty = 2:3)
legend(100, .8, c("Maintained", "Nonmaintained"), lty = 2:3)

#fit a Cox proportional hazards model and plot the
#predicted survival for a 60 year old
fit <- coxph(Surv(futime, fustat) ~ age, data = ovarian)
plot(survfit(fit, newdata=data.frame(age=60)),
     xscale=365.25, xlab = "Years", ylab="Survival")

# Here is the data set from Turnbull
#  There are no interval censored subjects, only left-censored (status=3),
#  right-censored (status 0) and observed events (status 1)
#
#                                Time
```

```
#                            1    2    3    4
# Type of observation
#           death        12    6    2    3
#           losses        3    2    0    3
#       late entry        2    4    2    5
#
tdata <- data.frame(time  =c(1,1,1,2,2,2,3,3,3,4,4,4),
                    status=rep(c(1,0,2),4),
                    n     =c(12,3,2,6,2,4,2,0,2,3,3,5))
fit  <- survfit(Surv(time, time, status, type='interval') ~1,
              data=tdata, weight=n)


#
# Time to progression/death for patients with monoclonal gammopathy
#  Competing risk curves (cumulative incidence)
fit1 <- survfit(Surv(stop, event=='progression') ~1, data=mgus1,
                  subset=(start==0))
fit2 <- survfit(Surv(stop, status) ~1, data=mgus1,
                  subset=(start==0), etype=event) #competing risks
# CI curves are always plotted from 0 upwards, rather than 1 down
plot(fit2, fun='event', xscale=365.25, xmax=7300, mark.time=FALSE,
           col=2:3, xlab="Years post diagnosis of MGUS")
lines(fit1, fun='event', xscale=365.25, xmax=7300, mark.time=FALSE,
           conf.int=FALSE)
text(10, .4, "Competing Risk: death", col=3)
text(16, .15,"Competing Risk: progression", col=2)
text(15, .30,"KM:prog")
```

---

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. Class of objects that represent a fitted survival curve.

Arguments

n   total number of subjects in each curve.

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.

n.enter   for counting process data only, the number of subjects that enter at time t.

n.censor	for counting process data only, the number of subjects who exit the risk set, without an event, at time t. (For right censored data, this number can be computed from the successive values of the number at risk).
surv	the estimate of survival at time t+0. This may be a vector or a matrix.
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.
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.
start.time	the value specified for the start.time argument, if it was used in the call.
n.all	for counting process data, and any time that the start.time argument was used, this contains the total number of observations that were available. Not all may have been used in creating the curve, in which case this value will be larger than n above. of observations that were available
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	an image of the call that produced the object.
type	type of survival censoring.

Structure

The following components must be included in a legitimate survfit 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

plot.survfit, summary.survfit, print.survfit, survfit.

---

survfitcoxph.fit	*A direct interface to the 'computational engine' of survfit.coxph*

Description

This program is mainly supplied to allow other packages to invoke the survfit.coxph function at a 'data' level rather than a 'user' level. It does no checks on the input data that is provided, which can lead to unexpected errors if that data is wrong.

Usage

```
survfitcoxph.fit(y, x, wt, x2, risk, newrisk, strata, se.fit, survtype,
vartype, varmat, id, y2, strata2, unlist=TRUE)
```

Arguments

y	the response variable used in the Cox model. (Missing values removed of course.)
x	covariate matrix used in the Cox model
wt	weight vector for the Cox model. If the model was unweighted use a vector of 1s.
x2	matrix describing the hypothetical subjects for which a curve is desired. Must have the same number of columns as x.
risk	the risk score exp(X beta) from the fitted Cox model. If the model had an offset, include it in the argument to exp.
newrisk	risk scores for the hypothetical subjects
strata	strata variable used in the Cox model. This will be a factor.
se.fit	if TRUE the standard errors of the curve(s) are returned
survtype	1=Kalbfleish-Prentice, 2=Nelson-Aalen, 3=Efron. It is usual to match this to the approximation for ties used in the coxph model: KP for 'exact', N-A for 'breslow' and Efron for 'efron'.
vartype	1=Greenwood, 2=Aalen, 3=Efron
varmat	the variance matrix of the coefficients
id	optional; if present and not NULL this should be a vector of identifiers of length nrow(x2). A mon-null value signifies that x2 contains time dependent covariates, in which case this identifies which rows of x2 go with each subject.
y2	survival times, for time dependent prediction. It gives the time range (time1,time2] for each row of x2. Note: this must be a Surv object and thus contains a status indicator, which is never used in the routine, however.
strata2	vector of strata indicators for x2. This must be a factor.
unlist	if FALSE the result will be a list with one element for each strata. Otherwise the strata are "unpacked" into the form found in a survfit object.

Value

a list containing nearly all the components of a `survfit` object. All that is missing is to add the confidence intervals, the type of the original model's response (as in a coxph object), and the class.

Note

The source code for for both this function and `survfit.coxph` is written using noweb. For complete documentation see the `inst/sourcecode.pdf` file.

Author(s)

Terry Therneau

See Also

[survfit.coxph](#)

| 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 that 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

O'Brien, Peter, "A Nonparametric Test for Association with Censored Data", *Biometrics* 34: 243-250, 1978.

See Also

survdiff

Examples

```
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

Fit a parametric survival regression model. These are location-scale models for an arbitrary transform of the time variable; the most common cases use a log transformation, leading to accelerated failure time models.

Usage

```
survreg(formula, data, weights, subset,
        na.action, dist="weibull", init=NULL, scale=0,
        control,parms=NULL,model=FALSE, x=FALSE,
        y=TRUE, robust=FALSE, score=FALSE, ...)
```

Arguments

formula   a formula expression as for other regression models. The response is usually
            a survival object as returned by the Surv function. See the documentation for
            Surv, lm and formula for details.

data	a data frame in which to interpret the variables named in the `formula`, `weights` or the `subset` arguments.
weights	optional vector of case weights
subset	subset of the observations to 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`.
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"`. 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`. The default value is `survreg.control()`
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.
score	return the score vector. (This is expected to be zero upon successful convergence.)
robust	Use robust 'sandwich' standard errors, based on independence of individuals if there is no `cluster()` term in the formula, based on independence of clusters if there is.
...	other arguments which will be passed to `survreg.control`.

Value

an object of class `survreg` is returned.

See Also

survreg.object, survreg.distributions, pspline, frailty, ridge

Examples

```
# Fit an exponential model: the two fits are the same
survreg(Surv(futime, fustat) ~ ecog.ps + rx, ovarian, dist='weibull',
                               scale=1)
survreg(Surv(futime, fustat) ~ ecog.ps + rx, ovarian,
       dist="exponential")


#
# A model with different baseline survival shapes for two groups, i.e.,
#    two different scales
```

```
survreg(Surv(time, status) ~ ph.ecog + age + strata(sex), lung)

# There are multiple ways to parameterize a Weibull distribution. The survreg
# function imbeds it in a general location-scale familiy, which is a
# different parameterization than the rweibull function, and often leads
# to confusion.
#   survreg's scale  =    1/(rweibull shape)
#   survreg's intercept = log(rweibull scale)
#   For the log-likelihood all parameterizations lead to the same value.
y <- rweibull(1000, shape=2, scale=5)
survreg(Surv(y)~1, dist="weibull")

# Economists fit a model called `tobit regression', which is a standard
# linear regression with Gaussian errors, and left censored data.
tobinfit <- survreg(Surv(durable, durable>0, type='left') ~ age + quant,
            data=tobin, dist='gaussian')
```

---

survreg.control   *Package options for survreg and coxph*

Description

This functions checks and packages the fitting options for survreg

Usage

```
survreg.control(maxiter=30, rel.tolerance=1e-09,
toler.chol=1e-10, iter.max, debug=0, outer.max=10)
```

Arguments

maxiter   maximum number of iterations

rel.tolerance
               relative tolerance to declare convergence

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

Value

A list with the same elements as the input

See Also

survreg

```
survreg.distributions
```
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 two basic formats, the first defines a distribution de novo, the second defines a new distribution in terms of an old one.

name:	name of distribution
variance:	function(parms) returning the variance (currently unused)
init(x,weights,...):	Function returning an initial
	estimate of the mean and variance
	(used for initial values in the iteration)
density(x,parms):	Function returning a matrix with columns F,
	$1 - F, f, f'/f, f''/f$
quantile(p,parms):	Quantile function
scale:	Optional fixed value for the scale parameter
parms:	Vector of default values and names for any additional parameters
deviance(y,scale,parms):	Function returning the deviance for a
	saturated model; used only for deviance residuals.

and to define one distribution in terms of another

name:	name of distribution
dist:	name of parent distribution
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 location-scale parameterizaion of a Weibull
distribution found in survreg is not the same as the parameterization of 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.

For speed parts of the three most commonly used distributions are hardcoded in C; for this reason
the elements of survreg.distributions with names of "Extreme value", "Logisitic" and
"Gaussian" should not be modified. (The order of these in the list is not important, recognition is
by name.) As an alternative to modifying survreg.distributions a new distribution can be
specified as a separate list. This is the preferred method of addition and is illustrated below.

See Also

survreg, pweibull, pnorm,plogis, pt, survregDtest

Examples

```
# time transformation
survreg(Surv(time, status) ~ ph.ecog + sex, dist='weibull', data=lung)
# change the transformation to work in years
# intercept changes by log(365), everything else stays the same
my.weibull <- survreg.distributions$weibull
my.weibull$trans <- function(y) log(y/365)
my.weibull$itrans <- function(y) 365*exp(y)
survreg(Surv(time, status) ~ ph.ecog + sex, lung, dist=my.weibull)

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

# Cauchy fit
mycauchy <- list(name='Cauchy',
                 init= function(x, weights, ...)
                       c(median(x), mad(x)),
                 density= function(x, parms) {
                     temp <- 1/(1 + x^2)
                     cbind(.5 + atan(temp)/pi, .5+ atan(-temp)/pi,
                           temp/pi, -2 *x*temp, 2*temp^2*(4*x^2*temp -1))
                     },
                 quantile= function(p, parms) tan((p-.5)*pi),
                 deviance= function(...) stop('deviance residuals not defined')
                 )
survreg(Surv(log(time), status) ~ ph.ecog + sex, lung, dist=mycauchy)
```

| survreg.object | *Parametric Survival Model Object* |

Description

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

COMPONENTS

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

coefficients the coefficients of the `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 mulitple 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): `linear predictors`, `weights`, `x`, `y`, `model`, `call`, `terms` and `formula`. See `lm`.

See Also

`survreg`, `lm`

---

survregDtest	*Verify a survreg distribution*

Description

This routine is called by `survreg` to verify that a distribution object is valid.

Usage

```
survregDtest(dlist, verbose = F)
```

Arguments

dlist	the list describing a survival distribution
verbose	return a simple TRUE/FALSE from the test for validity (the default), or a verbose description of any flaws.

Details

If the `survreg` function rejects your user-supplied distribution as invalid, this routine will tell you why it did so.

Value

TRUE if the distribution object passes the tests, and either FALSE or a vector of character strings if not.

Author(s)

Terry Therneau

See Also

[survreg.distributions](), [survreg]()

Examples

```
# An invalid distribution (it should have "init =" on line 2)
#   surveg would give an error message
mycauchy <- list(name='Cauchy',
                 init<- function(x, weights, ...)
                     c(median(x), mad(x)),
                 density= function(x, parms) {
                     temp <- 1/(1 + x^2)
                     cbind(.5 + atan(temp)/pi, .5+ atan(-temp)/pi,
                           temp/pi, -2 *x*temp, 2*temp^2*(4*x^2*temp -1))
                     },
                 quantile= function(p, parms) tan((p-.5)*pi),
                 deviance= function(...) stop('deviance residuals not defined')
```

```
                )

    survregDtest(mycauchy, TRUE)
```

---

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
end	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.

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)
```

---

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

```
mdy.date <- function(m,d,y)
    as.Date(paste(ifelse(y<100, y+1900, y), m, d, sep='/'))
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
py1
```

```
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)

Source

J Tobin (1958), Estimation of relationships for limited dependent variables. *Econometrica* **26**, 24–36.

Examples

```
tfit <- survreg(Surv(durable, durable>0, type='left') ~age + quant,
                data=tobin, dist='gaussian')

predict(tfit,type="response")
```

---

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

```
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=adeno, 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

D Kalbfleisch and RL Prentice (1980), *The Statistical Analysis of Failure Time Data*. Wiley, New York.

Index