Package 'abc.n'

May 13, 2013

Version 1.0-0

Date 2013/02/15

Title abc.n

Author {Oliver Ratmann, Anton Camacho}

Maintainer Oliver Ratmann <o.ratmann@imperial.ac.uk>

Depends R (>= 2.10)

Description tools for ABC based on n summary values

License GPL (>=2)

LazyLoad yes

Collate 'nabc_functions.R''KLdiv_functions.R'

R topics documented:

`dchisqstretch_lkl` *Compute the density of the (possibly truncated) summary likelihood for population dispersion of normal summary values*

Description

Compute the density of the (possibly truncated) summary likelihood for population dispersion of normal summary values

Usage

```
dchisqstretch_lkl(rho, n.of.x, s.of.x, norm = 1,
  support = c(0, Inf), log = FALSE)
```

Arguments

`rho`	vector of quantile
`norm`	scalar, 0<norm<=1, normalization constant for the truncated summary likelihood.
`support`	vector of dimension 2, support of the truncated summary likelihood.
`log`	logical; if `TRUE`, densities d are given as log(d).
`n.of.x`	number of observed summary values
`s.of.x`	standard deviation of observed summary values

Note

The summary likelihood can be truncated to `support` and then standardized with `norm`. For computational efficiency, both `norm` and `support` must be provided although each one can be derived from the other. `support=qigamma(c(1-norm,1+norm)/2,(n.of.x-2)/2,s.of.x^2*(n.of.x-1)/2)` and `norm=diff(pigamma(support,(n.of.x-2)/2,s.of.x^2*(n.of.x-1)/2)`.

`dchisqstretch_pow` *Compute the density of the (possible truncated) power of the equivalence test for population dispersion of normal summary values*

Description

Compute the density of the (possible truncated) power of the equivalence test for population dispersion of normal summary values

Usage

```
dchisqstretch_pow(rho, scale, df, c.l, c.u, norm = 1,
  support = c(0, Inf), log = FALSE)
```

Arguments

`rho`	vector of quantile
`scale`	scaling of T apart from rho, either n-1 for unbiased ABC or n for exact MAP
`df`	degrees of freedom
`c.l,c.u`	lower and upper ABC tolerance
`norm`	normalization constant for the truncated power function.
`support`	vector of dimension 2. Support of the truncated power function.
`log`	logical; if `TRUE`, densities d are given as log(d).

Note

The summary likelihood can be truncated to `support` and then standardized with `norm`. For computational efficiency, both `norm` and `support` must be provided although each one can be derived (numerically) from the other.

`dMuTOST_lkl` *Compute the density of the (possibly truncated) summary likelihood for population means of normal summary values*

Description

Compute the density of the (possibly truncated) summary likelihood for population means of normal summary values

Usage

```
dMuTOST_lkl(rho, n.of.x, s.of.x, norm = 1,
  support = c(-Inf, Inf), log = FALSE)
```

Arguments

rho	vector of quantile
norm	scalar, 0<norm<=1, normalization constant for the truncated summary likelihood.
support	vector of dimension 2, support of the truncated summary likelihood.
log	logical; if TRUE, densities d are given as log(d).
n.of.x	number of observed summary values
s.of.x	standard deviation of observed summary values

Note

The summary likelihood can be truncated to support and then standardized with norm. For computational efficiency, both norm and support must be provided although each one can be derived from the other. support=s.of.x/sqrt(n.of.x)*qt(c(1-norm,1+norm)/2,n.of.x-1) and norm=diff(pt(support/(s.of.x/sqrt(n.of.x)),n.of.x-1)).

---

dMuTOST_pow *Compute the density of the (possible truncated) power of the equivalence test for population means of normal summary values*

Description

Compute the density of the (possible truncated) power of the equivalence test for population means of normal summary values

Usage

```
dMuTOST_pow(rho, df, s.of.T, tau.u, alpha, norm = 1,
  support = c(-Inf, Inf), log = FALSE)
```

Arguments

rho	vector of quantile
norm	normalization constant for the truncated power function.
support	vector of dimension 2. Support of the truncated power function.
log	logical; if TRUE, densities d are given as log(d).
tau.u	upper tolerance of the equivalence region. If calibrate.tau.u==TRUE, it's just a guess on an upper bound on the upper tolerance of the equivalence region.
alpha	level of the equivalence test
df	degrees of freedom of the simulated summary values
s.of.T	standard deviation of the test statistic

Note

The summary likelihood can be truncated to support and then standardized with norm. For computational efficiency, both norm and support must be provided although each one can be derived (numerically) from the other.

integrand_KL_divergence_1D

Integrand of the Kullback-Leibler divergence D(P||Q) for generic distributions

Description

Integrand of the Kullback-Leibler divergence D(P||Q) for generic distributions

Usage

```
integrand_KL_divergence_1D(x, dP, dQ, P_arg, Q_arg)
```

Arguments

x	value at which integrand is evaluated. Can be a vector.
dP,dQ	name of the functions that compute the density of P and Q
P_arg,Q_arg	list of arguments for dP and dQ

KL_divergence_chisqstretch

Compute Kullback-Leibler divergence between the summary likelihood and the power function of chisqstretch

Description

Compute Kullback-Leibler divergence between the summary likelihood and the power function of chisqstretch

Usage

```
KL_divergence_chisqstretch(n.of.x, s.of.x, n.of.y,
  s.of.y, mx.pw, alpha, calibrate.tau.u = F, tau.u = 1,
  for.mle = 0, pow_scale = 1.5, debug = 0, plot = F)
```

Arguments

n.of.x	number of observed summary values
s.of.x	standard deviation of observed summary values
n.of.y	number of simulated summary values
s.of.y	standard deviation of simulated summary values
mx.pw	maximum power at the point of reference (rho.star=0) (only when calibrate.tau.u==TRUE).
alpha	level of the equivalence test
calibrate.tau.u	if TRUE the upper tolerance of the equivalence region (tau.u) is calibrated so that power at the point of reference is equal to mx.pw

tau.u	upper tolerance of the equivalence region. If calibrate.tau.u==TRUE, tau.u is just a guess on an upper bound on the upper tolerance of the equivalence region.
for.mle	calibrate so that the mode of the power is at the MLE
pow_scale	scale for the support of the standardized power. The power is truncated between [tau.l/pow_scale,tau.u*pow_scale] and then standardized.
debug	flag if C implementation is used
plot	whether to plot the two distributions

Value

vector of length 6

KL_div	the Kullback Leibler divergence
tau.l	lower tolerance of the equivalence region
tau.u	upper tolerance of the equivalence region
c.l	lower ABC tolerance
c.u	upper ABC tolerance
pw.cmx	actual maximum power associated with the equivalence region

Note

Whatever the value of calibrate.tau.u, the lower tolerance of the equivalence region (tau.l) is always numerically calibrated using nabc.chisqstretch.tau.low.

Examples

```
KL_divergence_chisqstretch(n.of.x=60,s.of.x=0.1,n.of.y=60,s.of.y=0.3, mx.pw=0.9,
alpha=0.01, calibrate.tau.u=T, tau.u=1, plot=T)
```

KL_divergence_mutost *Compute Kullback-Leibler divergence between the summary likelihood and the power function of mutost*

Description

Compute Kullback-Leibler divergence between the summary likelihood and the power function of mutost

Usage

```
KL_divergence_mutost(n.of.x, s.of.x, n.of.y, s.of.y,
  mx.pw, alpha, calibrate.tau.u = F, tau.u = 1,
  pow_scale = 1.5, debug = 0, plot = F)
```

Arguments

n.of.x	number of observed summary values
s.of.x	standard deviation of observed summary values
n.of.y	number of simulated summary values
s.of.y	standard deviation of simulated summary values
mx.pw	maximum power at the point of reference (rho.star=0) (only when calibrate.tau.u==TRUE).
alpha	level of the equivalence test
calibrate.tau.u	if TRUE the upper tolerance of the equivalence region (tau.u) is calibrated so that power at the point of reference is equal to mx.pw
tau.u	upper tolerance of the equivalence region. If calibrate.tau.u==TRUE, it's just a guess on an upper bound on the upper tolerance of the equivalence region.
pow_scale	scale for the support of the standardized power. The power is truncated between pow_scale*[-tau.u,tau.u] and then standardized.
debug	flag if C implementation is used
plot	whether to plot the two distributions

Value

vector of length 3

KL_div	the Kullback Leibler divergence
tau.u	upper tolerance of the equivalence region
pw.cmx	actual maximum power associated with the equivalence region

Examples

```
KL_divergence_mutost(n.of.x=60,s.of.x=0.1,n.of.y=60,s.of.y=0.3, mx.pw=0.9,
alpha=0.01, calibrate.tau.u=T, tau.u=1, plot=T)
```

KL_divergence_optimize_1D

A wrapper to minimize KL_divergence *over the parameter* x_name *using the function optimize*

Description

A wrapper to minimize KL_divergence over the parameter x_name using the function optimize

Usage

```
KL_divergence_optimize_1D(x_value, x_name,
  is_integer = FALSE, KL_divergence, args,
  verbose = FALSE)
```

Arguments

x_value	value tested.
x_name	name of the parameter over which minimization is performed.
is_integer	if TRUE, x_value is rounded to the closest integer. See round.
KL_divergence	character, name of the function that computes the KL divergence.
args	additional arguments to be passed to KL_divergence.
verbose	logical, if TRUE, print warnings.

Value

the minimized Kullback-Leibler divergence (scalar).

---

nabc.acf.equivalence *Perform the asymptotic equivalence test for autocorrelations at lag 1*

Description

Perform the asymptotic equivalence test for autocorrelations at lag 1

Usage

```
nabc.acf.equivalence(sim, obs, args = NA,
  verbose = FALSE, alpha = 0, leave.out = 0,
  normal.test = "sf.test")
```

Arguments

sim	simulated summary values
obs	observed summary values
args	argument that contains the equivalence region and the level of the test (see Examples). This is the preferred method for specifying arguments and overwrites the dummy default values
verbose	flag if detailed information on the computations should be printed to standard out
alpha	level of the equivalence test
leave.out	thinning, how many values in the pair sequence (x_i,x_i-1) should be left out. Defaults to zero.
normal.test	name of function with which normality of the summary values is tested

Value

vector containing

error	test statistic. Here, instead of T we return the p-value of the TOST.
cil	lower ABC tolerance. Here, instead of c^- we return 0.
cir	upper ABC tolerance. Here, instead of c^+ we return 'alpha'.
al	free entry. Here set to c^-.
ar	free entry. Here set to c^+.
mx.pw	Maximum power at the point of equality
rho.mc	sample estimate of 'rho'

Examples

```
leave.out<- 2
tau.u<- 0.09
alpha<- 0.01
n<- 5e3
sigma<- 1
a<- 0.1
args<- paste("acfequiv",leave.out,tau.u,alpha,sep='/')
x<-rnorm(n+1,0,sigma)
x<- x[-1] + x[-(n+1)]*a
y<-rnorm(n+1,0,sigma)
y<- y[-1] + y[-(n+1)]*a
nabc.acf.equivalence(y,x,args)
```

---

nabc.acf.equivalence.abctol

Compute the ABC tolerances of the asymptotic equivalence test for autocorrelations at lag 1

Description

Compute the ABC tolerances of the asymptotic equivalence test for autocorrelations at lag 1

Usage

```
nabc.acf.equivalence.abctol(tau.l, tau.u, n, alpha)
```

Arguments

tau.l	lower tolerance of the equivalence region
tau.u	upper tolerance of the equivalence region
n	number of pairs (x_i,x_i-1) after thinning of the time series x_1, x_2, ...
alpha	level of the equivalence test

Value

vector of length 2, first entry is lower ABC tolerance, second entry is upper ABC tolerance

Examples

```
tau.u<- 0.09
tau.l<- -tau.u
sim.n<-5e3
leave.out<- 2
nabc.acf.equivalence.abctol(tau.l, tau.u, floor(sim.n / (1+leave.out)), 0.01)
```

nabc.acf.equivalence.cor

Compute the autocorrelation in a time series along with some other info

Description

Compute the autocorrelation in a time series along with some other info

Usage

```
nabc.acf.equivalence.cor(x, leave.out = 0)
```

Arguments

x	time series (simply vector)
leave.out	thinning, how many values in the pair sequence (x_i,x_i-1) should be left out. Defaults to zero.

Value

cor	autocorrelation in the thinned sequence
z	Z-transformation of the autocorrelation (this is atanh of "cor")
n	Number of pairs (x_i,x_i-1) after thinning

Examples

```
nabc.acf.equivalence.cor(rnorm(100,0,1), leave.out=2)
```

nabc.acf.equivalence.pow

Compute power of the asymptotic equivalence test for autocorrelations at lag 1

Description

Compute power of the asymptotic equivalence test for autocorrelations at lag 1

Usage

```
nabc.acf.equivalence.pow(rho, tau.u, alpha, s)
```

Arguments

rho	true difference in simulated and observed autocorrelation at lag 1
tau.u	upper tolerance of the equivalence region
alpha	level of the equivalence test
s	standard deviation of the test statistic

Value

power of the asymptotic test. this is approximate because the test is asymptotic

Examples

```
tau.u<- 0.09
tau.l<- -tau.u
sim.n<-5e3
rho<- seq(tau.l,tau.u,0.001)
pw<- nabc.acf.equivalence.pow(rho, tau.u, alpha, 1/sqrt(floor(sim.n/3)-3))
```

---

nabc.acf.equivalence.tau.lowup

Calibrate the equivalence region of the asymptotic equivalence test for autocorrelations at lag 1 for given maximum power

---

Description

Calibrate the equivalence region of the asymptotic equivalence test for autocorrelations at lag 1 for given maximum power

Usage

```
nabc.acf.equivalence.tau.lowup(mx.pw, tau.up.ub, n,
  alpha, rho.star = 0, tol = 1e-05, max.it = 100)
```

Arguments

mx.pw	maximum power at the point of reference (rho.star).
tau.up.ub	guess on an upper bound on the upper tolerance of the equivalence region
n	number of pairs (x_i,x_i-1) after thinning of the time series x_1, x_2, ...
alpha	level of the equivalence test
rho.star	point of reference. Defaults to the point of equality rho.star=0.
tol	this algorithm stops when the actual maximum power is less than 'tol' from 'mx.pw'
max.it	this algorithm stops prematurely when the number of iterations to find the equivalence region exceeds 'max.it'

Value

vector of length 4

1	lower tolerance of the equivalence region
2	upper tolerance of the equivalence region
3	actual maximum power associated with the equivalence region
4	error ie abs(actual power - mx.pw)

Examples

```
tau.u<- 0.09
tau.l<- -tau.u
sim.n<-5e3
leave.out<- 2
nabc.acf.equivalence.tau.lowup(0.9, 2, floor(sim.n / (1+leave.out)), 0.01)
```

`nabc.adjust.n.of.y.KLdiv`

Calibrate the number of simulated summary values and the equivalence region for a specified test of equivalence by minimising the Kullback-Leibler divergence from the standardized power function to the summary likelihood.

Description

Calibrate the number of simulated summary values and the equivalence region for a specified test of equivalence by minimising the Kullback-Leibler divergence from the standardized power function to the summary likelihood.

Usage

```
nabc.adjust.n.of.y.KLdiv(KL_divergence, args,
  max.it = 100, debug = 0, plot = F)
```

Arguments

`KL_divergence`	character, name of the function to compute the KL divergence for the test of equivalence. This function must have at least two arguments: `n.of.y`, the number of simulated summaries, and `plot`, which flag some plotting options.
`args`	list of argument to be passed to `KL_divergence`.
`max.it`	this algorithm stops prematurely when the number of iterations to calibrate the number of simulated data points exceeds 'max.it'
`debug`	flag if C implementation is used.
`plot`	if `TRUE`, plot the result of minimization. Only if implemented in `KL_divergence`.

Value

vector of variable length. The first value is `n.of.y`: the adjusted number of simulated summaries. The rest is returned from `KL_divergence`.

Examples

```
xn <- 60
yn <- xn
xmean <- 1
xsigma <- 1
ymean <- 1
ysigma <- 2

obs <- rnorm(xn, xmean, xsigma)
```

```
obs <- (obs - mean(obs))/sd(obs) * xsigma + xmean
sim <- rnorm(yn, ymean, ysigma)

n.of.x <- xn
s.of.x <- sd(obs)
n.of.y <- yn
s.of.y <- sd(sim)
mx.pw <- 0.9
alpha <- 0.01
tau.u.ub <- 2

## example 1: test of location equivalence for normally distributed variables (muTOST)
#compute the Kullback-Leibler divergence between the summary likelihood and the standardized power; and plo
KL_divergence_mutost(n.of.x, s.of.x, n.of.y, s.of.y, mx.pw, alpha, calibrate.tau.u = T, tau.u = tau.u.ub, p

#adjust n.of.y to minimize the Kullback-Leibler divergence, and plot result.
nabc.adjust.n.of.y.KLdiv("KL_divergence_mutost", args = list(n.of.x = n.of.x, s.of.x = s.of.x, n.of.y = n.o
mx.pw = mx.pw, alpha = alpha, calibrate.tau.u = T, tau.u = tau.u.ub), plot = T)

## example 2: test of dispersion equivalence for normally distributed variables (chisqstretch)
#compute the Kullback-Leibler divergence between the summary likelihood and the standardized power; and plo
KL_divergence_chisqstretch(n.of.x, s.of.x, n.of.y, s.of.y, mx.pw, alpha, calibrate.tau.u = T, tau.u = tau.u

#adjust n.of.y to minimize the Kullback-Leibler divergence, and plot result.
nabc.adjust.n.of.y.KLdiv("KL_divergence_chisqstretch", args = list(n.of.x = n.of.x, s.of.x = s.of.x, n.of.y
s.of.y = s.of.y, mx.pw = mx.pw, alpha = alpha, calibrate.tau.u = T, tau.u = tau.u.ub), plot = T)
```

`nabc.adjust.tau.lowup.KLdiv` *Calibrate the equivalence region for a specified test of equivalence by minimising the Kullback-Leibler divergence between the power function and the summary likelihood.*

Description

Calibrate the equivalence region for a specified test of equivalence by minimising the Kullback-Leibler divergence between the power function and the summary likelihood.

Usage

```
nabc.adjust.tau.lowup.KLdiv(KL_divergence, args,
  tau.u.lb = 1, max.it = 100, debug = 0, plot = F)
```

Arguments

`KL_divergence`	character, name of the function to compute the KL divergence for the test of equivalence. This function must have at least two arguments: `tau.u`, the upper tolerance of the equivalence region, and `plot`, which flag some plotting options.
`tau.u.lb`	guess on an lower bound on the upper tolerance of the equivalence region
`args`	list of argument to be passed to `KL_divergence`.

max.it	this algorithm stops prematurely when the number of iterations to calibrate the number of simulated data points exceeds 'max.it'
debug	flag if C implementation is used.
plot	if TRUE, plot the result of minimization. Only if implemented in KL_divergence.

Value

vector of variable length. The first value is n.of.y: the number of simulated summaries. The rest is returned from KL_divergence.

Note

This procedure is not relevant for the test of dispersion equivalence for normally distributed variables (chisqstretch).

Examples

```
xn <- 60
yn <- xn
xmean <- 2
xsigma <- 1
ymean <- 2
ysigma <- 0.5

obs <- rnorm(xn, xmean, xsigma)
obs <- (obs - mean(obs))/sd(obs) * xsigma + xmean
sim <- rnorm(yn, ymean, ysigma)

n.of.x <- xn
s.of.x <- sd(obs)
n.of.y <- yn
s.of.y <- sd(sim)
mx.pw <- 0.9
alpha <- 0.01
tau.u.ub <- 2

## test of location equivalence for normally distributed variables (muTOST)
#compute the Kullback-Leibler divergence between the summary likelihood and the standardized power; and plo
tmp <- KL_divergence_mutost(n.of.x, s.of.x, n.of.y, s.of.y, mx.pw, alpha, calibrate.tau.u = T, tau.u = tau.
plot = T)
print(tmp)

#adjust tau.u to minimize the Kullback-Leibler divergence, and plot result.
nabc.adjust.tau.lowup.KLdiv("KL_divergence_mutost", args = list(n.of.x = n.of.x, s.of.x = s.of.x, n.of.y =
s.of.y = s.of.y, mx.pw = mx.pw, alpha = alpha), tau.u.lb = tmp["tau.u"], plot = T)
```

---

nabc.chisqstretch *Perform the exact test for dispersion equivalence when the summary values are normally distributed*

Description

Perform the exact test for dispersion equivalence when the summary values are normally distributed

Usage

```
nabc.chisqstretch(sim, obs.mc, args = NA,
  verbose = FALSE, tau.l = 1, tau.u = 1, guess.tau.l = 0,
  alpha = 0, normal.test = "sf.test", for.mle = 0)
```

Arguments

sim	simulated summary values
obs.mc	variance of the observed summary values
args	argument that contains the equivalence region and the level of the test (see Examples). This is the preferred method for specifying arguments and overwrites the dummy default values
verbose	flag if detailed information on the computations should be printed to standard out
tau.l	lower tolerance of the equivalence region
tau.u	upper tolerance of the equivalence region
guess.tau.l	guess on the lower tolerance of the equivalence region. Used when the tolerances are annealed and calibration is numerically unstable.
alpha	level of the equivalence test
leave.out	thinning, how many values in the pair sequence (x_i,x_i-1) should be left out. Defaults to zero.
normal.test	name of function with which normality of the summary values is tested
for.mle	calibrate so that the mode of the power is at the MLE

Value

vector containing

error	test statistic, here var(sim)/obs.mc
cil	lower ABC tolerance c^-
cir	upper ABC tolerance c^+
mx.pw	Maximum power at the point of equality
rho.mc	log(var(sim) / obs.mc)

Examples

```
alpha<- 0.01; xn<- yn<- 60; xsigma2<- 1; tau.u<- 2.2
tau.l<- nabc.chisqstretch.tau.low(tau.u, yn-1, alpha)
args<- paste("chisqstretch",tau.l,tau.u,alpha,sep='/')
x<- rnorm(xn,0,sd=sqrt(xsigma2))
y<- rnorm(yn,0,sd=sqrt(xsigma2))
nabc.chisqstretch(y, var(x), args=args, verbose= 0)
```

`nabc.chisqstretch.n.of.y` *Calibrate the number of simulated summary values and the equivalence region for the test of dispersion equivalence*

Description

Calibrate the number of simulated summary values and the equivalence region for the test of dispersion equivalence

Usage

```
nabc.chisqstretch.n.of.y(n.of.x, s.of.Sx, mx.pw, alpha,
  tau.u.ub = 2, tol = 1e-05, max.it = 100, for.mle = 0)
```

Arguments

`n.of.x`	number of observed summary values
`s.of.Sx`	standard deviation in the observed summary likelihood
`mx.pw`	maximum power at the point of reference (rho.star).
`alpha`	level of the equivalence test
`tau.up.ub`	guess on an upper bound on the upper tolerance of the equivalence region
`tol`	this algorithm stops when the actual variation in the ABC approximation to the summary likelihood is less than 'tol' from 's.of.Sx*s.of.Sx'
`max.it`	this algorithm stops prematurely when the number of iterations to calibrate the number of simulated data points exceeds 'max.it'
`for.mle`	calibrate so that the mode of the power is at the MLE

Value

vector of length 8

1	number of simulated summary values
2	lower tolerance of the equivalence region
3	upper tolerance of the equivalence region
4	lower ABC tolerance c^-
5	upper ABC tolerance c^+
6	actual variation of the power
7	actual maximum power associated with the equivalence region
8	error ie abs(actual variation - variation in the observed summary likelihood)

Examples

```
xn<- 60; alpha <- 0.01; prior.u <- 3; prior.l <- 1/3; tau.u<- 2.5; xsig2 <- 1
#summary likelihood of sigma2 given sample mean and sum of squares
th <- seq(prior.l,prior.u,length.out=1e3)
shape <- (xn-2)/2
scale <- xsig2*xn*xn/(xn-1)/2
y <- densigamma(th, shape, scale)
var.Sx <- scale*scale/((shape-1)*(shape-1)*(shape-2))
#abc approximation to summary likelihood
nabc.chisqstretch.n.of.y(xn, sqrt(var.Sx), 0.9, alpha, tau.u.ub=tau.u)
yn <- tmp[1]
tau.l <- tmp[2]
tau.u <- tmp[3]
c.l <- tmp[4]
c.u <- tmp[5]
y2 <- nabc.chisqstretch.pow(th, yn-1, yn-1, c.l, c.u)
#plot the summary likelihood and the abc approximation
plot(th,y/mean(y),ylim=range(c(y/mean(y),y2/mean(y2))),type='l')
lines(th,y2/mean(y2),col="blue")
```

`nabc.chisqstretch.pow` *Compute power of the exact equivalence test for dispersion*

Description

Compute power of the exact equivalence test for dispersion

Usage

```
nabc.chisqstretch.pow(rho, scale, df, cl, cu)
```

Arguments

`rho`	true ratio in simulated variance / observed variance
`scale`	scaling of T apart from rho, either n-1 for unbiased ABC or n for exact MAP
`df`	degrees of freedom
`cl`	lower ABC tolerance
`cu`	upper ABC tolerance

Value

power of the exact test. this is exact.

Examples

```
alpha<- 0.01
tau.up<- 1.09
yn<- 5e3
tau.low<- nabc.chisqstretch.tau.low(tau.up, yn-1, alpha)
rej<- .Call("abcScaledChiSq",c(yn-1,yn-1,tau.low,tau.up,alpha,1e-10,100,0.05) )
rho<- seq(tau.low,tau.up,by=0.001)
nabc.chisqstretch.pow(rho,yn-1,yn-1,rej[1],rej[2])
```

nabc.chisqstretch.tau.low

Calibrate the lower tolerance interval of the equivalence region for the test of dispersion equivalence

Description

Calibrate the lower tolerance interval of the equivalence region for the test of dispersion equivalence

Usage

```
nabc.chisqstretch.tau.low(tau.up, df, alpha,
  rho.star = 1, tol = 1e-05, max.it = 100, for.mle = 0)
```

Arguments

tau.up	upper tolerance of the equivalence region
df	degrees of freedom
alpha	level of the equivalence test
rho.star	point of reference. Defaults to the point of equality rho.star=1
tol	this algorithm stops when the actual point of reference is less than 'tol' from 'rho.star'
max.it	this algorithm stops prematurely when the number of iterations to find the equivalence region exceeds 'max.it'
for.mle	calibrate so that the mode of the power is at the MLE

Value

tau.low, lower tolerance of the equivalence region

Examples

```
tau.u<- 2.2
 yn<- 60
tau.l<- nabc.chisqstretch.tau.low(tau.u, yn-1, 0.01)
```

nabc.chisqstretch.tau.lowup

Calibrate the equivalence region for the test of dispersion equivalence for given maximum power

Description

Calibrate the equivalence region for the test of dispersion equivalence for given maximum power

Usage

```
nabc.chisqstretch.tau.lowup(mx.pw, tau.up.ub, df, alpha,
  rho.star = 1, tol = 1e-05, max.it = 100, for.mle = 0)
```

Arguments

mx.pw	maximum power at the point of reference (rho.star).
tau.up.ub	guess on an upper bound on the upper tolerance of the equivalence region
df	degrees of freedom
alpha	level of the equivalence test
rho.star	point of reference. Defaults to the point of equality rho.star=1.
tol	this algorithm stops when the actual maximum power is less than 'tol' from 'mx.pw'
max.it	this algorithm stops prematurely when the number of iterations to find the equivalence region exceeds 'max.it'
for.mle	calibrate so that the mode of the power is at the MLE

Value

vector of length 6

1	lower tolerance of the equivalence region
2	upper tolerance of the equivalence region
3	actual maximum power associated with the equivalence region
4	error ie abs(actual power - mx.pw)
5	lower point of critical region
6	upper point of critical region

Examples

```
yn<- 60
nabc.chisqstretch.tau.lowup(0.9, 2.5, yn-1, 0.01)
```

NABC.DEFAULT.ANS *this file contains all R functions of the abc-n package*

Description

this file contains all R functions of the abc-n package

Usage

```
NABC.DEFAULT.ANS
```

Format

Named num [1:17] 0 50 1 NA NA NA 0 0 0 0 ... - attr(*, "names")= chr [1:17] "lkl" "error" "pval" "link.mc.obs" ...

nabc.exprho.at.theta *Estimate summary parameter errors rho from unbiased Monte Carlo estimates rho.mc for all proposed theta including rejections*

Description

Estimate summary parameter errors rho from unbiased Monte Carlo estimates rho.mc for all proposed theta including rejections

Usage

```
nabc.exprho.at.theta(df, theta.names, rho.names,
  thin = 1)
```

Arguments

df	data frame with all proposed theta and corresponding rho.mc for each summary of interest
theta.names	vector of theta names (columns in df)
rho.names	vector of rho names (columns in df)
thin	thinning factor in case there are many rows in df

Value

matrix containing the estimated rho (per column). The ith row corresponds to the ith theta in df.

nabc.generic.tost *Perform a generic two one sided test. This is an internal function.*

Description

Perform a generic two one sided test. This is an internal function.

Usage

```
nabc.generic.tost(tost.args, tau.l, tau.u, alpha,
  tost.distr = "t")
```

Arguments

tost.args	vector of arguments for generic TOST
tau.l	lower tolerance of equivalence region
tau.u	upper tolerance of equivalence region
alpha	level of equivalence test
tost.distr	name of distribution of tost

Value

vector of length 7

nabc.get.pfam.pval *Test if summary values are normally distributed*

Description

Test if summary values are normally distributed

Usage

```
nabc.get.pfam.pval(x, normal.test)
```

Arguments

x	summary values
norma.test	name of function with which normality of the summary values is tested

Value

p value of the test

Examples

```
nabc.get.pfam.pval(rnorm(1e4),"shapiro.test")
```

nabc.mutost.onesample *Perform the exact TOST for location equivalence when the summary values are normally distributed*

Description

Perform the exact TOST for location equivalence when the summary values are normally distributed

Usage

```
nabc.mutost.onesample(sim, obs, obs.n = NA, obs.sd = NA,
  args = NA, verbose = FALSE, tau.u = 0, tau.l = -tau.u,
  alpha = 0, mx.pw = 0.9, annealing = 1,
  normal.test = "sf.test", plot = 0, legend.txt = "")
```

Arguments

sim	simulated summary values
obs	observed summary values
args	argument that contains the equivalence region and the level of the test (see Examples). This is the preferred method for specifying arguments and overwrites the dummy default values
verbose	flag if detailed information on the computations should be printed to standard out
s.of.x	standard deviation of the observed summary values

tau.u	upper tolerance of the equivalence region
tau.l	lower tolerance of the equivalence region
alpha	level of the equivalence test
mx.pw	maximum power at the point of equality
annealing	inflation factor of tolerances of the equivalence region
normal.test	name of function with which normality of the summary values is tested

Value

vector containing

error	test statistic, here p-value of TOST
cil	lower ABC tolerance, here 0
cir	upper ABC tolerance, here alpha
mx.pw	Maximum power at the point of equality
rho.mc	mean(sim) - obs.mean

Examples

```
tau.u<- 0.5; tau.l<- -tau.u; alpha<- 0.01; xn<- yn<- 60; xmu<- ymu<- 0.5; xsigma2<- ysigma2<- 2
args<- paste("mutost",1,tau.u,alpha,sep='/')
x<- rnorm(xn,xmu,sd=sqrt(xsigma2))
y<- rnorm(yn,ymu,sd=sqrt(ysigma2))
nabc.mutost.onesample(y, x, args= args, verbose= 0)
```

---

nabc.mutost.onesample.n.of.y

Calibrate the number of simulated summary values and the equivalence region for the test of location equivalence

---

Description

Calibrate the number of simulated summary values and the equivalence region for the test of location equivalence

Usage

```
nabc.mutost.onesample.n.of.y(n.of.x, s.of.Sx, mx.pw,
  s.of.y, alpha, tau.u.ub = 2, tol = 1e-05, max.it = 100,
  debug = 0)
```

Arguments

n.of.x	number of observed summary values
s.of.Sx	standard deviation in the observed summary likelihood
mx.pw	maximum power at the point of reference (rho.star).
s.of.y	standard deviation in the simulated summary values
alpha	level of the equivalence test

tau.up.ub	guess on an upper bound on the upper tolerance of the equivalence region
tol	this algorithm stops when the actual variation in the ABC approximation to the summary likelihood is less than 'tol' from 's.of.Sx*s.of.Sx'
max.it	this algorithm stops prematurely when the number of iterations to calibrate the number of simulated data points exceeds 'max.it'
debug	Flag if C implementation is used.

Value

vector of length 6

1	number of simulated summary values
2	lower tolerance of the equivalence region
3	upper tolerance of the equivalence region
4	actual variation of the power
5	actual maximum power associated with the equivalence region
6	error ie abs(actual variation - variation in the observed summary likelihood)

Examples

```
prior.u<- 2; prior.l<- -prior.u; tau.u<- 0.75; xn<- yn<- 60; xmu<- 0.5; xsigma2<- ysigma2<- 2; alpha<- 0.0
rho<- seq(prior.l,prior.u,length.out=1e3)
#summary likelihood
y<-dnorm(rho,0,sqrt(xsigma2/xn))
y<- y / diff(pnorm(c(prior.l,prior.u),0,sqrt(xsigma2/xn)))
#abc approximation to summary likelihood based on equivalence test
tmp <- nabc.mutost.onesample.n.of.y(xn, sqrt(xsigma2/xn), 0.9, sqrt(ysigma2), alpha, tau.u.ub=2*tau.u )
yn <- tmp[1]
tau.u <- tmp[3]
y2<- nabc.mutost.pow(rho, yn-1, tau.u, sqrt(ysigma2/yn), alpha)
rho2<- rho[which(y2!=0)]
y2<- y2[which(y2!=0)]
y2<- y2/sum(diff(rho2)*y2[-1])
#plot summary likelihood and abc approximation thereof
plot(1,1,type='n',xlim=range(rho),ylim=range(c(y,y2)),xlab=expression(rho))
lines(rho,y,col="red")
lines(rho2,y2,col="blue")
abline(v=0,col="red")
```

---

nabc.mutost.onesample.tau.lowup.pw

Calibrate the equivalence region for the test of location equivalence for given maximum power

---

Description

Calibrate the equivalence region for the test of location equivalence for given maximum power

Usage

```
nabc.mutost.onesample.tau.lowup.pw(mx.pw, df, s.of.T,
  tau.up.ub, alpha, rho.star = 0, tol = 1e-05,
  max.it = 100, debug = 0)
```

Arguments

mx.pw	maximum power at the point of reference (rho.star).
df	degrees of freedom
s.of.T	standard deviation of the test statistic
tau.up.ub	guess on an upper bound on the upper tolerance of the equivalence region
alpha	level of the equivalence test
rho.star	point of reference. Defaults to the point of equality rho.star=0.
tol	this algorithm stops when the actual maximum power is less than 'tol' from 'mx.pw'
max.it	this algorithm stops prematurely when the number of iterations to find the equivalence region exceeds 'max.it'
debug	Flag if C implementation is used.

Value

vector of length 4

1	lower tolerance of the equivalence region
2	upper tolerance of the equivalence region
3	actual maximum power associated with the equivalence region
4	error ie abs(actual power - mx.pw)

Examples

```
yn<- 60; ysigma2<- 1; alpha<- 0.01
nabc.mutost.onesample.tau.lowup.pw(0.9, yn-1, sqrt(ysigma2/yn), 2, alpha )
```

---

nabc.mutost.onesample.tau.lowup.var

Calibrate the equivalence region for the test of location equivalence for given variance of the summary likelihood

---

Description

Calibrate the equivalence region for the test of location equivalence for given variance of the summary likelihood

Usage

```
nabc.mutost.onesample.tau.lowup.var(s.of.Sx, df, s.of.T,
  tau.up.ub, alpha, rho.star = 0, tol = 1e-05,
  max.it = 100, debug = 0)
```

Arguments

s.of.Sx	standard deviation of the summary likelihood
df	degrees of freedom
s.of.T	standard deviation of the test statistic
tau.up.ub	guess on an upper bound on the upper tolerance of the equivalence region
alpha	level of the equivalence test
rho.star	point of reference. Defaults to the point of equality rho.star=0.
tol	this algorithm stops when the actual maximum power is less than 'tol' from 'mx.pw'
max.it	this algorithm stops prematurely when the number of iterations to find the equivalence region exceeds 'max.it'
debug	Flag if C implementation is used.

Value

vector of length 4

1	lower tolerance of the equivalence region
2	upper tolerance of the equivalence region
3	actual variance associated with the power
4	error ie abs(actual var(power) - var(summary likelihood))

Examples

```
yn<- 60; ysigma2<- 1; alpha<- 0.01
nabc.mutost.onesample.tau.lowup.var(0.002, yn-1, sqrt(ysigma2/yn), 2, alpha )
```

---

nabc.mutost.pow — *Compute power of the equivalence test for population means of normal summary values*

Description

Compute power of the equivalence test for population means of normal summary values

Usage

```
nabc.mutost.pow(rho, df, tau.u, s.of.T, alpha,
  rtn.fun = FALSE, force = FALSE)
```

Arguments

rho	true difference in simulated and observed population means
df	degrees of freedom of the simulated summary values
tau.u	upper tolerance of the equivalence region
s.of.T	standard deviation of the test statistic
alpha	level of the equivalence test
rtn.fun	indicator if a function to compute the power should be returned. Defaults to 0.
force	if TRUE, enforce power computation outside of acceptance region

Value

approximate power of the exact test. this is approximate because the standard deviation of the normal model for the simulated summary values is not known.

Examples

```
prior.u<- 5; prior.l<- -prior.u; tau.u <- 0.75; yn<- 60; ysigma2<- 1; alpha<- 0.01
rho <- seq(prior.l,prior.u,length.out=1e3)
nabc.mutost.pow(rho, yn-1, tau.u, sqrt(ysigma2/yn), alpha)
```

Index