Package 'gmwmx'

July 23, 2025

Title Estimate Functional and Stochastic Parameters of Linear Models with Correlated Residuals

Version 1.0.3

License AGPL-3

Description Implements the Generalized Method of Wavelet Moments with Exogenous Inputs estimator (GMWMX) presented in Cucci, D. A., Voirol, L., Kermarrec, G., Montillet, J. P., and Guerrier, S. (2023) <doi:10.1007/s00190-023-01702-8>.
The GMWMX estimator allows to estimate functional and stochastic parameters of linear models with correlated residuals.
The 'gmwmx' package provides functions to estimate, compare and analyze models, utilities to load and work with Global Navigation Satellite System (GNSS) data as well as methods to compare results with the Maximum Likelihood Estimator (MLE) implemented in Hector.

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Depends R (>= 4.0.0)

Suggests rmarkdown, knitr, simts

VignetteBuilder knitr

LinkingTo Rcpp, RcppArmadillo

Imports Rcpp, fs, stringi, wv, Matrix, longmemo, rjson, ltsa

NeedsCompilation yes

Author Davide Antonio Cucci [aut],
Lionel Voirol [aut, cre],
Stéphane Guerrier [aut],
Jean-Philippe Montillet [ctb],
Gaël Kermarrec [ctb]

Maintainer Lionel Voirol <lionelvoirol@hotmail.com>

Repository CRAN

Date/Publication 2023-03-20 15:20:02 UTC

Contents

cola . . . 2
compare_fits . . . 3
create.gnssts . . . 4
create_A_matrix . . . 5
estimate_gmwmx . . . 5
estimate_hector . . . 7
PBO_get_offsets . . . 8
PBO_get_station . . . 8
plot.gnsstsmodel . . . 9
print.gnsstsmodel . . . 10
read.gnssts . . . 11
remove_outliers_hector . . . 12
write.gnssts . . . 13

Index **15**

`cola` *GNSS time series for PBO station COLA*

Description

Data from station COLA of the Plate Boundary Observatory

Usage

```
cola
```

Format

A `gnssts` object of the East position (dE) of the COLA station

Source

https://data.unavco.org/archive/gnss/products/position/COLA/COLA.pbo.igs14.pos

compare_fits *Compare graphically two* gnsstsmodel *objects.*

Description

Compare graphically two gnsstsmodel objects.

Usage

```
compare_fits(fit_1, fit_2, main = NULL, y_unit = "mm", x_unit = "days")
```

Arguments

fit_1	A gnsstsmodel object.
fit_2	A gnsstsmodel object.
main	A string specifying the plot title.
y_unit	A string specifying the y axis label.
x_unit	A string specifying the x axis label.

Value

No return value. Produce a plot comparing two estimated models.

Examples

```
## Not run:
data(cola)
fit_gmwmx_1 = estimate_gmwmx(x = cola,
                             theta_0 = c(0.1,0.1,0.1,0.1),
                             n_seasonal = 1,
                             model_string = "wn+matern")
fit_gmwmx_2 = estimate_gmwmx(x = cola,
                             theta_0 = c(0.1,0.1,0.1),
                             n_seasonal = 1,
                             model_string = "wn+powerlaw")
compare_fits(fit_gmwmx_1, fit_gmwmx_2)

## End(Not run)
```

`create.gnssts` *Create a gnssts object*

Description

Create a gnssts object

Usage

```
create.gnssts(t, y, jumps = NULL, sampling_period = 1)
```

Arguments

`t`	A `vector` specifying the time of each observation of the time series.
`y`	A `vector` specifying the values of each observation of the time series.
`jumps`	A `vector` specifying the time values for which there is a jump.
`sampling_period`	An `integer` specifying the sampling period.

Value

A `gnssts` object.

Examples

```
phase <- 0.45
amplitude <- 2.5
sigma2_wn <- 15
bias <- 0
trend <- 5 / 365.25
cosU <- amplitude * cos(phase)
sinU <- amplitude * sin(phase)
year <- 5
n <- year * 365
jump_vec <- c(200, 300, 500)
jump_height <- c(10, 15, 20)
nbr_sin <- 1
A <- create_A_matrix(1:n, jump_vec, n_seasonal = nbr_sin)
x_0 <- c(bias, trend, cosU, sinU, jump_height)
eps <- rnorm(n = n, sd = sqrt(sigma2_wn))
yy <- A %*% x_0 + eps
gnssts_obj <- create.gnssts(t = 1:length(yy), y = yy, jumps = jump_vec)
str(gnssts_obj)
```

`create_A_matrix` *Define matrix A of the functional model*

Description

Define matrix A of the functional model

Usage

```
create_A_matrix(t_nogap, jumps, n_seasonal)
```

Arguments

`t_nogap`	A `vector` specifying the index of the time series.
`jumps`	A `vector` specifying the time at which there is a mean shift of the time series. Should be specified to `NULL` if there is not presence of offsets in the signal.
`n_seasonal`	An `integer` specifying the number of sinusoidal signals in the time series.

Value

Matrix A in order to compute the functional component of the model in a linear fashion

Examples

```
n= 10*365
jump_vec <- c(200, 300, 500)
nbr_sin = 2
A <- create_A_matrix(1:n, jump_vec, n_seasonal = nbr_sin)
head(A)
A <- create_A_matrix(1:n, jumps = NULL, n_seasonal = nbr_sin)
head(A)
```

`estimate_gmwmx` *Estimate a stochastic model in a two-steps procedure using the GMWMX estimator.*

Description

Estimate a stochastic model in a two-steps procedure using the GMWMX estimator.

Usage

```
estimate_gmwmx(
  x,
  theta_0,
  n_seasonal = 1,
  model_string,
  method = "L-BFGS-B",
  maxit = 1e+06,
  ci = FALSE,
  k_iter = 1
)
```

Arguments

x	A gnssts object
theta_0	A vector specifying the initial values for the vector of parameter of the stochastic model considered.
n_seasonal	An integer specifying the number of seasonal component in the time series.
model_string	A string specifying the model to be estimated.
method	A string specifying the numerical optimization method that should be supplied to optim()
maxit	An integer specifying the maximum number of iterations for the numerical optimization procedure.
ci	A boolean specifying if confidence intervals for the estimated parameters should be computed.
k_iter	An integer specifying the number of time the two steps GMWMX procedure should be run.

Value

A gnsstsmodel object.

Examples

```
## Not run:
data(cola)
fit_gmwmx = estimate_gmwmx(x = cola,
                           theta_0 = c(0.1,0.1,0.1,0.1),
                           n_seasonal = 1,
                           model_string = "wn+matern")

## End(Not run)
```

`estimate_hector` *Estimate a stochastic model based on the MLE and the Hector implementation.*

Description

Estimate a stochastic model based on the MLE and the Hector implementation.

Usage

```
estimate_hector(
  x,
  n_seasonal = 1,
  model_string,
  likelihood_method = "AmmarGrag",
  cleanup = TRUE
)
```

Arguments

`x`	A `gnssts` object
`n_seasonal`	An `integer` specifying the number of seasonal component in the time series.
`model_string`	A `string` specifying the model to be estimated.
`likelihood_method`	A `string` taking either value "FullCov" or "AmmarGrag" that specify the method for the Likelihood computation.
`cleanup`	A `boolean` specifying if the files created by the estimation procedure should be cleaned.

Value

A `gnsstsmodel` object.

Examples

```
## Not run:
cola = PBO_get_station(station_name = "COLA", column = "dE", time_range = c(51130, 52000))
fit_mle = estimate_hector(x = cola,
                          n_seasonal = 1,
                          model_string = "wn+matern")


## End(Not run)
```

PBO_get_offsets *Extract offsets for a PBO station*

Description

Extract offsets for a PBO station

Usage

```
PBO_get_offsets(station_name)
```

Arguments

station_name	A string specifying the PBO station name.

Value

A vector specifying the offsets of a PBO station.

Examples

```
## Not run:
pbo_cola_offsets = PBO_get_offsets(station_name = "COLA")
pbo_cola_offsets

## End(Not run)
```

PBO_get_station *Load station data from PBO*

Description

Load station data from PBO

Usage

```
PBO_get_station(station_name, column, time_range = c(-Inf, Inf), scale = 1)
```

Arguments

station_name	A string specifying the PBO station name.
column	A string specifying the name of the column to extract.
time_range	A vector of 2 specifying the time range of data to extract.
scale	A scalar specifying an optional scaling parameter applied to the extracted data.

Value

A gnssts object that contains the data associated with the specified PBO station.

Examples

```
## Not run:
pbo_cola_data = PBO_get_station("COLA", column="dE")
str(pbo_cola_data)

## End(Not run)
```

`plot.gnsstsmodel` *Plotting method for a* `gnsstsmodel` *object.*

Description

Plotting method for a gnsstsmodel object.

Usage

```
## S3 method for class 'gnsstsmodel'
plot(
  x,
  main = NULL,
  y_unit = "mm",
  x_unit = "days",
  legend_position = "bottomright",
  legend_position_wv = "bottomleft",
  ...
)
```

Arguments

x	A gnsstsmodel object.
main	A string specifying the title of the plot.
y_unit	A string specifying the y axis label of the plot.
x_unit	A string specifying the x axis label of the plot.
legend_position	A string specifying the legend position of the plot.
legend_position_wv	A string specifying the legend position for the wv plot.
...	Additional graphical parameters.

Value

No return value. Plot a gnsstsmodel object.

Examples

```
## Not run:
data(cola)
fit_gmwmx = estimate_gmwmx(x = cola,
                           theta_0 = c(0.1,0.1,0.1,0.1),
                           n_seasonal = 1,
                           model_string = "wn+matern")
plot(fit_gmwmx)

## End(Not run)
```

`print.gnsstsmodel` *Print method for a* `gnsstsmodel` *object.*

Description

Print method for a `gnsstsmodel` object.

Usage

```
## S3 method for class 'gnsstsmodel'
print(x, ...)
```

Arguments

`x`	A `gnsstsmodel` object.
`...`	Additional graphical parameters.

Value

No return value. Print a `gnsstsmodel` object.

Examples

```
## Not run:
data(cola)
fit_gmwmx = estimate_gmwmx(x = cola,
                           theta_0 = c(0.1,0.1,0.1,0.1),
                           n_seasonal = 1,
                           model_string = "wn+matern")

print(fit_gmwmx)

## End(Not run)
```

read.gnssts *Read a gnssts object*

Description

Read a gnssts object

Usage

```
read.gnssts(filename, format = "mom")
```

Arguments

filename	A string specifying the name of the file to read.
format	A string specifying the format of the file to read.

Value

Return a gnssts object.

Examples

```
phase <- 0.45
amplitude <- 2.5
sigma2_wn <- 15
bias <- 0
trend <- 5 / 365.25
cosU <- amplitude * cos(phase)
sinU <- amplitude * sin(phase)
year <- 5
n <- year * 365
jump_vec <- c(200, 300, 500)
jump_height <- c(10, 15, 20)
nbr_sin <- 1
A <- create_A_matrix(1:n, jump_vec, n_seasonal = nbr_sin)
x_0 <- c(bias, trend, cosU, sinU, jump_height)
eps <- rnorm(n = n, sd = sqrt(sigma2_wn))
yy <- A %*% x_0 + eps
gnssts_obj <- create.gnssts(t = 1:length(yy), y = yy, jumps = jump_vec)
str(gnssts_obj)
## Not run:
write.gnssts(x = gnssts_obj, filename = "test.mom")
gnssts_obj <-read.gnssts(filename = "test.mom", format = "mom")

## End(Not run)
```

`remove_outliers_hector` *Remove outliers from a gnssts object using Hector*

Description

Remove outliers from a gnssts object using Hector

Usage

```
remove_outliers_hector(x, n_seasonal, IQ_factor = 3, cleanup = TRUE)
```

Arguments

x	A gnssts object
n_seasonal	An integer specifying the number of seasonal component in the time series.
IQ_factor	A double specifying the number used to scale the interquartile range and corresponding to the argument IQ_factor in Hector removeoutliers.ctl
cleanup	An boolean specifying if temporary files should be cleaned.

Value

A gnssts object.

Examples

```
phase =     0.45
amplitude = 2.5
sigma2_wn =       15
bias =            0
trend =           5/365.25
cosU =            amplitude*cos(phase)
sinU =            amplitude*sin(phase)
n= 2*365
# define time at which there are jumps
jump_vec =  c(100, 200)
jump_height = c(10, 20)
# generate residuals
eps = rnorm(n = n, sd = sqrt(sigma2_wn))
# add trend, gaps and sin
A = create_A_matrix(1:length(eps), jump_vec, n_seasonal =  1)
# define beta
x_0 = c(bias, trend, jump_height,  cosU,  sinU)
# create time series
yy = A %*% x_0 + eps
plot(yy, type="l")
n_outliers = 30
set.seed(123)
```

```
id_outliers=sample(150:350, size = n_outliers)
val_outliers = rnorm(n = n_outliers, mean = max(yy)+10, sd = 5)
yy[id_outliers] = val_outliers
plot(yy, type="l")
# save signal in temp
gnssts_obj = create.gnssts(t = 1:length(yy), y = yy, jumps = jump_vec)
## Not run:
clean_yy = remove_outliers_hector(x=gnssts_obj, n_seasonal = 1)
plot(clean_yy$t, clean_yy$y, type="l")

## End(Not run)
```

`write.gnssts` *Write a gnssts object*

Description

Write a gnssts object

Usage

```
write.gnssts(x, filename, format = "mom")
```

Arguments

`x`	A R object to save as a `gnssts` object.
`filename`	A `string` specifying the name of the file to write.
`format`	A `string` specifying the format of the file to write.

Value

No return value. Write a `gnssts` object in a .mom file by default.

Examples

```
phase <- 0.45
amplitude <- 2.5
sigma2_wn <- 15
bias <- 0
trend <- 5 / 365.25
cosU <- amplitude * cos(phase)
sinU <- amplitude * sin(phase)
year <- 5
n <- year * 365
jump_vec <- c(200, 300, 500)
jump_height <- c(10, 15, 20)
nbr_sin <- 1
A <- create_A_matrix(1:n, jump_vec, n_seasonal = nbr_sin)
x_0 <- c(bias, trend, cosU, sinU, jump_height)
```

```
eps <- rnorm(n = n, sd = sqrt(sigma2_wn))
yy <- A %*% x_0 + eps
gnssts_obj <- create.gnssts(t = 1:length(yy), y = yy, jumps = jump_vec)
str(gnssts_obj)
## Not run:
write.gnssts(x = gnssts_obj, filename = "test.mom")

## End(Not run)
```

Index

∗ **datasets**
cola, 2

cola, 2
compare_fits, 3
create.gnssts, 4
create_A_matrix, 5

estimate_gmwmx, 5
estimate_hector, 7

PBO_get_offsets, 8
PBO_get_station, 8
plot.gnsstsmodel, 9
print.gnsstsmodel, 10

read.gnssts, 11
remove_outliers_hector, 12

write.gnssts, 13