

Package: MStest (via r-universe)

October 22, 2026

Type Package

Title Hypothesis Testing for Markov Switching Models

Date 2026-09-14

Version 0.1.9.9015

Maintainer Gabriel Rodriguez-Rondon <gabrodriguezrondon@gmail.com>

Description Implementation of hypothesis testing procedures described in Hansen (1992) <[doi:10.1002/jae.3950070506](https://doi.org/10.1002/jae.3950070506)>, Carrasco, Hu, & Ploberger (2014) <[doi:10.3982/ECTA8609](https://doi.org/10.3982/ECTA8609)>, Dufour & Luger (2017) <[doi:10.1080/07474938.2017.1307548](https://doi.org/10.1080/07474938.2017.1307548)>, and Rodriguez-Rondon & Dufour (2026a) <[doi:10.34989/swp-2026-23](https://doi.org/10.34989/swp-2026-23)> that can be used to identify the number of regimes in Markov switching models. For a detailed description of the package, see Rodriguez-Rondon & Dufour (2026b) <[doi:10.34989/swp-2026-7](https://doi.org/10.34989/swp-2026-7)>.

License GPL(>= 2)

LazyData TRUE

RoxygenNote 7.3.3

Encoding UTF-8

URL <https://github.com/roga11/MStest>

BugReports <https://github.com/roga11/MStest/issues>

Imports stats, rlang, nloptr, Rcpp (>= 1.0.1), numDeriv, pracma, GenSA, pso, GA, graphics

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 4.0.0)

Suggests testthat (>= 3.0.0), callr, knitr, rmarkdown

VignetteBuilder knitr

Config/testthat/edition 3

SystemRequirements C++17

Config/pak/sysreqs cmake

Repository <https://roga11.r-universe.dev>

Date/Publication 2026-09-14 18:32:28 UTC

RemoteUrl <https://github.com/roga11/mstest>

RemoteRef HEAD

RemoteSha 030123697a62022f6036cd4ec7fbfdc95155a421

Contents

MSTest-package	3
aBfixLMCTest	4
ARmdl	5
ARXmdl	7
aURdiffLMCTest	10
aURdiffMMCTest	11
aURfixLMCTest	12
canonicalize_regimes	14
chp10GNP	15
CHPbootCV	15
CHPTTest	16
DLMCTest	18
DLMMCTest	19
hamilton84GNP	22
HLRTest	23
HMmdl	25
LMCLRTTest	30
MCpval	33
MMCLRTTest	34
MSARmdl	41
MSARXmdl	47
MSVARmdl	52
MSVARXmdl	57
Nmdl	63
simuAR	65
simuARX	66
simuHMM	67
simuMSAR	68
simuMSARX	69
simuMSVAR	70
simuMSVARX	71
simuNorm	73
simuVAR	74
simuVARX	75
USGNP	76
USRGDP	77
VARmdl	77
VARXmdl	79
warmstart_theta	82

Description

This package implements hypothesis testing procedures that can be used to identify the number of regimes in a Markov-Switching model.

Author(s)

Gabriel Rodriguez-Rondon, gabrodriguezrondon@gmail.com (Maintainer)

Jean-Marie Dufour, jean-marie.dufour@mcgill.ca

References

- Carrasco, Marine, Liang Hu, and Werner Ploberger. 2014. "Optimal test for Markov switching parameters." *Econometrica* 82 (2): 765–784.
- Dempster, A. P., N. M. Laird, and D. B. Rubin. 1977. "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society. Series B* 39 (1): 1–38.
- Dufour, Jean-Marie, and Richard Luger. 2017. "Identification-robust moment-based tests for Markov switching in autoregressive models." *Econometric Reviews* 36 (6-9): 713–727.
- Kasahara, Hiroyuk, and Katsum Shimotsu. 2018. "Testing the number of regimes in Markov regime switching models." arXiv preprint arXiv:1801.06862.
- Krolzig, Hans-Martin. 1997. "The markov-switching vector autoregressive model.". *Springer*.
- Hamilton, James D. 1989. "A new approach to the economic analysis of nonstationary time series and the business cycle." *Econometrica* 57 (2): 357–384.
- Hamilton, James D. 1994. "Time series analysis". *Princeton university press*.
- Hansen, Bruce E. 1992. "The likelihood ratio test under nonstandard conditions: testing the Markov switching model of GNP." *Journal of applied Econometrics* 7 (S1): S61–S82.
- Rodriguez-Rondon, G., & Dufour, J.-M. 2026a. "Monte Carlo Likelihood-Ratio Tests for Markov Switching Models." *Bank of Canada Staff Working Paper*, No. 2026-23. doi: 10.34989/swp-2026-23.
- Rodriguez-Rondon, G., & Dufour, J.-M. 2026b. "MSTest: An R-Package for Testing Markov Switching Models." *Bank of Canada Staff Working Paper*, No. 2026-7. doi: 10.34989/swp-2026-7.
- Qu, Zhongjun, and Fan Zhuo. 2021. "Likelihood Ratio-Based Tests for Markov Regime Switching." *The Review of Economic Studies* 88 (2): 937–968.

aBfixLMCTest

*Boundary-augmented (fix) Local Monte Carlo Likelihood Ratio Test***Description**

Runs the Local Monte Carlo likelihood ratio test of Rodriguez-Rondon & Dufour (2026a) twice on the same observed series and the same pre-drawn simulation draws, for a $k_0 \geq 2$ null in the boundary (possibly absorbing regime) setting: once at the fitted null-model parameters, and once at their boundary projection – the absorbing regime’s self-transition probability set to exactly 1 (its column’s remaining entries to 0), everything else at its estimate. The regimes are first relabeled canonically ([canonicalize_regimes](#): transient regimes first, absorbing regime last), and every simulated sample starts with the chain initialized in the transient regime with no burn-in (the change-point-like family of Chib, 1998) – which is why `burnin = 0` is required. The test rejects only if BOTH component p-values are below the chosen level (reported as `pval = max(pval_lmc, pval_fix)`). At $p = 0$ the first observation’s regime is a draw from the transient regime’s transition distribution (the hidden Markov simulator applies a transition before the first emission). Multivariate models are supported: the boundary projection edits only the absorbing regime’s transition-matrix column, so the fixed simulation parameter keeps the fitted regime covariances – estimates of an admissible model, positive definite by construction.

Usage

```
aBfixLMCTest(Y, p, k0, k1, Z = NULL, control = list())
```

Arguments

Y	Series to be tested. Must be a (T x q) matrix.
p	Number of autoregressive lags. Must be greater than or equal to 0.
k0	Number of regimes under null hypothesis. Must be greater than or equal to 2. With $k_0 \geq 3$ the boundary family’s initial state is canonical state 1 of the fitted null (the least persistent transient regime at the fit, inherited by index for the simulation parameter) – a fixed-vertex family definition on which the finite-sample validity is conditional.
k1	Number of regimes under alternative hypothesis. Must be greater than k_0 .
Z	Optional (T x qz) matrix of fixed (deterministic) or strictly exogenous regressors, or NULL. Dummy variables are deterministic, hence fixed regressors, and are fully supported. The regressors’ common coefficients never switch and are identified by the whole sample, so both components treat them like the other common parameters.
control	List with test procedure options as in aURfixLMCTest , except that <code>burnin</code> defaults to 0 and must be 0. <code>boundary_R</code> is accepted for interface symmetry with MMCLRTTest ’s boundary modes but is not used here (the second component is a single fixed evaluation, not a search).

Value

List of class `aBfixLMCTest` with the same structure as `aURfixLMCTest` (component `p-values` `pval_1mc`, `pval_fix`, composition `pval`, the two null distributions, and `theta_fix` – here the boundary projection). `mdl_h0` is returned in canonical regime order.

References

Rodriguez-Rondon, G., & Dufour, J.-M. 2026a. "Monte Carlo Likelihood-Ratio Tests for Markov Switching Models." *Bank of Canada Staff Working Paper*, No. 2026-23. doi: 10.34989/swp-2026-23.

Chib, S. 1998. "Estimation and Comparison of Multiple Change-Point Models." *Journal of Econometrics* 86(2): 221-241.

ARmdl	<i>Autoregressive Model</i>
-------	-----------------------------

Description

This function estimates an autoregressive model with `p` lags. This can be used for the null hypothesis of a linear model against an alternative hypothesis of a Markov switching autoregressive model with `k` regimes.

Usage

```
ARmdl(Y, p, control = list())
```

Arguments

<code>Y</code>	A ($T \times 1$) matrix of observations.
<code>p</code>	Integer determining the number of autoregressive lags.
<code>control</code>	List with model options including: • <code>const</code>: Boolean determining whether to estimate model with constant if TRUE or not if FALSE. Default is TRUE. • <code>getSE</code>: Boolean determining whether to compute standard errors of parameters if TRUE or not if FALSE. Default is TRUE. • <code>stationarity_constraint</code>: Double giving a stationarity margin <code>delta</code> on the autoregressive coefficients: the fitted process satisfies companion spectral radius $\leq 1 - \delta$. 0 (the default) or a negative value disables the constraint and reproduces the ordinary least-squares fit, including a nonstationary one. At $p = 1$ the constrained fit is exact – the clip of the unconstrained coefficient, with the remaining coefficients re-solved exactly at the constrained value; for $p \geq 2$ the autoregressive block is moved by a root-scaling projection of the offending companion eigenvalues (a projection, not the constrained least-squares optimizer). Intended for comparability with procedures whose validity theory assumes stationarity, such as parametric bootstrap tests (Basawa, Mallik, McCormick, Reeves and Taylor,

1991); the Monte Carlo likelihood-ratio tests themselves do not require it. Distinct from MMCLRTTest's `stationary_constraint` (the search's stationarity penalty); by default MMCLRTTest aligns its nuisance search set with the margin (control `box_match_constraints`). A binding fit is flagged in `phi_constraint_binding` and the autoregressive block's standard errors are set to NA (the constraint couples the whole block). Markov-switching estimation controls forwarded by the testing functions (`use_diff_init`, `em_variance_constraint`, `em_transition_constraint`) are accepted and ignored without a warning.

Value

List of class `ARmdl` (S3 object) with model attributes including:

- `y`: a $(T-p \times 1)$ matrix of observations.
- `X`: a $(T-p \times p + \text{const})$ matrix of lagged observations with a leading column of 1s if `const=TRUE` or not if `const=FALSE`.
- `x`: a $(T-p \times p)$ matrix of lagged observations.
- `fitted`: a $(T-p \times 1)$ matrix of fitted values.
- `resid`: a $(T-p \times 1)$ matrix of residuals.
- `mu`: estimated mean of the process.
- `beta`: a $((1 + p) \times 1)$ matrix of estimated coefficients.
- `intercept`: estimate of intercept.
- `phi`: estimates of autoregressive coefficients.
- `stdev`: estimated standard deviation of the process.
- `sigma`: estimated variance of the process.
- `theta`: vector containing: `mu`, `sigma`, and `phi`.
- `theta_mu_ind`: vector indicating location of mean with 1 and 0 otherwise.
- `theta_sig_ind`: vector indicating location of variance with 1 and 0 otherwise.
- `theta_var_ind`: vector indicating location of variance with 1 and 0 otherwise. This is the same as `theta_sig_ind` in `ARmdl`.
- `theta_phi_ind`: vector indicating location of autoregressive coefficients with 1 and 0 otherwise.
- `stationary`: Boolean indicating if process is stationary if `TRUE` or non-stationary if `FALSE`.
- `n`: number of observations after lag transformation (i.e., $n = T-p$).
- `p`: number of autoregressive lags.
- `q`: number of series. This is always 1 in `ARmdl`.
- `k`: number of regimes. This is always 1 in `ARmdl`.
- `control`: List with model options used.
- `logLike`: log-likelihood.
- `AIC`: Akaike information criterion.
- `BIC`: Bayesian (Schwarz) information criterion.

- Hess: Hessian matrix. Approximated using [hessian](#) and only returned if getSE=TRUE.
- info_mat: Information matrix. Computed as the inverse of -Hess. If matrix is not PD then nearest PD matrix is obtained using [nearest_spd](#). Only returned if getSE=TRUE.
- nearPD_used: Boolean determining whether nearPD function was used on info_mat if TRUE or not if FALSE. Only returned if getSE=TRUE.
- theta_se: standard errors of parameters in theta. Only returned if getSE=TRUE.

References

Basawa, I. V., Mallik, A. K., McCormick, W. P., Reeves, J. H., & Taylor, R. L. 1991. "Bootstrapping Unstable First-Order Autoregressive Processes." *The Annals of Statistics* 19(2): 1098-1101.

See Also

[MSARmdl](#)

Examples

```
set.seed(1234)
# Define DGP of AR process
mdl_ar <- list(n      = 500,
              mu      = 5,
              sigma    = 2,
              phi      = c(0.5, 0.2))

# Simulate process using simuAR() function
y_simu <- simuAR(mdl_ar)

# Set options for model estimation
control <- list(const = TRUE,
                getSE  = TRUE)

# Estimate model
y_ar_mdl <- ARmdl(y_simu$y, p = 2, control)
y_ar_mdl
```

ARXmdl

Autoregressive X Model

Description

This function estimates an ARX model with p lags. This can be used for the null hypothesis of a linear model against an alternative hypothesis of a Markov switching autoregressive model with k regimes.

Usage

```
ARXmdl(Y, p, Z, control = list())
```

Arguments

Y	A (T x 1) matrix of observations.
p	Integer determining the number of autoregressive lags.
Z	A (T x qz) matrix of exogenous regressors.
control	List with model options including: • <code>const</code>: Boolean determining whether to estimate model with constant if TRUE or not if FALSE. Default is TRUE. • <code>getSE</code>: Boolean determining whether to compute standard errors of parameters if TRUE or not if FALSE. Default is TRUE. • <code>stationarity_constraint</code>: Double giving a stationarity margin δ on the autoregressive coefficients: the fitted process satisfies companion spectral radius $\leq 1 - \delta$. 0 (the default) or a negative value disables the constraint and reproduces the ordinary least-squares fit, including a nonstationary one. At $p = 1$ the constrained fit is exact – the clip of the unconstrained coefficient, with the remaining coefficients re-solved exactly at the constrained value; for $p \geq 2$ the autoregressive block is moved by a root-scaling projection of the offending companion eigenvalues (a projection, not the constrained least-squares optimizer). Intended for comparability with procedures whose validity theory assumes stationarity, such as parametric bootstrap tests (Basawa, Mallik, McCormick, Reeves and Taylor, 1991); the Monte Carlo likelihood-ratio tests themselves do not require it. Distinct from <code>MMCLRTTest</code>'s <code>stationarity_constraint</code> (the search's stationarity penalty); by default <code>MMCLRTTest</code> aligns its nuisance search set with the margin (<code>control box_match_constraints</code>). A binding fit is flagged in <code>phi_constraint_binding</code> and the autoregressive block's standard errors are set to NA (the constraint couples the whole block). Markov-switching estimation controls forwarded by the testing functions (<code>use_diff_init</code>, <code>em_variance_constraint</code>, <code>em_transition_constraint</code>) are accepted and ignored without a warning.

Value

List of class `ARmdl` (S3 object) with model attributes including:

- `y`: a (T-p x 1) matrix of observations.
- `X`: a (T-p x p + const) matrix of lagged observations with a leading column of 1s if `const=TRUE` or not if `const=FALSE`.
- `x`: a (T-p x p) matrix of lagged observations.
- `fitted`: a (T-p x 1) matrix of fitted values.
- `resid`: a (T-p x 1) matrix of residuals.
- `mu`: estimated mean of the process.
- `beta`: a ((1 + p + qz) x 1) matrix of estimated coefficients.
- `betaZ`: a (qz x q) matrix of estimated exogenous regressor coefficients.
- `intercept`: estimate of intercept.

- phi: estimates of autoregressive coefficients.
- stdev: estimated standard deviation of the process.
- sigma: estimated variance of the process.
- theta: vector containing: mu, sigma, and phi.
- theta_mu_ind: vector indicating location of mean with 1 and 0 otherwise.
- theta_sig_ind: vector indicating location of variance with 1 and 0 otherwise.
- theta_var_ind: vector indicating location of variance with 1 and 0 otherwise. This is the same as theta_sig_ind in ARmdl.
- theta_phi_ind: vector indicating location of autoregressive coefficients with 1 and 0 otherwise.
- stationary: Boolean indicating if process is stationary if TRUE or non-stationary if FALSE.
- n: number of observations after lag transformation (i.e., $n = T - p$).
- p: number of autoregressive lags.
- q: number of series. This is always 1 in ARmdl.
- k: number of regimes. This is always 1 in ARmdl.
- control: List with model options used.
- logLike: log-likelihood.
- AIC: Akaike information criterion.
- BIC: Bayesian (Schwarz) information criterion.
- Hess: Hessian matrix. Approximated using [hessian](#) and only returned if getSE=TRUE.
- info_mat: Information matrix. Computed as the inverse of -Hess. If matrix is not PD then nearest PD matrix is obtained using [nearest_spd](#). Only returned if getSE=TRUE.
- nearPD_used: Boolean determining whether nearPD function was used on info_mat if TRUE or not if FALSE. Only returned if getSE=TRUE.
- theta_se: standard errors of parameters in theta. Only returned if getSE=TRUE.

References

Basawa, I. V., Mallik, A. K., McCormick, W. P., Reeves, J. H., & Taylor, R. L. 1991. "Bootstrapping Unstable First-Order Autoregressive Processes." *The Annals of Statistics* 19(2): 1098-1101.

See Also

[MSARmdl](#)

Examples

```
set.seed(1234)
# Define DGP of AR process
mdl_ar <- list(n      = 500,
              mu      = 5,
              sigma    = 2,
              phi      = c(0.5, 0.2))

# Simulate process using simuAR() function
```

```

y_simu <- simuAR mdl_ar)

# Set options for model estimation
control <- list(const = TRUE,
               getSE = TRUE)

# Estimate model
y_ar_mdl <- ARmdl(y_simu$y, p = 2, control)
y_ar_mdl

```

aURdiffLMCTest	<i>Unit-root-augmented (differencing) Local Monte Carlo Likelihood Ratio Test</i>
----------------	---

Description

Runs the Local Monte Carlo likelihood ratio test of Rodriguez-Rondon & Dufour (2026a) twice: on the observed series in levels with p autoregressive lags, and on its first difference with $p - 1$ lags (at $p = 1$ the differenced null has no lagged term). The test rejects only if BOTH component p -values are below the chosen level (an intersection-union composition, reported as $pval = \max(pval_levels, pval_diff)$). Differencing imposes the null unit root without estimating it: at a one-regime null with $p = 1$ the differenced component's statistic does not depend on any estimated autoregressive parameter, and its unit-root guarantee is exact in finite samples – including with exogenous regressors present. For $p \geq 2$ the differenced model keeps lagged differences, and the component is asymptotically valid at the unit root rather than exact. Exogenous regressors are kept IN LEVELS for the differenced component (rows 2 to T of Z) – they enter the observation equation additively, outside the mean adjustment, so differencing the series does not difference them. Both components are full [LMCLRTTest](#) runs sharing the same control (including `mc_seed`, so their simulation streams start from the same seed position); each holds its own pre-drawn innovations fixed internally. Multivariate series run the same way – the difference is taken per series and both components are ordinary multivariate tests – but the unit-root validity argument is a univariate construction: a multivariate non-stationary null spans cointegration ranks 0 to $q - 1$, and first-differencing all series imposes the no-cointegration case, so at $q > 1$ the composition's unit-root guarantee is conditional on that maintained assumption.

Usage

```
aURdiffLMCTest(Y, p, k0, k1, Z = NULL, control = list())
```

Arguments

<code>Y</code>	Series to be tested. Must be a $(T \times q)$ matrix.
<code>p</code>	Number of autoregressive lags for the levels component. Must be greater than or equal to 1.
<code>k0</code>	Number of regimes under null hypothesis. Must be greater than or equal to 1.
<code>k1</code>	Number of regimes under alternative hypothesis. Must be greater than <code>k0</code> .

Z	Optional ($T \times q_z$) matrix of fixed (deterministic) or strictly exogenous regressors, or NULL. Dummy variables are deterministic, hence fixed regressors, and are fully supported. The regressors' common coefficients never switch and are identified by the whole sample, so both components treat them like the other common parameters.
control	List with test procedure options, passed to BOTH component LMCLRTTest calls unchanged.

Value

List of class `aURdiffLMCTest` with `pval_levels`, `pval_diff` (the component p-values), `pval` (their maximum; reject only if it is below the level), `lmc_levels` and `lmc_diff` (the two full `LMCLRTTest` objects), and `control`.

References

Rodriguez-Rondon, G., & Dufour, J.-M. 2026a. "Monte Carlo Likelihood-Ratio Tests for Markov Switching Models." *Bank of Canada Staff Working Paper*, No. 2026-23. doi: 10.34989/swp-2026-23.

aURdiffMMCTest	<i>Unit-root-augmented (differencing) Maximized Monte Carlo Likelihood Ratio Test</i>
----------------	---

Description

Runs the Maximized Monte Carlo likelihood ratio test of Rodriguez-Rondon & Dufour (2026a) twice: on the observed series in levels with p autoregressive lags (typically with the unit-root search box of [MMCLRTTest](#)'s `box_include` control, so the search set contains the unit root), and on its first difference with $p - 1$ lags, each with its own consistent-set search box around its own fitted null model. The test rejects only if BOTH component p-values are below the chosen level (reported as `pval = max(pval_levels, pval_diff)`). Multivariate series are handled as in [aURdiffLMCTest](#), with the same scope note: first-differencing all series imposes the no-cointegration case, so at $q > 1$ the unit-root guarantee is conditional on that maintained assumption.

Usage

```
aURdiffMMCTest(Y, p, k0, k1, Z = NULL, control = list(), control_diff = NULL)
```

Arguments

Y	Series to be tested. Must be a ($T \times q$) matrix.
p	Number of autoregressive lags for the levels component. Must be greater than or equal to 1.
k0	Number of regimes under null hypothesis. Must be greater than or equal to 1.
k1	Number of regimes under alternative hypothesis. Must be greater than k0.

Z	Optional ($T \times q_Z$) matrix of fixed (deterministic) or strictly exogenous regressors, or NULL. Dummy variables are deterministic, hence fixed regressors, and are fully supported. The regressors' common coefficients never switch and are identified by the whole sample, so both components treat them like the other common parameters.
control	List with test procedure options for the LEVELS component, passed to <code>MMCLRTTest</code> unchanged (for a unit-root setting, pass <code>stationary_constraint = FALSE</code> , leave <code>phi_upp</code> unset, and put the unit-root point in <code>box_include</code>).
control_diff	List with test procedure options for the DIFFERENCED component. Default NULL derives it from <code>control</code> by removing the levels-specific box settings – <code>box_include</code> (whose length is the levels parameter vector's), <code>phi_low/phi_upp</code> , the boundary controls, and any vector-valued <code>eps</code> – and restoring <code>stationary_constraint = TRUE</code> , since the differenced component is the stationary world by construction. Pass it explicitly to override.

Value

List of class `aURdiffMMCTest` with `pval_levels`, `pval_diff`, `pval` (their maximum), `mmc_levels` and `mmc_diff` (the two full `MMCLRTTest` objects), `control`, and `control_diff`.

References

Rodriguez-Rondon, G., & Dufour, J.-M. 2026a. "Monte Carlo Likelihood-Ratio Tests for Markov Switching Models." *Bank of Canada Staff Working Paper*, No. 2026-23. doi: 10.34989/swp-2026-23.

aURfixLMCTest

Unit-root-augmented (fix) Local Monte Carlo Likelihood Ratio Test

Description

Runs the Local Monte Carlo likelihood ratio test of Rodriguez-Rondon & Dufour (2026a) twice on the same observed series and the same pre-drawn simulation draws: once with the null distribution simulated at the fitted null-model parameters (the ordinary LMC-LRT), and once with the null-simulation parameter's long-run autoregressive coefficient moved to exactly 1 (everything else unchanged), so the second component's simulated samples have an exact unit root – the Monte Carlo counterpart of imposing the unit root in the resampling scheme (Basawa, Mallik, McCormick, Reeves & Taylor, 1991). The test rejects only if BOTH component p-values are below the chosen level (an intersection-union composition, reported as `pval = max(pval_lmc, pval_fix)`), which protects the level when the data may carry a unit root. For $p > 1$ the long-run coefficient `sum(phi)` is moved to 1 while the short-run (augmented Dickey-Fuller) coefficients are held fixed – linear arithmetic on the fitted vector, nothing re-estimated; the resulting vector is reported as `theta_fix`. Exogenous regressors, when present, keep their estimated coefficients in the unit-root-fixed simulation parameter. Estimation on the simulated samples stays free (unrestricted) in both components, identically to the observed side. At a one-regime null with $p = 1$ and no exogenous regressors the imposed-root component's guarantee at the unit root is exact in finite samples: the fixed simulation

parameter carries no estimated autoregressive coefficient, and the remaining location-scale nuisance is absorbed by invariance (the level cancels at the unit root, the scale by equivariance). For $p \geq 2$ the short-run coefficients enter the fixed parameter at their estimates and the component is asymptotically valid at the unit root, as it is with regressors present; it is the differencing construction that retains finite-sample exactness at $p = 1$ with regressors – see [aURdiffLMCTest](#) when exactness with Z is needed. Univariate only: the augmented Dickey-Fuller map has no single multivariate counterpart (a VAR's unit-root null spans cointegration ranks 0 to $q - 1$, so a fixed point would impose an unstated rank choice); for a multivariate series use [aURdiffLMCTest](#), whose differencing construction needs no fixed point.

Usage

```
aURfixLMCTest(Y, p, k0, k1, Z = NULL, control = list())
```

Arguments

<code>Y</code>	Series to be tested. Must be a $(T \times 1)$ matrix (univariate only).
<code>p</code>	Number of autoregressive lags. Must be greater than or equal to 1.
<code>k0</code>	Number of regimes under null hypothesis. Must be greater than or equal to 1.
<code>k1</code>	Number of regimes under alternative hypothesis. Must be greater than <code>k0</code> .
<code>Z</code>	Optional $(T \times qz)$ matrix of fixed (deterministic) or strictly exogenous regressors, or NULL. Dummy variables are deterministic, hence fixed regressors, and are fully supported. The regressors' common coefficients never switch and are identified by the whole sample, so both components treat them like the other common parameters.
<code>control</code>	List with test procedure options: <code>N</code> , <code>burnin</code> , <code>converge_check</code> , <code>workers</code> , <code>mc_seed</code> , <code>init_method</code> , <code>mdl_h0_control</code> , <code>mdl_h1_control</code> , <code>use_diff_init_sim</code> , with the same meanings and defaults as in LMCLRTTest . With <code>mc_seed</code> set, <code>pval_lmc</code> is identical to LMCLRTTest 's p-value for the same inputs, and the second component reuses the same pre-drawn innovations and the same estimation-stream randomness, so the two components differ only through the simulation parameter.

Value

List of class `aURfixLMCTest` with `mdl_h0`, `mdl_h1`, `LRT_0` (the shared observed statistic), `pval_lmc` and `pval_fix` (the component p-values), `pval` (their maximum; reject only if it is below the level), `LRN` and `LRN_fix` (the two simulated null distributions), `LRN_cv` and `LRN_fix_cv`, `theta_fix` (the unit-root-fixed parameter vector), and `control`.

References

- Rodriguez-Rondon, G., & Dufour, J.-M. 2026a. "Monte Carlo Likelihood-Ratio Tests for Markov Switching Models." *Bank of Canada Staff Working Paper*, No. 2026-23. doi: 10.34989/swp-2026-23.
- Basawa, I. V., Mallik, A. K., McCormick, W. P., Reeves, J. H., & Taylor, R. L. 1991. "Bootstrapping Unstable First-Order Autoregressive Processes." *The Annals of Statistics* 19(2): 1098-1101.

Basawa, I. V., Mallik, A. K., McCormick, W. P., & Taylor, R. L. 1991. "Bootstrap Test of Significance and Sequential Bootstrap Estimation for Unstable First Order Autoregressive Processes." *Communications in Statistics - Theory and Methods* 20(3): 1015-1026.

canonicalize_regimes *Canonical regime labeling*

Description

Relabels the regimes of a fitted Markov-switching model in ascending order of fitted persistence ($P[j, j]$), so that state 1 is the least persistent (transient) regime and the last state is the most persistent. Estimation is invariant to regime relabeling, so a fitted model's labels are otherwise arbitrary; this makes them deterministic. Never called automatically by any estimation or testing function.

Usage

```
canonicalize_regimes mdl, by = "persistence")
```

Arguments

mdl	A fitted HMmdl, MSARmdl or MSVARmdl object (the exogenous variants MSARXmdl/MSVARXmdl return one of these base classes) with $k > 1$ regimes.
by	Character, the ordering rule. Currently only "persistence" (ascending fitted $P[j, j]$) is implemented.

Value

An object of the same class as mdl, with every field on the regime axis permuted to the canonical order (theta, mu, sigma, stdev, P, pinf, St, intercept, beta, standard errors when present, and the constraint-binding diagnostics). logLike, AIC, BIC, resid and fitted are unchanged (a relabeling does not change the likelihood). Fields with no regime axis (phi, betaZ, y, Z, n, p, q, k, control, the theta*_ind position maps, theta_0, init_used, trace) are unchanged.

References

Rodriguez-Rondon, G., & Dufour, J.-M. 2026a. "Monte Carlo Likelihood-Ratio Tests for Markov Switching Models." *Bank of Canada Staff Working Paper*, No. 2026-23. doi: 10.34989/swp-2026-23.

chp10GNP

Carrasco, Hu, & Ploberger 2010 GNP data

Description

This data is the extension of the GNP series used in CHP (2014), *Econometrica*. This series ranges from 1951Q2 to 2010Q4.

Usage

chp10GNP

Format

Date Vector of dates

GNP US GNP series

GNP_gr log difference of US GNP series

Source

<https://www.econometricsociety.org/content/supplement-optimal-test-markov-switching-parameters>

References

Carrasco, Marine, Liang Hu, and Werner Ploberger. 2014. "Optimal test for Markov switching parameters." *Econometrica* 82 (2): 765–784.

CHPbootCV

Bootstrap critical values for CHP 2014 parameter stability test

Description

Parametric bootstrap described on p. 771 of Carrasco, Hu & Ploberger (2014): simulate under the null at the fitted parameters, re-estimate, and recompute the test statistic, entirely in C++ (no calls back into R per replication).

Usage

CHPbootCV mdl, rho_b, N, msvar, burnin = 100L)

Arguments

mdl	List containing model attributes (see ARmdl): n, mu, sigma, phi.
rho_b	Number determining bounds for the grid search over rho.
N	Number of bootstrap replications.
msvar	Boolean. If TRUE, the alternative has switching mean and variance; if FALSE, switching mean only.
burnin	Burn-in length used to simulate each bootstrap sample from its stationary distribution. Default is 100.

Value

A (N x 2) matrix with the bootstrapped supTS statistic in the first column and expTS in the second.

References

Carrasco, Marine, Liang Hu, and Werner Ploberger. 2014. “Optimal test for Markov switching parameters.” *Econometrica* 82 (2): 765–784.

CHPTest

Carrasco, Hu, and Ploberger (2014) parameter stability test

Description

This function performs the CHP (2014) parameter stability test as outline in Carrasco, M., Hu, L. and Ploberger, W. (2014). Original source code can be found [here](#).

Usage

```
CHPTest(Y, p, control = list())
```

Arguments

Y	A (T x 1) matrix of observations.
p	Integer determining the number of autoregressive lags.
control	List with test procedure options including: • N: Integer determining the number of Bootstrap iterations. Default is set to 3000 as in paper. • rho_b: Number determining bounds for distribution of ρ_0 (i.e. $\rho \sim [-\rho_b, \rho_b]$). • msvar: Boolean indicator. If TRUE, there is a switch in variance. If FALSE only switch in mean is considered. • getSE: Boolean indicator. If TRUE, standard errors for restricted model are estimated. If FALSE no standard errors are estimated. Default is TRUE.

Value

List of class CHPTest (S3 object) with model attributes including:

- mdl_h0: List with restricted model attributes. This will be of class ARmdl (S3 object). See [ARmdl](#).
- supTS: supTS test statistic value.
- expTS: expTS test statistic value.
- supTS_N: A (N x 1) vector with simulated supTS test statistics under null hypothesis.
- expTS_N: A (N x 1) vector with simulated expTS test statistics under null hypothesis.
- pval_supTS: P-value for supTS version of parameter stability test.
- pval_expTS: P-value for expTS version of parameter stability test.
- supTS_cv: Vector with 90%, 95%, and 99% bootstrap critical values for supTS version of parameter stability test.
- expTS_cv: Vector with 90%, 95%, and 99% bootstrap critical values for expTS version of parameter stability test.
- control: List with test procedure options used.

References

Carrasco, Marine, Liang Hu, and Werner Ploberger. 2014. “Optimal test for Markov switching parameters.” *Econometrica* 82 (2): 765–784.

Examples

```
# load data used in Hamilton 1989
y84 <- as.matrix(hamilton84GNP$GNP_gr)

# Set test procedure options
control = list(N = 1000,
              rho_b = 0.7,
              msvar = FALSE)

# perform test with switch in mean only on Hamilton 1989 data

mdl_84_msmu <- CHPTest(y84, p = 4, control = control)
summary(mdl_84_msmu)
```

DLMCTest

*Monte Carlo moment-based test for Markov switching model***Description**

This function performs the Local Monte Carlo moment-based test for Markov switching autoregressive models proposed in Dufour & Luger (2017).

Usage

```
DLMCTest(Y, p, control = list())
```

Arguments

- | | |
|---------|--|
| Y | Series to be tested |
| p | Number of autoregressive lags. |
| control | List with test procedure options including: <ul style="list-style-type: none"> • N: Integer determining the number of Monte Carlo simulations. Default is set to 99 as in paper. • simdist_N: Integer determining the number of simulations for CDF distribution approximation. Default is set to 10000. • getSE: Boolean indicator. If TRUE, standard errors for restricted model are estimated. If FALSE no standard errors are estimated. Default is TRUE. |

Value

List of class DLMCTest (S3 object) with attributes including:

- mdl_h0: List with restricted model attributes. This will be of class ARmdl if $p > 0$ or Nmdl otherwise (S3 objects). See [ARmdl](#) or [Nmdl](#).
- theta: Value of nuisance parameters. Specifically, these are the consistent estimates of nuisance parameters as discussed in Dufour & Luger (2017) LMC procedure.
- S0: A (1×4) matrix containing the four moment-based test statistics defined in (11) - (14) in Dufour & Luger (2017).
- F0_min: Test statistic value for min version of Local Monte Carlo moment-based test.
- F0_prod: Test statistic value for prod version of Local Monte Carlo moment-based test.
- FN_min: A $(N \times 1)$ vector with simulated test statistics for min version of Local Monte Carlo moment-based test under null hypothesis.
- FN_prod: A $(N \times 1)$ vector with simulated test statistics for prod version of Local Monte Carlo moment-based test under null hypothesis.
- pval_min: P-value for min version of Local Monte Carlo moment-based test.
- pval_prod: P-value for prod version of Local Monte Carlo moment-based test.
- FN_min_cv: Vector with 90%, 95%, and 99% Monte Carlo critical values for min version of Local Monte Carlo moment-based test.

- FN_prod_cv: Vector with 90%, 95%, and 99% Monte Carlo critical values for prod version of Local Monte Carlo moment-based test.
- control: List with test procedure options used.

References

Dufour, J. M., & Luger, R. 2017. "Identification-robust moment-based tests for Markov switching in autoregressive models." *Econometric Reviews*, 36(6-9), 713-727.

Examples

```
set.seed(1234)
# load data used in Hamilton 1989 and extended data used in CHP 2014
y84 <- as.matrix(hamilton84GNP$GNP_gr)
y10 <- as.matrix(chp10GNP$GNP_gr)

# Set test procedure options
lmc_control = list(N = 99,
                  simdist_N = 10000,
                  getSE = TRUE)

# perform test on Hamilton 1989 data
lmc_gnp84 <- DLMMCTest(y84, p = 4, control = lmc_control)
summary(lmc_gnp84)
```

DLMMCTest	<i>Maximized Monte Carlo moment-based test for Markov switching model</i>
-----------	---

Description

This function performs the maximized Monte Carlo moment-based test for Markov switching autoregressive models proposed in Dufour & Luger (2017).

Usage

```
DLMMCTest(Y, p, control = list())
```

Arguments

Y	Series to be tested
p	Number of autoregressive lags.
control	List with test procedure options including: • N: Integer determining the number of Monte Carlo simulations. Default is set to 99 as in paper. • simdist_N: Integer determining the number of simulations for CDF distribution approximation. Default is set to 10000.

- `getSE`: Boolean indicator. If TRUE, standard errors for restricted model are estimated. If FALSE no standard errors are estimated. Default is TRUE.
- `eps`: Fixed non-negative constant that does not depend on T, or a vector of length p giving a different value per autoregressive coefficient, used to determine lower and upper bounds on consistent set considered for nuisance parameter space. Every entry must be finite and non-negative; `eps = 0` is valid and typically relies on `CI_union` to widen the box (with `optim_type = "GenSA"`, a zero entry is nudged to $1e-8$, since GenSA treats a genuinely zero-width bound as invalid, unlike `pso/GA`).
- `CI_union`: Boolean indicator determining if union between `eps` and confidence interval is used to determine lower and upper bound on consistent set considered for nuisance parameter space. If TRUE union is used and if FALSE only `eps` is used. A coefficient whose standard error is not finite falls back to `eps` alone for that coefficient only; other coefficients still use the union. Default is FALSE.
- `lambda`: Numeric value for penalty on stationary constraint not being met. Default is 100.
- `stationary_ind`: Boolean indicator determining if only stationary solutions should be considered if TRUE or any solution can be considered if FALSE. Default is TRUE.
- `phi_low`: Vector with lower bound for autoregressive parameters when optimizing. Default is NULL.
- `phi_upp`: Vector with upper bound for autoregressive parameters when optimizing. Default is NULL.
- `optim_type`: String determining type of numerical optimization algorithm to use. Available options are: `pso`, `GenSA`, `GA`. Default is `pso`.
- `silence`: Boolean indicator determining if optimization updates should be silenced if TRUE or not if FALSE. Default is FALSE.
- `threshold_stop`: Numeric value determining the maximum possible p-value attainable. Default is 1.
- `maxit`: Int determining the maximum number of function evaluations. Default is 1000.
- `optim_control`: List containing other optimization options specific to the numerical optimization algorithm used. For other options see documentation of numerical algorithm chosen.

Value

List of class `DLMMCTest` (S3 object) with attributes including:

- `mdl_h0`: List with restricted model attributes. This will be of class `ARmdl` if $p > 0$ or `Nmdl` otherwise (S3 objects). See `ARmdl` or `Nmdl`.
- `theta_max_min`: Value of nuisance parameters when min version of p-value is maximized as discussed in Dufour & Luger (2017) MMC procedure.
- `theta_max_prod`: Value of nuisance parameters when prod version of p-value is maximized as discussed in Dufour & Luger (2017) MMC procedure.

- `theta_low`: Lower bound on nuisance parameter values used when searching for maximum p-value.
- `theta_upper`: Upper bound on nuisance parameter values used when searching for maximum p-value.
- `S0_min`: A (1 x 4) matrix containing the four moment-based test statistics defined in (11) - (14) in Dufour & Luger (2017) when `theta_min` is used.
- `S0_prod`: A (1 x 4) matrix containing the four moment-based test statistics defined in (11) - (14) in Dufour & Luger (2017) when `theta_prod` is used.
- `F0_min`: Test statistic value for min version of Maximized Monte Carlo moment-based test.
- `F0_prod`: Test statistic value for prod version of Maximized Monte Carlo moment-based test.
- `FN_min`: A (N x 1) vector with simulated test statistics for min version of Maximized Monte Carlo moment-based test under null hypothesis.
- `FN_prod`: A (N x 1) vector with simulated test statistics for prod version of Maximized Monte Carlo moment-based test under null hypothesis.
- `pval_min`: Maximum p-value for min version of Maximized Monte Carlo moment-based test.
- `pval_prod`: Maximum p-value for prod version of Local Monte Carlo moment-based test.
- `FN_min_cv`: Vector with 90%, 95%, and 99% Monte Carlo critical values for min version of Local Monte Carlo moment-based test.
- `FN_prod_cv`: Vector with 90%, 95%, and 99% Monte Carlo critical values for prod version of Local Monte Carlo moment-based test.
- `control`: List with test procedure options used.
- `optim_min_output`: List with optimization output for min version of Maximized Monte Carlo moment-based test.
- `optim_prod_output`: List with optimization output for prod version of Maximized Monte Carlo moment-based test.

References

Dufour, J. M., & Luger, R. 2017. "Identification-robust moment-based tests for Markov switching in autoregressive models." *Econometric Reviews*, 36(6-9), 713-727.

Examples

```
set.seed(1234)
# load data used in Hamilton 1989 and extended data used in CHP 2014
y84 <- as.matrix(hamilton84GNP$GNP_gr)
y10 <- as.matrix(chp10GNP$GNP_gr)

# Set test procedure options
mmc_control <- list(N = 99,
                    simdist_N = 10000,
                    getSE = TRUE,
                    eps = 0,
                    CI_union = TRUE,
                    lambda = 100,
                    stationary_ind = TRUE,
```

```

optim_type = "GenSA",
silence = FALSE,
threshold_stop = 1,
maxit = 200)

# perform test on Hamilton 1989 data

mmc_gnp84 <- DLMCTest(y84, p = 4, control = mmc_control)
summary(mmc_gnp84)

```

hamilton84GNP

Hamilton 1984 & Hansen 1992 GNP data

Description

This data set is used in Hansen (1992) to test the US GNP model proposed by Hamilton (1989). This series ranges from 1951Q2 to 1984Q4.

Usage

```
hamilton84GNP
```

Format

Date Vector of dates

GNP US GNP series

GNP_gr US GNP log difference

Source

https://www.ssc.wisc.edu/~bhansen/progs/jae_92.html

References

Hansen, Bruce E. 1992. "The likelihood ratio test under nonstandard conditions: testing the Markov switching model of GNP." *Journal of applied Econometrics* 7 (S1): S61–S82.

Hamilton, James D. 1989. "A new approach to the economic analysis of nonstationary time series and the business cycle." *Econometrica* 57 (2): 357–384.

HLRTest

*Hansen (1992) likelihood ratio test***Description**

This function performs Hansen's likelihood ratio test as described in Hansen (1992). Original source code can be found [here](#).

Usage

```
HLRTest(Y, p, control = list())
```

Arguments

- | | |
|---------|--|
| Y | A (T x 1) matrix of observations. |
| p | Integer determining the number of autoregressive lags. |
| control | <p>List with test procedure options including:</p> <ul style="list-style-type: none"> • ix: List of Markov Switching parameters. 1 = just mean $c(1,2)$ = mean and first param, (default: 1). • msvar: Boolean indicator. If TRUE, there is a switch in variance. If FALSE only switch in mean is considered. Default is FALSE. • qbound: Indicator that bounds q by 1-p (default: FALSE). • gridsize: Integer determining the number of grid points for markov switching parameters. Default is 20. • pgrid_from: Double determining the initial grid point for transition probabilities. Default is 0.1. • pgrid_by: Double determining the step size for grid points of transition probabilities. This, along with p_gridsize will determine the bounds of search space. Default is 0.075. • pgrid_to: Double determining the end grid point for transition probabilities. Default is 0.925. • mugrid_from: Double determining the minimum value of mean in second regime. Default is 0.1. • mugrid_by: Double determining the step size for grid points of mean in second regime. This, along with gridsize will determine the max value of mean in second regime. Default is 0.1. • siggrid_from: Double determining the minimum value of sigma in second regime (if msvar = TRUE). Default is 0.1. • siggrid_by: Double determining the step size for grid points of sigma in second regime. This, along with gridsize will determine the max value of sigma in second regime. Default is 0.1. • N: Integer determining the number of replications. Default is 1000. • nwband: Integer determining maximum bandwidth in Bartlett kernel. Critical values and p-values are returned for each bandwidth from 0:nwband as suggested in Hansen (1996). Default is 4. |

- `theta_null_low`: Vector determining lower bound on parameters under the null hypothesis. Length of vector should be number of model coefficients + 1 for variance. Default is to only bound variance at 0.01.
- `theta_null_upp`: Vector determining upper bound on parameters under the null hypothesis. Length of vector should be number of model coefficients + 1 for variance. Default is to no bounds (i.e. Inf).
- `optim_method`: String determining the type of optimization procedure used. Allowed options are "gp-optim" for general purpose optimization using `optim` from or `slsqp`. Default is "gp-optim".

Value

List of class HLRTest (S3 object) with model attributes including:

- `mdl_h0`: List with restricted model attributes. This will be of class ARmdl (S3 object). See [ARmdl](#).
- `LR0`: Likelihood ratio test statistic value.
- `LRN`: A (N x 1) vector with simulated LRT statistics under null hypothesis.
- `pval`: P-value.
- `LR_cv`: A (nwband x 3) matrix with 90%, 95%, and 99% critical values in each column respectively.
- `coef`: Vector of coefficients from restricted model and grid search that maximized standardized LRT.
- `control`: List with test procedure options used.

References

Hansen, Bruce E. 1992. "The likelihood ratio test under nonstandard conditions: testing the Markov switching model of GNP." *Journal of applied Econometrics* 7 (S1): S61–S82.

Hansen, Bruce E. 1996. "Erratum: The likelihood ratio test under nonstandard conditions: testing the Markov switching model of GNP." *Journal of applied Econometrics* 7 (S1): S61–S82.

Examples

```
# ----- Use simulated process -----
set.seed(1234)
# Define DGP of MS AR process
mdl_ms2 <- list(n      = 200,
               mu      = c(5,1),
               sigma    = c(1,1),
               phi      = c(0.5),
               k        = 2,
               P        = rbind(c(0.90, 0.10),
                                c(0.10, 0.90)))

# Simulate process using simuMSAR() function
y_ms_simu <- simuMSAR(mdl_ms2)
```

```

hlrt_control <- list(ix      = 1,
                    gridsize = 7,
                    pgrid_from = 0.05,
                    pgrid_by  = 0.05,
                    pgrid_to  = 0.95,
                    mugrid_from = 0,
                    mugrid_by  = 1)

hlrt <- HLRTTest(y_ms_simu$y, p = 1, control = hlrt_control)
summary(hlrt)

```

HMmdl

Hidden Markov model

Description

This function estimates a Hidden Markov model with k regimes.

Usage

```
HMmdl(Y, k, Z = NULL, control = list())
```

Arguments

- | | |
|---------|--|
| Y | a ($T \times q$) matrix of observations. |
| k | integer determining the number of regimes to use in estimation. Must be greater than or equal to 2. |
| Z | an optional ($T \times q_z$) matrix of exogenous regressors. Default is NULL. |
| control | <p>List with model options including:</p> <ul style="list-style-type: none"> • <code>getSE</code>: Boolean. If TRUE standard errors are computed and returned. If FALSE standard errors are not computed. Default is TRUE. • <code>se_method</code>: String determining the standard-error method. Options are "hessian" (default) and "louis" (Louis 1982 expected-information); the Louis method is used automatically as a fallback when the Hessian is ill-conditioned. • <code>msmu</code>: Boolean. If TRUE model is estimated with switch in mean. If FALSE model is estimated with constant mean. Default is TRUE. • <code>msvar</code>: Boolean. If TRUE model is estimated with switch in variance. If FALSE model is estimated with constant variance. Default is TRUE. • <code>init_theta</code>: vector of initial values. vector must contain $(1 \times q)$ vector μ, $\text{vech}(\sigma)$, and $\text{vec}(P)$ where σ is a $(q \times q)$ covariance matrix. This is optional. Default is NULL, in which case <code>initVals_MSARmdl</code> is used to generate initial values. The last $k \times k$ entries must be $\text{vec}(P)$ (column-major) with entries in $[0,1]$ and columns summing to 1; invalid values raise an error. |

- `init_theta_extra`: List of additional deterministic starting vectors (same parameter ordering as `theta`) that are attempted first and compete with the `use_diff_init` random starts on log-likelihood (used internally by the Monte Carlo LR tests for warm starts; see `warmstart_theta`). Unlike `init_theta`, these do not bypass the multi-start search. Non-finite vectors are dropped; a wrong-length vector falls back to a random start. Default is NULL.
- `init_method`: Passive marker used by the Monte Carlo LR testing functions ("warmstart" or "random"); it has no effect on direct estimation. Default is NULL.
- `em_variance_constraint`: Double determining the lower bound imposed on the regime variances during EM estimation, as a fraction of the variance of the one-regime (linear) fit of the same data: $\sigma_k^2 \geq \text{em_variance_constraint} * \sigma_{\text{linear}}^2$ (for multivariate models, the smallest eigenvalue of each regime covariance is bounded by $\text{em_variance_constraint} * \text{trace}(\Sigma_{\text{linear}})/q$). Default is 0.01 (i.e. a regime standard deviation of at least 10% of the linear-model standard deviation); 0 or a negative value disables the constraint and reproduces the unconstrained behaviour of earlier versions. The constraint is needed because the Gaussian mixture (and Markov-switching) likelihood is unbounded: driving a regime variance to zero on a few observations sends the likelihood to infinity, so the unconstrained maximum likelihood estimate does not exist (Kiefer and Wolfowitz, 1956; Day, 1969; Hamilton, 1994, p. 689). Bounding the regime variances restores a well-defined maximizer (Hathaway, 1985, 1986, who instead bounds the ratios of the component scales); the data-dependent form used here follows Kasahara and Shimotsu (2018), on the variance rather than the standard-deviation scale. Clipping the variance update at the bound in each EM iteration is the exact constrained maximization step, so the algorithm retains its monotone ascent property. Fitted variances that end at the bound are reported in `sigma_floor_binding` and their standard errors are set to NA (they are boundary estimates).
- `em_transition_constraint`: Double bounding the transition probabilities away from the boundary during EM estimation: $p_{ij} \geq \text{em_transition_constraint}$ for every entry of the transition matrix (which implies $p_{ij} \leq 1 - \text{em_transition_constraint}$). Must be less than $1/k$; 0 (the default) or a negative value disables the constraint and reproduces the unconstrained update. Applies to `method = "EM"` only; `P_constraint_binding` is NULL for MLE fits. Because columns sum to 1, the bound implies each transition-matrix diagonal entry is at most $1 - (k-1)*\text{em_transition_constraint}$; at $k > 2$ choose the bound with this cap in mind (a bound of 0.05 caps diagonal persistence at 0.90 for $k = 3$). The constrained update is the exact maximizer of the expected complete-data log-likelihood over the bounded simplex (a water-filling solution), so the EM monotone-ascent property is preserved (Kim and Taylor, 1995). Intended for comparability with procedures that require transition probabilities bounded away from 0 and 1 (Kasahara and Shimotsu, 2018; Qu and Zhuo, 2021). Applies to the EM path only; for `method = "MLE"` use `mle_theta_low/mle_theta_upp`. Fitted entries at the bound are reported in `P_constraint_binding` and their standard errors are set to NA (bound-

ary estimates).

- `method`: string determining which method to use. Options are 'EM' for EM algorithm or 'MLE' for Maximum Likelihood Estimation. Default is 'EM'. Any other value is an error.
- `maxit`: integer determining the maximum number of EM iterations.
- `thtol`: double determining the convergence criterion for the relative change in the parameter estimates θ between iterations (used when `conv` is "theta", "both", or "both-A"). Default is $1e-6$.
- `ltol`: double determining the convergence criterion for the relative change in the log-likelihood between iterations (used when `conv` is "loglik", "both", "loglik-A", or "both-A"). Default is $1e-7$.
- `conv`: string determining the EM convergence criterion. "loglik" (the default) stops on the relative change in the log-likelihood; "theta" stops on the relative change in the parameters (the convention of Hamilton (1994) and of earlier versions of this package); "both" requires both, following Krolzig (1997); "loglik-A" and "both-A" replace the log-likelihood test with the Aitken-accelerated criterion of Bohning et al. (1994) and McLachlan and Krishnan (2008). Default is "loglik".
- `em_best_iterate`: Boolean. If TRUE (the default), the EM returns the iterate with the highest log-likelihood visited along the iteration path, with the smoothed probabilities, residuals, and log-likelihood evaluated at that iterate. The filter is initialized at the ergodic distribution implied by the transition matrix, whose dependence on the transition probabilities the EM update does not account for, so individual iterations can reduce the log-likelihood slightly; this option makes the returned fit the best one visited. If FALSE, the final iterate is returned (the convention of earlier versions). The output element `descents` reports the number of likelihood decreases observed along the path.
- `maxit_converge`: integer determining the maximum number of initial values attempted until solution is finite. For example, if parameters in `theta` or `logLike` are NaN another set of initial values (up to `maxit_converge`) is attempted until finite values are returned. This does not occur frequently for most types of data but may be useful in some cases. Once finite values are obtained, this counts as one iteration towards `use_diff_init`. Default is 10.
- `use_diff_init`: integer determining how many different initial values to try (that do not return NaN; see `maxit_converge`). Default is 1.
- `mle_variance_constraint`: Double used to determine the lower bound on the smallest eigenvalue of each regime covariance matrix. Value should be between 0 and 1 as it is multiplied by the average variance of the single regime fit (i.e., $\text{tr}(\text{Sigma})/q$). Default is 0.01. This is the method = "MLE" counterpart of `em_variance_constraint`, which imposes the same bound on the EM path.
- `mle_theta_low`: Vector with lower bounds on parameters (Used only if `method` = "MLE"). Default is NULL.
- `mle_theta_upp`: Vector with upper bounds on parameters (Used only if `method` = "MLE"). Default is NULL.

Value

List of class HMmdl (S3 object) with model attributes including:

- `y`: a $(T \times q)$ matrix of observations.
- `fitted`: a $(T \times q)$ matrix of fitted values.
- `resid`: a $(T \times q)$ matrix of residuals.
- `mu`: a $(k \times q)$ matrix of estimated means of each process.
- `beta`: if $q=1$, this is a $((1 + qz) \times k)$ matrix of estimated coefficients. If $q>1$, this is a list containing k separate $((1 + qz) \times q)$ matrix of estimated coefficients for each regime.
- `betaZ`: a $(qz \times q)$ matrix of estimated exogenous regressor coefficients.
- `intercept`: a $(k \times q)$ matrix of estimated intercept of each process. If Z is Null, this is the same as `mu`.
- `stdev`: If $q=1$, this is a $(k \times 1)$ matrix with estimated standard. If $q>1$, this is a List with k $(q \times q)$ matrices with estimated standard deviation on the diagonal.
- `sigma`: If $q=1$, this is a $(k \times 1)$ matrix with variances. If $q>1$, this is a List with k $(q \times q)$ estimated covariance matrix.
- `theta`: vector containing: `mu` and `vech(sigma)`.
- `theta_mu_ind`: vector indicating location of mean with 1 and 0 otherwise.
- `theta_sig_ind`: vector indicating location of variance and covariances with 1 and 0 otherwise.
- `theta_var_ind`: vector indicating location of variances with 1 and 0 otherwise.
- `theta_P_ind`: vector indicating location of transition matrix elements with 1 and 0 otherwise.
- `n`: number of observations (same as T).
- `q`: number of series.
- `k`: number of regimes in estimated model.
- `P`: a $(k \times k)$ transition matrix.
- `pinf`: a $(k \times 1)$ vector with limiting probabilities of each regime.
- `St`: a $(T \times k)$ vector with smoothed probabilities of each regime at each time t .
- `deltath`: double with the maximum relative change in vector `theta` on the last iteration.
- `iterations`: number of EM iterations performed to achieve convergence (if less than `maxit`).
- `converged`: Boolean. TRUE if the conv convergence criterion was met before `maxit` iterations, FALSE otherwise.
- `theta_0`: vector of initial values used.
- `init_used`: number of different initial values used to get a finite solution. See description of input `maxit_converge`.
- `msmu`: Boolean. If TRUE model was estimated with switch in mean. If FALSE model was estimated with constant mean.
- `msvar`: Boolean. If TRUE model was estimated with switch in variance. If FALSE model was estimated with constant variance.
- `control`: List with model options used.
- `logLike`: log-likelihood.

- AIC: Akaike information criterion.
- BIC: Bayesian (Schwarz) information criterion.
- Hess: Hessian matrix. Approximated using [hessian](#) and only returned if getSE=TRUE.
- info_mat: Information matrix. Computed as the inverse of -Hess. If matrix is not PD then nearest PD matrix is obtained using [nearest_spd](#). Only returned if getSE=TRUE.
- nearPD_used: Boolean determining whether nearPD function was used on info_mat if TRUE or not if FALSE. Only returned if getSE=TRUE.
- theta_se: standard errors of parameters in theta. Only returned if getSE=TRUE.
- trace: List with Lists of estimation output for each initial value used due to use_diff_init > 1.

References

- Dempster, A. P., N. M. Laird, and D. B. Rubin. 1977. "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society. Series B* 39 (1): 1–38..
- Böhning, D., E. Dietz, R. Schaub, P. Schlattmann, and B. G. Lindsay. 1994. "The distribution of the likelihood ratio for mixtures of densities from the one-parameter exponential family." *Annals of the Institute of Statistical Mathematics* 46 (2): 373–388.
- McLachlan, G. J., and T. Krishnan. 2008. *The EM Algorithm and Extensions*. 2nd ed. Hoboken, New Jersey: John Wiley & Sons.
- Hamilton, James D. 1990. "Analysis of time series subject to changes in regime." *Journal of econometrics*, 45 (1-2): 39–70.
- Krolzig, Hans-Martin. 1997. "The markov-switching vector autoregressive model.". Springer.

See Also

[Nmdl](#)

Examples

```
set.seed(1234)
# ----- Univariate ----- #
# Define DGP
mdl_hmm <- list(n      = 1000,
               q      = 1,
               mu      = as.matrix(c(5,-2)),
               sigma   = list(as.matrix(5.0),
                             as.matrix(7.0)),
               k      = 2,
               P      = rbind(c(0.90, 0.10),
                             c(0.10, 0.90)))

# Simulate process using simuHMM() function
y_hmm_simu <- simuHMM(mdl_hmm)

# Set options for model estimation
control <- list(msmu   = TRUE,
```

```

        msvar = TRUE,
        method = "EM",
        use_diff_init = 1)

# Estimate model
y_hmm_md1 <- HMmdl(y_hmm_simu$y, k = 2, control = control)
summary(y_hmm_md1)

# ----- Multivariate normal process ----- #
# Define DGP
mdl_hmm <- list(n      = 500,
               q      = 2,
               mu      = rbind(c(5, -2),
                               c(10, 2)),
               sigma = list(rbind(c(5.0, 1.5),
                                   c(1.5, 1.0)),
                             rbind(c(7.0, 3.0),
                                   c(3.0, 2.0))),
               k      = 2,
               P      = rbind(c(0.90, 0.10),
                               c(0.10, 0.90)))

# Simulate process using simuHMM() function
y_hmm_simu <- simuHMM(mdl_hmm)

# Set options for model estimation
control <- list(msmu = TRUE,
               msvar = TRUE,
               method = "EM",
               use_diff_init = 1)

# Estimate model

y_hmm_md1 <- HMmdl(y_hmm_simu$y, k = 2, control = control)
summary(y_hmm_md1)

```

Description

This function performs the Local Monte Carlo likelihood ratio test (LMC-LRT) proposed in Rodriguez-Rondon & Dufour (2026a). As discussed in their work, this test can be applied in very general settings and can be used to compare various regimes under the null and under the alternative.

Usage

```
LMCLRTTest(Y, p, k0, k1, Z = NULL, control = list())
```

Arguments

- | | |
|---------|--|
| Y | Series to be tested. Must be a (T x q) matrix where T is the number of time observations and q is the number of variables. |
| p | Number of autoregressive lags. Must be greater than or equal to 0. |
| k0 | Number of regimes under null hypothesis. Must be greater than or equal to 1. |
| k1 | Number of regimes under alternative hypothesis. Must be greater than k0. |
| Z | Exogenous regressors. Optional input and default is NULL. When used, it should be a (T x qz) matrix where T is the number of time observations and q is the number of exogenous variables. |
| control | <p>List with test procedure options including:</p> <ul style="list-style-type: none"> • N: Integer determining the number of Monte Carlo simulations. Default is set to 99 as in paper. • burnin: Number of simulated observations to remove from beginning. Default is 100. • converge_check: String or NULL determining if convergence of model(s) should be verified. Allowed inputs are: "null", "alt", "both", or NULL. If NULL (default) no model convergence is verified. • workers: Integer determining the number of workers to use for parallel computing. Default is 0 (sequential). If workers > N, the effective count is capped at N and an informational message is printed. • mc_seed: Integer seed for reproducible Monte Carlo simulations. When set, seeds the RNG before observed-data estimation and pre-draws innovations once, ensuring fully reproducible results (including the observed LRT statistic). When workers > 0, also sets L'Ecuyer-CMRG parallel RNG streams. Default is NULL, in which case all internal randomness is drawn from the ambient RNG state, so a script-level <code>set.seed()</code> placed before the call makes the procedure reproducible as well (sequential and parallel). • init_method: String determining how the alternative (k1-regime) model is initialized. "warmstart" (the default) adds deterministic warm starts built by embedding the null-model fit into the k1-regime space (see warmstart_theta) to the <code>use_diff_init</code> random starts, applied identically to the observed data and to every simulated draw; combined with the <code>LR >= 0</code> policy this makes negative LR statistics essentially impossible. "random" reproduces the legacy behavior (random starts only). • mdl_h0_control: List with restricted model options. See Nmdl, ARmdl, VARmdl, HMmdl, MSARmdl, or MSVARmdl documentation for available and default values. • mdl_h1_control: List with unrestricted model options. See HMmdl, MSARmdl, or MSVARmdl documentation for available and default values. • use_diff_init_sim: Value which determines the number of initial values to use when estimating models for null distribution. Default is set to use the same as specified in <code>mdl_h0_control</code> and <code>mdl_h1_control</code>. |

Value

List of class LMCLRTTest (S3 object) with attributes including:

- mdl_h0: List with restricted model attributes.
- mdl_h1: List with unrestricted model attributes.
- LRT_0: Value of test statistic from observed data.
- LRN: A (N x 1) vector of test statistics from data simulated under the null hypothesis.
- pval: P-value of Local Monte Carlo Likelihood Ratio Test.
- LRN_cv: Vector with 90%, 95%, and 99% Monte Carlo simulated critical values (from vector LRN). These are not asymptotic critical values.
- control: List with test procedure options used.

Numerical policy for the LR statistic

For nested models the LR statistic is non-negative by construction (the null fit embeds into the alternative parameter space with equal likelihood), so a negative computed value can only be a numerical optimization artifact. Both the observed statistic and every simulated statistic are therefore floored at 0 (identically on both sides, which preserves the exchangeability underlying the Monte Carlo p-value); a warning reports the magnitude when it is non-negligible. For $k_0 > 1$, the null log-likelihood is additionally floored at the one-regime (linear) fit on both sides. Ties at 0 are handled by the randomized tie-breaking rule of Dufour (2006) already implemented in `MCpval`; ties at the bottom of the support can only make the test conservative. The regime variances are bounded away from zero during estimation (control `em_variance_constraint` of the model constructors, default 0.01 of the one-regime fit variance), because the Markov-switching likelihood is otherwise unbounded and degenerate solutions produce spurious likelihood-ratio spikes; the statistic is therefore a constrained-likelihood ratio, computed by the same rule for the observed and the simulated samples, which is all the Monte Carlo argument requires. Transition probabilities are not constrained.

References

Rodriguez-Rondon, G., & Dufour, J.-M. 2026a. "Monte Carlo Likelihood-Ratio Tests for Markov Switching Models." *Bank of Canada Staff Working Paper*, No. 2026-23. doi: 10.34989/swp-2026-23.

Examples

```
set.seed(1234)
# Define DGP of MS AR process
mdl_ms2 <- list(n      = 200,
               mu      = c(5,10),
               sigma   = c(1,4),
               phi     = c(0.5),
               k        = 2,
               P        = rbind(c(0.90, 0.10),
                                c(0.10, 0.90)))

# Simulate process using simuMSAR() function
```

```

y_ms_simu <- simuMSAR mdl_ms2)

# ----- MS-AR example ----- #
# Set test procedure options
lmc_control = list(N = 19,
                   burnin = 100,
                   converge_check = NULL,
                   mdl_h0_control = list(const = TRUE,
                                           getSE = FALSE),
                   mdl_h1_control = list(msmu = TRUE,
                                           msvar = TRUE,
                                           getSE = FALSE,
                                           method = "EM",
                                           use_diff_init = 1))

# lmctest <- LMCLRTTest(y_ms_simu$y, p = 1, k0 = 1, k1 = 2, control = lmc_control)
# summary(lmctest)

```

MCpval

Monte Carlo P-value

Description

This function computes the Monte Carlo P-value.

Usage

```
MCpval(test_stat, null_vec, type = "geq")
```

Arguments

test_stat	Test statistic under the alternative (e.g. S_0).
null_vec	A ($N \times 1$) vector with test statistic under the null hypothesis.
type	String determining the type of test. Options are "geq" (right/upper-tail test; the default), "leq" (left/lower-tail test), "two-tailed" (or "two-tail"; two-sided test), and "absolute" (or "abs"; the upper-tail rule, intended for non-negative statistics). An unrecognized value raises an error.

Value

MC p-value of test

References

- Dufour, Jean-Marie 2006. "Monte Carlo tests with nuisance parameters: A general approach to finite-sample inference and nonstandard asymptotics". *Journal of Econometrics*, 133(2), 443-477.
- Dufour, Jean-Marie, and Richard Luger. 2017. "Identification-robust moment-based tests for Markov switching in autoregressive models". *Econometric Reviews*, 36(6-9), 713-727.

MMCLRTTest

Maximized Monte Carlo Likelihood Ratio Test

Description

This function performs the Maximized Monte Carlo likelihood ratio test (MMC-LRT) proposed in Rodriguez-Rondon & Dufour (2026a).

Usage

```
MMCLRTTest(Y, p, k0, k1, Z = NULL, control = list())
```

Arguments

- | | |
|---------|---|
| Y | Series to be tested. Must be a (T x q) matrix where T is the number of time observations and q is the number of variables. |
| p | Number of autoregressive lags. Must be greater than or equal to 0. |
| k0 | Number of regimes under null hypothesis. Must be greater than or equal to 1. |
| k1 | Number of regimes under alternative hypothesis. Must be greater than k0. |
| Z | Exogenous regressors. Optional input and default is NULL. When used, it should be a (T x qz) matrix where T is the number of time observations and q is the number of exogenous variables. |
| control | List with test procedure options including: <ul style="list-style-type: none"> • N: Integer determining the number of Monte Carlo simulations. Default is set to 99 as in paper. • burnin: Number of simulated observations to remove from beginning. Default is 100. • converge_check: String of NULL determining if convergence of model(s) should be verified. Allowed inputs are: "null", "alt", "both", or NULL. If NULL (default) no model convergence is verified. • workers: Integer determining the number of workers to use for parallel computing. Default is 0 (sequential). If workers > N, the effective count is capped at N and an informational message is printed (unless silence = TRUE). |

- `mc_seed`: Integer seed for reproducible Monte Carlo simulations and fixed-error MMC (Dufour 2006, Prop. 4.2). When set, seeds the RNG before observed-data estimation, pre-draws innovations once and holds them fixed across all optimizer evaluations, ensuring full reproducibility (including the observed LRT statistic) and theoretical size control. Internal seeds (optimizer trajectory, worker streams, CRN scheme) are drawn from the `mc_seed` stream rather than derived by arithmetic, so consecutive `mc_seed` values across replications of an outer simulation study do not produce colliding RNG streams; for studies with more than about 10^4 replications, well-separated seeds are still recommended. Default is `NULL`, in which case internal seeds are drawn from the ambient RNG state, so a script-level `set.seed()` placed before the call makes the procedure reproducible as well (sequential and parallel). Note: GenSA is deterministic given the fixed starting value used here, so its search trajectory does not depend on any seed; pso and GA use the optimizer seed.
- `init_method`: String determining how the alternative (k1-regime) model is initialized. "warmstart" (the default) adds deterministic warm starts built by embedding the null-model fit into the k1-regime space (see [warmstart_theta](#)) to the `use_diff_init` random starts, applied identically to the observed data and to every simulated draw. "random" reproduces the legacy behavior.
- `crn`: Boolean determining whether common random numbers are used across the nuisance-parameter search (default `TRUE`): an identical estimation-RNG stream (EM starting values; master and worker streams) is replayed at every candidate theta evaluation, so that together with the pre-drawn innovations the MC p-value is a deterministic function of theta (Dufour 2006, Prop. 4.2, with the estimation randomness folded into the fixed disturbance vector). This removes optimizer objective noise and guarantees `pval >= pval_0` exactly. Set to `FALSE` for the legacy behavior (estimation randomness evolves across evaluations; still valid, but the objective is noisy and the reported maximum tends to be conservative). Note: with `init_method = "random"`, `crn = FALSE`, `workers = 0` and no `mc_seed`, a script-seeded call reproduces version 0.1.9 bit-for-bit.
- `type`: String that determines the type of optimization algorithm used. Arguments allowed are: "pso", "GenSA", and "GA". Default is "pso".
- `eps`: Double, or a vector of length `length(mdl_h0$theta)`, determining the half-width of the search box around `theta_0` for each parameter (a scalar is recycled to every coordinate; a vector lets the search be narrower for some parameters, e.g. transition probabilities, and wider for others, e.g. means – entries are matched to `names(mdl_h0$theta)` positionally, i.e. in theta's own order, so mind that a transition-probability entry named `p_12` is matrix entry `P[2,1]`). Every entry must be finite and non-negative; a variance coordinate whose lower bound would reach zero is overridden by `variance_constraint` regardless of `eps`. `eps = 0` is valid and typically relies on `CI_union` to widen the box; with `type = "GenSA"` a zero entry is nudged to $1e-8$ (GenSA errors on a genuinely zero-width bound; pso/GA do not). Default is `0.1`.
- `CI_union`: Boolean determining if union of set determined by `eps` and con-

fidence set should be used to define consistent set for search. Default is TRUE.

- `lambda`: Double determining penalty on nonlinear constraint. Default is 100.
- `stationary_constraint`: Boolean determining if only stationary solutions are considered (if TRUE) or not (if FALSE). Default is TRUE.
- `phi_low`: Vector with lower bound for autoregressive parameters when optimizing. Default is NULL.
- `phi_upp`: Vector with upper bound for autoregressive parameters when optimizing. Default is NULL.
- `P_low`: Value with lower bound for transition probabilities when optimizing. Default is 0.
- `P_upp`: Value with upper bound for transition probabilities when optimizing. Default is 1.
- `variance_constraint`: Double used to determine the lower bound for variance in parameter set for search. Value should be between 0 and 1 as it is multiplied by consistent point estimates of variances. Default is 0.01 (i.e., 1% of consistent point estimates).
- `box_include`: Optional numeric vector of length `length(mdl_h0$theta)` giving, per parameter, a value the search box must contain (NA entries are left untouched). Applied after the `eps/CI_union` construction, so it only ever widens the box, and only where the ordinary construction does not already reach the value. The widening is TWO-SIDED: the coordinate's interval becomes the convex hull of the ordinary interval and the requested value, `[min(lower, v), max(upper, v)]`, so the value is covered even when the fitted estimate lies entirely on its far side – under a true unit root the fitted autoregressive coefficient exceeds 1 in a non-negligible fraction of samples, and raising only the upper endpoint would then leave the unit root outside the box. The admissible-region controls (`variance_constraint`, `P_low/P_upp`, `phi_low/phi_upp`) keep their usual meaning; if one of them excludes a requested value the call errors (the controls are contradictory) rather than silently searching a box that misses the requested point. The main use is unit-root testing: with `stationary_constraint = FALSE` and `phi_upp` not binding below 1, setting the autoregressive coordinate(s) of `box_include` to the unit-root point guarantees the search set contains it in every sample, per Rodriguez-Rondon & Dufour (2026a); for $p > 1$ pass the coefficients obtained by moving the long-run coefficient `sum(phi)` to 1 while holding the short-run (augmented Dickey-Fuller) coefficients fixed, as computed e.g. by `aURfixLMCTest`'s reported `theta_fix`. In this configuration keep the box tight above 1 (a small `eps`, e.g. 0.02, or `phi_upp` slightly above 1, e.g. 1.02): a box reaching far into the explosive region admits candidates whose simulated series overflow, so every one of their draws fails estimation and the candidate is only penalized after the full per-draw retry budget – valid, but a large waste of computation per such candidate. Default is NULL.
- `box_boundary`: Optional string enabling the boundary-setting search box for a null with $k_0 \geq 2$ regimes, one of which may be absorbing. Requires

burnin = 0: the boundary family simulates every candidate transient-start with no burn-in, and the regimes are relabeled canonically first via `canonicalize_regimes` so the absorbing regime is last. The family's initial condition is part of its definition: every candidate's chain is initialized in **canonical state 1 of the fitted null** – the least persistent transient regime at the fit – and over the search box that initial state is inherited by index (candidates are never re-sorted, even where a candidate's persistence ordering differs from the fit's). With $k_0 = 2$ the transient regime is unique, so this pins the initial state with no choice made; with $k_0 \geq 3$ it is a fixed-vertex family definition – a maintained modeling statement about the data, on the same footing as transient-start itself – and the procedure's finite-sample validity is conditional on it. At $p = 0$ the chain is initialized in the transient regime at time 0 (the first observation's state is a draw from that state's transition distribution). Multivariate models are supported – the transient block's ranges are built per series, and since a per-coordinate box cannot express positive definiteness, the search runs over the box intersected with the admissible parameter space (a candidate whose assembled regime covariance is not positive definite is excluded before simulation; see `pd_penalty_hits` in the output). Exogenous regressors are supported: their common coefficients belong to the clean block (identified by the whole sample, like the autoregressive coefficients), so they keep their ordinary range in "union" mode and are held at their estimates in "face" mode; Z must be fixed (deterministic) or strictly exogenous regressors – dummy variables are deterministic and fully supported. One scope note for the clean-block ranges: the consistency argument behind them attaches per coefficient, to coefficients whose regressor's information grows with the sample; a coefficient identified only by a fixed observation window (e.g. a short-lived dummy) is instead covered through the maintained containment of its range, so widen that coordinate's range if it matters. "union" stretches the ordinary search box per coordinate: the transient (rough) block's ranges are widened to the moment-anchored ranges of `boundary_R`, and the absorbing regime's transition-matrix column is stretched so the boundary face (its diagonal equal to 1) is inside the box; one ordinary search runs over the stretched box. "face" searches the boundary face alone: the transient block over the `boundary_R` ranges with everything else fixed (the absorbing diagonal at exactly 1, the remaining parameters at their estimates); the reported `pval_proj` is then the evaluation at the (unclamped) boundary projection of the fitted parameters, and the search seed is that projection clamped into the box (for a multivariate model the clamped seed's transient covariance is repaired to positive definiteness by deterministic off-diagonal shrinkage when needed, reported in `seed_repair`). Default is NULL (off; the ordinary search).

- `boundary_R`: List of moment-anchored ranges for the transient block when `box_boundary` is used: `mu_mult` (default 5; each transient mean searched over its own series' $\text{mean}(y_i) \pm \text{mu_mult} * \text{sd}(y_i)$), `sigma_mult` (default 4; each transient variance up to $\text{sigma_mult} * \text{var}(y_i)$ and down to the EM regime-variance floor; transient covariances range over $\pm \sqrt{U_i * U_j}$ with U_i the realized variance upper ends, so any positive semi-definite covariance matrix whose variances fit the box has its covariances

inside it automatically), and `p_range` (default `c(0.05, 0.95)`; transient-column transition probabilities). These ranges do the theory's coverage work for the parameters the pre-absorption era identifies, so a too-narrow choice loosens the size control; widen rather than narrow them.

- `box_match_constraints`: Boolean. If TRUE (the default), the nuisance search set is aligned with the estimation constraints carried by `mdl_h0_control`: with a stationarity margin (`stationarity_constraint = delta`) the search stays inside the margin region – the autoregressive box is clamped to $[-(1 - \delta), 1 - \delta]$ at $p = 1$, and for any p a candidate whose companion spectral radius exceeds $1 - \delta$ is excluded before simulation (counted in `margin_penalty_hits`; the margin itself is defined for univariate models only); with a transition-probability bound (`em_transition_constraint = eps`, meaningful for $k_0 > 1$ and for EM nulls – the bound is not part of the MLE path, so it is not aligned when `mdl_h0_control` sets `method = "MLE"`) `P_low` is raised to `eps` and `P_upp` lowered to $1 - (k_0 - 1) * \text{eps}$. A `box_include` autoregressive point outside the margin is contradictory and errors. The constrained fits obey these constraints on the observed and simulated sides, so this keeps the maximized test's search set equal to the maintained null parameter space, and the local test remains the search's seed point. It has no effect when no estimation constraint is set. A stationarity margin together with `stationarity_constraint = FALSE` is contradictory and errors. If FALSE, the box is built as usual and a warning notes that the search set may extend beyond the constrained space. Default is TRUE.
- `silence`: Boolean determining if optimization steps should be silenced (if TRUE) or not (if FALSE). Default is FALSE.
- `threshold_stop`: Double determining the p-value level at which the nuisance-parameter search stops early (the search seeks the maximum p-value, so once a candidate reaches this level the test decision at any smaller significance level is settled). If the p-value at the initial values already satisfies `pval_0 >= threshold_stop`, the optimizer is skipped entirely and `pval = pval_0` is returned (with `mmc_optimout` set to an early-stop marker). Default is 1 (no early stopping).
- `mdl_h0_control`: List with restricted model options. See [Nmdl](#), [ARmdl](#), [VARmdl](#), [HMmdl](#), [MSARmdl](#), or [MSVARmdl](#) documentation for available and default values.
- `mdl_h1_control`: List with unrestricted model options. See [HMmdl](#), [MSARmdl](#), or [MSVARmdl](#) documentation for available and default values.
- `use_diff_init_sim`: Value which determines the number of initial values to use when estimating models for null distribution. Default is set to use the same as specified in `mdl_h0_control` and `mdl_h1_control`.
- `maxit`: Int determining the budget of the nuisance-parameter search, as a cap on objective-function evaluations: passed as `maxf` to `pso` and as `max.call` to `GenSA` (both are evaluation counts, not iterations), and as `maxiter` to `GA` (generations, so roughly `popSize` times as many evaluations). Note that `GenSA` treats its cap as a soft bound and can substantially exceed it for higher-dimensional searches. Default is 50.
- `optim_control`: List with optimization algorithm options. See [psoptim](#), [GenSA](#), [ga](#). Default is `list()` (an empty list); the search budget is instead

controlled by the separate `maxit` control above.

Value

List of class `MMCLRTTest` (S3 object) with attributes including:

- `mdl_h0`: List with restricted model attributes.
- `mdl_h1`: List with unrestricted model attributes.
- `mdl_h0_mmc`: List with restricted model attributes at the maximizing nuisance parameter values.
- `mdl_h1_mmc`: List with unrestricted model attributes (same as `mdl_h1`).
- `LRT_0`: Value of test statistic from observed data.
- `pval`: Maximized p-value of the Maximized Monte Carlo Likelihood Ratio Test.
- `theta_h0`: Maximizing nuisance parameter vector for the restricted model.
- `theta_h1`: Parameter vector of the unrestricted model.
- `control`: List with test procedure options used.
- `mmc_optimout`: Optimization output object returned by the selected optimizer (`pso`, `GenSA`, or `GA`). For $k_0 \geq 2$ the search itself runs over a reduced parameter vector (the transition matrix contributes only its free entries), and the optimizer's reported solution (`$par` for `pso/GenSA`, `@solution` for `GA`) is mapped back to the full `theta` coordinates used everywhere else in the return value; other diagnostic fields inside `mmc_optimout` (e.g. `pso`'s `trace.stats$x`, `GenSA`'s `trace.mat`) are left in the reduced search coordinates.
- `pval_0`: Monte Carlo p-value at the initial (estimated) nuisance parameter values `theta_0` – the Local MC point of the search. Present for every estimate-centered search; the reported `pval` satisfies `pval >= pval_0` by construction (enforced directly, not merely relied on from the optimizer's own search). With `box_boundary = "face"` the fitted point lies outside the searched set, so this field is replaced by `pval_proj`: the evaluation at the unclamped boundary projection of the fitted parameters (reported as `theta_face_proj`), with `pval >= pval_proj` enforced the same way.
- `theta_low`, `theta_upp`: The realized search box (full parameter coordinates, named) after every widening and clamp – stored so it can be verified from saved output whether the box contained a target value.
- `optim_diagnostics`: List with `type` (the optimizer, or "early_stop"), `convergence` (the optimizer's own exit code where it reports one), `feval` (exact number of objective evaluations, counted directly), `optimizer_feval` (the optimizer's own reported call count where one exists, as a cross-check), `pd_penalty_hits` (number of candidates excluded because an assembled regime covariance was not positive definite; NA when that gate is not active, i.e. for univariate models or when neither `box_boundary` nor `box_include` is in use), `margin_penalty_hits` (number of candidates excluded by the stationarity-margin gate of `box_match_constraints`; NA when no margin is in force), and `evaluated_range` (per-coordinate min/max of the candidates the optimizer evaluated, in the reduced search coordinates, named by the searched positions; NULL when the search was skipped by `threshold_stop`).
- `boundary_mode`, `absorbing_regime`, `theta_face_proj`, `seed_repair`: Present only when `box_boundary` is used – the mode, the absorbing regime's index after canonical labeling (always k_0), and (face mode) the unclamped boundary projection whose evaluation is `pval_proj` together with

seed_repair (whether the clamped search seed's transient covariance needed the positive-definiteness repair, and the off-diagonal shrinkage factor applied). With box_boundary set, mdl_h0 is returned in canonical regime order (see [canonicalize_regimes](#)).

Numerical policy for the LR statistic

For nested models the LR statistic is non-negative by construction (the null fit embeds into the alternative parameter space with equal likelihood), so a negative computed value can only be a numerical optimization artifact. Both the observed statistic and every simulated statistic are therefore floored at 0 (identically on both sides, which preserves the exchangeability underlying the Monte Carlo p-value); a warning reports the magnitude when it is non-negligible. For $k_0 > 1$, the null log-likelihood is additionally floored at the one-regime (linear) fit on both sides. Ties at 0 are handled by the randomized tie-breaking rule of Dufour (2006) already implemented in [MCpval](#); ties at the bottom of the support can only make the test conservative. The regime variances are bounded away from zero during estimation (control `em_variance_constraint` of the model constructors, default 0.01 of the one-regime fit variance), because the Markov-switching likelihood is otherwise unbounded and degenerate solutions produce spurious likelihood-ratio spikes; the statistic is therefore a constrained-likelihood ratio, computed by the same rule for the observed and the simulated samples, which is all the Monte Carlo argument requires. Transition probabilities are not constrained.

References

Rodriguez-Rondon, G., & Dufour, J.-M. 2026a. "Monte Carlo Likelihood-Ratio Tests for Markov Switching Models." *Bank of Canada Staff Working Paper*, No. 2026-23. doi: 10.34989/swp-2026-23.

Examples

```
set.seed(1234)
# Define DGP of MS AR process
mdl_ms2 <- list(n      = 200,
               mu      = c(5,10),
               sigma    = c(1,4),
               phi      = c(0.5),
               k        = 2,
               P        = rbind(c(0.90, 0.10),
                                c(0.10, 0.90)))

# Simulate process using simuMSAR() function
y_ms_simu <- simuMSAR(mdl_ms2)

# Set test procedure options
mmc_control = list(N = 19,
                  burnin = 100,
                  converge_check = NULL,
                  eps = 0.1,
                  CI_union = TRUE,
                  silence = FALSE,
                  threshold_stop = 0.05 + 1e-6,
```

```

      type = "pso",
      maxit = 50,
      mdl_h0_control = list(const = TRUE,
                             getSE = TRUE),
      mdl_h1_control = list(msmu = TRUE,
                             msvar = TRUE,
                             getSE = TRUE,
                             method = "EM",
                             use_diff_init = 1))

# MMCtest <- MMCLRTTest(y_ms_simu$y, p = 1 , k0 = 1, k1 = 2, control = mmc_control)
# summary(MMCtest)

```

MSARmdl

*Markov-switching autoregressive model***Description**

This function estimates a Markov-switching autoregressive model

Usage

```
MSARmdl(Y, p, k, control = list())
```

Arguments

- | | |
|---------|---|
| Y | (T x 1) vector with observational data. |
| p | integer for the number of lags to use in estimation. Must be greater than or equal to 1. |
| k | integer for the number of regimes to use in estimation. Must be greater than or equal to 2. |
| control | List with model options including: <ul style="list-style-type: none"> • <code>getSE</code>: Boolean. If TRUE standard errors are computed and returned. If FALSE standard errors are not computed. Default is TRUE. • <code>se_method</code>: String determining the standard-error method. Options are "hessian" (default) and "louis" (Louis 1982 expected-information); the Louis method is used automatically as a fallback when the Hessian is ill-conditioned. • <code>msmu</code>: Boolean. If TRUE model is estimated with switch in mean. If FALSE model is estimated with constant mean. Default is TRUE. • <code>msvar</code>: Boolean. If TRUE model is estimated with switch in variance. If FALSE model is estimated with constant variance. Default is TRUE. |

- `init_theta`: vector of initial values. vector must contain $(1 \times q)$ vector μ , $\text{vech}(\sigma)$, and $\text{vec}(P)$ where σ is a $(q \times q)$ covariance matrix. This is optional. Default is NULL, in which case `initVals_MSARmdl` is used to generate initial values. The last $k \times k$ entries must be $\text{vec}(P)$ (column-major) with entries in $[0,1]$ and columns summing to 1; invalid values raise an error.
- `init_theta_extra`: List of additional deterministic starting vectors (same parameter ordering as `theta`) that are attempted first and compete with the `use_diff_init` random starts on log-likelihood (used internally by the Monte Carlo LR tests for warm starts; see `warmstart_theta`). Unlike `init_theta`, these do not bypass the multi-start search. Non-finite vectors are dropped; a wrong-length vector falls back to a random start. Default is NULL.
- `init_method`: Passive marker used by the Monte Carlo LR testing functions ("warmstart" or "random"); it has no effect on direct estimation. Default is NULL.
- `em_variance_constraint`: Double determining the lower bound imposed on the regime variances during EM estimation, as a fraction of the variance of the one-regime (linear) fit of the same data: $\sigma_k^2 \geq \text{em_variance_constraint} \times \sigma_{\text{linear}}^2$ (for multivariate models, the smallest eigenvalue of each regime covariance is bounded by $\text{em_variance_constraint} \times \text{trace}(\Sigma_{\text{linear}})/q$). Default is 0.01 (i.e. a regime standard deviation of at least 10% of the linear-model standard deviation); 0 or a negative value disables the constraint and reproduces the unconstrained behaviour of earlier versions. The constraint is needed because the Gaussian mixture (and Markov-switching) likelihood is unbounded: driving a regime variance to zero on a few observations sends the likelihood to infinity, so the unconstrained maximum likelihood estimate does not exist (Kiefer and Wolfowitz, 1956; Day, 1969; Hamilton, 1994, p. 689). Bounding the regime variances restores a well-defined maximizer (Hathaway, 1985, 1986, who instead bounds the ratios of the component scales); the data-dependent form used here follows Kasahara and Shimotsu (2018), on the variance rather than the standard-deviation scale. Clipping the variance update at the bound in each EM iteration is the exact constrained maximization step, so the algorithm retains its monotone ascent property. Fitted variances that end at the bound are reported in `sigma_floor_binding` and their standard errors are set to NA (they are boundary estimates).
- `em_transition_constraint`: Double bounding the transition probabilities away from the boundary during EM estimation: $p_{ij} \geq \text{em_transition_constraint}$ for every entry of the transition matrix (which implies $p_{ij} \leq 1 - \text{em_transition_constraint}$). Must be less than $1/k$; 0 (the default) or a negative value disables the constraint and reproduces the unconstrained update. Applies to `method = "EM"` only; `P_constraint_binding` is NULL for MLE fits. Because columns sum to 1, the bound implies each transition-matrix diagonal entry is at most $1 - (k-1) \times \text{em_transition_constraint}$; at $k > 2$ choose the bound with this cap in mind (a bound of 0.05 caps diagonal persistence at 0.90 for $k = 3$). The constrained update is the exact maximizer of the expected complete-data log-likelihood over the bounded simplex (a water-filling solution), so

the EM monotone-ascent property is preserved (Kim and Taylor, 1995). Intended for comparability with procedures that require transition probabilities bounded away from 0 and 1 (Kasahara and Shimotsu, 2018; Qu and Zhuo, 2021). Applies to the EM path only; for method = "MLE" use `mle_theta_low/mle_theta_upper`. Fitted entries at the bound are reported in `P_constraint_binding` and their standard errors are set to NA (boundary estimates).

- `stationarity_constraint`: Double giving a stationarity margin δ on the (common, non-switching) autoregressive coefficients: the fitted process satisfies companion spectral radius $\leq 1 - \delta$. 0 (the default) or a negative value disables the constraint and reproduces the long-standing unconstrained update, including its margin-0 stationarity guard, decision-for-decision. At $p = 1$ the constrained EM update is exact (the clip of the weighted-least-squares solve is the constrained maximizer of the expected complete-data log-likelihood, with the exogenous coefficients – when present – re-solved exactly at the clipped value); for $p \geq 2$ the update uses a root-scaling projection accepted only when it does not lower the M-step objective, so the autoregressive step never lowers the expected complete-data log-likelihood. Starting values are projected into the margin region when the constraint is enabled. Intended for comparability with procedures whose validity theory assumes stationarity, such as parametric bootstrap tests (Basawa, Mallik, McCormick, Reeves and Taylor, 1991); the Monte Carlo likelihood-ratio tests themselves do not require it. Distinct from `MMCLRTTest`'s `stationary_constraint` (the search's stationarity penalty); by default `MMCLRTTest` aligns its nuisance search set with the margin (control `box_match_constraints`). A binding fit is flagged in `phi_constraint_binding` and the autoregressive block's standard errors are set to NA. For method = "MLE" the margin imposes the characteristic-root bound moduli $\geq 1/(1 - \delta)$ whenever it is set (independently of `mle_stationary_constraint`), and a fit the solver leaves marginally outside the bound (its constraint tolerance) is projected onto it.
- `method`: string determining which method to use. Options are 'EM' for EM algorithm or 'MLE' for Maximum Likelihood Estimation. Default is 'EM'. Any other value is an error.
- `maxit`: integer determining the maximum number of EM iterations.
- `thtol`: double determining the convergence criterion for the relative change in the parameter estimates θ between iterations (used when `conv` is "theta", "both", or "both-A"). Default is $1e-6$.
- `ltol`: double determining the convergence criterion for the relative change in the log-likelihood between iterations (used when `conv` is "loglik", "both", "loglik-A", or "both-A"). Default is $1e-7$.
- `conv`: string determining the EM convergence criterion. "loglik" (the default) stops on the relative change in the log-likelihood; "theta" stops on the relative change in the parameters (the convention of Hamilton (1994) and of earlier versions of this package); "both" requires both, following Krolzig (1997); "loglik-A" and "both-A" replace the log-likelihood test with the Aitken-accelerated criterion of Bohning et al. (1994) and McLachlan and Krishnan (2008). Default is "loglik".

- `em_best_iterate`: Boolean. If TRUE (the default), the EM returns the iterate with the highest log-likelihood visited along the iteration path, with the smoothed probabilities, residuals, and log-likelihood evaluated at that iterate. The filter is initialized at the ergodic distribution implied by the transition matrix, whose dependence on the transition probabilities the EM update does not account for, so individual iterations can reduce the log-likelihood slightly; this option makes the returned fit the best one visited. If FALSE, the final iterate is returned (the convention of earlier versions). The output element `descents` reports the number of likelihood decreases observed along the path.
- `maxit_converge`: integer determining the maximum number of initial values attempted until solution is finite. For example, if parameters in `theta` or `logLike` are NaN another set of initial values (up to `maxit_converge`) is attempted until finite values are returned. This does not occur frequently for most types of data but may be useful in some cases. Once finite values are obtained, this counts as one iteration towards `use_diff_init`. Default is 10.
- `use_diff_init`: integer determining how many different initial values to try (that do not return NaN; see `maxit_converge`). Default is 1.
- `mle_stationary_constraint`: Boolean determining if only stationary solutions are considered (if TRUE) or not (if FALSE). Default is TRUE.
- `mle_variance_constraint`: Double used to determine the lower bound for variance in each regime. Value should be between 0 and 1 as it is multiplied by single regime variance. Default is 0.01 (i.e., 1% of single regime variance. This is the `method = "MLE"` counterpart of `em_variance_constraint`, which imposes the same bound on the EM path.
- `mle_theta_low`: Vector with lower bounds on parameters (Used only if `method = "MLE"`). Default is NULL.
- `mle_theta_upp`: Vector with upper bounds on parameters (Used only if `method = "MLE"`). Default is NULL.

Value

List of class `MSARmdl` (S3 object) with model attributes including:

- `y`: a $(T \times 1)$ matrix of observations.
- `X`: a $(T-p \times p + \text{const})$ matrix of lagged observations with a leading column of 1s.
- `x`: a $(T-p \times p)$ matrix of lagged observations.
- `fitted`: a $(T \times 1)$ matrix of fitted values.
- `resid`: a $(T \times 1)$ matrix of residuals.
- `intercept`: a $(k \times 1)$ vector of estimated intercepts of each process.
- `mu`: a $(k \times 1)$ vector of estimated means of each process.
- `beta`: a $((1 + p) \times k)$ matrix of estimated coefficients.
- `phi`: estimates of autoregressive coefficients.
- `stdev`: a $(k \times 1)$ vector of estimated standard deviation of each process.

- sigma: a (k x 1) estimated covariance matrix.
- theta: vector containing: mu and `vech(sigma)`.
- theta_mu_ind: vector indicating location of mean with 1 and 0 otherwise.
- theta_sig_ind: vector indicating location of variances with 1 and 0 otherwise.
- theta_var_ind: vector indicating location of variances with 1 and 0 otherwise. This is the same as `theta_sig_ind` in MSARmdl.
- theta_P_ind: vector indicating location of transition matrix elements with 1 and 0 otherwise.
- stationary: Boolean indicating if process is stationary if TRUE or non-stationary if FALSE.
- n: number of observations (same as T).
- p: number of autoregressive lags.
- q: number of series. This is always 1 in MSARmdl.
- k: number of regimes in estimated model.
- P: a (k x k) transition matrix.
- pinf: a (k x 1) vector with limiting probabilities of each regime.
- St: a (T x k) vector with smoothed probabilities of each regime at each time t.
- deltath: double with the maximum relative change in vector theta on the last iteration.
- iterations: number of EM iterations performed to achieve convergence (if less than `maxit`).
- converged: Boolean. TRUE if the conv convergence criterion was met before `maxit` iterations, FALSE otherwise.
- theta_0: vector of initial values used.
- init_used: number of different initial values used to get a finite solution. See description of input `maxit_converge`.
- msmu: Boolean. If TRUE model was estimated with switch in mean. If FALSE model was estimated with constant mean.
- msvar: Boolean. If TRUE model was estimated with switch in variance. If FALSE model was estimated with constant variance.
- control: List with model options used.
- logLike: log-likelihood.
- AIC: Akaike information criterion.
- BIC: Bayesian (Schwarz) information criterion.
- Hess: Hessian matrix. Approximated using `hessian` and only returned if `getSE=TRUE`.
- info_mat: Information matrix. Computed as the inverse of `-Hess`. If matrix is not PD then nearest PD matrix is obtained using `nearest_spd`. Only returned if `getSE=TRUE`.
- nearPD_used: Boolean determining whether `nearPD` function was used on `info_mat` if TRUE or not if FALSE. Only returned if `getSE=TRUE`.
- theta_se: standard errors of parameters in theta. Only returned if `getSE=TRUE`.
- trace: List with Lists of estimation output for each initial value used due to `use_diff_init > 1`.

References

- Dempster, A. P., N. M. Laird, and D. B. Rubin. 1977. "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society. Series B* 39 (1): 1–38.
- Böhning, D., E. Dietz, R. Schaub, P. Schlattmann, and B. G. Lindsay. 1994. "The distribution of the likelihood ratio for mixtures of densities from the one-parameter exponential family." *Annals of the Institute of Statistical Mathematics* 46 (2): 373–388.
- McLachlan, G. J., and T. Krishnan. 2008. *The EM Algorithm and Extensions*. 2nd ed. Hoboken, New Jersey: John Wiley & Sons.
- Hamilton, James D. 1990. "Analysis of time series subject to changes in regime." *Journal of econometrics*, 45 (1-2): 39–70.
- Basawa, I. V., Mallik, A. K., McCormick, W. P., Reeves, J. H., & Taylor, R. L. 1991. "Bootstrapping Unstable First-Order Autoregressive Processes." *The Annals of Statistics* 19(2): 1098-1101.

See Also

[ARmdl](#)

Examples

```
# ----- Use simulated process -----
set.seed(1234)
# Define DGP of MS AR process
mdl_ms2 <- list(n      = 500,
               mu      = c(5,10),
               sigma    = c(1,4),
               phi      = c(0.5),
               k        = 2,
               P        = rbind(c(0.90, 0.10),
                                c(0.10, 0.90)))

# Simulate process using simuMSAR() function
y_ms_simu <- simuMSAR(mdl_ms2)

# Set options for model estimation
control <- list(msmu    = TRUE,
               msvar    = TRUE,
               method    = "EM",
               use_diff_init = 1)

# Estimate model

ms_mdl <- MSARmdl(y_ms_simu$y, p = 1, k = 2, control = control)
summary(ms_mdl)
```

MSARXmdl

*Markov-switching autoregressive model with exogenous regressors***Description**

This function estimates a Markov-switching autoregressive model with exogenous regressors.

Usage

```
MSARXmdl(Y, p, k, Z, control = list())
```

Arguments

- | | |
|---------|--|
| Y | a (T x 1) vector with observational data. |
| p | integer for the number of lags to use in estimation. Must be greater than or equal to 1. |
| k | integer for the number of regimes to use in estimation. Must be greater than or equal to 2. |
| Z | a (T x qz) matrix of exogenous regressors. |
| control | <p>List with model options including:</p> <ul style="list-style-type: none"> • <code>getSE</code>: Boolean. If TRUE standard errors are computed and returned. If FALSE standard errors are not computed. Default is TRUE. • <code>se_method</code>: String determining the standard-error method. Options are "hessian" (default) and "louis" (Louis 1982 expected-information); the Louis method is used automatically as a fallback when the Hessian is ill-conditioned. • <code>msmu</code>: Boolean. If TRUE model is estimated with switch in mean. If FALSE model is estimated with constant mean. Default is TRUE. • <code>msvar</code>: Boolean. If TRUE model is estimated with switch in variance. If FALSE model is estimated with constant variance. Default is TRUE. • <code>init_theta</code>: vector of initial values. vector must contain (1 x q) vector μ, $\text{vech}(\sigma)$, and $\text{vec}(P)$ where σ is a (q x q) covariance matrix. This is optional. Default is NULL, in which case <code>initVals_MSARmdl</code> is used to generate initial values. The last k*k entries must be $\text{vec}(P)$ (column-major) with entries in [0,1] and columns summing to 1; invalid values raise an error. • <code>init_theta_extra</code>: List of additional deterministic starting vectors (same parameter ordering as <code>theta</code>) that are attempted first and compete with the <code>use_diff_init</code> random starts on log-likelihood (used internally by the Monte Carlo LR tests for warm starts; see <code>warmstart_theta</code>). Unlike <code>init_theta</code>, these do not bypass the multi-start search. Non-finite vectors are dropped; a wrong-length vector falls back to a random start. Default is NULL. |

- `init_method`: Passive marker used by the Monte Carlo LR testing functions ("warmstart" or "random"); it has no effect on direct estimation. Default is NULL.
- `em_variance_constraint`: Double determining the lower bound imposed on the regime variances during EM estimation, as a fraction of the variance of the one-regime (linear) fit of the same data: $\sigma_k^2 \geq \text{em_variance_constraint} * \sigma_{\text{linear}}^2$ (for multivariate models, the smallest eigenvalue of each regime covariance is bounded by $\text{em_variance_constraint} * \text{trace}(\Sigma_{\text{linear}})/q$). Default is 0.01 (i.e. a regime standard deviation of at least 10% of the linear-model standard deviation); 0 or a negative value disables the constraint and reproduces the unconstrained behaviour of earlier versions. The constraint is needed because the Gaussian mixture (and Markov-switching) likelihood is unbounded: driving a regime variance to zero on a few observations sends the likelihood to infinity, so the unconstrained maximum likelihood estimate does not exist (Kiefer and Wolfowitz, 1956; Day, 1969; Hamilton, 1994, p. 689). Bounding the regime variances restores a well-defined maximizer (Hathaway, 1985, 1986, who instead bounds the ratios of the component scales); the data-dependent form used here follows Kasahara and Shimotsu (2018), on the variance rather than the standard-deviation scale. Clipping the variance update at the bound in each EM iteration is the exact constrained maximization step, so the algorithm retains its monotone ascent property. Fitted variances that end at the bound are reported in `sigma_floor_binding` and their standard errors are set to NA (they are boundary estimates).
- `em_transition_constraint`: Double bounding the transition probabilities away from the boundary during EM estimation: $p_{ij} \geq \text{em_transition_constraint}$ for every entry of the transition matrix (which implies $p_{ij} \leq 1 - \text{em_transition_constraint}$). Must be less than $1/k$; 0 (the default) or a negative value disables the constraint and reproduces the unconstrained update. Applies to `method = "EM"` only; `P_constraint_binding` is NULL for MLE fits. Because columns sum to 1, the bound implies each transition-matrix diagonal entry is at most $1 - (k-1)*\text{em_transition_constraint}$; at $k > 2$ choose the bound with this cap in mind (a bound of 0.05 caps diagonal persistence at 0.90 for $k = 3$). The constrained update is the exact maximizer of the expected complete-data log-likelihood over the bounded simplex (a water-filling solution), so the EM monotone-ascent property is preserved (Kim and Taylor, 1995). Intended for comparability with procedures that require transition probabilities bounded away from 0 and 1 (Kasahara and Shimotsu, 2018; Qu and Zhuo, 2021). Applies to the EM path only; for `method = "MLE"` use `mle_theta_low/mle_theta_upper`. Fitted entries at the bound are reported in `P_constraint_binding` and their standard errors are set to NA (boundary estimates).
- `stationarity_constraint`: Double giving a stationarity margin δ on the (common, non-switching) autoregressive coefficients: the fitted process satisfies companion spectral radius $\leq 1 - \delta$. 0 (the default) or a negative value disables the constraint and reproduces the long-standing unconstrained update, including its margin-0 stationarity guard, decision-for-decision. At $\rho = 1$ the constrained EM update is exact (the clip of the

weighted-least-squares solve is the constrained maximizer of the expected complete-data log-likelihood, with the exogenous coefficients – when present – re-solved exactly at the clipped value); for $p \geq 2$ the update uses a root-scaling projection accepted only when it does not lower the M-step objective, so the autoregressive step never lowers the expected complete-data log-likelihood. Starting values are projected into the margin region when the constraint is enabled. Intended for comparability with procedures whose validity theory assumes stationarity, such as parametric bootstrap tests (Bassawa, Mallik, McCormick, Reeves and Taylor, 1991); the Monte Carlo likelihood-ratio tests themselves do not require it. Distinct from `MMCLRTTest`'s `stationary_constraint` (the search's stationarity penalty); by default `MMCLRTTest` aligns its nuisance search set with the margin (control box `match_constraints`). A binding fit is flagged in `phi_constraint_binding` and the autoregressive block's standard errors are set to NA. For `method = "MLE"` the margin imposes the characteristic-root bound $\text{moduli} \geq 1/(1 - \delta)$ whenever it is set (independently of `mle_stationary_constraint`), and a fit the solver leaves marginally outside the bound (its constraint tolerance) is projected onto it.

- `method`: string determining which method to use. Options are 'EM' for EM algorithm or 'MLE' for Maximum Likelihood Estimation. Default is 'EM'. Any other value is an error.
- `maxit`: integer determining the maximum number of EM iterations.
- `thtol`: double determining the convergence criterion for the relative change in the parameter estimates `theta` between iterations (used when `conv` is "theta", "both", or "both-A"). Default is $1e-6$.
- `ltol`: double determining the convergence criterion for the relative change in the log-likelihood between iterations (used when `conv` is "loglik", "both", "loglik-A", or "both-A"). Default is $1e-7$.
- `conv`: string determining the EM convergence criterion. "loglik" (the default) stops on the relative change in the log-likelihood; "theta" stops on the relative change in the parameters (the convention of Hamilton (1994) and of earlier versions of this package); "both" requires both, following Krolzig (1997); "loglik-A" and "both-A" replace the log-likelihood test with the Aitken-accelerated criterion of Bohning et al. (1994) and McLachlan and Krishnan (2008). Default is "loglik".
- `em_best_iterate`: Boolean. If TRUE (the default), the EM returns the iterate with the highest log-likelihood visited along the iteration path, with the smoothed probabilities, residuals, and log-likelihood evaluated at that iterate. The filter is initialized at the ergodic distribution implied by the transition matrix, whose dependence on the transition probabilities the EM update does not account for, so individual iterations can reduce the log-likelihood slightly; this option makes the returned fit the best one visited. If FALSE, the final iterate is returned (the convention of earlier versions). The output element `descents` reports the number of likelihood decreases observed along the path.
- `maxit_converge`: integer determining the maximum number of initial values attempted until solution is finite. For example, if parameters in `theta` or `logLike` are NaN another set of initial values (up to `maxit_converge`) is

attempted until finite values are returned. This does not occur frequently for most types of data but may be useful in some cases. Once finite values are obtained, this counts as one iteration towards `use_diff_init`. Default is 10.

- `use_diff_init`: integer determining how many different initial values to try (that do not return NaN; see `maxit_converge`). Default is 1.
- `mle_stationary_constraint`: Boolean determining if only stationary solutions are considered (if TRUE) or not (if FALSE). Default is TRUE.
- `mle_variance_constraint`: Double used to determine the lower bound for variance in each regime. Value should be between 0 and 1 as it is multiplied by single regime variance. Default is 0.01 (i.e., 1% of single regime variance. This is the method = "MLE" counterpart of `em_variance_constraint`, which imposes the same bound on the EM path.
- `mle_theta_low`: Vector with lower bounds on parameters (Used only if method = "MLE"). Default is NULL.
- `mle_theta_upper`: Vector with upper bounds on parameters (Used only if method = "MLE"). Default is NULL.

Value

List of class `MSARXmdl` (S3 object) with model attributes including:

- `y`: a $(T \times 1)$ matrix of observations.
- `X`: a $(T-p \times p + \text{const})$ matrix of lagged observations with a leading column of 1s.
- `x`: a $(T-p \times p)$ matrix of lagged observations.
- `fitted`: a $(T \times 1)$ matrix of fitted values.
- `resid`: a $(T \times 1)$ matrix of residuals.
- `intercept`: a $(k \times 1)$ vector of estimated intercepts of each process.
- `mu`: a $(k \times 1)$ vector of estimated means of each process.
- `beta`: a $((1 + p + qz) \times k)$ matrix of estimated coefficients.
- `betaZ`: a $(qz \times q)$ matrix of estimated exogenous regressor coefficients.
- `phi`: estimates of autoregressive coefficients.
- `stdev`: a $(k \times 1)$ vector of estimated standard deviation of each process.
- `sigma`: a $(k \times 1)$ estimated covariance matrix.
- `theta`: vector containing: `mu` and `vech(sigma)`.
- `theta_mu_ind`: vector indicating location of mean with 1 and 0 otherwise.
- `theta_sig_ind`: vector indicating location of variances with 1 and 0 otherwise.
- `theta_var_ind`: vector indicating location of variances with 1 and 0 otherwise. This is the same as `theta_sig_ind` in `MSARmdl`.
- `theta_P_ind`: vector indicating location of transition matrix elements with 1 and 0 otherwise.
- `stationary`: Boolean indicating if process is stationary if TRUE or non-stationary if FALSE.
- `n`: number of observations (same as `T`).

- p: number of autoregressive lags.
- q: number of series. This is always 1 in MSARmdl.
- k: number of regimes in estimated model.
- P: a (k x k) transition matrix.
- pinf: a (k x 1) vector with limiting probabilities of each regime.
- St: a (T x k) vector with smoothed probabilities of each regime at each time t.
- deltath: double with the maximum relative change in vector theta on the last iteration.
- iterations: number of EM iterations performed to achieve convergence (if less than maxit).
- converged: Boolean. TRUE if the conv convergence criterion was met before maxit iterations, FALSE otherwise.
- theta_0: vector of initial values used.
- init_used: number of different initial values used to get a finite solution. See description of input maxit_converge.
- msmu: Boolean. If TRUE model was estimated with switch in mean. If FALSE model was estimated with constant mean.
- msvar: Boolean. If TRUE model was estimated with switch in variance. If FALSE model was estimated with constant variance.
- control: List with model options used.
- logLike: log-likelihood.
- AIC: Akaike information criterion.
- BIC: Bayesian (Schwarz) information criterion.
- Hess: Hessian matrix. Approximated using [hessian](#) and only returned if getSE=TRUE.
- info_mat: Information matrix. Computed as the inverse of -Hess. If matrix is not PD then nearest PD matrix is obtained using [nearest_spd](#). Only returned if getSE=TRUE.
- nearPD_used: Boolean determining whether nearPD function was used on info_mat if TRUE or not if FALSE. Only returned if getSE=TRUE.
- theta_se: standard errors of parameters in theta. Only returned if getSE=TRUE.
- trace: List with Lists of estimation output for each initial value used due to use_diff_init > 1.

References

- Dempster, A. P., N. M. Laird, and D. B. Rubin. 1977. "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society. Series B* 39 (1): 1–38.
- Böhning, D., E. Dietz, R. Schaub, P. Schlattmann, and B. G. Lindsay. 1994. "The distribution of the likelihood ratio for mixtures of densities from the one-parameter exponential family." *Annals of the Institute of Statistical Mathematics* 46 (2): 373–388.
- McLachlan, G. J., and T. Krishnan. 2008. *The EM Algorithm and Extensions*. 2nd ed. Hoboken, New Jersey: John Wiley & Sons.
- Hamilton, James D. 1990. "Analysis of time series subject to changes in regime." *Journal of econometrics*, 45 (1-2): 39–70.
- Basawa, I. V., Mallik, A. K., McCormick, W. P., Reeves, J. H., & Taylor, R. L. 1991. "Bootstrapping Unstable First-Order Autoregressive Processes." *The Annals of Statistics* 19(2): 1098-1101.

See Also[ARmdl](#)**Examples**

```
# ----- Use simulated process -----
set.seed(1234)
# Define DGP of MS AR process
mdl_ms2 <- list(n      = 500,
               mu      = c(5,10),
               sigma    = c(1,4),
               phi      = c(0.5),
               k        = 2,
               P        = rbind(c(0.90, 0.10),
                                c(0.10, 0.90)))

# Simulate process using simuMSAR() function
y_ms_simu <- simuMSAR(mdl_ms2)

# Set options for model estimation
control <- list(msmu    = TRUE,
               msvar    = TRUE,
               method    = "EM",
               use_diff_init = 1)

# Estimate model

ms_mdl <- MSARmdl(y_ms_simu$y, p = 1, k = 2, control = control)
summary(ms_mdl)
```

MSVARmdl

*Markov-switching vector autoregressive model***Description**

This function estimates a Markov-switching vector autoregressive model

Usage

```
MSVARmdl(Y, p, k, control = list())
```

Arguments

Y (T x q) vector with observational data.

p	integer for the number of lags to use in estimation. Must be greater than or equal to 0.
k	integer for the number of regimes to use in estimation. Must be greater than or equal to 2.
control	List with optimization options including: • <code>getSE</code>: Boolean. If TRUE standard errors are computed and returned. If FALSE standard errors are not computed. Default is TRUE. • <code>se_method</code>: String determining the standard-error method. Options are "hessian" (default) and "louis" (Louis 1982 expected-information); the Louis method is used automatically as a fallback when the Hessian is ill-conditioned. • <code>msmu</code>: Boolean. If TRUE model is estimated with switch in mean. If FALSE model is estimated with constant mean. Default is TRUE. • <code>msvar</code>: Boolean. If TRUE model is estimated with switch in variance. If FALSE model is estimated with constant variance. Default is TRUE. • <code>init_theta</code>: vector of initial values. vector must contain (1 x q) vector μ, $\text{vech}(\sigma)$, and $\text{vec}(P)$ where σ is a (q x q) covariance matrix. This is optional. Default is NULL, in which case <code>initVals_MSARmdl</code> is used to generate initial values. The last k*k entries must be $\text{vec}(P)$ (column-major) with entries in [0,1] and columns summing to 1; invalid values raise an error. • <code>init_theta_extra</code>: List of additional deterministic starting vectors (same parameter ordering as theta) that are attempted first and compete with the <code>use_diff_init</code> random starts on log-likelihood (used internally by the Monte Carlo LR tests for warm starts; see <code>warmstart_theta</code>). Unlike <code>init_theta</code>, these do not bypass the multi-start search. Non-finite vectors are dropped; a wrong-length vector falls back to a random start. Default is NULL. • <code>init_method</code>: Passive marker used by the Monte Carlo LR testing functions ("warmstart" or "random"); it has no effect on direct estimation. Default is NULL. • <code>em_variance_constraint</code>: Double determining the lower bound imposed on the regime variances during EM estimation, as a fraction of the variance of the one-regime (linear) fit of the same data: $\sigma_k^2 \geq \text{em_variance_constraint} * \sigma_{\text{linear}}^2$ (for multivariate models, the smallest eigenvalue of each regime covariance is bounded by $\text{em_variance_constraint} * \text{trace}(\Sigma_{\text{linear}})/q$). Default is 0.01 (i.e. a regime standard deviation of at least 10% of the linear-model standard deviation); 0 or a negative value disables the constraint and reproduces the unconstrained behaviour of earlier versions. The constraint is needed because the Gaussian mixture (and Markov-switching) likelihood is unbounded: driving a regime variance to zero on a few observations sends the likelihood to infinity, so the unconstrained maximum likelihood estimate does not exist (Kiefer and Wolfowitz, 1956; Day, 1969; Hamilton, 1994, p. 689). Bounding the regime variances restores a well-defined maximizer (Hathaway, 1985, 1986, who instead bounds the ratios of the component scales); the data-dependent form used here follows Kasahara and Shimotsu (2018), on the variance rather than the standard-deviation

scale. Clipping the variance update at the bound in each EM iteration is the exact constrained maximization step, so the algorithm retains its monotone ascent property. Fitted variances that end at the bound are reported in `sigma_floor_binding` and their standard errors are set to NA (they are boundary estimates).

- `em_transition_constraint`: Double bounding the transition probabilities away from the boundary during EM estimation: $p_{ij} \geq \text{em_transition_constraint}$ for every entry of the transition matrix (which implies $p_{ij} \leq 1 - \text{em_transition_constraint}$). Must be less than $1/k$; 0 (the default) or a negative value disables the constraint and reproduces the unconstrained update. Applies to `method = "EM"` only; `P_constraint_binding` is NULL for MLE fits. Because columns sum to 1, the bound implies each transition-matrix diagonal entry is at most $1 - (k-1) \times \text{em_transition_constraint}$; at $k > 2$ choose the bound with this cap in mind (a bound of 0.05 caps diagonal persistence at 0.90 for $k = 3$). The constrained update is the exact maximizer of the expected complete-data log-likelihood over the bounded simplex (a water-filling solution), so the EM monotone-ascent property is preserved (Kim and Taylor, 1995). Intended for comparability with procedures that require transition probabilities bounded away from 0 and 1 (Kasahara and Shimotsu, 2018; Qu and Zhuo, 2021). Applies to the EM path only; for `method = "MLE"` use `mle_theta_low/mle_theta_upper`. Fitted entries at the bound are reported in `P_constraint_binding` and their standard errors are set to NA (boundary estimates).
- `method`: string determining which method to use. Options are 'EM' for EM algorithm or 'MLE' for Maximum Likelihood Estimation. Default is 'EM'. Any other value is an error.
- `maxit`: integer determining the maximum number of EM iterations.
- `thtol`: double determining the convergence criterion for the relative change in the parameter estimates `theta` between iterations (used when `conv` is "theta", "both", or "both-A"). Default is $1e-6$.
- `ltol`: double determining the convergence criterion for the relative change in the log-likelihood between iterations (used when `conv` is "loglik", "both", "loglik-A", or "both-A"). Default is $1e-7$.
- `conv`: string determining the EM convergence criterion. "loglik" (the default) stops on the relative change in the log-likelihood; "theta" stops on the relative change in the parameters (the convention of Hamilton (1994) and of earlier versions of this package); "both" requires both, following Krolzig (1997); "loglik-A" and "both-A" replace the log-likelihood test with the Aitken-accelerated criterion of Bohning et al. (1994) and McLachlan and Krishnan (2008). Default is "loglik".
- `em_best_iterate`: Boolean. If TRUE (the default), the EM returns the iterate with the highest log-likelihood visited along the iteration path, with the smoothed probabilities, residuals, and log-likelihood evaluated at that iterate. The filter is initialized at the ergodic distribution implied by the transition matrix, whose dependence on the transition probabilities the EM update does not account for, so individual iterations can reduce the log-likelihood slightly; this option makes the returned fit the best one visited. If FALSE, the final iterate is returned (the convention of earlier versions).

The output element descents reports the number of likelihood decreases observed along the path.

- `maxit_converge`: integer determining the maximum number of initial values attempted until solution is finite. For example, if parameters in `theta` or `logLike` are NaN another set of initial values (up to `maxit_converge`) is attempted until finite values are returned. This does not occur frequently for most types of data but may be useful in some cases. Once finite values are obtained, this counts as one iteration towards `use_diff_init`. Default is 10.
- `use_diff_init`: integer determining how many different initial values to try (that do not return NaN; see `maxit_converge`). Default is 1.
- `mle_stationary_constraint`: Boolean determining if only stationary solutions are considered (if TRUE) or not (if FALSE). Default is TRUE.
- `mle_variance_constraint`: Double used to determine the lower bound on the smallest eigenvalue of each regime covariance matrix. Value should be between 0 and 1 as it is multiplied by the average variance of the single regime fit (i.e., $\text{tr}(\text{Sigma})/q$). Default is 0.01. This is the method = "MLE" counterpart of `em_variance_constraint`, which imposes the same bound on the EM path.
- `mle_theta_low`: Vector with lower bounds on parameters (Used only if `method = "MLE"`). Default is NULL.
- `mle_theta_upp`: Vector with upper bounds on parameters (Used only if `method = "MLE"`). Default is NULL.

Value

List of class MSVARmdl (S3 object) with model attributes including:

- `y`: a $(T-p \times q)$ matrix of observations.
- `X`: a $(T-p \times p \times q + \text{const})$ matrix of lagged observations with a leading column of 1s.
- `x`: a $(T-p \times p \times q)$ matrix of lagged observations.
- `fitted`: a $(T \times q)$ matrix of fitted values.
- `resid`: a $(T-p \times q)$ matrix of residuals.
- `intercept`: a $(k \times q)$ matrix of estimated intercepts of each process.
- `mu`: a $(k \times q)$ matrix of estimated means of each process.
- `beta`: a list containing k separate $((1+p) \times q)$ matrix of estimated coefficients for each regime.
- `phi`: estimates of autoregressive coefficients.
- `Fmat`: Companion matrix containing autoregressive coefficients.
- `stdev`: List with k $(q \times q)$ matrices with estimated standard deviation on the diagonal.
- `sigma`: List with k $(q \times q)$ matrices with estimated covariance matrix.
- `theta`: vector containing: `mu` and `vech(sigma)`.
- `theta_mu_ind`: vector indicating location of mean with 1 and 0 otherwise.
- `theta_sig_ind`: vector indicating location of variance and covariances with 1 and 0 otherwise.

- `theta_var_ind`: vector indicating location of variances with 1 and 0 otherwise.
- `theta_P_ind`: vector indicating location of transition matrix elements with 1 and 0 otherwise.
- `stationary`: Boolean indicating if process is stationary if TRUE or non-stationary if FALSE.
- `n`: number of observations (same as `T`).
- `p`: number of autoregressive lags.
- `q`: number of series.
- `k`: number of regimes in estimated model.
- `P`: a $(k \times k)$ transition matrix.
- `pinf`: a $(k \times 1)$ vector with limiting probabilities of each regime.
- `St`: a $(T \times k)$ vector with smoothed probabilities of each regime at each time `t`.
- `deltath`: double with the maximum relative change in vector `theta` on the last iteration.
- `iterations`: number of EM iterations performed to achieve convergence (if less than `maxit`).
- `converged`: Boolean. TRUE if the conv convergence criterion was met before `maxit` iterations, FALSE otherwise.
- `theta_0`: vector of initial values used.
- `init_used`: number of different initial values used to get a finite solution. See description of input `maxit_converge`.
- `msmu`: Boolean. If TRUE model was estimated with switch in mean. If FALSE model was estimated with constant mean.
- `msvar`: Boolean. If TRUE model was estimated with switch in variance. If FALSE model was estimated with constant variance.
- `control`: List with model options used.
- `logLike`: log-likelihood.
- `AIC`: Akaike information criterion.
- `BIC`: Bayesian (Schwarz) information criterion.
- `Hess`: Hessian matrix. Approximated using `hessian` and only returned if `getSE=TRUE`.
- `info_mat`: Information matrix. Computed as the inverse of `-Hess`. If matrix is not PD then nearest PD matrix is obtained using `nearest_spd`. Only returned if `getSE=TRUE`.
- `nearPD_used`: Boolean determining whether `nearPD` function was used on `info_mat` if TRUE or not if FALSE. Only returned if `getSE=TRUE`.
- `theta_se`: standard errors of parameters in `theta`. Only returned if `getSE=TRUE`.
- `trace`: List with Lists of estimation output for each initial value used due to `use_diff_init > 1`.

References

- Dempster, A. P., N. M. Laird, and D. B. Rubin. 1977. "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society. Series B* 39 (1): 1–38.
- Böhning, D., E. Dietz, R. Schaub, P. Schlattmann, and B. G. Lindsay. 1994. "The distribution of the likelihood ratio for mixtures of densities from the one-parameter exponential family." *Annals of the Institute of Statistical Mathematics* 46 (2): 373–388.
- McLachlan, G. J., and T. Krishnan. 2008. *The EM Algorithm and Extensions*. 2nd ed. Hoboken, New Jersey: John Wiley & Sons.
- Krolzig, Hans-Martin. 1997. "The markov-switching vector autoregressive model.". Springer.

See Also[VARmdl](#)**Examples**

```

set.seed(123)
# Define DGP of MS VAR process
mdl_msvar2 <- list(n      = 200,
                  p      = 1,
                  q      = 2,
                  mu      = rbind(c(5, -2),
                                   c(10, 2)),
                  sigma = list(rbind(c(5.0, 1.5),
                                   c(1.5, 1.0)),
                               rbind(c(7.0, 3.0),
                                   c(3.0, 2.0))),
                  phi     = rbind(c(0.50, 0.30),
                                   c(0.20, 0.70)),
                  k       = 2,
                  P       = rbind(c(0.90, 0.10),
                                   c(0.10, 0.90)))

# Simulate process using simuMSVAR() function
y_msvar_simu <- simuMSVAR(mdl_msvar2)

# Set options for model estimation
control <- list(msmu    = TRUE,
               msvar    = TRUE,
               method   = "EM",
               use_diff_init = 1)

# Estimate model

y_msvar_mdl <- MSVARmdl(y_msvar_simu$y, p = 1, k = 2, control = control)
summary(y_msvar_mdl)

```

MSVARXmdl

Markov-switching vector autoregressive model with exogenous regressors

Description

This function estimates a Markov-switching vector autoregressive model with exogenous regressors.

Usage

```
MSVARXmdl(Y, p, k, Z, control = list())
```

Arguments

Y	(T x q) vector with observational data.
p	integer for the number of lags to use in estimation. Must be greater than or equal to 0.
k	integer for the number of regimes to use in estimation. Must be greater than or equal to 2.
Z	a (T x qz) matrix of exogenous regressors.
control	List with optimization options including: • <code>getSE</code>: Boolean. If TRUE standard errors are computed and returned. If FALSE standard errors are not computed. Default is TRUE. • <code>se_method</code>: String determining the standard-error method. Options are "hessian" (default) and "louis" (Louis 1982 expected-information); the Louis method is used automatically as a fallback when the Hessian is ill-conditioned. • <code>msmu</code>: Boolean. If TRUE model is estimated with switch in mean. If FALSE model is estimated with constant mean. Default is TRUE. • <code>msvar</code>: Boolean. If TRUE model is estimated with switch in variance. If FALSE model is estimated with constant variance. Default is TRUE. • <code>init_theta</code>: vector of initial values. vector must contain (1 x q) vector μ, $\text{vech}(\sigma)$, and $\text{vec}(P)$ where σ is a (q x q) covariance matrix. This is optional. Default is NULL, in which case <code>initVals_MSARmdl</code> is used to generate initial values. The last k*k entries must be $\text{vec}(P)$ (column-major) with entries in [0,1] and columns summing to 1; invalid values raise an error. • <code>init_theta_extra</code>: List of additional deterministic starting vectors (same parameter ordering as theta) that are attempted first and compete with the <code>use_diff_init</code> random starts on log-likelihood (used internally by the Monte Carlo LR tests for warm starts; see <code>warmstart_theta</code>). Unlike <code>init_theta</code>, these do not bypass the multi-start search. Non-finite vectors are dropped; a wrong-length vector falls back to a random start. Default is NULL. • <code>init_method</code>: Passive marker used by the Monte Carlo LR testing functions ("warmstart" or "random"); it has no effect on direct estimation. Default is NULL. • <code>em_variance_constraint</code>: Double determining the lower bound imposed on the regime variances during EM estimation, as a fraction of the variance of the one-regime (linear) fit of the same data: $\sigma_k^2 \geq \text{em_variance_constraint} * \sigma_{\text{linear}}^2$ (for multivariate models, the smallest eigenvalue of each regime covariance is bounded by $\text{em_variance_constraint} * \text{trace}(\Sigma_{\text{linear}})/q$). Default is 0.01 (i.e. a regime standard deviation of at least 10% of the linear-model standard deviation); 0 or a negative value disables the constraint and reproduces the unconstrained behaviour of earlier versions. The constraint is needed because the Gaussian mixture (and Markov-switching) likelihood is unbounded: driving a regime variance to zero on a few observations sends the likelihood to infinity, so the unconstrained maximum

likelihood estimate does not exist (Kiefer and Wolfowitz, 1956; Day, 1969; Hamilton, 1994, p. 689). Bounding the regime variances restores a well-defined maximizer (Hathaway, 1985, 1986, who instead bounds the ratios of the component scales); the data-dependent form used here follows Kasahara and Shimotsu (2018), on the variance rather than the standard-deviation scale. Clipping the variance update at the bound in each EM iteration is the exact constrained maximization step, so the algorithm retains its monotone ascent property. Fitted variances that end at the bound are reported in `sigma_floor_binding` and their standard errors are set to NA (they are boundary estimates).

- `em_transition_constraint`: Double bounding the transition probabilities away from the boundary during EM estimation: $p_{ij} \geq \text{em_transition_constraint}$ for every entry of the transition matrix (which implies $p_{ij} \leq 1 - \text{em_transition_constraint}$). Must be less than $1/k$; 0 (the default) or a negative value disables the constraint and reproduces the unconstrained update. Applies to `method = "EM"` only; `P_constraint_binding` is NULL for MLE fits. Because columns sum to 1, the bound implies each transition-matrix diagonal entry is at most $1 - (k-1) \times \text{em_transition_constraint}$; at $k > 2$ choose the bound with this cap in mind (a bound of 0.05 caps diagonal persistence at 0.90 for $k = 3$). The constrained update is the exact maximizer of the expected complete-data log-likelihood over the bounded simplex (a water-filling solution), so the EM monotone-ascent property is preserved (Kim and Taylor, 1995). Intended for comparability with procedures that require transition probabilities bounded away from 0 and 1 (Kasahara and Shimotsu, 2018; Qu and Zhuo, 2021). Applies to the EM path only; for `method = "MLE"` use `mle_theta_low/mle_theta_upper`. Fitted entries at the bound are reported in `P_constraint_binding` and their standard errors are set to NA (boundary estimates).
- `method`: string determining which method to use. Options are 'EM' for EM algorithm or 'MLE' for Maximum Likelihood Estimation. Default is 'EM'. Any other value is an error.
- `maxit`: integer determining the maximum number of EM iterations.
- `thtol`: double determining the convergence criterion for the relative change in the parameter estimates `theta` between iterations (used when `conv` is "theta", "both", or "both-A"). Default is $1e-6$.
- `ltol`: double determining the convergence criterion for the relative change in the log-likelihood between iterations (used when `conv` is "loglik", "both", "loglik-A", or "both-A"). Default is $1e-7$.
- `conv`: string determining the EM convergence criterion. "loglik" (the default) stops on the relative change in the log-likelihood; "theta" stops on the relative change in the parameters (the convention of Hamilton (1994) and of earlier versions of this package); "both" requires both, following Krolzig (1997); "loglik-A" and "both-A" replace the log-likelihood test with the Aitken-accelerated criterion of Bohning et al. (1994) and McLachlan and Krishnan (2008). Default is "loglik".
- `em_best_iterate`: Boolean. If TRUE (the default), the EM returns the iterate with the highest log-likelihood visited along the iteration path, with the smoothed probabilities, residuals, and log-likelihood evaluated at that

iterate. The filter is initialized at the ergodic distribution implied by the transition matrix, whose dependence on the transition probabilities the EM update does not account for, so individual iterations can reduce the log-likelihood slightly; this option makes the returned fit the best one visited. If FALSE, the final iterate is returned (the convention of earlier versions). The output element descents reports the number of likelihood decreases observed along the path.

- `maxit_converge`: integer determining the maximum number of initial values attempted until solution is finite. For example, if parameters in `theta` or `logLike` are NaN another set of initial values (up to `maxit_converge`) is attempted until finite values are returned. This does not occur frequently for most types of data but may be useful in some cases. Once finite values are obtained, this counts as one iteration towards `use_diff_init`. Default is 10.
- `use_diff_init`: integer determining how many different initial values to try (that do not return NaN; see `maxit_converge`). Default is 1.
- `mle_stationary_constraint`: Boolean determining if only stationary solutions are considered (if TRUE) or not (if FALSE). Default is TRUE.
- `mle_variance_constraint`: Double used to determine the lower bound on the smallest eigenvalue of each regime covariance matrix. Value should be between 0 and 1 as it is multiplied by the average variance of the single regime fit (i.e., $\text{tr}(\text{Sigma})/q$). Default is 0.01. This is the method = "MLE" counterpart of `em_variance_constraint`, which imposes the same bound on the EM path.
- `mle_theta_low`: Vector with lower bounds on parameters (Used only if `method = "MLE"`). Default is NULL.
- `mle_theta_upper`: Vector with upper bounds on parameters (Used only if `method = "MLE"`). Default is NULL.

Value

List of class `MSVARXmdl` (S3 object) with model attributes including:

- `y`: a $(T-p \times q)$ matrix of observations.
- `X`: a $(T-p \times p \times q + \text{const})$ matrix of lagged observations with a leading column of 1s.
- `x`: a $(T-p \times p \times q)$ matrix of lagged observations.
- `resid`: a $(T-p \times q)$ matrix of residuals.
- `fitted`: a $(T \times q)$ matrix of fitted values.
- `intercept`: a $(k \times q)$ matrix of estimated intercepts of each process.
- `mu`: a $(k \times q)$ matrix of estimated means of each process.
- `beta`: a list containing k separate $((1 + p + qz) \times q)$ matrix of estimated coefficients for each regime.
- `betaZ`: a $(qz \times q)$ matrix of estimated exogenous regressor coefficients.
- `phi`: estimates of autoregressive coefficients.
- `Fmat`: Companion matrix containing autoregressive coefficients.

- stdev: List with k ($q \times q$) matrices with estimated standard deviation on the diagonal.
- sigma: List with k ($q \times q$) matrices with estimated covariance matrix.
- theta: vector containing: mu and $\text{vech}(\text{sigma})$.
- theta_mu_ind: vector indicating location of mean with 1 and 0 otherwise.
- theta_sig_ind: vector indicating location of variance and covariances with 1 and 0 otherwise.
- theta_var_ind: vector indicating location of variances with 1 and 0 otherwise.
- theta_P_ind: vector indicating location of transition matrix elements with 1 and 0 otherwise.
- stationary: Boolean indicating if process is stationary if TRUE or non-stationary if FALSE.
- n: number of observations (same as T).
- p: number of autoregressive lags.
- q: number of series.
- k: number of regimes in estimated model.
- P: a ($k \times k$) transition matrix.
- pinf: a ($k \times 1$) vector with limiting probabilities of each regime.
- St: a ($T \times k$) vector with smoothed probabilities of each regime at each time t.
- deltath: double with the maximum relative change in vector theta on the last iteration.
- iterations: number of EM iterations performed to achieve convergence (if less than maxit).
- converged: Boolean. TRUE if the conv convergence criterion was met before maxit iterations, FALSE otherwise.
- theta_0: vector of initial values used.
- init_used: number of different initial values used to get a finite solution. See description of input maxit_converge.
- msmu: Boolean. If TRUE model was estimated with switch in mean. If FALSE model was estimated with constant mean.
- msvar: Boolean. If TRUE model was estimated with switch in variance. If FALSE model was estimated with constant variance.
- control: List with model options used.
- logLike: log-likelihood.
- AIC: Akaike information criterion.
- BIC: Bayesian (Schwarz) information criterion.
- Hess: Hessian matrix. Approximated using [hessian](#) and only returned if getSE=TRUE.
- info_mat: Information matrix. Computed as the inverse of -Hess. If matrix is not PD then nearest PD matrix is obtained using [nearest_spd](#). Only returned if getSE=TRUE.
- nearPD_used: Boolean determining whether nearPD function was used on info_mat if TRUE or not if FALSE. Only returned if getSE=TRUE.
- theta_se: standard errors of parameters in theta. Only returned if getSE=TRUE.
- trace: List with Lists of estimation output for each initial value used due to use_diff_init > 1.

References

- Dempster, A. P., N. M. Laird, and D. B. Rubin. 1977. "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society. Series B* 39 (1): 1–38.
- Böhning, D., E. Dietz, R. Schaub, P. Schlattmann, and B. G. Lindsay. 1994. "The distribution of the likelihood ratio for mixtures of densities from the one-parameter exponential family." *Annals of the Institute of Statistical Mathematics* 46 (2): 373–388.
- McLachlan, G. J., and T. Krishnan. 2008. *The EM Algorithm and Extensions*. 2nd ed. Hoboken, New Jersey: John Wiley & Sons.
- Krolzig, Hans-Martin. 1997. "The markov-switching vector autoregressive model". Springer.

See Also

[VARmdl](#)

Examples

```
set.seed(123)
# Define DGP of MS VAR process
mdl_msvar2 <- list(n      = 200,
                  p      = 1,
                  q      = 2,
                  mu      = rbind(c(5, -2),
                                   c(10, 2)),
                  sigma = list(rbind(c(5.0, 1.5),
                                       c(1.5, 1.0)),
                                rbind(c(7.0, 3.0),
                                       c(3.0, 2.0))),
                  phi    = rbind(c(0.50, 0.30),
                                   c(0.20, 0.70)),
                  k      = 2,
                  P      = rbind(c(0.90, 0.10),
                                   c(0.10, 0.90)))

# Simulate process using simuMSVAR() function
y_msvar_simu <- simuMSVAR(mdl_msvar2)

# Set options for model estimation
control <- list(msmu    = TRUE,
                msvar   = TRUE,
                method  = "EM",
                use_diff_init = 1)

# Estimate model

y_msvar_mdl <- MSVARmdl(y_msvar_simu$y, p = 1, k = 2, control = control)
summary(y_msvar_mdl)
```

Nmdl	<i>Normal distribution model</i>
------	----------------------------------

Description

This function estimates a univariate or multivariate normally distributed model. This can be used for the null hypothesis of a linear model against an alternative hypothesis of a HMM with k regimes.

Usage

```
Nmdl(Y, Z = NULL, control = list())
```

Arguments

- | | |
|----------------|--|
| Y | a $(T \times q)$ matrix of observations. |
| Z | an optional $(T \times q_z)$ matrix of exogenous regressors. Default is NULL. |
| control | List with model options including: <ul style="list-style-type: none"> • const: Boolean determining whether to estimate model with constant if TRUE or not if FALSE. Default is TRUE. • getSE: Boolean determining whether to compute standard errors of parameters if TRUE or not if FALSE. Default is TRUE. |

Value

List of class Nmdl (S3 object) with model attributes including:

- **y**: a $(T \times q)$ matrix of observations.
- **fitted**: a $(T \times q)$ matrix of fitted values.
- **resid**: a $(T \times q)$ matrix of residuals.
- **mu**: a $(1 \times q)$ vector of estimated means of each process.
- **intercept**: a $(1 \times q)$ vector of estimated intercept of each process. If **Z** is NULL, it is the same as **mu**.
- **beta**: a $((1 + p + q_z) \times q)$ matrix of estimated coefficients.
- **betaZ**: a $(q_z \times q)$ matrix of estimated exogenous regressor coefficients (If **Z** is provided).
- **stdev**: a $(q \times 1)$ vector of estimated standard deviation of each process.
- **sigma**: a $(q \times q)$ estimated covariance matrix.
- **theta**: vector containing: **mu**, **betaZ** (if matrix **Z** is provided), and **vech(sigma)**.
- **theta_mu_ind**: vector indicating location of mean with 1 and 0 otherwise.
- **theta_sig_ind**: vector indicating location of variance and covariances with 1 and 0 otherwise.
- **theta_var_ind**: vector indicating location of variances with 1 and 0 otherwise.
- **n**: number of observations (same as **T**).
- **q**: number of series.

- k: number of regimes. This is always 1 in Nmdl.
- control: List with model options used.
- logLike: log-likelihood.
- AIC: Akaike information criterion.
- BIC: Bayesian (Schwarz) information criterion.
- Hess: Hessian matrix. Approximated using [hessian](#) and only returned if getSE=TRUE.
- info_mat: Information matrix. Computed as the inverse of -Hess. If matrix is not PD then nearest PD matrix is obtained using [nearest_spd](#). Only returned if getSE=TRUE.
- nearPD_used: Boolean determining whether nearPD function was used on info_mat if TRUE or not if FALSE. Only returned if getSE=TRUE.
- theta_se: standard errors of parameters in theta. Only returned if getSE=TRUE.

Examples

```
set.seed(1234)

# ----- Univariate ----- #
# Define DGP
mdl_norm <- list(n      = 1000,
                q      = 1,
                mu      = as.matrix(5),
                sigma   = as.matrix(5.0))

# Simulate process using simuNorm() function
y_norm_simu <- simuNorm(mdl_norm)

# estimate parameters
y_norm_mdl <- Nmdl(y_norm_simu$y)
summary(y_norm_mdl)

# ----- Multivariate ----- #
# Define DGP
mdl_norm <- list(n      = 1000,
                q      = 2,
                mu      = c(5, -2),
                sigma   = rbind(c(5.0, 1.5),
                                c(1.5, 1.0)))

# Simulate process using simuNorm() function
y_norm_simu <- simuNorm(mdl_norm)

# estimate parameters
y_norm_mdl <- Nmdl(y_norm_simu$y)
summary(y_norm_mdl)
```

simuAR	<i>Simulate autoregressive process</i>
--------	--

Description

This function simulates an autoregressive process.

Usage

```
simuAR mdl_h0, burnin = 100)
```

Arguments

mdl_h0	List containing the following DGP parameters • n: Length of series. • mu: Mean of process. • sigma: Variance of process (the innovation standard deviation used in the simulation is $\sqrt{\text{sigma}}$). • phi: Vector of autoregressive coefficients. • eps: An optional $(T + \text{burnin} \times q)$ matrix with standard normal errors to be used. Errors will be generated if not provided.
burnin	Number of simulated observations to remove from beginning. Default is 100.

Value

List with simulated autoregressive series and its DGP parameters.

Examples

```
set.seed(1234)
# Define DGP of AR process
mdl_ar <- list(n      = 500,
              mu      = 5,
              sigma    = 2,
              phi      = c(0.5, 0.2))

# Simulate process using simuAR() function
y_simu <- simuAR(mdl_ar)

plot(y_simu)
```

 simuARX

Simulate autoregressive X process

Description

This function simulates an autoregressive X process.

Usage

```
simuARX mdl_h0, burnin = 100)
```

Arguments

mdl_h0	List containing the following DGP parameters • n: Length of series. • mu: Mean of process. • sigma: Variance of process (the innovation standard deviation used in the simulation is $\sqrt{\text{sigma}}$). • phi: Vector of autoregressive coefficients. • eps: An optional $(T + \text{burnin} \times q)$ matrix with standard normal errors to be used. Errors will be generated if not provided. • Z: A $(T \times qz)$ matrix with exogenous regressors (Optional) and where qz is the number of exogenous variables. • betaZ: A $(qz \times 1)$ matrix true coefficients on exogenous regressors (Optional) and where qz is the number of exogenous variables.
burnin	Number of simulated observations to remove from beginning. Default is 100.

Value

List with simulated autoregressive series and its DGP parameters.

Examples

```
set.seed(1234)
# Define DGP of AR process
mdl_ar <- list(n      = 500,
              mu      = 5,
              sigma    = 2,
              phi      = c(0.5, 0.2))

# Simulate process using simuAR() function
y_simu <- simuAR(mdl_ar)

plot(y_simu)
```

simuHMM

*Simulate Hidden Markov model with normally distributed errors***Description**

This function simulates a Hidden Markov Model process.

Usage

```
simuHMM mdl_h0, burnin = 100)
```

Arguments

mdl_h0	List containing the following DGP parameters • n: Length of series. • k: Number of regimes. • mu: A (k x q) vector of means. • sigma: A (q x q) covariance matrix. • q: Number of series. • P: A (k x k) transition matrix (columns must sum to one). • eps: An optional (T+burnin x q) matrix with standard normal errors to be used. Errors will be generated if not provided. • Z: A (T x qz) matrix with exogenous regressors (Optional) and where qz is the number of exogenous variables. • betaZ: A (qz x q) matrix true coefficients on exogenous regressors (Optional) and where qz is the number of exogenous variables.
burnin	Number of simulated observations to remove from beginning. Default is 100.

Value

List with simulated series and its DGP parameters.

Examples

```
set.seed(1234)

# ----- Univariate ----- #
# Define DGP
mdl_hmm <- list(n      = 1000,
               q       = 1,
               mu      = as.matrix(c(5,
                                     -2)),
               sigma    = list(as.matrix(5.0),
                              as.matrix(7.0)),
               k       = 2,
               P       = rbind(c(0.90, 0.10),
```

```

                                c(0.10, 0.90)))

# Simulate process using simuHMM() function
y_hmm_simu <- simuHMM mdl_hmm

plot(y_hmm_simu)

# ----- Multivariate ----- #
# Define DGP
mdl_hmm <- list(n      = 1000,
               q       = 2,
               mu      = rbind(c(5, -2),
                               c(10, 2)),
               sigma = list(rbind(c(5.0, 1.5),
                                   c(1.5, 1.0)),
                             rbind(c(7.0, 3.0),
                                   c(3.0, 2.0))),
               k       = 2,
               P       = rbind(c(0.90, 0.10),
                               c(0.10, 0.90)))

# Simulate process using simuHMM() function
y_hmm_simu <- simuHMM(mdl_hmm)

plot(y_hmm_simu)

```

simuMSAR

Simulate Markov-switching autoregressive process

Description

This function simulates a Markov-switching autoregressive process.

Usage

```
simuMSAR(mdl_h0, burnin = 100)
```

Arguments

mdl_h0	List containing the following DGP parameters • n: Length of series. • k: Number of regimes. • mu: A (k x 1) vector with mean of process in each regime. • sigma: A (k x 1) vector with the variance of the process in each regime (the innovation standard deviations used in the simulation are <code>sqrt(sigma)</code>). • phi: Vector of autoregressive coefficients. • P: A (k x k) transition matrix (columns must sum to one).
--------	--

- `eps`: An optional $(T + \text{burnin} \times q)$ matrix with standard normal errors to be used. Errors will be generated if not provided.
- `burnin` Number of simulated observations to remove from beginning. Default is 100.

Value

List with simulated Markov-switching autoregressive process and its DGP properties.

Examples

```
set.seed(1234)
# Define DGP of MS AR process
mdl_ms2 <- list(n      = 500,
               mu      = c(5,10),
               sigma    = c(1,2),
               phi      = c(0.5, 0.2),
               k        = 2,
               P        = rbind(c(0.90, 0.10),
                                c(0.10, 0.90)))

# Simulate process using simuMSAR() function
y_ms_simu <- simuMSAR(mdl_ms2)

plot(y_ms_simu)
```

simuMSARX

Simulate Markov-switching ARX process

Description

This function simulates a Markov-switching autoregressive process.

Usage

```
simuMSARX(mdl_h0, burnin = 100)
```

Arguments

- `mdl_h0` List containing the following DGP parameters
- `n`: Length of series.
 - `k`: Number of regimes.
 - `mu`: A $(k \times 1)$ vector with mean of process in each regime.
 - `sigma`: A $(k \times 1)$ vector with the variance of the process in each regime (the innovation standard deviations used in the simulation are `sqrt(sigma)`).
 - `phi`: Vector of autoregressive coefficients.
 - `P`: A $(k \times k)$ transition matrix (columns must sum to one).
 - `eps`: An optional $(T + \text{burnin} \times q)$ matrix with standard normal errors to be used. Errors will be generated if not provided.

- Z : A ($T \times q_z$) matrix with exogenous regressors (Optional) and where q_z is the number of exogenous variables.
- β_Z : A ($q_z \times 1$) matrix true coefficients on exogenous regressors (Optional) and where q_z is the number of exogenous variables.

burnin Number of simulated observations to remove from beginning. Default is 100.

Value

List with simulated Markov-switching autoregressive process and its DGP properties.

Examples

```
set.seed(1234)
# Define DGP of MS AR process
mdl_ms2 <- list(n      = 500,
               mu      = c(5,10),
               sigma    = c(1,2),
               phi      = c(0.5, 0.2),
               k        = 2,
               P        = rbind(c(0.90, 0.10),
                                c(0.10, 0.90)))

# Simulate process using simuMSAR() function
y_ms_simu <- simuMSAR(mdl_ms2)

plot(y_ms_simu)
```

simuMSVAR

Simulate Markov-switching vector autoregressive process

Description

This function simulates a Markov-switching vector autoregressive process.

Usage

```
simuMSVAR(mdl_h0, burnin = 100)
```

Arguments

mdl_h0 List containing the following DGP parameters

- n : Length of series.
- k : Number of regimes.
- μ : A ($k \times q$) matrix of means.
- σ : List with k ($q \times q$) covariance matrices.
- ϕ : A ($q \times qp$) matrix of autoregressive coefficients.
- p : Number of autoregressive lags.

- q: Number of series.
- P: A (k x k) transition matrix (columns must sum to one).
- eps: An optional (T+burnin x q) matrix with standard normal errors to be used. Errors will be generated if not provided.

burnin Number of simulated observations to remove from beginning. Default is 100.

Value

List with simulated vector autoregressive series and its DGP parameters.

Examples

```
set.seed(1234)
# Define DGP of MS VAR process
mdl_msvar2 <- list(n      = 1000,
                  p      = 1,
                  q      = 2,
                  mu     = rbind(c(5, -2),
                                c(10, 2)),
                  sigma = list(rbind(c(5.0, 1.5),
                                    c(1.5, 1.0)),
                              rbind(c(7.0, 3.0),
                                    c(3.0, 2.0))),
                  phi    = rbind(c(0.50, 0.30),
                                c(0.20, 0.70)),
                  k      = 2,
                  P      = rbind(c(0.90, 0.10),
                                c(0.10, 0.90)))

# Simulate process using simuMSVAR() function
y_msvar_simu <- simuMSVAR(mdl_msvar2)

plot(y_msvar_simu)
```

simuMSVARX

Simulate Markov-switching VARX process

Description

This function simulates a Markov-switching VARX process.

Usage

```
simuMSVARX(mdl_h0, burnin = 100)
```

Arguments

mdl_h0	List containing the following DGP parameters • n: Length of series. • k: Number of regimes. • mu: A (k x q) matrix of means. • sigma: List with k (q x q) covariance matrices. • phi: A (q x qp) matrix of autoregressive coefficients. • p: Number of autoregressive lags. • q: Number of series. • P: A (k x k) transition matrix (columns must sum to one). • eps: An optional (T+burnin x q) matrix with standard normal errors to be used. Errors will be generated if not provided. • Z: A (T x qz) matrix with exogenous regressors (Optional) and where qz is the number of exogenous variables. • betaZ: A (qz x q) matrix true coefficients on exogenous regressors (Optional) and where qz is the number of exogenous variables.
burnin	Number of simulated observations to remove from beginning. Default is 100.

Value

List with simulated vector autoregressive series and its DGP parameters.

Examples

```

set.seed(1234)
# Define DGP of MS VAR process
mdl_msvar2 <- list(n      = 1000,
                  p      = 1,
                  q      = 2,
                  mu      = rbind(c(5, -2),
                                   c(10, 2)),
                  sigma = list(rbind(c(5.0, 1.5),
                                       c(1.5, 1.0)),
                                rbind(c(7.0, 3.0),
                                       c(3.0, 2.0))),
                  phi     = rbind(c(0.50, 0.30),
                                   c(0.20, 0.70)),
                  k       = 2,
                  P       = rbind(c(0.90, 0.10),
                                   c(0.10, 0.90)))

# Simulate process using simuMSVAR() function
y_msvar_simu <- simuMSVAR(mdl_msvar2)

plot(y_msvar_simu)

```

simuNorm

*Simulate normally distributed process***Description**

This function simulates a normally distributed process.

Usage

```
simuNorm mdl_h0, burnin = 0)
```

Arguments

mdl_h0	List containing the following DGP parameters • n: Length of series. • mu: A (q x 1) vector of means. • sigma: A (q x q) covariance matrix. • q: Number of series. • eps: An optional (T+burnin x q) matrix with standard normal errors to be used. Errors will be generated if not provided. • Z: A (T x qz) matrix with exogenous regressors (Optional) and where qz is the number of exogenous variables. • betaZ: A (qz x q) matrix true coefficients on exogenous regressors (Optional) and where qz is the number of exogenous variables.
burnin	Number of simulated observations to remove from beginning. Default is 0.

Value

List with simulated series and its DGP parameters.

Examples

```
set.seed(1234)
# Define DGP
mdl_norm <- list(n      = 1000,
                q       = 2,
                mu      = c(5, -2),
                sigma   = rbind(c(5.0, 1.5),
                                c(1.5, 1.0)))

# Simulate process using simuNorm() function
y_norm_simu <- simuNorm(mdl_norm)

plot(y_norm_simu)
```

simuVAR	<i>Simulate VAR process</i>
---------	-----------------------------

Description

This function simulates a vector autoregressive process.

Usage

```
simuVAR mdl_h0, burnin = 100)
```

Arguments

mdl_h0	List containing the following DGP parameters • n: Length of series. • mu: A (q x 1) vector of means. • sigma: A (q x q) covariance matrix. • phi: A (q x qp) matrix of autoregressive coefficients. • p: Number of autoregressive lags. • q: Number of series. • eps: An optional (T+burnin x q) matrix with standard normal errors to be used. Errors will be generated if not provided.
burnin	Number of simulated observations to remove from beginning. Default is 100.

Value

List with simulated vector autoregressive series and its DGP parameters.

Examples

```
set.seed(1234)
# Define DGP of VAR process
mdl_3var2 <- list(n      = 1000,
                 p       = 2,
                 q       = 3,
                 mu      = c(5, -2, 1),
                 sigma    = rbind(c(5.0, 1.5, 2.5),
                                   c(1.5, 1.0, 1.5),
                                   c(2.5, 1.5, 4.2)),
                 phi     = rbind(c(0.70, 0.30, 0.35, -0.50, -0.20, 0.25),
                                   c(0.20, 0.40, 0.35, -0.30, 0.30, 0.25),
                                   c(0.20, 0.30, 0.50, -0.30, -0.20, -0.40)))

# Simulate process using simuVAR() function
y3var2_simu <- simuVAR(mdl_3var2)

plot(y3var2_simu)
```

simuVARX

*Simulate VARX process***Description**

This function simulates a vector autoregressive process with exogenous regressors.

Usage

```
simuVARX mdl_h0, burnin = 100)
```

Arguments

mdl_h0	List containing the following DGP parameters • n: Length of series. • mu: A (q x 1) vector of means. • sigma: A (q x q) covariance matrix. • phi: A (q x (q x p)) matrix of autoregressive coefficients. • p: Number of autoregressive lags. • q: Number of series. • eps: An optional (T+burnin x q) matrix with standard normal errors to be used. Errors will be generated if not provided. • Z: A (T x qz) matrix with exogenous regressors (Optional) and where qz is the number of exogenous variables. • betaZ: A (qz x q) matrix true coefficients on exogenous regressors (Optional) and where qz is the number of exogenous variables.
burnin	Number of simulated observations to remove from beginning. Default is 100.

Value

List with simulated vector autoregressive series and its DGP parameters.

Examples

```
set.seed(1234)
# Define DGP of VAR process
mdl_3var2 <- list(n      = 1000,
                 p       = 2,
                 q       = 3,
                 mu      = c(5, -2, 1),
                 sigma = rbind(c(5.0, 1.5, 2.5),
                               c(1.5, 1.0, 1.5),
                               c(2.5, 1.5, 4.2)),
                 phi     = rbind(c(0.70, 0.30, 0.35, -0.50, -0.20, 0.25),
                               c(0.20, 0.40, 0.35, -0.30, 0.30, 0.25),
                               c(0.20, 0.30, 0.50, -0.30, -0.20, -0.40)))
```

```
# Simulate process using simuVAR() function
y3var2_simu <- simuVAR mdl_3var2

plot(y3var2_simu)
```

USGNP	<i>US GNP data 1947Q2 - 2024Q2</i>
-------	------------------------------------

Description

This data is used in Rodriguez-Rondon & Dufour (2026a). The series ranges from 1947Q2 to 2024Q2.

Usage

USGNP

Format

- Date** Vector of dates
- GNP** US GNP series
- GNP_gr** log difference of US GNP series

Source

<https://fred.stlouisfed.org/graph/?g=UKDQ>

References

U.S. Bureau of Economic Analysis, Gross National Product [GNP], retrieved from FRED, Federal Reserve Bank of St. Louis; <https://fred.stlouisfed.org/series/GNP>, September, 2024.

Rodriguez-Rondon, G., & Dufour, J.-M. 2026a. "Monte Carlo Likelihood-Ratio Tests for Markov Switching Models." *Bank of Canada Staff Working Paper*, No. 2026-23. doi: 10.34989/swp-2026-23.

USRGDP	<i>US Real GDP data 1947Q2 - 2024Q2</i>
--------	---

Description

This data is used in Rodriguez-Rondon & Dufour (2026a). The series ranges from 1947Q2 to 2024Q2.

Usage

USRGDP

Format

- Date** Vector of dates
- RGDP** US Real GDP series
- RGDP_gr** log difference of US Real GDP series

Source

<https://fred.stlouisfed.org/series/GDPC1>

References

U.S. Bureau of Economic Analysis, Real Gross Domestic Product [GDPC1], retrieved from FRED, Federal Reserve Bank of St. Louis; <https://fred.stlouisfed.org/series/GDPC1>, September, 2024.

Rodriguez-Rondon, G., & Dufour, J.-M. 2026a. "Monte Carlo Likelihood-Ratio Tests for Markov Switching Models." *Bank of Canada Staff Working Paper*, No. 2026-23. doi: 10.34989/swp-2026-23.

VARmdl	<i>Vector autoregressive model</i>
--------	------------------------------------

Description

This function estimates a vector autoregressive model with p lags. This can be used for the null hypothesis of a linear model against an alternative hypothesis of a Markov switching vector autoregressive model with k regimes.

Usage

VARmdl(Y, p, control = list())

Arguments

<code>Y</code>	a $(T \times q)$ matrix of observations.
<code>p</code>	integer determining the number of autoregressive lags.
<code>control</code>	List with model options including: • <code>const</code>: Boolean determining whether to estimate model with constant if TRUE or not if FALSE. Default is TRUE. • <code>getSE</code>: Boolean determining whether to compute standard errors of parameters if TRUE or not if FALSE. Default is TRUE. • <code>stationarity_constraint</code>: Not implemented for multivariate models; a positive value is a loud error. Markov-switching estimation controls forwarded by the testing functions (<code>use_diff_init</code>, <code>em_variance_constraint</code>, <code>em_transition_constraint</code>) are accepted and ignored without a warning.

Value

List of class `VARmdl` (S3 object) with model attributes including:

- `y`: a $(T-p \times q)$ matrix of observations.
- `X`: a $(T-p \times p \times q + \text{const})$ matrix of lagged observations with a leading column of 1s if `const=TRUE` or not if `const=FALSE`.
- `x`: a $(T-p \times p \times q)$ matrix of lagged observations.
- `fitted`: a $(T-p \times q)$ matrix of fitted values.
- `resid`: a $(T-p \times q)$ matrix of residuals.
- `mu`: a $(1 \times q)$ vector of estimated means of each process.
- `beta`: a $((1 + p) \times q)$ matrix of estimated coefficients. .
- `intercept`: estimate of intercepts.
- `phi`: a $(q \times p \times q)$ matrix of estimated autoregressive coefficients.
- `Fmat`: Companion matrix containing autoregressive coefficients.
- `stdev`: a $(q \times 1)$ vector of estimated standard deviation of each process.
- `sigma`: a $(q \times q)$ estimated covariance matrix.
- `theta`: vector containing: `mu`, `vech(sigma)`, and `vec(t(phi))`.
- `theta_mu_ind`: vector indicating location of mean with 1 and 0 otherwise.
- `theta_sig_ind`: vector indicating location of variance and covariances with 1 and 0 otherwise.
- `theta_var_ind`: vector indicating location of variances with 1 and 0 otherwise.
- `theta_phi_ind`: vector indicating location of autoregressive coefficients with 1 and 0 otherwise.
- `stationary`: Boolean indicating if process is stationary if TRUE or non-stationary if FALSE.
- `n`: number of observations after lag transformation (i.e., $n = T-p$).
- `p`: number of autoregressive lags.
- `q`: number of series.
- `k`: number of regimes. This is always 1 in `VARmdl`.

- Fmat: matrix from companion form. Used to determine if process is stationary.
- control: List with model options used.
- logLike: log-likelihood.
- AIC: Akaike information criterion.
- BIC: Bayesian (Schwarz) information criterion.
- Hess: Hessian matrix. Approximated using [hessian](#) and only returned if getSE=TRUE.
- info_mat: Information matrix. Computed as the inverse of -Hess. If matrix is not PD then nearest PD matrix is obtained using [nearest_spd](#). Only returned if getSE=TRUE.
- nearPD_used: Boolean determining whether nearPD function was used on info_mat if TRUE or not if FALSE. Only returned if getSE=TRUE.
- theta_se: standard errors of parameters in theta. Only returned if getSE=TRUE.

See Also

[MSVARmdl](#)

Examples

```
# ----- Bivariate VAR(1) process ----- #
set.seed(1234)
# Define DGP of VAR process
mdl_var <- list(n      = 1000,
               p       = 1,
               q       = 2,
               mu      = c(5,-2),
               sigma   = rbind(c(5.0, 1.5),
                               c(1.5, 1.0)),
               phi     = rbind(c(0.50, 0.30),
                               c(0.20, 0.70)))

# Simulate process using simuVAR() function
y_simu <- simuVAR(mdl_var)

# Set options for model estimation
control <- list(const = TRUE,
                 getSE = TRUE)

# Estimate model
y_var_mdl <- VARmdl(y_simu$y, p = 2, control = control)
summary(y_var_mdl)
```

Description

This function estimates a vector autoregressive model with p lags. This can be used for the null hypothesis of a linear model against an alternative hypothesis of a Markov switching vector autoregressive model with k regimes.

Usage

```
VARXmdl(Y, p, Z, control = list())
```

Arguments

- | | |
|---------|---|
| Y | a ($T \times q$) matrix of observations. |
| p | integer determining the number of autoregressive lags. |
| Z | a ($T \times qz$) matrix of exogenous regressors. |
| control | List with model options including: <ul style="list-style-type: none"> • <code>const</code>: Boolean determining whether to estimate model with constant if TRUE or not if FALSE. Default is TRUE. • <code>getSE</code>: Boolean determining whether to compute standard errors of parameters if TRUE or not if FALSE. Default is TRUE. • <code>stationarity_constraint</code>: Not implemented for multivariate models; a positive value is a loud error. Markov-switching estimation controls forwarded by the testing functions (<code>use_diff_init</code>, <code>em_variance_constraint</code>, <code>em_transition_constraint</code>) are accepted and ignored without a warning. |

Value

List of class VARmdl (S3 object) with model attributes including:

- `y`: a ($T-p \times q$) matrix of observations.
- `X`: a ($T-p \times p*q + \text{const}$) matrix of lagged observations with a leading column of 1s if `const=TRUE` or not if `const=FALSE`.
- `x`: a ($T-p \times p*q$) matrix of lagged observations.
- `fitted`: a ($T-p \times q$) matrix of fitted values.
- `resid`: a ($T-p \times q$) matrix of residuals.
- `mu`: a ($1 \times q$) vector of estimated means of each process.
- `beta`: a ($((1 + p + qz) \times q)$) matrix of estimated coefficients.
- `betaZ`: a ($qz \times q$) matrix of estimated exogenous regressor coefficients.
- `intercept`: estimate of intercepts.
- `phi`: a ($q \times p*q$) matrix of estimated autoregressive coefficients.
- `Fmat`: Companion matrix containing autoregressive coefficients.
- `stdev`: a ($q \times 1$) vector of estimated standard deviation of each process.
- `sigma`: a ($q \times q$) estimated covariance matrix.
- `theta`: vector containing: `mu`, `vech(sigma)`, and `vec(t(phi))`.

- `theta_mu_ind`: vector indicating location of mean with 1 and 0 otherwise.
- `theta_sig_ind`: vector indicating location of variance and covariances with 1 and 0 otherwise.
- `theta_var_ind`: vector indicating location of variances with 1 and 0 otherwise.
- `theta_phi_ind`: vector indicating location of autoregressive coefficients with 1 and 0 otherwise.
- `stationary`: Boolean indicating if process is stationary if TRUE or non-stationary if FALSE.
- `n`: number of observations after lag transformation (i.e., $n = T - p$).
- `p`: number of autoregressive lags.
- `q`: number of series.
- `k`: number of regimes. This is always 1 in VARmdl.
- `Fmat`: matrix from companion form. Used to determine if process is stationary.
- `control`: List with model options used.
- `logLike`: log-likelihood.
- `AIC`: Akaike information criterion.
- `BIC`: Bayesian (Schwarz) information criterion.
- `Hess`: Hessian matrix. Approximated using [hessian](#) and only returned if `getSE=TRUE`.
- `info_mat`: Information matrix. Computed as the inverse of $-Hess$. If matrix is not PD then nearest PD matrix is obtained using [nearest_spd](#). Only returned if `getSE=TRUE`.
- `nearPD_used`: Boolean determining whether `nearPD` function was used on `info_mat` if TRUE or not if FALSE. Only returned if `getSE=TRUE`.
- `theta_se`: standard errors of parameters in `theta`. Only returned if `getSE=TRUE`.

See Also

[MSVARmdl](#)

Examples

```
# ----- Bivariate VAR(1) process ----- #
set.seed(1234)
# Define DGP of VAR process
mdl_var <- list(n      = 1000,
               p       = 1,
               q       = 2,
               mu      = c(5,-2),
               sigma    = rbind(c(5.0, 1.5),
                                c(1.5, 1.0)),
               phi     = rbind(c(0.50, 0.30),
                                c(0.20, 0.70)))

# Simulate process using simuVAR() function
y_simu <- simuVAR(mdl_var)

# Set options for model estimation
control <- list(const = TRUE,
                 getSE = TRUE)
```

```
# Estimate model
y_var_md1 <- VARmdl(y_simu$y, p = 2, control = control)
summary(y_var_md1)
```

warmstart_theta

Warm-start values for the alternative model of a Monte Carlo LR test

Description

Builds deterministic starting values for the k_1 -regime (alternative) model by embedding a fitted k_0 -regime (null) model into the larger parameter space. Each embedding duplicates one or more regimes of the null fit (with the transition-matrix column split so that the likelihood at the exact embedding equals the null model's log-likelihood) and then applies a small deterministic antisymmetric perturbation to the duplicated regimes (means shifted by $\pm c_pert$ regime standard deviations when the mean switches; variances scaled by $(1 + c_pert)^{\pm 1}$ when the variance switches) so that the EM algorithm can leave the symmetric fixed point. One starting vector is returned per way of distributing the $k_1 - k_0$ extra regimes among the k_0 regimes of the null fit. These starts are used in addition to the random starting values when `init_method = "warmstart"` in [LMCLRTTest](#) and [MMCLRTTest](#).

Usage

```
warmstart_theta(mdl_h0, k1, msmu = NULL, msvar = NULL, c_pert = 0.5)
```

Arguments

<code>mdl_h0</code>	Fitted model under the null hypothesis (any model object returned by estimMdl ; a one-regime linear model or a Markov switching model with $k_0 > 1$ regimes).
<code>k1</code>	integer specifying the number of regimes of the alternative model (must exceed the number of regimes of <code>mdl_h0</code>).
<code>msmu</code>	Boolean indicating whether the mean switches in the alternative model. Default is <code>NULL</code> , in which case it is taken from <code>mdl_h0</code> when available and <code>TRUE</code> otherwise.
<code>msvar</code>	Boolean indicating whether the variance switches in the alternative model. Default is <code>NULL</code> (as for <code>msmu</code>).
<code>c_pert</code>	Double determining the size of the antisymmetric perturbation in regime standard-deviation units. Default is <code>0.5</code> .

Value

List of numeric vectors, each a valid starting value for the k_1 -regime model (in the parameter ordering used by the corresponding model constructor). Starting vectors containing non-finite values are dropped; the list may be empty if the null fit is degenerate.

References

- Kasahara, H. & Shimotsu, K. 2018. "Testing the number of regimes in Markov regime switching models." arXiv:1801.06862.
- Dufour, J.-M. 2006. "Monte Carlo tests with nuisance parameters: A general approach to finite-sample inference and nonstandard asymptotics." *Journal of Econometrics*, 133(2), 443-477.
- Rodriguez-Rondon, G. & Dufour, J.-M. 2026. "Monte Carlo Likelihood-Ratio Tests for Markov Switching Models." Bank of Canada Staff Working Paper 2026-23.

Index

- * **datasets**
 - chp10GNP, [15](#)
 - hamilton84GNP, [22](#)
 - USGNP, [76](#)
 - USRGDP, [77](#)
- * **package**
 - MSTest-package, [3](#)
- aBfixLMCTest, [4](#)
- ARmdl, [5](#), [16–18](#), [20](#), [24](#), [31](#), [38](#), [46](#), [52](#)
- ARXmdl, [7](#)
- aURdiffLMCTest, [10](#), [11](#), [13](#)
- aURdiffMMCTest, [11](#)
- aURfixLMCTest, [4](#), [5](#), [12](#), [36](#)
- canonicalize_regimes, [4](#), [14](#), [37](#), [40](#)
- chp10GNP, [15](#)
- CHPbootCV, [15](#)
- CHPTest, [16](#)
- DLMCTest, [18](#)
- DLMCTest, [19](#)
- estimMdl, [82](#)
- GA, [20](#)
- ga, [38](#)
- GenSA, [20](#), [38](#)
- hamilton84GNP, [22](#)
- hessian, [7](#), [9](#), [29](#), [45](#), [51](#), [56](#), [61](#), [64](#), [79](#), [81](#)
- HLRTest, [23](#)
- HMmdl, [25](#), [31](#), [38](#)
- initVals_MSARmdl, [25](#), [42](#), [47](#), [53](#), [58](#)
- LMCLRTTest, [10](#), [11](#), [13](#), [30](#), [82](#)
- MCpval, [32](#), [33](#), [40](#)
- MMCLRTTest, [4](#), [11](#), [12](#), [34](#), [82](#)
- MSARmdl, [7](#), [9](#), [31](#), [38](#), [41](#)
- MSARXmdl, [47](#)
- MSTest (MSTest-package), [3](#)
- MSTest-package, [3](#)
- MSVARmdl, [31](#), [38](#), [52](#), [79](#), [81](#)
- MSVARXmdl, [57](#)
- nearest_spd, [7](#), [9](#), [29](#), [45](#), [51](#), [56](#), [61](#), [64](#), [79](#), [81](#)
- Nmdl, [18](#), [20](#), [29](#), [31](#), [38](#), [63](#)
- optim, [24](#)
- pso, [20](#)
- psoptim, [38](#)
- simuAR, [65](#)
- simuARX, [66](#)
- simuHMM, [67](#)
- simuMSAR, [68](#)
- simuMSARX, [69](#)
- simuMSVAR, [70](#)
- simuMSVARX, [71](#)
- simuNorm, [73](#)
- simuVAR, [74](#)
- simuVARX, [75](#)
- slsqp, [24](#)
- USGNP, [76](#)
- USRGDP, [77](#)
- VARmdl, [31](#), [38](#), [57](#), [62](#), [77](#)
- VARXmdl, [79](#)
- warmstart_theta, [26](#), [31](#), [35](#), [42](#), [47](#), [53](#), [58](#), [82](#)