Package ‘nivm’

September 26, 2015

Type Package

Title Noninferiority Tests with Variable Margins

Version 0.3

Date 2015-09-25

Author Michael P. Fay

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

Depends stats, bpcp, ssanv

Description
Noninferiority tests for difference in failure rates at a prespecified control rate or prespecified time.

NeedsCompilation no

Repository CRAN

Date/Publication 2015-09-26 01:31:34

R topics documented:

`nivm-package` *Non-inferiority Tests with Variable Margins*

Description

This package was developed to give the control quantile non-inferiority test described in Fay and Follmann (2015), and the function to calculate that test is `nicqTest`. Some competitors to that test are some tests described in Rohmel and Kieser (2013): `brkTest`, `fmecTest`, and `fmecExact`.

Details

Package:	nivm
Type:	Package
Version:	0.3
Date:	2015-09-25
License:	>=GPL3

Author(s)

Michael P. Fay

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

References

Fay, MP and Follmann DA (2015). Non-inferiority Tests for Anti-Infective Drugs using Control Quantiles. (unpublished manuscript).

Rohmel, J and Kieser, M (2013). Investigations on non-inferiority–the Food and Drug Administration draft guidance on treatments for nosocomial pneumonia as a case for exact tests for binomial proportions. Statistics in Medicine. 32: 2335-2348.

See Also

`nicqTest`

`brkControl` *Arguments for Algorithm Control for* `brkTest`.

Description

Function that gives a list.

Usage

```
brkControl(alpha = 0.025, alphastar = 0.001, ngrid = 1000)
```

Arguments

`alpha`	significance level for test
`alphastar`	a value that is much much less than the significance level. Used to speed up calculations since we group all values less than alphastar together and do not need to add them one-at-a-time.
`ngrid`	number of elements in the grid search over the control proportion.

Value

a list with values names the same as the arguments.

See Also

`brkTest`

`brkTest` *Barnard-Rohmel-Kieser Test*

Description

A variable margin difference in proportion test for non-inferiority. The test is based on Barnard's test.

Usage

```
brkTest(x1, n1, x2, n2, threshold = 0.2, delta = 0.1, control = brkControl())
```

Arguments

`x1`	number of events in the control group
`n1`	number of individuals in the control group
`x2`	number of events in the test group
`n2`	number of events in the test group
`threshold`	proportion in the control group associated with the threshold, above that threshold use a constant difference margin, below the threshold use a difference margin with a constant odds ratio. We use only continuous variable margins that meet at the threshold.
`delta`	difference in proportions at the threshold
`control`	list of parameters for algorithm control, see `brkControl`

Details

This test is labeled T4 in Rohmel and Keiser (2013).

Value

a list of class brk, with elements:

statistic	the threshold, delta (difference margin at threshold), and odds ratio at threshold
data.name	gives x1,x2,n1,n2 as a character string
method	description of test
p.value	one-sided p-value
FullResults	a list with 4 matrices, each n1+1 by n2+1 representing the total sample space. R=a matrix with logical values with TRUE elements representing the rejection region, its 'sig.level' attribute gives the significance level of the test; PVALbounds=a matrix of p-value bounds, pb; PVALsymbols=a matrix of symbols that describe the pb, '<=' means 'p<=pb', '=' means 'p=pb' and '>' means 'p>pb'; PVALUES=a matrix giving the p-value expression, e.g., 'p<=.00321' or 'p>0.025'.

Author(s)

Michael P. Fay

References

Rohmel, J, and Kieser, M (2013). "Investigations on non-inferiority - - the Food and Drug Administration draft guidance on treatments for nosocomial pneumonia as a case for exact tests for binomial proportions" Statistics in Medicine 32:2335-2348.

See Also

See Also nicqTest, ~~~

Examples

```
x<-brkTest(3,8,0,6)
x
x$FullResults$PVALUES
```

findPowerR *Get power from brkTest).*

Description

Power under an alternative F1 and F2 relationship, represented by a F2=g(F1).

Usage

```
findPowerR(R, g, psearch = (0:1000)/1000)
```

Arguments

R	matrix of rejection region, if x is the output from `brk.test`, then R=x$FullResults$R
g	function under which to calculate the power, F2=g(F1).
psearch	vector of values over which to calculate the power

fmecTest *Odds Ratio/Difference Non-inferiority tests*

Description

Rohmel and Keier (2013) developed these non-inferioirty tests with variable margins. One margin function, NiM3, has the variable margin measuring a constant difference in proportions (0.10 in paper) after a threshold (0.20 proportion in the control group), or tests for differences defined in terms of a constant odds ratio (1.71 in paper) at values less than the threshold. The `fmecTest` with type='max' gives the maximum of two p-values, either a difference in proportions test (one-sided asymptotic method of Farrington and Manning, 1990) or an odds ratio test (one-sided Fisher's exact). This test is NiM3/T2 in Rohmel and Keier (2013). We also provide an exact version of this test with `fmecExact`, denoted NiM3/T3 in Rohmel and Keier (2013). When type='switch' the tests are like T1 of Rohmel and Keier (2013).

Usage

```
fmecTest(x1, n1, x2, n2, threshold = 0.2, delta = 0.1,
    alternative = c("less","greater"),
    type = c("max", "switch"))

fmecExact(x1, n1, x2, n2, threshold = 0.2, delta = 0.1,
    alternative = c("less", "greater"),
    type = c("max", "switch"), ngrid = 1000)
```

Arguments

`x1`	number of failures in control group
`n1`	number of individuals in control group
`x2`	number of failures in test group
`n2`	number of individuals in test group
`threshold`	threshold on proportion in control group: above it use constant difference margin, below it use difference margin with constant odds ratio
`delta`	difference margin at threshold
`alternative`	must be 'less'. the value 'greater' is not supported at this time.
`type`	either 'max' (maximum of Fisher's exact p-value or Farrington and Manning p-value) or 'switch' (Fisher's exact p-value below threhold and Farington and Manning p-value above threhold).
`ngrid`	grid size for the search for the maximum p-value. Search over the control proportion values 0:ngrid/ngrid.

Details

For details see Rohmel and Keier (2013, Section 3). These functions only use NiM3.

Value

a list of class 'htest':

`statistic`	the threshold, delta (difference margin at threshold), and odds ratio at threshold
`data.name`	gives x1,x2,n1,n2 as a character string
`method`	description of test
`p.value`	one-sided p-value
`null.value`	delta, the difference margin at threshold
`alternative`	direction of alternative hypothesis

Author(s)

Michael P. Fay

References

Farrington, CP and Manning G (1990). "Test statistics and sample size formulae for comparative binomial trials with null hypothesis of non-zero risk difference or non-unit relative risk" Statistics in Medicine 9:1447-1454.

Rohmel, J, and Kieser, M (2013). "Investigations on non-inferiority - - the Food and Drug Administration draft guidance on treatments for nosocomial pneumonia as a case for exact tests for binomial proportions" Statistics in Medicine 32:2335-2348.

See Also

`nicqTest`

Examples

```
fmecTest(6,10,2,12,alternative="less",type="max")
fmecExact(6,10,2,12,alternative="less",type="max")
```

`nicqControl` *Function that returns a list of algorithm controls for nicq*

Description

Controls for numeric integration, etc. Mostly used in `getfx2` that is called by `nicqTest`.

Defined as a function instead of a list, so sanity checks can be built in (but none have been included yet).

Usage

```
nicqControl(rdig = 5, slowint = FALSE, mint = 100,
    interr = 10^-3,  epsilon=10^(-4), alpha = 0.025,
    tau.conf.level=0.95)
```

Arguments

`rdig`	number of digits for rounding, used to eliminate some computer errors. Used in `getimaxpower` called by `nicqTest` when ic="maxpower".
`slowint`	use slow integration for `getfx2`
`mint`	number of summands in numeric integration for `getfx2`
`interr`	tolerance for integration for `getfx2`
`epsilon`	small value to give the range for the uniroot function that calculates the confidence intervals. It searches from -q+epsilon to 1-q-epsilon. Used in `nicq.calc` called by `nicqTest`.
`alpha`	significance level for calculation of `getimaxpower` called by `nicqTest` when ic="maxpower".
`tau.conf.level`	confidence level for tau, where F1(tau)=q. Uses `bpcp` then `quantile.kmciLR`.

Value

a list with each argument as a named value

See Also

`nicqTest`

`nicqTest` *Non-inferiority control quantile test*

Description

Tests for a difference in proportion of failures between test and control by the time the qth quantile of the contol group has failed. Uses a variable margin function, and the time of the qth quantile of the control group is unknown.

If the cumulative distributions for the two groups are F1 (control) and F2 (test), then we are interested in the difference: delta=F2(t0)-F1(t0), where F1(t0)=q. Note F1,F2 are unknown and non-parametric, and t0 is unknown. In this case, using a constant delta does not give practical non-inferiority margins, therefore we use a variable margin function, so that we test (when alternative='less') H0: F2(t) >= g(F1(t)) versus H1: F2(t) < g(F1(t)) for all t.

The test also works for other types of continuous responses besides time to failure (see details), but the help description uses time to failure for brevity.

Usage

```
nicqTest(x,delta0,q,g=nimDiffOR,yc=NULL,nc=NULL,nt=NULL,
    ic="prop",
    z=NULL,status=NULL,ties=c("cons","approx"),
    alternative=c("less","greater"),
    conf.level=0.95,
    conf.int=TRUE,
    conf.sided=c("two.sided","one.sided"),
    gname=NULL,
    control=nicqControl())
```

Arguments

- `x` either a vector of failure times for the both groups (when z is given), a vector of failure times for the test group (when yc is given), or the number of failures in the test group that have occured by the ic^th failure in the control group (when ic is an integer, nc and nt are given). See details.
- `delta0` difference, F2(t0)-F1(t0), on the boundary between the null and alternative hypotheses, where t0 is defined so that F1(t0)=q.
- `q` probability associated with the quantile of interest in control group
- `g` non-inferiority margin function. Must have arguments `q` (representing the quantile of interest in the control group) and `delta` (representing the difference, F2(t0)-F1(t0), where F2 and F2 are the cumulative distributions of failures for the test (F2) and control (F1) at t, where F1(t)=q). Default values for `q` and `delta` are ignored. Default function is `nimDiffOR`.
- `yc` vector of failure times in the control group. If given, `x` is the vector of failure times in the test group.
- `nc` number of individuals in the control group. Not needed if `z` or `yc` is given.

`nt`	number of individuals in the test group. Not needed if `z` or `yc` is given.
`ic`	used to find i. The test is based on the number of failures in the test group that have occured by the ith failure in the control group. ic="prop" gives i=ceiling(q*nc), ic="maxpower" gives the i value that maximizes the power given F1=F2 and g, and ic=a postive integer gives i=ic (with ic between 1 and nc inclusive).
`z`	a vector of group indicators, with either 1 (for control) or 2 (for test). If given, x is a vector of all failures in both groups.
`status`	a vector denoting right censoring (0) or not (1). Not needed if there is no censoring. Only used when `z` is given. If any censoring occurs at or before the ith failure (see ic argument) in the control group, then the test is undefined.
`ties`	how should ties be handled, "cons" use a conservative adjustment for ties, "approx" use an approximate adjustment. See details.
`alternative`	direction of alternative hypothesis.
`conf.level`	confidence level
`conf.int`	logical, do confidence intervals
`conf.sided`	character, either 'one.sided' or 'two.sided' (see warning)
`gname`	name for g function, if NULL uses name of inputed g function
`control`	a list of arguments for numeric calculation settings, see `nicqControl`.

Details

The data may be entered in 3 different formats, and the first argument `x` changes depending on which format. When `z` is given then `x` is the vector of failure times from both groups and z gives the group membership of each of those failures. If there is right censoring this may be given using `status`, and the `nicq` function will make sure that the censoring happens late enough so that the test can still be calculated. When `yc` is given then `x` is a vector of failure times in the test group and `yc` are the failure times in the control group. When `ic` is an integer, then `x` represents the number of failure times that have occurred in the test group at or before the time of the icth failure in the control group. In this last format only `nc` (number in control group) and `nt` (number in the test group) must be given.

The confidence interval is calculated on the difference, F2(t0)-F1(t0), where t0 is unknonwn and defined so that F1(t0)=q, with q given.

The responses can be any numeric values, as long as the difference, F2(t0)- F1(t0), is of interest.

For more details see Fay and Follmann (2015).

The confidence intervals for the qth quantile of the control is calculated using the `bpcp` function followed by the `quantile.kmciLR` from the bpcp R package.

Value

An `nicq` object which inherits from `htest` class (the print method for is slightly different). A list with elements:

`statistic`	number of failure in test group at or before the qth quantile of the control group
`parameter`	vector with elements: q (quantile of interest in control group),i (rank of qth quantile), n1 (number in control group), n2 (number in test group)

`p.value`	one-sided p.value
`conf.int`	confidence interval on F2(t0)-F1(t0), may be one- or two-sided, see attributes
`estimate`	vector of estimates. Values are: x2/n2= proportion of failures in test group by i, i/n1= proportion failures in control group by i, 'x2/n2-i/n1'=difference, tau=qth quantile of control (same at t0), lower CL=lower confidence limit for tau, upper CL, conf.level=conf.level for CI on tau
`null.value`	null value for the difference
`alternative`	either 'less' or 'greater'. two.sided is not allowed
`method`	description of test

warning

Since 'two.sided' alternatives are not allowed, the p-values may not match the confidence intervals in the usual way if `conf.sided`='two.sided' (the default). Consider the example below, with `alternative`='less' and `delta0`=0.10. The p-value is 0.04, so we might expect that the upper limit of the 95 percent confidence limit would be less than 0.10, but this is not so because `conf.sided`='two.sided' and we are using the two-sided confidence interval and p is greater than 0.05/2=0.025.

Author(s)

Michael P. Fay (mfay@niaid.nih.gov)

References

Fay, MP and Follmann DA (2015). Non-inferiority Tests for Anti-Infective Drugs using Control Quantiles. (unpublished manuscript).

Examples

```
## if you know that q=0.20 and there are no ties then ic=q*nc=40
nicqTest(66,g=nimDiffOR,delta0=.1,q=.2,nc=200,nt=300,ic=40,conf.int=FALSE)
## examples with confidence intervals may be slower: see
##  demo(nicqTest.examples)
```

`nimDiffOR` *Variable margin functions*

Description

For testing the alternative F2(t)< g(F1(t)). We give several built-in choices for the function g. All functions must be defined in terms of delta and q, where F1(t0)=q and t0 is defined implicitly, and delta = F2(t0) - g(F1(t0)).

Usage

```
nimDiffOR(p, delta = 0.1, q = 0.2)
nimOR(p, delta=0.1, q=0.2)
nimDiff(p,delta=.1, q=NULL)
```

Arguments

`p`	a vector of F1(t) values, where F1(t) is the proportion of control that failed by t.
`q`	the probability associated with the control quantile of interest, not used for calculations in `nimDiff` but needs to be in the call.
`delta`	the difference: F2(t0) - g(F1(t0))

Details

The functions are defined in terms of `delta` and `q` so that the function can change as a function of `delta` and we can use the function to get confidence intervals for delta (defined in terms of q, since q=F1(t0) which defines t0).

Functions should handle vectors of F1(t) values, and the output is a vector of the same length. The results should be between 0 and 1.

The function `nimDiffOR` gives the minimum of the difference (defined by delta) or the odds ratio (defined in terms of q and delta) when delta>0, and the maximum when delta<0.

For plots of the functions see Fay and Follmann (2015).

Value

a vector of values g(F1(t)).

References

Fay, MP and Follmann DA (2015). Non-inferiority Tests for Anti-Infective Drugs using Control Quantiles. (unpublished manuscript).

See Also

`nicqTest`

Examples

```
## notice that the second values, F1(t)=0.20=q,
## all equal
##  q+delta=0.30
nimDiff(c(1:9)/10)
nimOR(c(1:9)/10)
nimDiffOR(c(1:9)/10)
## for delta<0, take max of difference and odds ratio
nimDiffOR(c(1:9)/10,delta=-.1)
```

`powerNicqTest` *Power or Sample Size for Non-inferiority Control Quantile Test*

Description

Function gives power (if n1=NULL) or sample size (if power=NULL). Assumes no ties.

Usage

```
powerNicqTest(n1 = NULL, n2 = NULL, power = NULL,
    sig.level = 0.025, n2.over.n1 = 1, q = 0.2,
    delta0 = 0.1, alternative = c("less", "greater"),
    gnull = nimDiffOR, galt = function(x){x},
    minn=5, maxn = 10^5, ...)
```

Arguments

`n1`	sample size of control group, calculated if NULL
`n2`	sample size of test group. If n1=NULL, n2 is ignored and calculated based on power and n2.over.n1. If power=NULL, then n2=ceiling(n2.over.n1*n1).
`power`	power under galt, calculated if NULL
`sig.level`	significance level
`n2.over.n1`	ratio of sample sizes
`q`	probability associated with control quantile of interest
`delta0`	difference in proportions at control quantile of interest
`alternative`	alternative hypothesis direction, 'less' means F2(t) less than gnull(F1(t)) for some t.
`gnull`	variable margin function under null hypothesis (more formally, at the boundary between the null and alternative hypotheses for the pre-specified hypotheses)
`galt`	variable margin function for which we calculate the power
`minn`	minimum value for sample size for n1, input into `uniroot.integer`.
`maxn`	maximum value for sample size for n1, input into `uniroot.integer`.
`...`	extra arguments passed to `uniroot.integer`.

Details

The function either calculates the power (if n1=NULL) or calculates n1 and n2 (if power=NULL). In the latter case, we use `uniroot.integer` to find the smallest n1 that gives power at least as large as the given power [with n2 defined as ceiling(n2.over.n1*n1)].

Value

a `power.htest` object. A list with elements:

n1	sample size for control group
n2	sample size for test group
delta0	F2(tau)-F1(tau), with tau defined by F1(tau)=q
q	probability associated with tau
sig.level	significance level
power	power under galt
method	character description of method

See Also

`nicqTest`

Examples

```
# to calculate power, leave power=NULL and supply n1 and n2
powerNicqTest(n1=200,n2=300)
# or supply n1 and n2.over.n1
powerNicqTest(n1=200,n2.over.n1=3/2)
## to calculate n1 and n2, supply power
## find minimum n1 that have power greater than 0.80
## takes 13 iterations to find n1=346
## so do not run it here
#powerNicqTest(power=.80,print.steps=TRUE)
```

---

print.brk — *Print Method for* `brk` *or* `nicq` *Object.*

Description

For `brk` did not use `print.htest` because the p-values are just bounds for some values (e.g., p greater than 0.025). So I needed to print the results differently.

For `nicq`, uses `print.htest` except for $estimates has some special printing instructions since there may be some extra confidence intervals on the control quantile of interest.

Usage

```
## S3 method for class 'brk'
print(x, digits = getOption("digits"), prefix = "\t", ...)

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

Arguments

`x`	the `brk` object
`digits`	number of significant digits for printing
`prefix`	prefix below some values
`...`	for passing arguments. In `nicq` passed to `print.htest`.

Value

Does not print out FullResults list because it is generally too large.

Index