

Package ‘proptest’

January 2, 2012

Type Package

Title Tests of the Proportional Hazards Assumption

Version 0.1-5

Date 2010-11-07

Author David Kraus

Maintainer David Kraus <david.kraus@matfyz.cz>

Depends survival (>= 2.35-4)

Description Tests of the proportional hazards assumption in the Cox model: data-driven Neyman type smooth tests and score process based tests for identifying nonproportional covariates and for global checks.

License GPL (>= 2)

URL <http://www.davidkraus.net/proptest/>

Repository CRAN

Date/Publication 2010-11-08 07:32:30

R topics documented:

proptest-package	2
plot.scoreproptest	3
scoreproptest	5
smoothproptest	8

Index	11
--------------	-----------

Description

Tests of the proportional hazards assumption in the Cox model: data-driven Neyman type smooth tests and score process based tests for identifying nonproportional covariates and for global checks.

Details

The package provides two functions for testing proportional hazards: `smoothproptest` for smooth tests, and `scoreproptest` for tests based on the score process.

Smooth tests consist of expressing the coefficient of the tested covariate or coefficients of all of the covariates as linear combinations of some basis functions, i.e., they consist of introducing artificial time-dependent covariates and testing their significance. A data-driven choice of these covariates is implemented.

Score process tests are tests based on functionals (Kolmogorov–Smirnov, Cramer–von Mises, Anderson–Darling) of components of the score process or on the supremum of a test process formed from the whole score process. Both numerical and visual assessment based on simulations is possible.

Both types of tests may be used for *global* verification as well as for testing *individual* covariates because the covariates that are not tested may be modeled by smooth functions.

Author(s)

David Kraus, <http://www.davidkraus.net/>

References

Kraus, D. (2007) Data-driven smooth tests of the proportional hazards assumption. *Lifetime Data Anal.* **13**, 1–16.

Kraus, D. (2008) Identifying nonproportional covariates in the Cox model. *Comm. Statist. Theory Methods* **37**, 617–625.

Lin, D.Y., Wei, L.J. and Ying, Z. (1993) Checking the Cox model with cumulative sums of martingale-based residuals. *Biometrika*, **80**, 557–572.

Examples

```
## Case 4 of Kvaloy & Neef (2004, Lifetime Data Anal.):
## data generated from the distribution with hazard rate
## \lambda(t)=\exp(0.5tZ_1+Z_2-8)
## (Z_1,Z_2) jointly normal with E=4, var=1, cor=rho
## censoring times uniform(0,5)

n = 200
rho = .3
z = matrix(rnorm(n*2),ncol=2) %*% chol(matrix(c(1,rho,rho,1),2)) + 4
```

```

a = .5
tim = 1/(a*z[,1]) * log(1-a*z[,1]*exp(-z[,2]+8)*log(runif(n)))
ct = 5*runif(n)
nc = tim<=ct
tim = pmin(tim,ct)
fit = coxph(Surv(tim,nc)~z)

## Data-driven smooth tests

test1 = smoothproptest(fit,covariate=1) # testing Z_1 (nonproportional)
summary(test1) # print details on the alternative models

test2 = smoothproptest(fit,covariate=2) # testing Z_2 (proportional)
summary(test2) # print details on the alternative models

test = smoothproptest(fit,global=TRUE) # global test
summary(test) # print details on the alternative models

## Tests based on the score process

par(mfrow=c(3,1))

test1 = scoreproptest(fit,covariate=1) # testing Z_1 (nonproportional)
print(test1)
plot(test1,main="Score process for z1")

test2 = scoreproptest(fit,covariate=2) # testing Z_2 (proportional)
print(test2)
plot(test2,main="Score process for z2")

test = scoreproptest(fit,global=TRUE) # global test of proportional hazards
print(test)
plot(test,main="Global test process")

par(mfrow=c(1,1))

```

plot.scoreproptest *Plotting the Observed Test Process and its Simulations*

Description

The function plots the observed test process and a number of its realisations simulated under the hypothesis of proportional hazards.

Usage

```

## S3 method for class 'scoreproptest'
plot(x, nsim.plot = x$nsim.plot, ...)

```

Arguments

x	an object of class "scoreproptest" (output of <code>scoreproptest</code>).
nsim.plot	the number of simulated paths of the test process to be plotted. It must not be greater than <code>x\$nsim.plot</code> .
...	further plotting parameters.

Details

By plotting the observed path of the test process along with its simulations, one can visually assess the time-constancy of the effect of the corresponding covariate or the global proportional hazards assumption.

The function plots `x$score.process` or `x$test.process` and the first `nsim.plot` realisations contained in `x$score.process.sim` or `x$test.process.sim`.

Author(s)

David Kraus, <http://www.davidkraus.net/>

References

Lin, D.Y., Wei, L.J. and Ying, Z. (1993) Checking the Cox model with cumulative sums of martingale-based residuals. *Biometrika*, **80**, 557–572.

See Also

[scoreproptest](#)

Examples

```
## Case 4 of Kvaloy & Neef (2004, Lifetime Data Anal.):
## data generated from the distribution with hazard rate
## \lambda(t)=\exp(0.5tZ_1+Z_2-8)
## (Z_1,Z_2) jointly normal with E=4, var=1, cor=rho
## censoring times uniform(0,5)

n = 200
rho = .3
z = matrix(rnorm(n*2),ncol=2) %*% chol(matrix(c(1,rho,rho,1),2)) + 4
a = .5
tim = 1/(a*z[,1]) * log(1-a*z[,1]*exp(-z[,2]+8)*log(runif(n)))
ct = 5*runif(n)
nc = tim<=ct
tim = pmin(tim,ct)
fit = coxph(Surv(tim,nc)~z)

par(mfrow=c(3,1))

test1 = scoreproptest(fit,covariate=1) # testing Z_1 (nonproportional)
print(test1)
plot(test1,main="Score process for z1")
```

```

test2 = scoreproptest(fit,covariate=2) # testing Z_2 (proportional)
print(test2)
plot(test2,main="Score process for z2")

test = scoreproptest(fit,global=TRUE) # global test of proportional hazards
print(test)
plot(test,main="Global test process")

par(mfrow=c(1,1))

```

scoreproptest	<i>Test of the Proportional Hazards Assumption Based on the Score Process</i>
---------------	-------------------------------------------------------------------------------

Description

The function performs tests of the proportional hazards assumption for an *individual* covariate or a *global* test of proportionality in the Cox model for right censored survival data. The individual covariate tests of the Kolmogorov–Smirnov, Cramer–von Mises and Anderson–Darling type are based on the corresponding components of the score process. The global test is a supremum test using all of the components. p -values are approximated by simulations.

Usage

```

scoreproptest(fit, covariate = 1, global = FALSE, dims = 4,
              basis = "legendre", time.transf = "F", nsim = 1000,
              nsim.plot = 50, weight = "unit")

```

Arguments

<code>fit</code>	a Cox model fit (an output of <code>coxph</code>).
<code>covariate</code>	integer determining which covariate is to be tested for proportionality. Only used if <code>global</code> is <code>FALSE</code> .
<code>global</code>	logical. Should the global test be performed?
<code>dims</code>	a vector or a single value. <code>dims</code> is only used if an individual covariate test is performed. It gives dimensions for smooth modeling of the effects of the covariates that are not tested. If <code>dims</code> is a single value and there is more than one covariate, the value is replicated.
<code>basis</code>	a character string. <code>basis</code> is only used if an individual covariate test is performed. Which basis of smooth functions is to be used? Possible values are "legendre" and "cos" (or "cosine").
<code>time.transf</code>	a character string. <code>time.transf</code> is only used if an individual covariate test is performed. The basis functions are evaluated at transformed times. With <code>time.transf="F"</code> , the transformation is $F_0(t)/F_0(\tau)$ (F_0 is the distribution function corresponding to the baseline hazard). For <code>time.transf="L"</code> , the transformation is $\Lambda_0(t)/\Lambda_0(\tau)$ (Λ_0 is the cumulative baseline hazard). F_0 and Λ_0 are estimated from the input model fit.

<code>nsim</code>	the number of simulations to be carried out to approximate the p -value.
<code>nsim.plot</code>	the number of simulated paths of the score process to be returned (intended for plotting with <code>plot.scoreproptest</code>).
<code>weight</code>	a character string. The weighted process can be used. Possible values are "unit" (unweighted, default), "gehan" ($nrisk/nevent$) and "mm" ($(nrisk - nrisk[nevent])^2 / (nevent^2)$) (Marzec and Marzec, 1997).

Details

The score process is used for assessment of the proportional hazards assumption (Lin, Wei and Ying, 1993). Either global or individual covariate tests are possible.

Each component of the score process reflects departures from proportionality (time-constancy of the effect) of the corresponding covariate. However, tests based directly on individual components of the process are generally not capable to distinguish which covariate is proportional and which not. The method is only valid if the other covariates are proportional. Therefore, the potentially time-varying effects of the covariates that are not tested are modeled as combinations of basis functions. The vector `dims` gives the number of the basis functions for each covariate. The test is then based on the score process from this large model with artificial time-dependent covariates. This makes it possible to perform *individual* covariate tests. See Kraus (2006).

The *global* test of the PH assumption is based on the supremum of a test process which is a sum of absolute values of normalised components of the score process (Lin, Wei and Ying, 1993).

Value

A list (an object of class "scoreproptest"). The most important components are:

<code>time</code>	the vector of the event times.
<code>score.process</code>	the component of the score process corresponding to the tested covariate. (For individual tests.)
<code>stat.ks</code>	the Kolmogorov–Smirnov test statistic. (For individual tests.)
<code>p.ks</code>	the simulated p -value for the Kolmogorov–Smirnov test. (For individual tests.)
<code>stat.cm</code>	the Cramer–von Mises test statistic. (For individual tests.)
<code>p.cm</code>	the simulated p -value for the Cramer–von Mises test. (For individual tests.)
<code>stat.ad</code>	the Anderson–Darling test statistic. (For individual tests.)
<code>p.ad</code>	the simulated p -value for the Anderson–Darling test. (For individual tests.)
<code>score.process.sim</code>	a matrix with <code>nsim.plot</code> columns containing simulated paths of the score process. These may be plotted with <code>plot.scoreproptest</code> . (For individual tests.)
<code>test.process</code>	the test process. (For the global test.)
<code>stat</code>	the test statistic (the supremum of the test process). (For the global test.)
<code>p</code>	the simulated p -value. (For the global test.)
<code>test.process.sim</code>	a matrix with <code>nsim.plot</code> columns containing simulated paths of the test process. These may be plotted with <code>plot.scoreproptest</code> . (For the global test.)

Author(s)

David Kraus, <http://www.davidkraus.net/>

References

- Kraus, D. (2008) Identifying nonproportional covariates in the Cox model. *Comm. Statist. Theory Methods* **37**, 617–625.
- Lin, D.Y., Wei, L.J. and Ying, Z. (1993) Checking the Cox model with cumulative sums of martingale-based residuals. *Biometrika*, **80**, 557–572.
- Marzec, L. and Marzec, P. (1997) Generalized martingale-residual processes for goodness-of-fit inference in Cox's type regression models. *Ann. Statist.*, **25**, 683–714.

See Also

[plot.scoreproptest](#), [smoothproptest](#), [coxph](#)

Examples

```
## Case 4 of Kvaloy & Neef (2004, Lifetime Data Anal.):
## data generated from the distribution with hazard rate
## \lambda(t)=\exp(0.5tZ_1+Z_2-8)
## (Z_1,Z_2) jointly normal with E=4, var=1, cor=rho
## censoring times uniform(0,5)

n = 200
rho = .3
z = matrix(rnorm(n*2),ncol=2) %*% chol(matrix(c(1,rho,rho,1),2)) + 4
a = .5
tim = 1/(a*z[,1]) * log(1-a*z[,1]*exp(-z[,2]+8)*log(runif(n)))
ct = 5*runif(n)
nc = tim<=ct
tim = pmin(tim,ct)
fit = coxph(Surv(tim,nc)~z)

par(mfrow=c(3,1))

test1 = scoreproptest(fit,covariate=1) # testing Z_1 (nonproportional)
print(test1)
plot(test1,main="Score process for z1")

test2 = scoreproptest(fit,covariate=2) # testing Z_2 (proportional)
print(test2)
plot(test2,main="Score process for z2")

test = scoreproptest(fit,global=TRUE) # global test of proportional hazards
print(test)
plot(test,main="Global test process")

par(mfrow=c(1,1))
```

Description

The function performs the Neyman type smooth test of the proportional hazards assumption for an *individual* covariate or *globally* for all of the covariates in the Cox model for right censored survival data. Both a fixed and data-driven choice of the alternative model is possible.

Usage

```
smoothproptest(fit, covariate = 1, global = FALSE, dims = 4,
               basis = "legendre", time.transf = "F",
               data.driven = TRUE, nsim = 1000)
```

```
## S3 method for class 'smoothproptest'
summary(object, ...)
```

Arguments

<code>fit</code>	a Cox model fit (an output of <code>coxph</code>).
<code>covariate</code>	integer determining which covariate is to be tested for proportionality. Only used if <code>global</code> is FALSE.
<code>global</code>	logical. Should the global test be performed?
<code>dims</code>	a vector or a single value. <code>dims</code> gives dimensions for smooth modelling of the effects of respective covariates. If <code>dims</code> is a single value and there is more than one covariate, the value is replicated.
<code>basis</code>	a character string. Which basis of smooth functions is to be used? Possible values are "legendre" and "cos" (or "cosine").
<code>time.transf</code>	a character string. The basis functions are evaluated at transformed times. With <code>time.transf="F"</code> , the transformation is $F_0(t)/F_0(\tau)$ (F_0 is the distribution function corresponding to the baseline hazard). For <code>time.transf="L"</code> , the transformation is $\Lambda_0(t)/\Lambda_0(\tau)$ (Λ_0 is the cumulative baseline hazard). F_0 and Λ_0 are estimated.
<code>data.driven</code>	logical. Should the BIC be used to select the alternative?
<code>nsim</code>	the number of simulations to be carried out to compute the two-term approximation of the p -value. This is only used when the test is global.
<code>object</code>	an object of class "smoothproptest", as returned by the function <code>smoothproptest</code> .
<code>...</code>	further parameters for printing.

Details

The Neyman type smooth test of proportionality (time-constancy of the coefficient) of an *individual* covariate against the alternative of the time-varying coefficient consists of expressing the coefficient of the tested covariate as a linear combination of basis functions and testing significance of the new artificial time-dependent covariates using the partial likelihood score test. The potentially time-varying effects of the covariates that are not tested should be modelled as combinations of basis functions too (`dims` gives dimensions for smooth modelling of the effects of the covariates which are not tested, and of the tested covariate). This makes it possible to perform individual covariate tests. Not modelling the other covariates would be dangerous: the test generally would not be able to distinguish which covariate is proportional and which not. See Kraus (2008).

In the *global* test the coefficients of all of the covariates are expressed as linear combinations of basis functions and tested for significance. The vector `dims` gives the number of the basis functions for each covariate.

Both *individual* and *global* tests can be data-driven which means that the alternative is selected by a BIC-like rule. For individual (covariate-specific) tests the number of basis functions for the tested covariate is selected (see Kraus, 2007a). For global tests, the order of approximation of possibly time-varying effects is selected for each of the covariates (see Kraus, 2007b, Chapter 6).

The distribution of data-driven test statistics is approximated by the two-term H -approximation (Kraus, 2007a,b) because the one-term asymptotic approximation is inaccurate.

If the test is data-driven, the summary method prints details on the selection procedure (statistics and penalised statistics for each alternative model). This is equivalent to `print(x, detail=TRUE, ...)`.

Value

A list (an object of class "smoothproptest") containing some of input values, the test statistic(s) and p -value(s). The most important components are:

<code>stat</code>	the test statistic (<code>stat.bic</code> if <code>data.driven=TRUE</code> , <code>stat.d</code> otherwise).
<code>p</code>	the p -value corresponding to <code>stat</code> (one of <code>p.d</code> or <code>p.bic.h</code> below).
<code>stat.d</code> , <code>stat.bic</code>	the test statistic of the fixed dimension and data-driven test (the latter present only if <code>data.driven</code> is <code>TRUE</code>).
<code>p.d</code>	the p -value of the fixed dimension test based on the asymptotic chi-square with d df (<code>p.d</code>). Here d is either <code>dims[covariate]</code> for the individual covariate test or <code>sum(dims)</code> for the global test.
<code>p.bic.asympt</code> , <code>p.bic.h</code>	p -values for the data-driven test. <code>p.bic.asympt</code> is based on the asymptotic distribution (chi-square with 1 df for covariate-specific tests, max of chi-square with 1 df for global tests) (inaccurate). <code>p.bic.h</code> is based on the two-term approximation.
<code>scorestats</code> , <code>scorestats.penal</code>	statistics and penalised statistics for all alternative models (only for data-driven tests).
<code>alt</code>	all alternative models (only for data-driven tests).
<code>S</code>	the index of the selected alternative (only for data-driven tests). <code>alt[S,]</code> is the selected alternative.

Author(s)

David Kraus, <http://www.davidkraus.net/>

References

Kraus, D. (2007a) Data-driven smooth tests of the proportional hazards assumption. *Lifetime Data Anal.* **13**, 1–16.

Kraus, D. (2007b) Neyman's smooth tests in survival analysis. PhD thesis. Charles University in Prague, Dept. of Statistics.

Kraus, D. (2008) Identifying nonproportional covariates in the Cox model. *Comm. Statist. Theory Methods* **37**, 617–625.

See Also

[scoreproptest](#), [coxph](#)

Examples

```
## Case 4 of Kvaloy & Neef (2004, Lifetime Data Anal.):
## data generated from the distribution with hazard rate
## \lambda(t)=\exp(0.5tZ_1+Z_2-8)
## (Z_1,Z_2) jointly normal with E=4, var=1, cor=rho
## censoring times uniform(0,5)

n = 200
rho = .3
z = matrix(rnorm(n*2),ncol=2) %%% chol(matrix(c(1,rho,rho,1),2)) + 4
a = .5
tim = 1/(a*z[,1]) * log(1-a*z[,1]*exp(-z[,2]+8)*log(runif(n)))
ct = 5*runif(n)
nc = tim<=ct
tim = pmin(tim,ct)
fit = coxph(Surv(tim,nc)~z)

## Covariate-specific tests

test1 = smoothproptest(fit,covariate=1) # testing Z_1 (nonproportional)
summary(test1) # print details on the alternative models

test2 = smoothproptest(fit,covariate=2) # testing Z_2 (proportional)
summary(test2) # print details on the alternative models

## Global test

test = smoothproptest(fit,global=TRUE) # global test
summary(test) # print details on the alternative models
```

Index

*Topic **package**

proptest-package, [2](#)

*Topic **survival**

plot.scoreproptest, [3](#)

proptest-package, [2](#)

scoreproptest, [5](#)

smoothproptest, [8](#)

coxph, [5](#), [7](#), [8](#), [10](#)

plot.scoreproptest, [3](#), [6](#), [7](#)

proptest (proptest-package), [2](#)

proptest-package, [2](#)

scoreproptest, [2](#), [4](#), [5](#), [10](#)

smoothproptest, [2](#), [7](#), [8](#)

summary.smoothproptest
(smoothproptest), [8](#)