

Package ‘surv2sample’

April 19, 2009

Type Package

Title Two-Sample Tests for Survival Analysis

Version 0.1-2

Date 2008-02-28

Author David Kraus

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

Depends survival

Description The package provides tests for comparing two survival distributions, testing equality of two cumulative incidence functions under competing risks and checking goodness of fit of proportional rate models (proportional hazards, proportional odds) for two samples.

License GPL (>= 2)

URL <http://www.davidkraus.net/surv2sample/>

Repository CRAN

Date/Publication 2008-02-28 07:59:40

R topics documented:

surv2sample-package	2
bmt1	3
cif	4
cif2.int	5
cif2.ks	7
cif2.logrank	8
cif2.neyman	10
gastric	12
hepatitis	13
plot.cif	13
plot.lwy.test	15

plot.proprate2.fit	16
proprate2	17
proprate2.gs	19
proprate2.ks	21
proprate2.neyman	22
surv2.ks	25
surv2.logrank	27
surv2.neyman	29
Survcomp	31

Index	33
--------------	-----------

surv2sample-package

Two-Sample Tests for Survival Analysis

Description

The package provides tests for comparing two survival distributions, testing equality of two cumulative incidence functions under competing risks and checking goodness of fit of proportional rate models (proportional hazards, proportional odds) for two samples.

Details

The package implements many two-sample tests for right-censored survival data. Three main areas and corresponding methods are:

- comparison of two survival distributions
 - `surv2.logrank`: weighted logrank tests and their combinations (max, sum)
 - `surv2.neyman`: Neyman’s smooth test and its data-driven version
 - `surv2.ks`: Kolmogorov–Smirnov, Cramer–von Mises and Anderson–Darling test
- comparison of two cumulative incidence functions for competing risks data
 - `cif`: estimation and plotting of cumulative incidence functions
 - `cif2.logrank`: logrank-type test for subdistribution hazards
 - `cif2.neyman`: Neyman’s smooth test and its data-driven version
 - `cif2.ks`: Kolmogorov–Smirnov test
 - `cif2.int`: integrated-difference test
- goodness of fit tests of the proportional rate assumption (proportional hazards or proportional odds functions in two samples)
 - `proprate2`: estimation based on the simplified partial likelihood
 - `proprate2.ks`: Kolmogorov–Smirnov test
 - `proprate2.neyman`: Neyman’s smooth test and its data-driven version
 - `proprate2.gs`: Gill–Schumacher type test

The package also provides three datasets:

- `gastric`: gastric cancer data
- `bmt1`: bone marrow transplant data
- `hepatitis`: chronic active hepatitis data

Author(s)

David Kraus (<http://www.davidkraus.net/>)

References

There are several technical reports on the topics covered by this package available at <http://www.davidkraus.net/surv2sample/>.

bmt1

Bone Marrow Transplant Data

Description

Data on the treatment of leukaemia by the bone marrow transplantation from two types of donors (HLA-identical sibling, HLA-matched unrelated), with two competing causes of failure (relapse, death in remission).

Usage

```
data(bmt1)
```

Format

A data frame with 1607 observations on the following 3 variables:

time time from bone marrow transplantation, in months

event 0 = censoring, 1 = relapse (recurrence of the primary disease), 2 = death in remission (treatment related mortality)

donor 1 = HLA-identical sibling donor, 2 = HLA-matched unrelated donor (HLA = human leukocyte antigen)

Source

Klein, J. P. and Andersen, P. K. (2005) Regression modeling of competing risks data based on pseudovalues of the cumulative incidence function. *Biometrics* **61**, 223–229.

The original dataset was obtained from the website of Biometrics (<http://biometrics.tibs.org/>), where it is available as supplementary material to Klein and Andersen's paper (direct link: <http://biometrics.tibs.org/datasets/031209.txt>). This is a subset (1607 observations) of the original dataset (1715), 108 cases with mismatched unrelated donors were omitted.

Examples

```
data(bmt1)

## plot aggregate cumulative incidence functions for each
## donor type
with(bmt1, plot(cif(Survcomp(time, event), donor)))
```

cif *Estimates of Cumulative Incidence Functions for Competing Risks Data*

Description

Compute estimates of cumulative incidence functions for one or more samples of censored data with several competing risks (types of failure).

Usage

```
cif(x, group)
```

Arguments

`x` an object of class "`Survcomp`".

`group` a vector giving for each observation the group to which the observation belongs. If there are K samples, `group` may only contain values $1, \dots, K$. If `group` is missing, observations are assumed to be one sample.

Details

The cumulative incidence function $F(t, j)$ is defined as the probability of failure by time t from a particular cause j in the presence of other competing risks $1, \dots, j-1, j+1, \dots, J$, that is, $F(t, j) = P(T \leq t, \epsilon = j)$, where T is the failure time and ϵ is the failure cause.

The function `cif` estimates cumulative incidence functions for all causes and all groups.

Value

`cif` returns a list of class "`cif`" with the following components:

<code>[[k]]</code>	a list containing estimates for the k th sample. It has components <code>time</code> (sorted times in the group k), <code>surv</code> (the Kaplan–Meier estimate of the overall survival in this group), and <code>f</code> , which is a matrix containing estimates of cumulative incidence functions for the k th sample. The j th column of <code>f</code> is the cumulative incidence function for the cause j .
<code>time</code>	the vector of sorted times.
<code>event</code>	the vector of corresponding event types.
<code>group</code>	the vector of corresponding group indicators.
<code>ncauses</code>	the number of causes present in the data.
<code>ngroups</code>	the number of samples.

Author(s)

David Kraus (<http://www.davidkraus.net/>)

References

Klein, J. P. and Moeschberger, M. L. (2003) *Survival Analysis. Techniques for Censored and Truncated Data*. Springer, New York. (Section 2.7)

See Also

[Survcomp](#), [plot.cif](#)

Examples

```
## bone marrow transplant data
data(bmt1)

print(a <- cif(Survcomp(bmt1$time, bmt1$event), bmt1$donor))
str(a)

## several first times and cifs for group 1 (HLA-identical
## sibling donors)
head(cbind(a[[1]]$time, a[[1]]$f))
## several last times and cifs for group 2 (HLA-matched
## unrelated donors)
tail(cbind(a[[2]]$time, a[[2]]$f))

## plot aggregate cumulative incidence functions for each
## donor type to see probabilities within groups
plot(a)
```

cif2.int

Two-Sample Integrated-Difference Test for Cumulative Incidence Functions

Description

Compares cumulative incidence functions (CIF) for one failure cause in two samples of censored competing risks data using the test based on the integrated difference of CIFs.

Usage

```
cif2.int(x, group, cause = 1, tau, nsim = 0)
```

Arguments

x	a "Survcomp" object, as returned by the <code>Survcomp</code> function.
group	a vector indicating to which group each observation belongs. May contain values 1 and 2 only.
cause	For which cause of failure should the CIFs be compared?
tau	the upper limit of the integral in the test statistic. If missing, defaults to the maximum of times.
nsim	the number of simulations used to approximate the distribution of the test statistic. If 0, no simulations are carried out and the asymptotic normal approximation is used.

Details

The test compares cumulative incidence functions $F_1(t, k)$, $F_2(t, k)$ for a particular failure cause k .

The method is based on the statistic proposed by Pepe (1991) which is the integral of $F_2(t, k) - F_1(t, k)$ from 0 to τ . The martingale-based simulation technique and the variance estimator are described in Bajorunaite and Klein (2007).

Value

A list of class "cif2.int" with components:

stat	the test statistic.
pval.asympt	the p -value based on the asymptotic normality.
pval.sim	the p -value based on simulations (if <code>nsim</code> >0).

Further components are `cause`, `tau`, `nsim`, the same as on input.

Author(s)

David Kraus (<http://www.davidkraus.net/>)

References

Bajorunaite, R. and Klein, J. P. (2007) Two-sample tests of the equality of two cumulative incidence functions. *Comput. Statist. Data Anal.* **51**, 4269–4281.

Pepe, M. S. (1991) Inference for events with dependent risks in multiple endpoint studies. *J. Amer. Statist. Assoc.* **86**, 770–778.

See Also

`cif` and `plot.cif` for estimation and plotting of CIFs

`cif2.ks`, `cif2.logrank` and `cif2.neyman` for other two-sample tests

Examples

```
## bone marrow transplant data
data(bmt1)

## compare CIFs for cause 1 (relapse)
cif2.int(Survcomp(bmt1$time, bmt1$event), bmt1$donor, cause = 1)

## compare CIFs for cause 2 (death in remission)
cif2.int(Survcomp(bmt1$time, bmt1$event), bmt1$donor, cause = 2)
```

cif2.ks	<i>Kolmogorov–Smirnov Two-Sample Test for Cumulative Incidence Functions</i>
---------	--

Description

Compares cumulative incidence functions (CIF) for one failure cause in two samples of censored competing risks data using the Kolmogorov–Smirnov-type test.

Usage

```
cif2.ks(x, group, cause = 1, nsim = 2000, nsim.plot = 50)
```

Arguments

x	a "Survcomp" object, as returned by the Survcomp function.
group	a vector indicating to which group each observation belongs. May contain values 1 and 2 only.
cause	For which cause of failure should the CIFs be compared?
nsim	the number of simulations to approximate the p -value. Must be positive.
nsim.plot	the number of simulated paths of the test process to be returned (for possible plotting). Must be at most nsim.

Details

The test compares cumulative incidence functions $F_1(t, k)$, $F_2(t, k)$ for a particular failure cause k .

The test statistic is the maximum absolute difference of the two cumulative incidence functions. Its asymptotic distribution is complicated, therefore the martingale-based simulation approximation is employed. See Lin (1997).

Value

A list with class attributes "cif2.int" and "lwy.test", with components:

stat	the test statistic.
pval.sim	the simulation based p -value.
test.process	the test process (difference of the two CIFs).
test.process.sim	simulated paths of the test process (a matrix with <code>nsim.plot</code> columns).
time	sorted times.

Further components are `cause`, `nsim`, `nsim.plot`, the same as on input.

Author(s)

David Kraus (<http://www.davidkraus.net/>)

References

Lin, D. Y. (1997) Non-parametric inference for cumulative incidence functions in competing risks studies. *Stat. Med.* **16**, 901–910.

See Also

See the `plot` method inherited from the class "lwy.test".

See `cif` and `plot.cif` for estimation and plotting of CIFs, `cif2.int`, `cif2.logrank` and `cif2.neyman` for other two-sample tests.

Examples

```
## bone marrow transplant data
data(bmt1)

## compare CIFs for cause 1 (relapse)
## print results
print(a <- cif2.ks(Survcomp(bmt1$time, bmt1$event), bmt1$donor,
  cause = 1))
## plot the test process and simulated paths
plot(a)

## compare CIFs for cause 2 (death in remission)
## print results
print(a <- cif2.ks(Survcomp(bmt1$time, bmt1$event), bmt1$donor,
  cause = 2))
## plot the test process and simulated paths
plot(a)
```

cif2.logrank

*Two-Sample Logrank-Type Test for Cumulative Incidence Functions***Description**

Compares cumulative incidence functions (CIF) for one failure cause in two samples of censored competing risks data using Gray's G^ρ -weighted logrank-type test based on subdistribution hazards corresponding to CIFs.

Usage

```
cif2.logrank(x, group, cause = 1, rho = 0)
```

Arguments

x	a "Survcomp" object, as returned by the <code>Survcomp</code> function.
group	a vector indicating to which group each observation belongs. May contain values 1 and 2 only.
cause	For which cause of failure should the CIFs be compared?
rho	the parameter (exponent) of the weight.

Details

The test compares cumulative incidence functions $F_1(t, k)$, $F_2(t, k)$ for a particular failure cause k . This test was proposed by Gray (1988). The statistic is similar to the weighted logrank test, but instead of ordinary hazards this test compares subdistribution hazards, which is equivalent to comparing CIFs (due to the one-to-one relationship between subdistribution hazards and CIFs). The G^ρ weight is of the form $(1 - \hat{F}_0(t, k))^\rho$, where $\hat{F}_0(t, k)$ is a pooled sample estimate of the cause k cumulative incidence curve.

This logrank-type test compares subdistribution hazards, and should not be confused with the ordinary logrank test applied to comparing cause-specific hazards. Hypotheses on subdistribution hazards (or CIFs) and cause-specific hazards are not equivalent.

Value

A list of class "cif2.logrank" with components:

stat	the test statistic.
pval	the p -value based on the asymptotic normality.

Further components are `cause` and `rho`, the same as on input.

Author(s)

David Kraus (<http://www.davidkraus.net/>)

References

Gray, R. J. (1988) A class of k -sample tests for comparing the cumulative incidence of a competing risk. *Ann. Statist.* **16**, 1141–1154.

See Also

`cif` and `plot.cif` for estimation and plotting of CIFs, `cif2.ks`, `cif2.int` and `cif2.neyman` for other two-sample tests.

Examples

```
## bone marrow transplant data
data(bmt1)

## compare CIFs for cause 1 (relapse)
cif2.logrank(Survcomp(bmt1$time, bmt1$event), bmt1$donor, cause = 1)

## compare CIFs for cause 2 (death in remission)
cif2.logrank(Survcomp(bmt1$time, bmt1$event), bmt1$donor, cause = 2)
```

cif2.neyman	<i>Two-Sample Neyman's Smooth Test for Cumulative Incidence Functions</i>
-------------	---

Description

Compares cumulative incidence functions (CIF) for one failure cause in two samples of censored competing risks data using (possibly data-driven) Neyman's smooth test.

Usage

```
cif2.neyman(x, group, cause = 1, data.driven = FALSE,
            d = ifelse(data.driven, 5, 3), basis = "legendre",
            choltol = 1e-07)

## S3 method for class 'cif2.neyman':
summary(object, ...)
```

Arguments

<code>x</code>	a "Survcomp" object, as returned by the <code>Survcomp</code> function.
<code>group</code>	a vector indicating to which group each observation belongs. May contain values 1 and 2 only.
<code>cause</code>	For which cause of failure should the CIFs be compared?
<code>data.driven</code>	Should the test be data-driven?
<code>d</code>	the number of basis functions for the test with fixed dimension, the maximum number of basis functions for the data-driven test.

basis	the basis of functions. Possible values are "legendre" for Legendre polynomials and "cos" for cosines.
choltol	a tolerance parameter for the Cholesky decomposition.
object	an object of class "cif2.neyman", as returned by the function <code>cif2.neyman</code> .
...	further parameters for printing.

Details

The test compares cumulative incidence functions $F_1(t, k)$, $F_2(t, k)$ for a particular failure cause k . Neyman-type smooth tests are based on embedding the null hypothesis in a d -dimensional alternative. The embedding is here formulated in terms of subdistribution hazards derived from CIFs. The log-ratio of subdistribution hazards is expressed as a combination of d basis functions (Legendre polynomials or cosines) in transformed time, and their significance is tested by a score test. See Kraus (2007b) for details. The quadratic test statistic is asymptotically chi-square distributed with d degrees of freedom.

A data-driven choice of the number of basis functions is possible. The selection is based on a Schwarz-type criterion which is the maximiser of penalised score statistics for dimensions $1, \dots, d$. For the p -value of the data-driven test a two-term approximation is used (see Kraus (2007a), eq. (12)), as the asymptotic chi-square with 1 d.f. is inaccurate.

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

Value

`cif2.neyman` returns a list with class attributes "cif2.neyman" and "neyman.test". Its main components are:

<code>stat</code>	the test statistic.
<code>pval</code>	the p -value (based on the chi-square distribution for the fixed-dimension test and on the two-term approximation for the data-driven test).
<code>stats, stats.penal</code>	statistics and penalised statistics for dimensions $1, \dots, d$ (only for data-driven tests).
<code>S.dim</code>	the selected dimension (only for data-driven tests).

Most input parameters and some further components are included.

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) Smooth tests of equality of cumulative incidence functions in two samples. Research Report 2197, Institute of Information Theory and Automation, Prague. Available at <http://www.davidkraus.net/surv2sample/>.

See Also

`cif` and `plot.cif` for estimation and plotting of CIFs, `cif2.ks`, `cif2.int` and `cif2.logrank` for other two-sample tests.

Examples

```
## bone marrow transplant data
data(bmt1)

## compare CIFs for cause 1 (relapse)
## test with fixed dimension
cif2.neyman(Survcomp(bmt1$time, bmt1$event), bmt1$donor, cause = 1,
  data.driven = FALSE)
## data-driven test
print(a <- cif2.neyman(Survcomp(bmt1$time, bmt1$event), bmt1$donor,
  cause = 1, data.driven = TRUE))
## print details on the selection procedure
summary(a)

## compare CIFs for cause 2 (death in remission)
cif2.neyman(Survcomp(bmt1$time, bmt1$event), bmt1$donor, cause = 2)
```

gastric

Gastric Cancer Data

Description

Survival data from a trial comparing chemotherapy versus combined chemotherapy plus radiotherapy in the treatment of gastric cancer.

Usage

```
data(gastric)
```

Format

A data frame with 90 observations (45 in each treatment group) with the following 3 variables:

time survival or censoring time, in days

event 1 = death, 0 = censoring

treatment 1 = chemotherapy, 2 = chemotherapy plus radiotherapy

Source

Stablein, D. M. and Koutrouvelis, I. A. (1985) A two-sample test sensitive to crossing hazards in uncensored and singly censored data. *Biometrics* **41**, 643–652. (Page 649)

Examples

```
data(gastric)
plot(survfit(Surv(time, event) ~ treatment, data = gastric))
```

hepatitis

Chronic Active Hepatitis Data

Description

Survival data from a randomised clinical trial of the drug prednisolone for chronic active hepatitis.

Usage

```
data(hepatitis)
```

Format

A data frame with 44 observations, with the following 3 variables:

treatment treatment type (1 = drug prednisolone, 2 = untreated control group)

time time following admission to the trial, in months

status censoring status (1 = death, 0 = censored)

Source

Collett, D. (2003) *Modelling Survival Data in Medical Research. Second Edition.* Chapman & Hall/CRC. (Appendix D.1, p. 363)

http://www.crcpress.com/e_products/downloads/webdownload/C3251/C3251.zip

Examples

```
data(hepatitis)
plot(survfit(Surv(time, status) ~ treatment, data = hepatitis))
```

Description

Plot cumulative incidence functions for one or more samples of censored data with several competing risks (types of failure).

Usage

```
## S3 method for class 'cif':
plot(x, by = "group", aggreg.cif = TRUE, orient = "land",
     lwds = 1, cols = 1, ltys = if ((by == "cause") || (by == "c"))
     rep(1:6, len = x$ngroups) else 1, xlab = "", ylab = "",
     ylim, mfrow, mfcoll, mains, ...)
```

Arguments

<code>x</code>	an object of class "cif", as returned by <code>cif</code> .
<code>by</code>	If "group" or "g", for every group one plot containing all cumulative incidence functions is produced. If "cause" or "c", for every cause one plot containing CIFs for all groups is produced.
<code>aggreg.cif</code>	logical. Used only when curves are plotted by group. If <code>aggreg.cif</code> is TRUE, a summary plot with aggregate CIFs in each group is produced. This means that in each plot the lowest curve is the CIF for cause 1, the second lowest is the sum of CIFs for causes 1 and 2, etc. The area above the top curve is the disease-free survival.
<code>orient</code>	If "land", multiple plots are arranged to give a landscape plot. Otherwise a portrait orientation is assumed. <code>orient</code> is ignored, if <code>mfrow</code> or <code>mfcoll</code> is present.
<code>lwds, cols, ltys</code>	vectors of length equal to the number of curves in plots (number of groups or number of causes). These give line widths, colours and line types for each curve.
<code>xlab, ylab</code>	labels for axes.
<code>ylim</code>	limits on the vertical axis. If missing, determined automatically.
<code>mfrow, mfcoll</code>	parameters determining the arrangement of plots, passed to <code>par</code> .
<code>mains</code>	the main title(s) for plots. If missing, titles for plots are automatically produced. If <code>mains</code> is present, it must be of length equal to the number of plots or 1 (in which case it is replicated).
<code>...</code>	other standard parameters for plotting.

Author(s)

David Kraus (<http://www.davidkraus.net/>)

See Also[cif](#)**Examples**

```
## bone marrow transplant data
data(bmt1)

## plot aggregate cumulative incidence functions for each
## donor type, i.e., plotting by groups, to see the probability
## structure within groups
plot(cif(Survcomp(bmt1$time, bmt1$event), bmt1$donor))

## plot CIFs by causes, to compare donor type effects on each
## cause of failure
plot(cif(Survcomp(bmt1$time, bmt1$event), bmt1$donor), by = "cause")
```

plot.lwy.test

*Plot an Observed Test Process and Simulated Paths***Description**

This function plots an observed test process along with simulated paths. Such a plot may be used to visually assess the hypothesis.

Usage

```
## S3 method for class 'lwy.test':
plot(x, lwds = c(3, 1), cols = c("black", "grey"),
     ltys = c(1, 1), ...)
```

Arguments

x	an object of class "lwy.test".
lwds	a vector of length 2, giving the line width for the observed process and for simulated processes.
cols	a vector of length 2, giving the line colour for the observed process and for simulated processes.
ltys	a vector of length 2, giving the line type for the observed process and for simulated processes.
...	other standard parameters for plotting.

Details

In survival analysis, p -values of Kolmogorov–Smirnov and other tests based on a test process may be approximated by the martingale-based simulation technique originally proposed by Lin, Wei and Ying (1993). By plotting the observed test process along with a suitable number of simulated paths, one may visually assess the validity of the hypothesis.

This function is a general plot method for such tests. Results of such tests are objects with class attribute "lwy.test", at least containing sorted times in `x$time`, the observed process in `x$test.process` and a number (`x$nsim.plot`) of simulated processes in `x$test.process.sim`. Results of functions listed in ‘See Also’ below are handled by this plot method.

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

[surv2.ks](#), [cif2.ks](#), [proprate2.ks](#)

Examples

```
## gastric cancer data
data(gastric)

## Kolmogorov--Smirnov test of equal survival distributions
## test process = difference of Nelson--Aalen estimates
## plot the observed test process and simulated paths
plot(surv2.ks(Surv(gastric$time, gastric$event),
             gastric$treatment, approx="mart"))
```

plot.proprate2.fit *Plot the Two-Sample Proportional Rate Model for Censored Data*

Description

This function plots estimates of cumulative rates (cumulative hazards or odds functions) for two samples of censored data. It may plot both separate estimates from the two samples and estimates based on the proportional rate model.

Usage

```
## S3 method for class 'proprate2.fit':
plot(x, log.transform = FALSE, diff = FALSE, lwds = 1, cols = 1,
     ltys, ...)
```

Arguments

`x` a "proprate2.fit" object, as returned by [proprate2](#).

`log.transform` logical. Should the logarithms of cumulative rates be plotted?

`diff` logical. Instead of two curves, should the difference of their logarithms be plotted?

`lwds, cols, ltys` vectors of length equal to the number of curves in plots (4 if `diff` is FALSE, 2 if TRUE). These give line widths, colours and line types for each curve. If of length 1, the value is replicated.

`...` further plotting parameters.

Details

If `diff` is FALSE, four curves are plotted (two individual sample estimates and two model based estimates). In this case, `ltys` defaults to `c(1, 1, 2, 2)`. If `diff` is TRUE, the function plots their differences. Then `ltys` defaults to `c(1, 1)`. To omit a curve, set the corresponding component of `ltys` to 0.

Using these plots one may visually assess the validity of the proportional rate assumption.

Author(s)

David Kraus (<http://www.davidkraus.net/>)

See Also

[proprate2](#) for estimation

[proprate2.neyman](#), [proprate2.ks](#), [proprate2.gs](#) for tests of the proportional rate assumption

Examples

```
## chronic active hepatitis data
data(hepatitis)

## fit the proportional odds model
fit = with(hepatitis, proprate2(Surv(time, status), treatment,
                               model = 1))

## plot model-based and model-free estimates of odds functions
plot(fit)
## their logarithms
plot(fit, log.transform = TRUE)
## differences of log-functions
plot(fit, diff = TRUE)
```

proprate2 *Fitting the Two-Sample Proportional Rate Transformation Model for Censored Data*

Description

proprate2 estimates the two-sample proportional rate transformation model (proportional hazards, proportional odds) for censored data using the simplified partial likelihood.

Usage

```
proprate2(x, group, model = 0, beta.init = 0, maxiter = 20,
          eps = 1e-09)
```

Arguments

x	a "Surv" object, as returned by the Surv function.
group	a vector indicating to which group each observation belongs. May contain values 1 and 2 only.
model	the type of model. Possible values are 0 for proportional hazards, 1 for proportional odds.
beta.init	the initial parameter value for iteration.
maxiter	the maximum number of iterations.
eps	the convergence tolerance parameter. The convergence criterion is $ (l-l_{old})/l < \text{eps}$.

Details

This function fits the proportional rate model for two samples of censored survival data. Currently two most important models are implemented: proportional hazards and proportional odds. The estimation procedure is based on a two-sample simplification of the partial for the two-sample situation, see Bagdonavicius and Nikulin (2000). (For proportional hazards, this method is the usual partial likelihood.)

Value

A list of class "proprate2.fit" with main components:

beta	the estimate.
var	its variance.
G0	the cumulative baseline rate (at times time).
time	sorted times.
iter	the number of iterations used.
converged	logical. Did the iterations (appear to) converge?
loglik.init	the simplified partial likelihood at the initial value of the parameter.

loglik the simplified partial likelihood at the estimate.
 d11 the derivative of the score.
 sigma11 variance of the score (for proportional hazards sigma11 equals d11).
 G1, G2 cumulative transformation rates computed separately in the two groups (both of length n , at times `time`).

Author(s)

David Kraus (<http://www.davidkraus.net/>)

References

Bagdonavicius, V. and Nikulin, M. (2000) On goodness-of-fit for the linear transformation and frailty models. *Statist. Probab. Lett.* **47**, 177–188.

Kraus, D. (2007) Checking proportional rates in the two-sample transformation model. Research Report 2203, Institute of Information Theory and Automation, Prague. Available at <http://www.davidkraus.net/surv2sample/>.

See Also

There is a `plot` method for objects returned by `proprate2`.

See `proprate2.neyman`, `proprate2.ks`, `proprate2.gs` for tests of the proportional rate assumption.

Examples

```
## chronic active hepatitis data
data(hepatitis)

## fit the proportional odds model
fit = with(hepatitis, proprate2(Surv(time, status), treatment,
  model = 1))
fit

## plot model-based and model-free estimates of odds functions
plot(fit)
```

proprate2.gs *Gill–Schumacher Test of Proportional Rates in Two Samples of Censored Data*

Description

Checks the assumption of proportional rates (proportional hazards, proportional odds) in two samples of right-censored data using the Gill–Schumacher test based on the comparison of two estimates of the rate ratio.

Usage

```
proprate2.gs(x, group, model = 0, weight1 = "logrank",
             weight2 = "prentice")
```

Arguments

<code>x</code>	a "Surv" object, as returned by the <code>Surv</code> function.
<code>group</code>	a vector indicating to which group each observation belongs. May contain values 1 and 2 only.
<code>model</code>	the type of model. Possible values are 0 for proportional hazards, 1 for proportional odds.
<code>weight1, weight2</code>	weight functions for the ratio estimates. Possible values are "logrank", "prentice", "gehan".

Details

This function performs the Gill–Schumacher test of the hypothesis that transformation rates (currently hazard rates or odds functions) are proportional (their ratio is constant in time) in two samples of censored survival data.

The test was proposed by Gill and Schumacher (1987) for proportional hazards, see Kraus (2007) for its extension to proportional transformation rates. The test statistic compares two weighted estimates of the ratio of rates. Possible weights are of the logrank, Prentice–Wilcoxon or Gehan type.

Value

A "proprate2.gs" object with components:

<code>stat</code>	the test statistic.
<code>pval</code>	the p -value.
<code>eta1, eta2</code>	weighted estimates of the rate ratio.

Some of input parameters are included too.

Author(s)

David Kraus (<http://www.davidkraus.net/>)

References

Gill, R. and Schumacher, M. (1987) A simple test of the proportional hazards assumption. *Biometrika* **74**, 289–300.

Kraus, D. (2007) Checking proportional rates in the two-sample transformation model. Research Report 2203, Institute of Information Theory and Automation, Prague. Available at <http://www.davidkraus.net/surv2sample/>.

See Also

[proprate2.neyman](#), [proprate2.ks](#) for other tests of the proportional rate assumption
[proprate2](#) for the simplified partial likelihood estimation

Examples

```
## chronic active hepatitis data
data(hepatitis)

## perform the Gill--Schumacher test of proportional odds
proprate2.gs(Surv(hepatitis$time, hepatitis$status),
             hepatitis$treatment, model = 1)
```

proprate2.ks	<i>Kolmogorov–Smirnov Test of Proportional Rates in Two Samples of Censored Data</i>
--------------	--

Description

Checks the assumption of proportional rates (proportional hazards, proportional odds) in two samples of right-censored data using the Kolmogorov–Smirnov test based on the simplified partial likelihood score process.

Usage

```
proprate2.ks(x, group, model = 0, nsim = 2000, nsim.plot = 50,
             beta.init = 0, maxiter = 20, eps = 1e-09)
```

Arguments

x	a "Surv" object, as returned by the Surv function.
group	a vector indicating to which group each observation belongs. May contain values 1 and 2 only.
model	the type of model. Possible values are 0 for proportional hazards, 1 for proportional odds.
nsim	the number of simulations to approximate the p -value. Must be positive.
nsim.plot	the number of simulated paths of the test process to be returned (for possible plotting). Must be at most <code>nsim</code> .
beta.init	the initial parameter value for iteration in the simplified partial likelihood estimation.
maxiter	the maximum number of iterations.
eps	the convergence tolerance parameter. The convergence criterion is $ (l-l_{old})/l < \text{eps}$.

Details

This function tests the hypothesis that transformation rates (currently hazard rates or odds functions) are proportional (their ratio is constant in time) in two samples of censored survival data.

The proportional rate model is estimated by a two-sample simplification of the partial likelihood. The test then uses the Kolmogorov–Smirnov supremum statistic based on the simplified partial likelihood score process. The p -value is computed using the martingale simulation technique.

Value

A list of class "proprate2.ks" and "lwy.test", with main components:

```
stat          the test statistic.
pval.sim      the simulation based  $p$ -value.
test.process  the test process.
test.process.sim
              simulated paths of the test process (a matrix with nsim.plot columns).
time         sorted times.
```

Some of input parameters and further components are included too.

Author(s)

David Kraus (<http://www.davidkraus.net/>)

References

Bagdonavicius, V. and Nikulin, M. (2000) On goodness-of-fit for the linear transformation and frailty models. *Statist. Probab. Lett.* **47**, 177–188.

See Also

[plot](#) method inherited from the class "lwy.test"
[proprate2.neyman](#), [proprate2.gs](#) for other tests of the proportional rate assumption
[proprate2](#) for estimation

Examples

```
## chronic active hepatitis data
data(hepatitis)

## perform the Komogorov--Smirnov test of proportional odds
a = proprate2.ks(Surv(hepatitis$time, hepatitis$status),
  hepatitis$treatment, model = 1)
a
## plot the test process and simulated paths
plot(a)
```

proprate2.neyman *Neyman's Smooth Test of Proportional Rates in Two Samples of Censored Data*

Description

Checks the assumption of proportional rates (proportional hazards, proportional odds) in two samples of right-censored data using (possibly data-driven) Neyman's smooth test.

Usage

```
proprate2.neyman(x, group, model = 0, data.driven = TRUE,
                d = ifelse(data.driven, 5, 3),
                basis = "legendre", time.transf = "F",
                beta.init = 0, maxiter = 20, eps = 1e-09,
                choltol = 1e-07)

## S3 method for class 'proprate2.neyman':
summary(object, ...)
```

Arguments

x	a "Surv" object, as returned by the Surv function.
group	a vector indicating to which group each observation belongs. May contain values 1 and 2 only.
model	the type of model. Possible values are 0 for proportional hazards, 1 for proportional odds.
data.driven	Should the test be data-driven?
d	the number of basis functions for the test with fixed dimension, the maximum number of basis functions for the data-driven test.
basis	the basis of functions. Possible values are "legendre" for Legendre polynomials and "cos" for cosines.
time.transf	the time transformation for basis functions. Possible values are "F" for the distribution function ($F(t)/F(\tau)$) estimated from the pooled sample (recommended), "A" for the cumulative hazard ($A(t)/A(\tau)$) and "I" for no transformation (the linear transformation t/τ).
beta.init	the initial parameter value for iteration in the simplified partial likelihood estimation.
maxiter	the maximum number of iterations.
eps	the convergence tolerance parameter. The convergence criterion is $ (l-l_{\text{old}})/l < \text{eps}$.
choltol	a tolerance parameter for the Cholesky decomposition.
object	an object of class "proprate2.neyman", as returned by <code>proprate2.neyman</code> .
...	further parameters for printing.

Details

This function tests the hypothesis that transformation rates (currently hazard rates or odds functions) are proportional (their ratio is constant in time) in two samples of censored survival data.

The proportional rate model is estimated by a two-sample simplification of the partial likelihood. Then Neyman's smooth test of fit is performed. In general, Neyman's smooth tests are based on embedding the null hypothesis in a d -dimensional alternative. Here it consists of expressing the logarithm of the possibly time-varying ratio of rates as a linear combination of d basis functions (Legendre polynomials or cosines) in transformed time. Their significance is tested by a score test. The score is derived from the simplified partial likelihood. See Kraus (2007a) for details. The quadratic test statistic is asymptotically chi-square distributed with d degrees of freedom.

A data-driven choice of the number of basis functions is possible. The selection is based on a Schwarz-type criterion which is the maximiser of penalised score statistics for dimensions $1, \dots, d$. For the p -value of the data-driven test a two-term approximation is used (see Kraus (2007b), eq. (12)), as the asymptotic chi-square with 1 d.f. is inaccurate.

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

Value

`proprate2.neyman` returns a list of class `"proprate2.neyman"` and `"neyman.test"`. Its main components are:

<code>stat</code>	the test statistic.
<code>pval</code>	the p -value (based on the chi-square distribution for the fixed-dimension test and on the two-term approximation for the data-driven test).
<code>stats, stats.penal</code>	statistics and penalised statistics for dimensions $1, \dots, d$ (only for data-driven tests).
<code>S.dim</code>	the selected dimension (only for data-driven tests).

Most input parameters and some further components are included.

Author(s)

David Kraus (<http://www.davidkraus.net/>)

References

Bagdonavicius, V. and Nikulin, M. (2000) On goodness-of-fit for the linear transformation and frailty models. *Statist. Probab. Lett.* **47**, 177–188.

Kraus, D. (2007a) Checking proportional rates in the two-sample transformation model. Research Report 2203, Institute of Information Theory and Automation, Prague. Available at <http://www.davidkraus.net/surv2sample/>.

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

See Also

[proprate2.ks](#), [proprate2.gs](#) for other tests of the proportional rate assumption
[proprate2](#) for estimation

Examples

```
## chronic active hepatitis data
data(hepatitis)

## Neyman's test of proportional odds
## test with fixed dimension
proprate2.ks(Surv(hepatitis$time, hepatitis$status),
             hepatitis$treatment, model = 1, data.driven = FALSE)
## data-driven test
print(a <- proprate2.ks(Surv(hepatitis$time, hepatitis$status),
                       hepatitis$treatment, model = 1, data.driven = TRUE))
## details of the selection procedure
summary(a)
```

surv2.ks

*Two-Sample Kolmogorov–Smirnov, Cramer–von Mises and
Anderson–Darling Test for Censored Data*

Description

Performs the Kolmogorov–Smirnov, Cramer–von Mises and Anderson–Darling test to compare the distribution of survival times in two samples of censored data.

Usage

```
surv2.ks(x, group, process = "w", approx = "lwy", nsim = 2000,
         nsim.plot = 50)
```

Arguments

<code>x</code>	a "Surv" object, as returned by the Surv function.
<code>group</code>	a vector indicating to which group each observation belongs. May contain values 1 and 2 only.
<code>process</code>	the type of the test process. Possible values are "w" for the difference of Nelson–Aalen estimates (asymptotically a Brownian motion, i.e., Wiener process), "b" for a transformation of this process (asymptotically a Brownian bridge).
<code>approx</code>	the method of approximating the distribution of test statistics. Possible values are "lwy" or "mart" for the martingale-based simulation, "perm" for permutations, "boot" for the bootstrap.
<code>nsim</code>	the number of simulations (martingale simulations, permutations or bootstrap samples).

`nsim.plot` the number of simulated paths of the test process to be returned (for possible plotting). Must be at most `nsim`.

Details

The function implements tests based on functionals of the logrank process $U(t)$ (which is the process of logrank statistics computed from observations in $(0, t)$, see Section 7.5 of Fleming and Harrington (1991)). This process (properly normalised) is asymptotically a Brownian motion in transformed time. If `process` is "w", the supremum (KS) and integral (CM, AD) test statistics are computed from this process. If `process` is "b", the tests are instead based on the process $U(t)(1 + \hat{v}(t)/\hat{v}(\tau))^{-1}$, which is asymptotically a Brownian bridge in transformed time.

Value

A list with class attributes "surv2.int" and "lwy.test", with main components:

<code>stat.ks</code>	the Kolmogorov–Smirnov statistic.
<code>pval.ks</code>	the corresponding p -value.
<code>pval.ks.asympt</code>	the asymptotic p -value.
<code>stat.cm</code>	the Cramer–von Mises statistic.
<code>pval.cm</code>	the corresponding p -value.
<code>stat.ad</code>	the Anderson–Darling statistic.
<code>pval.ad</code>	the corresponding p -value.
<code>time</code>	sorted times.
<code>test.process</code>	the test process.
<code>test.process.sim</code>	simulated paths of the test process (a matrix with <code>nsim.plot</code> columns).

Some of input arguments are also contained in the output.

Author(s)

David Kraus (<http://www.davidkraus.net/>)

References

Andersen, P. K., Borgan, O., Gill, R. D. and Keiding, N. (1993) *Statistical Models Based on Counting Processes*. Springer, New York.

Fleming, T. R. and Harrington, D. P. (1991) *Counting Processes and Survival Analysis*. Wiley, New York.

See Also

See the `plot` method inherited from the class "lwy.test".

See also `surv2.neyman`, `surv2.logrank`, `survdiff`, `survfit`.

Examples

```
## gastric cancer data
data(gastric)

## print results
print(a <- surv2.ks(Surv(gastric$time, gastric$event),
  gastric$treatment))
## plot the test process and simulated paths
plot(a)
```

surv2.logrank

*Two-Sample Weighted Logrank Tests and Their Combinations***Description**

Compares the distribution of survival times in two samples of censored data using the weighted logrank test with the $G^{\rho,\gamma}$ class of weights, or combinations (maximum or weighted sum) of several weighted logrank statistics.

Usage

```
surv2.logrank(x, group, rho.gamma = c(0, 0), comb, sum.weights,
  approx = "perm", nsim = 2000, choltol = 1e-07)
```

Arguments

x	a "Surv" object, as returned by the <code>Surv</code> function.
group	a vector indicating to which group each observation belongs. May contain values 1 and 2 only.
rho.gamma	parameters (ρ, γ) of the weight. This must be a vector of length 2 or a list of vectors of length 2.
comb	the method of combining test statistics. Possible values are "max" and "sum". comb is ignored, if there is only one set of parameters in rho.gamma, and defaults to "max" otherwise.
sum.weights	weights for the weighted sum in the "sum" combination method. Defaults to equal weights for all statistics.
approx	the method of approximating the distribution of the test statistic. Possible values are "perm" for permutations, "boot" for the bootstrap, "asympt" for asymptotics.
nsim	the number of simulations. This means the number of permutations or bootstrap samples when approx is "perm" or "boot". When approx is "asympt", nsim is the number of simulations to approximate the asymptotic distribution (only needed for combination methods).
choltol	a tolerance parameter for the Cholesky decomposition.

Details

The logrank test uses $G^{\rho,\gamma}$ weights defined as $\hat{S}(t)^\rho(1 - \hat{S}(t))^\gamma$, where $\hat{S}(t)$ is the Kaplan–Meier estimate computed from the pooled sample.

Combination tests are based on a cluster of several weighted logrank statistics with different parameters (ρ, γ) . The maximum statistic uses the maximum of absolute values of standardised statistics. The sum statistic uses the weighted sum of absolute values of standardised statistics. See Chapter 7 of Fleming and Harrington (1991) for details.

Value

A list of class "surv2.logrank" with components:

stat	the test statistic.
pval	the p -value.
rho.gamma	as on input (for a test using a single statistic).
stats	a matrix containing parameters of weights, corresponding statistics and p -values (for a combination test).

Further components include approx, nsim, comb, sum.weights from input.

Author(s)

David Kraus (<http://www.davidkraus.net/>)

References

Andersen, P. K., Borgan, O., Gill, R. D. and Keiding, N. (1993) *Statistical Models Based on Counting Processes*. Springer, New York.

Fleming, T. R. and Harrington, D. P. (1991) *Counting Processes and Survival Analysis*. Wiley, New York.

See Also

[surv2.neyman](#), [surv2.ks](#), [survdiff](#), [survfit](#)

Examples

```
## gastric cancer data
data(gastric)

## Prentice--Wilcoxon (G^1) test
surv2.logrank(Surv(gastric$time, gastric$event),
              gastric$treatment, rho.gamma = c(1,0))

## combination of G(0,0), G(1,0), G(1,1) statistics
## maximum test
print(a <- surv2.logrank(Surv(gastric$time, gastric$event),
                        gastric$treatment, rho.gamma = list(c(0,0), c(1,0), c(1,1)),
                        comb = "max"))
```

```
## print individual statistics
a$stats
```

```
surv2.neyman
```

Two-Sample Neyman's Smooth Test for Censored Data

Description

Compares survival distributions in two samples of censored data using (possibly data-driven) Neyman's smooth test.

Usage

```
surv2.neyman(x, group, data.driven = FALSE, subsets = "nested",
             d = ifelse(data.driven, 5, 3), d0 = 0,
             basis = "legendre", time.transf = "F",
             approx = "perm", nsim = 2000, choltol = 1e-07)

## S3 method for class 'surv2.neyman':
summary(object, ...)
```

Arguments

<code>x</code>	a "Surv" object, as returned by the Surv function.
<code>group</code>	a vector indicating to which group each observation belongs. May contain values 1 and 2 only.
<code>data.driven</code>	Should the test be data-driven?
<code>subsets</code>	the class of subsets of basis functions among which the data-driven test selects. Possible values are "nested" and "all".
<code>d</code>	the number of basis functions for the test with fixed dimension, the maximum number of basis functions for the data-driven test.
<code>d0</code>	the number of high-priority functions for the data-driven test. The selection rule selects among subsets containing basis functions 1,...,d0. For nested subsets, d0 equal to 0 or 1 is equivalent. For all subsets, d0 equal to 0 means that there is no high-priority function and any nonempty subset may be selected.
<code>basis</code>	the basis of functions. Possible values are "legendre" for Legendre polynomials and "cos" for cosines.
<code>time.transf</code>	the time transformation for basis functions. Possible values are "F" for the distribution function ($F(t)/F(\tau)$) (recommended), "A" for the cumulative hazard ($A(t)/A(\tau)$) and "I" for no transformation (the linear transformation t/τ).
<code>approx</code>	the method of approximating the distribution of the test statistic. Possible values are "perm" for permutations, "boot" for the bootstrap, "asympt" for asymptotics.

<code>nsim</code>	the number of simulations. This means the number of permutations or bootstrap samples when <code>approx</code> is "perm" or "boot". When <code>approx</code> is "asympt", <code>nsim</code> is the number of simulations to approximate the asymptotic distribution (only needed for the data-driven test with all subsets and <code>d0</code> equal to 0).
<code>choltol</code>	a tolerance parameter for the Cholesky decomposition.
<code>object</code>	an object of class "surv2.neyman", as returned by the function <code>surv2.neyman</code> .
<code>...</code>	further parameters for printing.

Details

In general, Neyman's smooth tests are based on embedding the null hypothesis in a d -dimensional alternative. The embedding is here formulated in terms of hazard functions. The logarithm of the hazard ratio is expressed as a combination of d basis functions (Legendre polynomials or cosines) in transformed time, and their significance is tested by a score test. See Kraus (2007a) for details. The quadratic test statistic is asymptotically chi-square distributed with d degrees of freedom.

A data-driven choice of basis functions is possible. The selection is based on a Schwarz-type criterion which is the maximiser of penalised score statistics for over a class of nonempty subsets of $\{1, \dots, d\}$. Either nested subsets with increasing dimension or all subsets may be used. By choosing $d_0 > 0$ one requires that functions with indexes $\{1, \dots, d_0\}$ be always included in subsets.

If all subsets are used with $d_0 = 0$, the data-driven test statistic is asymptotically distributed as the maximum of (generally dependent) chi-square variables with 1 d.f. This asymptotic approximation is accurate. In other cases, the statistic is asymptotically chi-square distributed with $d^* = \max(1, d_0)$ degrees of freedom. For nested subsets with $d^* = 1$ a two-term approximation may be used (see Kraus (2007b), eq. (12)). Otherwise the asymptotics is unreliable.

In any case, one may use permutations or the bootstrap.

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

Value

A list of class "surv2.neyman" and "neyman.test", with main components:

<code>stat</code>	the test statistic.
<code>pval</code>	the p -value.
<code>stats, stats.penal</code>	the score statistic and penalised score statistic for each considered subset (only for data-driven tests).
<code>S.dim</code>	the dimension of the selected set (only for data-driven tests).
<code>S.set</code>	the selected set (only for data-driven tests).

Most input parameters and some further components are included.

Author(s)

David Kraus (<http://www.davidkraus.net/>)

References

Kraus, D. (2007a) Adaptive Neyman's smooth tests of homogeneity of two samples of survival data. Research Report 2187, Institute of Information Theory and Automation, Prague. Available at <http://www.davidkraus.net/surv2sample/>.

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

See Also

[surv2.logrank](#), [surv2.ks](#), [survdiff](#), [survfit](#)

Examples

```
## gastric cancer data
data(gastric)

## test with fixed dimension
surv2.neyman(Surv(gastric$time, gastric$event), gastric$treatment,
             data.driven = FALSE)

## data-driven test with nested subsets
## without minimum dimension (i.e., minimum dimension 1)
summary(surv2.neyman(Surv(gastric$time, gastric$event),
                    gastric$treatment, data.driven = TRUE, subsets = "nested"))
## with minimum dimension 3
summary(surv2.neyman(Surv(gastric$time, gastric$event),
                    gastric$treatment, data.driven = TRUE, subsets = "nested",
                    d0 = 3))

## data-driven test with all subsets
## without high-priority functions
summary(surv2.neyman(Surv(gastric$time, gastric$event),
                    gastric$treatment, data.driven = TRUE, subsets = "all"))
## with 2 high-priority functions
summary(surv2.neyman(Surv(gastric$time, gastric$event),
                    gastric$treatment, data.driven = TRUE, subsets = "all",
                    d0 = 2))
```

Survcomp

Competing Risks Survival Data Object

Description

Create an object representing right-censored data with competing risks (types of failure).

Usage

```
Survcomp(time, event)
is.Survcomp(x)
```

Arguments

<code>time</code>	survival times (possibly right-censored).
<code>event</code>	the status indicator, 0 = censored, 1 = dead from cause 1, 2 = dead from cause 2, ... Successive integers 1,2,... must be used for causes.
<code>x</code>	any object.

Details

This is an extension of the standard function `Surv` of the package **survival**. `Surv` allows only death and censoring, while `Survcomp` handles causes of death.

Value

`Survcomp` returns a matrix of class "Survcomp" with two columns (survival times, event types) with attribute "ncauses" containing the number of different causes of death (which is `max(event)`). There is a print method for objects of class "Survcomp", and the function `is.Survcomp` for testing whether an object is of this class.

Author(s)

David Kraus (<http://www.davidkraus.net/>)

See Also

`cif`, `Surv`

Examples

```
## bone marrow transplant data
data(bmt1)

## cause of failure printed in brackets
## censored observations have a "+"
print(a <- Survcomp(bmt1$time, bmt1$event))
is.Survcomp(a)
## what's inside
str(a)
```

Index

*Topic **datasets**

bmt1, 3
gastric, 12
hepatitis, 13

*Topic **package**

surv2sample-package, 2

*Topic **survival**

cif, 4
cif2.int, 5
cif2.ks, 7
cif2.logrank, 8
cif2.neyman, 10
plot.cif, 13
plot.lwy.test, 15
plot.proprate2.fit, 16
proprate2, 17
proprate2.gs, 19
proprate2.ks, 21
proprate2.neyman, 22
surv2.ks, 25
surv2.logrank, 27
surv2.neyman, 29
surv2sample-package, 2
Survcomp, 31

bmt1, 2, 3

cif, 2, 4, 6, 8, 9, 11, 14, 32
cif2.int, 2, 5, 8, 9, 11
cif2.ks, 2, 6, 7, 9, 11, 16
cif2.logrank, 2, 6, 8, 8, 11
cif2.neyman, 2, 6, 8, 9, 10
coef.proprate2.fit (*proprate2*), 17

gastric, 2, 12

hepatitis, 2, 13

is.Survcomp (*Survcomp*), 31

plot, 8, 19, 22, 26

plot.cif, 5, 6, 8, 9, 11, 13
plot.lwy.test, 15
plot.proprate2.fit, 16
proprate2, 2, 16, 17, 17, 20, 22, 24
proprate2.gs, 2, 17, 19, 19, 22, 24
proprate2.ks, 2, 16, 17, 19, 20, 21, 24
proprate2.neyman, 2, 17, 19, 20, 22, 22

summary.cif2.neyman
 (*cif2.neyman*), 10
summary.proprate2.neyman
 (*proprate2.neyman*), 22
summary.surv2.neyman
 (*surv2.neyman*), 29
Surv, 18, 19, 21, 23, 25, 27, 29, 32
surv2.ks, 2, 16, 25, 28, 31
surv2.logrank, 2, 26, 27, 31
surv2.neyman, 2, 26, 28, 29
surv2sample
 (*surv2sample-package*), 2
surv2sample-package, 2
Survcomp, 4, 5, 7, 8, 10, 31
survdiff, 26, 28, 31
survfit, 26, 28, 31