# Package ‘mtarm’

June 12, 2026

**Type** Package

**Title** Bayesian Estimation of Multivariate Threshold Autoregressive Models

**Version** 0.1.9

**Author** Luis Hernando Vanegas [aut, cre],
Sergio Alejandro Calderón [aut],
Luz Marina Rondón [aut]

**Maintainer** Luis Hernando Vanegas <lhvanegasp@unal.edu.co>

**Description** Estimation, inference and forecasting using the Bayesian approach for multivariate threshold autoregressive (TAR) models in which the distribution used to describe the noise process belongs to the class of Gaussian variance mixtures.

**Imports** methods, stats, utils, graphics, Formula, grDevices, GIGrvg, coda, mvtnorm, future.apply, progressr, future

**Suggests** cli, knitr, rmarkdown, testthat (>= 3.0.0)

**BugReports** https://github.com/lhvanegasp/mtarm/issues

**Encoding** UTF-8

**LazyData** false

**RoxygenNote** 7.3.3

**NeedsCompilation** no

**License** GPL-2 | GPL-3

**Config/testthat/edition** 3

**URL** https://lhvanegasp.github.io/mtarm/

**VignetteBuilder** knitr

**Repository** CRAN

**Date/Publication** 2026-06-12 15:10:02 UTC

# Contents



---

`ars` *Auxiliary function to specify the number of regimes and lag orders*

---

## Description

This auxiliary function defines the regime structure of a multivariate TAR model by specifying the number of regimes and the corresponding lag orders for the endogenous, exogenous, and threshold series in each regime.

## Usage

```
ars(nregim = 1, p = 1, q = 0, d = 0)
```

## Arguments

| | |
|---|---|
| `nregim` | A positive integer indicating the total number of regimes. |
| `p` | A list of positive integers specifying the autoregressive order of the output series within each regime. |

| | |
|---|---|
| q | A list of non-negative integers specifying the maximum lag of the exogenous series within each regime. |
| d | A list of non-negative integers specifying the maximum lag of the threshold series within each regime. |

### Value

A list containing the number of regimes and the regime-specific lag-order specifications.

---

`as.mcmc.mtar` *Coercion of* `mtar` *objects to* `mcmc` *objects*

---

### Description

This method converts an object of class `mtar` into a list of `mcmc` objects, each corresponding to a Markov chain produced during Bayesian estimation.

### Usage

```
## S3 method for class 'mtar'
as.mcmc(x, ...)
```

### Arguments

| | |
|---|---|
| x | an object of class `mtar` obtained from a call to `mtar()`. |
| ... | additional arguments passed to specific coercion methods. |

### Value

A list of `mcmc` objects containing the posterior simulation draws generated by the `mtar()` routine.

### See Also

`as.mcmc`

### Examples

```
###### Example 1: Returns of the closing prices of three financial indexes
data(returns)
fit1 <- mtar(~ COLCAP + BOVESPA | SP500, data=returns, row.names=Date,
             subset={Date<="2015-12-07"}, dist="Student-t",
             ars=ars(nregim=3,p=c(1,1,2)), n.burnin=1000, n.sim=2000,
             n.thin=2, ssvs=TRUE)
fit1.mcmc <- coda::as.mcmc(fit1)
summary(fit1.mcmc)
#plot(fit1.mcmc)

###### Example 2: Rainfall and two river flows in Colombia
```

```
data(riverflows)
fit2 <- mtar(~ Bedon + LaPlata | Rainfall, data=riverflows, row.names=Date,
             subset={Date<="2009-02-13"}, dist="Laplace",
             ars=ars(nregim=3,p=5), n.burnin=1000, n.sim=2000, n.thin=2)
fit2.mcmc <- coda::as.mcmc(fit2)
summary(fit2.mcmc)
#plot(fit2.mcmc)

###### Example 3: Temperature, precipitation, and two river flows in Iceland
data(iceland.rf)
fit3 <- mtar(~ Jokulsa + Vatnsdalsa | Temperature | Precipitation,
             data=iceland.rf, subset={Date<="1974-11-06"}, row.names=Date,
             ars=ars(nregim=2,p=15,q=4,d=2), n.burnin=1000, n.sim=2000,
             n.thin=2, dist="Slash")
fit3.mcmc <- coda::as.mcmc(fit3)
summary(fit3.mcmc)
#plot(fit3.mcmc)

###### Example 4: U.S. stock returns
data(US.returns)
fit4 <- mtar(~ CCR | dVIX, data=US.returns, subset={Date<="2025-11-28"},
             row.names=Date, ars=ars(nregim=2,p=3,d=3), n.burnin=1000,
             n.sim=2000, n.thin=2, dist="Student-t")
fit4.mcmc <- coda::as.mcmc(fit4)
summary(fit4.mcmc)
#plot(fit4.mcmc)
```

---

`coef.mtar` *coef method for objects of class* `mtar`

---

## Description

coef method for objects of class `mtar`

## Usage

```
## S3 method for class 'mtar'
coef(object, ..., FUN = mean)
```

## Arguments

| | |
|---|---|
| `object` | an object of class `mtar` obtained from a call to the `mtar()` function. |
| `...` | additional arguments passed to `FUN`. |
| `FUN` | a function to be applied to the MCMC chains associated with each model parameter. By default, `FUN` is set to `mean`. |

### Value

A list containing the summary statistics obtained by applying `FUN` to the MCMC chains of each model parameter.

---

`DIC` *Deviance Information Criterion (DIC)*

---

### Description

Computes the Deviance Information Criterion (DIC), an adjusted within-sample measure of predictive accuracy, for models estimated using Bayesian methods.

### Usage

```
DIC(...)
```

### Arguments

| | |
|---|---|
| `...` | one or more fitted model objects of the same class. |

### Value

A numeric matrix containing the DIC values corresponding to each fitted object supplied in `...`.

### References

Spiegelhalter D.J., Best N.G., Carlin B.P. and Van Der Linde A. (2002) Bayesian Measures of Model Complexity and Fit. Journal of the Royal Statistical Society Series B (Statistical Methodology), 64(4), 583–639.

Spiegelhalter D.J., Best N.G., Carlin B.P. and Van der Linde A. (2014). The deviance information criterion: 12 years on. Journal of the Royal Statistical Society Series B (Statistical Methodology), 76(3), 485–493.

---

`DIC.mtar` *Deviance Information Criterion (DIC) for objects of class* `mtar`

---

### Description

This function computes the Deviance Information Criterion (DIC) for objects of class `mtar`.

### Usage

```
## S3 method for class 'mtar'
DIC(...)
```

### Arguments

... one or several objects of the class *mtar*.

### Value

A numeric matrix containing the DIC values corresponding to each *mtar* object in the input.

### See Also

WAIC

### Examples

```
###### Example 1: Returns of the closing prices of three financial indexes
data(returns)
fit1 <- mtar_grid(~ COLCAP + BOVESPA | SP500, data=returns, row.names=Date,
                  subset={Date<="2015-12-07"}, dist=c("Gaussian","Student-t",
                  "Slash","Laplace"), nregim.min=2, nregim.max=3, p.min=2,
                  p.max=2, n.burnin=1000, n.sim=2000, n.thin=2,
                  plan_strategy="multisession")
DIC(fit1)

###### Example 2: Rainfall and two river flows in Colombia
data(riverflows)
fit2 <- mtar_grid(~ Bedon + LaPlata | Rainfall, data=riverflows,
                  row.names=Date, subset={Date<="2009-02-13"},dist="Laplace",
                  nregim.min=2, nregim.max=3, p.min=1, p.max=3,n.burnin=1000,
                  n.sim=2000, n.thin=2, plan_strategy="multisession")
DIC(fit2)

###### Example 3: Temperature, precipitation, and two river flows in Iceland
data(iceland.rf)
fit3 <- mtar_grid(~ Jokulsa + Vatnsdalsa | Temperature | Precipitation,
                  data=iceland.rf,subset={Date<="1974-11-06"},row.names=Date,
                  dist=c("Slash","Student-t"), nregim.min=1, nregim.max=2,
                  p.min=15, p.max=15, q.min=4, q.max=4, d.min=2, d.max=2,
                  n.burnin=1000, n.sim=2000, n.thin=2,
                  plan_strategy="multisession")
DIC(fit3)

###### Example 4: U.S. stock returns
data(US.returns)
fit4 <- mtar_grid(~ CCR | dVIX, data=US.returns, subset={Date<="2025-11-28"},
                  row.names=Date, dist=c("Laplace","Student-t","Slash"),
                  nregim.min=2, nregim.max=2, p.min=3, p.max=3, d.min=3,
                  d.max=3, n.burnin=1000, n.sim=2000, n.thin=2,
                  plan_strategy="multisession")
DIC(fit4)
```

## effectiveSize_TAR    *Effective sample size for* `mtar` *objects*

### Description

This function computes the effective sample size, adjusted for autocorrelation, of Markov chain Monte Carlo (MCMC) output obtained from the Bayesian estimation of multivariate TAR models. It serves as a wrapper around `effectiveSize()`, applying this function to the posterior chains returned by `mtar()`.

### Usage

```
effectiveSize_TAR(x)
```

### Arguments

| | |
|---|---|
| x | An object of class `mtar` produced by `mtar()`. |

### Value

A list with the effective sample sizes for each parameter of the `mtar` model.

### See Also

`effectiveSize`

### Examples

```
###### Example 1: Returns of the closing prices of three financial indexes
data(returns)
fit1 <- mtar(~ COLCAP + BOVESPA | SP500, data=returns, row.names=Date,
             subset={Date<="2015-12-07"}, dist="Student-t",
             ars=ars(nregim=3,p=c(1,1,2)), n.burnin=1000, n.sim=2000,
             n.thin=2)
effectiveSize_TAR(fit1)

###### Example 2: Rainfall and two river flows in Colombia
data(riverflows)
fit2 <- mtar(~ Bedon + LaPlata | Rainfall, data=riverflows, row.names=Date,
             subset={Date<="2009-02-13"}, dist="Laplace",
             ars=ars(nregim=3,p=5), n.burnin=1000, n.sim=2000, n.thin=2)
effectiveSize_TAR(fit2)

###### Example 3: Temperature, precipitation, and two river flows in Iceland
data(iceland.rf)
fit3 <- mtar(~ Jokulsa + Vatnsdalsa | Temperature | Precipitation,
             data=iceland.rf, subset={Date<="1974-11-06"}, row.names=Date,
             ars=ars(nregim=2,p=15,q=4,d=2), n.burnin=1000, n.sim=2000,
             n.thin=2, dist="Slash")
```

```
effectiveSize_TAR(fit3)

###### Example 4: U.S. stock returns
data(US.returns)
fit4 <- mtar(~ CCR | dVIX, data=US.returns, subset={Date<="2025-11-28"},
             row.names=Date, ars=ars(nregim=2,p=3,d=3), n.burnin=1000,
             n.sim=2000, n.thin=2, dist="Student-t")
effectiveSize_TAR(fit4)
```

---

geweke_diagTAR *Geweke's convergence diagnostic for* `mtar` *objects*

---

## Description

This function computes Geweke's convergence diagnostic for Markov chain Monte Carlo (MCMC) output obtained from Bayesian estimation of multivariate TAR models. It is a wrapper around `geweke.diag()` that applies the diagnostic to the posterior chains returned by a call to `mtar()`.

## Usage

```
geweke_diagTAR(x, frac1 = 0.1, frac2 = 0.5)
```

## Arguments

| | |
|---|---|
| `x` | An object of class `mtar` returned by the function `mtar()`. |
| `frac1` | A numeric value in $(0, 1)$ specifying the fraction of the initial part of each chain to be used in the diagnostic. |
| `frac2` | A numeric value in $(0, 1)$ specifying the fraction of the final part of each chain to be used in the diagnostic. |

## Value

A list containing the Geweke z-scores for the parameters of the `mtar` model.

## See Also

`geweke.diag`

## Examples

```
###### Example 1: Returns of the closing prices of three financial indexes
data(returns)
fit1 <- mtar(~ COLCAP + BOVESPA | SP500, data=returns, row.names=Date,
             subset={Date<="2015-12-07"}, dist="Student-t",
             ars=ars(nregim=3,p=c(1,1,2)), n.burnin=1000, n.sim=2000,
             n.thin=2)
geweke_diagTAR(fit1)

###### Example 2: Rainfall and two river flows in Colombia
data(riverflows)
fit2 <- mtar(~ Bedon + LaPlata | Rainfall, data=riverflows, row.names=Date,
             subset={Date<="2009-02-13"}, dist="Laplace",
             ars=ars(nregim=3,p=5), n.burnin=1000, n.sim=2000, n.thin=2)
geweke_diagTAR(fit2)

###### Example 3: Temperature, precipitation, and two river flows in Iceland
data(iceland.rf)
fit3 <- mtar(~ Jokulsa + Vatnsdalsa | Temperature | Precipitation,
             data=iceland.rf, subset={Date<="1974-11-06"}, row.names=Date,
             ars=ars(nregim=2,p=15,q=4,d=2), n.burnin=1000, n.sim=2000,
             n.thin=2, dist="Slash")
geweke_diagTAR(fit3)

###### Example 4: U.S. stock returns
data(US.returns)
fit4 <- mtar(~ CCR | dVIX, data=US.returns, subset={Date<="2025-11-28"},
             row.names=Date, ars=ars(nregim=2,p=3,d=3), n.burnin=1000,
             n.sim=2000, n.thin=2, dist="Student-t")
geweke_diagTAR(fit4)
```

---

`geweke_plotTAR` *Geweke-Brooks plot for objects of class* `mtar`

---

## Description

This function is a wrapper around `geweke.plot()` that applies the Geweke-Brooks convergence diagnostic to the MCMC chains obtained from a fitted `mtar` model.

## Usage

```
geweke_plotTAR(
  x,
  frac1 = 0.1,
  frac2 = 0.5,
  nbins = 20,
```

```
  pvalue = 0.05,
  auto.layout = TRUE,
  ask,
  ...
)
```

## Arguments

| | |
|---|---|
| x | An object of class `mtar` returned by a call to `mtar()`. |
| frac1 | fraction to use from beginning of chain |
| frac2 | fraction to use from end of chain |
| nbins | Number of segments |
| pvalue | p-value used to plot confidence limits for the null hypothesis |
| auto.layout | If `TRUE` then, set up own layout for plots, otherwise use existing one |
| ask | If `TRUE` then prompt user before displaying each page of plots. Default is `dev.interactive()`. |
| ... | Additional graphical parameters passed to the plotting routines. |

## See Also

geweke.plot

## Examples

```
###### Example 1: Returns of the closing prices of three financial indexes
data(returns)
fit1 <- mtar(~ COLCAP + BOVESPA | SP500, data=returns, row.names=Date,
             subset={Date<="2015-12-07"}, dist="Student-t",
             ars=ars(nregim=3,p=c(1,1,2)), n.burnin=1000, n.sim=2000,
             n.thin=2)
geweke_plotTAR(fit1)

###### Example 2: Rainfall and two river flows in Colombia
data(riverflows)
fit2 <- mtar(~ Bedon + LaPlata | Rainfall, data=riverflows, row.names=Date,
             subset={Date<="2009-02-13"}, dist="Laplace",
             ars=ars(nregim=3,p=5), n.burnin=1000, n.sim=2000, n.thin=2)
geweke_plotTAR(fit2)

###### Example 3: Temperature, precipitation, and two river flows in Iceland
data(iceland.rf)
fit3 <- mtar(~ Jokulsa + Vatnsdalsa | Temperature | Precipitation,
             data=iceland.rf, subset={Date<="1974-11-06"}, row.names=Date,
             ars=ars(nregim=2,p=15,q=4,d=2), n.burnin=1000, n.sim=2000,
             n.thin=2, dist="Slash")
geweke_plotTAR(fit3)

###### Example 4: U.S. stock returns
data(US.returns)
fit4 <- mtar(~ CCR | dVIX, data=US.returns, subset={Date<="2025-11-28"},
```

```
             row.names=Date, ars=ars(nregim=2,p=3,d=3), n.burnin=1000,
             n.sim=2000, n.thin=2, dist="Student-t")
geweke_plotTAR(fit4)
```

---

HPDinterval.mtar *Highest Posterior Density intervals for objects of class* mtar

---

## Description

Highest Posterior Density intervals for objects of class mtar

## Usage

```
## S3 method for class 'mtar'
HPDinterval(obj, prob = 0.95, ...)
```

## Arguments

| | |
|---|---|
| obj | an object of class mtar generated by a call to the function mtar(). |
| prob | a numeric scalar in the interval $(0, 1)$ giving the target probability content of the intervals. By default, prob is set to 0.95. |
| ... | Optional additional arguments for methods. None are used at present. |

## See Also

HPDinterval

## Examples

```
###### Example 1: Returns of the closing prices of three financial indexes
data(returns)
fit1 <- mtar(~ COLCAP + BOVESPA | SP500, data=returns, row.names=Date,
             subset={Date<="2015-12-07"}, dist="Student-t",
             ars=ars(nregim=3,p=c(1,1,2)), n.burnin=1000, n.sim=2000,
             n.thin=2, ssvs=TRUE)
coda::HPDinterval(fit1)

###### Example 2: Rainfall and two river flows in Colombia
data(riverflows)
fit2 <- mtar(~ Bedon + LaPlata | Rainfall, data=riverflows, row.names=Date,
             subset={Date<="2009-02-13"}, dist="Laplace",
             ars=ars(nregim=3,p=5), n.burnin=1000, n.sim=2000, n.thin=2)
coda::HPDinterval(fit2)

###### Example 3: Temperature, precipitation, and two river flows in Iceland
data(iceland.rf)
fit3 <- mtar(~ Jokulsa + Vatnsdalsa | Temperature | Precipitation,
```

```
             data=iceland.rf, subset={Date<="1974-11-06"}, row.names=Date,
             ars=ars(nregim=2,p=15,q=4,d=2), n.burnin=1000, n.sim=2000,
             n.thin=2, dist="Slash")
coda::HPDinterval(fit3)

###### Example 4: U.S. stock returns
data(US.returns)
fit4 <- mtar(~ CCR | dVIX, data=US.returns, subset={Date<="2025-11-28"},
             row.names=Date, ars=ars(nregim=2,p=3,d=3), n.burnin=1000,
             n.sim=2000, n.thin=2, dist="Student-t")
coda::HPDinterval(fit4)
```

---

`iceland.rf` *Temperature, precipitation, and two river flows in Iceland*

---

## Description

This data set contains two daily river-flow time series, measured in cubic meters per second, for rivers in Iceland from January 1, 1972, to December 12, 1974. Additionally, daily measurements of precipitation (in millimeters) and temperature (in degrees Celsius) were recorded at the Hveravellir meteorological station. The precipitation values correspond to the accumulated precipitation up to 9:00 A.M. relative to the same time on the previous day.

## Usage

```
data(iceland.rf)
```

## Format

A data frame with 1,096 rows and 5 variables:

**Vatnsdalsa** Numeric vector representing the daily flow of the Vatnsdalsá river.

**Jokulsa** Numeric vector representing the daily flow of the Jökulsá Eystri river.

**Precipitation** Numeric vector of daily precipitation amounts (millimeters).

**Temperature** Numeric vector of daily temperature measurements (degrees Celsius).

**Date** Vector indicating the calendar date of each observation.

## References

Tong, Howell (1990) Non-linear Time Series: A Dynamical System Approach. Oxford University Press. Oxford, UK.

Ruey S., Tsay (1998) Testing and Modeling Multivariate Threshold Models. Journal of the American Statistical Association, 93, 1188-1202.

## Examples

```
data(iceland.rf)
dev.new()
plot(ts(as.matrix(iceland.rf[,-5])), main="Iceland")
```

---

mtar — *Bayesian estimation of a multivariate Threshold Autoregressive (TAR) model.*

---

## Description

This function implements a Gibbs sampling algorithm to generate draws from the posterior distribution of the parameters of a multivariate Threshold Autoregressive (TAR) model, including special cases such as SETAR and VAR models. The approach accommodates a wide range of noise process distributions, including Gaussian, Student-$t$, Slash, symmetric hyperbolic, contaminated normal, Laplace, skew-normal, and skew-Student-$t$.

## Usage

```
mtar(
  formula,
  data,
  subset,
  Intercept = TRUE,
  trend = c("none", "linear", "quadratic"),
  nseason = NULL,
  ars = ars(),
  row.names,
  dist = c("Gaussian", "Student-t", "Hyperbolic", "Laplace", "Slash",
    "Contaminated normal", "Skew-Student-t", "Skew-normal"),
  prior = list(),
  n.sim = 500,
  n.burnin = 100,
  n.thin = 1,
  ssvs = FALSE,
  setar = NULL,
  progress = TRUE,
  ...
)
```

## Arguments

`formula` A three-part expression of class `Formula` describing the TAR model to be fitted. The first part specifies the variables in the multivariate output series, the second part defines the threshold series, and the third part specifies the variables in the multivariate exogenous series.

| | |
|---|---|
| data | A data frame containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which mtar_grid() is called. |
| subset | An optional vector specifying a subset of observations to be used in the fitting process. |
| Intercept | An optional logical indicating whether an intercept should be included within each regime. |
| trend | An optional character string specifying the degree of deterministic time trend to be included in each regime. Available options are "linear", "quadratic", and "none". By default, trend is set to "none". |
| nseason | An optional integer, greater than or equal to 2, specifying the number of seasonal periods. When provided, nseason - 1 seasonal dummy variables are added to the regressors within each regime. By default, nseason is set to NULL, thereby indicating that the TAR model has no seasonal effects. |
| ars | A list defining the autoregressive structure of the model. It contains four components: the number of regimes (nregim), the autoregressive order within each regime (p), and the maximum lags for the exogenous (q) and threshold (d) series in each regime. The object can be validated using the helper function ars(). |
| row.names | An optional variable in data labelling the time points corresponding to each row of the data set. |
| dist | A character string specifying the multivariate distributions used to model the noise process. Available options are "Gaussian", "Student-t", "Slash", "Hyperbolic", "Laplace", "Contaminated normal", "Skew-normal", and "Skew-Student-t". By default, dist is set to "Gaussian". |
| prior | An optional list specifying the hyperparameter values that define the prior distribution. This list can be validated using the priors() function. By default, prior is set to an empty list, thereby indicating that the hyperparameter values should be set so that a non-informative prior distribution is obtained. |
| n.sim | An optional positive integer specifying the number of simulation iterations after the burn-in period. By default, n.sim is set to 500. |
| n.burnin | An optional positive integer specifying the number of burn-in iterations. By default, n.burnin is set to 100. |
| n.thin | An optional positive integer specifying the thinning interval. By default, n.thin is set to 1. |
| ssvs | An optional logical indicating whether the Stochastic Search Variable Selection (SSVS) procedure should be applied to identify relevant lags of the output, exogenous, and threshold series. By default, ssvs is set to FALSE. |
| setar | An optional positive integer indicating the component of the output series used as the threshold variable. By default, setar is set to NULL, indicating that the fitted model is not a SETAR model. |
| progress | An optional logical indicating whether a progress bar should be displayed during execution. By default, progress is set to TRUE. |
| ... | further arguments passed to or from other methods. |

### Value

an object of class *mtar* in which the main results of the model fitted to the data are stored, i.e., a list with components including

| | |
|---|---|
| `chains` | list with several arrays, which store the values of each model parameter in each iteration of the simulati |
| `n.sim` | number of iterations of the simulation after the burn-in period, |
| `n.burnin` | number of burn-in iterations in the simulation, |
| `n.thin` | thinning interval in the simulation, |
| `ars` | list composed of four objects, namely: `nregim`, `p`, `q` and `d`, each of which corresponds to a vector of no |
| `dist` | name of the multivariate distribution used to describe the behavior of the noise process, |
| `threshold.series` | vector with the values of the threshold series, |
| `output.series` | matrix with the values of the output series, |
| `exogenous.series` | matrix with the values of the exogenous series, |
| `Intercept` | If `TRUE`, then the model included an intercept term in each regime, |
| `trend` | the degree of the deterministic time trend, if any, |
| `nseason` | the number of seasonal periods, if any, |
| `formula` | the formula, |
| `call` | the original function call. |

### References

Nieto, F.H. (2005) Modeling Bivariate Threshold Autoregressive Processes in the Presence of Missing Data. Communications in Statistics - Theory and Methods, 34, 905-930.

Romero, L.V. and Calderon, S.A. (2021) Bayesian estimation of a multivariate TAR model when the noise process follows a Student-t distribution. Communications in Statistics - Theory and Methods, 50, 2508-2530.

Calderon, S.A. and Nieto, F.H. (2017) Bayesian analysis of multivariate threshold autoregressive models with missing data. Communications in Statistics - Theory and Methods, 46, 296-318.

Vanegas, L.H. and Calderón, S.A. and Rondón, L.M. (2025) Bayesian estimation of a multivariate tar model when the noise process distribution belongs to the class of gaussian variance mixtures. International Journal of Forecasting.

## See Also

DIC, WAIC

## Examples

```
###### Example 1: Returns of the closing prices of three financial indexes
data(returns)
fit1 <- mtar(~ COLCAP + BOVESPA | SP500, data=returns, row.names=Date,
             subset={Date<="2015-12-07"}, dist="Student-t",
             ars=ars(nregim=3,p=c(1,1,2)), n.burnin=1000, n.sim=2000,
             n.thin=2)
summary(fit1)
DIC(fit1)
WAIC(fit1)

###### Example 2: Rainfall and two river flows in Colombia
data(riverflows)
fit2 <- mtar(~ Bedon + LaPlata | Rainfall, data=riverflows, row.names=Date,
             subset={Date<="2009-02-13"}, dist="Laplace",
             ars=ars(nregim=3,p=5), n.burnin=1000, n.sim=2000, n.thin=2)
summary(fit2)
DIC(fit2)
WAIC(fit2)

###### Example 3: Temperature, precipitation, and two river flows in Iceland
data(iceland.rf)
fit3 <- mtar(~ Jokulsa + Vatnsdalsa | Temperature | Precipitation,
             data=iceland.rf, subset={Date<="1974-11-06"}, row.names=Date,
             ars=ars(nregim=2,p=15,q=4,d=2), n.burnin=1000, n.sim=2000,
             n.thin=2, dist="Slash")
summary(fit3)
DIC(fit3)
WAIC(fit3)

###### Example 4: U.S. stock returns
data(US.returns)
fit4 <- mtar(~ CCR | dVIX, data=US.returns, subset={Date<="2025-11-28"},
             row.names=Date, ars=ars(nregim=2,p=3,d=3), n.burnin=1000,
             n.sim=2000, n.thin=2, dist="Student-t")
summary(fit4)
DIC(fit4)
WAIC(fit4)
```

---

mtar_grid *Bayesian Estimation of Multivariate TAR Models*

---

## Description

This function is a wrapper that applies `mtar()` over a grid of model specifications defined by all combinations of the noise distribution (`dist`), the number of regimes (from `nregim.min` to `nregim.max`), the autoregressive order within each regime (from `p.min` to `p.max`), the maximum lag of the exogenous series within each regime (from `q.min` to `q.max`), and the maximum lag of the threshold series within each regime (from `d.min` to `d.max`). In all calls to `mtar()`, the same set of time points is used for model fitting. This is achieved by appropriately adjusting the `subset` argument of `mtar()` for each model specification, thereby ensuring comparability across models.

## Usage

```
mtar_grid(
  formula,
  data,
  subset,
  Intercept = TRUE,
  trend = c("none", "linear", "quadratic"),
  nseason = NULL,
  nregim.min = 1,
  nregim.max = NULL,
  p.min = 1,
  p.max = NULL,
  q.min = 0,
  q.max = 0,
  d.min = 0,
  d.max = 0,
  row.names,
  dist = "Gaussian",
  prior = list(),
  n.sim = 500,
  n.burnin = 100,
  n.thin = 1,
  ssvs = FALSE,
  setar = NULL,
  plan_strategy = c("multisession", "sequential"),
  progress = TRUE
)
```

## Arguments

`formula` A three-part expression of class `Formula` describing the TAR model to be fitted. The first part specifies the variables in the multivariate output series, the second part defines the threshold series, and the third part specifies the variables in the multivariate exogenous series.

`data` A data frame containing the variables in the model. If not found in `data`, the variables are taken from `environment(formula)`, typically the environment from which `mtar_grid()` is called.

| | |
|---|---|
| subset | An optional vector specifying a subset of observations to be used in the fitting process. |
| Intercept | An optional logical indicating whether an intercept should be included within each regime. |
| trend | An optional character string specifying the degree of deterministic time trend to be included in each regime. Available options are "linear", "quadratic", and "none". By default, trend is set to "none". |
| nseason | An optional integer, greater than or equal to 2, specifying the number of seasonal periods. When provided, nseason - 1 seasonal dummy variables are added to the regressors within each regime. By default, nseason is set to NULL, thereby indicating that the TAR model has no seasonal effects. |
| nregim.min | An optional integer specifying the minimum number of regimes. By default, nregim.min is set to 1. |
| nregim.max | An integer specifying the maximum number of regimes. |
| p.min | An optional integer specifying the minimum autoregressive order within each regime. By default, p.min is set to 1. |
| p.max | An integer specifying the maximum autoregressive order within each regime. |
| q.min | An optional integer specifying the minimum value of the maximum lag of the exogenous series within each regime. By default, q.min is set to 0. |
| q.max | An optional integer specifying the maximum value of the maximum lag of the exogenous series within each regime. By default, q.max is set to 0. |
| d.min | An optional integer specifying the minimum value of the maximum lag of the threshold series within each regime. By default, d.min is set to 0. |
| d.max | An optional integer specifying the maximum value of the maximum lag of the threshold series within each regime. By default, d.max is set to 0. |
| row.names | An optional variable in data labelling the time points corresponding to each row of the data set. |
| dist | A character vector specifying the multivariate distributions used to model the noise process. Available options are "Gaussian", "Student-t", "Slash", "Hyperbolic", "Laplace", "Contaminated normal", "Skew-normal", and "Skew-Student-t". By default, dist is set to "Gaussian". |
| prior | An optional list specifying the hyperparameter values that define the prior distribution. This list can be validated using the priors() function. By default, prior is set to an empty list, thereby indicating that the hyperparameter values should be set so that a non-informative prior distribution is obtained. |
| n.sim | An optional positive integer specifying the number of simulation iterations after the burn-in period. By default, n.sim is set to 500. |
| n.burnin | An optional positive integer specifying the number of burn-in iterations. By default, n.burnin is set to 100. |
| n.thin | An optional positive integer specifying the thinning interval. By default, n.thin is set to 1. |
| ssvs | An optional logical indicating whether the Stochastic Search Variable Selection (SSVS) procedure should be applied to identify relevant lags of the output, exogenous, and threshold series. By default, ssvs is set to FALSE. |

| | |
|---|---|
| setar | An optional positive integer indicating the component of the output series used as the threshold variable. By default, setar is set to NULL, indicating that the fitted model is not a SETAR model. |
| plan_strategy | An optional character string specifying the execution strategy for parallel computation. Available options are "sequential" and "multisession". By default, plan_strategy is set to "sequential". |
| progress | An optional logical indicating whether a progress bar should be displayed during execution. By default, progress is set to TRUE. |

## Value

A list whose elements are objects of class mtar, each corresponding to a distinct model specification considered in the grid.

## See Also

mtar

## Examples

```
###### Example 1: Returns of the closing prices of three financial indexes
data(returns)
fit1 <- mtar_grid(~ COLCAP + BOVESPA | SP500, data=returns, row.names=Date,
                  subset={Date<="2015-12-07"}, dist=c("Gaussian","Student-t",
                  "Slash","Laplace"), nregim.min=2, nregim.max=3, p.min=2,
                  p.max=2, n.burnin=1000, n.sim=2000, n.thin=2,
                  plan_strategy="multisession")
summary(fit1)

###### Example 2: Rainfall and two river flows in Colombia
data(riverflows)
fit2 <- mtar_grid(~ Bedon + LaPlata | Rainfall, data=riverflows,
                  row.names=Date, subset={Date<="2009-02-13"},dist="Laplace",
                  nregim.min=2, nregim.max=3, p.min=1, p.max=3,n.burnin=1000,
                  n.sim=2000, n.thin=2, plan_strategy="multisession")
fit2

###### Example 3: Temperature, precipitation, and two river flows in Iceland
data(iceland.rf)
fit3 <- mtar_grid(~ Jokulsa + Vatnsdalsa | Temperature | Precipitation,
                  data=iceland.rf,subset={Date<="1974-11-06"},row.names=Date,
                  dist=c("Slash","Student-t"), nregim.min=1, nregim.max=2,
                  p.min=15, p.max=15, q.min=4, q.max=4, d.min=2, d.max=2,
                  n.burnin=1000, n.sim=2000, n.thin=2,
                  plan_strategy="multisession")
fit3

###### Example 4: U.S. stock returns
data(US.returns)
fit4 <- mtar_grid(~ CCR | dVIX, data=US.returns, subset={Date<="2025-11-28"},
                  row.names=Date, dist=c("Laplace","Student-t","Slash"),
```

```
                 nregim.min=2, nregim.max=2, p.min=3, p.max=3, d.min=3,
                 d.max=3, n.burnin=1000, n.sim=2000, n.thin=2,
                 plan_strategy="multisession")
fit4
```

---

`out_of_sample` *Out-of-sample predictive accuracy measures*

---

## Description

Computes out-of-sample predictive accuracy measures for one or more fitted models of the same class, based on their predictive distributions.

## Usage

```
out_of_sample(..., newdata, n.ahead)
```

## Arguments

| | |
|---|---|
| `...` | one or more fitted model objects of the same class. |
| `newdata` | a data frame containing the future values of the output series required to evaluate predictive performance. |
| `n.ahead` | a positive integer specifying the number of forecast steps ahead to use in the predictive performance evaluation. |

---

`out_of_sample.listmtar`
*Computing Out-of-Sample predictive accuracy measures*

---

## Description

Computes Out-of-Sample predictive accuracy measures for an object of class `listmtar`.

## Usage

```
## S3 method for class 'listmtar'
out_of_sample(
  x,
  newdata,
  n.ahead = NULL,
  credible = 0.95,
  FUN = mean,
  rolling = NULL,
  ...
)
```

### Arguments

x
: An object of class `listmtar` returned by the routine `mtar_grid()`.

newdata
: A `data.frame` containing future values of the threshold series (if included in the fitted model), the exogenous series (if included in the fitted model), and the realized values of the output series.

n.ahead
: A positive integer specifying the number of steps ahead to forecast.

credible
: An optional numeric value in $(0, 1)$ specifying the level of the required prediction intervals. By default, `credible` is set to `0.95`.

FUN
: An optional function used to summarize the `n.ahead` values computed for each predictive accuracy measure. By default, `FUN` is set to `mean`.

rolling
: An optional positive integer specifying the rolling-window size used for forecasting. By default, `rolling = NULL`, indicating that rolling-window forecasting is not performed.

...
: optional arguments to FUN.

### Examples

```
###### Example 1: Returns of the closing prices of three financial indexes
data(returns)
fit1 <- mtar_grid(~ COLCAP + BOVESPA | SP500, data=returns, row.names=Date,
                  subset={Date<="2015-12-07"}, dist=c("Gaussian","Student-t",
                  "Slash","Laplace"), nregim.min=2, nregim.max=3, p.min=2,
                  p.max=2, n.burnin=1000, n.sim=2000, n.thin=2,
                  plan_strategy="multisession")
oos1 <- out_of_sample(fit1, newdata=subset(returns, Date>"2015-12-07"),
                      n.ahead=75, FUN=median)
oos1

###### Example 2: Rainfall and two river flows in Colombia
data(riverflows)
fit2 <- mtar_grid(~ Bedon + LaPlata | Rainfall, data=riverflows,
                  row.names=Date, subset={Date<="2009-02-13"},dist="Laplace",
                  nregim.min=2, nregim.max=3, p.min=1, p.max=3,n.burnin=1000,
                  n.sim=2000, n.thin=2, plan_strategy="multisession")
oos2 <- out_of_sample(fit2, newdata=subset(riverflows, Date>"2009-02-13"),
                      n.ahead=60, FUN=median)
oos2

###### Example 3: Temperature, precipitation, and two river flows in Iceland
data(iceland.rf)
fit3 <- mtar_grid(~ Jokulsa + Vatnsdalsa | Temperature | Precipitation,
                  data=iceland.rf,subset={Date<="1974-11-06"},row.names=Date,
                  dist=c("Slash","Student-t"), nregim.min=1, nregim.max=2,
                  p.min=15, p.max=15, q.min=4, q.max=4, d.min=2, d.max=2,
                  n.burnin=1000, n.sim=2000, n.thin=2,
                  plan_strategy="multisession")
oos3 <- out_of_sample(fit3, newdata=subset(iceland.rf, Date>"1974-11-06"),
                      n.ahead=55, FUN=median)
oos3
```

```
###### Example 4: U.S. stock returns
data(US.returns)
fit4 <- mtar_grid(~ CCR | dVIX, data=US.returns, subset={Date<="2025-11-28"},
                  row.names=Date, dist=c("Laplace","Student-t","Slash"),
                  nregim.min=2, nregim.max=2, p.min=3, p.max=3, d.min=3,
                  d.max=3, n.burnin=1000, n.sim=2000, n.thin=2,
                  plan_strategy="multisession")
oos4 <- out_of_sample(fit4, newdata=subset(US.returns, Date>"2025-11-28"),
                      n.ahead=100, FUN=median)
oos4
```

---

`out_of_sample.mtar` *Computing Out-of-Sample predictive accuracy measures*

---

## Description

Computes Out-of-Sample predictive accuracy measures for two or more objects of class `mtar`.

## Usage

```
## S3 method for class 'mtar'
out_of_sample(
  ...,
  newdata,
  n.ahead = NULL,
  credible = 0.95,
  FUN = mean,
  rolling = NULL
)
```

## Arguments

| | |
|---|---|
| `...` | one or more objects of class `mtar`. |
| `newdata` | A `data.frame` containing future values of the threshold series (if included in the fitted model), the exogenous series (if included in the fitted model), and the realized values of the output series. |
| `n.ahead` | A positive integer specifying the number of steps ahead to forecast. |
| `credible` | An optional numeric value in $(0, 1)$ specifying the level of the required prediction intervals. By default, `credible` is set to `0.95`. |
| `FUN` | An optional function used to summarize the `n.ahead` values computed for each predictive accuracy measure. By default, `FUN` is set to `mean`. |
| `rolling` | An optional positive integer specifying the rolling-window size used for forecasting. By default, `rolling = NULL`, indicating that rolling-window forecasting is not performed. |

predict.mtar *Forecasting for multivariate TAR models*

## Description

Computes forecasts from a fitted multivariate Threshold Autoregressive (TAR) model.

## Usage

```
## S3 method for class 'mtar'
predict(
  object,
  ...,
  newdata,
  n.ahead = NULL,
  row.names,
  credible = 0.95,
  out.of.sample = FALSE,
  rolling = NULL
)
```

## Arguments

| | |
|---|---|
| `object` | An object of class `mtar` obtained from a call to `mtar()`. |
| `...` | Additional arguments that may affect the prediction method. |
| `newdata` | An optional `data.frame` containing future values of the threshold series (if included in the fitted model), the exogenous series (if included in the fitted model), and, when `out.of.sample = TRUE`, the realized values of the output series. |
| `n.ahead` | A positive integer specifying the number of steps ahead to forecast. |
| `row.names` | An optional variable in `newdata` specifying labels for the time |
| `credible` | An optional numeric value in $(0, 1)$ specifying the level of the required credible intervals. By default, `credible` is set to `0.95`. |
| `out.of.sample` | An optional logical indicator. If `TRUE` then the log-score, Energy-Score (ES), Absolute Error (AE), Absolute Percentage Error (APE), Squared Error (SE), are computed as measures of predictive accuracy. In this case, `newdata` must include the observed values of the output series. |
| `rolling` | An optional positive integer specifying the rolling-window size used for forecasting with fixed parameters. By default, `rolling = NULL`, indicating that rolling-window forecasting is not performed. |

## References

Nieto, F.H. (2005) Modeling Bivariate Threshold Autoregressive Processes in the Presence of Missing Data. Communications in Statistics - Theory and Methods, 34, 905-930.

Romero, L.V. and Calderon, S.A. (2021) Bayesian estimation of a multivariate TAR model when the noise process follows a Student-t distribution. Communications in Statistics - Theory and Methods, 50, 2508-2530.

Calderon, S.A. and Nieto, F.H. (2017) Bayesian analysis of multivariate threshold autoregressive models with missing data. Communications in Statistics - Theory and Methods, 46, 296-318.

Karlsson, S. (2013) Chapter 15-Forecasting with Bayesian Vector Autoregression. In Elliott, G. and Timmermann, A. Handbook of Economic Forecasting, Volume 2, 791–89, Elsevier.

Vanegas, L.H. and Calderón, S.A. and Rondón, L.M. (2025) Bayesian estimation of a multivariate tar model when the noise process distribution belongs to the class of gaussian variance mixtures. International Journal of Forecasting.

## Examples

```
###### Example 1: Returns of the closing prices of three financial indexes
data(returns)
fit1 <- mtar(~ COLCAP + BOVESPA | SP500, data=returns, row.names=Date,
             subset={Date<="2015-12-07"}, dist="Student-t",
             ars=ars(nregim=3,p=c(1,1,2)), n.burnin=1000, n.sim=2000,
             n.thin=2)
p1 <- predict(fit1, newdata=subset(returns,Date>"2015-12-07"), n.ahead=75,
              credible=0.8, out.of.sample=TRUE)
with(p1,summary)
with(p1,cbind(LS,ES,APE,CR))
plot(p1,last=100)

###### Example 2: Rainfall and two river flows in Colombia
data(riverflows)
fit2 <- mtar(~ Bedon + LaPlata | Rainfall, data=riverflows, row.names=Date,
             subset={Date<="2009-02-13"}, dist="Laplace",
             ars=ars(nregim=3,p=5), n.burnin=1000, n.sim=2000, n.thin=2)
p2 <- predict(fit2, newdata=subset(riverflows,Date>"2009-02-13"), n.ahead=60,
              credible=0.8, out.of.sample=TRUE)
with(p2,summary)
with(p2,cbind(LS,ES,APE,CR))
plot(p2,last=100)

###### Example 3: Temperature, precipitation, and two river flows in Iceland
data(iceland.rf)
fit3 <- mtar(~ Jokulsa + Vatnsdalsa | Temperature | Precipitation,
             data=iceland.rf, subset={Date<="1974-11-06"}, row.names=Date,
             ars=ars(nregim=2,p=15,q=4,d=2), n.burnin=1000, n.sim=2000,
             n.thin=2, dist="Slash")
p3 <- predict(fit3, newdata=subset(iceland.rf,Date>"1974-11-06"), n.ahead=55,
              credible=0.8, out.of.sample=TRUE)
with(p3,summary)
with(p3,cbind(LS,ES,APE,CR))
plot(p3,last=100)

###### Example 4: U.S. stock returns
data(US.returns)
fit4 <- mtar(~ CCR | dVIX, data=US.returns, subset={Date<="2025-11-28"},
```

```
              row.names=Date, ars=ars(nregim=2,p=3,d=3), n.burnin=1000,
              n.sim=2000, n.thin=2, dist="Student-t")
p4 <- predict(fit4, newdata=subset(US.returns,Date>"2025-11-28"),n.ahead=100,
               credible=0.8, out.of.sample=TRUE)
with(p4,summary)
with(p4,cbind(LS,ES,APE,CR))
plot(p4,last=100)
```

---

`priors` *Auxiliary function for setting hyperparameter values*

---

## Description

This function constructs and validates the list of hyperparameter values used to define the prior distributions of the model parameters. Hyperparameters not explicitly provided by the user are assigned their default values, which define non-informative prior distributions.

## Usage

```
priors(prior, regim, k, dist, setar, ssvs)
```

## Arguments

- `prior` A list specifying user-defined values for the hyperparameters. Any hyperparameters not included in this list are set to their default values.
- `regim` A positive integer indicating the number of regimes in the model.
- `k` A positive integer indicating the dimension of the multivariate output series.
- `dist` A character string specifying the distribution chosen to model the noise process.
- `setar` A positive integer indicating the component of the output series that acts as the threshold variable in a SETAR specification. If `NULL` is specified then the model is not a SETAR.
- `ssvs` A logical indicating whether the Stochastic Search Variable Selection (SSVS) procedure should be applied.

## Value

A list containing the hyperparameter values defining the prior distributions of all model parameters.

| returns | *Returns of the closing prices of three financial indexes* |
|---|---|

## Description

This dataset contains daily returns computed from the closing prices of the COLCAP, BOVESPA, and S&P 500 stock market indexes over the period from February 10, 2010, to March 31, 2016, comprising 1,505 observations. The COLCAP index reflects the price dynamics of the 20 most liquid stocks traded on the Colombian stock market. The BOVESPA index represents the Brazilian stock market, the largest and most important exchange in Latin America and among the largest worldwide. The Standard & Poor's 500 (S&P 500) index tracks the performance of 500 large-cap companies listed in the United States.

## Usage

```
data(returns)
```

## Format

A data frame with 1,505 rows and 4 variables:

**Date** A vector indicating the date of each observation.

**COLCAP** A numeric vector containing the returns of the COLCAP index.

**SP500** A numeric vector containing the returns of the S&P 500 index.

**BOVESPA** A numeric vector containing the returns of the BOVESPA index.

## References

Romero, L.V. and Calderon, S.A. (2021) Bayesian estimation of a multivariate TAR model when the noise process follows a Student-t distribution. Communications in Statistics - Theory and Methods, 50, 2508-2530.

## Examples

```
data(returns)
dev.new()
plot(ts(as.matrix(returns[,-1])), main="Returns")
```

`riverflows` *Rainfall and two river flows in Colombia*

## Description

This dataset contains daily observations of rainfall (in millimeters) and river flows (in $m^3$/s) for two rivers in southern Colombia. Rainfall was recorded at a meteorological station located at an altitude of 2400 meters above sea level. River flows were measured at two hydrological stations: El Trébol station, which monitors the Bedón River at an altitude of 1720 meters, and Villalosada station, which monitors the La Plata River at an altitude of 1300 meters. The stations are located near the equator, a geographic feature that helps reduce seasonal distortions and facilitates the analysis of the dynamic relationship between rainfall and river flows. The sample period spans from January 1, 2006, to April 14, 2009.

## Usage

```
data(riverflows)
```

## Format

A data frame with 1200 rows and 4 variables:

**Date** A vector indicating the date of each observation.

**Bedon** A numeric vector giving the daily flow of the Bedón River.

**LaPlata** A numeric vector giving the daily flow of the La Plata River.

**Rainfall** A numeric vector indicating daily rainfall amounts.

## References

Calderon, S.A. and Nieto, F.H. (2017) Bayesian analysis of multivariate threshold autoregressive models with missing data. Communications in Statistics - Theory and Methods, 46, 296-318.

## Examples

```
data(riverflows)
dev.new()
plot(ts(as.matrix(riverflows[,-1])), main="Rainfall and river flows")
```

simtar — *Simulation of multivariate time series from a TAR model*

## Description

This function simulates multivariate time series generated by a user-specified Threshold Autoregressive (TAR) model.

## Usage

```
simtar(
  n,
  k = 2,
  ars = ars(),
  Intercept = TRUE,
  trend = c("none", "linear", "quadratic"),
  nseason = NULL,
  parms,
  delay = 0,
  thresholds = NULL,
  t.series = NULL,
  ex.series = NULL,
  dist = c("Gaussian", "Student-t", "Hyperbolic", "Laplace", "Slash",
    "Contaminated normal", "Skew-Student-t", "Skew-normal"),
  skewness = NULL,
  extra = NULL,
  setar = NULL,
  Verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| `n` | A positive integer specifying the length of the simulated output series. |
| `k` | A positive integer specifying the dimension of the multivariate output series. |
| `ars` | A list defining the TAR model structure, composed of four elements: the number of regimes (`nregim`), the autoregressive order (`p`), and the maximum lags of the exogenous (`q`) and threshold (`d`) series within each regime. This object can be validated using the `ars()` function. |
| `Intercept` | An optional logical indicating whether an intercept term is included in each regime. |
| `trend` | An optional character string specifying the degree of deterministic time trend to be included in each regime. Available options are a linear trend (`"linear"`), a quadratic trend (`"quadratic"`), or no trend (`"none"`). By default, `trend` is set to `"none"`. |

| | |
|---|---|
| `nseason` | An optional integer, greater than or equal to 2, specifying the number of seasonal periods. When provided, `nseason - 1` seasonal dummy variables are added to the regressors within each regime. By default, `nseason` is set to `NULL`, thereby indicating that the TAR model has no seasonal effects. |
| `parms` | A list with one sublist per regime. Each sublist contains two matrices: the first matrix corresponds to the location parameters, and the second matrix corresponds to the scale parameters of the model. |
| `delay` | An optional non-negative integer specifying the delay parameter of the threshold series. By default, `delay` is set to `0`. |
| `thresholds` | A numeric vector of length `nregim-1` containing the threshold values, sorted in ascending order. |
| `t.series` | A matrix containing the values of the threshold series. |
| `ex.series` | A matrix containing the values of the multivariate exogenous series. |
| `dist` | An optional character string specifying the multivariate distribution chosen to model the noise process. Supported options include Gaussian (`"Gaussian"`), Student-$t$ (`"Student-t"`), Slash (`"Slash"`), Symmetric Hyperbolic (`"Hyperbolic"`), Laplace (`"Laplace"`), and Contaminated Normal (`"Contaminated normal"`). By default, `dist` is set to `"Gaussian"`. |
| `skewness` | An optional numeric vector specifying the skewness parameters of the noise distribution, when applicable. |
| `extra` | An optional value specifying the extra parameter of the noise distribution, when required. |
| `setar` | An optional positive integer indicating which component of the output series is used as the threshold variable. By default, `setar` is set to `NULL`, indicating that the model is not of SETAR type. |
| `Verbose` | An optional logical indicating whether a description of the simulated TAR model should be printed. By default, `Verbose` is set to `TRUE`. |

## Value

A `data.frame` containing the simulated multivariate output series, and, if specified, the threshold series and multivariate exogenous series.

## References

Vanegas, L.H. and Calderón, S.A. and Rondón, L.M. (2025) Bayesian estimation of a multivariate tar model when the noise process distribution belongs to the class of gaussian variance mixtures. International Journal of Forecasting.

## Examples

```
###### Simulation of a trivariate TAR model with two regimes
n <- 2000
k <- 3
myars <- ars(nregim=2,p=c(1,2))
Z <- as.matrix(arima.sim(n=n+max(myars$p),list(ar=c(0.5))))
```

```
probs <- sort((0.6 + runif(myars$nregim-1)*0.8)*c(1:(myars$nregim-1))/myars$nregim)
dist <- "Student-t"; extra <- 6
parms <- list()
for(j in 1:myars$nregim){
    np <- 1 + myars$p[j]*k
    parms[[j]] <- list()
   parms[[j]]$location <- c(ifelse(runif(np*k)<=0.5,1,-1)*rbeta(np*k,shape1=4,shape2=16))
    parms[[j]]$location <- matrix(parms[[j]]$location,np,k)
    parms[[j]]$scale <- rgamma(k,shape=1,scale=1)*diag(k)
}
thresholds <- quantile(Z,probs=probs)
out1 <- simtar(n=n, k=k, ars=myars, parms=parms, thresholds=thresholds,
               t.series=Z, dist=dist, extra=extra)
str(out1)

fit1 <- mtar(~ Y1 + Y2 + Y3 | Z, data=out1, ars=myars, dist=dist,
             n.burn=2000, n.sim=3000, n.thin=2)
summary(fit1)

###### Simulation of a trivariate VAR model
n <- 2000
k <- 3
myars <- ars(nregim=1,p=2)
dist <- "Slash"; extra <- 2
parms <- list()
for(j in 1:myars$nregim){
    np <- 1 + myars$p[j]*k
    parms[[j]] <- list()
   parms[[j]]$location <- c(ifelse(runif(np*k)<=0.5,1,-1)*rbeta(np*k,shape1=4,shape2=16))
    parms[[j]]$location <- matrix(parms[[j]]$location,np,k)
    parms[[j]]$scale <- rgamma(k,shape=1,scale=1)*diag(k)
}
out2 <- simtar(n=n, k=k, ars=myars, parms=parms, dist=dist, extra=extra)
str(out2)

fit2 <- mtar(~ Y1 + Y2 + Y3, data=out2, ars=myars, dist=dist,
             n.burn=2000, n.sim=3000, n.thin=2)
summary(fit2)

n <- 5000
k <- 3
myars <- ars(nregim=2,p=c(1,2))
dist <- "Laplace"
parms <- list()
for(j in 1:myars$nregim){
    np <- 1 + myars$p[j]*k
    parms[[j]] <- list()
   parms[[j]]$location <- c(ifelse(runif(np*k)<=0.5,1,-1)*rbeta(np*k,shape1=4,shape2=16))
    parms[[j]]$location <- matrix(parms[[j]]$location,np,k)
    parms[[j]]$scale <- rgamma(k,shape=1,scale=1)*diag(k)
}
out3 <- simtar(n=n, k=k, ars=myars, parms=parms, delay=2,
               thresholds=-1, dist=dist, setar=2)
```

```
str(out3)

fit3 <- mtar(~ Y1 + Y2 + Y3, data=out3, ars=myars, dist=dist,
             n.burn=2000, n.sim=3000, n.thin=2, setar=2)
summary(fit3)
```

---

`US.returns` *U.S. Stock Returns*

---

## Description

The dataset comprises observations of both continuously compounded and simple returns derived from the S&P 500 index, along with the difference of the Chicago Board Options Exchange Market Volatility Index (VIX). The sample period spans from January 5, 2005, to April 24, 2026.

## Usage

```
data(US.returns)
```

## Format

A data frame with 5420 rows and 6 variables:

**Date** A vector indicating the date of each observation.

**SP500** A numeric vector giving the S&P500 index.

**VIX** A numeric vector giving the Chicago Board Options Exchange Market Volatility Index (VIX).

**CCR** A numeric vector giving the continuously compounded returns.

**SR** A numeric vector giving the simple returns.

**dVIX** A numeric vector giving the difference $VIX_{t-1} - VIX_{t-2}$.

## References

Massacci, D. (2014) A two-regime threshold model with conditional skewed student-t distributions for stock returns. Economic Modelling, 43, 9-20.

## Examples

```
data(US.returns)
dev.new()
plot(ts(as.matrix(US.returns[,-1])))
```

## vcov.mtar — *vcov method for objects of class* mtar

### Description

Computes estimates of the variance–covariance matrices for the scale parameters of a fitted multivariate TAR model.

### Usage

```
## S3 method for class 'mtar'
vcov(object, ..., FUN = mean)
```

### Arguments

| | |
|---|---|
| object | an object of class mtar, typically obtained from a call to mtar(). |
| ... | additional arguments passed to FUN. |
| FUN | a function to be applied to the MCMC chains of the scale parameters in order to obtain point estimates. By default, FUN is set to mean(). |

### Value

A list containing the variance–covariance estimates obtained by applying FUN to the MCMC chains associated with the scale parameters.

## WAIC — *Watanabe-Akaike or Widely Available Information Criterion (WAIC)*

### Description

Computes Watanabe-Akaike or Widely Available Information Criterion (WAIC), an adjusted within-sample measure of predictive accuracy, for models estimated using Bayesian methods.

### Usage

```
WAIC(...)
```

### Arguments

| | |
|---|---|
| ... | one or more fitted model objects of the same class. |

### Value

A numeric matrix containing the WAIC values corresponding to each fitted object supplied in ....

## References

Watanabe S. (2010). Asymptotic Equivalence of Bayes Cross Validation and Widely Applicable Information Criterion in Singular Learning Theory. The Journal of Machine Learning Research, 11, 3571–3594.

---

# WAIC.mtar — *Watanabe-Akaike or Widely Available Information Criterion (WAIC) for objects of class* `mtar`

---

## Description

This function computes the Watanabe-Akaike or Widely Available Information Criterion (WAIC), for objects of class `mtar`.

## Usage

```
## S3 method for class 'mtar'
WAIC(...)
```

## Arguments

| | |
|---|---|
| `...` | one or several objects of the class *mtar*. |

## Value

A numeric matrix containing the WAIC values corresponding to each *mtar* object in the input.

## See Also

DIC

## Examples

```
###### Example 1: Returns of the closing prices of three financial indexes
data(returns)
fit1 <- mtar_grid(~ COLCAP + BOVESPA | SP500, data=returns, row.names=Date,
                  subset={Date<="2015-12-07"}, dist=c("Gaussian","Student-t",
                  "Slash","Laplace"), nregim.min=2, nregim.max=3, p.min=2,
                  p.max=2, n.burnin=1000, n.sim=2000, n.thin=2,
                  plan_strategy="multisession")
WAIC(fit1)

###### Example 2: Rainfall and two river flows in Colombia
data(riverflows)
fit2 <- mtar_grid(~ Bedon + LaPlata | Rainfall, data=riverflows,
                  row.names=Date, subset={Date<="2009-02-13"},dist="Laplace",
                  nregim.min=2, nregim.max=3, p.min=1, p.max=3,n.burnin=1000,
                  n.sim=2000, n.thin=2, plan_strategy="multisession")
WAIC(fit2)
```

```
###### Example 3: Temperature, precipitation, and two river flows in Iceland
data(iceland.rf)
fit3 <- mtar_grid(~ Jokulsa + Vatnsdalsa | Temperature | Precipitation,
                  data=iceland.rf,subset={Date<="1974-11-06"},row.names=Date,
                  dist=c("Slash","Student-t"), nregim.min=1, nregim.max=2,
                  p.min=15, p.max=15, q.min=4, q.max=4, d.min=2, d.max=2,
                  n.burnin=1000, n.sim=2000, n.thin=2,
                  plan_strategy="multisession")
WAIC(fit3)

###### Example 4: U.S. stock returns
data(US.returns)
fit4 <- mtar_grid(~ CCR | dVIX, data=US.returns, subset={Date<="2025-11-28"},
                  row.names=Date, dist=c("Laplace","Student-t","Slash"),
                  nregim.min=2, nregim.max=2, p.min=3, p.max=3, d.min=3,
                  d.max=3, n.burnin=1000, n.sim=2000, n.thin=2,
                  plan_strategy="multisession")
WAIC(fit4)
```

# Index