Package ‘serieslcb’

October 14, 2022

Type Package

Title Lower Confidence Bounds for Binomial Series System

Version 0.4.0

Author Edward Schuberg

Maintainer Edward Schuberg <eschu003@ucr.edu>

Description Calculate and compare lower confidence bounds for binomial series system reliability. The R 'shiny' application, launched by the function launch_app(), weaves together a workflow of customized simulations and delta coverage calculations to output recommended lower confidence bound methods.

Depends R (>= 3.3.0), shiny

Imports gplots, stats

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-07-09 04:50:10 UTC

R topics documented:

`bayes` *Bayesian method*

Description

Calculate a binomial series lower confidence bound using Bayes' method with a Beta prior distribution.

Usage

```
bayes(s, n, alpha, MonteCarlo, beta.a, beta.b, ...)
```

Arguments

`s`	Vector of successes.
`n`	Vector of sample sizes.
`alpha`	The significance level; to calculate a 100(1-α)% lower confidence bound.
`MonteCarlo`	Number of samples to draw from the posterior distribution for the Monte Carlo estimate.
`beta.a`	Shape1 parameter for the Beta prior distribution.
`beta.b`	Shape2 parameter for the Beta prior distribution.
`...`	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
bayes(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10, MonteCarlo=1000, beta.a=1, beta.b=1)
```

`bayes_jeffreys` *Bayesian method (Jeffrey's prior)*

Description

Calculate a binomial series lower confidence bound using Bayes' method with Jeffrey's prior.

Usage

```
bayes_jeffreys(s, n, alpha, MonteCarlo, ...)
```

Arguments

`s`	Vector of successes.
`n`	Vector of sample sizes.
`alpha`	The significance level; to calculate a 100(1-α)% lower confidence bound.
`MonteCarlo`	Number of samples to draw from the posterior distribution for the Monte Carlo estimate.
`...`	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
bayes_jeffreys(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10, MonteCarlo=1000)
```

`bayes_nlg` *Bayesian method (Negative Log Gamma Prior)*

Description

Caclulate a binomal series lower confidence bound using Bayes' method with negative log gamma priors on the components, defined such that the prior on the system is a uniform distribution.

Usage

```
bayes_nlg(s, n, alpha, MonteCarlo, ...)
```

Arguments

s	Vector of successes.
n	Vector of sample sizes.
alpha	The significance level; to calculate a 100(1-α)% lower confidence bound.
MonteCarlo	Number of samples to draw from the posterior distribution for the Monte Carlo estimate.
...	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
bayes_nlg(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10, MonteCarlo=1000)
```

---

bayes_uniform *Bayesian method (Uniform prior)*

Description

Calculate a binomial series lower confidence bound using Bayes' method with a uniform prior distribution.

Usage

```
bayes_uniform(s, n, alpha, MonteCarlo, ...)
```

Arguments

s	Vector of successes.
n	Vector of sample sizes.
alpha	The significance level; to calculate a 100(1-α)% lower confidence bound.
MonteCarlo	Number of samples to draw from the posterior distribution for the Monte Carlo estimate.
...	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
bayes_uniform(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10, MonteCarlo=1000)
```

`chao_huwang` *Chao-Huwang method*

Description

Calculate a binomial series lower confidence bound using Chao and Huwang's (1987) method.

Usage

```
chao_huwang(s, n, alpha, MonteCarlo, ...)
```

Arguments

`s`	Vector of successes.
`n`	Vector of sample sizes.
`alpha`	The significance level; to calculate a 100(1-α)% lower confidence bound.
`MonteCarlo`	Number of samples to draw from the posterior distribution for the Monte Carlo estimate.
`...`	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
chao_huwang(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10, MonteCarlo=1000)
```

`coit` *Coit's method*

Description

Calculate a binomial series lower confidence bound using Coit's (1997) method.

Usage

```
coit(s, n, alpha, use.backup = FALSE, backup.method, ...)
```

Arguments

s	Vector of successes.
n	Vector of sample sizes.
alpha	The significance level; to calculate a 100(1-α)% lower confidence bound.
use.backup	If TRUE, then a backup.method in the will be used for the methods with calculate LCB = 1 in the case of no failures across all components. If FALSE (default), no backup.method is used.
backup.method	The backup method which is used for the methods which calculate LCB = 1 in the case of zero failures. Use function name.
...	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
coit(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10)
```

---

easterling *Easterling's method*

Description

Calculate a binomial series lower confidence bound using Easterling's (1972) method.

Usage

```
easterling(s, n, alpha, ...)
```

Arguments

s	Vector of successes.
n	Vector of sample sizes.
alpha	The significance level; to calculate a 100(1-α)% lower confidence bound.
...	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
easterling(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10)
```

launch_app *Launch Shiny App*

Description

Launches an instance of an R Shiny App, which runs locally on the user's computer.

Usage

```
launch_app(MonteCarlo = 1000, use.backup = TRUE,
  backup.method = lindstrom_madden_AC, sample.omega = "corners",
  number = 50)
```

Arguments

MonteCarlo	The number of Monte Carlo samples to take. E.g. In a Bayesian method, how many samples to take from a posterior distribution to estimate the lower α-th quantile. The default value is 1000.
use.backup	If TRUE (default), then a backup.method in the will be used for the methods with calculate LCB = 1 in the case of no failures across all components. If FALSE, no backup.method is used.
backup.method	The backup method which is used for the methods which calculate LCB = 1 in the case of zero failures. The default is lindstrom_madden_AC.
sample.omega	The method used to define component reliabilities. Can be only one of "corners" (default), "random", or "both". See Details below.
number	The number of component reliability vectors sampled if sample.omega = "random" or "both". Default is 50.

Details

If the "Download Histograms" button does not work, it can be fixed by launching the Shiny App on your local browser. This can be done by clicking on "Open in Browser" located at the top of your Shiny App. This seems to be an issue with the Download Handler that Shiny uses.

Define

$$\Omega = \{(p_1, p_2, \ldots, p_m) : \prod_{i=1}^{m} p_i \in [R_L, R_U]\}$$

and

$$\Omega' = \{(p_1, p_2, \ldots, p_m) : p_i = R_L^{1/m} or R_U^{1/m} \forall i\}$$

. If sample.omega = "corners" (the default), then the elements of

$$\Omega'$$

are used for component reliabilities, of which there are

$$2^m$$

combinations. If sample.omega = "random", then each component reliability is sampled uniformly from the interval

$$[R_L^m, R_U^m]$$

. If sample.omega = "both", then the results of "corners" and "random" are appended together and both are used.

`lindstrom_madden` *Lindstrom and Madden's method*

Description

Calculate a binomial series lower confidence bound using Lindstrom and Madden's (1962) method.

Usage

```
lindstrom_madden(s, n, alpha, ...)
```

Arguments

s	Vector of successes.
n	Vector of sample sizes.
alpha	The significance level; to calculate a 100(1-α)% lower confidence bound.
...	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
lindstrom_madden(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10)
```

`lindstrom_madden_AC` *Lindstrom and Madden's method with Agresti-Coull*

Description

Calculate a binomial series lower confidence bound using Agresti-Coull (1998) lower confidence bound calculation in the Lindstrom and Madden's (1962) method.

Usage

```
lindstrom_madden_AC(s, n, alpha, ...)
```

Arguments

`s`	Vector of successes.
`n`	Vector of sample sizes.
`alpha`	The significance level; to calculate a 100(1-α)% lower confidence bound.
`...`	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
lindstrom_madden_AC(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10)
```

`madansky` *Madansky's method*

Description

Calculate a binomial series lower confidence bound using Madansky's (1965) method.

Usage

```
madansky(s, n, alpha, use.backup = FALSE, backup.method, ...)
```

Arguments

`s`	Vector of successes.
`n`	Vector of sample sizes.
`alpha`	The significance level; to calculate a 100(1-α)% lower confidence bound.
`use.backup`	If TRUE, then a backup.method in the will be used for the methods with calculate LCB = 1 in the case of no failures across all components. If FALSE (default), no backup.method is used.
`backup.method`	The backup method which is used for the methods which calculate LCB = 1 in the case of zero failures. Use function name.
`...`	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound. Note that if there are zero observed failures across all components, the output is LCB = 0.

Examples

```
madansky(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10)
```

madansky.fun *Lagrange multiplier in Madansky's method*

Description

This function is called in the `madansky()` function to solve for the Lagrange multipliers.

Usage

```
madansky.fun(lam, s, n, alpha)
```

Arguments

`lam`	The value of the Lagrange multiplier
`s`	Vector of successes.
`n`	Vector of sample sizes.
`alpha`	The significance level; to calculate a 100(1-α)% lower confidence bound.

mann_grubbs *Mann and Grubb's method*

Description

Calculate a binomial series lower confidence bound using Mann and Grubb's (1974) method.

Usage

```
mann_grubbs(s, n, alpha, ...)
```

Arguments

`s`	Vector of successes.
`n`	Vector of sample sizes.
`alpha`	The significance level; to calculate a 100(1-α)% lower confidence bound.
`...`	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
mann_grubbs(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10)
```

`mann_grubbs_calc` *Function to calculate the LCB in the Mann-Grubbs method.*

Description

Calculate the LCB in the Mann-Grubbs method.

Usage

```
mann_grubbs_calc(s, n, A, alpha)
```

Arguments

`s`	Vector of successes.
`n`	Vector of sample sizes.
`A`	The restricted sum, as caclulated by the mann_grubbs_sum() function.
`alpha`	The significance level; to calculate a 100(1-α)% lower confidence bound.

Value

The LCB for the Mann-Grubbs method.

`mann_grubbs_sum` *Function to calculate the restricted sum in the Mann-Grubbs method.*

Description

Calculate the restricted sum in the Mann-Grubbs method.

Usage

```
mann_grubbs_sum(s, n)
```

Arguments

`s`	Vector of successes.
`n`	Vector of sample sizes.

Value

The restricted sum.

mr.fun — *Function of β in the Myhre-Rennie 2 method*

Description

This function is called in `myhre_rennie2()` function to solve for the β value.

Usage

```
mr.fun(beta, s, n)
```

Arguments

`beta`	The value of β.
`s`	Vector of successes.
`n`	Vector of sample sizes.

myhre_rennie1 — *Myhre and Rennie (modified ML) method*

Description

Calculate a binomial series lower confidence bound using the Myhre-Rennie (modified ML) method (1986).

Usage

```
myhre_rennie1(s, n, alpha, use.backup = FALSE, backup.method, ...)
```

Arguments

`s`	Vector of successes.
`n`	Vector of sample sizes.
`alpha`	The significance level; to calculate a 100(1-α)% lower confidence bound.
`use.backup`	If TRUE, then a backup.method in the will be used for the methods with calculate LCB = 1 in the case of no failures across all components. If FALSE (default), no backup.method is used.
`backup.method`	The backup method which is used for the methods which calculate LCB = 1 in the case of zero failures. Use function name.
`...`	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
myhre_rennie1(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10)
```

`myhre_rennie2` *Myhre and Rennie (reliability invariant) method*

Description

Calculate a binomial series lower confidence bound using the Myhre-Rennie (reliability invariant) method (1986).

Usage

```
myhre_rennie2(s, n, alpha, use.backup = FALSE, backup.method, ...)
```

Arguments

`s`	Vector of successes.
`n`	Vector of sample sizes.
`alpha`	The significance level; to calculate a 100(1-α)% lower confidence bound.
`use.backup`	If TRUE, then a backup.method in the will be used for the methods with calculate LCB = 1 in the case of no failures across all components. If FALSE (default), no backup.method is used.
`backup.method`	The backup method which is used for the methods which calculate LCB = 1 in the case of zero failures. Use function name.
`...`	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
myhre_rennie2(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10)
```

nishime *Nishime's method*

Description

Calculate a binomial series lower confidence bound using Nishime's (1959) method.

Usage

```
nishime(s, n, alpha, ...)
```

Arguments

`s`	Vector of successes.
`n`	Vector of sample sizes.
`alpha`	The significance level; to calculate a 100(1-α)% lower confidence bound.
`...`	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
nishime(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10)
```

nlg.post.sample *Sampling from Posterior of Negative Log Gamma prior and Binomial data.*

Description

Randomly sample from the posterior distribution resulting from a NLG prior and Binomial data.

Usage

```
nlg.post.sample(sample.size, shape, scale, s, n)
```

Arguments

`sample.size`	The number of draws from the posterior distribution.
`shape`	The shape parameter for the NLG prior.
`scale`	The scale parameter for the NLG prior.
`s`	The number of successes for the binomial data (should be a scalar).
`n`	The number of tests for the binomial data (should be a scalar).

Examples

```
nlg.post.sample(sample.size=50, shape=.2, scale=1, s=29, n=30)
```

---

normal_approximation *Normal approximation method*

Description

Calculate a binomial series lower confidence bound using a normal approximation with MLE estimates.

Usage

```
normal_approximation(s, n, alpha, use.backup = FALSE, backup.method, ...)
```

Arguments

`s`	Vector of successes.
`n`	Vector of sample sizes.
`alpha`	The significance level; to calculate a 100(1-α)% lower confidence bound.
`use.backup`	If TRUE, then a backup.method in the will be used for the methods with calculate LCB = 1 in the case of no failures across all components. If FALSE (default), no backup.method is used.
`backup.method`	The backup method which is used for the methods which calculate LCB = 1 in the case of zero failures. Use function name.
`...`	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
normal_approximation(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10)
```

pm — *Matrix of* p*-vector combinations*

Description

Calculate a matrix of *p*-vector combinations (component reliabilities) which lie in the specified interval of system reliability. Rows correspond to *p*-vectors and columns correspond to components.

Usage

```
pm(Rs.int, m)
```

Arguments

`Rs.int`	Interval (or single number) of total system reliability.
`m`	Number of components.

Details

Denote Rs.int $= (R_L, R_U)$. This function calculates all elements of the set

$$\Omega' = \{(p_1, p_2, \ldots, p_m) : p_i = R_L^{1/m} or R_U^{1/m} \forall i\}$$

.

Value

The 2^m by m matrix of *p*-vector combinations.

Examples

```
pm(Rs.int = c(.9, .95), m=3)
```

pm.random — *Matrix of* p*-vector combinations sampled randomly.*

Description

Randomly sample to build a matrix of *p*-vector combinations (component reliabilities) which lie in the specified interval of system reliability. Rows correspond to *p*-vectors and columns correspond to components.

Usage

```
pm.random(Rs.int, m, number)
```

Arguments

`Rs.int`	Interval (or single number) of total system reliability.
`m`	Number of components.
`number`	The number of random samples to draw.

Examples

```
pm.random(Rs.int=c(.9, .95), m=3, number=100)
```

`rice_moore` *Rice and Moore's method*

Description

Calculate a binomial series lower confidence bound using Rice and Moore's (1983) method.

Usage

```
rice_moore(s, n, alpha, MonteCarlo, f.star = 1.5 - min(n) + 0.5 * sqrt((3 - 2
  * min(n))^2 - 4 * (min(n) - 1) * log(alpha) * qchisq(p = alpha, df = 2)), ...)
```

Arguments

`s`	Vector of successes.
`n`	Vector of sample sizes.
`alpha`	The significance level; to calculate a 100(1-α)% lower confidence bound.
`MonteCarlo`	Number of samples to draw from the posterior distribution for the Monte Carlo estimate.
`f.star`	The number of psuedo-failures to use for a component that exhibits zero observed failures. The default value is from the log-gamma procedure proposed by Gatliffe (1976), and is the value used by Rice and Moore.
`...`	Additional arguments to be ignored.

Value

The 100(1-α)% lower confidence bound.

Examples

```
rice_moore(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10, MonteCarlo=1000)
```

rmse.LCB	*Root Mean Square Error*

Description

Calculate the root mean squared errors of the LCB's from the true system reliability. A measure of spread.

Usage

```
rmse.LCB(LCB, R)
```

Arguments

`LCB`	Vector of LCB's.
`R`	The true system reliability .

Value

The root mean squared error of the LCB's from the true system reliability.

Index