Package 'WQM'

October 11, 2024

Type Package

Title Wavelet-Based Quantile Mapping for Postprocessing Numerical Weather Predictions

Version 0.1.4

Author Ze Jiang [aut, cre] (<https://orcid.org/0000-0002-3472-0829>), Fiona Johnson [aut] (<https://orcid.org/0000-0001-5708-1807>)

Maintainer Ze Jiang <ze.jiang@unsw.edu.au>

Description The wavelet-based quantile mapping (WQM) technique is designed to correct biases in spatio-temporal precipitation forecasts across multiple time scales. The WQM method effectively enhances forecast accuracy by generating an ensemble of precipitation forecasts that account for uncertainties in the prediction process. For a comprehensive overview of the methodologies employed in this package, please refer to Jiang, Z., and Johnson, F. (2023) <doi:10.1029/2022EF003350>. The package relies on two packages for continuous wavelet transforms: 'WaveletComp', which can be installed automatically, and 'wmtsa', which is optional and available from the CRAN archive <https://cran.r-project.org/src/contrib/Archive/wmtsa/>. Users need to manually install 'wmtsa' from this archive if they prefer to use 'wmtsa' based decomposition.

License GPL (>= 3)

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

Imports MBC, WaveletComp, matrixStats, ggplot2

Suggests stats, tidyr, dplyr, wmtsa, scales, data.table, graphics, testthat (>= 3.0.0), knitr, rmarkdown, bookdown

Config/testthat/edition 3

RoxygenNote 7.3.2

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2024-10-11 08:00:18 UTC

Contents

`bc_cwt` *CWT based quantile mapping*

Description

CWT based quantile mapping

Usage

```
bc_cwt(
  data,
  subset,
  variable,
  theta = 0.1,
  QM = c("MBC", "MRS", "QDM"),
  number_sim = 5,
  wavelet = "morlet",
  dt = 1,
  dj = 1,
  method = "M2",
  block = 3,
  seed = NULL,
  PR.cal = FALSE,
  do.plot = FALSE,
  ...
)
```

Arguments

`data`	a list of input dataset
`subset`	a index of number denoting the subset for calibration
`variable`	a character string denoting the type of variable.
`theta`	threshold of rainfall.
`QM`	a character string denoting the qm method used.

number_sim	The total number of realizations.
wavelet	a character string denoting the wavelet filter to use in calculating the CWT.
dt	sampling resolution in the time domain.
dj	sampling resolution in the frequency domain.
method	Shuffling method, M1: non-shuffling and M2: shuffling. M2 by default.
block	Block size.
seed	Seed for shuffling process.
PR.cal	Logical value for phase randomization of calibration.
do.plot	Logical value for ploting.
...	Additional arguments for QDM.

Value

a list of post-processed data

---

fun_cwt_J *Function: Total number of decomposition levels*

Description

Function: Total number of decomposition levels

Usage

```
fun_cwt_J(n, dt, dj)
```

Arguments

n	sample size.
dt	sampling resolution in the time domain.
dj	sampling resolution in the frequency domain.

Value

the total number of decomposition levels.

fun_icwt *Inverse of continuous wavelet transform*

Description

Inverse of continuous wavelet transform

Usage

```
fun_icwt(x.wave, dt, dj, flag.wav = "WaveletComp", scale = NULL)
```

Arguments

x.wave	input complex matrix.
dt	sampling resolution in the time domain.
dj	sampling resolution in the frequency domain.
flag.wav	String for two different CWT packages.
scale	Wavelet scales.

Value

reconstructed time series

References

fun_stoch_sim_wave in PRSim, Brunner and Furrer, 2020.

Examples

```
set.seed(100)

dt<-1
dj<-1/8
flag.wav <- switch(2, "wmtsa", "WaveletComp")

n <- 100
x <- rnorm(n)
x.wave <- t(WaveletComp::WaveletTransform(x=x)$Wave)
rec <- fun_icwt(x.wave, dt, dj, flag.wav)

x.wt <- WaveletComp::analyze.wavelet(data.frame(x=x),"x",dt=dt,dj=dj)
rec_orig <- WaveletComp::reconstruct(x.wt,only.sig = FALSE, plot.rec = FALSE)$series$x.r

### compare to original series
op <- par(mfrow = c(1, 1), mar=c(3,3,1,1), mgp=c(1, 0.5, 0))
plot(1:n, x, type="l", lwd=5, xlab=NA, ylab=NA)
lines(1:n, rec, col="red",lwd=3)
lines(1:n, rec_orig, col="blue", lwd=1)
```

```
legend("topright",legend=c("Raw","Inverse","Inverse_orig"),
       lwd=c(5,3,1),bg="transparent",bty = "n",
       col=c("black","red","blue"),horiz=TRUE)
par(op)
```

`fun_ifft` *Inverse Fourier transform*

Description

Inverse Fourier transform

Usage

```
fun_ifft(x, do.plot = FALSE)
```

Arguments

`x`	input time series.
`do.plot`	Logical value of plot.

Value

reconstruction time series

References

fun_stoch_sim in PRSim, Brunner and Furrer, 2020.

Examples

```
x <- rnorm(100)
x.new <- fun_ifft(x, do.plot=TRUE)
```

`NWP.rain` *Australia NWP rainfall forecasts at lead 1h over Sydney region*

Description

A dataset containing 160 stations including observation and raw forecasts.

Usage

```
data(NWP.rain)
```

prsim *Phase randomization and shuffling*

Description

Phase randomization and shuffling

Usage

```
prsim(
  modulus,
  phases,
  noise_mat,
  method = c("M1", "M2")[2],
  size = 3,
  seed = NULL
)
```

Arguments

modulus	Modulus of complex values.
phases	Argument of complex values.
noise_mat	Complex matrix from random time series.
method	Shuffling method, M1: non-shuffling and M2: shuffling. M2 by default.
size	Block size.
seed	Seed for shuffling process.

Value

A new complex matrix

RankHist *Verification Rank and Histogram*

Description

Verification Rank and Histogram

Usage

```
RankHist(forecasts, observations, do.plot = FALSE)
```

Arguments

forecasts	A matrix of ensemble forecasts, in which the rows corresponds to locations and times and the columns correspond to the individual ensemble members.
observations	A vector of observations corresponding to the locations and times of the forecasts.
do.plot	Logical value of plot.

Value

A vector giving the rank of verifying observations relative to the corresponding ensemble forecasts. The verification rank historgram is plotted.

References

ensembleBMA::verifRankHist

---

sample *Sample data: Rainfall forecasts data*

Description

A dataset containing 2 stations including observation and raw forecasts.

Usage

```
data(sample)
```

Index