Package ‘bpcp’

October 12, 2022

Type Package
Title Beta Product Confidence Procedure for Right Censored Data
Version 1.4.2
Date 2022-03-11
Author Michael P. Fay [aut, cre], Allyson Mateja [ctb]
Maintainer Michael P. Fay <mfay@niaid.nih.gov>
Depends stats, survival, ggplot2, methods
Description Calculates nonparametric pointwise confidence intervals for the survival distribution for right censored data, and for medians [Fay and Britain <DOI:10.1002/sim.6905>]. Has two-sample tests for dissimilarity (e.g., difference, ratio or odds ratio) in survival at a fixed time, and differences in medians [Fay, Proschan, and Britain <DOI:10.1111/biom.12231>]. Basically, the package gives exact inference methods for one- and two-sample exact inferences for Kaplan-Meier curves (e.g., generalizing Fisher’s exact test to allow for right censoring), which are especially important for latter parts of the survival curve, small sample sizes or heavily censored data. Includes mid-p options.
License GPL (>= 2)
LazyLoad yes
NeedsCompilation no
Repository CRAN
Date/Publication 2022-03-11 23:10:11 UTC

R topics documented:

  bpcp-package ....................................................... 2
  betaMeldTest ...................................................... 4
  bpcp2samp ........................................................... 5
  bpcp2sampControl .................................................. 7
  bpcpControl ......................................................... 8
  fixtdiff ............................................................... 9
  kmci.object ......................................................... 10
  kmciLR.object ..................................................... 11
  kmciLRgroup.object .............................................. 12
**Description**

The package has functions to give several different methods for calculating pointwise confidence intervals for a single survival distribution for right censored data. There is also a two-sample test for dissimilarity (measured by difference, ratio, or odds ratio) between two survival distributions at a fixed point in time.

The recommended confidence interval for a single sample is the beta product confidence procedure (using `bpcp`), and the recommended test for the two-sample test is the melded BPCP test (using `bpcp2samp`).

Other confidence intervals and two sample tests are included in the package primarily to compare them to the recommended ones. (And justify the recommendations).

Also included is a difference in medians test that applies only to non-censored data and is designed to guarantee coverage for all sample sizes (see `mdiffmedian.test`). The test makes no assumptions about the distributions, so that, unlike the Hodges-Lehmann method, tied data are allowed and a shift assumption is not needed.

**Details**

- **Package:** bpcp
- **Type:** Package
- **Version:** 1.4.2
- **Date:** 2022-03-11
- **License:** GPL2
- **LazyLoad:** yes
The most important function for the single sample case is the \texttt{bpcp} which gives confidence intervals for the survival distribution for right censored data with nice small sample properties. That function creates an \texttt{kmc1LR} object which has 6 methods: \texttt{summary} (create a data frame with estimates and confidence intervals), \texttt{plot} (plot Kaplan-Meier with confidence intervals), \texttt{lines} (add confidence intervals to a plot), \texttt{StCI} (pick out survival and confidence interval at specific times), \texttt{median} (pick out median and confidence interval on median), and \texttt{quantile} (pick out any quantile and its confidence interval). A mid-p option for \texttt{bpcp} is now available. It gives closer to nominal coverage than the standard (\texttt{midp=FALSE}) BPCP. For details see Fay et al (2013) on the standard BPCP and Fay and Brittain (2016) on the mid-p option.

For the two-sample test see \texttt{bpcp2samp}. This test for equality reduces to Fisher’s exact test when there is no censoring. When there is censoring, the test is expected to maintain at least nominal coverage. For details see Fay et al (2015).

\textbf{Author(s)}

Michael P. Fay

Maintainer: Michael P. Fay <mfay@niaid.nih.gov>

\textbf{References}


\textbf{See Also}

\texttt{bpcp}

\textbf{Examples}

data(leuk)
## since there are ties at time=6
## and the data are truncated to the nearest integer, use Delta=1
bfit<-bpcp(leuk$time,leuk$status,Delta=1)
## plot Kaplan-Meier and 95 pct Beta Product Confidence Intervals
plot(bfit,xlab="time (weeks)")
## details
summary(bfit)
quantile(bfit)
StCI(bfit,2)
betaMeldTest

**Melded Beta Test**

**Description**

This function gives a two sample melded beta test together with the associated melded confidence intervals. It can be used when the confidence distributions (or upper and lower confidence distributions) for the one sample parameters are beta, and one is interested in either the difference, odds ratio, or ratio of those two one sample parameters. The betaMeldTest is usually called by bpcp2samp, and not called directly by the user.

**Usage**

```r
betaMeldTest(betaParms1, betaParms2, 
nullparm = NULL, 
parmtype = c("difference", "oddsratio", "ratio","cdfratio"), 
conf.level = 0.95, conf.int = TRUE, 
alternative = c("two.sided", "less", "greater"), 
eps = 10^-8, dname = "", estimate1 = NA, estimate2 = NA)
```

**Arguments**

- `betaParms1`: a list of the beta parameters for group 1
- `betaParms2`: a list of the beta parameters for group 2
- `nullparm`: null value of the parameter of interest, default of NULL gives 0 if parmtype='difference' and 1 otherwise
- `parmtype`: parameter type for comparing the survival function of the two groups, either 'difference' 'ratio' 'oddsratio' or 'cdfratio'
- `conf.level`: confidence level, e.g., 0.95
- `conf.int`: logical, calculate confidence interval?
- `alternative`: character, either 'two.sided','less', or 'greater'
- `eps`: small value to make integration tractable
- `dname`: name describing data
- `estimate1`: estimate of mean for beta parameter of group 1 (statistic of htest object)
- `estimate2`: estimate of mean for beta parameter for group 2 (parameter of htest object)

**Details**

If the upper and lower confidence distributions for both samples are described by beta distributions, then you can create a CD test using this function. For example, if you have sample 1 is binomial with x (with 0<x<n) out of n positive responses, then the 100(1-alpha) confidence interval is qbeta(alpha/2, x, n-x+1) and qbeta(1-alpha/2, x+1, n-x). So the lower confidence distribution is beta with parameters a=x and b=n-x+1, and the upper CD is beta with parameters a=x+1 and b=n-x.

See bpcp2samp for a full description of the parmtypes.
bpcp2samp

Value

an object of class 'htest'

Examples

fisher.test(matrix(c(4,5,22),2,2),alternative="greater")

betaMeldTest(
  betaParms1=list(alower=2,blower=22+1,aupper=2+1,bupper=22),
  betaParms2=list(alower=4,blower=5+1,aupper=4+1,bupper=5),
  alternative="greater",parmtype="oddsratio",
  estimate1=2/24,estimate2=4/9)

bpcp2samp

Melded BPCP test

Description

Tests for dissimilarity between two groups in their survival distributions at a fixed point in time. Can operationalize that dissimilarity as 'difference', 'ratio' or 'oddsratio'.

Usage

bpcp2samp(time, status, group, testtime,
  parmtype = c("difference", "oddsratio", "ratio","cdfratio",
  "one.minus.ratio","one.minus.cdfratio"),
  nullparm = NULL,
  alternative = c("two.sided", "less", "greater"),
  conf.level = 0.95,
  midp=FALSE,
  changeGroupOrder=FALSE,
  control = bpcp2sampControl())

Arguments

time time to event for each observation
status status of event time, 1 is observed, 0 is right censored
group group for test, should have two levels, to change order use as factor and change order of levels
testtime fixed time when you want to test for a difference
parmtype parameter type for comparing the survival function of the two groups, either 'difference' 'ratio' 'oddsratio' 'cdfratio' 'one.minus.ratio' or 'one.minus.cdfratio'
nullparm null value of the parameter of interest, default of NULL gives 0 if parmtype='difference' and 1 otherwise
alternative character, either 'two.sided','less', or 'greater'
The melded confidence interval method is a very general procedure to create confidence intervals for the two sample tests by combining one sample confidence intervals. If \( S_1 \) and \( S_2 \) are the survival value at \( \text{testtime} \) from sample 1 (first value of group) and sample 2 (second value of group) respectively, then we can get confidence intervals on the \( S_2 - S_1 \) (\( \text{parmtype} = \text{'Var difference'} \)), \( S_2/S_1 \) (\( \text{parmtype} = \text{'Var ratio'} \)), \((S_2/(1-S_1))/(S_1/(1-S_2))\) (\( \text{parmtype} = \text{'oddsratio'} \)), \((1-S_1)/(1-S_2) = F_1/F_2\) (\( \text{parmtype} = \text{'cdfratio'} \)), \(1-S_2/S_1 \) (\( \text{parmtype} = \text{'one.minus.ratio'} \)), or \(1-(1-S_1)/(1-S_2) = 1-F_1/F_2\) (\( \text{parmtype} = \text{'one.minus.cdfratio'} \)).

The resulting melded CIs appear to guarantee coverage as long as the one sample confidence intervals from which the melded CIs are derived have guaranteed coverage themselves. So since we use the BPCP for the one sample intervals and they appear to guarantee coverage (see Fay, Brittain, and Proschan, 2013), we expect the melded BPCP intervals to have at least nominal coverage. Note that when there is no censoring the melded CIs derived from the one-sample BPCPs, give matching inferences to Fisher’s exact test (i.e., give theoretically identical p-values) when testing the null hypothesis of equality \( S_1 = S_2 \). For details see Fay, Proschan and Brittain (2015).

The original melded CIs focused on combining one sample CIs that guarantee coverage. We can apply the melding to other CIs as well, such as the mid-p style CIs. The mid-p CIs are not designed to guarantee coverage, but are designed to have close to the nominal coverage ‘on average’ over all the possible values of the parameters. The usual p-value is derived from \( \Pr[ \text{see observed data or more extreme under null}] \), while the mid p-value version comes from \((1/2) \Pr[ \text{see obs data}] + \Pr[ \text{see more extreme data}] \). Mid-p CIs come from inverting the test that uses the mid p-value instead of the usual p-value.

**Value**

A list with class "htest" containing the following components:

- **statistic**: estimate of \( S_1 \), survival at testtime for group 1
- **parameter**: estimate of \( S_2 \), survival at testtime for group 2
- **p.value**: p-value for the test
- **conf.int**: a confidence interval for the parameter determined by \( \text{parmtype} \)
- **estimate**: estimate of parameter determined by \( \text{parmtype} \)
- **null.value**: the specified null hypothesized value of the parameter determined by \( \text{parmtype} \)
- **alternative**: type of alternative with respect to the null.value, either ‘two.sided’, ‘greater’ or ‘less’
- **method**: a character string describing the test
- **data.name**: a character string describing the parameter determined by \( \text{parmtype} \)

**Author(s)**

Michael P. Fay
References


Examples

data(leuk2)
# test difference of S(20) values
# S(20)=survival function at 20 weeks
bpcp2samp(leuk2$time,leuk2$status,leuk2$treatment, 20,parmtype="difference")
# test ratio of S(20) in two treatment groups,
bpcp2samp(leuk2$time,leuk2$status,leuk2$treatment, 20,parmtype="ratio")
# change the order of the group variable to get the other ratio
bpcp2samp(leuk2$time,leuk2$status,leuk2$treatment,20,
parmtype="ratio",changeGroupOrder=TRUE)
# estimate treatment effect= 1 - F(20,trt)/F(20,plac),
# where F(20)=1-S(20) = Pr(T <=20) is the cumulative distribution function
# Test whether treatment effect is greater than 30 pct
bpcp2samp(leuk2$time,leuk2$status,leuk2$treatment,20,
parmtype="one.minus.cdfratio",nullparm=0.30,
alternative="greater",
changeGroupOrder=FALSE)

bpcp2sampControl

Control function for bpcp2samp

Description

Call function to change any one of options, and outputs a list with all defaults except argument that you changed.

Usage

bpcp2sampControl(Delta = 0, stype = "km", eps = 10^-8,  
nmc=10^6, method="mm.mc", seed=391291)

Arguments

Delta width of grouped confidence intervals, defaults to 0
stype type of survival estimate, either "km" for Kaplan-Meier or "mue" for median unbiased estimator
eps small value to make integration tractable
bpcpControl

Function that returns a list of arguments.

Usage

bpcpControl(midpMMTol = .Machine$double.eps^0.25,
seed=49911,
tolerance=.Machine$double.eps^0.5)

Arguments

midpMMTol value used for tol argument in uniroot call for calculating the midp method of moments method.
seed seed for set.seed() when using Monte Carlo method. If is.null(seed) then do not set the seed.
tolerance lowest positive value, such that abs(x-y)<tolerance treats x as equal to y. Used in bpcp for seeing if difference between times are equal to Delta or not.
fixtdiff

Details

When doing simulations on the Monte Carlo method, set seed=NULL. Then the seed will not be set at each replication. The default is to set the seed to 49911, so two analyses of the same data on the same version of R will give identical results.

Value

A list with components named as the arguments.

Description


Usage

```r
fixtdiff(time, status, group, testtime,
         trans=c("identity", "cloglog", "log"),
         varpooled=TRUE, correct=FALSE, doall=FALSE)
```

Arguments

- **time**: time to event for each observation
- **status**: status of event time, 1 is observed, 0 is right censored
- **group**: group for test, should have two levels, to change order use as factor and change order of levels
- **testtime**: fixed time when you want to test for a difference
- **trans**: type of transformation, one of 'identity', 'cloglog' or 'log'
- **varpooled**: logical, pool the variance?
- **correct**: logical, do continuity correction? Continuity correction for when trans='identity' and varpooled (see Warning)
- **doall**: logical, do all transformations and corrections

Details

This function provides p-values for the two sample tests that the survival distributions are equal at time testtime. The tests are asymptotically normal tests and are described in Klein, et al (2007). These functions are mostly for simulations to evaluate the melded BPCP tests, see bpcp2samp and Fay et al (2015).
kmci.object

Value

A list with the following components:

- `plo`: one-sided p-value, alternative: S1(testtime)>S2(testtime)
- `phi`: one-sided p-value, alternative: S1(testtime)<S2(testtime)
- `p2`: two-sided p-value, min(1,2*plo,2*phi)

Warning

Continuity correction derived from the case with no censoring (see Fleiss et al 3rd edition, pp. 50-55). May not make sense when there is censoring. Use at own risk.

Author(s)

Michael P. Fay

References


Examples

data(leuk2)
# Note that since the Kaplan-Meier survival at time=35 goes to
# zero for one group, the results for the log and cloglog
# transformations are undefined
fixtdiff(leuk2$time,leuk2$status,leuk2$treatment,35,doall=TRUE)

kmci.object

Kaplan-Meier (Survival Curve) Confidence Interval Object

Description

The kmci class is returned by the functions kmciTG or kmciSW. The class represents a fitted survival curve with pointwise confidence intervals.

Unlike the kmciLR class, which allows for confidence intervals to change at any time point, the kmci class only has the confidence intervals change at observed failures.

Objects of this class has methods for the functions summary, plot, lines.
Arguments

time the time points of observed failures (assumed surv and lower and upper steps that these times)
cens time points where there is censoring but no observed failure
surv the estimate of survival at time t+0. This is a vector.
upper upper confidence limit for the survival curve.
lower lower confidence limit for the survival curve.
conf.level the level of the confidence limits, e.g., 0.95.

Structure

The following components must be included in a legitimate kmci object.

See Also

kmciLR.object plot.kmci, summary.kmci, StCI.kmci, median.kmci, quantile.kmci.

kmciLR.object Kaplan-Meier (Survival Curve) Confidence Interval LR Tidy Object

Description

The kmciLR class allows for confidence intervals to change at any time point, while the kmci class only has the confidence intervals change at observed failures.

Objects of this class has methods for the functions summary, plot.

Arguments

cens time points where there is censoring but no observed failure
surv the estimate of survival in the interval described by L and R. This is a vector.
upper upper confidence limit for the survival curve in the interval described by L and R.
lower lower confidence limit for the survival curve in the interval described by L and R.
L vector of left ends of interval associated with lower and upper
Lin vector of logicals, should left end of interval be included?
R vector of right ends of interval associated with lower and upper
Rin vector of logicals, should right end of interval be included?
Interval character vector describing intervals
stype character vector giving type of survival estimate, either 'km' or 'mue'
conf.level the level of the confidence limits, e.g., 0.95.
kmciLRgroup.object

Structure

The following components must be included in a legitimate kmciLR object.

See Also

plot.kmciLR, summary.kmciLR, bpcp.

kmciLRgroup.object Kaplan-Meier (Survival Curve) Confidence Interval LR Group Object

Description

The kmciLRgroup class is returned by the function bpcpfit when the plotstyle = "standard", and represents a fitted survival curve with pointwise confidence intervals. This object will contain more than one fitted survival curve with the corresponding pointwise confidence intervals based on a treatment/grouping variable.

The kmciLRgroup class allows for confidence intervals to change at any time point (similar to the kmciLR class), while the kmci class only has the confidence intervals change at observed failures.

Objects of this class has methods for the functions summary, plot, print, quantile, and median. When printed, objects of this class will display the total number of subjects, total number of events, median survival time, and the confidence limits (for the pre-specified confidence level), by treatment group if applicable.

Arguments

cens  time points where there is censoring but no observed failure
surv  the estimate of survival in the interval described by L and R. This is a vector.
upper upper confidence limit for the survival curve in the interval described by L and R.
lower lower confidence limit for the survival curve in the interval described by L and R.
L     vector of left ends of interval associated with lower and upper
Lin    vector of logicals, should left end of interval be included?
R     vector of right ends of interval associated with lower and upper
Rin    vector of logicals, should right end of interval be included?
Interval character vector describing intervals
stype  character vector giving type of survival estimate, either 'km' or 'mue'
conf.level  the level of the confidence limits, e.g., 0.95.
um  total number of subjects at the start of that survival curve.
events  total number of events (observed failures) for that survival curve.
Structure

A kmciLRgroup object is a list of kmciLR objects (also lists). The length of the kmciLRgroup object corresponds to the number of treatment/grouping variables; each item in the list is a kmciLR object for the given treatment (the name of each item in the list is the name of each group). If no grouping variable is given, the output is a kmciLR object.

The following components must be included in each element of a legitimate kmciLRgroup object.

See Also

plot.kmciLRgroup, summary.kmciLRgroup, print.kmciLRgroup, quantile.kmciLRgroup, median.kmciLRgroup, bpcpfit.

kmciLRtidy.object  Kaplan-Meier (Survival Curve) Confidence Interval LR Tidy Object

Description

The kmciLRtidy class is returned by the function bpcpfit when the plotstyle = "ggplot", and represents a fitted survival curve with pointwise confidence intervals. This object may contain more than one fitted survival curve with the corresponding pointwise confidence intervals based on a treatment/grouping variable, if one is specified in the formula of bpcpfit.

The kmciLRtidy class allows for confidence intervals to change at any time point (similar to the kmciLR class), while the kmci class only has the confidence intervals change at observed failures.

Objects of this class has methods for the functions summary, plot, print, quantile, and median. When printed, objects of this class will display the total number of subjects, total number of events, median survival time, and the confidence limits(for the pre-specified confidence level), by treatment group if applicable.

Arguments

cens  time points where there is censoring but no observed failure
surv  the estimate of survival in the interval described by L and R. This is a vector.
upper upper confidence limit for the survival curve in the interval described by L and R.
lower lower confidence limit for the survival curve in the interval described by L and R.
L vector of left ends of interval associated with lower and upper
Lin vector of logicals, should left end of interval be included?
R vector of right ends of interval associated with lower and upper
Rin vector of logicals, should right end of interval be included?
Interval character vector describing intervals
stype character vector giving type of survival estimate, either 'km' or 'mue'
conf.level the level of the confidence limits, e.g., 0.95.
um total number of subjects at the start of that survival curve.
events total number of events (observed failures) for that survival curve.
Structure

A kmciLRtidy object is a list of kmciLR objects (also lists). The length of the kmciLRtidy object corresponds to the number of treatment/grouping variables; each item in the list is a kmciLR object for the given treatment (the name of each item in the list is the name of each group). If no grouping variable is given, the output is a list of 1 kmciLR object.

The following components must be included in each element of a legitimate kmciLRtidy object.

See Also

plot.kmciLRtidy, summary.kmciLRtidy, print.kmciLRtidy, quantile.kmciLRtidy, median.kmciLRtidy, bpcpf.

kmtestALL  

Pointwise confidence intervals for survival for right censored data.

Description

These functions give several different methods for calculating pointwise confidence intervals for the survival distribution for right censored data. The recommended confidence intervals are the beta product ones given by bpcp or bpcpf.

The other confidence intervals are included primarily to show that the beta product confidence procedure (using bpcp) has better coverage than the best alternatives. See details for a description of all the methods.

Usage

bpcp(time,status,nmc=0,alpha=.05,Delta=0,stype="km",midp=FALSE, 
    monotonic=NULL, control=bpcpControl())

bpcpf(time, ...)

## Default S3 method:
bpcpf(time, status = NULL, group = NULL, formula=NULL, nmc=0, alpha=NULL, conf.level=0.95, Delta=0, stype="km", midp=FALSE, 
    monotonic=NULL, control=bpcpControl(), plotstyle = "ggplot", 
    data=NULL, subset=NULL, na.action=NULL, ...)

## S3 method for class 'formula'
bpcpf(formula, data, subset, na.action, ...)

kmciBorkowf(time, status, type="log", alpha = 0.05)
kmtestBoot(time, status, tstar, pstar, M = 1000, alpha = 0.05)
kmtestConstrainBoot(time, status, tstar, pstar, M = 1000, alpha = 0.05)
kmtestConstrainBeta(time,status,tstar,pstar,alpha=.05)
kmciSW(time, status, alpha = 0.05)
kmcITG(time, status, alpha = 0.05)
kmtestALL

kmci1TG(time, status, tstar, alpha = 0.05)

kmtestALL(time, status, t0, S0, cens = NULL, M = 10^3, NMC = 10^5, alpha = 0.05)

**Arguments**

time  
time to event or censoring

status  
status vector, 1 is event, 0 is censoring (if NULL assumes all are events)

group  
vector of treatments or groups, if applicable

formula  
a formula object, which must have a Surv object as the response on the left of the ~ operator and, if desired, a grouping/treatment variable on the right. For a single survival curve the right hand side should be ~ 1.

data  
a data frame in which to interpret the variables named in the formula.

subset  
an optional vector specifying a subset of observations to be used.

na.action  
a function which indicates what should happen when the data contain NAs.

alpha  
1 - conf.level

conf.level  
confidence level. To be used in place of alpha starting in version 1.4

nmc  
number of Monte Carlo replications from each beta distribution, nmc=0 means use method of moments for beta parameters instead

NMC  
same as nmc

Delta  
width of grouped confidence intervals, defaults to 0 (rarely need to change this, even with ties, see details)

stype  
type of survival estimate, either "km" for Kaplan-Meier or "mue" for median unbiased estimator

midp  
logical, calculate the mid-p type of interval?

monotonic  
logical, force lower and upper confidence limits to be monotonic over time? If NULL: nmc=0 gives TRUE, nmc>0 gives FALSE

control  
list with arguments for adjusting numeric calculation. Generally does not need to be changed. See bpcpControl

plotstyle  
which type of plot to use, "ggplot" will produce a kmciLRtidy object that will be plotted with ggplot. "standard" will produce a kmciLRgroup object (if there are groups) which will be plotted with base R. If there are no groups, "standard" will produce a kmciLR object.

tstar  
time to test survival distribution

pstar  
null survival distribution

M  
number of bootstrap replications

t0  
null hypothesis time for survival test

S0  
null hypothesis value of survival at t0

cens  
vector of censoring times (even those with failures before it), used for Binomial test. If NULL gives NA for binom test

type  
see details

...  
Extra parameters to be passed.
Details

The recommended functions to calculate beta product confidence intervals are either \texttt{bpcp} or \texttt{bpcpfit}. The function \texttt{bpcp} has the original function arguments, while the \texttt{bpcpfit} function allows formulas similar to those used in the \texttt{survival} package, allowing separate analyses by group, and allowing easier use of \texttt{ggplot2} methods. The numerical output for the \texttt{bpcpfit} function is the same as multiple calls of \texttt{bpcp} for each group level.

The standard beta product confidence procedure (i.e., with midp=FALSE) will give pointwise confidence intervals for the survival function, $S$, with right censored data. This means that for any given $t$, we get confidence intervals for $S(t)$ with the following properties. When there is no censoring or Progressive Type II censoring the BPCP guarantees central coverage (e.g., the error rate on either side of the 95 percent confidence interval is guaranteed to be less than 2.5 percent). For general independent censoring the BPCP is asymptotically equivalent to standard methods such as the normal approximation with Greenwood variance, and hence the BPCP (as with the other confidence intervals given here) has asymptotically accurate coverage of $S(t)$ for any $t>0$.

The \texttt{bpcpfit} function will produce multiple survival curves with the pointwise confidence intervals for right censored data for different treatment/grouping variables. Only a treatment/grouping variable can be specified in this function. No other covariates should be included. Data can be input as either a formula or as a default method. If the plotstyle argument is "ggplot" (the default), then \texttt{bpcpfit} will return a \texttt{kmciLRtidy} object that can be passed into \texttt{plot}, and it will return a generic ggplot. If there is no group variable, a \texttt{kmciLRtidy} object will still be created. The \texttt{kmciLRtidy} object can also be passed to \texttt{tidykmciLR}, which returns a dataframe that can be passed into ggplot for custom plotting. If the plotstyle argument is "standard", then \texttt{bpcpfit} will return a \texttt{kmciLRgroup} object that can be passed into \texttt{"plot"}, which will return a base R plot. If there is no treatment variable, a \texttt{kmciLR} object will be created.

There is also a mid-p version of the BPCP. The BPCP is derived from using the known distribution of the failure times, and acting conservatively between the failure times (see Fay, Brittain, and Proschan, 2013 for details). Instead of acting conservatively between the failure times, the midp=TRUE version combines the distributions for the previous failure and the future failure time (see Fay and Brittain, 2016).

For a description of how \texttt{bpcp} with different values of Delta works, see "Beta Product Confidence Intervals for Discrete Failure Times" vignette (especially Section 2.2). Note especially that confidence intervals exactly at the failure times when Delta=0 are handled differently before Version 1.3.0 than from Version>=1.3.0. For users not interested in details who only want to know the recommended confidence intervals on right censored data when ties are allowed, we recommend the \texttt{bpcp} function version 1.3.0 or greater using the default Delta=0 argument. That recommendation will give pointwise confidence intervals that treats ties similarly to the way that the Kaplan-Meier estimator treats ties, and hence will give confidence intervals that enclose the Kaplan-Meier estimate.

Now we describe the other methods.

In general the functions are of three naming types: \texttt{kmtestXX}, \texttt{kmci1XX} and \texttt{kmciXX}, where \texttt{XX} changes for different methods. Functions \texttt{kmtestXX} only test whether $S(tstar)=pstar$ and return a vector of 1s for reject and 0s for fail to rejecting either of the one-sided or the two-sided hypotheses. Functions \texttt{kmci1XX} only give confidence intervals at $S(tstar)$, while \texttt{kmciXX} give confidence intervals for all values of $t$. The standard methods calculate the confidence intervals at the observed failure times and carry them forward (e.g., \texttt{kmciTG}, \texttt{kmciSW}) and the results are objects of class \texttt{kmci}. More involved methods allow confidence intervals to change after censored objects (e.g., \texttt{kmciBorkowf}, \texttt{bpcp}) and the results are objects of class \texttt{kmciLR}.
The function `kmtestBoot` tests S(tstar)=pstar using the nonparametric bootstrap (sampling vectors of (time,status) with replacement) with the percentile method as described in Efron (1981). The function `kmtestConstrainBoot` and `kmtestConstrainBeta` tests S(tstar)=pstar using the constrained Bootstrap or constrained Beta method described in Barber and Jennison (1999).

The function `kmci1TG` does a confidence interval only at tstar, while `kmciTG` does a confidence interval at all the observed event times. The method can be derived as a likelihood ratio test and is described in Thomas and Grunkemeier (1975). It has asymptotically correct coverage, which is rigorously proved in Murphy (1995). You can also think of the method as the empirical likelihood applied to the survival distribution for right censored data (see Owen, 2001, p. 144-145).

The function `kmciSW` calculates confidence intervals using Edgeworth expansions as described in Strawderman and Wells (1997). Note, Strawderman, Parzen and Wells (1997) is easier to understand than Strawderman and Wells (1997).

Borkowf (2005) creates confidence intervals for the Kaplan-Meier survival estimate for right censored data. He allows the confidence interval to change at censoring times as well as at failure times.

Four types of confidence intervals may be selected. The asymptotic normal approximation (type="norm"), the shifted K-M estimate with normal approximation (type="norms"), the log transformed normal approximation using the delta method (type="log"), and the log transformed normal approximation using the delta method with the shifted K-M (type="logs").

The function `kmtestALL` performs hypothesis tests on all the methods except the unconstrained bootstrap method (unless M=0 then it does not test the constrained bootstrap method either). The output is a matrix with three columns with a value of 1 representing either (1) rejection for two-sided test implying the estimate is greater than the null, (2) rejection for two-sided test implying the estimate less than the null, or (3) any rejection of the two-sided test. Each row represents a different test.

The `kmci`, `kmciLR`, `kmciLRtidy`, or `kmciLRgroup` classes have the following methods: "plot", "lines", "summary", "quantile", and "median". Additionally, you can pull out survival and confidence intervals from these objects at specific times using "StCI".

**Value**

The functions return an object of class either `kmci`, `kmciLR`, `kmciLRtidy` or `kmciLRgroup` see details).

`kmci`, `kmciLR`, `kmciLRtidy`, and `kmciLRgroup` objects are lists. `kmciLRtidy` and `kmciLRgroup` are lists of `kmciLR` objects, one element for each treatment/group. They contain elements

- `surv`: survival distribution in interval/at time point
- `lower`: lower pointwise confidence limit in interval/at time point
- `upper`: upper pointwise confidence limit in interval/at time point
- `time`: time of survival or confidence interval

while the `kmciLR` have intervals represented by the four elements

- `L`: left endpoint of interval
- `Lin`: logical vector, include left endpoint?
- `R`: right endpoint of interval
logical vector, include right endpoint?

and results from bpcp additionally have an element

betaParms list with 4 elements of beta parameters associated with the CIs: alower, blower, aupper, bupper

kmciLR objects contained in the kmciLRgroup and kmciLRgroup have the elements

num total number of subjects

events total number of events (observed failures)

Author(s)

Michael Fay

References


Thomas and Grunkemeier (1975) JASA 70: 865-871.

See Also

The kmci, kmciLR, kmciLRtidy, or kmciLRgroup objects have methods: "plot", "lines", "summary", "quantile", and "median", "StCI".

Examples

library(bpcp)
data(leuk)
data(leuk2)

### Recommended method is bpcp
### since the data are truncated to the nearest integer
### use Delta=1 option
out<-bpcp(leuk$time,leuk$status,Delta=1)
summary(out)
median(out)
plot(out)

### bpcpfit for multiple survival curves
a <- bpcpfit(Surv(time, status)~treatment, data=leuk2)
b <- bpcpfit(Surv(time, status)~1, data=leuk2)
fitBPsurv <- bpcpfit(Surv(time, status)~treatment, data=leuk2)
fitBPsurv

bpcpfit(leuk2$time, leuk2$status, leuk2$treatment)
bpcpfit(leuk2$time, leuk2$status, plotstyle = "standard")

### Borkowf 2005 method
norm<-kmciBorkowf(leuk$time,leuk$status,type="norm")
norms<-kmciBorkowf(leuk$time,leuk$status,type="norms")
## check Table VII of Borkowf
I<-c(1,2,3,5,7,8,9,11,13,15,17,19,21,23,25,27,29,31,33)
round(data.frame(lowerNorm=norm$lower[I],
                 upperNorm=norm$upper[I],
                 lowerNormS=norms$lower[I],
                 upperNorms=norms$upper[I],
                 row.names=norm$Interval[I]),3)

### Strawderman and Wells (1997) method
swci<-kmciSW(leuk$time,leuk$status)
swcic <- kmciSW(leuk$time,leuk$status)

### Thomas and Grunkemeier 1975 method
x<-kmciTG(leuk$time,leuk$status)
summary(x)
## compare to Table 1, Sample 2, of Thomas and Grunkemeier (1975)
StCI(x,c(10,20))

---

**leuk**

*Acute Leukemia data (treatment only) from Freireich et al (1963).*

---

**Description**

This is only the 21 patients who received 6-mercaptopurine (6-MP). There were 21 patients who got placebo (see leuk2 for complete data).

See also Borkowf (2005)

**Usage**

data(leuk)
Format

A data frame with 21 observations on the following 2 variables.

- time  time in remission (in weeks)
- status  event status, 1 is relapse, 0 is censored

References


See Also

leuk2 for complete data.

Examples

data(leuk)

---


Description

In this study there were 21 pairs of subjects, and within each pair one subject received 6-mercaptopurine (6-MP) and one got placebo. The data are right censored.

See also Gehan (1965) who used the data ignoring the pairing so that he could illustrate his famous two-sample (non-paired) rank test.

Usage

data(leuk2)

Format

A data frame with 42 observations on the following variables.

- time  time in remission (in weeks)
- status  event status, 1 is relapse, 0 is censored
- treatment  treatment group: either 'placebo' or '6-MP'
- pair  pair id number

References

mdiffmedian.test

See Also

leuk is only the treated group

Examples

data(leuk2)

Description

Tests for a difference in two medians. No assumptions about the two distributions are needed (may be discrete with ties allowed, no shift assumption is required). Uses the melded confidence interval derived from the one sample confidence intervals associated with the sign test (a version that allows for ties). Derivation of the test does not require large samples, and confidence intervals are intended to guarantee coverage regardless of sample size.

Usage

mdiffmedian.test(x1, x2, nulldiff = 0,
    alternative = c(“two.sided”, “less”, “greater”),
    conf.level = 0.95)

Arguments

x1 vector of numeric responses from group 1
x2 vector of numeric responses from group 2
nulldiff difference in medians under the null, median(x2)-median(x1)
alternative a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.
conf.level confidence level of the interval.

Details

The melded confidence interval method is a general method for combining two one-sample confidence intervals (CIs). In this function, we use the melded CI method on the two one-sample CIs from the sign test that allows for ties. This creates CIs for the difference in medians that requires very few assumptions. In particular, ties are allowed and no shift assumption is needed. For details see Fay, Proschan and Brittain (2015).
Value

a list of class 'htest' with elements:

- **statistic**: median of x1
- **parameter**: median of x2
- **p.value**: p-value of the test
- **conf.int**: confidence interval for the difference in medians
- **estimate**: median(x2)-median(x1)
- **null.value**: null hypothesis value for difference in medians
- **alternative**: type of alternative hypothesis
- **method**: description of test
- **data.name**: description of input

Note

This function does not allow censoring. Also, there is a price for not needing large samples nor assumptions about the distributions: if you do not have enough data, your confidence intervals may be the entire real line. For example, if you have continuous data with equal sample sizes in both groups, then if you have 6 or fewer observations in each group, then the 95 percent confidence interval on the difference in medians will be (-Inf,Inf).

Author(s)

Michael P. Fay

References


Examples

```r
set.seed(1)
trtA<-rpois(20,1.5)
trtB<-rpois(23,5.5)
mdiffmedian.test(trtA,trtB)
```
plot.kmciLR

Plot and lines methods for kmci, kmciLR, kmciLRtidy, and kmciLRgroup objects.

Description

Plots survival curves and/or confidence intervals. kmciLR objects from the bpcp function will plot using base R plot. If a kmciLRtidy object is created using bpcpfit (plotstyle = "ggplot"), the plot will display using ggplot. If a kmciLRgroup object is created using bpcpfit (plotstyle = "standard"), the plot will display using base R.

Usage

```r
## S3 method for class 'kmci'
plot(x, ...)

## S3 method for class 'kmciLR'
plot(x, XLAB = "time", YLAB = "Survival", YLIM = c(0, 1),
     ciLTY = 2, ciCOL = gray(0.8), mark.time = NULL,
     linetype = "both", ...)

## S3 method for class 'kmciLR'
lines(x, lty = c(2, 1), col = c(gray(0.8), gray(0)),
     linetype = "both", mark.time = NULL, ...)

## S3 method for class 'kmci'
lines(x, ...)

## S3 method for class 'kmciLRtidy'
plot(x, ...)

## S3 method for class 'kmciLRgroup'
plot(x, XLAB = "Time", YLAB = "Survival",
     YLIM = c(0, 1), ciLTY = 2, ciCOL = gray(.8), linetype = "both", ...)
```

Arguments

- `x`: kmci, kmciLR, kmciLRtidy, or kmciLRgroup object (created by functions described in `kmtestALL`)
- `XLAB`: label for x axis
- `YLAB`: label for y axis
- `YLIM`: limits for y axis
- `ciLTY`: lty (line type) for confidence intervals
- `ciCOL`: col (color) for confidence intervals
col vector of colors, first element used for ci second for survival curve
lty vector of line types, first element used for ci second for survival curve
mark.time put hash marks for censored objects (default puts marks of stype="km" but not if stype="mue")
linetype character, which lines to draw: either 'both', 'surv' or 'ci'
... Extra parameters to be passed. Any argument in plot.kmciLR can be passed from plot.kmci, similarly for line. Other parameters are usually graphical parameters passed to plot and segment calls within function.

Examples

data(leuk)
## kmciTG creates kmci object
fitTG<-kmciTG(leuk$time,leuk$status)
plot(fitTG)
## bpcp creates kmciLR object
fitBP<-bpcp(leuk$time,leuk$status)
lines(fitBP,lwd=3,lty=1,col=gray(.5),linetype="ci")
legend(0,.2,legend=c("Kaplan-Meier","Thomas-Grunkemeier 95 pct CI","Beta Product 95 pct CI"),
       lwd=c(1,1,3),lty=c(1,2,1),col=c(gray(0),gray(.8),gray(.5)))
data(leuk2)
## bpcpfit creates kmciLR object which is plotted with ggplot
fitBPsurv <- bpcpfit(Surv(time, status)~treatment, data=leuk2)
plot(fitBPsurv)
## this works even if no treatment variable is specified
plot(bpcpfit(Surv(time, status)~1, data=leuk2))

## if plotstyle "standard" is specified, a base R plot is produced
a <- bpcpfit(leuk2$time, leuk2$status, leuk2$treatment, plotstyle = "standard")
plot(a)

#can also create a tidy object to customize ggplot further
tidy <- tidykmciLR(fitBPsurv)
ggplot(tidy, aes(x = time, y = surv, ymin = lower, ymax = upper, col = group)) +
       geom_line(show.legend=FALSE) + geom_ribbon(alpha = .2, aes(fill=group)) +
       xlab("Time") + ylab("Survival") +
       ggtitle("K-M curves with bpcp CIs")

print.kmciLRtidy  

Print A Short Summary of a kmciLRtidy, kmciLRgroup, or kmciLR Object

Description

Print number of observations, number of events, and the median survival with confidence limits for the median of a kmciLRtidy, kmciLRgroup, or kmciLR object with pointwise confidence intervals. The confidence limits will match those specified in the bpcp or bpcpfit function.
Usage

```r
## S3 method for class 'kmciLR'
print(x, ...)
## S3 method for class 'kmciLRtidy'
print(x, ...)
## S3 method for class 'kmciLRgroup'
print(x, ...)
```

Arguments

- `x` a `kmciLRtidy`, `kmciLRgroup`, or `kmciLR` object
- `...` extra arguments

Examples

```r
library(bpcp)
data(leuk2)
practice <- bpcpfit(Surv(time, status)~treatment, data=leuk2)
practice

bpcpfit(Surv(time, status)~1, data=leuk2)
bpcpfit(Surv(time, status)~1, data=leuk2, plotstyle = "standard")
bpcpfit(leuk2$time, leuk2$status, leuk2$treatment, plotstyle = "standard")
```

---

quantile.kmciLR | **Quantiles or Medians from** `kmci`, `kmciLR`, `kmciLRtidy`, or `kmciLRgroup` objects.

Description

Get quantiles or median with the associated confidence intervals from a `kmci`, `kmciLR`, `kmciLRtidy`, or `kmciLRgroup` object.

Usage

```r
## S3 method for class 'kmciLR'
quantile(x, probs = c(0.25, 0.5, 0.75), ...)
## S3 method for class 'kmci'
quantile(x, probs = c(0.25, 0.5, 0.75), ...)
## S3 method for class 'kmciLRtidy'
quantile(x, probs = c(0.25, 0.5, 0.75), ...)
## S3 method for class 'kmciLRgroup'
quantile(x, probs = c(0.25, 0.5, 0.75), ...)
## S3 method for class 'kmciLR'
median(x, ...)
## S3 method for class 'kmci'
```
median(x, ...)  
## S3 method for class 'kmciLRtidy'
median(x, ...)
## S3 method for class 'kmciLRgroup'
median(x, ...)

Arguments

x  a kmci, kmciLR, kmciLRtidy, or kmciLRgroup object
probs  vector of probability to calculate quantiles
...  parameters passed

Value

A kmciLRtidy or kmciLRgroup object will produce a list of matrices.
The matrix has same number of rows as probs and 4 columns

S(q)  probs, survival estimate at quantile
q  quantile
lower  lower confidence limit of q
upper  upper confidence limit of q

Examples

data(leuk)
data(leuk2)
## kmciTG creates kmci object
fitTG<-kmciTG(leuk$time, leuk$status)
quantile(fitTG)
## bpcp creates kmciLR object
fitBP<-bpcp(leuk$time, leuk$status)
median(fitBP)

## kmciLRtidy and kmciLRgroups from bpcpfit
practice <- bpcpfit(Surv(time, status)~treatment, data=leuk2)
quantile(practice)
median(practice)
quantile(bpcpfit(leuk2$time, leuk2$status, leuk2$treatment, plotstyle = "standard"))
sclerosis


Description

Severe systemic sclerosis is a serious autoimmune disease affecting multiple organs including the heart, lungs, kidney, and skin. Between 1997 and 2005, a cohort of 34 patients was enrolled in a single arm pilot study of high-dose immunosuppressive therapy and autologous hematopoietic cell transplantation

Usage

data(sclerosis)

Format

A data frame with 34 observations on the following 3 variables.

day  time to death or censoring, in days
year  time to death or censoring, in years (day/365.25)
status  0 is censored, 1 is event

References


Examples

data(sclerosis)
plot(bpcp(sclerosis$year,sclerosis$status))

StCI

Get survival and confidence interval at t from kmci, kmciLR, or survfit object

Description

Just picks out the survival function and confidence interval in a different way depending on the type of object.
Usage

## Default S3 method:
StCI(x, tstar, afterMax = "continue", ...)

## S3 method for class 'kmciLR'
StCI(x, tstar, ...)

Arguments

- **x**: a kmci or kmciLR object
- **tstar**: a vector of times that you want survival and CI values
- **afterMax**: character, what to do after tmax (see details)
- **...**: further arguments to be passed to or from methods.

Details

Since the Kaplan-Meier estimator is undefined after the last observation if it is censored and many confidence interval methods are not defined there either, we need to explicitly define what to do. (For objects of the kmciLR class, the confidence intervals are defined over the positive real line and the afterMax is ignored.) The afterMax has four options for this: 'continue' (keep surv and ci values the same as the last calculated one), 'zero' (surv and lower go to zero, upper stays same), 'zeroNoNA' (surv and lower go to zero, upper stays same unless it is NA, then it takes on the last non-missing upper value), 'half' (surv goes to half value, lower goes to zero, upper stays same).

Value

The function StCI returns a data frame with the following variables. (It also has an attribute: 'conf.level').

- **time**: this is tstar
- **survival**: survival at tstar
- **lower**: lower confidence limit at tstar
- **upper**: upper confidence limit at tstar

Author(s)

Michael Fay

See Also

kmci, kmciLR
**Examples**

```r
data(leuk)
## compare to table 1 of Thomas and Grunkmeier (1975)
StCI(kmciTG(leuk$time, leuk$status), c(10, 20))
```

**Summary method for** kmci, kmciLR, kmciLRtidy, or kmciLRgroup **object.**

**Description**

Creates a data frame with time (for kmci) or time interval (for kmciLR, kmciLRtidy, and kmciLRgroup), survival, lower and upper pointwise confidence intervals. For kmciLRtidy and kmciLRgroup objects, the group (treatment) variable is also included.

**Usage**

```r
## S3 method for class 'kmciLR'
summary(object, ...)
## S3 method for class 'kmci'
summary(object, ...)
## S3 method for class 'kmciLRtidy'
summary(object, ...)
## S3 method for class 'kmciLRgroup'
summary(object, ...)
```

**Arguments**

- `object`: kmci, kmciLR, kmciLRtidy, or kmciLRgroup object
- `...`: extra arguments

**Value**

creates a data frame. See description.

**Examples**

```r
data(leuk)
## kmciTG creates kmci object
fitTG <- kmciTG(leuk$time, leuk$status)
summary(fitTG)
## bpcp creates kmciLR object
fitBP <- bpcp(leuk$time, leuk$status)
summary(fitBP)
data(leuk2)
## bpcpfit creates kmciLRtidy or kmciLRgroup object
fitBPsurv <- bpcpfit(Surv(time, status) ~ treatment, data = leuk2)
summary(fitBPsurv)
summary(Surv(time, status) ~ treatment, data = leuk2, plotstyle = "standard")
```
tidykmciLR  

Dataframe of kmciLRTidy, kmciLRgroup, or a kmciLR object.

Description
Takes a kmciLRTidy, kmciLRgroup, or a kmciLR object (a list) and converts it into a dataframe, which can further be used in plotting. Every two time points represents a time interval.

Usage

```r
tidykmciLR(x)
```

Arguments

- `x` a kmciLRTidy, kmciLRgroup, or kmciLR object

Details
creates a dataframe. See description.

Value
This function returns a dataframe with the following columns:

- `time` Time
- `surv` Value of survival curve at that time point
- `lower` Lower bound of the CI for the survival curve
- `upper` Upper bound of the CI for the survival curve
- `group` treatment or grouping variable (if applicable)

There are two rows per time point representing the change in either the survival function or confidence bands.

Examples

```r
library(bpcp)
data(leuk2)
practice <- bpcpfit(Surv(time, status)~treatment, data=leuk2)
tidy <- tidykmciLR(practice)
ggplot(tidy, aes(x = time, y = surv, ymin = lower, ymax = upper, col = group)) + geom_line(show.legend=FALSE) + geom_ribbon(alpha = .2, aes(fill=group)) + xlab("Time") + ylab("Survival") + ggtitle("K-M curves with bpcp CIs")
```
Index

* datasets
  leuk, 19
  leuk2, 20
  sclerosis, 27

* htest
  betaMeldTest, 4
  bpcp2samp, 5
  fixtdiff, 9
  kmtestALL, 14
  mdiffmedian.test, 21
  StCI, 27

* misc
  bpcp2sampControl, 7

* nonparametric
  bpcp-package, 2
  kmtestALL, 14
  StCI, 27

* optimize
  bpcpControl, 8

* package
  bpcp-package, 2

* survival
  bpcp-package, 2
  bpcp2samp, 5
  kmci.object, 10
  kmciLR.object, 11
  kmciLRgroup.object, 12
  kmciLRtidy.object, 13
  kmtestALL, 14
  plot.kmciLR, 23
  print.kmciLR, 24
  quantile.kmciLR, 25
  StCI, 27
  summary.kmciLR, 29
  tidykmciLR, 30

  betaMeldTest, 4
  bpcp, 2, 3, 12
  bpcp (kmtestALL), 14
  bpcp-package, 2

  bpcp2samp, 2–4, 5, 7–9
  bpcp2sampControl, 6, 7
  bpcpControl, 8, 15
  bpcpfit, 12–14
  bpcpfit (kmtestALL), 14

  fixtdiff, 9

  kmci, 11–13, 17, 18, 28
  kmci (kmci.object), 10
  kmci.object, 10
  kmci1TG (kmtestALL), 14
  kmciBorkowf (kmtestALL), 14
  kmciLR, 10, 12, 13, 16–18, 28
  kmciLR (kmciLR.object), 11
  kmciLR.object, 11, 11
  kmciLRgroup, 16–18
  kmciLRgroup (kmciLRgroup.object), 12
  kmciLRgroup.object, 12
  kmciLRtidy, 16–18
  kmciLRtidy (kmciLRtidy.object), 13
  kmciLRtidy.object, 13
  kmciSW, 10
  kmciSW (kmtestALL), 14
  kmciTG, 10
  kmciTG (kmtestALL), 14
  kmtestALL, 14, 23
  kmtestBoot (kmtestALL), 14
  kmtestConstrainBeta (kmtestALL), 14
  kmtestConstrainBoot (kmtestALL), 14

  leuk, 19, 21
  leuk2, 19, 20, 20
  lines, 17, 18
  lines.kmci (plot.kmciLR), 23
  lines.kmciLR (plot.kmciLR), 23

  mdiffmedian.test, 2, 21
  median, 17, 18
  median.kmci, 11
median.kmci (quantile.kmciLR), 25
median.kmciLR (quantile.kmciLR), 25
median.kmciLRgroup, 13
median.kmciLRgroup (quantile.kmciLR), 25
median.kmciLRtidy, 14
median.kmciLRtidy (quantile.kmciLR), 25

plot, 16–18
plot.kmci, 11
plot.kmci (plot.kmciLR), 23
plot.kmciLR, 12, 23
plot.kmciLRgroup, 13
plot.kmciLRgroup (plot.kmciLR), 23
plot.kmciLRtidy, 14
plot.kmciLRtidy (plot.kmciLR), 23
print.kmciLR (print.kmciLRtidy), 24
print.kmciLRgroup, 13
print.kmciLRgroup (print.kmciLRtidy), 24
print.kmciLRtidy, 14, 24

quantile, 17, 18
quantile.kmci, 11
quantile.kmci (quantile.kmciLR), 25
quantile.kmciLR, 25
quantile.kmciLRgroup, 13
quantile.kmciLRgroup (quantile.kmciLR), 25
quantile.kmciLRtidy, 14
quantile.kmciLRtidy (quantile.kmciLR), 25

tidykmciLR, 16, 30