Package: FCVAR (via r-universe)

September 17, 2024

Title Estimation and Inference for the Fractionally Cointegrated VAR

Version 0.1.4

Description Estimation and inference using the Fractionally Cointegrated Vector Autoregressive (VAR) model. It includes functions for model specification, including lag selection and cointegration rank selection, as well as a comprehensive set of options for hypothesis testing, including tests of hypotheses on the cointegrating relations, the adjustment coefficients and the fractional differencing parameters. An article describing the FCVAR model with examples is available on the Webpage <https://sites.google.com/view/mortennielsen/software>.

Depends R (>= 3.5)

Imports pracma, fracdist

URL https://github.com/LeeMorinUCF/FCVAR

BugReports https://github.com/LeeMorinUCF/FCVAR/issues

License GPL-3

Encoding UTF-8

LazyData true

Language en-US

RoxygenNote 7.1.2

Suggests knitr, rmarkdown, testthat

NeedsCompilation no

Author Lealand Morin [aut, cre] (<https://orcid.org/0000-0001-8539-1386>), Morten Nielsen [aut] (<https://orcid.org/0000-0002-1337-9844>), Michal Popiel [aut]

Maintainer Lealand Morin <lealand.morin@ucf.edu>

Repository https://leemorinucf.r-universe.dev

RemoteUrl https://github.com/leemorinucf/fcvar

RemoteRef HEAD

RemoteSha 3a7684ade7d27dbaad0907229c11ed8ed8d8ad85

Contents

FCVAR *A package for estimating the Fractionally Cointegrated VAR model.*

Description

The FCVAR package estimates the Fractionally Cointegrated Vector Autoregressive (VAR) model. It includes functions for lag selection, cointegration rank selection and hypothesis testing.

Details

Functions in the FCVAR package are divided into four categories: Estimation, Postestimation, Specification and Auxiliary functions.

Value

Returns `NULL`. Object included for description only.

Estimation functions

The estimation functions include the primary estimation function `FCVARestn` and associated functions to set estimation options and display results. Some of these functions define, modify and test the user-specified options for estimation. `FCVARoptions` defines the default estimation options used in the FCVAR estimation procedure and the related programs. The user can then revise the options such as the settings for optimization and restrictions for testing hypotheses. After making these changes, an internal function `FCVAROptionUpdates` sets and tests estimation options for validity and compatibility.

Postestimation functions

The postestimation functions are used to display summary statistics, test hypotheses and test the goodness of fit of the estimated model. These include:

`FCVARhypoTest` for a likelihood ratio test of a restricted vs. an unrestricted model

`FCVARboot` for generating a distribution of a likelihood ratio test statistic

`FCVARforecast` for calculating recursive forecasts with the FCVAR model

Specification functions

The specification functions are used to estimate a series of models in order to make model specfication decisions. These include:

`FCVARlagSelect` for selection of the lag order

`FCVARrankTests` for choosing the cointegrating rank

`FCVARbootRank` for generating a distribution of a likelihood ratio test statistic for the rank test

Auxiliary functions

The auxiliary functions are used to perform intermediate calculations for estimation. These functions are mainly designed for use only within the estimation function. Some exceptions include:

`FracDiff` for fractionally differencing a multivariate series

`FCVARsimBS` for generating bootstrap samples from the FCVAR model

`FCVARlikeGrid` for performing a grid-search optimization with the FCVAR likelihood function

Examples

A dataset `votingJNP2014` is included for examples of the model building process. Sample model builds with hypothesis tests and examples of other extensions are found in the example script `FCVAR_demo_JNP2014.R`. See FCVAR_README.pdf for details at

`https://github.com/LeeMorinUCF/FCVAR/blob/master/FCVAR_README.pdf`

and also see `https://sites.google.com/view/mortennielsen/software` for more information about estimating the FCVAR model.

FCVARboot *Bootstrap Likelihood Ratio Test*

Description

FCVARboot generates a distribution of a likelihood ratio test statistic using a wild bootstrap, following the method of Boswijk, Cavaliere, Rahbek, and Taylor (2016). It takes two sets of options as inputs to estimate the model under the null and the unrestricted model.

Usage

```
FCVARboot(x, k, r, optRES, optUNR, B)
```

Arguments

x	A matrix of variables to be included in the system.
k	The number of lags in the system.
r	The cointegrating rank.
optRES	An S3 object of class FCVAR_opt that stores the chosen estimation options for the restricted model, as generated from FCVARoptions(), with adjustments as necessary.
optUNR	An S3 object of class FCVAR_opt that stores the chosen estimation options for the unrestricted model.
B	The number of bootstrap samples.

Value

A list FCVARboot_stats containing the estimation results, including the following parameters:

LRbs A $Bx1$ vector of simulated likelihood ratio statistics

pv An approximate p-value for the likelihood ratio statistic based on the bootstrap distribution.

H A list containing the likelihood ratio test results. It is identical to the output from FCVARhypoTest, with one addition, namely H$pvBS which is the bootstrap p-value

mBS The model estimates under the null hypothesis.

mUNR The model estimates under the alternative hypothesis.

References

Boswijk, Cavaliere, Rahbek, and Taylor (2016) "Inference on co-integration parameters in heteroskedastic vector autoregressions," Journal of Econometrics 192, 64-85.

See Also

FCVARoptions to set default estimation options. FCVARestn is called to estimate the models under the null and alternative hypotheses.

Other FCVAR postestimation functions: FCVARhypoTest(), GetCharPolyRoots(), MVWNtest(), plot.FCVAR_roots(), summary.FCVAR_roots(), summary.MVWN_stats()

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
opt$plotRoots <- 0
optUNR <- opt
optRES <- opt
optRES$R_Beta <- matrix(c(1, 0, 0), nrow = 1, ncol = 3)
set.seed(42)
FCVARboot_stats <- FCVARboot(x, k = 2, r = 1, optRES, optUNR, B = 2)
# In practice, set the number of bootstraps so that (B+1)*alpha is an integer,
# where alpha is the chosen level of significance.
# For example, set B = 999 (but it takes a long time to compute).
```

FCVARbootRank *Distribution of LR Test Statistic for the Rank Test*

Description

FCVARbootRank generates a distribution of a likelihood ratio test statistic for the rank test using a wild bootstrap, following the method of Cavaliere, Rahbek, and Taylor (2010). It takes the two ranks as inputs to estimate the model under the null and the model under the alternative.

Usage

```
FCVARbootRank(x, k, opt, r1, r2, B)
```

Arguments

x	A matrix of variables to be included in the system. If k>0, actual data is used for initial values.
k	The number of lags in the system.
opt	An S3 object of class FCVAR_opt that stores the chosen estimation options, generated from FCVARoptions().
r1	The cointegrating rank under the null hypothesis.
r2	The cointegrating rank under the alternative hypothesis.
B	The number of bootstrap samples.

Value

A list FCVARbootRank_stats containing the test results, including the following parameters:

LRbs A B x 1 vector of simulated likelihood ratio statistics.

pv An approximate p-value for the LR statistic based on the bootstrap distribution.

H A list containing LR test results. It is identical to the output from HypoTest, with one addition, namely H$pvBS which is the bootstrap p-value)

mBS Model estimates under the null hypothesis.

mUNR Model estimates under the alternative hypothesis.

References

Cavaliere, G., A. Rahbek, and A. M. R. Taylor (2010). "Testing for co-integration in vector autoregressions with non-stationary volatility," Journal of Econometrics 158, 7-24.

See Also

FCVARoptions to set default estimation options. HypoTest for the format of a hypothesis test results. FCVARestn for the estimates from a rectricted and unrestricted model within a hypothesis test.

Other FCVAR specification functions: FCVARlagSelect(), FCVARrankTests(), summary.FCVAR_lags(), summary.FCVAR_ranks()

Examples

```
opt <- FCVAROptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
opt$plotRoots <- 0
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
set.seed(42)
FCVARbootRank_stats <- FCVARbootRank(x, k = 2, opt, r1 = 0, r2 = 1, B = 2)
# In practice, set the number of bootstraps so that (B+1)*alpha is an integer,
# where alpha is the chosen level of significance.
# For example, set B = 999 (but it takes a long time to compute).
```

FCVARestn *Estimate FCVAR model*

Description

FCVARestn estimates the Fractionally Cointegrated VAR model. It is the central function in the FCVAR package with several nested functions, each described below. It estimates the model parameters, calculates the standard errors and the number of free parameters, obtains the residuals and the roots of the characteristic polynomial. print.FCVARestn prints the estimation results from the output of FCVARestn.

Usage

```
FCVARestn(x, k, r, opt)
```

Arguments

x	A matrix of variables to be included in the system.
k	The number of lags in the system.
r	The cointegrating rank.
opt	An S3 object of class FCVAR_opt that stores the chosen estimation options, generated from FCVARoptions().

Value

An S3 object of class FCVAR_model containing the estimation results, including the following parameters:

startVals Starting values used for optimization.

options Estimation options.

like Model log-likelihood.

coeffs Parameter estimates.

rankJ Rank of Jacobian for the identification condition.

fp Number of free parameters.

SE Standard errors.

NegInvHessian Negative of inverse Hessian matrix.

Residuals Model residuals.

cPolyRoots Roots of characteristic polynomial.

printVars Additional variables required only for printing the output of FCVARestn to screen.

k The number of lags in the system.

r The cointegrating rank.

p The number of variables in the system.

cap_T The sample size.

opt An S3 object of class FCVAR_opt that stores the chosen estimation options, generated from FCVARoptions().

See Also

FCVARoptions to set default estimation options. FCVARestn calls this function at the start of each estimation to verify validity of options. summary.FCVAR_model prints the output of FCVARestn to screen.

Other FCVAR estimation functions: FCVARoptions(), summary.FCVAR_model()

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
m1 <- FCVARestn(x, k = 2, r = 1, opt)



opt1 <- opt
opt1$R_psi <- matrix(c(1, 0), nrow = 1, ncol = 2)
opt1$r_psi <- 1
m1r1 <- FCVARestn(x, k = 2, r = 1, opt1)



opt1 <- opt
opt1$R_Beta <- matrix(c(1, 0, 0), nrow = 1, ncol = 3)
m1r2 <- FCVARestn(x, k = 2, r = 1, opt1)



opt1 <- opt
opt1$R_Alpha <- matrix(c(0, 1, 0), nrow = 1, ncol = 3)
m1r4 <- FCVARestn(x, k = 2, r = 1, opt1)
```

FCVARforecast *Forecasts with the FCVAR Model*

Description

FCVARforecast calculates recursive forecasts with the FCVAR model.

Usage

```
FCVARforecast(x, model, NumPeriods)
```

Arguments

x	A matrix of variables to be included in the system. The forecast will be calculated using these values as starting values.
model	A list of estimation results, just as if estimated from FCVARest. The parameters in model can also be set or adjusted by assigning new values.
NumPeriods	The number of time periods in the simulation.

Value

A NumPeriods $\times p$ matrix xf of forecasted values.

See Also

FCVARoptions to set default estimation options. FCVARestn for the specification of the model. FCVARforecast calls FracDiff and Lbk to calculate the forecast.

Other FCVAR auxiliary functions: FCVARlikeGrid(), FCVARsimBS(), FCVARsim(), FracDiff(), plot.FCVAR_grid()

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
opt1 <- opt
opt1$R_Alpha <- matrix(c(0, 1, 0), nrow = 1, ncol = 3)
m1r4 <- FCVARestn(x, k = 2, r = 1, opt1)
xf <- FCVARforecast(x, m1r4, NumPeriods = 12)
```

FCVARhypoTest *Test of Restrictions on FCVAR Model*

Description

FCVARhypoTest performs a likelihood ratio test of the null hypothesis: "model is modelR" against the alternative hypothesis: "model is modelUNR".

Usage

```
FCVARhypoTest(modelUNR, modelR)
```

Arguments

modelUNR	A list of estimation results created for the unrestricted model.
modelR	A list of estimation results created for the restricted model.

Value

A list LRtest containing the test results, including the following parameters:

loglikUNR The log-likelihood for the unrestricted model.

loglikR The log-likelihood for the restricted model.

`df` The degrees of freedom for the test.

`LRstat` The likelihood ratio test statistic.

`p_LRtest` The p-value for the likelihood ratio test.

See Also

The test is calculated using the results of two calls to `FCVARestn`, under the restricted and unrestricted models. Use `FCVARoptions` to set default estimation options for each model, then set restrictions as needed before `FCVARestn`.

Other FCVAR postestimation functions: `FCVARboot()`, `GetCharPolyRoots()`, `MVWNtest()`, `plot.FCVAR_roots()`, `summary.FCVAR_roots()`, `summary.MVWN_stats()`

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
m1 <- FCVARestn(x, k = 2, r = 1, opt)
opt1 <- opt
opt1$R_psi <- matrix(c(1, 0), nrow = 1, ncol = 2)
opt1$r_psi <- 1
m1r1 <- FCVARestn(x, k = 2, r = 1, opt1)
Hdb <- FCVARhypoTest(modelUNR = m1, modelR = m1r1)



opt1 <- opt
opt1$R_Beta <- matrix(c(1, 0, 0), nrow = 1, ncol = 3)
m1r2 <- FCVARestn(x, k = 2, r = 1, opt1)
Hbeta1 <- FCVARhypoTest(m1, m1r2)



opt1 <- opt
opt1$R_Alpha <- matrix(c(0, 1, 0), nrow = 1, ncol = 3)
m1r4 <- FCVARestn(x, k = 2, r = 1, opt1)
Halpha2 <- FCVARhypoTest(m1, m1r4)
```

`FCVARlagSelect` *Select Lag Order*

Description

`FCVARlagSelect` takes a matrix of variables and performs lag selection on it by using the likelihood ratio test. Output and test results are printed to the screen.

Usage

```
FCVARlagSelect(x, kmax, r, order, opt)
```

Arguments

x	A matrix of variables to be included in the system.
kmax	The maximum number of lags in the system.
r	The cointegrating rank.This is often set equal to p, the number of variables in the system, since it is better to overspecify than underspecify the model.
order	The order of serial correlation for white noise tests.
opt	An S3 object of class FCVAR_opt that stores the chosen estimation options, generated from FCVARoptions().

Value

An S3 object of type FCVAR_lags containing the results from repeated estimation of the FCVAR model with different orders of the autoregressive lag length. Note that row j of each of the vectors in the FCVAR_lags object contains the associated results for lag length j+1. The FCVAR_lags object includes the following parameters:

D A (kmax + 1) x 2 vector of estimates of d and b.

loglik A (kmax + 1) x 1 vector of log-likelihood values.

LRtest A (kmax + 1) x 1 vector of likelihood ratio test statistics for tests of significance of Γ_{j+1}.

pvLRtest A (kmax + 1) x 1 vector of P-values for the likelihood ratio tests of significance of Γ_{j+1}.

i_aic The lag corresponding to the minimum value of the Akaike information criteria.

aic A (kmax + 1) x 1 vector of values of the Akaike information criterion.

i_bic The lag corresponding to the minimum value of the Bayesian information criteria.

bic A (kmax + 1) x 1 vector of values of the Bayesian information criterion.

pvMVq A scalar P-value for the Q-test for multivariate residual white noise.

pvWNQ A (kmax + 1) x 1 vector of P-values for the Q-tests for univariate residual white noise.

pvWNLM A (kmax + 1) x 1 vector of P-values for the LM-tests for univariate residual white noise.

kmax The maximum number of lags in the system.

r The cointegrating rank. This is often set equal to p, the number of variables in the system, since it is better to overspecify than underspecify the model.

p The number of variables in the system.

cap_T The sample size.

order The order of serial correlation for white noise tests.

opt An S3 object of class FCVAR_opt that stores the chosen estimation options, generated from FCVARoptions().

See Also

FCVARoptions to set default estimation options. FCVARestn is called repeatedly within this function for each candidate lag order. summary.FCVAR_lags prints a summary of the output of FCVARlagSelect to screen.

Other FCVAR specification functions: FCVARbootRank(), FCVARrankTests(), summary.FCVAR_lags(), summary.FCVAR_ranks()

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
FCVARlagSelectStats <- FCVARlagSelect(x, kmax = 3, r = 3, order = 12, opt)
```

FCVARlikeGrid *Grid Search to Maximize Likelihood Function*

Description

FCVARlikeGrid performs a grid-search optimization by calculating the likelihood function on a grid of candidate parameter values. This function evaluates the likelihood over a grid of values for c(d,b) (or phi). It can be used when parameter estimates are sensitive to starting values to give an approximation of the global maximum that can then be used as the starting value in the numerical optimization in FCVARestn. plot.FCVAR_grid plots the likelihood function from FCVARlikeGrid.

Usage

```
FCVARlikeGrid(x, k, r, opt)
```

Arguments

x	A matrix of variables to be included in the system.
k	The number of lags in the system.
r	The cointegrating rank.
opt	An S3 object of class FCVAR_opt that stores the chosen estimation options, generated from FCVARoptions().

Value

An S3 object of type FCVAR_grid containing the optimization results, including the following parameters:

params A vector params of d and b (and mu if level parameter is selected) corresponding to a maximum over the grid of c(d,b) or phi.

dbHatStar A vector of d and b corresponding to a maximum over the grid of c(d,b) or phi.

muHatStar A vector of the optimal mu if level parameter is selected.

Grid2d An indicator for whether or not the optimization is conducted over a 2-dimensional parameter space, i.e. if there is no equality restriction on d and b.

dGrid A vector of the grid points in the parameter d, after any transformations for restrictions, if any.

bGrid A vector of the grid points in the parameter b, after any transformations for restrictions, if any.

dGrid_orig A vector of the grid points in the parameter d, in units of the fractional integration parameter.

bGrid_orig A vector of the grid points in the parameter b, in units of the fractional integration parameter.

like The maximum value of the likelihood function over the chosen grid.

k The number of lags in the system.

r The cointegrating rank.

opt An S3 object of class FCVAR_opt that stores the chosen estimation options, generated from FCVARoptions().

Note

If opt$LocalMax == 0, FCVARlikeGrid returns the parameter values corresponding to the global maximum of the likelihood on the grid. If opt$LocalMax == 1, FCVARlikeGrid returns the parameter values for the local maximum corresponding to the highest value of b. This alleviates the identification problem mentioned in Johansen and Nielsen (2010, section 2.3).

References

Johansen, S. and M. Ø. Nielsen (2010). "Likelihood inference for a nonstationary fractional autoregressive model," Journal of Econometrics 158, 51-66.

See Also

FCVAROptions to set default estimation options. plot.FCVAR_grid plots the likelihood function from FCVARlikeGrid.

Other FCVAR auxiliary functions: FCVARforecast(), FCVARsimBS(), FCVARsim(), FracDiff(), plot.FCVAR_grid()

Examples

```
# Restrict equality of fractional parameters.

opt <- FCVARoptions()
opt$dbStep1D     <- 0.2 # Coarser grid for plotting example.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
opt$restrictDB   <- 1 # impose restriction d=b ? 1 <- yes, 0 <- no.
opt$progress     <- 2 # Show progress report on each value of b.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
likeGrid_params <- FCVARlikeGrid(x, k = 2, r = 1, opt)
plot(likeGrid_params)


# Linear restriction on fractional parameters.

opt <- FCVARoptions()
opt$dbStep1D     <- 0.2 # Coarser grid for plotting example.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
opt$restrictDB   <- 0 # impose restriction d=b ? 1 <- yes, 0 <- no.
# Impose linear restriction on d and b:
opt$R_psi        <- matrix(c(2, -1), nrow = 1, ncol = 2)
opt$r_psi        <- 0.5
opt$progress     <- 2 # Show progress report on each value of b.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
likeGrid_params <- FCVARlikeGrid(x, k = 2, r = 1, opt)
plot(likeGrid_params)


# Constrained 2-dimensional optimization.
# Impose restriction dbMax >= d >= b >= dbMin.

opt <- FCVARoptions()
opt$dbStep1D     <- 0.2 # Coarser grid for plotting example.
opt$dbStep2D     <- 0.2 # Coarser grid for plotting example.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 1 # impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
opt$restrictDB   <- 0 # impose restriction d=b ? 1 <- yes, 0 <- no.
opt$progress     <- 2 # Show progress report on each value of b.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
likeGrid_params <- FCVARlikeGrid(x, k = 2, r = 1, opt)


# Unconstrained 2-dimensional optimization.

opt <- FCVARoptions()
opt$dbStep1D     <- 0.1 # Coarser grid for plotting example.
opt$dbStep2D     <- 0.2 # Coarser grid for plotting example.
```

```
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
opt$restrictDB   <- 0 # impose restriction d=b ? 1 <- yes, 0 <- no.
opt$progress     <- 2 # Show progress report on each value of b.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
likeGrid_params <- FCVARlikeGrid(x, k = 2, r = 1, opt)
```

FCVARoptions *Set Estimation Options*

Description

FCVARoptions defines the estimation options used in the FCVAR estimation procedure and the related programs.

Usage

```
FCVARoptions(...)
```

Arguments

...	A list of arguments to set to values other than the default settings. See the argument names in the return value below.

Value

An S3 object of class FCVAR_opt that stores the default estimation options, which includes the following parameters:

unc_optim_control A list of options in the form of the argument control in the optim function for *unconstrained* optimization of the likelihood function over the fractional integration parameters. This is also used in the switching algorithm employed when linear constraints are imposed on the cointegrating relations beta or the adjustment coefficients alpha, so it must at least contain the arguments maxit and reltol, since it uses those parameters.

con_optim_control A list of options in the form of the argument control in either the optim or the constrOptim function for *constrained* optimization of the likelihood function over the fractional integration parameters, using the 'L-BFGS-B' algorithm. It must at least contain the arguments maxit and pgtol.

LineSearch Indicator for conducting a line search optimization within the switching algorithm when optimizing over constraints on the cointegrating relations β or the adjustment coefficients α. See Doornik (2018, Section 2.2) for details.

LocalMax Indicator to select the local maximum with the highest value of b when there are multiple local optima. This is meant to alleviate the identification problem discussed in Johansen and Nielsen (2010, Section 2.3) and Carlini and de Magistris (2019). When LocalMax <- 0, the optimization returns the values of d and b corresponding to the global optimum.

dbMax Upper bound for the fractional integration parameters d, b.

dbMin Lower bound for the fractional integration parameters d, b.

db0 The starting values for optimization of the fractional integration parameters d, b.

constrained Indicator to impose restriction dbMax >= d >= b >= dbMin.

restrictDB Indicator to impose restriction d = b.

N The number of initial values: the observations to condition upon.

unrConstant Indicator to include an unrestricted constant.

rConstant Indicator to include a restricted constant.

levelParam Indicator to include level parameter.

C_db CHECK whether still used.

c_db CHECK whether still used.

UB_db An upper bound on the fractional integration parameters d and b, after transforming the parameters to account for any restrictions imposed.

LB_db A lower bound on the fractional integration parameters d and b, after transforming the parameters to account for any restrictions imposed.

R_psi A matrix for defining restrictions on the fractional integration parameters d and b, of the form $R_\psi(d, b)' = r_\psi$.

r_psi A vector for defining restrictions on the fractional integration parameters d and b, of the form $R_\psi(d, b)' = r_\psi$.

R_Alpha A matrix for defining restrictions on the adjustment coefficients of the form $R_\alpha \alpha = r_\alpha$.

r_Alpha A vector for defining restrictions on the adjustment coefficients of the form $R_\alpha \alpha = r_\alpha$.

R_Beta A matrix for defining restrictions on the cointegrating relations of the form $R_\beta \beta = r_\beta$.

r_Beta A vector for defining restrictions on the cointegrating relations of the form $R_\beta \beta = r_\beta$.

print2screen Indicator to print output to screen.

printGammas Indicator to print estimates and standard errors on autoregressive coefficients $\Gamma_i, i = i, ..., k$.

printRoots Indicator to print roots of characteristic polynomial.

plotRoots Indicator to plot roots of characteristic polynomial.

CalcSE Indicator to calculate the standard errors. It is used when displaying results.

hess_delta Size of increment for numerical calculation of derivatives of the likelihood function for numerical calculation of the Hessian matrix. The default is 10^(-4), which works well in practice to balance errors between precision and truncation.

gridSearch Indicator to perform a grid search for the optimization over the fractional integration parameters, for more accurate estimation. This will make estimation take longer.

dbStep1D The step size for the grid search over the fractional integration parameters for the 1-dimensional grid search (such as when restrictions are imposed between d and b.).

dbStep2D The step size for the grid search over the fractional integration parameters for the 2-dimensional grid search.

plotLike Indicator to plot the likelihood (only if gridSearch <- 1).

progress Show a waitbar for a progress indicator for the grid search.

updateTime How often progress is updated in the waitbar for the grid search (in seconds).

References

Doornik, J. A. (2018) "Accelerated Estimation of Switching Algorithms: The Cointegrated VAR Model and Other Applications." Scandinavian Journal of Statistics, Volume 45, Issue 2.

Johansen, Søren, and Morten Ørregaard Nielsen (2010) "Likelihood inference for a nonstationary fractional autoregressive model." Journal of Econometrics 158, 51–66.

Carlini, F., and P. S. de Magistris (2019) "On the identification of fractionally cointegrated VAR models with the F(d) condition." Journal of Business & Economic Statistics 37(1), 134–146.

See Also

`FCVARoptionUpdates` to set and test estimation options for validity and compatibility. `FCVARestn` for use of these options in estimation.

Other FCVAR estimation functions: `FCVARestn()`, `summary.FCVAR_model()`

Examples

```
opt <- FCVARoptions()
opt <- FCVARoptions(
    gridSearch   = 0, # Disable grid search in optimization.
    dbMin        = c(0.01, 0.01), # Set lower bound for d,b.
    dbMax        = c(2.00, 2.00), # Set upper bound for d,b.
    constrained  = 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
)
```

FCVARrankTests *Test for Cointegrating Rank*

Description

`FCVARrankTests` performs a sequence of likelihood ratio tests for cointegrating rank.

Usage

```
FCVARrankTests(x, k, opt)
```

Arguments

`x`	A matrix of variables to be included in the system.
`k`	The number of lags in the system.
`opt`	An S3 object of class `FCVAR_opt` that stores the chosen estimation options, generated from `FCVARoptions()`.

Value

An S3 object of type FCVAR_ranks containing the results from cointegrating rank tests, containing the following (p+1) vectors with ith element corresponding to rank = i-1, including the following parameters:

dHat Estimates of d.

bHat Estimates of b.

LogL Maximized log-likelihood.

LRstat LR trace statistic for testing rank r against rank p.

pv The p-value of LR trace test, or "999" if p-value is not available.

k The number of lags in the system.

p The number of variables in the system.

cap_T The sample size.

opt An S3 object of class FCVAR_opt that stores the chosen estimation options, generated from FCVARoptions().

See Also

FCVARoptions to set default estimation options. FCVARestn is called repeatedly within this function for each candidate cointegrating rank. summary.FCVAR_ranks prints a summary of the output of FCVARrankTests to screen.

Other FCVAR specification functions: FCVARbootRank(), FCVARlagSelect(), summary.FCVAR_lags(), summary.FCVAR_ranks()

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
rankTestStats <- FCVARrankTests(x, k = 2, opt)
```

FCVARsim *Draw Samples from the FCVAR Model*

Description

FCVARsim simulates the FCVAR model as specified by input model and starting values specified by data. Errors are drawn from a normal distribution.

Usage

```
FCVARsim(x, model, NumPeriods)
```

Arguments

x	A Nxp matrix of N starting values for the simulated observations.
model	A list of estimation results, just as if estimated from FCVARest. The parameters in model can also be set or adjusted by assigning new values.
NumPeriods	The number of time periods in the simulation.

Value

A NumPeriods by p matrix xBS of simulated observations.

See Also

FCVARoptions to set default estimation options. FCVARestn for the specification of the model. Use FCVARsim to draw a sample from the FCVAR model. For simulations intended for bootstrapping statistics, use FCVARsimBS.

Other FCVAR auxiliary functions: FCVARforecast(), FCVARlikeGrid(), FCVARsimBS(), FracDiff(), plot.FCVAR_grid()

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
results <- FCVARestn(x, k = 2, r = 1, opt)
x_sim <- FCVARsim(x[1:10, ], results, NumPeriods = 100)
```

FCVARsimBS *Draw Bootstrap Samples from the FCVAR Model*

Description

FCVARsimBS simulates the FCVAR model as specified by input model and starting values specified by data. It creates a wild bootstrap sample by augmenting each iteration with a bootstrap error. The errors are sampled from the residuals specified under the model input and have a positive or negative sign with equal probability (the Rademacher distribution).

Usage

```
FCVARsimBS(data, model, NumPeriods)
```

Arguments

data	A Txp matrix of starting values for the simulated realizations.
model	A list of estimation results, just as if estimated from FCVARest. The parameters in model can also be set or adjusted by assigning new values.
NumPeriods	The number of time periods in the simulation.

Value

A NumPeriods by p matrix xBS of simulated bootstrap values.

See Also

FCVARoptions to set default estimation options. FCVARestn for the specification of the model. Use FCVARsim to draw a sample from the FCVAR model. For simulations intended for bootstrapping statistics, use FCVARsimBS.

Other FCVAR auxiliary functions: FCVARforecast(), FCVARlikeGrid(), FCVARsim(), FracDiff(), plot.FCVAR_grid()

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
results <- FCVARestn(x, k = 2, r = 1, opt)
xBS <- FCVARsimBS(x[1:10, ], results, NumPeriods = 100)
```

FracDiff *Fast Fractional Differencing*

Description

FracDiff is a fractional differencing procedure based on the fast fractional difference algorithm of Jensen & Nielsen (2014).

Usage

```
FracDiff(x, d)
```

Arguments

x	A matrix of variables to be included in the system.
d	The order of fractional differencing.

Value

A vector or matrix dx equal to $(1-L)^d x$ of the same dimensions as x.

Note

This function differs from the `diffseries` function in the `fracdiff` package, in that the `diffseries` function demeans the series first. In particular, the difference between the out put of the function calls `FCVAR::FracDiff(x - mean(x), d = 0.5)` and `fracdiff::diffseries(x, d = 0.5)` is numerically small.

References

Jensen, A. N. and M. Ø. Nielsen (2014). "A fast fractional difference algorithm," Journal of Time Series Analysis 35, 428-436.

See Also

`FCVARoptions` to set default estimation options. `FCVARestn` calls `GetParams`, which calls `TransformData` to estimate the FCVAR model. `TransformData` in turn calls `FracDiff` and `Lbk` to perform the transformation.

Other FCVAR auxiliary functions: `FCVARforecast()`, `FCVARlikeGrid()`, `FCVARsimBS()`, `FCVARsim()`, `plot.FCVAR_grid()`

Examples

```
set.seed(42)
WN <- matrix(stats::rnorm(200), nrow = 100, ncol = 2)
MVWNtest_stats <- MVWNtest(x = WN, maxlag = 10, printResults = 1)
x <- FracDiff(x = WN, d = - 0.5)
MVWNtest_stats <- MVWNtest(x = x, maxlag = 10, printResults = 1)
WN_x_d <- FracDiff(x, d = 0.5)
MVWNtest_stats <- MVWNtest(x = WN_x_d, maxlag = 10, printResults = 1)
```

GetCharPolyRoots *Roots of the Characteristic Polynomial*

Description

`GetCharPolyRoots` calculates the roots of the characteristic polynomial and plots them with the unit circle transformed for the fractional model, see Johansen (2008). `summary.FCVAR_roots` prints the output of `GetCharPolyRoots` to screen.

Usage

```
GetCharPolyRoots(coeffs, opt, k, r, p)
```

Arguments

coeffs	A list of coefficients for the FCVAR model. An element of the list of estimation results output from FCVARestn.
opt	An S3 object of class FCVAR_opt that stores the chosen estimation options, generated from FCVAROptions().
k	The number of lags in the system.
r	The cointegrating rank.
p	The number of variables in the system.

Value

An S3 object of type FCVAR_roots with the following elements:

cPolyRoots A vector of the roots of the characteristic polynomial. It is an element of the list of estimation results output from FCVARestn.

b A numeric value of the fractional cointegration parameter.

Note

The roots are calculated from the companion form of the VAR, where the roots are given as the inverse eigenvalues of the coefficient matrix.

References

Johansen, S. (2008). "A representation theory for a class of vector autoregressive models for fractional processes," Econometric Theory 24, 651-676.

See Also

FCVAROptions to set default estimation options. FCVARestn to estimate the model for which to calculate the roots of the characteristic polynomial. summary.FCVAR_roots prints the output of GetCharPolyRoots to screen.

Other FCVAR postestimation functions: FCVARboot(), FCVARhypoTest(), MVWNtest(), plot.FCVAR_roots(), summary.FCVAR_roots(), summary.MVWN_stats()

Examples

```
opt <- FCVAROptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
results <- FCVARestn(x, k = 2, r = 1, opt)
FCVAR_CharPoly <- GetCharPolyRoots(results$coeffs, opt, k = 2, r = 1, p = 3)
```

MVWNtest *Multivariate White Noise Tests*

Description

MVWNtest performs multivariate tests for white noise. It performs both the Ljung-Box Q-test and the LM-test on individual series for a sequence of lag lengths. summary.MVWN_stats prints a summary of these statistics to screen.

Usage

```
MVWNtest(x, maxlag, printResults)
```

Arguments

x	A matrix of variables to be included in the system, typically model residuals.
maxlag	The number of lags for serial correlation tests.
printResults	An indicator to print results to screen.

Value

An S3 object of type MVWN_stats containing the test results, including the following parameters:

Q A 1xp vector of Q statistics for individual series.

pvQ A 1xp vector of P-values for Q-test on individual series.

LM A 1xp vector of LM statistics for individual series.

pvLM A 1xp vector of P-values for LM-test on individual series.

mvQ A multivariate Q statistic.

pvMVQ A p-value for multivariate Q-statistic using p^2*maxlag degrees of freedom.

maxlag The number of lags for serial correlation tests.

p The number of variables in the system.

Note

The LM test is consistent for heteroskedastic series; the Q-test is not.

See Also

FCVARoptions to set default estimation options. FCVARestn produces the residuals intended for this test. LagSelect uses this test as part of the lag order selection process. summary.MVWN_stats prints a summary of the MVWN_stats statistics to screen.

Other FCVAR postestimation functions: FCVARboot(), FCVARhypoTest(), GetCharPolyRoots(), plot.FCVAR_roots(), summary.FCVAR_roots(), summary.MVWN_stats()

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
results <- FCVARestn(x, k = 2, r = 1, opt)
MVWNtest_stats <- MVWNtest(x = results$Residuals, maxlag = 12, printResults = 1)

set.seed(27)
WN <- stats::rnorm(100)
RW <- cumsum(stats::rnorm(100))
MVWN_x <- as.matrix(data.frame(WN = WN, RW = RW))
MVWNtest_stats <- MVWNtest(x = MVWN_x, maxlag = 10, printResults = 1)
```

---

plot.FCVAR_grid *Plot the Likelihood Function for the FCVAR Model*

Description

`plot.FCVAR_grid` plots the likelihood function from `FCVARlikeGrid`. `FCVARlikeGrid` performs a grid-search optimization by calculating the likelihood function on a grid of candidate parameter values. This function evaluates the likelihood over a grid of values for `c(d,b)` (or `phi`, when there are constraints on `c(d,b)`). It can be used when parameter estimates are sensitive to starting values to give an approximation of the global max which can then be used as the starting value in the numerical optimization in `FCVARestn`.

Usage

```
## S3 method for class 'FCVAR_grid'
plot(x, y = NULL, ...)
```

Arguments

`x`	An S3 object of type `FCVAR_grid` output from `FCVARlikeGrid`.
`y`	An argument for generic method `plot` that is not used in `plot.FCVAR_grid`.
`...`	Arguments to be passed to methods, such as graphical parameters for the generic plot function.

Note

Calls `graphics::persp` when `x$Grid2d == TRUE` and calls `graphics::plot` when `x$Grid2d == FALSE`.

See Also

FCVARoptions to set default estimation options. plot.FCVAR_grid plots the likelihood function from FCVARlikeGrid.

Other FCVAR auxiliary functions: FCVARforecast(), FCVARlikeGrid(), FCVARsimBS(), FCVARsim(), FracDiff()

Examples

```
opt <- FCVARoptions()
opt$dbStep1D     <- 0.1 # Coarser grid for plotting example.
opt$dbStep2D     <- 0.2 # Coarser grid for plotting example.
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
opt$progress <- 2 # Show progress report on each value of b.
likeGrid_params <- FCVARlikeGrid(x, k = 2, r = 1, opt)
graphics::plot(likeGrid_params)
```

---

plot.FCVAR_roots *Plot Roots of the Characteristic Polynomial*

Description

plot.FCVAR_roots plots the output of GetCharPolyRoots to screen or to a file. GetCharPolyRoots calculates the roots of the characteristic polynomial and plots them with the unit circle transformed for the fractional model, see Johansen (2008).

Usage

```
## S3 method for class 'FCVAR_roots'
plot(x, y = NULL, ...)
```

Arguments

x	An S3 object of type FCVAR_roots with the following elements: #'
	cPolyRoots A vector of the roots of the characteristic polynomial. It is an element of the list of estimation results output from FCVARestn.
	b A numeric value of the fractional cointegration parameter.
y	An argument for generic method plot that is not used in plot.FCVAR_roots.
...	Arguments to be passed to methods, such as graphical parameters for the generic plot function.

Note

The roots are calculated from the companion form of the VAR, where the roots are given as the inverse eigenvalues of the coefficient matrix.

References

Johansen, S. (2008). "A representation theory for a class of vector autoregressive models for fractional processes," Econometric Theory 24, 651-676.

See Also

`FCVARoptions` to set default estimation options. `FCVARestn` to estimate the model for which to calculate the roots of the characteristic polynomial. `summary.FCVAR_roots` prints the output of `GetCharPolyRoots` to screen.

Other FCVAR postestimation functions: `FCVARboot()`, `FCVARhypoTest()`, `GetCharPolyRoots()`, `MVWNtest()`, `summary.FCVAR_roots()`, `summary.MVWN_stats()`

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
results <- FCVARestn(x, k = 2, r = 1, opt)
FCVAR_CharPoly <- GetCharPolyRoots(results$coeffs, opt, k = 2, r = 1, p = 3)
summary(object = FCVAR_CharPoly)
graphics::plot(x = FCVAR_CharPoly)
```

---

summary.FCVAR_lags *Summarize Statistics from Lag Order Selection*

Description

`summary.FCVAR_lags` prints a summary of the table of statistics from the output of `FCVARlagSelect`. `FCVARlagSelect` takes a matrix of variables and performs lag selection on it by using the likelihood ratio test.

Usage

```
## S3 method for class 'FCVAR_lags'
summary(object, ...)
```

Arguments

object	An S3 object of type FCVAR_lags containing the results from repeated estimation of the FCVAR model with different orders of the autoregressive lag length. It is the output of FCVARlagSelect.
...	additional arguments affecting the summary produced.

See Also

FCVARoptions to set default estimation options. FCVARestn is called repeatedly within this function for each candidate lag order. summary.FCVAR_lags prints a summary of the output of FCVARlagSelect to screen.

Other FCVAR specification functions: FCVARbootRank(), FCVARlagSelect(), FCVARrankTests(), summary.FCVAR_ranks()

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
FCVAR_lag_1 <- FCVARlagSelect(x, kmax = 3, r = 3, order = 12, opt)
summary(object = FCVAR_lag_1)
```

---

summary.FCVAR_model *Summarize Estimation Results from the FCVAR model*

Description

summary.FCVAR_model prints a summary of the estimation results from the output of FCVARestn. FCVARestn estimates the Fractionally Cointegrated VAR model. It is the central function in the FCVAR package with several nested functions. It estimates the model parameters, calculates the standard errors and the number of free parameters, obtains the residuals and the roots of the characteristic polynomial.

Usage

```
## S3 method for class 'FCVAR_model'
summary(object, ...)
```

Arguments

object	An S3 object containing the estimation results of FCVARestn.
...	additional arguments affecting the summary produced.

See Also

FCVARoptions to set default estimation options. FCVARestn calls this function at the start of each estimation to verify validity of options. summary.FCVAR_model prints a summary of the output of FCVARestn to screen.

Other FCVAR estimation functions: FCVARestn(), FCVAROptions()

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
FCVARresults <- FCVARestn(x, k = 2, r = 1, opt)
summary(object = FCVARresults)
```

---

summary.FCVAR_ranks *Summarize Results of Tests for Cointegrating Rank*

Description

summary.FCVAR_ranks prints the table of statistics from the output of FCVARrankTests. FCVARrankTests performs a sequence of likelihood ratio tests for cointegrating rank.

Usage

```
## S3 method for class 'FCVAR_ranks'
summary(object, ...)
```

Arguments

object	An S3 object of type FCVAR_ranks containing the results from repeated estimation of the FCVAR model with different cointegrating ranks. It is the output of FCVARrankTests.
...	additional arguments affecting the summary produced.

See Also

FCVARoptions to set default estimation options. FCVARestn is called repeatedly within this function for each candidate cointegrating rank. summary.FCVAR_ranks prints a summary of the output of FCVARrankTests to screen.

Other FCVAR specification functions: FCVARbootRank(), FCVARlagSelect(), FCVARrankTests(), summary.FCVAR_lags()

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
rankTestStats <- FCVARrankTests(x, k = 2, opt)
summary(object = rankTestStats)
```

---

summary.FCVAR_roots *Print Summary of Roots of the Characteristic Polynomial*

Description

summary.FCVAR_roots prints the output of GetCharPolyRoots to screen. GetCharPolyRoots calculates the roots of the characteristic polynomial to plot them with the unit circle transformed for the fractional model, see Johansen (2008).

Usage

```
## S3 method for class 'FCVAR_roots'
summary(object, ...)
```

Arguments

object	An S3 object of type FCVAR_roots with the following elements:
	cPolyRoots A vector of the roots of the characteristic polynomial. It is an element of the list of estimation results output from FCVARestn.
	b A numeric value of the fractional cointegration parameter.
...	additional arguments affecting the summary produced.

Note

The roots are calculated from the companion form of the VAR, where the roots are given as the inverse eigenvalues of the coefficient matrix.

References

Johansen, S. (2008). "A representation theory for a class of vector autoregressive models for fractional processes," Econometric Theory 24, 651-676.

See Also

FCVARoptions to set default estimation options. FCVARestn to estimate the model for which to calculate the roots of the characteristic polynomial. summary.FCVAR_roots prints the output of GetCharPolyRoots to screen.

Other FCVAR postestimation functions: FCVARboot(), FCVARhypoTest(), GetCharPolyRoots(), MVWNtest(), plot.FCVAR_roots(), summary.MVWN_stats()

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
results <- FCVARestn(x, k = 2, r = 1, opt)
FCVAR_CharPoly <- GetCharPolyRoots(results$coeffs, opt, k = 2, r = 1, p = 3)
summary(object = FCVAR_CharPoly)
graphics::plot(x = FCVAR_CharPoly)
```

---

summary.MVWN_stats *Summarize Statistics for Multivariate White Noise Tests*

Description

summary.MVWN_stats is an S3 method for objects of class MVWN_stats that prints a summary of the statistics from MVWNtest to screen. MVWNtest performs multivariate tests for white noise. It performs both the Ljung-Box Q-test and the LM-test on individual series for a sequence of lag lengths.

Usage

```
## S3 method for class 'MVWN_stats'
summary(object, ...)
```

Arguments

object	An S3 object of type MVWN_stats containing the results from multivariate tests for white noise. It is the output of MVWNtest.
...	additional arguments affecting the summary produced.

Note

The LM test is consistent for heteroskedastic series, the Q-test is not.

See Also

FCVARoptions to set default estimation options. FCVARestn produces the residuals intended for this test. LagSelect uses this test as part of the lag order selection process. summary.MVWN_stats is an S3 method for class MVWN_stats that prints a summary of the output of MVWNtest to screen.

Other FCVAR postestimation functions: FCVARboot(), FCVARhypoTest(), GetCharPolyRoots(), MVWNtest(), plot.FCVAR_roots(), summary.FCVAR_roots()

Examples

```
opt <- FCVARoptions()
opt$gridSearch   <- 0 # Disable grid search in optimization.
opt$dbMin        <- c(0.01, 0.01) # Set lower bound for d,b.
opt$dbMax        <- c(2.00, 2.00) # Set upper bound for d,b.
opt$constrained  <- 0 # Impose restriction dbMax >= d >= b >= dbMin ? 1 <- yes, 0 <- no.
x <- votingJNP2014[, c("lib", "ir_can", "un_can")]
results <- FCVARestn(x, k = 2, r = 1, opt)
MVWNtest_stats <- MVWNtest(x = results$Residuals, maxlag = 12, printResults = 1)
summary(object = MVWNtest_stats)



set.seed(27)
WN <- stats::rnorm(100)
RW <- cumsum(stats::rnorm(100))
MVWN_x <- as.matrix(data.frame(WN = WN, RW = RW))
MVWNtest_stats <- MVWNtest(x = MVWN_x, maxlag = 10, printResults = 1)
summary(object = MVWNtest_stats)
```

---

votingJNP2014 *Aggregate support for Canadian political parties.*

Description

A dataset containing the aggregate support for Canadian political parties and economic indicators from Canada and the United States.

Usage

```
votingJNP2014
```

Format

A data frame with 316 rows and 6 variables:

lib aggregate support for the Liberal party

pc aggregate support for the Conservative party

ir_can Canadian 3-month T-bill rates

ir_us US 3-month T-bill rates

un_can Canadian unemployment rate

un_us US unemployment rate

Source

https://sites.google.com/view/mortennielsen/software

Index