Package ‘EstemPMM’

November 7, 2025

Type Package

Title Polynomial Maximization Method for Non-Gaussian Regression

Version 0.1.1

Date 2025-10-22

Description Implements the Polynomial Maximization Method ('PMM') for parameter estimation in linear and time series models when error distributions deviate from normality. The 'PMM2' variant achieves lower variance parameter estimates compared to ordinary least squares ('OLS') when errors exhibit significant skewness. Includes methods for linear regression, 'AR'/'MA'/'ARMA'/'ARIMA' models, and bootstrap inference. Methodology described in Zabolotnii, Warsza, and Tkachenko (2018) <doi:10.1007/978-3-319-77179-3_75>,
Zabolotnii, Tkachenko, and Warsza (2022) <doi:10.1007/978-3-031-03502-9_37>, and
Zabolotnii, Tkachenko, and Warsza (2023) <doi:10.1007/978-3-031-25844-2_21>.

License GPL-3

Encoding UTF-8

Depends R (>= 3.5.0)

Imports methods, stats, graphics, utils

Suggests dplyr, ggplot2, gridExtra, testthat (>= 3.0.0), rmarkdown, knitr, MASS

LazyData true

RoxygenNote 7.3.3

Config/testthat/edition 3

URL https://github.com/SZabolotnii/EstemPMM

BugReports https://github.com/SZabolotnii/EstemPMM/issues

VignetteBuilder knitr

NeedsCompilation no

Author Serhii Zabolotnii [aut, cre] (ORCID: <https://orcid.org/0000-0003-0242-2234>)

Maintainer Serhii Zabolotnii <zabolotniua@gmail.com>

Repository CRAN

Date/Publication 2025-11-07 13:50:12 UTC

Contents

AIC,PMM2fit-method *Calculate AIC for PMM2fit object*

Description

Calculate AIC for PMM2fit object

Usage

```
## S4 method for signature 'PMM2fit'
AIC(object, ..., k = 2)
```

Arguments

`object`	PMM2fit object
`...`	Additional arguments (not used)
`k`	Penalty per parameter to be used; default is 2

Value

AIC value

ARIMAPMM2-class *S4 class for storing PMM2 ARIMA model results*

Description

S4 class for storing PMM2 ARIMA model results

arima_pmm2 *Fit an ARIMA model using PMM2 (wrapper)*

Description

Fit an ARIMA model using PMM2 (wrapper)

Usage

```
arima_pmm2(
  x,
  order = c(1, 1, 1),
  method = "pmm2",
  max_iter = 50,
  tol = 1e-06,
  include.mean = TRUE,
  initial = NULL,
  na.action = na.fail,
  regularize = TRUE,
  reg_lambda = 1e-08,
  verbose = FALSE
)
```

Arguments

`x`	Numeric vector of time series data
`order`	Model order specification: - For AR models: a single integer (AR order) - For MA models: a single integer (MA order) - For ARMA models: vector c(p, q) (AR and MA orders) - For ARIMA models: vector c(p, d, q) (AR, differencing, and MA orders)
`method`	String: estimation method, one of "pmm2" (default), "css", "ml", "yw", "ols"
`max_iter`	Integer: maximum number of iterations for the algorithm
`tol`	Numeric: tolerance for convergence
`include.mean`	Logical: whether to include a mean (intercept) term
`initial`	List or vector of initial parameter estimates (optional)
`na.action`	Function for handling missing values, default is na.fail
`regularize`	Logical, add small values to diagonal for numerical stability
`reg_lambda`	Regularization parameter (if regularize=TRUE)
`verbose`	Logical: whether to print progress information

Value

An S4 object of class `ARIMAPMM2` containing fitted AR and MA coefficients, residual series, central moments, differencing order, intercept, original series, and convergence diagnostics.

`ARMAPMM2-class` *S4 class for storing PMM2 ARMA model results*

Description

S4 class for storing PMM2 ARMA model results

arma_pmm2 *Fit an ARMA model using PMM2 (wrapper)*

Description

Fit an ARMA model using PMM2 (wrapper)

Usage

```
arma_pmm2(
  x,
  order = c(1, 1),
  method = "pmm2",
  max_iter = 50,
  tol = 1e-06,
  include.mean = TRUE,
  initial = NULL,
  na.action = na.fail,
  regularize = TRUE,
  reg_lambda = 1e-08,
  verbose = FALSE
)
```

Arguments

`x`	Numeric vector of time series data
`order`	Model order specification: - For AR models: a single integer (AR order) - For MA models: a single integer (MA order) - For ARMA models: vector c(p, q) (AR and MA orders) - For ARIMA models: vector c(p, d, q) (AR, differencing, and MA orders)
`method`	String: estimation method, one of "pmm2" (default), "css", "ml", "yw", "ols"
`max_iter`	Integer: maximum number of iterations for the algorithm
`tol`	Numeric: tolerance for convergence
`include.mean`	Logical: whether to include a mean (intercept) term
`initial`	List or vector of initial parameter estimates (optional)
`na.action`	Function for handling missing values, default is na.fail
`regularize`	Logical, add small values to diagonal for numerical stability
`reg_lambda`	Regularization parameter (if regularize=TRUE)
`verbose`	Logical: whether to print progress information

Value

An S4 object of class `ARMAPMM2` containing fitted AR and MA coefficients, residuals, central moments, model specification, intercept, original series, and convergence diagnostics.

ARPMM2-class *S4 class for storing PMM2 AR model results*

Description

S4 class for storing PMM2 AR model results

ar_pmm2 *Fit an AR model using PMM2 (wrapper)*

Description

Fit an AR model using PMM2 (wrapper)

Usage

```
ar_pmm2(
  x,
  order = 1,
  method = "pmm2",
  max_iter = 50,
  tol = 1e-06,
  include.mean = TRUE,
  initial = NULL,
  na.action = na.fail,
  regularize = TRUE,
  reg_lambda = 1e-08,
  verbose = FALSE
)
```

Arguments

x — Numeric vector of time series data

order — Model order specification: - For AR models: a single integer (AR order) - For MA models: a single integer (MA order) - For ARMA models: vector c(p, q) (AR and MA orders) - For ARIMA models: vector c(p, d, q) (AR, differencing, and MA orders)

method — String: estimation method, one of "pmm2" (default), "css", "ml", "yw", "ols"

max_iter — Integer: maximum number of iterations for the algorithm

tol — Numeric: tolerance for convergence

include.mean — Logical: whether to include a mean (intercept) term

initial — List or vector of initial parameter estimates (optional)

na.action — Function for handling missing values, default is na.fail

`regularize`	Logical, add small values to diagonal for numerical stability
`reg_lambda`	Regularization parameter (if regularize=TRUE)
`verbose`	Logical: whether to print progress information

Value

An S4 object of class `ARPMM2` containing fitted autoregressive coefficients, residuals, central moment estimates (m2-m4), model order, intercept, original series, and convergence diagnostics.

`BasePMM2-class` *Base S4 class for storing PMM2 model results*

Description

Base S4 class for storing PMM2 model results

Slots

`coefficients` numeric vector of estimated parameters

`residuals` numeric vector of final residuals

`m2` numeric second central moment of initial residuals

`m3` numeric third central moment of initial residuals

`m4` numeric fourth central moment of initial residuals

`convergence` logical or integer code indicating whether algorithm converged

`iterations` numeric number of iterations performed

`call` original function call

`coef,PMM2fit-method` *Extract coefficients from PMM2fit object*

Description

Extract coefficients from PMM2fit object

Usage

```
## S4 method for signature 'PMM2fit'
coef(object, ...)
```

Arguments

`object`	PMM2fit object
`...`	Additional arguments (not used)

Value

Vector of coefficients

coef,TS2fit-method *Extract coefficients from TS2fit object*

Description

Extract coefficients from TS2fit object

Usage

```
## S4 method for signature 'TS2fit'
coef(object, ...)
```

Arguments

`object`	TS2fit object
`...`	Additional arguments (not used)

Value

Named vector of coefficients

compare_arima_methods *Compare ARIMA methods*

Description

Compare ARIMA methods

Usage

```
compare_arima_methods(
  x,
  order = c(1, 1, 1),
  include.mean = TRUE,
  pmm2_args = list()
)
```

Arguments

`x`	Numeric vector of time series data
`order`	Model order specification (see ts_pmm2 for format)
`include.mean`	Logical, whether to include intercept term
`pmm2_args`	List of additional arguments to pass to ts_pmm2()

Value

A named list containing the fitted objects for each estimation approach (e.g., YW/OLS/MLE or CSS/ML alongside PMM2) plus two data frames: `coefficients` (side-by-side parameter estimates) and `residual_stats` (residual RSS, MAE, skewness, and kurtosis).

`compare_arma_methods` *Compare ARMA methods*

Description

Compare ARMA methods

Usage

```
compare_arma_methods(
  x,
  order = c(1, 1),
  include.mean = TRUE,
  pmm2_args = list()
)
```

Arguments

`x`	Numeric vector of time series data
`order`	Model order specification (see ts_pmm2 for format)
`include.mean`	Logical, whether to include intercept term
`pmm2_args`	List of additional arguments to pass to ts_pmm2()

Value

A named list containing the fitted objects for each estimation approach (e.g., YW/OLS/MLE or CSS/ML alongside PMM2) plus two data frames: `coefficients` (side-by-side parameter estimates) and `residual_stats` (residual RSS, MAE, skewness, and kurtosis).

`compare_ar_methods` *Compare AR methods*

Description

Compare AR methods

Usage

```
compare_ar_methods(x, order = 1, include.mean = TRUE, pmm2_args = list())
```

Arguments

`x`	Numeric vector of time series data
`order`	Model order specification (see ts_pmm2 for format)
`include.mean`	Logical, whether to include intercept term
`pmm2_args`	List of additional arguments to pass to ts_pmm2()

Value

A named list containing the fitted objects for each estimation approach (e.g., YW/OLS/MLE or CSS/ML alongside PMM2) plus two data frames: `coefficients` (side-by-side parameter estimates) and `residual_stats` (residual RSS, MAE, skewness, and kurtosis).

`compare_ma_methods` *Compare MA methods*

Description

Compare MA methods

Usage

```
compare_ma_methods(x, order = 1, include.mean = TRUE, pmm2_args = list())
```

Arguments

`x`	Numeric vector of time series data
`order`	Model order specification (see ts_pmm2 for format)
`include.mean`	Logical, whether to include intercept term
`pmm2_args`	List of additional arguments to pass to ts_pmm2()

Value

A named list containing the fitted objects for each estimation approach (e.g., YW/OLS/MLE or CSS/ML alongside PMM2) plus two data frames: `coefficients` (side-by-side parameter estimates) and `residual_stats` (residual RSS, MAE, skewness, and kurtosis).

compare_ts_methods *Compare PMM2 with classical time series estimation methods*

Description

Compare PMM2 with classical time series estimation methods

Usage

```
compare_ts_methods(
  x,
  order,
  model_type = c("ar", "ma", "arma", "arima"),
  include.mean = TRUE,
  pmm2_args = list()
)
```

Arguments

`x`	Numeric vector of time series data
`order`	Model order specification (see ts_pmm2 for format)
`model_type`	Model type: "ar", "ma", "arma", or "arima"
`include.mean`	Logical, whether to include intercept term
`pmm2_args`	List of additional arguments to pass to ts_pmm2()

Value

A named list containing the fitted objects for each estimation approach (e.g., YW/OLS/MLE or CSS/ML alongside PMM2) plus two data frames: `coefficients` (side-by-side parameter estimates) and `residual_stats` (residual RSS, MAE, skewness, and kurtosis).

compare_with_ols *Compare PMM2 with OLS*

Description

Compare PMM2 with OLS

Usage

```
compare_with_ols(formula, data, pmm2_args = list())
```

Arguments

`formula`	Model formula
`data`	Data frame
`pmm2_args`	List of arguments to pass to lm_pmm2()

Value

List with OLS and PMM2 fit objects

`compute_moments` *Calculate moments and cumulants of error distribution*

Description

Calculate moments and cumulants of error distribution

Usage

```
compute_moments(errors)
```

Arguments

`errors`	numeric vector of errors

Value

list with moments, cumulants and theoretical variance reduction coefficient

`DCOILWTICO` *WTI Crude Oil Prices*

Description

Daily spot prices for West Texas Intermediate (WTI) crude oil in U.S. dollars per barrel.

Usage

```
DCOILWTICO
```

Format

A data frame with observations for each trading day:

observation_date Date of observation in YYYY-MM-DD format

DCOILWTICO Crude Oil Price: West Texas Intermediate (WTI) in USD per barrel

Source

Federal Reserve Economic Data (FRED), Federal Reserve Bank of St. Louis https://fred.stlouisfed.org/series/DCOILWTICO

Examples

```
data(DCOILWTICO)
head(DCOILWTICO)
summary(DCOILWTICO$DCOILWTICO)
```

`fitted,PMM2fit-method` *Extract fitted values from PMM2fit object*

Description

Extract fitted values from PMM2fit object

Usage

```
## S4 method for signature 'PMM2fit'
fitted(object, data = NULL, ...)
```

Arguments

`object`	PMM2fit object
`data`	Optional data source for model reconstruction, if object does not contain saved data
`...`	Additional arguments (not used)

Value

Vector of fitted values

`fitted,TS2fit-method` *Extract fitted values from TS2fit object*

Description

Extract fitted values from TS2fit object

Usage

```
## S4 method for signature 'TS2fit'
fitted(object, ...)
```

Arguments

`object`	TS2fit object
`...`	Additional arguments (not used)

Value

Vector of fitted values

`lm_pmm2` *PMM2: Main function for PMM2 (S=2)*

Description

Fits a linear model using the Polynomial Maximization Method (order 2), which is robust to non-Gaussian errors.

Usage

```
lm_pmm2(
  formula,
  data,
  max_iter = 50,
  tol = 1e-06,
  regularize = TRUE,
  reg_lambda = 1e-08,
  na.action = na.fail,
  weights = NULL,
  verbose = FALSE
)
```

Arguments

`formula`	R formula for the model
`data`	data.frame containing variables in the formula
`max_iter`	integer: maximum number of iterations for the algorithm
`tol`	numeric: tolerance for convergence
`regularize`	logical: add small value to diagonal for numerical stability
`reg_lambda`	numeric: regularization parameter (if regularize=TRUE)
`na.action`	function for handling missing values, default is na.fail
`weights`	optional weight vector (not yet implemented)
`verbose`	logical: whether to print progress information

Details

The PMM2 algorithm works as follows:

1. Fits ordinary least squares (OLS) regression to obtain initial estimates
2. Computes central moments (m2, m3, m4) from OLS residuals
3. Iteratively improves parameter estimates using a gradient-based approach

PMM2 is especially useful when error terms are not Gaussian.

Value

S4 object of class `PMM2fit`

Examples

```
set.seed(123)
n <- 80
x <- rnorm(n)
y <- 2 + 3 * x + rt(n, df = 3)
dat <- data.frame(y = y, x = x)

fit <- lm_pmm2(y ~ x, data = dat)
summary(fit, formula = y ~ x, data = dat)
```

`MAPMM2-class` *S4 class for storing PMM2 MA model results*

Description

S4 class for storing PMM2 MA model results

`ma_pmm2` *Fit an MA model using PMM2 (wrapper)*

Description

Fit an MA model using PMM2 (wrapper)

Usage

```
ma_pmm2(
  x,
  order = 1,
  method = "pmm2",
  max_iter = 50,
  tol = 1e-06,
  include.mean = TRUE,
  initial = NULL,
  na.action = na.fail,
  regularize = TRUE,
  reg_lambda = 1e-08,
  verbose = FALSE
)
```

Arguments

`x`	Numeric vector of time series data
`order`	Model order specification: - For AR models: a single integer (AR order) - For MA models: a single integer (MA order) - For ARMA models: vector c(p, q) (AR and MA orders) - For ARIMA models: vector c(p, d, q) (AR, differencing, and MA orders)
`method`	String: estimation method, one of "pmm2" (default), "css", "ml", "yw", "ols"
`max_iter`	Integer: maximum number of iterations for the algorithm
`tol`	Numeric: tolerance for convergence
`include.mean`	Logical: whether to include a mean (intercept) term
`initial`	List or vector of initial parameter estimates (optional)
`na.action`	Function for handling missing values, default is na.fail
`regularize`	Logical, add small values to diagonal for numerical stability
`reg_lambda`	Regularization parameter (if regularize=TRUE)
`verbose`	Logical: whether to print progress information

Value

An S4 object of class `MAPMM2` containing moving-average coefficients, residual innovations, central moments, model order, intercept, original series, and convergence diagnostics.

`plot,PMM2fit,missing-method`
Plot diagnostic plots for PMM2fit object

Description

Plot diagnostic plots for PMM2fit object

Usage

```
## S4 method for signature 'PMM2fit,missing'
plot(x, y, which = 1:4, ...)
```

Arguments

x	PMM2fit object
y	Not used (compatibility with generic)
which	Set of plots to display (values 1-4)
...	Additional arguments passed to plotting functions

Value

Invisibly returns the input object

`plot,TS2fit,missing-method`

Build diagnostic plots for TS2fit objects

Description

Build diagnostic plots for TS2fit objects

Usage

```
## S4 method for signature 'TS2fit,missing'
plot(x, y, which = c(1:4), ...)
```

Arguments

x	TS2fit object
y	Not used (for S4 method compatibility)
which	Integer vector indicating which plots to produce
...	additional arguments passed to plot functions

Value

Invisibly returns x

plot_pmm2_bootstrap *Plot bootstrap distributions for PMM2 fit*

Description

Plot bootstrap distributions for PMM2 fit

Usage

```
plot_pmm2_bootstrap(object, coefficients = NULL)
```

Arguments

`object`	Result from pmm2_inference
`coefficients`	Which coefficients to plot, defaults to all

Value

Invisibly returns histogram information

PMM2fit-class *S4 class for storing PMM2 regression model results*

Description

Class for storing results of linear model estimation using PMM2

Slots

`coefficients` numeric vector of estimated parameters

`residuals` numeric vector of final residuals

`m2` numeric second central moment of initial residuals

`m3` numeric third central moment of initial residuals

`m4` numeric fourth central moment of initial residuals

`convergence` logical or integer code indicating whether algorithm converged

`iterations` numeric number of iterations performed

`call` original function call

Slots

coefficients Estimated coefficients

residuals Final residuals

m2 Second central moment

m3 Third central moment

m4 Fourth central moment

convergence Convergence status

iterations Number of iterations performed

call Original call

---

pmm2_inference *Bootstrap inference for PMM2 fit*

Description

Bootstrap inference for PMM2 fit

Usage

```
pmm2_inference(
  object,
  formula,
  data,
  B = 200,
  seed = NULL,
  parallel = FALSE,
  cores = NULL
)
```

Arguments

object	object of class PMM2fit
formula	the same formula that was used initially
data	data frame that was used initially
B	number of bootstrap replications
seed	(optional) for reproducibility
parallel	logical, whether to use parallel computing
cores	number of cores to use for parallel computing, defaults to auto-detect

Value

data.frame with columns: Estimate, Std.Error, t.value, p.value

pmm2_monte_carlo_compare

Monte Carlo comparison of PMM2 estimation methods

Description

Function generates time series for given models, repeatedly estimates parameters using different methods and compares their accuracy by MSE criterion. Additionally outputs theoretical and empirical characteristics of the innovation distribution (skewness, excess kurtosis, theoretical gain of PMM2).

Usage

```
pmm2_monte_carlo_compare(
  model_specs,
  methods = c("css", "pmm2"),
  n,
  n_sim,
  innovations = list(type = "gaussian"),
  seed = NULL,
  include.mean = TRUE,
  progress = interactive(),
  verbose = FALSE
)
```

Arguments

`model_specs` List of model specifications. Each element must contain:

- **model** "ar", "ma" or "arma"
- **order** order (for AR/MA) or vector c(p, q) for ARMA
- **theta** numeric vector of true parameters; for ARMA a list `list(ar = ..., ma = ...)`
- **label** (optional) model name in report
- **innovations** (optional) description of innovation distribution: `list(type = "gamma", shape = 2)`, `list(type = "student_t", df = 5)`, etc. Can also pass an arbitrary generation function via `generator`.

`methods` Vector of estimation methods (e.g., `c("css","pmm2")`). The first method is considered baseline for relative MSE calculation.

`n` Sample size for simulation.

`n_sim` Number of Monte Carlo experiments.

`innovations` Function or distribution description, used by default for all models (if not specified in spec).

`seed` Initial seed for random number generator (optional).

`include.mean` Logical flag: whether to include intercept during estimation.

`progress` Logical flag: print Monte Carlo progress.

`verbose` Whether to print diagnostic messages on failures.

Value

List with three components:

parameter_results MSE and relative MSE for each parameter

summary Averaged MSE over parameters for each model/method

gain Comparison of theoretical and empirical PMM2 gain

`pmm2_variance_factor` *Calculate theoretical skewness, kurtosis coefficients and variance reduction factor*

Description

Calculate theoretical skewness, kurtosis coefficients and variance reduction factor

Usage

```
pmm2_variance_factor(m2, m3, m4)
```

Arguments

`m2`, `m3`, `m4`	central moments of second, third and fourth orders

Value

List with fields `c3`, `c4` and `g`

`pmm2_variance_matrices` *Calculate theoretical variance matrices for OLS and PMM2*

Description

Calculate theoretical variance matrices for OLS and PMM2

Usage

```
pmm2_variance_matrices(X, m2, m3, m4)
```

Arguments

`X`	Design matrix with column of ones
`m2`, `m3`, `m4`	central moments of OLS residuals

Value

List with fields `ols`, `pmm2`, `c3`, `c4`, `g`

`pmm_kurtosis` *Calculate kurtosis from data*

Description

Calculate kurtosis from data

Usage

```
pmm_kurtosis(x, excess = TRUE)
```

Arguments

`x` numeric vector

`excess` logical, whether to return excess kurtosis (kurtosis - 3)

Value

Kurtosis value

`pmm_skewness` *Calculate skewness from data*

Description

Calculate skewness from data

Usage

```
pmm_skewness(x)
```

Arguments

`x` numeric vector

Value

Skewness value

`predict,PMM2fit-method` *Prediction method for PMM2fit objects*

Description

Prediction method for PMM2fit objects

Usage

```
## S4 method for signature 'PMM2fit'
predict(object, newdata = NULL, debug = FALSE, ...)
```

Arguments

`object`	PMM2fit object
`newdata`	New data frame for prediction
`debug`	Logical value, whether to output debug information
`...`	additional arguments (not used)

Value

Vector of predictions

`predict,TS2fit-method` *Prediction method for TS2fit objects*

Description

Prediction method for TS2fit objects

Usage

```
## S4 method for signature 'TS2fit'
predict(object, n.ahead = 1, ...)
```

Arguments

`object`	TS2fit object
`n.ahead`	Number of steps ahead for prediction
`...`	additional arguments (not used)

Value

Vector or list of predictions, depending on model type

residuals,PMM2fit-method

Extract residuals from PMM2fit object

Description

Extract residuals from PMM2fit object

Usage

```
## S4 method for signature 'PMM2fit'
residuals(object, ...)
```

Arguments

object	PMM2fit object
...	Additional arguments (not used)

Value

Vector of residuals

residuals,TS2fit-method

Extract residuals from TS2fit object

Description

Extract residuals from TS2fit object

Usage

```
## S4 method for signature 'TS2fit'
residuals(object, ...)
```

Arguments

object	TS2fit object
...	Additional arguments (not used)

Value

Vector of residuals (innovations)

`summary,PMM2fit-method` *Generic summary method for PMM2fit objects*

Description

Generic summary method for PMM2fit objects

Usage

```
## S4 method for signature 'PMM2fit'
summary(object, formula = NULL, data = NULL, B = 100, ...)
```

Arguments

`object`	object of class "PMM2fit"
`formula`	(optional) formula used for the model
`data`	(optional) data used
`B`	number of bootstrap replications for statistical inference
`...`	additional arguments (not used)

Value

Prints summary to console; returns object (invisibly).

`summary,TS2fit-method` *Generic summary method for TS2fit objects*

Description

Generic summary method for TS2fit objects

Usage

```
## S4 method for signature 'TS2fit'
summary(object, ...)
```

Arguments

`object`	object of class "TS2fit" or subclass
`...`	additional arguments (not used)

Value

Prints summary to console; returns object (invisibly).

TS2fit-class *Base S4 class for storing PMM2 time series model results*

Description

Base class for storing results of time series model estimation using PMM2

Slots

`coefficients` numeric vector of estimated parameters

`residuals` numeric vector of final residuals

`m2` numeric second central moment of initial residuals

`m3` numeric third central moment of initial residuals

`m4` numeric fourth central moment of initial residuals

`convergence` logical or integer code indicating whether algorithm converged

`iterations` numeric number of iterations performed

`call` original function call

`model_type` character string indicating model type

`intercept` numeric value of intercept

`original_series` numeric vector of original time series

`order` list of order parameters

Slots

coefficients Estimated coefficients

residuals Final residuals

m2 Second central moment

m3 Third central moment

m4 Fourth central moment

convergence Convergence status

iterations Number of iterations performed

call Original call

model_type Model type

intercept Intercept

original_series Original time series

order Model orders

ts_pmm2 *Fit a time series model using the PMM2 method*

Description

Fit a time series model using the PMM2 method

Usage

```
ts_pmm2(
  x,
  order,
  model_type = c("ar", "ma", "arma", "arima"),
  method = "pmm2",
  max_iter = 50,
  tol = 1e-06,
  include.mean = TRUE,
  initial = NULL,
  na.action = na.fail,
  regularize = TRUE,
  reg_lambda = 1e-08,
  verbose = FALSE
)
```

Arguments

`x`	Numeric vector of time series data
`order`	Model order specification: - For AR models: a single integer (AR order) - For MA models: a single integer (MA order) - For ARMA models: vector c(p, q) (AR and MA orders) - For ARIMA models: vector c(p, d, q) (AR, differencing, and MA orders)
`model_type`	String specifying the model type: "ar", "ma", "arma", or "arima"
`method`	String: estimation method, one of "pmm2" (default), "css", "ml", "yw", "ols"
`max_iter`	Integer: maximum number of iterations for the algorithm
`tol`	Numeric: tolerance for convergence
`include.mean`	Logical: whether to include a mean (intercept) term
`initial`	List or vector of initial parameter estimates (optional)
`na.action`	Function for handling missing values, default is na.fail
`regularize`	Logical, add small values to diagonal for numerical stability
`reg_lambda`	Regularization parameter (if regularize=TRUE)
`verbose`	Logical: whether to print progress information

Details

The PMM2 algorithm works as follows:

1. Fits an initial model using a standard method (OLS, Yule-Walker, CSS or ML)
2. Computes central moments (m2, m3, m4) from initial residuals/innovations
3. Uses these moments with a specialized solver (pmm2_algorithm) to find robust parameter estimates

Value

An S4 object `TS2fit` of the corresponding subclass

`ts_pmm2_inference` *Bootstrap inference for PMM2 time series models*

Description

Bootstrap inference for PMM2 time series models

Usage

```
ts_pmm2_inference(
  object,
  x = NULL,
  B = 200,
  seed = NULL,
  block_length = NULL,
  method = c("residual", "block"),
  parallel = FALSE,
  cores = NULL,
  debug = FALSE
)
```

Arguments

`object`	object of class TS2fit
`x`	(optional) original time series; if NULL, uses object@original_series
`B`	number of bootstrap replications
`seed`	(optional) for reproducibility
`block_length`	block length for block bootstrap; if NULL, uses heuristic value
`method`	bootstrap type: "residual" or "block"
`parallel`	logical, whether to use parallel computing
`cores`	number of cores for parallel computing
`debug`	logical, whether to output additional diagnostic information

Value

data.frame with columns: Estimate, Std.Error, t.value, p.value

Index