Package ‘lognorm’

August 2, 2019

Title Functions for the Lognormal Distribution

Version 0.1.6

Author Thomas Wutzler

Maintainer Thomas Wutzler <twutz@bgc-jena.mpg.de>

Description The lognormal distribution
(Limpert et al. (2001) <doi:10.1641/0006-3568(2001)051%5B0341:lndats%5D2.0.co;2>)
can characterize uncertainty that is bounded by zero.
This package provides estimation of distribution parameters, computation of
moments and other basic statistics, and an approximation of the distribution
of the sum of several correlated lognormally distributed variables
(Lo 2013 <doi:10.12988/ams.2013.39511>).

Imports Matrix

Suggests testthat, knitr, dplyr, ggplot2, mvtnorm, purrr, tidyr

VignetteBuilder knitr

License GPL-2

LazyData true

RoxygenNote 6.1.1

URL https://github.com/bgctw/lognorm

NeedsCompilation no

Repository CRAN

Date/Publication 2019-08-02 08:00:02 UTC

R topics documented:

`lognorm-package` *Utilities for the lognormal distribution in R*

Description

Utilities for the lognormal distribution in R

- Compute moments.
- Estimate autocorrelation.
- Approximate the sum of correlated lognormals.

Details

Moments and mode

- Expected value and variance: `getLognormMoments`
- Mode: `getLognormMode`
- Median: `getLognormMedian`

Estimating parameters

- from sample: `estimateParmsLognormFromSample`
- from mean and variance at original scale: `getParmsLognormForMoments`
- from mean and multiplicative standard deviation at original scale: `getParmsLognormForExpval`
- from expected value and upper quantile: `getParmsLognormForMeanAndUpper`
- from median and upper quantile: `getParmsLognormForMedianAndUpper`
- from mode and upper quantile: `getParmsLognormForModeAndUpper`
- from lower and upper quantile: `getParmsLognormForLowerAndUpper`

Approximate the sum of correlated lognormals

- According to Lo 2013: `estimateSumLognormal`

Utilities for correlated data. These functions maybe moved to a separate package in future.

- Estimate standard error of the mean: `seCor`
- Compute the effective number of observations taking into account autocorrelation: `computeEffectiveNumObs`
- Return the vector of effective components of the autocorrelation: `computeEffectiveAutoCorr`
- Estimate the variance of a correlated time series: `varEffective`

Also have a look at the package vignettes.

Author(s)

Thomas Wutzler

References

`Limpert E, Stahel W & Abbt M (2001) Log-normal Distributions across the Sciences: Keys and Clues. BioScience, Oxford University Press (OUP), 51 , 341 10.1641/0006-3568(2001)051[0341:lndats]2.0.co;2`

`Lo C (2013) WKB approximation for the sum of two correlated lognormal random variables. Applied Mathematical Sciences, Hikari, Ltd., 7 , 6355-6367 10.12988/ams.2013.39511`

`computeEffectiveAutoCorr` *computeEffectiveAutoCorr*

Description

Return the vector of effective components of the autocorrelation

Usage

```
computeEffectiveAutoCorr(res, type = "correlation")
```

Arguments

`res`	numeric of autocorrelated numbers, usually observation - model residuals
`type`	

Details

Returns all components before first negative autocorrelation

Value

numeric vector: strongest components of the autocorrelation function

Author(s)

Thomas Wutzler

References

`Zieba 2011 Standard Deviation of the Mean of Autocorrelated Observations Estimated with the Use of the Autocorrelation Function Estimated From the Data`

Examples

```
# generate autocorrelated time series
res <- stats::filter(rnorm(1000), filter = rep(1,5), circular = TRUE)
res[100:120] <- NA
(effAcf <- computeEffectiveAutoCorr(res))
```

`computeEffectiveNumObs` *computeEffectiveNumObs*

Description

Compute the effective number of observations taking into account autocorrelation

Usage

```
computeEffectiveNumObs(res, effAcf = computeEffectiveAutoCorr(res),
    na.rm = FALSE)
```

Arguments

`res`	numeric of autocorrelated numbers, usually observation - model residuals
`effAcf`	autocorrelation coefficients. The first entry is fixed at 1 for zero distance. May provide precomputed for efficiency or computed from a larger sample.
`na.rm`	a logical value indicating whether NA values should be stripped before the computation proceeds.

Details

Handling of NA values: NAs at the beginning or end and are just trimmed before computation and pose no problem. However with NAs aside from edges, the return value is biased low, because correlation terms are subtracted for those positions.

Because of NA correlation terms, the computed effective number of observations can be smaller than 1. In this case 1 is returned.

Value

integer scalar: effective number of observations

Author(s)

Thomas Wutzler

References

Zieba & Ramza (2011) Standard Deviation of the Mean of Autocorrelated Observations Estimated with the Use of the Autocorrelation Function Estimated From the Data. Metrology and Measurement Systems,Walter de Gruyter GmbH,18 10.2478/v10178-011-0052-x

Bayley & Hammersley (1946) The "effective" number of independent observations in an autocorrelated time series. Supplement to the Journal of the Royal Statistical Society,JSTOR,8,184-197

Examples

```
# generate autocorrelated time series
res <- stats::filter(rnorm(1000), filter = rep(1,5), circular = TRUE)
res[100:120] <- NA
# plot the series of autocorrelated random variables
plot(res)
# plot their empirical autocorrelation function
acf(res, na.action = na.pass)
#effAcf <- computeEffectiveAutoCorr(res)
# the effective number of parameters is less than number of 1000 samples
(nEff <- computeEffectiveNumObs(res, na.rm = TRUE))
```

---

estimateParmsLognormFromSample

estimateParmsLognormFromSample

Description

get the lognormal parameters by expected value.

Usage

```
estimateParmsLognormFromSample(x, na.rm = FALSE)
```

Arguments

x	numeric vector of sampled values
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.

Author(s)

Thomas Wutzler

Examples

```
.mu <- log(1)
.sigma <- log(2)
x <- exp(rnorm(50, mean = .mu, sd = .sigma))
estimateParmsLognormFromSample(x)
```

`estimateSumLognormal` *estimateSumLognormal*

Description

Estimate the distribution parameters of the lognormal approximation to the sum

Usage

```
estimateSumLognormal(mu, sigma, corr = Diagonal(length(mu)),
    sigmaSum = numeric(0), corrLength = if (inherits(corr,
        "ddiMatrix")) 0 else nTerm, isStopOnNoTerm = FALSE,
    effAcf, na.rm = isStopOnNoTerm)
```

Arguments

`mu`	numeric vector of center parameters of terms at log scale
`sigma`	numeric vector of variance parameter of terms at log scale
`corr`	numeric matrix of correlations between the random variables
`sigmaSum`	numeric scalar: possibility to specify of a precomputed scale parameter
`corrLength`	integer scalar: set correlation length to smaller values to speed up computation by neglecting correlations among terms further apart. Set to zero to omit correlations.
`isStopOnNoTerm`	if no finite estimate is provided then by default NA is returned for the sum. Set this to TRUE to issue an error instead.
`effAcf`	numeric vector of effective autocorrelation This overrides arguments `corr` and `corrLength`
`na.rm`	if there are terms with NA values in mu or sigma by default also the sum coefficients are NA. Set to TRUE to neglect such terms in the sum.

Value

numeric vector with two components mu and sigma the parameters of the lognormal distribution at log scale

Author(s)

Thomas Wutzler

References

Lo C (2013) WKB approximation for the sum of two correlated lognormal random variables. Applied Mathematical Sciences,Hikari,Ltd.,7 ,6355-6367 10.12988/ams.2013.39511

Examples

```
# distribution of the sum of two lognormally distributed random variables
mu1 = log(110)
mu2 = log(100)
sigma1 = log(1.2)
sigma2 = log(1.6)
(coefSum <- estimateSumLognormal( c(mu1,mu2), c(sigma1,sigma2) ))
# repeat with correlation
(coefSumCor <- estimateSumLognormal( c(mu1,mu2), c(sigma1,sigma2), effAcf = c(1,0.9) ))
# expected value is equal, but variance with correlated variables is larger
getLognormMoments(coefSum["mu"],coefSum["sigma"])
getLognormMoments(coefSumCor["mu"],coefSumCor["sigma"])
```

`estimateSumLognormalSample`

estimateSumLognormalSample

Description

Estimate the parameters of the lognormal approximation to the sum

Usage

```
estimateSumLognormalSample(mu, sigma, resLog,
    effAcf = computeEffectiveAutoCorr(resLog),
    isGapFilled = logical(0), na.rm = TRUE)
```

Arguments

`mu`	numeric vector of center parameters of terms at log scale
`sigma`	numeric vector of variance parameter of terms at log scale
`resLog`	time series of model-residuals at log scale to estimate correlation
`effAcf`	effective autocorrelation coefficients (may provide precomputed for efficiency or if the sample of `resLog` is too small) set to 1 to assume uncorrelated sample
`isGapFilled`	logical vector whether entry is gap-filled rather than an original measurement, see details
`na.rm`	neglect terms with NA values in mu or sigma

Details

If there are no gap-filled values, i.e. `all(!isGapFilled)` or `!length(isGapFilled)` (the default), distribution parameters are estimated using all the samples. Otherwise, the scale parameter (uncertainty) is first estimated using only the non-gapfilled records.

Also use isGapFilled == TRUE for records, where sigma cannot be trusted. When setting sigma to missing, this is also affecting the expected value.

If there are only gap-filled records, assume uncertainty to be (before v0.1.5: the largest uncertainty of given gap-filled records.) the mean of the given multiplicative standard deviation

Value

numeric vector with components mu, sigma, and nEff, the parameters of the lognormal distribution at log scale (Result of link{estimateSumLognormal}) and the number of effective observations.

Author(s)

Thomas Wutzler

---

getCorrMatFromAcf *getCorrMatFromAcf*

Description

Construct the full correlation matrix from autocorrelation components.

Usage

```
getCorrMatFromAcf(nRow, effAcf)
```

Arguments

nRow	number of rows in correlation matrix
effAcf	numeric vector of effective autocorrelation components . The first entry, which is defined as 1, is not used.

Author(s)

Thomas Wutzler

---

getLognormMedian *getLognormMedian*

Description

get the median of a log-normal distribution

Usage

```
getLognormMedian(mu, sigma = NA)
```

Arguments

mu	center parameter (mean at log scale, log(median))
sigma	dummy not used, but signature as with Mode and moments

Value

the median

Author(s)

Thomas Wutzler

Examples

```
getLognormMedian(mu = log(1), sigma = log(2))
```

`getLognormMode` *getLognormMode*

Description

get the mode of a log-normal distribution

Usage

```
getLognormMode(mu, sigma)
```

Arguments

`mu`	center parameter (mean at log scale, log(median))
`sigma`	scale parameter (standard deviation at log scale)

Value

the mode

Author(s)

Thomas Wutzler

Examples

```
# with larger sigma, the distribution is more skewed
# with mode further away from median = 1
getLognormMode(mu = log(1), sigma = c(log(1.2),log(2)))
```

`getLognormMoments` *getLognormMoments*

Description

get the expected value and variance of a log-normal distribution

Usage

```
getLognormMoments(mu, sigma)
```

Arguments

`mu`	numeric vector of center parameter (mean at log scale, log(median))
`sigma`	numeric vector of scale parameter (standard deviation at log scale)

Value

numeric matrix with columns

`mean`	expected value at original scale
`var`	variance at original scale
`cv`	coefficient of variation: std/mean

Author(s)

Thomas Wutzler

References

Limpert E, Stahel W & Abbt M (2001) Log-normal Distributions across the Sciences: Keys and Clues. Oxford University Press (OUP) 51, 341, 10.1641/0006-3568(2001)051[0341:lndats]2.0.co;2

Examples

```
# start by estimating lognormal parameters from moments
.mean <- 1
.var <- c(1.3,2)^2
parms <- getParmsLognormForMoments(.mean, .var)
#
# computed moments must equal previous ones
(ans <- getLognormMoments(parms[,"mu"], parms[,"sigma"]))
cbind(.var, ans[,"var"])
```

`getParmsLognormForExpval` *getParmsLognormForExpval*

Description

get the lognormal parameters by expected value

Usage

```
getParmsLognormForExpval(mean, sigmaStar)
```

Arguments

`mean`	expected value at original scale
`sigmaStar`	multiplicative standard deviation

Author(s)

Thomas Wutzler

Examples

```
.mean <- 1
.sigmaStar <- c(1.3,2)
(parms <- getParmsLognormForExpval(.mean, .sigmaStar))
# multiplicative standard deviation must equal the specified value
cbind(exp(parms[,"sigma"]), .sigmaStar)
```

`getParmsLognormForLowerAndUpper` *getParmsLognormForLowerAndUpper*

Description

Calculates mu and sigma of lognormal from lower and upper quantile.

Usage

```
getParmsLognormForLowerAndUpper(lower, upper,
    sigmaFac = qnorm(0.99), isTransScale = FALSE)
```

Arguments

`lower`	value at the lower quantile, i.e. practical minimum
`upper`	value at the upper quantile, i.e. practical maximum
`sigmaFac`	sigmaFac = 2 is 95% sigmaFac = 2.6 is 99% interval
`isTransScale`	if true lower and upper are already on log scale

Value

named numeric vector: mu and sigma parameter of the lognormal distribution.

Author(s)

Thomas Wutzler

Examples

```
# sample in normal space
mu <- 5
sigma <- 2
rrNorm <- rnorm(1000, mean = mu, sd = sigma)
# transform to orignal scale
rrOrig <- exp(rrNorm)
# and re-estimate parameters from original scale
res <- getParmsLognormForMedianAndUpper(
  median(rrOrig), quantile(rrOrig, probs = 0.95), sigmaFac = qnorm(0.95))
expected <- c(mu = mu,sigma = sigma)
all.equal(res[1,], expected, tolerance = .1, scale = 1)
```

`getParmsLognormForMeanAndUpper`

getParmsLognormForMeanAndUpper

Description

Calculates mu and sigma of lognormal from median and upper quantile.

Usage

```
getParmsLognormForMeanAndUpper(mean, quant,
    sigmaFac = qnorm(0.99))
```

Arguments

`mean`	expected value at the original scale
`quant`	value at the upper quantile, i.e. practical maximum
`sigmaFac`	sigmaFac=2 is 95% sigmaFac=2.6 is 99% interval

Details

There are two valid solutions. This routine returns the one with lower sigma, i.e. the not so strongly skewed solution.

Value

numeric matrix: columns mu and sigma parameter of the lognormal distribution.

Author(s)

Thomas Wutzler

`getParmsLognormForMedianAndUpper`
getParmsLognormForMedianAndUpper

Description

Calculates mu and sigma of lognormal from median and upper quantile.

Usage

```
getParmsLognormForMedianAndUpper(median,
    quant, sigmaFac = qnorm(0.99))
```

Arguments

`median`	geometric mu (median at the original exponential scale)
`quant`	value at the upper quantile, i.e. practical maximum
`sigmaFac`	sigmaFac=2 is 95% sigmaFac=2.6 is 99% interval

Value

named numeric vector: mu and sigma parameter of the lognormal distribution.

Author(s)

Thomas Wutzler

getParmsLognormForModeAndUpper
getParmsLognormForModeAndUpper

Description

Calculates mu and sigma of lognormal from mode and upper quantile.

Usage

```
getParmsLognormForModeAndUpper(mle, quant,
    sigmaFac = qnorm(0.99))
```

Arguments

mle	numeric vector: mode at the original scale
quant	numeric vector: value at the upper quantile, i.e. practical maximum
sigmaFac	sigmaFac=2 is 95% sigmaFac=2.6 is 99% interval

Value

numeric matrix: columns mu and sigma parameter of the lognormal distribution. Rows correspond to rows of mle and quant

Author(s)

Thomas Wutzler

Examples

```
# example 1: a distribution with mode 1 and upper bound 5
(thetaEst <- getParmsLognormForModeAndUpper(1,5))
mle <- exp(thetaEst[1] - thetaEst[2]^2)
all.equal(mle, 1, check.attributes = FALSE)

# plot the distributions
xGrid = seq(0,8, length.out = 81)[-1]
dxEst <- dlnorm(xGrid, meanlog = thetaEst[1], sdlog = thetaEst[2])
plot( dxEst~xGrid, type = "l",xlab = "x",ylab = "density")
abline(v = c(1,5),col = "gray")

# example 2: true parameters, which should be rediscovered
theta0 <- c(mu = 1, sigma = 0.4)
mle <- exp(theta0[1] - theta0[2]^2)
perc <- 0.975 # some upper percentile, proxy for an upper bound
quant <- qlnorm(perc, meanlog = theta0[1], sdlog = theta0[2])
(thetaEst <- getParmsLognormForModeAndUpper(mle,quant = quant,sigmaFac = qnorm(perc)) )
```

```
#plot the true and the rediscovered distributions
xGrid = seq(0,10, length.out = 81)[-1]
dx <- dlnorm(xGrid, meanlog = theta0[1], sdlog = theta0[2])
dxEst <- dlnorm(xGrid, meanlog = thetaEst[1], sdlog = thetaEst[2])
plot( dx~xGrid, type = "l")
#plot( dx~xGrid, type = "n")
#overplots the original, coincide
lines( dxEst ~ xGrid, col = "red", lty = "dashed")

# example 3: explore varying the uncertainty (the upper quantile)
x <- seq(0.01,1.2,by = 0.01)
mle = 0.2
dx <- sapply(mle*2:8,function(q99){
  theta = getParmsLognormForModeAndUpper(mle,q99,qnorm(0.99))
  #dx <- dDistr(x,theta[,"mu"],theta[,"sigma"],trans = "lognorm")
  dx <- dlnorm(x,theta[,"mu"],theta[,"sigma"])
})
matplot(x,dx,type = "l")
```

---

getParmsLognormForMoments *getParmsLognormForMoments*

Description

get the mean and variance of a log-normal distribution

Usage

```
getParmsLognormForMoments(mean, var, sigmaOrig = sqrt(var))
```

Arguments

mean	expected value at original scale
var	variance at original scale
sigmaOrig	standard deviation at original scale , can be specified alternatively to the variance

Value

numeric matrix with columns

mu	center parameter (mean at log scale, log(median))
sigma	scale parameter (standard deviation at log scale)

Author(s)

Thomas Wutzler

References

Limpert E, Stahel W & Abbt M (2001) Log-normal Distributions across the Sciences: Keys and Clues. Oxford University Press (OUP) 51, 341, 10.1641/0006-3568(2001)051[0341:lndats]2.0.co;2

Examples

```
.mean <- 1
.var <- c(1.3,2)^2
getParmsLognormForMoments(.mean, .var)
```

---

seCor *seCor*

Description

Compute the standard error accounting for empirical autocorrelations

Usage

```
seCor(x, na.rm = FALSE, effCov = computeEffectiveAutoCorr(x,
    type = "covariance"))
```

Arguments

x	numeric vector
na.rm	logical. Should missing values be removed?
effCov	numeric vector of effective covariance components first entry is the variance. See computeEffectiveAutoCorr

Details

Computation follows https://stats.stackexchange.com/questions/274635/calculating-error-of-mean-of-time-series.

The default uses empirical autocorrelation estimates from the supplied data up to first negative component. For short series of x it is strongly recommended to to provide effCov that was estimated on a longer time series.

Value

numeric scalar of standard error of the mean of x

Author(s)

Thomas Wutzler

`setMatrixOffDiagonals` *setMatrixOffDiagonals*

Description

set off-diagonal values of the matrix

Usage

```
setMatrixOffDiagonals(x, diag = 1:length(value),
    value, isSymmetric = FALSE)
```

Arguments

`x`	numeric square matrix
`diag`	integer vector specifying the diagonals 0 is the center +1 the first row to upper and -2 the second row to lower
`value`	numeric vector of values to fill in
`isSymmetric`	set to TRUE to to only specify the upper diagonal element but also set the lower in the mirrored diagonal

Value

matrix with modified diagonal elements

Author(s)

Thomas Wutzler

`varEffective` *varEffective*

Description

Estimate the variance of a correlated time series

Usage

```
varEffective(res, nEff = computeEffectiveNumObs(res,
    na.rm = na.rm), na.rm = FALSE, ...)
```

Arguments

`res`	numeric of autocorrelated numbers, usually observation - model residuals
`nEff`	effective number of observations
`na.rm`	set to TRUE to remove NA cases before computation
`...`	further arguments to `var`

Details

The BLUE is not anymore the usual variance, but a modified variance as given in `Zieba 2011`

Value

The estimated variance of the sample

Author(s)

Thomas Wutzler

Examples

```
# generate autocorrelated time series
res <- stats::filter(rnorm(1000), filter = rep(1,5), circular = TRUE)
res[100:120] <- NA
# if correlations are neglected, the estimate of the variance is biased low
(varNeglectCorr <- var(res, na.rm = TRUE))
(varCorr <- varEffective(res, na.rm = TRUE))
```

Index