Package ‘coxphw’

November 28, 2023

Type Package
Title Weighted Estimation in Cox Regression
Version 4.0.3
Date 2023-10-31
Description Implements weighted estimation in Cox regression as proposed by
Schemper, Wakounig and Heinze (Statistics in Medicine, 2009,
<doi:10.1002/sim.3623>) and as described in Dunkler, Ploner, Schemper and
Weighted Cox regression provides unbiased average hazard ratio
estimates also in case of non-proportional hazards.
Approximated generalized concordance probability an effect size measure for clear-cut
decisions can be obtained.
The package provides options to estimate time-dependent effects conveniently by
including interactions of covariates with arbitrary functions of time, with or without
making use of the weighting option.
Depends R (>= 3.0.2), survival
Suggests knitr, rmarkdown, testthat
NeedsCompilation yes
License GPL-3
VignetteBuilder knitr
RoxygenNote 7.2.3
Encoding UTF-8
URL https://github.com/biometrician/coxphw
BugReports https://github.com/biometrician/coxphw/issues
Author Daniela Dunkler [aut, cre],
Georg Heinze [aut].
Meinhard Ploner [aut]
Maintainer Daniela Dunkler <daniela.dunkler@meduniwien.ac.at>
Repository CRAN
Date/Publication 2023-11-28 14:00:02 UTC
This package implements weighted estimation in Cox regression as proposed by Schemper, Waku arena and Heinze (Statistics in Medicine, 2009, doi:10.1002/sim.3623). Weighted Cox regression provides unbiased average hazard ratio estimates also in case of non-proportional hazards. The package provides options to estimate time-dependent effects conveniently by including interactions of covariates with arbitrary functions of time, with or without making use of the weighting option. For more details we refer to Dunkler, Ploner, Schemper and Heinze (Journal of Statistical Software, 2018, doi:10.18637/jss.v084.i02).

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>coxphw</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>4.0.2</td>
</tr>
<tr>
<td>Date:</td>
<td>2020-06-16</td>
</tr>
<tr>
<td>License:</td>
<td>GPL-2</td>
</tr>
</tbody>
</table>

Main functions included in the coxphw package are

- **coxphw**: weighted estimation of Cox regression: either (recommended) estimation of
average hazard ratios (Schemper et al., 2009), estimation of average regression
effects (Xu and O’Quigley, 2000), or proportional hazards regression.

plot plots the weights used in a weighted Cox regression analysis against time.

concord obtains generalized concordance probabilities with confidence intervals.

predict obtains the effect estimates (of e.g. a nonlinear or a time-dependent effect)
at specified values of a continuous covariable. With plot.coxphw.predict
these relative or log relative hazard versus values of the continuous covariable
can be plotted.

wald obtain Wald chi-squared test statistics and p-values for one or more regression
coefficients given their variance-covariance matrix.

Data sets included in the coxphw package are

<table>
<thead>
<tr>
<th>Biofeedback</th>
<th>Biofeedback treatment data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gastric</td>
<td>Gastric cancer data</td>
</tr>
</tbody>
</table>

Note

The SAS macro WCM with similar functionality can be obtained at

Important version changes:
Up to Version 2.13 coxphw used a slightly different syntax (arguments: AHR, AHR.norobust, ARE, PH, normalize, censcorr, prentice, breslow, taroneware). From Version 3.0.0 on the old syntax is disabled. From Version 4.0.0 fractional polynomials are disabled and plotshape is replaced with predict and plot.coxphw.predict.

Author(s)

Georg Heinze, Meinhard Ploner, Daniela Dunkler
Maintainer: <daniela.dunkler@meduniwien.ac.at>

References


**See Also**

`coxphw`, `concord`, `plot.coxphw`, `predict.coxphw`, `plot.coxphw.predict`, `wald`

**Examples**

```r
## for examples see coxphw

```

---

**biofeedback**

*Biofeedback Treatment Data*

**Description**

In this study the effect of biofeedback treatment on time until treatment success was evaluated in patients suffering from aspiration after head and neck surgery. The outcome of interest was the time from start of treatment until the patient achieved full swallowing rehabilitation (`thdur`). Patients were randomized into two groups (`bfb`): one group of patients received videoendoscopic biofeedback treatment; the other group received the conservative treatment including thermal stimulation with ice and exercises for the lips, tongue, laryngeal closure and elevation. Treatment was started as soon as the healing process after surgery was finished (`theal`).

**Usage**

```r
data(biofeedback)
```

**Format**

A data frame with 33 observations on the following 6 variables:

- `id` the patient id.
- `success` of treatment within the first 100 days; either 0 = no success or 1 = success.
- `thdur` the duration of therapy in days.
- `bfb` indicates the treatment group; either 0 = conservative or 1 = biofeedback.
- `theal` time from surgery to start of therapy in days.
- `log2heal` log2-transformed time from surgery to start of therapy.
Source

Data were supplied by Dr. D.-M. Denk, who gave permission to freely distribute the data and use them for non-commercial purposes.

References


Examples

```r
data("biofeedback")

plot(survfit(Surv(thdur, success) ~ bfb, data = biofeedback), lty = 1:2, las = 1, xlab = "time (days)", ylab = "probability of success")

coxphw(Surv(thdur, success) ~ bfb, data = biofeedback, template = "AHR")
```

### coef.coxphw

*Extract Model Coefficients for Objects of Class coxphw*

**Description**

This class of objects is returned by the coxphw function. Objects of this class have methods for the functions summary, print, coef, vcov, plot and confint.

**Usage**

```r
## S3 method for class 'coxphw'
coef(object, ...)
```

**Arguments**

- `object` object of class coxphw.
- `...` further arguments.

**Author(s)**

Daniela Dunkler

**See Also**

coxphw
**concord**

*Compute Generalized Concordance Probabilities for Objects of Class coxphw or coxph*

**Description**

Compute generalized concordance probabilities with accompanying confidence intervals for objects of class coxphw or coxph.

**Usage**

```r
concord(fit, digits = 4)
```

**Arguments**

- `fit` an object of class coxphw.
- `digits` integer indicating the number of decimal places to be used. Default is 4.

**Details**

The generalized concordance probability is defined as $P(T_i < T_j | x_i = x_j + 1)$ with $T_i$ and $T_j$ as survival times of randomly chosen subjects with covariate values $x_i$ and $x_j$, respectively. Assuming that $x_i$ and $x_j$ are 1 and 0, respectively, this definition includes a two-group comparison.

If proportional hazards can be assumed, the generalized concordance probability can also be derived from Cox proportional hazards regression (coxphw with `template = "PH"` or coxph) or weighted Cox regression as suggested by Xu and O'Quigley (2000) (coxphw with `template = "ARE"`).

If in a fit to coxphw the `betafix` argument was used, then for the fixed parameters only the point estimates are given.

**Value**

A matrix with estimates of the generalized concordance probability with accompanying confidence intervals for each explanatory variable in the model.

**Author(s)**

Daniela Dunkler

**References**


**Description**

Computes confidence intervals for one or more parameters in a model fitted by coxphw. Objects of this class have methods for the functions `summary`, `print`, `coef`, `vcov`, `plot`, and `confint`.

**Usage**

```r
## S3 method for class 'coxphw'
confint(object, parm, level = 0.95, ...)
```

**Arguments**

- `object`:
  a fitted model object of class `coxphw`.

- `parm`:
  a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.

- `level`:
  the confidence level required.

- `...`:
  additional argument(s) for methods.

**Value**

A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labeled as (1-level)/2 and 1 - (1-level)/2 in % (by default 2.5% and 97.5%).

**Author(s)**

Daniela Dunkler

**See Also**

`coxphw`
Weighted Estimation in Cox Regression

Description

Weighted Cox regression as proposed by Schemper et al. (2009) doi:10.1002/sim.3623 provides unbiased estimates of average hazard ratios also in case of non-proportional hazards. Time-dependent effects can be conveniently estimated by including interactions of covariates with arbitrary functions of time, with or without making use of the weighting option.

Usage

```r
coxphw(formula, data, template = c("AHR", "ARE", "PH"), subset, na.action,
       robust = TRUE, jack = FALSE, betafix = NULL, alpha = 0.05,
       trunc.weights = 1, control, caseweights, x = TRUE, y = TRUE,
       verbose = FALSE, sorted = FALSE, id = NULL, clusterid = NULL, ...)
```

Arguments

- **formula**: a formula object with the response on the left of the operator and the model terms on the right. The response must be a survival object as returned by `Surv`. `formula` may include offset-terms or functions of time (see example).
- **data**: a data frame in which to interpret the variables named in `formula`.
- **template**: choose among three pre-defined templates: "AHR" requests estimation of average hazard ratios (Schemper et al., 2009), "ARE" requests estimation of average regression effects (Xu and O’Quigley, 2000) and "PH" requests Cox proportional hazards regression. Recommended and default template is "AHR".
- **subset**: expression indicating which subset of the rows of data should be used in the fit. All observations are included by default.
- **na.action**: missing-data filtering. Defaults to `options$na.action`. Applied after subsetting data, but applied to the all variables in the data set (not only those listed in the formula).
- **robust**: if set to TRUE, the robust covariance estimate (Lin-Wei) is used; otherwise the Lin-Sasieni covariance estimate is applied. Default is TRUE.
- **jack**: if set to TRUE, the variance is based on a complete jackknife. Each individual (as identified by `id`) is left out in turn. The resulting matrix of DFBETA residuals D is then used to compute the variance matrix: \( V = D'D \). Default is FALSE.
- **betafix**: can be used to restrict the estimation of one or more regression coefficients to pre-defined values. A vector with one element for each model term as given in `formula` is expected (with an identical order as in `formula`). If estimation of a model term is requested, then the corresponding element in `betafix` has to be set to NA, otherwise it should be set to the fixed parameter value. The default value is `betafix = NULL`, yielding unrestricted estimation of all regression coefficients.
alpha

the significance level \((1 - \alpha)\), 0.05 as default.

trunc.weights

specifies a quantile at which the (combined normalized) weights are to be truncated. It can be used to increase the precision of the estimates, particularly if template = "AHR" or "ARE" is used. Default is 1 (no truncation). Recommended value is 0.95 for mild truncation.

control

Object of class coxphw.control specifying iteration limit and other control options. Default is coxphw.control(...).

caseweights

vector of case weights, equivalent to weights in coxph. If caseweights 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 caseweights. These weights should not be confused with the weights in weighted Cox regression which account for the non-proportional hazards.

x

requests copying explanatory variables into the output object. Default is TRUE.

y

requests copying survival information into the output object. Default is TRUE.

verbose

requests echoing of intermediate results. Default is FALSE.

sorted

if set to TRUE, the data set will not be sorted prior to passing it to FORTRAN. This may speed up computations. Default is FALSE.

id

a vector of subject identification integer numbers starting from 1 used only if the data are in the counting process format. These IDs are used to compute the robust covariance matrix. If id = NA (the default) the program assumes that each line of the data set refers to a distinct subject.

clusterid

a vector of cluster identification integer numbers starting from 1. These IDs are used to compute the robust covariance matrix. If clusterid = NA (the default) the program assumes that no cluster exist.

... additional arguments.

Details

If Cox’s proportional hazards regression is used in the presence of non-proportional hazards, i.e., with underlying time-dependent hazard ratios of prognostic factors, the average relative risk for such a factor is under- or overestimated and testing power for the corresponding regression parameter is reduced. In such a situation weighted estimation provides a parsimonious alternative to more elaborate modelling of time-dependent effects. Weighted estimation in Cox regression extends the tests by Breslow and Prentice to a multi-covariate situation as does the Cox model to Mantel’s logrank test. Weighted Cox regression can also be seen as a robust alternative to the standard Cox estimator, reducing the influence of outlying survival times on parameter estimates.

Three pre-defined templates can be requested:
1) "AHR", i.e., estimation of average hazard ratios (Schemper et al., 2009) using Prentice weights with censoring correction and robust variance estimation;
2) "ARE", i.e., estimation of average regression effects (Xu and O’Quigley, 2000) using censoring correction and robust variance estimation; or
3) "PH", i.e., Cox proportional hazards regression using robust variance estimation.

Breslow’s tie-handling method is used by the program, other methods to handle ties are currently not available.
A fit of coxphw with template = "PH" will yield identical estimates as a fit of coxph using Breslow’s tie handling method and robust variance estimation (using cluster).

If robust = FALSE, the program estimates the covariance matrix using the Lin (1991) and Sasieni (1993) sandwich estimate \( A^{-1}BA^{-1} \) with \( -A \) and \( -B \) denoting the sum of contributions to the second derivative of the log likelihood, weighted by \( w(t_j) \) and \( w(t_j)^2 \), respectively. This estimate is independent from the scaling of the weights and reduces to the inverse of the information matrix in case of no weighting. However, it is theoretically valid only in case of proportional hazards. Therefore, since application of weighted Cox regression usually implies a violated proportional hazards assumption, the robust Lin-Wei covariance estimate is used by default (robust = TRUE).

If some regression coefficients are held constant using betafix, no standard errors are given for these coefficients as they are not estimated in the model. The global Wald test only relates to those variables for which regression coefficients were estimated.

An offset term can be included in the formula of coxphw. In this way a variable can be specified which is included in the model but its parameter estimate is fixed at 1.

**Value**

A list with the following components:

- **coefficients** the parameter estimates.
- **var** the estimated covariance matrix.
- **df** the degrees of freedom.
- **ci.lower** the lower confidence limits of exp(beta).
- **ci.upper** the upper confidence limits of exp(beta).
- **prob** the p-values.
- **linear.predictors** the linear predictors.
- **n** the number of observations.
- **dfbeta.resid** matrix of DFBETA residuals.
- **iter** the number of iterations needed to converge.
- **method.ties** the ties handling method.
- **PTcoefs** matrix with scale and shift used for pretransformation of fp()-terms.
- **cov.j** the covariance matrix computed by the jackknife method (only computed if jack = TRUE).
- **cov.lw** the covariance matrix computed by the Lin-Wei method (robust covariance)
- **cov.ls** the covariance matrix computed by the Lin-Sasieni method.
- **cov.method** the method used to compute the (displayed) covariance matrix and the standard errors. This method is either "jack" if jack = TRUE, or "Lin-Wei" if jack = FALSE.
- **w.matrix** a matrix with four columns according to the number of uncensored failure times. The first column contains the failure times, the remaining columns (labeled w.raw, w.obsm, and w) contain the raw weights, the weights according to the inverse of the Kaplan-Meier estimates with reverse status indicator and the normalized product of both.
caseweights if x = TRUE the case weights.
Wald Wald-test statistics.
means the means of the covariates.
offset.values offset values.
dataline the first dataline of the input data set (required for plotfp).
x if x = TRUE the explanatory variables.
y the response.
alpha the significance level = 1 - confidence level.
template the requested template.
formula the model formula.
betafix the betafix vector.
call the function call.

Note
The SAS macro WCM with similar functionality is offered for download at https://cemsiis.meduniwien.ac.at/en/kb/science-research/software/statistical-software/wcmcoxphw/.
Up to Version 2.13 coxphw used a slightly different syntax (arguments: AHR, AHR.norobust, ARE, PH, normalize, censcorr, prentice, breslow, taroneware). From Version 3.0.0 on the old syntax is disabled. From Version 4.0.0 estimation of fractional polynomials is disabled.

Author(s)
Georg Heinze, Meinhard Ploner, Daniela Dunkler

References

**See Also**

`concord.plot.coxphw`, `predict.coxphw`, `plot.coxphw`, `predict.coxphw`

**Examples**

```r
data("gastric")

# weighted estimation of average hazard ratio
fit1 <- coxphw(Surv(time, status) ~ radiation, data = gastric, template = "AHR")
summary(fit1)

fit1$cov.lw # robust covariance
fit1$cov.ls # Lin-Sasieni covariance

# unweighted estimation, include interaction with years
# ("radiation" must be included in formula!)

years <- gastric$years / 365.25

fit2 <- coxphw(Surv(years, status) ~ radiation + years : radiation, data = gastric, template = "PH")

summary(fit2)

# unweighted estimation with a function of time

data("gastric")

years <- gastric$years / 365.25

fun <- function(t) { (t > 1) * 1 }

fit3 <- coxphw(Surv(yrs, status) ~ radiation + fun(yrs):radiation, data = gastric, template = "PH")

# for more examples see vignette or predict.coxphw
```

---

**coxphw.control**

Ancillary arguments for controlling coxphw fits

**Description**

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

**Usage**

```r
coxphw.control(iter.max = 200, maxhs = 5, xconv = 1e-4, gconv = 1e-4, maxstep = 1,
round.times.to = 0.00001, add.constant = 0, pc = TRUE, pc.time = TRUE,
normalize = TRUE)
```
Arguments

iter.max  maximum number of iterations to attempt for convergence. Default is 200.

maxhs maximum number of step-halvings per iterations. Default is 5. The increments of the parameter vector in one Newton-Raphson iteration step are halved, unless the new pseudo-likelihood is greater than the old one, maximally doing maxhs halvings.

xconv specifies the maximum allowed change in standardized parameter estimates to declare convergence. Default is 0.0001.

gconv specifies the maximum allowed change in score function to declare convergence. Default is 0.0001.

maxstep specifies the maximum change of (standardized) parameter values allowed in one iteration. Default is 1.

round.times.to rounds survival times to the nearest number that is a multiple of round.times.to. This may improve numerical stability if very small survival times are used (mostly in simulations). Default is 0.00001.

add.constant this number will be added to all times. It can be used if some survival times are exactly 0. Default is 0.

pc if set to TRUE, it will orthogonalize the model matrix to speed up convergence. Default is TRUE.

pc.time if set to TRUE, it will orthogonalize time-dependent covariates (i.e., interactions of covariates with functions of time) to speed up convergence. Default is TRUE.

normalize if set to TRUE, weights are normalized such that their sum is equal to the number of events. This may speed up or enable convergence, but has no consequences on the estimated regression coefficients.

Value

A list containing the values of each of the above constants

Author(s)

Daniela Dunkler

See Also
coxphw
Provides Fractional Polynomials as Accessible Function

Description

Provides fractional polynomials as accessible function.

Usage

fp.power(z, a, b = NULL)

Arguments

z
a scalar or vector of positive numerical values.

a
first power.

b
optional second power.

Details

The function returns fp(a) of z (and optionally fp(b) of z).

Value

A matrix with one or two columns (if a second power b was specified), and number of rows equal to the length of z. The columns are sorted by descending power.

Author(s)

Georg Heinze

References


See Also

coxphw

Examples

fp.power(z = c(1, 4, 6), a = 1)
fp.power(z = c(1, 4, 6), a = 0.5)
fp.power(z = c(1, 4, 6), a = 0.5, b = 0.5)
fp.power(z = c(1, 4, 6), a = 0, b = 2)
**Gastric Cancer Data**

**Description**
A data set of survival times of patients with locally advanced, nonresectable gastric carcinoma. The patients were either treated with chemotherapy plus radiation or chemotherapy alone.

**Usage**
data(gastric)

**Format**
A data frame with 90 observations on the following 4 variables:

- **id**: unique patient id.
- **radiation**: treatment of either 0 = chemotherapy alone or 1 = chemotherapy plus radiation.
- **time**: survival time in days.
- **status**: 0 = censored or 1 = death.

**Source**
https://www.mayo.edu/research/documents/gastrichtml/DOC-10027680

**References**

**Examples**
data("gastric")
plot(survfit(Surv(time, status) ~ radiation, data = gastric), lty = 1:2, las = 1, xscale = 365.25, xlab = "time (years)", ylab = "survival distribution function")
coxphw(Surv(time, status) ~ radiation, data = gastric, template = "AHR")
*plot.coxphw*

Plot Weights of Weighted Estimation in Cox Regression

**Description**

This function plots the weights used in a weighted Cox regression analysis against time.

**Usage**

```r
## S3 method for class 'coxphw'
plot(x, rank = FALSE, log = FALSE, legendxy = NULL,...)
```

**Arguments**

- `x` a `coxphw` object.
- `rank` if set to TRUE, plots the weights against ranked time. Default is FALSE.
- `log` if set to TRUE, shows logarithm of weights. Default is FALSE.
- `legendxy` an optional vector of length 2 of the x and y co-ordinates to be used to position the legend.
- `...` additional arguments for plotting

**Details**

The function plots the survival weights, i.e., the left-continuous survivor function estimates, the censoring weights, i.e., estimates of the follow-up distribution obtained by Kaplan-Meier estimation with reversed meaning of the status indicator and the combined normalized weights, i.e. the product of the survival and the censoring weights, rescaled to a mean of 1.

**Value**

No output value.

**Note**

In `coxphw` version 4.0.0 the new `plot` function replaces the old `plotw` function.

**Author(s)**

Georg Heinze, Daniela Dunkler

**See Also**

`coxphw`
Examples

```r
data("gastric")

# weighted estimation of average hazard ratio
fit1 <- coxphw(Surv(time, status) ~ radiation, data = gastric, template = "AHR")
plot(fit1)

# estimation of average regression effect by inverse probability of censoring weights;
# truncate weights at 95th percentile
fit2 <- coxphw(Surv(time, status) ~ radiation, data = gastric, template = "ARE",
              trunc.weights = 0.95)
plot(fit2)
```

Description

This function visualizes a nonlinear or a time-dependent effect of a predict.coxphw object.

Usage

```r
## S3 method for class 'coxphw.predict'
plot(x, addci = TRUE, xlab = NULL, ylab = NULL, ...)
```

Arguments

- `x`: an output object of coxphw.
- `addci`: confidence intervals are obtained. Default is TRUE.
- `xlab`: label for x-axis of plot, uses variable specified in x in predict as default.
- `ylab`: label for y-axis of plot, uses as appropriate either "relative hazard" or "log relative hazard" as default.
- `...`: further parameters, to be used for plots (e.g., scaling of axes).

Details

This function can be used to depict the estimated nonlinear or time-dependent effect of an object of class predict.coxphw. It supports simple nonlinear effects as well as interaction effects of continuous variables with binary covariates (see examples section in predict.coxphw.).

Value

No output value.

Note

In coxphw version 4.0.0 the old plotshape function is replaced with predict.coxphw and plot.coxphw.predict.
Author(s)
Georg Heinze, Meinhard Ploner, Daniela Dunkler

References

See Also
coxphw, predict.coxphw

Examples

```r
set.seed(30091)
n <- 300
x <- 1:n
true.func <- function(x) 3 * (x / 100)^2 - log(x / 100) - 3 * x / 100
x <- round(rnorm(n = x) * 10 + 40, digits = 0)
time <- rexp(n = n, rate = 1) / exp(true.func(x))
event <- rep(x = 1, times = n)
futime <- runif(n = n, min = 0, max = 309000)
event <- (time < futime) + 0
time[event == 0] <- futime[event == 0]
my.data <- data.frame(x, time, event)
fitahr <- coxphw(Surv(time, event) ~ x, data = my.data, template = "AHR")

# estimated function
plotx <- quantile(x, probs = 0.05):quantile(x, probs = 0.95)
plot(predict(fitahr, type = "shape", newx = plotx, refx = median(x), x = "x",
    verbose = FALSE))

# true function
lines(x = plotx, true.func(plotx) - true.func(median(plotx)), lty = 2)

legend("topright", legend=c("AHR estimates", "true"), bty = "n", lty = 1:2, inset = 0.05)

# for more examples see predict.coxphw
```

---

**predict.coxphw**

Predictions for a weighted Cox model
Description

This function obtains the effect estimates (e.g., of a nonlinear or a time-dependent effect) at specified values of a continuous covariable for a model fitted by `coxphw`. It prints the relative or log relative hazard. Additionally, the linear predictor \( \text{lp} \) or the risk score \( \exp(\text{lp}) \) can be obtained.

Usage

```r
## S3 method for class 'coxphw'
predict(object, type = c("shape", "slice.time", "slice.z", "slice.x", "lp", "risk"),
        x = NULL, newx = NA, refx = NA, z = NULL, at = NULL, exp = FALSE,
        se.fit = FALSE, pval = FALSE, digits = 4, verbose = FALSE, ...)
```

Arguments

- **object**: an output object of `coxphw`.
- **type**: the type of predicted value. Choices are: "lp" for the linear predictors, "risk" for the risk scores \( \exp(\text{lp}) \), "shape" for visualizing a nonlinear effect of \( x \), "slice.x" for slicing an interaction of type \( \text{fun}(x)z \) at values of \( x \), "slice.z" for slicing an interaction of type \( \text{fun}(x)z \) at a value of \( z \), "slice.time" for slicing a time-by-covariate interaction of type \( z + \text{fun}(\text{time})z \) at values of \( \text{time} \). See details.
- **x**: name of the continuous or time variable (use "") for type = "shape", "slice.time", "slice.x", or "slice.z".
- **newx**: the data values for \( x \) for which the effect estimates should be obtained (e.g., 30:70 for type = "shape", "slice.time", "slice.x", or "slice.z").
- **refx**: the reference value for variable \( x \) for type = "shape" or "slice.z". The log relative hazard at this value will be 0. (e.g., \( \text{refx} = 50 \)).
- **z**: variable which is in interaction with \( x \) (use "") for "slice.time", "slice.x", or "slice.z".
- **at**: if type = "slice.z" at which level ("slice") of \( z \) should the effect estimates of the \( x \) be obtained.
- **exp**: if set to TRUE (default), the log relative hazard is given, otherwise the relative hazard is requested for type = "shape", "slice.time", "slice.x", or "slice.z".
- **se.fit**: if set to TRUE, pointwise standard errors are produced for the predictions for type = "shape", "slice.time", "slice.x", or "slice.z".
- **pval**: if set to TRUE add Wald-test \( p \)-values to effect estimates at values of \( \text{newx} \) for type = "shape", "slice.time", "slice.x", or "slice.z". Default is set to FALSE.
- **digits**: number of printed digits. Default is 4.
- **verbose**: if set to TRUE (default), results are printed.
- **...**: further parameters.
Details

This function can be used to depict the estimated nonlinear or time-dependent effect of an object of class coxphw. It supports simple nonlinear effects as well as interaction effects of a continuous variable with a binary covariate or with time (see examples section).

If the effect estimates of a simple nonlinear effect of $x$ without interaction is requested with type = "shape", then $x$ (the usually continuous covariate), $\text{refx}$ (the reference value of $x$) and $\text{newx}$ (for these values of $x$ the effect estimates are obtained) must be given.

If the effect estimates of an interaction of $z$ with $x$ are requested with type = "slice.x", then $x$ (the usually continuous variable), $z$ (the categorical variable) and $\text{newx}$ (for these values of $x$ the effect estimates are obtained) must be given.

If the effect estimates of an interaction of $z$ with $x$ for one level of $z$ are requested with type = "slice.z", then $x$ (the usually continuous variable), $z$ (the categorical variable), $\text{at}$ (at which level of $z$), $\text{refx}$ (the reference value of $x$), and $\text{newx}$ (for these values of $x$ the effect estimates are obtained) must be given.

If the effect estimates of an interaction of $z$ with time are requested with type = "slice.time", then $x$ (the time), $z$ (the categorical variable) and $\text{newx}$ (for these values of $x$ the effect estimates are obtained) must be given.

Note that if the model formula contains time-by-covariate interactions, then the linear predictor and the risk score are obtained for the failure or censoring time of each subject.

Value

If type = "shape", "slice.time", "slice.x", or "slice.z" a list with the following components:

- estimates: a matrix with estimates of (log) relative hazard and corresponding confidence limits.
- se: pointwise standard errors, only if se.fit = TRUE.
- p: p-value, only if pval = TRUE.
- alpha: the significance level = 1 - confidence level.
- exp: an indicator if log relative hazard or relative hazard was obtained.
- x: name of $x$.

If type = "lp" or "risk", a vector.

Note

In coxphw version 4.0.0 the old plotshape function is replaced with predict.coxphw and plot.coxphw.predict.

Author(s)

Georg Heinze, Meinhard Ploner, Daniela Dunkler
References


See Also

coxphw, plot.coxphw.predict

Examples

```r
### Example for type = "slice.time"
data("gastric")
gastric$yrs <- gastric$time / 365.25

# check proportional hazards
fitcox <- coxph(Surv(yrs, status) ~ radiation + cluster(id), data = gastric, x = TRUE,
method = "breslow")
fitcox.ph <- cox.zph(fit = fitcox, transform = "identity")

## compare and visualize linear and log-linear time-dependent effects of radiation
fit1 <- coxphw(Surv(yrs, status) ~ yrs * radiation, data = gastric, template = "PH")
summary(fit1)

predict(fit1, type = "slice.time", x = "yrs", z = "radiation", newx = c(0.5, 1, 2),
verbose = TRUE, exp = TRUE, pval = TRUE)

fit2 <- coxphw(Surv(yrs, status) ~ log(yrs) * radiation, data = gastric, template = "PH")
summary(fit2)

predict(fit2, type = "slice.time", x = "yrs", z = "radiation", newx = c(0.5, 1, 2),
verbose = TRUE, exp = TRUE, pval = TRUE)

plotx <- seq(from = quantile(gastric$yrs, probs = 0.05),
to = quantile(gastric$yrs, probs = 0.95), length = 100)
y1 <- predict(fit1, type = "slice.time", x = "yrs", z = "radiation", newx = plotx)
y2 <- predict(fit2, type = "slice.time", x = "yrs", z = "radiation", newx = plotx)

plot(x = fitcox.ph, se = FALSE, xlim = c(0, 3), las = 1, lty = 3)
abline(a = 0, b = 0, lty = 3)
lines(x = plotx, y = y1$estimates[, "coef"], col = "red", lty = 1, lwd = 2)
lines(x = plotx, y = y2$estimates[, "coef"], col = "blue", lty = 2, lwd = 2)
legend(x = 1.7, y = 1.6, title = "time-dependent effect", title.col = "black",
legend = c("LOWESS", "linear", "log-linear"), col = c("black", "red", "blue"),
lty = c(3, 1:2), bty = "n", lwd = 2, text.col = c("black", "red", "blue"))
```
### Example for type = "shape"

```r
set.seed(512364)
n <- 200
x <- 1:n
true.func <- function(x) 2.5 * log(x) - 2
x <- round(runif(n) * 60 + 10, digits = 0)
time <- round(100000 * rexp(n = n, rate = 1) / exp(true.func(x)), digits = 1)
event <- rep(x = 1, times = n)
my.data <- data.frame(x, time, event)

fit <- coxphw(Surv(time, event) ~ log(x) + x, data = my.data, template = "AHR")
predict(fit, type = "shape", newx = c(30, 50), refx = 40, x = "x", verbose = TRUE)
```

```r
plotx <- seq(from = quantile(x, probs = 0.05), to = quantile(x, probs = 0.95), length = 100)
plot(predict(fit, type = "shape", newx = plotx, refx = 40, x = "x"))
```

### Example for type = "slice.x" and "slice.z"

```r
set.seed(75315)
n <- 200
trt <- rbinom(n = n, size = 1, prob = 0.5)
x <- 1:n
true.func <- function(x) 2.5 * log(x) - 2
x <- round(runif(n = x) * 60 + 10, digits = 0)
time <- 100 * rexp(n = n, rate = 1) / exp(true.func(x) / 4 * trt - (true.func(x) / 4)^2 * (trt==0))
event <- rep(x = 1, times = n)
my.data <- data.frame(x, trt, time, event)

fun <- function(x) x^(-2)
fit <- coxphw(Surv(time, event) ~ x * trt + fun(x) * trt, data = my.data, template = "AHR", verbose = FALSE)

# plots the interaction of trt with x (the effect of trt dependent on the values of x)
plotx <- quantile(x, probs = 0.05):quantile(x, probs = 0.95)
plot(predict(fit, type = "slice.x", x = "x", z = "trt", newx = plotx, refx = median(x), verbose = FALSE), main = "interaction of trt with x")

# plot the effect of x in subjects with trt = 0
y0 <- predict(fit, type = "slice.z", x = "x", z = "trt", at = 0, newx = plotx, refx = median(x), verbose = FALSE)
plot(y0, main = "effect of x in subjects with trt = 0")

# plot the effect of x in subjects with trt = 1
y1 <- predict(fit, type = "slice.z", x = "x", z = "trt", at = 1, newx = plotx, refx = median(x), verbose = FALSE)
plot(y1, main = "effect of x in subjects with trt = 1")
```
# Example for type = "slice.time"
set.seed(23917)
time <- 100 * rexp(n = n, rate = 1) / exp((true.func(x) / 10)^2 / 2000 * trt + trt)
event <- rep(x = 1, times = n)
my.data <- data.frame(x, trt, time, event)
plot.x <- seq(from = 1, to = 100, by = 1)

fun <- function(t) { PT(t)^-2 * log(PT(t)) }
fun2 <- function(t) { PT(t)^-2 }
fitahr <- coxphw(Surv(time, event) ~ fun(time) * trt + fun2(time) * trt + x,
data = my.data, template = "AHR")
yahr <- predict(fitahr, type = "slice.time", x = "time", z = "trt", newx = plot.x)

fitph <- coxphw(Surv(time, event) ~ fun(time) * trt + fun2(time) * trt + x,
data = my.data, template = "PH")
yph <- predict(fitph, type = "slice.time", x = "time", z = "trt", newx = plot.x)

plot(yahr, addci = FALSE)
lines(yph$estimates$time, yph$estimates$coef, lty = 2)
legend("bottomright", legend = c("AHR", "PH"), bty = "n", lty = 1:2,
inset = 0.05)

---

**print.coxphw**

**Print Method for Objects of Class coxphw**

**Description**

This class of objects is returned by the coxphw function. Objects of this class have methods for the functions summary, print, coef, vcov, plot, and confint.

**Usage**

```r
## S3 method for class 'coxphw'
print(x, ...)```

**Arguments**

- `x` object of class coxphw.
- `...` further arguments.

**Details**

If some regression coefficients were held fixed by `betafix`, then no standard errors are given for these coefficients as they are not estimated in the model. The global Wald test only relates to those variables for which regression coefficients were estimated.

**Author(s)**

Georg Heinze, Daniela Dunkler
See Also

coxphw

print.coxphw.predict  Print Method for Objects of Class predict.coxphw

Description

This class of objects is returned by the predict.coxphw function. Objects of this class have methods for the functions print and plot.

Usage

## S3 method for class 'coxphw.predict'
print(x, ...)

Arguments

x  object of class coxphw.predict.

...  further arguments.

Author(s)

Daniela Dunkler

See Also

coxphw, predict.coxphw plot.coxphw.predict

PT  Pretransformation function

Description

Provides automatic pretransformation of variables (to well-scaled and nonzero values).

Usage

PT(z)

Arguments

z  a vector of numerical values.
The function transforms a variable by shifting to positive values, and dividing by scaling factor (a power of 10) such that the standard deviation is approximately equal to 1.

\[(z + \text{shift}) / \text{scale}\]

Author(s)
Georg Heinze

See Also
coxphw

Examples
PT(z = c(-6, -1, 4, 6))
See Also

coxphw

vcov.coxphw

Obtain the Variance-Covariance Matrix for a Fitted Model Object of Class coxphw

Description

This class of objects is returned by the coxphw function. Objects of this class have methods for the functions summary, print, coef, vcov, plot, and confint.

Usage

## S3 method for class 'coxphw'
vcov(object, ...)

Arguments

object object of class coxphw.
... further arguments.

Author(s)

Daniela Dunkler

See Also

coxphw

wald

Wald-Test for Model Coefficients

Description

Obtain Wald chi-squared test statistics and p-values for one or more regression coefficients given their variance-covariance matrix.

Usage

wald(coef, cov, index = NULL, h0 = NULL)
Arguments

coeff  a vector with regression coefficients.
cov    a variance-covariance matrix.
index  an integer specifying which parameters should be jointly tested. Default is to
test all parameters given in coeff and cov.
h0     a vector with the same length as coeff stating the null hypothesis for the test.
        Default is 0 for all coefficients.

Details

The test is based on the assumption that the coefficients asymptotically follow a (multivariate)
normal distribution with mean coeff and a variance-covariance matrix cov.

Value

A vector with the following components:

chi2   the Wald-test statistic.
df     degrees of freedom.
p      p-value.

Author(s)

Daniela Dunkler

See Also

coxphw
Index

* datasets
  biofeedback, 4
  gastric, 15
* math
  fp.power, 14
  PT, 24
* package
  coxphw-package, 2
* regression
  coxphw, 8
  coxphw-package, 2
* survival
  concord, 6
  coxphw, 8
  coxphw-package, 2
  coxphw.control, 12
  fp.power, 14
  plot.coxphw, 16
  plot.coxphw.predict, 17
  predict.coxphw, 18
  wald, 26
* utilities
  coef.coxphw, 5
  confint.coxphw, 7
  print.coxphw, 23
  print.coxphw.predict, 24
  summary.coxphw, 25
  vcov.coxphw, 26

biofeedback, 3, 4
cluster, 10
coef.coxphw, 5
concord, 3, 4, 6, 12
confint.coxphw, 7
coxph, 9, 10, 12
coxphw, 2, 4, 5, 7, 8, 13, 14, 16, 18, 19, 21, 24–27
coxphw-package, 2
coxphw.control, 9, 12
fp.power, 14
gastric, 3, 15
offset, 10
plot, 3
plot.coxphw, 4, 12, 16
plot.coxphw.predict, 3, 4, 12, 17, 21, 24
predict, 3
predict.coxphw, 4, 12, 17, 18, 18, 24
print.coxphw, 23
print.coxphw.predict, 24
PT, 24
summary.coxphw, 25
Surv, 8
vcov.coxphw, 26
wald, 3, 4, 26