

Package: nivm (via r-universe)

August 23, 2024

Type Package

Title Noninferiority Tests with Variable Margins

Version 0.5

Date 2024-01-25

Author Michael P. Fay

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

Depends stats, bcp, ssnv

Description Noninferiority tests for difference in failure rates at a prespecified control rate or prespecified time. For details, see Fay and Follmann, 2016 <[DOI:10.1177/1740774516654861](https://doi.org/10.1177/1740774516654861)>.

License GPL (>= 3)

NeedsCompilation no

Repository CRAN

Date/Publication 2024-01-25 22:10:02 UTC

Contents

nivm-package	2
brkControl	3
brkTest	3
findPowerR	5
fmectest	5
nicqControl	7
nicqTest	8
nimDiffOR	10
powerNicqTest	12
print.brk	13

Index	15
--------------	-----------

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: 0.5
Date: 2024-01-25
License: >=GPL3

Author(s)

Michael P. Fay

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

References

Fay, MP and Follmann DA (2016). Non-inferiority Tests for Anti-Infective Drugs using Control Group Quantiles. *Clinical Trials*. 13(6): 632-640.

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 Investigations on non-inferiority—the Food and Drug Administration draft guidance on treatments for nosocomial pneumonia as a case for exact tests for binomial proportions.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	<i>Get power from brkTest.</i>
------------	--

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 <code>brk.test</code> , 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	<i>Odds Ratio/Difference Non-inferiority tests</i>
----------	--

Description

Rohmel and Keier (2013) developed these non-inferiority 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 `fmectestExact`, 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"))
```

```
fmectestExact(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 threshold and Farrington and Manning p-value above threshold).
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 q th quantile of the control group has failed. Uses a variable margin function, and the time of the q th quantile of the control group is unknown.

If the cumulative distributions for the two groups are F_1 (control) and F_2 (test), then we are interested in the difference: $\delta = F_2(t_0) - F_1(t_0)$, where $F_1(t_0) = q$. Note F_1, F_2 are unknown and non-parametric, and t_0 is unknown. In this case, using a constant δ does not give practical non-inferiority margins, therefore we use a variable margin function, so that we test (when alternative='less') $H_0: F_2(t) \geq g(F_1(t))$ versus $H_1: F_2(t) < g(F_1(t))$ 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

- | | |
|---------------------|---|
| <code>x</code> | either a vector of failure times for the both groups (when <code>z</code> is given), a vector of failure times for the test group (when <code>yc</code> is given), or the number of failures in the test group that have occurred by the <code>ic</code> th failure in the control group (when <code>ic</code> is an integer, <code>nc</code> and <code>nt</code> are given). See details. |
| <code>delta0</code> | difference, $F_2(t_0) - F_1(t_0)$, on the boundary between the null and alternative hypotheses, where t_0 is defined so that $F_1(t_0) = q$. |
| <code>q</code> | probability associated with the quantile of interest in control group |
| <code>g</code> | non-inferiority margin function. Must have arguments <code>q</code> (representing the quantile of interest in the control group) and <code>delta</code> (representing the difference, $F_2(t_0) - F_1(t_0)$, where F_2 and F_1 are the cumulative distributions of failures for the test (F_2) and control (F_1) at t , where $F_1(t) = q$). Default values for <code>q</code> and <code>delta</code> are ignored. Default function is <code>nimDiffOR</code> . |
| <code>yc</code> | vector of failure times in the control group. If given, <code>x</code> is the vector of failure times in the test group. |
| <code>nc</code> | number of individuals in the control group. Not needed if <code>z</code> or <code>yc</code> is given. |

<code>nt</code>	number of individuals in the test group. Not needed if <code>z</code> or <code>yc</code> is given.
<code>ic</code>	used to find <code>i</code> . The test is based on the number of failures in the test group that have occurred by the <code>i</code> th failure in the control group. <code>ic="prop"</code> gives <code>i=ceiling(q*nc)</code> , <code>ic="maxpower"</code> gives the <code>i</code> value that maximizes the power given $F_1=F_2$ and <code>g</code> , and <code>ic</code> =a positive integer gives <code>i=ic</code> (with <code>ic</code> between 1 and <code>nc</code> inclusive).
<code>z</code>	a vector of group indicators, with either 1 (for control) or 2 (for test). If given, <code>x</code> is a vector of all failures in both groups.
<code>status</code>	a vector denoting right censoring (0) or not (1). Not needed if there is no censoring. Only used when <code>z</code> is given. If any censoring occurs at or before the <code>i</code> th failure (see <code>ic</code> argument) in the control group, then the test is undefined.
<code>ties</code>	how should ties be handled, "cons" use a conservative adjustment for ties, "approx" use an approximate adjustment. See details.
<code>alternative</code>	direction of alternative hypothesis.
<code>conf.level</code>	confidence level
<code>conf.int</code>	logical, do confidence intervals
<code>conf.sided</code>	character, either 'one.sided' or 'two.sided' (see warning)
<code>gname</code>	name for <code>g</code> function, if NULL uses name of inputted <code>g</code> function
<code>control</code>	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 `ic`th 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, $F_2(t_0)-F_1(t_0)$, where t_0 is unknown and defined so that $F_1(t_0)=q$, with `q` given.

The responses can be any numeric values, as long as the difference, $F_2(t_0)-F_1(t_0)$, is of interest.

For more details see Fay and Follmann (2015).

The confidence intervals for the `q`th 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:

<code>statistic</code>	number of failure in test group at or before the <code>q</code> th quantile of the control group
<code>parameter</code>	vector with elements: <code>q</code> (quantile of interest in control group), <code>i</code> (rank of <code>q</code> th quantile), <code>n1</code> (number in control group), <code>n2</code> (number in test group)

p.value	one-sided p.value
conf.int	confidence interval on $F_2(t_0) - F_1(t_0)$, may be one- or two-sided, see attributes
estimate	vector of estimates. Values are: $x_2/n_2 =$ proportion of failures in test group by i , $i/n_1 =$ proportion failures in control group by i , $'x_2/n_2 - i/n_1'$ =difference, $\tau = q$ th quantile of control (same at t_0), lower CL=lower confidence limit for τ , upper CL, $\text{conf.level} = \text{conf.level}$ for CI on τ
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 $\text{conf.sided} = \text{'two.sided'}$ (the default). Consider the example below, with $\text{alternative} = \text{'less'}$ and $\text{delta} = 0.10$. 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 $\text{conf.sided} = \text{'two.sided'}$ 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 (2016). Non-inferiority Tests for Anti-Infective Drugs using Control Group Quantiles. *Clinical Trials*. 13(6): 632-640.

Examples

```
## if you know that q=0.20 and there are no ties then ic=q*nc=40
nicqTest(66,g=nimDiffOR,delta=0.1,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 $F_2(t) < g(F_1(t))$. We give several built-in choices for the function g . All functions must be defined in terms of delta and q , where $F_1(t_0) = q$ and t_0 is defined implicitly, and $\text{delta} = F_2(t_0) - g(F_1(t_0))$.

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 (2016). Non-inferiority Tests for Anti-Infective Drugs using Control Group Quantiles. (to appear in Clinical Trials).

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

<code>n1</code>	sample size of control group, calculated if <code>NULL</code>
<code>n2</code>	sample size of test group. If <code>n1=NULL</code> , <code>n2</code> is ignored and calculated based on power and <code>n2.over.n1</code> . If <code>power=NULL</code> , then <code>n2=ceiling(n2.over.n1*n1)</code> .
<code>power</code>	power under <code>galt</code> , calculated if <code>NULL</code>
<code>sig.level</code>	significance level
<code>n2.over.n1</code>	ratio of sample sizes
<code>q</code>	probability associated with control quantile of interest
<code>delta0</code>	difference in proportions at control quantile of interest
<code>alternative</code>	alternative hypothesis direction, 'less' means $F_2(t)$ less than $g_{null}(F_1(t))$ for some t .
<code>gnull</code>	variable margin function under null hypothesis (more formally, at the boundary between the null and alternative hypotheses for the pre-specified hypotheses)
<code>galt</code>	variable margin function for which we calculate the power
<code>minn</code>	minimum value for sample size for <code>n1</code> , input into uniroot.integer .
<code>maxn</code>	maximum value for sample size for <code>n1</code> , input into uniroot.integer .
<code>...</code>	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	$F_2(\tau) - F_1(\tau)$, with τ defined by $F_1(\tau) = q$
q	probability associated with τ
sig.level	significance level
power	power under g_0
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

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

Value

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

Index

* **h**test

- brkTest, 3
- fmectest, 5
- nicqTest, 8
- nivm-package, 2
- powerNicqTest, 12

* **m**isc

- brkControl, 3
- findPowerR, 5
- nicqControl, 7
- nimDiffOR, 10
- print.brk, 13

bpcp, 7, 9

brkControl, 3, 4

brkTest, 2, 3, 3, 5

findPowerR, 5

fmectest, 2

fmectest (fmectest), 5

fmectest, 2, 5

getfx2, 7

getimaxpower, 7

nicqControl, 7, 9

nicqTest, 2, 4, 6, 7, 8, 11, 13

nimDiff (nimDiffOR), 10

nimDiffOR, 8, 10

nimOR (nimDiffOR), 10

nivm (nivm-package), 2

nivm-package, 2

powerNicqTest, 12

print.brk, 13

print.nicq (print.brk), 13

quantile.kmciLR, 7, 9

uniroot.integer, 12