

Package: nbconv (via r-universe)

October 13, 2026

Title Evaluate Arbitrary Negative Binomial Convolutions

Version 2.0.0

URL <https://github.com/gbedwell/nbconv>

BugReports <https://github.com/gbedwell/nbconv/issues>

Imports parallel, matrixStats, stats, fftw

Description Five distinct methods are implemented for evaluating the sums of arbitrary negative binomial distributions. These methods are: Furman's exact series representation (Furman (2007) <[doi:10.1016/j.spl.2006.06.007](https://doi.org/10.1016/j.spl.2006.06.007)>), an FFT-based approach, a "hybrid" approach that uses FFT for non-extreme probabilities and falls back to the series representation for extremely low probabilities, saddlepoint approximation, and a method-of-moments approximation. Functions are provided to calculate probability masses, cumulative probabilities, and quantiles of the convolutions in question using said evaluation methods. Functions for generating random deviates from negative binomial convolutions and for directly calculating the mean, variance, skewness, and excess kurtosis from cumulants are also provided.

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

License GPL (>= 3)

Config/pak/sysreqs libfftw3-dev

Repository <https://gbedwell.r-universe.dev>

Date/Publication 2026-04-16 14:30:22 UTC

RemoteUrl <https://github.com/gbedwell/nbconv>

RemoteRef HEAD

RemoteSha 8671c707365ace7c7a733d3ec2394412bfb544d0

Contents

dnbconv	2
nb_sum_fft	3
nb_sum_hybrid	4
nb_sum_moments	4
nb_sum_saddlepoint	5
nb_sum_series	6
nbconv_params	6
pnbconv	7
qnbconv	8
rnbcnv	9

Index	11
--------------	-----------

dnbconv	<i>Probability Mass Function</i>
---------	----------------------------------

Description

Calculates the probability mass function for negative binomial convolutions.

Usage

```
dnbconv(
  counts,
  mus,
  phis,
  ps,
  method = c("hybrid", "fft", "series", "saddlepoint", "moments"),
  n.cores = 1,
  normalize = TRUE,
  log.p = FALSE,
  rm.threshold = 1e-06
)
```

Arguments

counts	Integer vector of counts over which the convolution is evaluated.
mus	Numeric vector of individual mean values
phis	Numeric vector of individual dispersion parameters. Equivalent to 'size' in dnbinom.
ps	Numeric vector of individual probabilities.
method	The method by which to evaluate the PMF. One of 'hybrid', 'fft', 'series', 'saddlepoint', or 'moments'.
n.cores	The number of CPU cores to use. Only used with 'fft' and 'hybrid' methods.

normalize	Boolean. If TRUE, the PMF is normalized.
log.p	Boolean. If TRUE, log-scale values are returned. Defaults to FALSE.
rm.threshold	The lower limit of mu. Any mus less-than rm.threshold are removed. Defaults to 1E-6. Must be >= 0.01 and cannot be negative.

Value

A numeric vector of probability densities.

Examples

```
dnbconv(counts = 0:100, mus = c(100, 10), phis = c(5, 8), method = "hybrid")
```

nb_sum_fft	<i>FFT Convolution</i>
------------	------------------------

Description

Implements FFT-based convolution for the evaluation of the sum of arbitrary NB random variables. Called by other functions. Not intended to be run alone.

Usage

```
nb_sum_fft(mus, phis, counts, n.cores = 1, log.p = TRUE)
```

Arguments

mus	Numeric vector of individual mean values
phis	Numeric vector of individual dispersion parameters. Equivalent to 'size' in dnbinom.
counts	Integer vector of counts over which the convolution is evaluated.
n.cores	The number of CPU cores to use.
log.p	Boolean. Whether to return probabilities on log-scale. Defaults to TRUE.

Value

A numeric vector of probability densities corresponding to the input counts.

Examples

```
nbconv:::nb_sum_fft(mus = c(100, 10), phis = c(5, 8), counts = 0:100)
```

nb_sum_hybrid

Hybrid FFT and Series Convolution

Description

Implements FFT-based convolution followed by series convolution for extreme probabilities. Called by other functions. Not intended to be run alone.

Usage

```
nb_sum_hybrid(mus, phis, counts, n.cores = 1, log.p = TRUE)
```

Arguments

mus	Numeric vector of individual mean values
phis	Numeric vector of individual dispersion parameters. Equivalent to 'size' in dnbinom.
counts	Integer vector of counts over which the convolution is evaluated.
n.cores	The number of CPU cores to use.
log.p	Boolean. Whether to return probabilities on log-scale. Defaults to TRUE.

Value

A numeric vector of probability densities corresponding to the input counts.

Examples

```
nbconv::nb_sum_hybrid(mus = c(100, 10), phis = c(5, 8), counts = 0:100)
```

nb_sum_moments

Method of Moments

Description

Implements the method of moments approximation for the sum of arbitrary NB random variables. Called by other functions. Not intended to be run alone.

Usage

```
nb_sum_moments(mus, phis, counts, log.p = TRUE)
```

Arguments

mus	Numeric vector of individual mean values
phis	Numeric vector of individual dispersion parameters. Equivalent to 'size' in dnbinom.
counts	Integer vector of counts over which the convolution is evaluated.
log.p	Boolean. Whether or not to return probabilities on the log-scale. Defaults to TRUE.

Value

A numeric vector of probability densities.

Examples

```
nbconv::nb_sum_moments(mus = c(100, 10), phis = c(5, 8), counts = 0:100)
```

nb_sum_saddlepoint	<i>Saddlepoint Approximation</i>
--------------------	----------------------------------

Description

Implements the saddlepoint approximation for the sum of arbitrary NB random variables. Called by other functions. Not intended to be run alone.

Usage

```
nb_sum_saddlepoint(mus, phis, counts, normalize = TRUE, log.p = TRUE)
```

Arguments

mus	Numeric vector of individual mean values
phis	Numeric vector of individual dispersion parameters. Equivalent to 'size' in dnbinom.
counts	Integer vector of counts over which the convolution is evaluated.
normalize	Boolean. If TRUE (default), the PMF is normalized to sum to 1.
log.p	Boolean. Whether or not to return probabilities on the log-scale. Defaults to TRUE.

Value

A numeric vector of probability densities.

Examples

```
nbconv::nb_sum_saddlepoint(mus = c(100, 10), phis = c(5, 8), counts = 0:100)
```

nb_sum_series	<i>Series Convolution/Furman's PMF</i>
---------------	--

Description

Implements Furman's series PMF for the evaluation of the sum of arbitrary NB random variables. Called by other functions. Not intended to be run alone.

Usage

```
nb_sum_series(phis, ps, counts, log.p = TRUE)
```

Arguments

phis	Numeric vector of individual dispersion parameters. Equivalent to 'size' in dnbinom.
ps	Numeric vector of individual probabilities.
counts	Integer vector of counts over which the convolution is evaluated.
log.p	Boolean. Whether or not to return probabilities on the log-scale. Defaults to TRUE.

Value

A numeric vector of probability densities.

Examples

```
nbconv:::nb_sum_series(ps = c(0.0476, 0.4444), phis = c(5, 8), counts = 0:100)
```

nbconv_params	<i>Summary statistics</i>
---------------	---------------------------

Description

Calculates distribution parameters for the convolution of arbitrary negative binomial random variables.

Usage

```
nbconv_params(mus, phis, ps, rm.threshold = 1e-06)
```

Arguments

mus	Numeric vector of individual mean values
phis	Numeric vector of individual dispersion parameters. Equivalent to 'size' in dnbinom.
ps	Numeric vector of individual probabilities of success.
rm.threshold	The lower limit of mu. Any mus < rm.threshold are removed. Defaults to 1E-6. Must be >= 0.01 and cannot be negative.

Value

A named numeric vector of distribution parameters.

Examples

```
nbconv_params(mus = c(100, 10), phis = c(5, 8))
```

pnbconv	<i>Cumulative Distribution Function</i>
---------	---

Description

Calculates the CDF for the convolution of arbitrary negative binomial random variables.

Usage

```
pnbconv(
  quants,
  mus,
  phis,
  ps,
  method = c("hybrid", "fft", "series", "saddlepoint", "moments"),
  n.cores = 1,
  normalize = TRUE,
  log.p = FALSE,
  rm.threshold = 1e-06
)
```

Arguments

quants	Integer vector of quantiles.
mus	Numeric vector of individual mean values
phis	Numeric vector of individual dispersion parameters. Equivalent to 'size' in dnbinom.
ps	Numeric vector of individual probabilities.

method	The method by which to evaluate the PMF. One of 'hybrid', 'fft', 'series', 'saddlepoint', or 'moments'.
n.cores	The number of CPU cores to use. Only used with 'fft' and 'hybrid' methods.
normalize	Boolean. If TRUE, the PMF is normalized.
log.p	Boolean. If TRUE, log-scale values are returned. Defaults to FALSE.
rm.threshold	The lower limit of mu. Any mus less-than rm.threshold are removed. Defaults to 1E-6. Must be ≥ 0.01 and cannot be negative.

Value

A numeric vector of cumulative probability densities.

Examples

```
pnbconv(quants = 100, mus = c(100, 10), phis = c(5, 8), method = "hybrid")
```

qnbconv

Quantile Function

Description

Calculates the quantile function for the convolution of arbitrary negative binomial random variables.

Usage

```
qnbconv(
  probs,
  counts,
  mus,
  phis,
  ps,
  method = c("hybrid", "fft", "series", "saddlepoint", "moments"),
  n.cores = 1,
  normalize = TRUE,
  log.p = FALSE,
  rm.threshold = 1e-06
)
```

Arguments

probs	Numeric vector of target (cumulative) probabilities.
counts	Integer vector of counts over which the convolution is evaluated.
mus	Numeric vector of individual mean values
phis	Numeric vector of individual dispersion parameters. Equivalent to 'size' in dnbinom.

ps	Numeric vector of individual probabilities.
method	The method by which to evaluate the PMF. One of 'hybrid', 'fft', 'series', 'saddlepoint', or 'moments'.
n.cores	The number of CPU cores to use. Only used with 'fft' and 'hybrid' methods.
normalize	Boolean. If TRUE, the PMF is normalized.
log.p	Boolean. If TRUE, log-scale values are returned. Defaults to FALSE.
rm.threshold	The lower limit of mu. Any mus less-than rm.threshold are removed. Defaults to 1E-6. Must be ≥ 0.01 and cannot be negative.

Value

A numeric vector of quantiles.

Examples

```
qnbconv(probs = c(0.05, 0.25, 0.5, 0.75, 0.95), counts = 0:200,
        mus = c(100, 10), phis = c(5, 8), method = "hybrid")
```

rnbconv

*Random Deviates***Description**

Generates random samples from the convolution of arbitrary negative binomial random variables.

Usage

```
rnbconv(mus, phis, ps, n.samp, n.cores = 1, rm.threshold = 1e-06)
```

Arguments

mus	Numeric vector of individual mean values
phis	Numeric vector of individual dispersion parameters. Equivalent to 'size' in dnbinom.
ps	Numeric vector of individual probabilities.
n.samp	The number of samples per distribution
n.cores	The number of CPU cores to use.
rm.threshold	The lower limit of mu. Any mus less-than rm.threshold are removed. Defaults to 1E-6. Must be ≥ 0.01 and cannot be negative.

Value

A numeric vector of random deviates.

Examples

```
rnbconv(mus = c(100, 10), phis = c(5, 8), n.samp = 10)
```

Index

dnbconv, [2](#)

nb_sum_fft, [3](#)

nb_sum_hybrid, [4](#)

nb_sum_moments, [4](#)

nb_sum_saddlepoint, [5](#)

nb_sum_series, [6](#)

nbconv_params, [6](#)

pnbconv, [7](#)

qnbconv, [8](#)

rnbconv, [9](#)