Package ‘serieslcb’

July 9, 2019

Type Package
Title Lower Confidence Bounds for Binomial Series System
Version 0.4.0
Author Edward Schuberg
Maintainer Edward Schuberg <eschu@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 ................................. 2
bayes_jeffreys ................................ 3
bayes_nlg .................................. 3
bayes_uniform ............................. 4
chao_huwang ............................. 5
coit ........................................ 5
easterling .................................. 6
launch_app ................................ 7
lindstrom_madden ........................ 8
lindstrom_madden_AC ..................... 8
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 $100(1-\alpha)$% 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 $100(1-\alpha)$% 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-\(\alpha\))% 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-\(\alpha\))% 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

Calculate a binomial 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 $100(1-\alpha)$% 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 $100(1-\alpha)$% lower confidence bound.

Examples

bayes_uniform(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 $100(1-\alpha)$% 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 $100(1-\alpha)$% 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 $100(1-\alpha)\%$ 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 $100(1-\alpha)\%$ 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, ...)

Table: chao_huwang

<table>
<thead>
<tr>
<th>chao_huwang</th>
<th>Chao-Huwang method</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>Calculate a binomial series lower confidence bound using Chao and Huwang’s (1987) method.</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>chao_huwang(s, n, alpha, MonteCarlo, ...)</td>
</tr>
<tr>
<td><strong>Arguments</strong></td>
<td></td>
</tr>
<tr>
<td><strong>s</strong></td>
<td>Vector of successes.</td>
</tr>
<tr>
<td><strong>n</strong></td>
<td>Vector of sample sizes.</td>
</tr>
<tr>
<td><strong>alpha</strong></td>
<td>The significance level; to calculate a $100(1-\alpha)%$ lower confidence bound.</td>
</tr>
<tr>
<td><strong>MonteCarlo</strong></td>
<td>Number of samples to draw from the posterior distribution for the Monte Carlo estimate.</td>
</tr>
<tr>
<td><strong>...</strong></td>
<td>Additional arguments to be ignored.</td>
</tr>
<tr>
<td><strong>Value</strong></td>
<td>The $100(1-\alpha)%$ lower confidence bound.</td>
</tr>
<tr>
<td><strong>Examples</strong></td>
<td>chao_huwang(s=c(35, 97, 59), n=c(35, 100, 60), alpha=.10, MonteCarlo=1000)</td>
</tr>
</tbody>
</table>

Table: coit

<table>
<thead>
<tr>
<th>coit</th>
<th>Coit’s method</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>Calculate a binomial series lower confidence bound using Coit’s (1997) method.</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>coit(s, n, alpha, use.backup = FALSE, backup.method, ...)</td>
</tr>
</tbody>
</table>
Arguments

\( s \) Vector of successes.
\( n \) Vector of sample sizes.
\( \alpha \) The significance level; to calculate a \( 100(1-\alpha)\% \) lower confidence bound.
\( \text{use.backup} \) 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.
\( \text{backup.method} \) The backup method which is used for the methods which calculate LCB = 1 in the case of zero failures. Use function name.
\( \ldots \) Additional arguments to be ignored.

Value

The \( 100(1-\alpha)\% \) lower confidence bound.

Examples

easterling(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, \ldots)

**Arguments**

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

**Value**

The \( 100(1-\alpha)\% \) lower confidence bound.

**Examples**

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

## Description

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

## Usage

```r
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 $\alpha$-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 $\text{LCB} = 1$ 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 $\text{LCB} = 1$ 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

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

. 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-\(\alpha\))% lower confidence bound.
- `...` Additional arguments to be ignored.

**Value**

The 100(1-\(\alpha\))% 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**


**Usage**

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

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

**Value**

The $100(1-\alpha)\%$ lower confidence bound.

**Examples**

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

**Description**

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

**Usage**

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

**Examples**

```r
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.
A The restricted sum, as calculated by the mann_grubbs_sum() function.
alpha The significance level; to calculate a $100(1-\alpha)\%$ 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.

Value
The restricted sum.
mr.fun  

Function of $\beta$ in the Myhre-Rennie 2 method

Description

This function is called in myhre_rennie2() function to solve for the $\beta$ value.

Usage

mr.fun(beta, s, n)

Arguments

- **beta**: The value of $\beta$.
- **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 $100(1-\alpha)$% 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 $100(1-\alpha)$% lower confidence bound.
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-\(\alpha\))% 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-\(\alpha\))% 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**

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

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

**Value**
The $100(1-\alpha)\%$ lower confidence bound.

**Examples**

```r
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**

```r
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).
normal_approximation

Examples

\texttt{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**

\texttt{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 $100(1-\alpha)\%$ 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 $100(1-\alpha)\%$ lower confidence bound.

**Examples**

\texttt{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**

\[
\text{pm}(\text{rs.int}, m)
\]

**Arguments**

- **rs.int** Interval (or single number) of total system reliability.
- **m** Number of components.

**Details**

Denote \( \text{Rs.int} = (R_L, R_U) \). This function calculates all elements of the set

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

**Value**

The \( 2^m \) by \( m \) matrix of \( p \)-vector combinations.

**Examples**

\[
\text{pm}(\text{rs.int} = c(.9, .95), m=3)
\]

**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**

\[
\text{pm.random}(\text{rs.int}, m, \text{number})
\]
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 pseudo-failures to use for a component that exhibits zero ob-
               served failures. The default value is from the log-gamma procedure proposed
               by Gatilffe (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

bayes, 2
bayes_jeffreys, 3
bayes_nlg, 3
bayes_uniform, 4

chao_huwang, 5
coit, 5

easterling, 6

launch_app, 7
lindstrom_madden, 8
lindstrom_madden_AC, 8

madansky, 9
madansky_fun, 10
mann_grubbs, 10
mann_grubbs_calc, 11
mann_grubbs_sum, 11
mr_fun, 12
myhre_rennie1, 12
myhre_rennie2, 13

nishime, 14
nlg.post.sample, 14
normal_approximation, 15

pm, 16
pm.random, 16

rice_moore, 17
rmse.LCB, 18