Package 'starvars'

May 4, 2020

Type Package

Title Vector Logistic Smooth Transition Models / Realized Covariances Construction

Version 0.1.7

Description Allows the user to estimate a vector logistic smooth transition autoregressive model via maximum log-likelihood or nonlinear least squares. It further permits to test for linearity in the multivariate framework against a vector logistic smooth transition autoregressive model with a single transition variable. The estimation method is discussed in Terasvirta and Yang (2014, <doi:10.1108/S0731-9053(2013)0000031008>). Also, realized covariances can be constructed from stock market prices or returns, as explained in Andersen et al. (2001, <doi:10.1016/S0304-405X(01)00055-1>).

License GPL

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

Imports MASS, ks, zoo, data.table, methods, matrixcalc, vars, nloptr, maxLik, rlist, stats4, highfrequency, fGarch, R.utils, lubridate, xts, lessR, quantmod

URL https://github.com/andbucci/starvars

NeedsCompilation no

Author Andrea Bucci [aut, cre, cph],
Giulio Palomba [aut],
Eduardo Rossi [aut],
Andrea Faragalli [ctb]

Maintainer Andrea Bucci <andrea.bucci@unich.it>

Repository CRAN

Date/Publication 2020-05-04 14:40:02 UTC

R topics documented:

`lrvarbart` *Long-run variance using Bartlett kernel*

Description

Function returns the long-run variance of a time series, relying on the Bartlett kernel. The window size of the kernel is the cube root of the sample size.

Usage

```
lrvarbart(x)
```

Arguments

`x`	a (T x 1) vector containing the time series over period *T*

Value

`lrv`	long-run variance
`bandwidth`	size of the window

Author(s)

The code was written by Andrea Bucci.

References

Hamilton J. D. (1994), Time Series Analysis. *Princeton University Press*

Tsay R.S. (2005), Analysis of Financial Time Series. *John Wiley & SONS*

Examples

```
data(Realized)

lrvarbart(Realized[,1])
```

`multiCUMSUM` *Multivariate CUMSUM test*

Description

Function returns the test statistics for the presence of co-breaks in a set of multivariate time series.

Usage

```
multiCUMSUM(data, alpha = 0.95)
## S3 method for class 'multiCUMSUM'
print(x, ...)
```

Arguments

`data` a (T x N) matrix/data.frame containing the N time series over period *T*

`alpha` Confidence level. By default set to 0.95

`x` ‘multiCUMSUM’ object

`...` further arguments to be passed to and from other methods

Value

`Lambda Test statistics`

a *test statistics on the presence of a co-break in the conditional mean*

`Omega Test statistics`

a *test statistics on the presence of a co-break in the conditional variance*

Author(s)

The code was written by Andrea Bucci and Giulio Palomba.

References

Aue A., Hormann S., Horvath L.and Reimherr M. (2009), Break detection in the covariance structure of multivariate time series models. *The Annals of Statistics*. 37: 4046-4087

Bai J., Lumsdaine R. L. and Stock J. H. (1998), Testing For and Dating Common Breaks in Multivariate Time Series. *Review of Economic Studies*. 65: 395-432

Barassi M., Horvath L. and Yuqian Z. (2018), Change-Point Detection in the Conditional Correlation Structure of Multivariate Volatility Models. *Journal of Business \& Economic Statistics*

Examples

```
data(Realized)

testCS <- multiCUMSUM(Realized[,1:10], alpha = 0.95)
```

predict.VLSTAR *VLSTAR Prediction*

Description

This function allows to obtain one-step ahead forecasts, with the prediction intervals, of a VLSTAR object.

Usage

```
## S3 method for class 'VLSTAR'
predict(object, ...)
```

Arguments

`object`	An object of class VLSTAR obtained through VLSTARnls or VLSTARml
`...`	further arguments to be passed to and from other methods

Value

A data frame of predictions for each dependent variable and the (1-α) prediction intervals.

Author(s)

The code was written by Andrea Bucci

References

Terasvirta T. and Yang Y. (2014), Specification, Estimation and Evaluation of Vector Smooth Transition Autoregressive Models with Applications. *CREATES Research Paper 2014-8*

See Also

`VLSTAR` for log-likehood and nonlinear least squares estimation of the VLSTAR model.

Examples

```
##
##See 'VLSTAR' examples
##
```

rcov *Realized Covariance*

Description

Function returns the vectorization of the lowest triangular of the Realized Covariance matrices for different frequency.

Usage

```
rcov(data, freq, ret = FALSE,
    cholesky = FALSE)
```

Arguments

`data`	a (T x N) xts object containing the N price/return series over period *T*
`freq`	a string defining the desired frequency for the Realized Covariance matrices between "daily", "monthly", "quarterly" or "yearly"
`ret`	boolean, in case it is TRUE the data are converted to returns, FALSE by default
`cholesky`	boolean, in case it is TRUE the Cholesky factors of the Realized Covariance matrices are calculated, FALSE by default

Value

`Realized Covariances`

a $M \times N(N+1)/2$ matrix of realized covariances, where *M* is the number of lower frequency data

`Cholesky Factors (optional)`

a $M \times N(N+1)/2$ matrix of Cholesky factors of the realized covariance matrices, where *M* is the number of lower frequency data

Author(s)

The code was written by Andrea Bucci

References

Andersen T.G., Bollerslev T., Diebold F.X. and Labys P. (2003), Modeling and Forecasting Realized Volatility. *Econometrica*. 71: 579-625

Barndorff-Nielsen O.E. and Shephard N. (2002), Econometric analysis of realised volatility and its use in estimating stochastic volatility models. *Journal of the Royal Statistical Society*. 64(2): 253-280

Examples

```
require(highfrequency)

data(sample_5minprices_jumps)

rc <- rcov(sample_5minprices_jumps, freq = 'daily', cholesky = TRUE, ret = TRUE)
rc
```

Realized *Monthly time series used to test VLSTAR models.*

Description

This data set contains the series of realized covariances in 4 stock market indices, i.e. SP-500, Nikkei, DAX, and FTSE, Dividend Yield and Earning Price growth rate, inflation growth rates for U.S., U.K., Japan and Germany, from August 1990 to June 2018.

Usage

```
data(Realized)
```

Format

A zoo data frame with 334 monthly observations, ranging from 1990:M8 until 2018:M6.

SP Monthly realized variances of S&P 500 index.
SP-NIKKEI Monthly realized covariances between S&P 500 and Nikkei.
SP-FTSE Monthly realized covariances between S&P 500 and FTSE.
SP-DAX Monthly realized covariances between S&P 500 and DAX.
NIKKEI Monthly realized variances of Nikkei index.
NIKKEI-FTSE Monthly realized covariances between Nikkei and FTSE.
NIKKEI-DAX Monthly realized covariances between Nikkei and DAX.
FTSE Monthly realized variances of FTSE index.
FTSE-DAX Monthly realized covariances between FTSE and DAX.
DAX Monthly realized variances of DAX index.
DP Monthly Dividends growth rate over the past year relative to current market prices; S&P 500 index.
EP Monthly Earnings growth rate over the past year relative to current market prices; S&P500 index.
Inf_US US monthly Industrial Production growth.
Inf_UK UK monthly Industrial Production growth.
Inf_JPN Japan monthly Industrial Production growth.
Inf_GER Germany monthly Industrial Production growth.

Author(s)

Andrea Bucci

See Also

`rcov` to build realized covariances from stock prices or returns.

VLSTAR *VLSTAR- Estimation*

Description

This function allows the user to estimate the coefficients of a VLSTAR model with m regimes through maximum likelihood or nonlinear least squares. The set of starting values of Gamma and C for the convergence algorithm can be either passed or obtained via searching grid.

Usage

```
VLSTAR(y1, x1 = NULL, p = NULL, m = 2, st = NULL, constant = TRUE, n.combi = 50,
       n.iter = 500, starting = NULL, epsilon = 10^(-3), exo = FALSE,
       method = c('ML', 'NLS'))
## S3 method for class 'VLSTAR'
print(x, ...)
## S3 method for class 'VLSTAR'
summary(object, ...)
## S3 method for class 'VLSTAR'
print.summary(x, ...)
```

Arguments

`y1` Data.frame or matrix of dependent variables of dimension Txn

`x1` (optional) Data.frame or matrix of exogenous variables of dimension TxK

`p` lag order

`m` number of regimes

`st` single transition variable for all the equation of dimension Tx1

`constant` TRUE or FALSE to include or not the constant

`n.combi` Number of combination for the searching grid of Gamma and C

`n.iter` number of iteration of the algorithm until forced convergence

`starting` (optional) set of intial values for Gamma and C

`epsilon` convergence check measure

`exo` TRUE or FALSE to specify if exogenous variables enter in the model

`method` Fitting method: maximum likelihood or nonlinear least squares.

`x, object` fitted ‘VLSTAR’ object

`...` further arguments to be passed to and from other methods

Details

The multivariate smooth transition model is an extension of the smooth transition regression model introduced by Bacon and Watts (1971) (see also Anderson and Vahid, 1998). The general model is

$$y_t = \mu_0 + \sum_{j=1}^{p} \Phi_{0,j}\, y_{t-j} + A_0 x_t \cdot G_t(s_t;\gamma,c)[\mu_1 + \sum_{j=1}^{p} \Phi_{1,j}\, y_{t-j} + A_1 x_t] + \varepsilon_t$$

where μ_0 and μ_1 are the $\tilde{n} \times 1$ vectors of intercepts, $\Phi_{0,j}$ and $\Phi_{1,j}$ are square $\tilde{n} \times \tilde{n}$ matrices of parameters for lags $j = 1, 2, \ldots, p$, A_0 and A_1 are $\tilde{n} \times k$ matrices of parameters, x_t is the $k \times 1$ vector of exogenous variables and ε_t is the innovation. Finally, $G_t(s_t;\gamma,c)$ is a $\tilde{n} \times \tilde{n}$ diagonal matrix of transition function at time *t*, such that

$$G_t(s_t;\gamma,c) = \{G_{1,t}(s_{1,t};\gamma_1,c_1), G_{2,t}(s_{2,t};\gamma_2,c_2), \ldots, G_{\tilde{n},t}(s_{\tilde{n},t};\gamma_{\tilde{n}},c_{\tilde{n}})\}.$$

Each diagonal element $G^r_{i,t}$ is specified as a logistic cumulative density functions, i.e.

$$G^r_{i,t}(s^r_{i,t};\gamma^r_i,c^r_i) = \left[1 + \exp\left\{ -\gamma^r_i(s^r_{i,t} - c^r_i)\right\}\right]^{-1}$$

for $latex$ and $r = 0, 1, \ldots, m-1$, so that the first model is a Vector Logistic Smooth Transition AutoRegressive (VLSTAR) model. The ML estimator of θ is obtained by solving the optimization problem

$$\hat{\theta}_{ML} = arg \max_{\theta} logL(\theta)$$

where $logL(\theta)$ is the log-likelihood function of VLSTAR model, given by

$$ll(y_t|I_t;\theta) = -\frac{T\tilde{n}}{2}\ln(2\pi) - \frac{T}{2}\ln|\Omega| - \frac{1}{2}\sum_{t=1}^{T}(y_t - \tilde{G}_t B\, z_t)'\Omega^{-1}(y_t - \tilde{G}_t B\, z_t)$$

The NLS estimators of the VLSTAR model are obtained by solving the optimization problem

$$\hat{\theta}_{NLS} = arg \min_{\theta} \sum_{t=1}^{T}(y_t - \Psi'_t B' x_t)'(y_t - \Psi'_t B' x_t).$$

Generally, the optimization algorithm may converge to some local minimum. For this reason, providing valid starting values of θ is crucial. If there is no clear indication on the initial set of parameters, θ, this can be done by implementing a grid search. Thus, a discrete grid in the parameter space of Γ and C is create to obtain the estimates of B conditionally on each point in the grid. The initial pair of Γ and C producing the smallest sum of squared residuals is chosen as initial values, then the model is linear in parameters. The algorithm is the following:

1. Construction of the grid for Γ and C, computing Ψ for each poin in the grid
2. Estimation of $\hat{B}$ in each equation, calculating the residual sum of squares, Q_t
3. Finding the pair of Γ and C providing the smallest Q_t
4. Once obtained the starting-values, estimation of parameters, *B*, via nonlinear least squares (NLS)
5. Estimation of Γ and C given the parameters found in step 4
6. Repeat step 4 and 5 until convergence.

Value

An object of class VLSTAR, with standard methods.

Author(s)

The code was written by Andrea Bucci

References

Anderson H.M. and Vahid F. (1998), Testing multiple equation systems for common nonlinear components. *Journal of Econometrics*. 84: 1-36

Bacon D.W. and Watts D.G. (1971), Estimating the transition between two intersecting straight lines. *Biometrika*. 58: 525-534

Terasvirta T. and Yang Y. (2014), Specification, Estimation and Evaluation of Vector Smooth Transition Autoregressive Models with Applications. *CREATES Research Paper 2014-8*

See Also

`VLSTARjoint` to test the presence of a unique transition variable among equations and `predict.VLSTAR` for details on predictions produced for this model

Examples

```
data(Realized)
y <- Realized[-1,1:10]
y <- y[-nrow(y),]
st <- Realized[-nrow(Realized),1]
st <- st[-length(st)]
fit.VLSTAR <- VLSTAR(y, p = 1, n.combi = 3,
n.iter = 3, st = st, method ='NLS')

# a few methods for VLSTAR
summary(fit.VLSTAR)
predict(fit.VLSTAR, newdata = Realized[nrow(Realized),1:10])
```

VLSTARjoint *Joint linearity test*

Description

This function allows the user to test linearity against a Vector Smooth Transition Autoregressive Model with a single transition variable.

Usage

```
VLSTARjoint(y1, x1, st, alpha = 0.05)
## S3 method for class 'VLSTARjoint'
print(x, ...)
```

Arguments

`y1` Data.frame or matrix of dependent variables of dimension Txn

`x1` (optional) Data.frame or matrix of exogenous variables of dimension TxK

`st` single transition variable for all the equation of dimension Tx1

`alpha` Confidence level

`x` 'VLSTARjoint' object

`...` further arguments to be passed to and from other methods

Details

Given a VLSTAR model with a unique transition variable, $s_{1t} = s_{2t} = \ldots = s_{\widetilde{n}t} = s_t$, a generalization of the linearity test presented in Luukkonen, Saikkonen and Terasvirta (1988) may be implemented.

Assuming a 2-state VLSTAR model, such that

$$y_t = B_1 z_t + G_t B_2 z_t + \varepsilon_t.$$

Where the null $H_0 : \gamma_j = 0$, $j = 1, \ldots, \widetilde{n}$, is such that $G_t \equiv (1/2)/\widetilde{n}$ and the previous Equation is linear. When the null cannot be rejected, an identification problem of the parameter c_j in the transition function emerges, that can be solved through a first-order Taylor expansion around $\gamma_j = 0$.

The approximation of the logistic function with a first-order Taylor expansion is given by

$$\begin{aligned} G(s_t; \gamma_j, c_j) &= (1/2) + (1/4)\gamma_j(s_t - c_j) + r_{jt} \\ &= a_j s_t + b_j + r_{jt} \end{aligned}$$

where $a_j = \gamma_j/4$, $b_j = 1/2 - a_j c_j$ and r_j is the error of the approximation. If G_t is specified as follows

$$\begin{aligned} G_t &= diag\big\{a_1 s_t + b_1 + r_{1t}, \ldots, a_{\widetilde{n}} s_t + b_{\widetilde{n}} + r_{\widetilde{n}t}\big\} \\ &= A s_t + B + R_t \end{aligned}$$

where $A = diag(a_1, \ldots, a_{\widetilde{n}})$, $B = diag(b_1, \ldots, b_{\widetilde{n}})$ e $R_t = diag(r_{1t}, \ldots, r_{\widetilde{n}t})$, y_t can be written as

$$\begin{aligned} y_t &= B_1 z_t + (A s_t + B + R_t) B_2 z_t + \varepsilon_t \\ &= (B_1 + B B_2) z_t + A B_2 z_t s_t + R_t B_2 z_t + \varepsilon_t \\ &= \Theta_0 z_t + \Theta_1 z_t s_t + \varepsilon_t^* \end{aligned}$$

where $\Theta_0 = B_1 + B_2' B$, $\Theta_1 = B_2' A$ and $\varepsilon_t^* = R_t B_2 + \varepsilon_t$. Under the null, $\Theta_0 = B_1$ and $\Theta_1 = 0$, while the previous model is linear, with $\varepsilon_t^* = \varepsilon_t$. It follows that the Lagrange multiplier test, under the null, is derived from the score

$$\frac{\partial \log L(\widetilde{\theta})}{\partial \Theta_1} = \sum_{t=1}^{T} z_t s_t (y_t - \widetilde{B}_1 z_t)' \widetilde{\Omega}^{-1} = S(Y - Z\widetilde{B}_1)\widetilde{\Omega}^{-1},$$

where

$$S = z_1' s_1 \vdots z_t' s_t$$

and where $\widetilde{B}_1$ and $\widetilde{\Omega}$ are estimated from the model in H_0. If $P_Z = Z(Z'Z)^{-1}Z'$ is the projection matrix of Z, the LM test is specified as follows

$$LM = tr\{\widetilde{\Omega}^{-1}(Y - Z\widetilde{B}_1)'S\big[S'(I_t - P_Z)S\big]^{-1}S'(Y - Z\widetilde{B}_1)\}.$$

Under the null, the test statistics is distributed as a χ^2 with $\widetilde{n}(p \cdot \widetilde{n} + k)$ degrees of freedom.

Value

An object of class VLSTARjoint.

Author(s)

The code was written by Andrea Bucci

References

Luukkonen R., Saikkonen P. and Terasvirta T. (1988), Testing Linearity Against Smooth Transition Autoregressive Models. *Biometrika*, 75: 491-499

Terasvirta T. and Yang Y. (2015), Linearity and Misspecification Tests for Vector Smooth Transition Regression Models. *CREATES Research Paper 2014-4*

See Also

`VLSTAR` for log-likehood and for NLS estimation of the VLSTAR model.

Examples

```
data(Realized)
VLSTARjoint(Realized[-1,1:10], Realized[-1,11:16], st=Realized[1:(nrow(Realized)-1),1])
```

Index