Package: nsarfima (via r-universe)

September 10, 2024

Title Methods for Fitting and Simulating Non-Stationary ARFIMA Models

Version 0.2.0.0

Author Benjamin Groebe [aut, cre]

Maintainer Benjamin Groebe <ben.groebe@gmail.com>

Description Routines for fitting and simulating data under autoregressive fractionally integrated moving average (ARFIMA) models, without the constraint of covariance stationarity. Two fitting methods are implemented, a pseudo-maximum likelihood method and a minimum distance estimator. Mayoral, L. (2007) <doi:10.1111/j.1368-423X.2007.00202.x>. Beran, J. (1995) <doi:10.1111/j.2517-6161.1995.tb02054.x>.

Depends R (>= 3.6.0)

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2020-08-06 14:30:03 UTC

Contents

arfima.sim *Simulate ARFIMA Process*

Description

Simulates a series under the given ARFIMA model by applying an MA filter to a series of innovations.

Usage

```
arfima.sim(
  n,
  d = 0,
  ar = numeric(),
  ma = numeric(),
  mu = 0,
  sig2 = 1,
  stat.int = FALSE,
  n.burn,
  innov,
  exact.innov = TRUE
)
```

Arguments

n	Desired series length.
d	Fractional differencing parameter.
ar	Vector of autoregressive parameters.
ma	Vector of moving average parameters, following the same sign convention as `arima`.
mu	Mean of process. By default, added after integer integration but before burn-in truncation (see `stat.int`).
sig2	Innovation variance if innovations not provided.
stat.int	Controls integration for non-stationary values of `d` (*i.e.* `d>=0.5`). If `TRUE`, `d` split into integer part and stationary part, which will result in a trend when `d>=0.5` and `mu!=0`.
n.burn	Number of burn-in steps. If not given, chosen based off presence of long memory (*i.e.* `d>0`).
innov	Series of innovations. Drawn from normal distribution if not given.
exact.innov	Whether to force the exact innovation series to be used. If `FALSE`, innovations will be prepended with resampled points as needed to match `n+n.burn`.

Details

The model is defined by values for the AR and MA parameters (ϕ and θ, respectively), along with the fractional differencing parameter *d*. When $d \geq 0.5$, then the integer part is taken as $m = \lfloor d + 0.5 \rfloor$, and the remainder (between -0.5 and 0.5) stored as *d*. For $m = 0$, the model is:

$$\left(1 - \sum_{i=1}^{p} \phi_i B^i\right) (1 - B)^d \left(y_t - \mu\right) = \left(1 + \sum_{i=1}^{q} \theta_i B^i\right) \epsilon_t$$

where *B* is the backshift operator ($By_t = y_{t-1}$) and ϵ_t is the innovation series. When $m > 0$, the model is defined by:

$$y_t = (1 - B)^{-m} x_t$$

$$\left(1 - \sum_{i=1}^{p} \phi_i B^i\right) (1 - B)^d (x_t - \mu) = \left(1 + \sum_{i=1}^{q} \theta_i B^i\right) \epsilon_t$$

When `stat.int = FALSE`, the differencing filter applied to the innovations is not split into parts, and the series model follows the first equation regardless of the value of *d*. This means that μ is added to the series after filtering and before any truncation. When `stat.int = TRUE`, $x_t - \mu$ is generated from filtered residuals, μ is added, and the result is cumulatively summed *m* times. For non-zero mean and $m > 0$, this will yield a polynomial trend in the resulting data.

Note that the burn-in length may affect the distribution of the sample mean, variance, and autocovariance. Consider this when generating ensembles of simulated data

Value

A numeric vector of length n.

Examples

```
## Generate ARFIMA(1,d,0) series with Gaussian innovations
x <- arfima.sim(1000, d=0.6, ar=c(-0.4))

## Generate ARFIMA(1,d,0) series with uniform innovations.
innov.series <- runif(1000, -1, 1)
x <- arfima.sim(1000, d=0.6, ar=c(-0.4), innov=innov.series, exact.innov=TRUE)
```

`mde.arfima` *Minimum Distance Estimation of ARFIMA Model*

Description

Fits an ARFIMA(*p*,*d*,*q*) model to a time series using a minimum distance estimator. For details see Mayoral (2007).

Usage

```
mde.arfima(
  y,
  p = 1,
  q = 0,
  d.range = c(0, 1),
  start,
  lag.max = floor(sqrt(length(y))),
  incl.mean = TRUE,
  verbose = FALSE,
  method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
  control = list()
)
```

Arguments

`y`	Numeric vector of the time series.
`p`	Autoregressive order.
`q`	Moving average order.
`d.range`	Range of allowable values for fractional differencing parameter. Smallest value must be greater than -1.
`start`	Named vector of length 1 + `p` + `q` containing initial fit values for the fractional differencing parameter, the AR parameters, and the MA parameters (*e.g.* `start = c(d=0.4, ar.1=-0.4, ma.1=0.3, ma.2=0.4)`). If missing, automatically selected.
`lag.max`	Maximum lag to use when calculating the residual autocorrelations. For details see Mayoral (2007).
`incl.mean`	Whether or not to include a mean term in the model. The default value of `TRUE` is recommended unless the true mean is known and previously subtracted. Mean is returned with standard error, which may be misleading for `d>=0.5`.
`verbose`	Whether to print summary of fit.
`method`	Method for `optim`, see `help(optim)`.
`control`	List of additional arguments for `optim`, see `help(optim)`.

Value

A list containing:

`pars`	A numeric vector of parameter estimates.
`std.errs`	A numeric vector of standard errors on parameters.
`cov.mat`	Parameter covariance matrix (excluding mean).
`fit.obj`	`optim` fit object.

`p.val`	Ljung-Box *p*-value for fit.
`residuals`	Fit residuals.

Note

This method makes no assumptions on the distribution of the innovation series, and the innovation variance does not factor into the covariance matrix of parameter estimates. For this reason, it is not included in the results, but can be estimated from the residuals—see Mayoral (2007).

References

Mayoral, L. (2007). Minimum distance estimation of stationary and non-stationary ARFIMA processes. *The Econometrics Journal*, **10**, 124-148. doi: 10.1111/j.1368-423X.2007.00202.x

See Also

`mle.arfima` for psuedo-maximum likelihood estimation.

Examples

```
set.seed(1)
x <- arfima.sim(1000, d=0.6, ar=c(-0.4))
fit <- mde.arfima(x, p=1, incl.mean=FALSE, verbose=TRUE)


## Fit Summary
## --------------------
## Ljung-Box p-val:  0.276
##           d     ar.1
## est 0.55031 -0.39261
## err 0.03145  0.03673
##
## Correlation Matrix for ARFIMA Parameters
##           d   ar.1
## d    1.0000 0.6108
## ar.1 0.6108 1.0000
```

`mle.arfima` *Pseudo-Maximum Likelihood Estimation of ARFIMA Model*

Description

Fits an ARFIMA(p,d,q) model to a time series using a pseudo-maximum likelihood estimator. For details see Beran (1995).

Usage

```
mle.arfima(
  y,
  p = 1,
  q = 0,
  d.range = c(0, 1),
  start,
  incl.mean = TRUE,
  verbose = FALSE,
  method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN", "Brent"),
  control = list()
)
```

Arguments

`y`	Numeric vector of the time series.
`p`	Autoregressive order.
`q`	Moving average order.
`d.range`	Range of allowable values for fractional differencing parameter. Smallest value must be greater than -1.
`start`	Named vector of length 1 + `p` + `q` containing initial fit values for the fractional differencing parameter, the AR parameters, and the MA parameters (*e.g.* `start = c(d=0.4, ar.1=-0.4, ma.1=0.3, ma.2=0.4)`). If missing, automatically selected.
`incl.mean`	Whether or not to include a mean term in the model. The default value of `TRUE` is recommended unless the true mean is known and previously subtracted. Mean is returned with standard error, which may be misleading for `d>=0.5`.
`verbose`	Whether to print summary of fit.
`method`	Method for `optim`, see `help(optim)`.
`control`	List of additional arguments for `optim`, see `help(optim)`.

Value

A list containing:

`pars`	A numeric vector of parameter estimates.
`std.errs`	A numeric vector of standard errors on parameters.
`cov.mat`	Parameter covariance matrix (excluding mean).
`fit.obj`	`optim` fit object.
`p.val`	Ljung-Box *p*-value for fit.
`residuals`	Fit residuals.

References

Beran, J. (1995). Maximum Likelihood Estimation of the Differencing Parameter for Short and Long Memory Autoregressive Integrated Moving Average Models. *Journal of the Royal Statistical Society. Series B (Methodological)*, **57**, No. 4, 659-672. doi: 10.1111/j.2517-6161.1995.tb02054.x

See Also

`mde.arfima` for minimum distance estimation.

Examples

```
set.seed(1)
x <- arfima.sim(1000, d=0.6, ar=c(-0.4))
fit <- mle.arfima(x, p=1, incl.mean=FALSE, verbose=TRUE)

## Fit Summary
## --------------------
## Ljung-Box p-val:  0.263
##        sig2       d     ar.1
## est 1.09351 0.53933 -0.37582
## err 0.05343 0.04442  0.05538
##
## Correlation Matrix for ARFIMA Parameters
##         sig2       d    ar.1
## sig2  1.0000 -0.3349  0.4027
## d    -0.3349  1.0000 -0.8318
## ar.1  0.4027 -0.8318  1.0000
```

Index