Package 'adaptTest'

May 7, 2026

Title Adaptive Two-Stage Tests

Version 1.2

Date 2023-10-05

Description The functions defined in this program serve for implementing adaptive two-stage tests. Currently, four tests are included: Bauer and Koehne (1994), Lehmacher and Wassmer (1999), Vandemeulebroecke (2006), and the horizontal conditional error function. User-defined tests can also be implemented. Reference: Vandemeulebroecke, An investigation of two-stage tests, Statistica Sinica 2006.

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.2.3

Imports graphics, lattice, stats

NeedsCompilation no

Author Marc Vandemeulebroecke [aut, cre]

Maintainer Marc Vandemeulebroecke <vandemem@gmx.de>

Repository CRAN

Date/Publication 2023-10-06 14:20:10 UTC

Contents

adaptTest-package *Adaptive two-stage tests*

Description

The functions defined in this program serve for implementing adaptive two-stage tests.

Details

Package:	adaptTest
Type:	Package
Version:	1.2
Date:	2023-10-05
License:	GPL (version 2 or later)
LazyLoad:	yes

An adaptive two-stage test can be considered as a family of decreasing functions $f[c](p_1)$ in the unit square. Each of these functions is a conditional error function, specifying the type I error conditional on the p-value p_1 of the first stage. For example, $f[c](p_1) = \min(1, c/p_1)$ corresponds to Fisher's combination test (Bauer and Koehne, 1994). Based on this function family, the test can be put into practice by specifying the desired overall level α, stopping bounds $\alpha_1 <= \alpha_0$ and a parameter α_2. After computing p_1, the test stops with or without rejection of the null hypothesis if $p_1 <= \alpha_1$ or $p_1 > \alpha_0$, respectively. Otherwise, the null hypothesis is rejected if and only if $p_2 <= f[c](p_1)$ holds for the p-value p_2 of the second stage, where c is such that the local level of this latter test is α_2 (e.g., $c = c(\alpha_2) = \exp(-\chi^2_{4,\alpha_2}/2)$ for Fisher's combination test).

This package provides functions for handling conditional error functions, performing calculations among the different parameters (α, α_0, α_1, α_2 and c) and computing overall p-values, in addition to graphical visualization routines. Currently, four predefined tests are included: Bauer and Koehne (1994), Lehmacher and Wassmer (1999), Vandemeulebroecke (2006), and the horizontal conditional error function. User-defined tests can also be implemented.

This package contains the following functions:

- Key functions are `CEF`, `plotCEF`, `tsT`, `ovP`.
- Further functions are `a1Table`, `getpar`, `parconv`, `pathCEF`, `plotBounds`, `eq`, `ne`, `ge`, `gt`, `le`, `lt`.

The functions `a1Table`, `getpar`, `parconv` and `tsT` can handle the four predefined tests mentioned above. The functions `CEF`, `plotCEF`, `pathCEF` and `ovP` can also handle these, and user-defined tests in addition. The functions `plotBounds`, `eq`, `ne`, `ge`, `gt`, `le` and `lt` do not directly handle tests.

Note

Note that a family of conditional error functions can be parameterized in two alternative ways: more "traditionally" by some parameter c that, in turn, depends on the local level α_2 of the test after the second stage, or - perhaps more conveniently - by α_2 itself.

In this implementation, early stopping bounds are *not* part of the conditional error function. Rather, they are specified separately and "imposed" on it.

I want to thank Niklas Hack for technical support.

Author(s)

Marc Vandemeulebroecke

Maintainer: Marc Vandemeulebroecke <vandemem(at)gmx.de>

References

Bauer, P., Koehne, K. (1994). Evaluation of experiments with adaptive interim analyses. *Biometrics* 50, 1029-1041.

Brannath, W., Posch, M., Bauer, P. (2002). Recursive combination tests. *J. Amer. Statist. Assoc.* 97, 236-244.

Lehmacher, W., Wassmer, G. (1999). Adaptive sample size calculations in group sequential trials. *Biometrics* 55, 1286-1290.

Vandemeulebroecke, M. (2006). An investigation of two-stage tests. *Statistica Sinica* 16, 933-951.

Vandemeulebroecke, M. (2006). *A general approach to two-stage tests.* Doctoral thesis, Otto-von-Guericke-Universitaet Magdeburg, `http://www.dissertation.de`.

Vandemeulebroecke, M. (2008). Group sequential and adaptive designs - a review of basic concepts and points of discussion. *Biometrical Journal* 50, 541-557.

See Also

`CEF`, `tsT`

Examples

```
## Example from Bauer and Koehne (1994)
alpha  <- 0.1
alpha2 <- 0.1
alpha0 <- 0.5
alpha1 <- tsT(typ="b", a=alpha, a0=alpha0, a2=alpha2)
plotCEF(typ="b", a2=alpha2, add=FALSE)
plotBounds(alpha1, alpha0)
CEF(typ="b", a2=alpha2)
```

`a1Table` *Function to produce tables of alpha1*

Description

This function produces tables of α_1 for a grid of different choices of α and α_0.

Usage

```
a1Table(typ, a = NA, a0 = NA, Pocock = FALSE, round = FALSE)
```

Arguments

typ	type of test: "b" for Bauer and Koehne (1994), "l" for Lehmacher and Wassmer (1999), "v" for Vandemeulebroecke (2006) and "h" for the horizontal conditional error function
a	*vector* of different choices of α, the overall test level
a0	*vector* of different choices of α_0, the futility stopping bound
Pocock	logical determining whether the "Pocock-type" should be calculated or the full level should be applied after the second stage (see details; default: full level after second stage).
round	rounding specification, logical or integer (see details; default: no rounding)

Details

This function produces tables of α_1 on a grid spanned by the *vectors* a and a0 (i.e., α and α_0). This is done either for the "Pocock-type" (i.e., under the condition $\alpha_1 = \alpha_2$: Pocock = TRUE) or using the full level after the second stage ($\alpha = \alpha_2$: Pocock = FALSE (the default)). The function a1Table can be a convenient shortcut for a repeated use of tsT; see this latter function for further details.

The result is rounded to round digits after the comma (round = TRUE rounds to 1 digit; round = FALSE and round = 0 prevent rounding).

Value

a1Table returns a matrix of α_1 values, with the corresponding α and α_0 values being displayed as dimnames.

Author(s)

Marc Vandemeulebroecke

References

Bauer, P., Koehne, K. (1994). Evaluation of experiments with adaptive interim analyses. *Biometrics* 50, 1029-1041.

Lehmacher, W., Wassmer, G. (1999). Adaptive sample size calculations in group sequential trials. *Biometrics* 55, 1286-1290.

Vandemeulebroecke, M. (2006). An investigation of two-stage tests. *Statistica Sinica* 16, 933-951.

See Also

adaptTest package description, tsT

Examples

```
## Produce basic reference tables for the test by Vandemeulebroecke (2006)
alpha  <- c(0.1, 0.05, 0.025, 0.01)
alpha0 <- 1:10/10
a1Table(typ="v", a=alpha, a0=alpha0, Pocock=FALSE)
a1Table(typ="v", a=alpha, a0=alpha0, Pocock=TRUE)
```

CEF *Function to specify a conditional error function*

Description

This function returns a conditional error function.

Usage

```
CEF(typ = NA, fun = NA, dis = NA, a2 = NA, c = NA, p1 = NA, p2 = p1)
```

Arguments

typ	type of test: "b" for Bauer and Koehne (1994), "l" for Lehmacher and Wassmer (1999), "v" for Vandemeulebroecke (2006) and "h" for the horizontal conditional error function
c	the parameter c
a2	α_2, the local level of the test after the second stage
p1	the p-value p_1 of the test after the first stage
p2	the p-value p_2 of the test after the second stage, defaults to p1
fun	a conditional error function
dis	a distortion method for a supplied conditional error function (see details): "pl" for power lines, "vt" for vertical translation

Details

There are two alternative ways of specifying the desired conditional error function:

- through a type typ, and either a parameter (either a2 or c) or a point (p1,p2), OR
- through an initial conditional error function fun, and possibly a distortion method dis together with either the parameter a2 or a point (p1,p2)

Most people will only need the first of these two ways; the second leads to user-defined non-standard tests.

If typ is specified, a parameter a2 or c or the point (p1,p2) must be provided. In this case, CEF returns the conditional error function of the chosen type with the given parameter or running through the given point.

If typ is not specified, a conditional error function (i.e., a nonincreasing function defined on [0,1] with values in [0,1]) fun must be provided. If no distortion method is selected (dis = NA), fun is returned unchanged. Otherwise, the function is distorted using the chosen distortion method, either to match a desired second stage level a2 or to run through a specified point (p1,p2) (one of which must be provided). Currently, two distortion methods are implemented:

- dis = "pl", Power lines: For an initial function f, define $f[r](x) = (f(x^r))^(1/r)$, r>0. Note that if f is a conditional error function of type "b" (Bauer and Koehne, 1994), so is $f[r]$.
- dis = "pl", Vertical translation: The initial function is shifted vertically.

See parconv for more information on the two alternative parameterizations by α_2 and c.

Value

These functions return a conditional error *function* (see details).

Note

Provide either typ or fun, not both! If typ is provided, then also specify a2, c, or p1 (and possibly p2). If fun is provided, then also specify dis and a2, or dis and p1 (and possibly p2), or none of these.

Warning: Values of a2 close to 0 or 1 may not work for dis = "pl".

Note that in this implementation of adaptive two-stage tests, early stopping bounds are *not* part of the conditional error function. Rather, they are specified separately (see also tsT).

Author(s)

Marc Vandemeulebroecke

References

Bauer, P., Koehne, K. (1994). Evaluation of experiments with adaptive interim analyses. *Biometrics* 50, 1029-1041.

Lehmacher, W., Wassmer, G. (1999). Adaptive sample size calculations in group sequential trials. *Biometrics* 55, 1286-1290.

Vandemeulebroecke, M. (2006). An investigation of two-stage tests. *Statistica Sinica* 16, 933-951.

See Also

adaptTest package description, parconv, plotCEF, tsT

Examples

```
## Plot two conditional error functions of the Lehmacher-Wassmer (1999) type:
## one to the local level alpha2=0.1, and one that runs through (p1,p2)=(0.3,0.7)
foo1 <- CEF(typ="l", a2=0.1)
foo2 <- CEF(typ="l", p1=0.3, p2=0.7)
plot(foo1, xlim=0:1)
plot(foo2, add=TRUE)
```

```
## A different way of doing the same
plotCEF(typ="l", a2=0.1, add=FALSE)
plotCEF(typ="l", p1=0.3, p2=0.7, plt.pt=FALSE)
```

`getpar` *Function to calculate the parameter that specifies the conditional error function running through a given point*

Description

This function calculates the parameter that specifies the conditional error function running through a given point (p_1, p_2), based on a chosen family of conditional error functions.

Usage

```
getpar(typ, p1 = NA, p2 = p1, c = FALSE)
```

Arguments

`typ`	type of test: `"b"` for Bauer and Koehne (1994), `"l"` for Lehmacher and Wassmer (1999), `"v"` for Vandemeulebroecke (2006) and `"h"` for the horizontal conditional error function
`p1`	the p-value p_1 of the test after the first stage
`p2`	the p-value p_2 of the test after the second stage, defaults to `p1`
`c`	logical determining whether the parameter α_2 or the parameter c is returned (α_2 is the default).

Details

See `parconv` for more information on the two alternative parameterizations by α_2 and c.

Value

`getpar` returns the parameter (either α_2 or c, depending on the chosen parameterization) that specifies the conditional error function running through (p_1, p_2).

Author(s)

Marc Vandemeulebroecke

References

Bauer, P., Koehne, K. (1994). Evaluation of experiments with adaptive interim analyses. *Biometrics* 50, 1029-1041.

Lehmacher, W., Wassmer, G. (1999). Adaptive sample size calculations in group sequential trials. *Biometrics* 55, 1286-1290.

Vandemeulebroecke, M. (2006). An investigation of two-stage tests. *Statistica Sinica* 16, 933-951.

See Also

adaptTest package description, parconv, CEF

Examples

```
## Plot the conditional error function of the Lehmacher-Wassmer (1999)
##  type that runs through (p1,p2)=(0.3,0.7)
alpha2 <- getpar(typ="l", p1=0.3, p2=0.7)
plotCEF(typ="l", a2=alpha2, add=FALSE)

## Other ways of doing the same as above
plotCEF(typ="l", p1=0.3, p2=0.7, add=FALSE)
plot(CEF(typ="l", p1=0.3, p2=0.7), xlim=0:1)
```

OrderComparisons *Functions to perform simple order comparisons*

Description

These functions perform simple order comparisons for two arguments, dealing with the machine inaccuracy for floating point arithmetics.

Usage

```
eq(x, y, tol = .Machine$double.eps^0.5)
ne(x, y, tol = .Machine$double.eps^0.5)
ge(x, y, tol = .Machine$double.eps^0.5)
gt(x, y, tol = .Machine$double.eps^0.5)
le(x, y, tol = .Machine$double.eps^0.5)
lt(x, y, tol = .Machine$double.eps^0.5)
```

Arguments

x	first argument (must be a numeric scalar)
y	second argument (must be a numeric scalar)
tol	comparison tolerance; differences smaller than tol are not considered.

Details

When comparing two numeric scalars (e.g., for equality), machine inaccuracy can be the source of obviously erroneous results. These functions perform binary order comparisons that are tolerant towards machine inaccuracy, as an alternative to the standard comparators ==, !=, >=, >, <= and <.

Value

The functions return a logical TRUE if their condition holds, and a logical FALSE otherwise.

`eq(x, y)` checks whether `x` is equal to `y`

`ne(x, y)` checks whether `x` is not equal to `y`

`ge(x, y)` checks whether `x` is greater than or equal to `y`

`gt(x, y)` checks whether `x` is greater than `y`

`le(x, y)` checks whether `x` is less than or equal to `y`

`lt(x, y)` checks whether `x` is less than `y`

Note

These functions cannot be used in a vectorized fashion.

Author(s)

Marc Vandemeulebroecke

See Also

`identical`, `all.equal`

Examples

```
v <- seq(0.7, 0.8, by=0.1)
v[2]==0.8
eq(v[2], 0.8)
```

---

ovP *Function to compute and visualize overall p-values*

Description

This function computes and plots overall p-values for adaptive two-stage tests.

Usage

```
ovP(typ = NA, fun = NA, dis = NA, p1 = 1:49/50, p2 = p1,
    a1 = 0, a0 = 1, grid = FALSE, plt = FALSE,
    invisible = FALSE, wire = FALSE, round = FALSE)
```

Arguments

typ	type of test: "b" for Bauer and Koehne (1994), "l" for Lehmacher and Wassmer (1999), "v" for Vandemeulebroecke (2006) and "h" for the horizontal conditional error function
fun	a conditional error function
dis	a distortion method for a supplied conditional error function (see details): "pl" for power lines, "vt" for vertical translation
p1	the p-value p_1 of the test after the first stage, or a vector of such p-values
p2	the p-value p_2 of the test after the second stage, or a vector of such p-values; defaults to p1
a1	α_1, the efficacy stopping bound and local level of the test after the first stage (default: no stopping for efficacy)
a0	α_0, the futility stopping bound (default: no stopping for futility)
grid	logical determining whether a grid should be spanned by p1 and p2 (default: no grid is spanned)
plt	logical determining whether the overall p-values should be plotted or not (default: not)
invisible	logical determining whether the printing of the overall p-values should be suppressed or not (default: not)
wire	logical determining whether the overall p-values should be plotted in wireframe-style or in cloud-style (default: cloud-style)
round	rounding specification, logical or integer (see details; default: no rounding)

Details

The overall p-value for an adaptive two-stage test is computed as p_1 if $p_1 <= \alpha_1$ or $p_1 > \alpha_0$, and as

$$\alpha_1 + \int_{\alpha_1}^{\alpha_0} cef_{(p_1,p_2)}(x)dx$$

otherwise, where $cef_{(p_1,p_2)}$ is the conditional error function (of a specified family) running through the observed pair of p-values (p1,p2).

There are two alternative ways of specifying the family of conditional error functions (i.e., the test): through a type typ, or through an initial conditional error function fun and a distortion method dis; see CEF for details.

If p1 and p2 are of length 1, a single overall p-value is computed (and not plotted). Otherwise, the behavior of ovP depends on grid:

- If grid = FALSE, overall p-values are computed (and not plotted) for the elementwise pairs of p1 and p2. Here, p1 and p2 must be of the same length.
- If grid = TRUE, a grid is spanned by p1 and p2, and p-values are computed (and possibly plotted) over this grid. Here, p1 and p2 may be of different length. Plotting is triggered by plt = TRUE, and the style of the plot (wireframe or cloud) is determined by wire. invisible = TRUE suppresses the printing of the p-values.

The p-values are rounded to round digits after the comma (round = TRUE rounds to 1 digit; round = FALSE and round = 0 prevent rounding). The plot always shows unrounded values.

Value

A p-value, a vector of p-values or a matrix of p-values.

Note

Provide either `typ` or `fun`, not both! If `fun` is provided, then also specify `dis`.

Author(s)

Marc Vandemeulebroecke

References

Bauer, P., Koehne, K. (1994). Evaluation of experiments with adaptive interim analyses. *Biometrics* 50, 1029-1041.

Brannath, W., Posch, M., Bauer, P. (2002). Recursive combination tests. *J. Amer. Statist. Assoc.* 97, 236-244.

Lehmacher, W., Wassmer, G. (1999). Adaptive sample size calculations in group sequential trials. *Biometrics* 55, 1286-1290.

Vandemeulebroecke, M. (2006). An investigation of two-stage tests. *Statistica Sinica* 16, 933-951.

See Also

`adaptTest` package description, `CEF`

Examples

```
## Visualize a Lehmacher Wassmer (1999) test to the overall level 0.1
## and compute and visualize the overall p-value for an observed (p1,p2)=(0.3,0.7)
alpha  <- .1
alpha0 <- .5
alpha1 <- .05
plotBounds(a1=alpha1, a0=alpha0, add=FALSE)
plotCEF(typ="l", a2=tsT(typ="l", a=alpha, a0=alpha0, a1=alpha1))
plotCEF(typ="l", p1=.3, p2=.7)
ovP(typ="l", p1=.3, p2=.7, a1=alpha1, a0=alpha0)
# The overall p-value is the area left of alpha1, plus the area below the
# conditional error function running though (0.3,0.7) between alpha1 and alpha0.

## Investigate the p-values of the Lehmacher Wassmer (1999) test from above
ovP(typ="l", a1=alpha1, a0=alpha0, grid=TRUE, p1=1:9/10, round=3)
ovP(typ="l", a1=alpha1, a0=alpha0, grid=TRUE, plt=TRUE, invisible=TRUE, wire=TRUE)
```

`parconv` *Function to convert between two different parameterizations of a family of conditional error functions*

Description

This function converts between two different parameterizations of a family of conditional error functions: a (more 'traditional') parameter c, and a (more convenient) parameter α_2 specifying the local level of the test after the second stage.

Usage

```
parconv(typ, a2 = NA, c = NA)
```

Arguments

`typ`	type of test: `"b"` for Bauer and Koehne (1994), `"l"` for Lehmacher and Wassmer (1999), `"v"` for Vandemeulebroecke (2006) and `"h"` for the horizontal conditional error function
`a2`	α_2, the local level of the test after the second stage (see details)
`c`	the parameter c (see details)

Details

Traditionally, a family of conditional error functions is often parameterized by some parameter c that, in turn, depends on the local level α_2 of the test after the second stage. However, it can be convenient to parameterize the family directly by α_2. The function `parconv` converts one parameter into the other: provide one, and it returns the other.

Essentially, the relation between the two parameterizations is implemented as:

- $c = \exp(-\chi^2_{4,\alpha_2}/2)$ for Fisher's combination test (Bauer and Koehne, 1994)
- $c = \Phi^{-1}(1-\alpha_2)$ for the inverse normal method (Lehmacher and Wassmer, 1999)
- $\alpha_2 = (\Gamma(1+1/r))^2/\Gamma(1+2/r)$ for Vandemeulebroecke (2006)
- $c = \alpha_2$ for the family of horizontal conditional error functions

Value

`parconv` returns α_2 corresponding to the supplied c, or c corresponding to the supplied α_2.

Note

Provide either `a2` or `c`, not both!

α_2 is the local level of the test after the second stage, and it equals the integral under the corresponding conditional error function:

$$\alpha_2 = \int_0^1 cef_{\alpha_2}(p_1)dp_1,$$

where cef_{α_2} is the conditional error function (of a specified family) with parameter α_2.

Note that in this implementation of adaptive two-stage tests, early stopping bounds are *not* part of the conditional error function. Rather, they are specified separately (see also `tsT`).

α_2 can take any value in $[0, 1]$; c can take values in

- $[0, 1]$ for Fisher's combination test (Bauer and Koehne, 1994)
- $(-\infty, \infty)$ for the inverse normal method (Lehmacher and Wassmer, 1999)
- $[0, \infty)$ for Vandemeulebroecke (2006)
- $[0, 1]$ for the family of horizontal conditional error functions

Author(s)

Marc Vandemeulebroecke

References

Bauer, P., Koehne, K. (1994). Evaluation of experiments with adaptive interim analyses. *Biometrics* 50, 1029-1041.

Lehmacher, W., Wassmer, G. (1999). Adaptive sample size calculations in group sequential trials. *Biometrics* 55, 1286-1290.

Vandemeulebroecke, M. (2006). An investigation of two-stage tests. *Statistica Sinica* 16, 933-951.

See Also

`adaptTest` package description, `getpar`, `CEF`

Examples

```
## Obtain the parameter c for Fisher's combination test, using
##  the local level 0.05 for the test after the second stage
parconv(typ="b", a2=0.05)
```

`pathCEF` *Function to plot several conditional error functions running through a "path" of given points*

Description

This function plots several conditional error functions of the same family such that each one runs through one of several given points.

Usage

```
pathCEF(typ = NA, fun = NA, dis = NA, p1 = 1:49/50, p2 = p1,
      x = 0:200/200, plt.pt = FALSE, plt.ptann = FALSE, xlab = NA, ylab = NA, ...)
```

Arguments

typ	type of test: "b" for Bauer and Koehne (1994), "l" for Lehmacher and Wassmer (1999), "v" for Vandemeulebroecke (2006) and "h" for the horizontal conditional error function
fun	a conditional error function
dis	a distortion method for a supplied conditional error function (see details): "pl" for power lines, "vt" for vertical translation
p1	a vector (at least of length 2) of p-values p_1 of the test after the first stage
p2	a vector (at least of length 2) of p-values p_2 of the test after the second stage, defaults to p1; must be of same length as p1
x	vector on which the conditional error functions are plotted (should be relatively dense in [0,1])
plt.pt	logical determining whether the points that the conditional error functions are made to run through should be plotted or not (default: not)
plt.ptann	logical determining whether the points that the conditional error functions are made to run through should be annotated or not (default: not)
xlab	a label for the x axis (default: no label)
ylab	a label for the y axis (default: no label)
...	arguments to be passed on to the underlying plot and points functions (e.g., graphical parameters)

Details

It can be instructive to plot not only one conditional error function, but to visualize a whole family. This can easily be done with pathCEF. The function is used in a similar way as plotCEF, but p1 and p2 are now vectors (of the same length, at least of length 2). Conditional error functions are plotted that run through the specified elementwise points (p1,p2) (which by default lie on the main diagonal).

Internally, pathCEF uses plotCEF to plot the individual conditional error functions; see this latter function for further details.

Value

The function pathCEF is invoked for its plotting effect; it returns no meaningful value.

Note

Provide either typ or fun, not both! If fun is provided, then also specify dis.

Unlike plotCEF, it is not possible with pathCEF to specify the conditional error functions by the parameter α_2 or the parameter c.

plt.ptann is not considered if plt.pt = FALSE.

Author(s)

Marc Vandemeulebroecke

References

Bauer, P., Koehne, K. (1994). Evaluation of experiments with adaptive interim analyses. *Biometrics* 50, 1029-1041.

Lehmacher, W., Wassmer, G. (1999). Adaptive sample size calculations in group sequential trials. *Biometrics* 55, 1286-1290.

Vandemeulebroecke, M. (2006). An investigation of two-stage tests. *Statistica Sinica* 16, 933-951.

See Also

adaptTest package description, CEF, plotCEF, tsT

Examples

```
## Compare the tests by Bauer and Koehne (1994),
##  Lehmacher and Wassmer (1999) and Vandemeulebroecke (2006)
oldmfcol <- par(mfcol=c(1,3))
pathCEF(typ="b", main="BK 94")
pathCEF(typ="l", main="LW 99")
pathCEF(typ="v", main="V 06")
par(oldmfcol)
```

---

plotBounds *Function to plot the stopping bounds of an adaptive two-stage test*

Description

This function plots the stopping bounds of an adaptive two-stage test.

Usage

```
plotBounds(a1 = 0, a0 = 1, add = TRUE, xlab = NA, ylab = NA, ...)
```

Arguments

a1	α_1, the efficacy stopping bound and local level of the test after the first stage (default: no stopping for efficacy)
a0	α_0, the futility stopping bound (default: no stopping for futility)
add	logical determining whether the bounds should be added to an existing plot (default) or a new plot should be opened
xlab	a label for the x axis (default: no label)
ylab	a label for the y axis (default: no label)
...	arguments to be passed on to the underlying lines functions (e.g., graphical parameters)

Details

This function plots the stopping bounds α_1 and α_0 of an adaptive two-stage test, either onto an existing plot or into a new plot.

Value

The function `plotBounds` is invoked for its plotting effect; it returns no meaningful value.

Note

Note that in this implementation of adaptive two-stage tests, early stopping bounds are *not* part of the conditional error function. Rather, they are specified separately (see also `tsT`).

Author(s)

Marc Vandemeulebroecke

See Also

`adaptTest` package description, `plotCEF`

Examples

```
## Example from Bauer and Koehne (1994): full level after final stage, alpha0 = 0.5
alpha  <- 0.1
alpha2 <- 0.1
alpha0 <- 0.5
alpha1 <- tsT(typ="b", a=alpha, a0=alpha0, a2=alpha2)
plotCEF(typ="b", a2=alpha2, add=FALSE)
plotBounds(alpha1, alpha0)
```

---

plotCEF *Function to plot a conditional error function*

Description

This function plots a conditional error function.

Usage

```
plotCEF(typ = NA, fun = NA, dis = NA, a2 = NA, c = NA, p1 = NA, p2 = p1,
        x = 0:200/200, add = TRUE, xlim = c(0, 1), ylim = c(0, 1),
        plt.pt = TRUE, plt.ptann = TRUE, xlab = NA, ylab = NA, ...)
```

Arguments

typ	type of test: "b" for Bauer and Koehne (1994), "l" for Lehmacher and Wassmer (1999), "v" for Vandemeulebroecke (2006) and "h" for the horizontal conditional error function
fun	a conditional error function
dis	a distortion method for a supplied conditional error function (see details): "pl" for power lines, "vt" for vertical translation
a2	α_2, the local level of the test after the second stage
c	the parameter c
p1	the p-value p_1 of the test after the first stage
p2	the p-value p_2 of the test after the second stage, defaults to p1
x	vector on which the conditional error function is plotted (should be relatively dense in [0,1])
add	logical determining whether the bounds should be added to an existing plot (default) or a new plot should be opened
xlim	the x limits of the plot (default: c(0, 1); other choices can be used to "zoom in")
ylim	the y limits of the plot (default: c(0, 1); other choices can be used to "zoom in")
plt.pt	logical determining whether the point that the conditional error function is made to run through should be plotted or not (default: yes)
plt.ptann	logical determining whether the point that the conditional error function is made to run through should be annotated or not (default: yes)
xlab	a label for the x axis (default: no label)
ylab	a label for the y axis (default: no label)
...	arguments to be passed on to the underlying plot and points functions (e.g., graphical parameters)

Details

There are two alternative ways of specifying the desired conditional error function:

- through a type typ, and either a parameter (either a2 or c) or a point (p1,p2), OR
- through an initial conditional error function fun, and possibly a distortion method dis together with either the parameter a2 or a point (p1,p2)

Most people will only need the first of these two ways; the second leads to user-defined non-standard tests.

If typ is specified, a parameter a2 or c or the point (p1,p2) must be provided. In this case, plotCEF plots the conditional error function of the chosen type with the given parameter or running through the given point.

If typ is not specified, a conditional error function (i.e., a nonincreasing function defined on [0,1] with values in [0,1]) fun must be provided. If no distortion method is selected (dis = NA), fun is

plotted unchanged. Otherwise, the function is distorted using the chosen distortion method, either to match a desired second stage level a2 or to run through a specified point (p1,p2) (one of which must be provided). Currently, two distortion methods are implemented:

- dis = "pl", Power lines: For an initial function fun, define $f[r](x) = (f(x^r))^(1/r)$, r>0. Note that if fun is a conditional error function of type "b" (Bauer and Koehne, 1994), so is f[r].
- dis = "pl", Vertical translation: The initial function fun is shifted vertically.

See parconv for more information on the two alternative parameterizations by α_2 and c.

Internally, plotCEF uses CEF to compute the conditional error function that is to be plotted.

Value

The function plotCEF is invoked for its plotting effect; it returns no meaningful value.

Note

Provide either typ or fun, not both! If typ is provided, then also specify a2, c, or p1 (and possibly p2). If fun is provided, then also specify dis and a2, or dis and p1 (and possibly p2), or none of these.

Warning: Values of a2 close to 0 or 1 may not work for dis = "pl".

plt.pt and plt.ptann are not considered if p1 = NA. plt.ptann is not considered if plt.pt = FALSE.

Note that in this implementation of adaptive two-stage tests, early stopping bounds are *not* part of the conditional error function. Rather, they are specified separately (see also tsT).

Author(s)

Marc Vandemeulebroecke

References

Bauer, P., Koehne, K. (1994). Evaluation of experiments with adaptive interim analyses. *Biometrics* 50, 1029-1041.

Lehmacher, W., Wassmer, G. (1999). Adaptive sample size calculations in group sequential trials. *Biometrics* 55, 1286-1290.

Vandemeulebroecke, M. (2006). An investigation of two-stage tests. *Statistica Sinica* 16, 933-951.

See Also

adaptTest package description, parconv, CEF, tsT

Examples

```
## Plot two conditional error functions of the Lehmacher-Wassmer (1999) type:
## one to the local level alpha2=0.1, and one that runs through (p1,p2)=(0.3,0.7)
plotCEF(typ="l", a2=0.1, add=FALSE)
plotCEF(typ="l", p1=0.3, p2=0.7)

## Plot an explicitly defined conditional error function, and distort it
plotCEF(fun=function(x) ifelse(x<.5,(1-x)^2, (1-x)/2), add=FALSE)
plotCEF(fun=function(x) ifelse(x<.5,(1-x)^2, (1-x)/2), dis="pl", a2=.5)
foo <- CEF(fun=function(x) ifelse(x<.5,(1-x)^2, (1-x)/2), dis="pl", a2=.5)
plotCEF(fun=foo, col="red")
```

---

tsT *Function to implement an adaptive two-stage test*

Description

There are four key quantities for the specification of an adaptive two-stage test: the overall test level α, stopping bounds $\alpha_1 <= \alpha_0$ and the local level α_2 of the test after the second stage. These quantities are interrelated through the overall level condition. The function `tsT` calculates any of these quantities based on the others.

Usage

```
tsT(typ, a = NA, a0 = NA, a1 = NA, a2 = NA)
```

Arguments

`typ`	type of test: `"b"` for Bauer and Koehne (1994), `"l"` for Lehmacher and Wassmer (1999), `"v"` for Vandemeulebroecke (2006) and `"h"` for the horizontal conditional error function
`a`	α, the overall test level
`a0`	α_0, the futility stopping bound
`a1`	α_1, the efficacy stopping bound and local level of the test after the first stage
`a2`	α_2, the local level of the test after the second stage

Details

An adaptive two-stage test can be viewed as a family of decreasing functions $f[c](p_1)$ in the unit square. Each of these functions is a conditional error function, specifying the type I error conditional on the p-value p_1 of the first stage. For example, $f[c](p_1) = \min(1, c/p_1)$ corresponds to Fisher's combination test (Bauer and Koehne, 1994). Based on this function family, the test can be put into practice by specifying the desired overall level α, stopping bounds $\alpha_1 <= \alpha_0$ and a parameter α_2. After computing p_1, the test stops with or without rejection of the null hypothesis if $p_1 <= \alpha_1$ or $p_1 > \alpha_0$, respectively. Otherwise, the null hypothesis is rejected if and only if $p_2 <= f[c](p_1)$ holds for the p-value p_2 of the second stage, where c is such that the local level of this latter test is α_2 (e.g., $c = c(\alpha_2) = \exp(-\chi^2_{4,\alpha_2}/2)$ for Fisher's combination test).

The four parameters α, α_0, α_1 and α_2 are interdependent: they must satisfy the level condition

$$\alpha_1 + \int_{\alpha_1}^{\alpha_0} cef_{\alpha_2}(p_1)dp_1 = \alpha,$$

where cef_{α_2} is the conditional error function (of a specified family) with parameter α_2. For example, this conditon translates to

$$\alpha = \alpha_1 + c(\alpha_2) * (\log(\alpha_0) - \log(\alpha_1))$$

for Fisher's combination test (assuming that $c(\alpha_2) < \alpha_1$; Bauer and Koehne, 1994). The function `tsT` calculates any of the four parameters based on the remaining ones. Currently, this is implemented for the following four tests: Bauer and Koehne (1994), Lehmacher and Wassmer (1999), Vandemeulebroecke (2006), and the horizontal conditional error function.

Value

If three of the four quantities α, α_0, α_1 and α_2 are provided, `tsT` returns the fourth. If only α and α_0 are provided, `tsT` returns α_1 under the condition $\alpha_1 = \alpha_2$ (the so-called "Pocock-type").

If the choice of arguments is not allowed (e.g., $\alpha_0 < \alpha_1$) or when a test cannot be constructed with this choice of arguments (e.g., $\alpha_0 = 1$ and $\alpha < \alpha_2$), `tsT` returns `NA`.

IMPORTANT: When the result is (theoretically) not unique, `tsT` returns the maximal α_1, maximal α_2 or minimal α_0.

In all cases, `tsT` returns the result for the test specified by `typ`.

Note

The argument `typ`, and either exactly three of α, α_0, α_1 and α_2, or only α and α_0, must be provided to `tsT`.

Author(s)

Marc Vandemeulebroecke

References

Bauer, P., Koehne, K. (1994). Evaluation of experiments with adaptive interim analyses. *Biometrics* 50, 1029-1041.

Lehmacher, W., Wassmer, G. (1999). Adaptive sample size calculations in group sequential trials. *Biometrics* 55, 1286-1290.

Vandemeulebroecke, M. (2006). An investigation of two-stage tests. *Statistica Sinica* 16, 933-951.

Vandemeulebroecke, M. (2008). Group sequential and adaptive designs - a review of basic concepts and points of discussion. *Biometrical Journal* 50, 541-557.

See Also

`adaptTest` package description

Examples

```
## Example from Bauer and Koehne (1994): full level after final stage, alpha0 = 0.5
alpha  <- 0.1
alpha2 <- 0.1
alpha0 <- 0.5
alpha1 <- tsT(typ="b", a=alpha, a0=alpha0, a2=alpha2)
plotCEF(typ="b", a2=alpha2, add=FALSE)
plotBounds(alpha1, alpha0)

## See how similar Lehmacher and Wassmer (1999) and Vandemeulebroecke (2006) are
alpha  <- 0.1
alpha1 <- 0.05
alpha0 <- 0.5
alpha2l <- tsT(typ="l", a=alpha, a0=alpha0, a1=alpha1)
alpha2v <- tsT(typ="v", a=alpha, a0=alpha0, a1=alpha1)
plotCEF(typ="l", a2=alpha2l, add=FALSE)
plotCEF(typ="v", a2=alpha2v, col="red")
plotBounds(alpha1, alpha0)

## A remark about numerics
tsT(typ="b", a=0.1, a1=0.05, a0=0.5)
tsT(typ="b", a=0.1, a2=0.104877, a0=0.5)
tsT(typ="b", a=0.1, a2=tsT(typ="b", a=0.1, a1=0.05, a0=0.5), a0=0.5)

## An example of non-uniqueness: the maximal alpha1 is returned; any
##  smaller value would also be valid
alpha  <- 0.05
alpha0 <- 1
alpha2 <- 0.05
alpha1 <- tsT(typ="b", a=alpha, a0=alpha0, a2=alpha2)
tsT(typ="b", a0=alpha0, a1=alpha1, a2=alpha2)
tsT(typ="b", a0=alpha0, a1=alpha1/2, a2=alpha2)
```

Index