Package ‘qmap’

May 3, 2016

Type Package
Title Statistical Transformations for Post-Processing Climate Model Output
Version 1.0-4
Date 2016-05-03
Author Lukas Gudmundsson
Maintainer Lukas Gudmundsson <lukas.gudmundsson@env.ethz.ch>
Depends R (>= 2.8.0), fitdistrplus
Description Empirical adjustment of the distribution of variables originating from (regional) climate model simulations using quantile mapping.
License GPL (>= 2)
LazyLoad yes
NeedsCompilation no
Repository CRAN
Date/Publication 2016-05-03 11:03:19

R topics documented:

qmap-package .................................................. 2
bernexp ....................................................... 2
berngamma .................................................... 4
bernlnorm ..................................................... 7
bernweibull ................................................... 8
fitQmap ......................................................... 10
fitQmapDIST ................................................ 13
fitQmapPTF .................................................. 17
fitQmapQUANT ............................................... 21
fitQmapRQUANT .............................................. 24
fitQmapSSPLIN .............................................. 27
obsprecip ..................................................... 30
startbernexp ................................................ 31
startberngamma .............................................. 32
Description

Empirical adjustment (bias correction) of variables originating from (regional) climate model simulations using quantile mapping. The workhorse functions of this package are `fitQmap` and `doQmap` which offer an easy to use interface to different statistical transformations, also referred to as quantile mapping methods.

Details

Package: qmap
Type: Package
Version: 1.0-4
Date: 2016-05-03
License: GPL >= 2
LazyLoad: yes

Author(s)

Lukas Gudmundsson

References


Description

Density, distribution function, quantile function and random generation for the Bernoulli-Exponential distribution with parameters prob, and rate.
Usage

dbernexp(x, prob, rate)
pbernexp(q, prob, rate)
qbernexp(p, prob, rate)
rbernexp(n, prob, rate)

Arguments

x, q  vector of quantiles.
p  vector of probabilities.
prob  probability of non-zero event.
n  number of random samples.
rate  rate parameter of the Exponential distribution.

Details

Mixture of the Bernoulli and the Exponential distribution. The mixture is analogue to the one described for the berngamma distribution.

Value

dbernexp gives the density (pdf), pbernexp gives the distribution function (cdf), qbernexp gives the quantile function (inverse cdf), and rbernexp generates random numbers.

Author(s)

Lukas Gudmundsson

See Also

Exponential, berngamma

Examples

data(opsprecip)

(ts <- startbernexp(opsprecip[,1]))
hist(opsprecip[,1],freq=FALSE)
lines(seq(0,max(opsprecip[,1])),
     dbernexp(seq(0,max(opsprecip[,1])),
              prob=ts$prob,
              rate=ts$rate),
     col="red")

pp <- seq(0.01,0.99,by=0.01)
qq <- quantile(opsprecip[,1],probs=pp)

plot(qq,pp)
lines(qberexp(pp,
berngamma

The Bernoulli-Gamma distribution

Description

Density, distribution function, quantile function and random generation for the Bernoulli-Gamma distribution with parameters \( \text{prob} \), shape, and scale.

Usage

\[
dberngamma(x, \text{prob}, \text{scale}, \text{shape})
\]
\[
pberngamma(q, \text{prob}, \text{scale}, \text{shape})
\]
\[
qberngamma(p, \text{prob}, \text{scale}, \text{shape})
\]
\[
rberngamma(n, \text{prob}, \text{scale}, \text{shape})
\]

Arguments

\( x, q \) vector of quantiles.
\( p \) vector of probabilities.
\( \text{prob} \) probability of non-zero event.
\( n \) number of random samples.
\( \text{scale}, \text{shape} \) shape and scale parameters of the gamma distribution.

Details

Mixture of the Bernoulli and the Gamma distribution. The Bernoulli distribution is used to model the occurrence of zero values with the probability of \( 1 - \text{prob} \). Non-zero values follow the Gamma distribution with shape and scale parameters.

The probability density function (PDF) is defined as:

\[
g(x) = \begin{cases} 
\pi \cdot \gamma(x) & \text{if } x > 0 \\
1 - \pi & \text{if } x \leq 0 
\end{cases}
\]
where $\gamma(x)$ is the probability density function of the gamma distribution and $\pi$ is probability of a non-zero event.

The cumulative distribution function (CDF) is defined as:

$$G(x) = \begin{cases} 
1 - \pi + \pi \cdot \Gamma(x) & \text{if } x > 0 \\
1 - \pi & \text{if } x \leq 0 
\end{cases}$$

where $\Gamma(x)$ is the cumulative distribution function of the gamma distribution.

The quantile function (inverse of the CDF) is defined as

$$G^{-1}(p) = \begin{cases} 
\Gamma^{-1}\left(\frac{p - 1 + \pi}{\pi}\right) & \text{if } \pi > 1 - p \\
0 & \text{if } p \leq 1 - p 
\end{cases}$$

where $\Gamma^{-1}(p)$ is the inverse CDF of the gamma distribution and $p$ is a probability.

**Value**

dberngamma gives the density (pdf), pberngamma gives the distribution function (cdf), qberngamma gives the quantile function (inverse cdf), and rberngamma generates random deviates.

**Note**

The implementation is largely based on the bgamma family in the CaDENCE-package (Cannon, 2012) that was only available as test version at time of implementation (Mar. 2012). The CaDENCE-package is available at [http://www.eos.ubc.ca/~acannon/CaDENCE/](http://www.eos.ubc.ca/~acannon/CaDENCE/).


**Author(s)**

Lukas Gudmundsson

**References**


See Also

GammaDist

Examples

data( obsprecip )

(ts <- startberngamma( obsprecip[,1] ))
hist( obsprecip[,1], freq = FALSE )
lines( seq(0,20), dberngamma(0:20, prob = ts$prob, 
scale = ts$scale, 
shape = ts$shape ), 
col = "red" )

pp <- seq(0.01, 0.99, by = 0.01)
qq <- quantile( obsprecip[,1], probs = pp )
plot(qq, pp)
lines(qberngamma(pp, 
prob = ts$prob, 
scale = ts$scale, 
shape = ts$shape ), 
pp, col = "red" )

plot(qq, pp)
lines(qq, 
pberngamma(qq, 
prob = ts$prob, 
scale = ts$scale, 
shape = ts$shape ), 
col = "red" )

hist(rberngamma(1000, 
prob = ts$prob, 
scale = ts$scale, 
shape = ts$shape ), freq = FALSE )
Describes, distribution function, quantile function and random generation for the Bernoulli-Log-Normal distribution with parameters prob, meanlog, and sdlog.

Usage

\[
\begin{align*}
dbernlnorm(x, \text{prob}, \text{meanlog}, \text{sdlog}) \\
pbernlnorm(q, \text{prob}, \text{meanlog}, \text{sdlog}) \\
qbernlnorm(p, \text{prob}, \text{meanlog}, \text{sdlog}) \\
rbernlnorm(n, \text{prob}, \text{meanlog}, \text{sdlog})
\end{align*}
\]

Arguments

- \(x, q\) vector of quantiles.
- \(p\) vector of probabilities.
- \(\text{prob}\) probability of non-zero event.
- \(n\) number of random samples.
- \(\text{meanlog, sdlog}\) meanlog and sdlog parameters of the Log-Normal distribution.

Details

Mixture of Bernoulli and Log-Normal distribution. The mixture is analogue to the one described for the berngamma distribution.

Value

- \(dbernlnorm\) gives the density (pdf), \(pbernlnorm\) gives the distribution function (cdf), \(qbernlnorm\) gives the quantile function (inverse cdf), and \(rbernlnorm\) generates random deviates.

Note

The implementation is largely based on the bnorm family in the CaDENCE-package (Cannon, 2012) that was only available as test version at time of implementation (Mar. 2012). The CaDENCE-package is available at http://www.eos.ubc.ca/~acannon/CaDENCE/.

Author(s)

Lukas Gudmundsson

References

See Also

Lognormal, berngamma

Examples

data(obsprecip)

(ts <- startbernlnorm(obsprecip[,1]))
hist(obsprecip[,1],freq=FALSE)
lines(seq(0,20),dbernlnorm(0:20,
   prob=ts$prob,
   meanlog=ts$meanlog,
   sdlog=ts$sdlog),
   col="red")

pp <- seq(0.01,0.99,by=0.01)
qq <- quantile(obsprecip[,1],probs=pp)

plot(qq,pp)
lines(qbernlnorm(pp,
   prob=ts$prob,
   meanlog=ts$meanlog,
   sdlog=ts$sdlog),
   pp,col="red")

plot(qq,pp)
lines(qq,
   pbernlnorm(qq,
   prob=ts$prob,
   meanlog=ts$meanlog,
   sdlog=ts$sdlog),
   col="red")

hist(rbernlnorm(1000,prob=ts$prob,
   meanlog=ts$meanlog,
   sdlog=ts$sdlog),freq=FALSE)

berweibull

The Bernoulli-Weibull distribution

Description

Density, distribution function, quantile function and random generation for the Bernoulli-Weibull distribution with parameters prob, shape, and scale.
**Usage**

- `dbernweibull(x, prob, scale, shape)`
- `pbernweibull(q, prob, scale, shape)`
- `qbernweibull(p, prob, scale, shape)`
- `rbernweibull(n, prob, scale, shape)`

**Arguments**

- `x, q` vector of quantiles.
- `p` vector of probabilities.
- `prob` probability of non-zero event.
- `n` number of random samples.
- `scale, shape` shape and scale parameters of the weibull distribution.

**Details**

Mixture of Bernoulli and Weibull distribution. The mixture is analogue to the one described for the `berngamma` distribution.

**Value**

- `dbernweibull` gives the density (pdf), `pbernweibull` gives the distribution function (cdf), `qbernweibull` gives the quantile function (inverse cdf), and `rbernweibull` generates random deviates.

**Note**

The implementation is largely based on the `bweibull` family in the CaDENCE-package (Cannon, 2012) that was only available as test version at time of implementation (Mar. 2012). The CaDENCE-package is available at [http://www.eos.ubc.ca/~acannon/CaDENCE/](http://www.eos.ubc.ca/~acannon/CaDENCE/).

**Author(s)**

Lukas Gudmundsson

**References**


**See Also**

`Weibull, berngamma`
Examples

data(obsprecip)

(ts <- startbernweibull(obsprecip[,1]))
hist(obsprecip[,1], freq=FALSE)
lines(seq(0, max(obsprecip[,1]))),
    dbernweibull(seq(0, max(obsprecip[,1]))),
    prob=ts$prob,
    shape=ts$shape,
    scale=ts$scale),
    col="red")

pp <- seq(0.01, 0.99, by=0.01)
qq <- quantile(obsprecip[,1], probs=pp)

plot(qq, pp)
lines(qbernweibull(pp,
    prob=ts$prob,
    scale=ts$scale,
    shape=ts$shape),
    pp, col="red")

plot(qq, pp)
lines(qq,
    pbernweibull(qq,
        prob=ts$prob,
        scale=ts$scale,
        shape=ts$shape),
    col="red")

hist(rbernweibull(1000, prob=ts$prob,
    shape=ts$shape,
    scale=ts$scale), freq=TRUE)


fitQmap

Quantile mapping

Description

fitQmap identifies the parameters of different quantile mapping methods. doQmap performs quantile mapping using previously identified parameters.

Usage

fitQmap(obs, mod, method=c("PTF", "DIST", "RQUANT", "QUANT", "SSPLIN"), ...)
doQmap(x, fobj, ...)
Arguments

- **obs**: numeric vector, column matrix or data.frame with observed time series.
- **mod**: numeric vector, column matrix or data.frame with modelled time series corresponding to obs.
- **method**: A character string indicating the method to be used. See Details.
- **x**: numeric vector or a column matrix of modelled time series. Should have the same number of columns as obs.
- **fobj**: output from fitQmap (or of method defined via method).
- **...**: arguments passed to the method specified by method.

Details

The method argument decides upon which method for quantile mapping is used:

- "PTF" selects fitQmapPTF.
- "DIST" selects fitQmapDIST
- "RQUANT" selects fitQmapRQUANT
- "QUANT" selects fitQmapQUANT
- "SSPLIN" selects fitQmapSSPLIN

doQmap investigates the class of fobj and chooses the appropriate method for quantile mapping.

Value

fitQmap returns an object which class and structure depends on the selected method (see Details).

doQmap returns a numeric vector, matrix or data.frame depending on the format of x.

Author(s)

Lukas Gudmundsson

References


See Also

fitQmapDIST, fitQmapPTF, fitQmapRQUANT, fitQmapQUANT, fitQmapSSPLIN
Examples

data(obsprecip)
data(modprecip)

## call to fitQmapPTF and doQmapPTF
qm1.fit <- fitQmap(obsprecip, modprecip,
                  method="PTF",
                  transfun="expasymp",
                  cost="RSS", wett.day=TRUE)
qm1 <- doQmap(modprecip, qm1.fit)

## call to fitQmapDIST and doQmapDIST
qm2.fit <- fitQmap(sqrt(obsprecip), sqrt(modprecip),
                  method="DIST", qstep=0.001,
                  transfun="berngamma")
qm2 <- doQmap(sqrt(modprecip), qm2.fit)^2

## call to fitQmapRQUANT and doQmapRQUANT
qm3.fit <- fitQmap(obsprecip, modprecip,
                  method="RQUANT", qstep=0.01)
qm3 <- doQmap(modprecip, qm3.fit, type="linear")

## call to fitQmapRQUANT and doQmapRQUANT
qm4.fit <- fitQmap(obsprecip, modprecip,
                  method="QUANT", qstep=0.01)
qm4 <- doQmap(modprecip, qm4.fit, type="tricub")

## call to fitQmapSSPLIN and doQmapSSPLIN
qm5.fit <- fitQmap(obsprecip, modprecip, qstep=0.01,
                  method="SSPLIN")
qm5 <- doQmap(modprecip, qm5.fit)

sqrtquant <- function(x, qstep=0.001){
  qq <- quantile(x, prob=seq(0,1,by=qstep))
  sqrt(qq)
}

op <- par(mfrow=c(1,3))
for(i in 1:3)
  plot(sqrtquant(modprecip[,i]),
       sqrtquant(obsprecip[,i],pch=19,col="gray",
                 main=names(obsprecip)[i])
       lines(sqrtquant(modprecip[,i]),
             sqrtquant(qm1[,i],col=1)
       lines(sqrtquant(modprecip[,i]),
             sqrtquant(qm2[,i],col=2)
       lines(sqrtquant(modprecip[,i]),
             sqrtquant(qm3[,i],col=3)
       lines(sqrtquant(modprecip[,i]),
             sqrtquant(qm4[,i],col=4)
       lines(sqrtquant(modprecip[,i]),
             sqrtquant(qm5[,i],col=5)
fitQmapDIST

Quantile mapping using distribution derived transformations

Description

fitQmapDIST fits a theoretical distribution to observed and to modelled time series and returns these parameters as well as a transfer function derived from the distribution. doQmapDIST uses the transfer function to transform the distribution of the modelled data to match the distribution of the observations.

Usage

fitQmapDIST(obs, mod, ...)
## Default S3 method:
fitQmapDIST(obs, mod, distr="berngamma", start.fun, qstep=NULL, mlepar,...)
## S3 method for class 'matrix'
fitQmapDIST(obs, mod, ...)
## S3 method for class 'data.frame'
fitQmapDIST(obs, mod, ...)

doQmapDIST(x, fobj, ...)
## Default S3 method:
doQmapDIST(x, fobj, ...)
## S3 method for class 'matrix'
doQmapDIST(x, fobj, ...)
## S3 method for class 'data.frame'
doQmapDIST(x, fobj, ...)

Arguments

obs numeric vector, column matrix or data.frame with observed time series.
mod numeric vector, column matrix or data.frame with modelled time series, corresponding to obs.
distr A character string "name" naming a distribution for which the corresponding density function (dname), the corresponding distribution function (pname) and the quantile function (qname) must be defined (see for example GammaDist, berngamma or bernweibull.
start.fun function estimating starting values for parameter optimisation. Default starting values are provided for berngamma, bernweibull, bernlnorm, bernexp and the distributions mentioned in the documentation of mledist.

qstep NULL or a numeric value between 0 and 1. If !is.null(qstep) than mod and obs are aggregated to quantiles before model identification as: quantile(x, probs=seq(0,1,by=qstep). This effectively reduces the sample-size and can be used to speedup computations - but may render estimates less reliable.

mlepar a named list. Names correspond to parameters passed to mledist note that start may be overwritten by start.fun See examples.

x numeric vector or a column matrix of modelled time series

fobj output from fitQmapDIST

... Further arguments passed to methods

Details

Quantile mapping using distribution derived transformations to adjust the distribution of a modelled variable (P_m) such that it matches the distribution of an observed variable (P_o). The distribution derived transfer function is defined as

\[ P_o = F_o^{-1}(F_m(P_m)) \]

where F is a CDF and F^{-1} is the corresponding quantile function (inverse CDF). The subscripts o and m indicate parameters of the distribution that correspond to observed and modelled data respectively.

Value

fitQmapDIST returns an object of class fitQmapDIST containing following elements:

tfun The function used to transform the distribution of modelled values such that the distribution of observations. The function is build internally based on the distribution function ("pname") and quantile function ("qname") corresponding to distr.

par A matrix. The (named) columns correspond to the parameters of the distribution specified in distr estimated for the observed (suffix .o) and the modelled (suffix .m) data. The rows correspond to each pair of time series in obs and mod.

doQmapDIST returns a numeric vector, matrix or data.frame depending on the format of x.

Author(s)

Lukas Gudmundsson
References


For a general assessment of the methods see:


See Also
doQmap, startberngamma, berngamma, startbernweibull, bernweibull, startbernlnorm, bernlnorm, startbernexp, bernexp, mledist, fitdist

Examples

data(obsprecip)
data(modprecip)

qm.fit <- fitQmapDIST(obsprecip[,1],modprecip[,1],
distr="berngamma",
qstep=0.001)
qm <- doQmapDIST(modprecip[,1],qm.fit)

qm.lnorm.fit <- fitQmapDIST(obsprecip[,1],modprecip[,1],
distr="bernlnorm",
qstep=0.001)
qm.lnorm <- doQmapDIST(modprecip[,1],qm.lnorm.fit)

qm.weibu.fit <- fitQmapDIST(obsprecip[,1],modprecip[,1],
distr="bernweibull",
qstep=0.001)
qm.weibu <- doQmapDIST(modprecip[,1],qm.weibu.fit)

qm.exp.fit <- fitQmapDIST(sqrt(obsprecip[,1]),sqrt(modprecip[,1]),
distr="bernexp",
qstep=0.001)
qm.exp <- doQmapDIST(sqrt(modprecip[,1]),qm.exp.fit)^2

## utility function.
## plots are easier to investigate if
## precipitation data are sqrt transformed
sqrtquant <- function(x,qstep=0.01){
  qq <- quantile(x,prob=seq(0,1,by=qstep))
  sqrt(qq)
}

plot(sqrtquant(modprecip[,1]),
     sqrtquant(obsprecip[,1]))
lines(sqrtquant(modprecip[,1]),
     sqrtquant(qm, col="red")
lines(sqrtquant(modprecip[,1]),
     sqrtquant(qm.lnorm), col="blue")
lines(sqrtquant(modprecip[,1]),
     sqrtquant(qm.weibu), col="green")
lines(sqrtquant(modprecip[,1]),
     sqrtquant(qm.exp), col="orange")
legend("topleft",
       legend=c("berngamma", "bernlnorm", "bernweibull", "bernexp"),
       lty=1,
       col=c("red", "blue", "green", "orange"))

## effect of qstep on speed of fitting process:
system.time(
  qm.a.fit <- fitQmapDIST(obsprecip[,2], modprecip[,2],
                           distr="berngamma",
                           start.fun=startberngamma,
                           qstep=0.001)
)

system.time(
  qm.b.fit <- fitQmapDIST(obsprecip[,2], modprecip[,2],
                           distr="berngamma",
                           start.fun=startberngamma,
                           qstep=0.01)
)

qm.a <- doQmapDIST(modprecip[,2], qm.a.fit)
qm.b <- doQmapDIST(modprecip[,2], qm.b.fit)

plot(sqrtquant(modprecip[,2]),
     sqrtquant(obsprecip[,2]))
lines(sqrtquant(modprecip[,2]),
     sqrtquant(qm.a, col="red")
lines(sqrtquant(modprecip[,2]),
     sqrtquant(qm.b, col="blue")
legend("topleft",
       legend=c("qstep=0.001", "qstep=0.01"),
       col=c("red", "blue"),
       lty=1)

## method for matrix
fitQmapPTF

Quantile mapping using parametric transformations

Description

fitQmapPTF fits a parametric transformations to the quantile-quantile relation of observed and modelled values. doQmapPTF uses the transformation to adjust the distribution of the modelled data to match the distribution of the observations.

Usage

fitQmapPTF(obs, mod, ...) # Default S3 method:
fitQmapPTF(obs, mod, transfun=c("power","linear","expasymp","scale","power.x0","expasymp.x0"), wet.day=TRUE, cost=c("RSS","MAE"), qstep=0.001, opar, ...)

# S3 method for class 'matrix'
fitQmapPTF(obs, mod, ...)

# S3 method for class 'data.frame'
fitQmapPTF(obs, mod, ...)

doQmapPTF(x,fobj,...) # Default S3 method:
doQmapPTF(x,fobj,...)

# S3 method for class 'matrix'
doQmapPTF(x,fobj,...)

# S3 method for class 'data.frame'
doQmapPTF(x,fobj,...)
Arguments

- **obs**: numeric vector, column matrix or data.frame with observed time series.
- **mod**: numeric vector, column matrix or data.frame with modelled time series, corresponding to obs.
- **transfun**: either a character string specifying a predefined function used for the transformation (see Details) or a function with x as first argument e.g. function(x, a, b)(a+x*b)
- **wet.day**: logical indicating whether to perform wet day correction or not. OR a numeric threshold below which all values are set to zero. See Details.
- **cost**: Criterion for optimisation. "RSS" minimises the residual sum of squares and produces a least square fit. "MAE" minimises the mean absolute error, which is less sensitive to outliers.
- **qstep**: NULL or a numeric value between 0 and 1. See Details.
- **opar**: a named list with arguments passed to optim. Note that method is chosen automatically. If transfun is a character string default values for par are available (but can be overwritten). See examples.
- **x**: numeric vector or a column matrix of modelled time series
- **fobj**: output from fitQmapDIST
- **...**: Further arguments passed to methods

Details

Before further computations the empirical cumulative distribution functions (CDF) of the observed (obs) and modelled (mod) are estimated. If !is.null(qstep) than mod and obs are aggregated to quantiles before model identification as: quantile(x, probs=seq(0,1,by=qstep)). If !is.null(qstep) than mod and obs are sorted to produce an estimate of the empirical CDF. In case of different length of mod and obs than quantile(x, probs=seq(0,1,1en=n)]is used, where n <- min(length(obs),length(mod)).

NOTE that large values of qstep effectively reduce the sample-size and can be used to speedup computations - but may render estimates less reliable.

wet.day is intended for the use for precipitation data. Wet day correction attempts to equalise the fraction of days with precipitation between the observed and the modelled data. If wet.day=TRUE the empirical probability of nonzero observations is found (obs>0) and the corresponding modelled value is selected as a threshold. All modelled values below this threshold are set to zero. If wet.day is numeric the same procedure is performed after setting all obs<wet.day to zero. The transformations are then only fitted to the portion of the distributions corresponding to observed wet days. See Piani et. al (2010) for further explanations.

Transformations (transfun):

NOTE: If wet day correction is performed (see wet.day), the transformations are only fitted to the portion of the empirical CDF with nonzero observations.

A series of predefined transformations are available and can be accessed by setting transfun to one of the following options (P_o refers to observed and P_m to modelled CDFs):

"power" :

\[ P_o = b \cdot P_m^c \]
"linear":

\[ P_o = a + b \cdot P_m \]

"expasympt" (exponential tendency to an asymptote):

\[ P_o = (a + b \cdot P_m) \cdot (1 - e^{(-P_m/\tau)}) \]

"scale":

\[ P_o = b \cdot P_m \]

"power.x0":

\[ P_o = b \cdot (P_m - x0)^c \]

"expasympt.x0" (exponential tendency to an asymptote):

\[ P_o = (a + b \cdot P_m) \cdot (1 - e^{(-(P_m - x0)/\tau)}) \]

Value

`fitQmapPTF` returns an object of class `fitQmapPTF` containing following elements:

- `tfun` The function used to transform the distribution of the modelled values to match the distribution of the observations.
- `par` A matrix. The (named) columns correspond to the parameters of the transfer function. The rows correspond to pairs of time series in `obs` and `mod`.
- `wet.day` logical, indicating whether to perform wet day correction or not. OR a numeric threshold below which all values are set to zero.

`doQmapPTF` returns a numeric vector, matrix or data.frame depending on the format of `x`.

**Author(s)**

Lukas Gudmundsson

**References**

The implementation is closely related to the methods published in:


For a general assessment of the methods see:

See Also

`fitQmap`, `optim` 

Examples

data(obsprecip)
data(modprecip)

## data.frame example
qm.fit <- fitQmapPTF(obsprecip, modprecip, 
                      transfun="power.x0", 
                      cost="RSS", wet.day=TRUE, 
                      qstep=0.001)
qm <- doQmapPTF(modprecip, qm.fit)

## application to "single time series"
qm.b.fit <- fitQmapPTF(obsprecip[,1], modprecip[,1], 
                        transfun="expasympt.x0", 
                        cost="RSS", wet.day=0.1, 
                        qstep=0.001)
qm.b <- doQmapPTF(modprecip[,1], qm.b.fit)
qm.c.fit <- fitQmapPTF(obsprecip[,1], modprecip[,1], 
                        transfun="expasympt", 
                        cost="RSS", wet.day=TRUE, 
                        qstep=0.001)
qm.c <- doQmapPTF(modprecip[,1], qm.c.fit)

## user defined transfer function
## and usage of the 'opar' argument
## (same as transfun="power")
myff <- function(x,a,b) a*x^b
qm3.fit <- fitQmapPTF(obsprecip[,1], modprecip[,1], 
                      transfun=myff, 
                      opar=list(par=c(a=1,b=1)), 
                      cost="RSS", wet.day=TRUE, 
                      qstep=0.001)
qm3 <- doQmapPTF(modprecip[,1], qm3.fit)

sqrtquant <- function(x,qstep=0.01){
  qq <- quantile(x, prob=seq(0,1,by=qstep))
  sqrt(qq)
}
plot(sqrtquant(modprecip[,1]), 
     sqrtquant(obsprecip[,1]))
lines(sqrtquant(modprecip[,1]), 
      sqrtquant(qm[,1]),col="red")
lines(sqrtquant(modprecip[,1]), 
      sqrtquant(qm.b),col="blue")
lines(sqrtquant(modprecip[,1]), 
      sqrtquant(qm.c),col="green")
**Description**

`fitQmapQUANT` estimates values of the empirical cumulative distribution function of observed and modelled time series for regularly spaced quantiles. `doQmapQUANT` uses these estimates to perform quantile mapping.

**Usage**

```r
fitQmapQUANT(obs, mod, ...) 
```

---

**Arguments**

- **obs** - numeric vector, column matrix or data.frame with observed time series.
- **mod** - numeric vector, column matrix or data.frame with modelled time series, corresponding to `obs`.
- **wet.day** - logical indicating whether to perform wet day correction or not. OR a numeric threshold below which all values are set to zero. See details.
- **qstep** - a numeric value between 0 and 1. The quantile mapping is fitted only for the quantiles defined by `quantile(0,1,probs=seq(0,1,by=qstep)).`
nboot

number of bootstrap samples used for estimation of the observed quantiles. If nboot==1 the estimation is based on all (and not resampled) data. See details.

x

numeric vector or a column matrix of modelled time series

fobj

output from fitQmapQUANT

type

type of interpolation between the fitted transformed values. See details.

Details

fitQmapQUANT estimates the empirical cumulative distribution function of mod and obs for the quantiles defined by seq(0,1,by=qstep). The quantiles of mod are estimated using the empirical quantiles. If nboot>1 the quantiles of obs are estimated as the mean of nboot bootstrap samples (if nboot>1).

doQmapQUANT transforms the variable x based on the transformation identified using fitQmapQUANT. For all values that are not in quantile(mod,probs=seq(0,1,by=qstep)) the transformation is estimated using interpolation of the fitted values. Available interpolation options are:

type="linear": linear interpolation using approx, but using the extrapolation suggested by Boe et al. (2007) for values of x larger than max(mod) (constant correction).

type="tricube": monotonic tricubic spline interpolation using splinefun. Spline interpolation is performed using a _monotone_ Hermite spline (method="monoh.FC" in splinefun).

wet.day is intended for the use for precipitation data. Wet day correction attempts to equalise the fraction of days with precipitation between the observed and the modelled data. If wet.day=TRUE the empirical probability of nonzero observations is found (obs>=0) and the corresponding modelled value is selected as a threshold. All modelled values below this threshold are set to zero. If wet.day is numeric the same procedure is performed after setting all obs<wet.day to zero.

Value

fitQmapQUANT returns an object of class fitQmapQUANT containing following elements:

par

A list containing:

par$modq

a matrix. Each column i corresponds to the output of quantile(mod[,i],probs=seq(0,1,by=qstep)).

par$fitq

observed empirical quantiles corresponding to par$modq.

wet.day

logical, indicating whether to perform wet day correction or not. OR a numeric threshold below which all values are set to zero.

doQmapQUANT returns a numeric vector or matrix depending on the format of x.

Author(s)

Lukas Gudmundsson
References


For a general assessment of the methods see:


See Also

fitQmap

Examples

data(obsprecip)
data(modprecip)

qm.fit <- fitQmapQUANT(obsprecip[,2],modprecip[,2],
qstep=0.1,nboot=1,wet.day=TRUE)
qm.a <- doQmapQUANT(modprecip[,2],qm.fit,type="linear")
qm.s <- doQmapQUANT(modprecip[,2],qm.fit,type="tricub")

sqrtquant <- function(x,qstep=0.01){
  qq <- quantile(x,prob=seq(0,1,by=qstep))
  sqrt(qq)
}

plot(sqrtquant(modprecip[,2]),
    sqrtquant(obsprecip[,2]))
lines(sqrtquant(modprecip[,2]),
    sqrtquant(qm.a),col="red")
lines(sqrtquant(modprecip[,2]),
    sqrtquant(qm.s),col="blue")
points(sqrt(qm.fit$par$modq),sqrt(qm.fit$par$fitq),
    pch=19,cex=0.5,col="green")
legend("topleft",
    legend=c("linear","tricub","support"),
    lty=c(1,1,NA),pch=c(NA,NA,19),
    col=c("red","blue","green"))

qm2.fit <- fitQmapQUANT(obsprecip,modprecip,
qstep=0.01,nboot=1,wet.day=TRUE)
qm2 <- doQmapQUANT(modprecip,qm2.fit,type="tricub")

op <- par(mfrow=c(1,3))
for(i in 1:3){
  plot(sqrtquant(modprecip[,i]),
...
fitQmapRQUANT

Non-parametric quantile mapping using robust empirical quantiles.

Description

fitQmapRQUANT estimates the values of the quantile-quantile relation of observed and modelled time series for regularly spaced quantiles using local linear least square regression. doQmapRQUANT performs quantile mapping by interpolating the empirical quantiles.

Usage

fitQmapRQUANT(obs, mod, ...)

## Default S3 method:
fitQmapRQUANT(obs, mod, wet.day=TRUE, qstep=0.01,
nlls = 10, nboot = 10,...)

## S3 method for class 'matrix'
fitQmapRQUANT(obs, mod, ...)

## S3 method for class 'data.frame'
fitQmapRQUANT(obs, mod, ...)

doQmapRQUANT(x, fobj, ...)

## Default S3 method:
doQmapRQUANT(x, fobj, slope.bound=c(lower=0, upper=Inf),
type=c("linear","linear2","tricub"),...)

## S3 method for class 'matrix'
doQmapRQUANT(x, fobj, ...)

## S3 method for class 'data.frame'
doQmapRQUANT(x, fobj, ...)

Arguments

obs numeric vector, column matrix or data.frame with observed time series.

mod numeric vector or column matrix/data.frame with modelled time series, corresponding to obs
wetNday logical, indicating whether to perform wet day correction or not. OR a numeric threshold below which all values are set to zero. See details.

qstep A numeric value between 0 and 1. The values quantile-quantile plot are estimated at the position of the values defined by: quantile(mod,probs=seq(0,1,by=qstep)).

nlls number of nearest data points to apply in the local regression

nboot number of bootstrap samples in the estimation of the transformation. If nboot=True the estimation is based on all (and not resampled) data.

x numeric vector or a column matrix of modelled time series

fobj output from fitQmapRQUANT

slopeNbound bounds for the slopes in case of extrapolation. Applies only if type="linear2"

type type of interpolation between the fitted transformed values. See details

... Further arguments passed to methods

Details

fitQmapRQUANT produces a robust estimate of the empirical quantile-quantile plot (QQ-plot) of mod vs obs for the seq(0,1,by=qstep) quantiles mod. The corresponding value of the quantiles of obs is estimated using local linear least squares regression. For each quantile of mod the nlls nearest data points in the QQ-plot are identified and used to fit a local regression line. This regression line is then used to estimate value of the quantile of obs. The estimation is replicated for nboot bootstrap samples and the mean of the bootstrap replicates is returned.

This procedure results in a table with empirical quantiles of mod and a corresponding table with robust estimates of the empirical quantiles of obs.

doQmapRQUANT uses the tables of robust empirical quantiles identified using fitQmapRQUANT to transform the variable x. For values that are not in quantile(mod,probs=seq(0,1,by=qstep)) the transformation is estimated using interpolation of the fitted values. Available interpolation options are:

type="linear": linear interpolation using approx, but using the extrapolation suggested by Boe et al. (2007) for values of x larger than max(mod) (constant correction).

type="linear2": linear interpolation using approx. For any value of x outside range(mod) the transformation is extrapolated using the slope of the local linear least squares regression at the outer most points.

type="tricube": monotonic tricubic spline interpolation using splinefun. Spline interpolation is performed using a _monotone_ Hermite spline (method="monoH.FC" in splinefun).

wetNday is intended for the use for precipitation data. Wet day correction attempts to equalise the fraction of days with precipitation between the observed and the modelled data. If wetNday=True the empirical probability of nonzero observations is found (obs>=0) and the corresponding modelled value is selected as a threshold. All modelled values below this threshold are set to zero. If wetNday is numeric the same procedure is performed after setting all obs<wetNday to zero.
**Value**

`fitQmapRQUANT` returns an object of class `fitQmapRQUANT` containing following elements:

- **par**
  A list containing:
  - `par$modq` a matrix. Each column i corresponds to the output of `quantile(mod[,1],probs=seq(0,1,by=qstep)).`
  - `par$fitq` the fitted values of the local linear least square regression corresponding to `par$modq`
  - `par$slope` a matrix. the columns correspond to the columns of `mod`. The rows contain the slope of the "lower" and the "upper" extreme points of the local linear fit and is used for extrapolation if `type="linear2"`.
- **wet.day**
  logical, indicating whether to perform wet day correction or not. OR a numeric threshold below which all values are set to zero.

`doQmapRQUANT` returns a numeric vector or matrix depending on the format of `x`.

**Author(s)**

John Bjornar Bremnes and Lukas Gudmundsson

**References**


**See Also**

`fitQmap`

**Examples**

```r
data(obsprecip)
data(modprecip)

## single series example
qm.fit <- fitQmapRQUANT(obsprecip[,2],modprecip[,2],
                           qstep=0.1,nboot=10,wet.day=TRUE)
qm.a <- doQmapRQUANT(modprecip[,2],qm.fit,type="linear")
qm.b <- doQmapRQUANT(modprecip[,2],qm.fit,type="tricub")

sqrtquant <- function(x,qstep=0.01){
  qq <- quantile(x,prob=seq(0,1,by=qstep))
  sqrt(qq)
}

plot(sqrtquant(modprecip[,2]),
sqrtquant(obsprecip[,2]))
lines(sqrtquant(modprecip[,2]),
sqrtquant(qm.a),col="red")
```
fitQmapSSPLIN

Quantile mapping using a smoothing spline

Description

fitQmapSSPLIN fits a smoothing spline to the quantile-quantile plot of observed and modelled time series. doQmapSSPLIN uses the spline function to adjust the distribution of the modelled data to match the distribution of the observations.

Usage

fitQmapSSPLIN(obs, mod, ...)
## Default S3 method:
fitQmapSSPLIN(obs, mod, wet.day=TRUE, qstep=0.01,
  spline.par, ...)
## S3 method for class 'matrix'
fitQmapSSPLIN(obs, mod, ...)
## S3 method for class 'data.frame'
fitQmapSSPLIN(obs, mod, ...)
doQmapSSPLIN(x, fobj, ...)  
## Default S3 method:  
doQmapSSPLIN(x, fobj, ...)  
## S3 method for class 'matrix'  
doQmapSSPLIN(x, fobj, ...)  
## S3 method for class 'data.frame'  
doQmapSSPLIN(x, fobj, ...)

Arguments

- **obs**: numeric vector, column matrix or data.frame with observed time series.
- **mod**: numeric vector, column matrix or data.frame with modelled time series, corresponding to obs.
- **wet.day**: logical, indicating whether to perform wet day correction or not. OR a numeric threshold below which all values are set to zero. See details.
- **qstep**: NULL or a numeric value between 0 and 1. See Details.
- **spline.par**: a named list with parameters passed to smooth.spline.
- **x**: numeric vector or a column matrix of modelled time series
- **fobj**: output from fitQmapDIST
- ... Further arguments passed to methods

Details

Before further computations the empirical cumulative distribution functions (CDF) of the observed (obs) and modelled (mod) are estimated. If !is.null(qstep) than mod and obs are aggregated to quantiles before model identification as: quantile(x, probs = seq(0, 1, by = qstep)). If !is.null(qstep) than mod and obs are sorted to produce an estimate of the empirical CDF. In case of different length of mod and obs than quantile(x, probs = seq(0, 1, len = n)) is used, where

\[ n \leftarrow \min(\text{length}(\text{obs}), \text{length}(\text{mod})) \]

NOTE that large values of qstep effectively reduce the sample-size and can be used to speedup computations - but may render estimates less reliable.

wet.day is intended for the use for precipitation data. Wet day correction attempts to equalise the fraction of days with precipitation between the observed and the modelled data. If wet.day=TRUE the empirical probability of nonzero observations is found (obs>=0) and the corresponding modelled value is selected as a threshold. All modelled values below this threshold are set to zero. If wet.day is numeric the same procedure is performed after setting all obs<wet.day to zero. The transformations are then only fitted to the portion of the distributions corresponding to observed wet days.

Value

fitQmapSSPLIN returns an object of class fitQmapSSPLIN containing following elements:

- **par**: A list containing objects of class smooth.spline.fit, which are equivalent to the value of the element fit in the output of smooth.spline. The spline coefficients are checked for monotony and adjusted if necessary by replacement with the previous value. If mod is a matrix the names of par correspond to colnames(mod).
fitQmapSSPLIN

wet.day logical, indicating whether to perform wet day correction or not. OR a numeric threshold below which all values are set to zero.

doQmapSSPLIN returns a numeric vector or matrix depending on the format of x.

Author(s)
Lukas Gudmundsson

References

See Also
fitQmap, smooth.spline

Examples
data(obsprecip)
data(modprecip)

qm.a.fit <- fitQmapSSPLIN(obsprecip[,2],modprecip[,2],
qstep=0.01,wet.day=TRUE)
qm.a <- doQmapSSPLIN(modprecip[,2],qm.a.fit)

## example on how to use spline.par
## (this example has little effect)
qm.b.fit <- fitQmapSSPLIN(obsprecip[,2],modprecip[,2],
qstep=0.01,wet.day=TRUE,
spline.par=list(cv=TRUE))
qm.b <- doQmapSSPLIN(modprecip[,2],qm.b.fit)

sqrtquant <- function(x,qstep=0.01){
  qq <- quantile(x,prob=seq(0,1,by=qstep))
  sqrt(qq)
}

plot(sqrtquant(modprecip[,2]),
sqrtquant(obsprecip[,2]))
lines(sqrtquant(modprecip[,2]),
sqrtquant(qm.a),col="red")
lines(sqrtquant(modprecip[,2]),
sqrtquant(qm.b),col="blue")
legend("topleft",legend=c("cv=FALSE","cv=TRUE"),
ly=1,col=c("red","blue"))

qm2.fit <- fitQmapSSPLIN(obsprecip,modprecip,
qstep=0.1,wet.day=TRUE)
obsprecip

```
qm2 <- doQmapSSPLIN(modprecip,qm2.fit)

op <- par(mfrow=c(1,3))
for(i in 1:3){
  plot(sqrtquant(modprecip[,i]),
       sqrtquant(obsprecip[,i]),
       main=names(qm2)[i])
  lines(sqrtquant(modprecip[,i]),
        sqrtquant(qm2[,i]), col="red")
}
par(op)
```

daily precipitation data at three locations in Norway.

Description

Observed (obsprecip) and simulated (modprecip) daily precipitation data for three locations in Norway covering the 1961-1990 period.

Usage

data(obsprecip)
data(modprecip)

Format

Data frame(s) with rows representing days and with the following 3 variables.

**Moss** Daily Precipitation at Moss [mm/day]

**Geiranger** Daily Precipitation at Geiranger [mm/day]

**Barkestad** Daily Precipitation at Barkestad [mm/day]

Details

The time series in obsprecip stem from the observation-system of the Norwegian Meteorological Institute.

The time series in modprecip are based on simulations of HIRHAM2/NorACIA regional climate model forced with simulation the HadAM3H. The simulation setup is further described in Forland et al. 2011. The simulations are free-running and there is consequently no direct correspondence in the temporal evolution of modprecip and obsprecip.

NOTE that all months in the modelled data (modprecip) have 30 days (in contrast to the observations (obsprecip) which have true calendar days.
Source

The observations are taken from the observation network of the Norwegian meteorological institute (www.met.no). The data are available for download at http://eklima.met.no.

References


Examples

data(obsprecip)
data(modprecip)

startbernexp

Rough parameter estimate for the Bernoulli-Exponential distribution

Description

Estimates rough starting values for the Bernoulli-Exponential distribution using the method of moments for the rate parameter. The probability of non-zero events is estimated as the fraction of values that are larger than zero.

Usage

startbernexp(x)

Arguments

x numeric vector.

Value

A list containing:

prob probability of non-zero event.
rate rate parameter of the Exponential distribution.

Note

In this package startbernexp is intended to be used in conjunction with fitQmapDIST (and mledist) with parameter distr="berexp".

Author(s)

Lukas Gudmundsson
See Also

fitQmapDIST, bernexp, fitdist

Examples

```r
gg <- rbernexp(n=300, prob=0.2, rate=1)
startbernexp(gg)
mledist(gg,"bernexp",startbernexp(gg))
```

---

**startberngamma**  
*Rough parameter estimate for the Bernoulli-Gamma distribution*

**Description**

Estimates rough starting values for the Bernoulli-Gamma distribution using the method of moments for the shape and the scale parameters. The probability of non-zero events is estimated as the fraction of values that are larger than zero.

**Usage**

```r
startberngamma(x)
```

**Arguments**

- `x` numeric vector.

**Value**

A list containing:

- `prob` probability of non-zero event.
- `scale` scale parameter of the gamma distribution.
- `shape` shape parameter of the gamma distribution.

**Note**

In this package `startberngamma` is intended to be used in conjunction with `fitQmapDIST` (and `mledist`) with parameter `distr="berngamma"`.

**Author(s)**

Lukas Gudmundsson

**See Also**

fitQmapDIST, berngamma, fitdist
**Examples**

```r
gg <- rbernlnorm(n=300, prob=0.2, scale=1, shape=1)
startbernlnorm(gg)
mledist(gg,"berngamma",startberngamma(gg))
```

---

**Description**

Estimates rough starting values for the Bernoulli-Log-Normal distribution using the method of moments for the meanlog and the sdlog parameters. The probability of non-zero events is estimated as the fraction of values that are larger than zero.

**Usage**

```r
startbernlnorm(x)
```

**Arguments**

- `x` numeric vector.

**Value**

A list containing:

- `prob` probability of non-zero event.
- `meanlog` meanlog parameter of the Log-Normal distribution.
- `sdlog` sdlog parameter of the Log-Normal distribution.

**Note**

In this package `startbernlnorm` is intended to be used in conjunction with `fitQmapDIST` (and `mledist`) with parameter `distr="bernlnorm"`.

**Author(s)**

Lukas Gudmundsson

**See Also**

- `fitQmapDIST`, `bernlnorm`, `fitdist`

**Examples**

```r
gg <- rbernlnorm(n=300, prob=0.2, meanlog=1, sdlog=1)
startbernlnorm(gg)
mledist(gg,"bernlnorm",startbernlnorm(gg))
```
Description

Estimates rough starting values for the Bernoulli-Weibull distribution using the method of moments for the shape and the scale parameters. The probability of non-zero events is estimated as the fraction of values that are larger than zero.

Usage

`startbernweibull(x)`

Arguments

- `x` numeric vector.

Value

A list containing:

- `prob` probability of non-zero event.
- `scale` scale parameter of the weibull distribution.
- `shape` shape parameter of the weibull distribution.

Note

In this package `startbernweibull` is intended to be used in conjunction with `fitQmapDIST` (and `mledist`) with parameter `distr="bernweibull"`.

Author(s)

Lukas Gudmundsson

See Also

`fitQmapDIST`, `bernweibull`, `fitdist`

Examples

```r
gg <- rbernweibull(n=300, prob=0.2, scale=1, shape=1)
startbernweibull(gg)
mledist(gg,"bernweibull",startbernweibull(gg))
```
Index

*Topic datasets
obsprecip, 30

*Topic distribution
bernexp, 2
berngamma, 4
bernlnorm, 7
bernweibull, 8

equalisation, histogram matching, probability mapping, distribution
qmapMpackage, 2

*Function (cdf) matching, quantile - quantile transformation, histogram
qmapMpackage, 2

*Topic mapping, model output statistics, bias correction, downscaling
qmapMpackage, 2

*Topic quantile mapping, quantile matching, cumulative distribution
qmapMpackage, 2

approx, 22, 25

bernexp, 2, 14, 15, 32
berngamma, 3, 4, 7–9, 13–15, 32
bernlnorm, 7, 14, 15, 33
bernweibull, 8, 13–15, 34

dbernexp (bernexp), 2
dberngamma (berngamma), 4
dbernlnorm (bernlnorm), 7
dbernweibull (bernweibull), 8
doQmap, 2, 15
doQmap (fitQmap), 10
doQmapDIST (fitQmapDIST), 13
doQmapPTF (fitQmapPTF), 17
doQmapQUANT (fitQmapQUANT), 21
doQmapRQUANT (fitQmapRQUANT), 24
doQmapSSPLIN (fitQmapSSPLIN), 27

Exponential, 3

fitdist, 15, 32–34
fitQmap, 2, 10, 20, 23, 26, 29
fitQmapDIST, 11, 13, 31–34
fitQmapPTF, 11, 17
fitQmapQUANT, 11, 21
fitQmapRQUANT, 11, 24
fitQmapSSPLIN, 11, 27

GammaDist, 6, 13

Lognormal, 8

mledist, 14, 15, 31–34
modprecip (obsprecip), 30

obsprecip, 30
optim, 20

pbernexp (bernexp), 2
pberngamma (berngamma), 4
pbernlnorm (bernlnorm), 7
pbernweibull (bernweibull), 8

qbernexp (bernexp), 2
qberngamma (berngamma), 4
qbernlnorm (bernlnorm), 7
qbernweibull (bernweibull), 8

qmap (qmapMpackage), 2
qmapMpackage, 2

rbernexp (bernexp), 2
rberngamma (berngamma), 4
rbernlnorm (bernlnorm), 7
rbernweibull (bernweibull), 8

smooth.spline, 28, 29

35
splinefun, 22, 25
startbernextp, 15, 31
startberngamma, 15, 32
startbernlnorm, 15, 33
startbernweibull, 15, 34

Weibull, 9