Package ‘vars’

March 22, 2023

Type Package
Title VAR Modelling
Version 1.5-9
Date 2023-03-22
Depends R (>= 2.0.0), MASS, strucchange, urca (>= 1.1-6), lmtest (>= 0.9-26), sandwich (>= 2.2-4)
Imports methods
LazyLoad yes
Description Estimation, lag selection, diagnostic testing, forecasting, causality analysis, forecast error variance decomposition and impulse response functions of VAR models and estimation of SVAR and SVEC models.
License GPL (>= 2)
URL https://www.pfaffikus.de
Author Bernhard Pfaff [aut, cre], Matthieu Stigler [ctb]
Maintainer Bernhard Pfaff <bernhard@pfaffikus.de>
LazyData true
Repository CRAN
Repository/R-Forge/Project vars
Repository/R-Forge/Revision 116
Repository/R-Forge/DateTimeStamp 2023-03-22 21:14:45
Date/Publication 2023-03-22 22:20:03 UTC
NeedsCompilation no

R topics documented:

Acoef ................................................................. 2
arch.test ........................................................... 3
Bcoef ................................................................. 5
Acoef

Coefficient matrices of the lagged endogenous variables

Description

Returns the estimated coefficient matrices of the lagged endogenous variables as a list of matrices each with dimension \((K \times K)\).

Usage

\texttt{Acoef(x)}

Arguments

\texttt{x} \hspace{1cm} An object of class \texttt{'varest'}, generated by \texttt{VAR()}.
arch.test

Details
Given an estimated VAR(p) of the form:
\[
y_t = \hat{A}_1 y_{t-1} + \ldots + \hat{A}_p y_{t-p} + \hat{C} D_t
\]
the function returns the matrices \((\hat{A}_1, \ldots, \hat{A}_p)\) each with dimension \((K \times K)\) as a list object.

Value
A list object with coefficient matrices for the lagged endogenous variables.

Note
This function was named \(A\) in earlier versions of package \texttt{vars}; it is now deprecated. See \texttt{vars-deprecated} too.

Author(s)
Bernhard Pfaff

See Also
\texttt{Bcoef}, \texttt{VAR}

Examples
\begin{verbatim}
data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
Acoef(var.2c)
\end{verbatim}

Description
This function computes univariate and multivariate ARCH-LM tests for a VAR(p).

Usage
\begin{verbatim}
arch.test(x, lags.single = 16, lags.multi = 5, multivariate.only = TRUE)
\end{verbatim}

Arguments
\begin{itemize}
\item \textbf{x} Object of class ‘varest’: generated by \texttt{VAR()}, or an object of class ‘vec2var’;
  generated by \texttt{vec2var()}.
\item \textbf{lags.single} An integer specifying the lags to be used for the univariate ARCH statistics.
\item \textbf{lags.multi} An integer specifying the lags to be used for the multivariate ARCH statistic.
\item \textbf{multivariate.only} If \texttt{TRUE} (the default), only the multivariate test statistic is computed.
\end{itemize}
Details

The multivariate ARCH-LM test is based on the following regression (the univariate test can be considered as special case of the exhibition below and is skipped):

\[ \text{vech}(\hat{u}_t \hat{u}_t^\prime) = \beta_0 + B_1 \text{vech}(\hat{u}_{t-1} \hat{u}_{t-1}^\prime) + \ldots + B_q \text{vech}(\hat{u}_{t-q} \hat{u}_{t-q}^\prime + v_t) \]

whereby \( v_t \) assigns a spherical error process and \( \text{vech} \) is the column-stacking operator for symmetric matrices that stacks the columns from the main diagonal on downwards. The dimension of \( \beta_0 \) is \( \frac{1}{2} K(K+1) \) and for the coefficient matrices \( B_i \) with \( i = 1, \ldots, q \), \( \frac{1}{2} K(K+1) \times \frac{1}{2} K(K+1) \). The null hypothesis is: \( H_0 := B_1 = B_2 = \ldots = B_q = 0 \) and the alternative is: \( H_1 : B_1 \neq 0 \text{ or } B_2 \neq 0 \text{ or } \ldots B_q \neq 0 \). The test statistic is:

\[ VARCH_{LM}(q) = \frac{1}{2} TK(K + 1)R^2_m, \]

with

\[ R^2_m = 1 - \frac{2}{K(K + 1)} \text{tr}(\hat{\Omega}^{-1}_0) \]

and \( \hat{\Omega} \) assigns the covariance matrix of the above defined regression model. This test statistic is distributed as \( \chi^2(qK^2(K + 1)^2/4) \).

Value

A list with class attribute ‘varcheck’ holding the following elements:

- **resid**: A matrix with the residuals of the VAR.
- **arch.uni**: A list with objects of class ‘htest’ containing the univariate ARCH-LM tests per equation. This element is only returned if `multivariate.only = FALSE` is set.
- **arch.mul**: An object with class attribute ‘htest’ containing the multivariate ARCH-LM statistic.

Note

This function was named arch in earlier versions of package `vars`; it is now deprecated. See `vars-deprecated` too.

Author(s)

Bernhard Pfaff

References


See Also

VAR, vec2var, resid.plot

Examples

data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
arch.test(var.2c)

Bcoef

Coefficient matrix of an estimated VAR(p)

Description

Returns the estimated coefficients of a VAR(p) as a matrix.

Usage

Bcoef(x)

Arguments

x
An object of class ‘varest’, generated by VAR().

Details

Given an estimated VAR of the form:

\[ \hat{y}_t = \hat{A}_1 y_{t-1} + \ldots + \hat{A}_p y_{t-p} + \hat{C} D_t \]

the function returns the matrices (\( \hat{A}_1 | \ldots | \hat{A}_p | \hat{C} \)) as a matrix object.

Value

A matrix holding the estimated coefficients of a VAR.

Note

This function was named B in earlier versions of package vars; it is now deprecated. See vars-deprecated too.

Author(s)

Bernhard Pfaff

See Also

Acoef, VAR
Examples

data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
Bcoef(var.2c)

BQ

Estimates a Blanchard-Quah type SVAR

Description

This function estimates a SVAR of type Blanchard and Quah. It returns a list object with class attribute 'svarest'.

Usage

BQ(x)

Arguments

x

Object of class 'varest'; generated by VAR().

Details

For a Blanchard-Quah model the matrix $A$ is set to be an identity matrix with dimension $K$. The matrix of the long-run effects is assumed to be lower-triangular and is defined as:

$$(I_K - A_1 - \cdots - A_p)^{-1}B$$

Hence, the residual of the second equation cannot exert a long-run influence on the first variable and likewise the third residual cannot impact the first and second variable. The estimation of the Blanchard-Quah model is achieved by a Choleski decomposition of:

$$(I_K - \hat{A}_1 - \cdots - \hat{A}_p)^{-1}\hat{\Sigma}_u(I_K - \hat{A}_1' - \cdots - \hat{A}_p')^{-1}$$

The matrices $\hat{A}_i$ for $i = 1, \ldots, p$ assign the reduced form estimates. The long-run impact matrix is the lower-triangular Choleski decomposition of the above matrix and the contemporaneous impact matrix is equal to:

$$(I_K - A_1 - \cdots - A_p)Q$$

where $Q$ assigns the lower-triangular Choleski decomposition.
Value

A list of class ‘svarest’ with the following elements is returned:

- A: An identity matrix.
- Ase: NULL.
- B: The estimated contemporaneous impact matrix.
- Bse: NULL.
- LRIM: The estimated long-run impact matrix.
- Sigma.U: The variance-covariance matrix of the reduced form residuals times 100.
- LR: NULL.
- opt: NULL.
- start: NULL.
- type: Character: “Blanchard-Quah”.
- var: The ‘varest’ object ‘x’.
- call: The call to BQ().

Author(s)

Bernhard Pfaff

References


See Also

- SVAR, VAR

Examples

data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
BQ(var.2c)
Canada: Macroeconomic time series

Description
The original time series are published by the OECD. The sample range is from the 1stQ 1980 until 4thQ 2000. The following series have been utilised in the construction of the series provided in Canada:

Main Economic Indicators:
- Canadian unemployment rate in % 444113DSA
- Canadian manufacturing real wage 444321KSA
- Canadian consumer price index 445241K

Quarterly National Accounts:
- Canadian nominal GDP CAN1008S1

Labour Force Statistics:
- Canadian civil employment in 1000 persons 445005DSA

The series in Canada are constructed as:

\[ \text{prod} := 100*\ln(\frac{\text{CAN1008S1}}{\text{445241K}}) - \ln(\text{445005DSA}) \]
\[ e := 100*\ln(\text{445005DSA}) \]
\[ U := \text{444113DSA} \]
\[ rw := 100*\ln(100*\text{444321KSA}) \]

Hence, prod is used as a measure of labour productivity; e is used for employment; U is the unemployment rate and rw assigns the real wage.

Usage
Canada
causality

Format
An object with class attributes mts and ts containing four variables with 84 observations.

Source

causality Causality Analysis

Description
Computes the test statistics for Granger- and Instantaneous causality for a VAR(p).

Usage
causality(x, cause = NULL, vcov.=NULL, boot=FALSE, boot.runs=100)

Arguments
x Object of class 'varest'; generated by VAR().
cause A character vector of the cause variable(s). If not set, then the variable in the first column of x$y is used as cause variable and a warning is printed.
vcov. a specification of the covariance matrix of the estimated coefficients. This can be specified as a matrix or as a function yielding a matrix when applied to x.
boot Logical. Whether a wild bootstrap procedure should be used to compute the critical values. Default is no
boot.runs Number of bootstrap replications if boot=TRUE

Details
Two causality tests are implemented. The first is a F-type Granger-causality test and the second is a Wald-type test that is characterized by testing for nonzero correlation between the error processes of the cause and effect variables. For both tests the vector of endogenous variables $y_t$ is split into two subvectors $y_{1t}$ and $y_{2t}$ with dimensions $(K_1 \times 1)$ and $(K_2 \times 1)$ with $K = K_1 + K_2$. For the rewritten VAR(p):

$$[y_{1t}, y_{2t}] = \sum_{i=1}^{p} [\alpha_{11,i}, \alpha_{12,i}, [\alpha_{21,i}, \alpha_{22,i}]][y_{1,t-i}, y_{2,t-i}] + CD_t + [u_{1t}, u_{2t}],$$

the null hypothesis that the subvector $y_{1t}$ does not Granger-cause $y_{2t}$, is defined as $\alpha_{21,i} = 0$ for $i = 1, 2, \ldots, p$. The alternative is: $\exists \alpha_{21,i} \neq 0$ for $i = 1, 2, \ldots, p$. The test statistic is distributed as $F(pK_1K_2, KT - n^*)$, with $n^*$ equal to the total number of parameters in the above VAR(p).
The null hypothesis for instantaneous causality is defined as: $H_0: C\sigma = 0$, where $C$ is a $(N \times K(K+1)/2)$ matrix of rank $N$ selecting the relevant co-variances of $u_{1t}$ and $u_{2t}$; $\sigma = vech(\Sigma_u)$. The Wald statistic is defined as:

$$\lambda_W = T\bar{\sigma}'C'[2CD_K^+(\hat{\Sigma}_u \otimes \hat{\Sigma}_u)D_K^+C']^{-1}C\bar{\sigma},$$

hereby assigning the Moore-Penrose inverse of the duplication matrix $D_K$ with $D_K^+$ and $\hat{\Sigma}_u = \frac{1}{T}\sum_{t=1}^{T} \hat{u}_t\hat{u}_t'$. The duplication matrix $D_K$ has dimension $(K^2 \times \frac{1}{2}K(K+1))$ and is defined such that for any symmetric $(K \times K)$ matrix $A$, $vec(A) = D_K vech(A)$ holds. The test statistic $\lambda_W$ is asymptotically distributed as $\chi^2(N)$.

For the Granger causality test, a robust covariance-matrix estimator can be used in case of heteroskedasticity through argument vcov. It can be either a pre-computed matrix or a function for extracting the covariance matrix. See vcovHC from package sandwich for further details.

A wild bootstrap computation (imposing the restricted model as null) of the p values is available through argument boot and boot.runs following Hafner and Herwartz (2009).

Value

A list with elements of class ‘htest’:

- Granger: The result of the Granger-causality test.
- Instant: The result of the instantaneous causality test.

Note

The Moore-Penrose inverse matrix is computed with the function ginv contained in the package ‘MASS’.

The Granger-causality test is problematic if some of the variables are nonstationary. In that case the usual asymptotic distribution of the test statistic may not be valid under the null hypothesis.

Author(s)

Bernhard Pfaff

References


See Also

VAR

Examples

data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
causality(var.2c, cause = "e")

# use a robust HC variance-covariance matrix for the Granger test:
causality(var.2c, cause = "e", vcov.=vcovHC(var.2c))

# use a wild-bootstrap procedure to for the Granger test
## Not run: causality(var.2c, cause = "e", boot=TRUE, boot.runs=1000)

coef

Coefficient method for objects of class varest

Description

Returns the coefficients of a VAR(p)-model for objects generated by VAR(). Thereby the coef-
method is applied to the summary of the list element varresult, which is itself a list of summary.lm-
objects.

Usage

## S3 method for class 'varest'
coef(object, ...)

Arguments

object An object of class 'varest': generated by VAR()

... Currently not used.

Author(s)

Bernhard Pfaff

References


See Also

VAR
Examples

```r
data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
coef(var.2c)
```

fanchart

Fanchart plot for objects of class varprd

Description

Time Series plots of VAR forecasts with differently shaded confidence regions (fanchart) for each endogenous variable.

Usage

```r
fanchart(x, colors = NULL, cis = NULL, names = NULL, main = NULL, ylab = NULL, xlab = NULL, col.y = NULL, plot.type = c("multiple", "single"), mar = par("mar"), oma = par("oma"), ...)
```

Arguments

- **x**: An object of class ‘varprd’; generated by `predict()`.
- **colors**: Character vector of colors to be used for shading. If unset, a gray color scheme is used.
- **cis**: A numeric vector of confidence intervals. If unset the sequence from 0.1 to 0.9 is used (step size 0.1).
- **names**: Character vector, names of variables for fancharts. If unset, all variables are plotted.
- **main**: Character vector, title for fanchart plots.
- **ylab**: Character, ylab for fanchart.
- **xlab**: Character, xlab for fanchart.
- **col.y**: Character, color for plotted time series.
- **plot.type**: Character, if multiple all fancharts appear in one device.
- **nc**: Integer, number of columns if plot.type is multiple.
- **mar**: Vector, setting of margins.
- **oma**: Vector, setting of outer margins.
- **...**: Dot argument, passed to plot.ts.

Author(s)

Bernhard Pfaff
**fevd**

**References**


**See Also**

VAR, predict, plot.par

**Examples**

```r
## Not run:
data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
var.2c.prd <- predict(var.2c, n.ahead = 8, ci = 0.95)
fanchart(var.2c.prd)
## End(Not run)
```

---

### fevd

**Forecast Error Variance Decomposition**

**Description**

Computes the forecast error variance decomposition of a VAR(p) for n.ahead steps.

**Usage**

```r
## S3 method for class 'varest'
fevd(x, n.ahead=10, ...)
## S3 method for class 'svarest'
fevd(x, n.ahead=10, ...)
## S3 method for class 'sveccest'
fevd(x, n.ahead=10, ...)
## S3 method for class 'vec2var'
fevd(x, n.ahead=10, ...)
```

**Arguments**

- `x` Object of class ‘varest’: generated by VAR(), or an object of class ‘svarest’: generated by SVAR(), or an object of class ‘vec2var’: generated by vec2var(), or an object of class ‘sveccest’: generated by SVEC().
- `n.ahead` Integer specifying the steps.
- `...` Currently not used.
Details

The forecast error variance decomposition is based upon the orthogonalised impulse response coefficient matrices $\Psi_h$ and allow the user to analyse the contribution of variable $j$ to the $h$-step forecast error variance of variable $k$. If the orthogonalised impulse responses are divided by the variance of the forecast error $\sigma_k^2(h)$, the resultant is a percentage figure. Formally:

$$\sigma_k^2(h) = \sum_{n=0}^{h-1} (\psi_{k1,n}^2 + \ldots + \psi_{kK,n}^2)$$

which can be written as:

$$\sigma_k^2(h) = \sum_{j=1}^{K} (\psi_{kj,0}^2 + \ldots + \psi_{kj,h-1}^2) .$$

Dividing the term $(\psi_{kj,0}^2 + \ldots + \psi_{kj,h-1}^2)$ by $\sigma_k^2(h)$ yields the forecast error variance decompositions in percentage terms.

Value

A list with class attribute `var.fevd` of length $K$ holding the forecast error variances as matrices.

Author(s)

Bernhard Pfaff

References


See Also

`VAR`, `SVAR`, `vec2var`, `SVEC`, `Phi`, `Psi`, `plot`

Examples

data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
fevd(var.2c, n.ahead = 5)
fit method for objects of class `varest` or `vec2var`

Description

Returns the fitted values of a VAR(p)-model for objects generated by `VAR()` or `vec2var()`. For objects of class `varest` the fitted.values-method is applied to the list element `varresult`, which is itself a list of `lm`-objects.

Usage

```r
## S3 method for class 'varest'
fitted(object, ...)
## S3 method for class 'vec2var'
fitted(object, ...)
```

Arguments

- `object`: An object of class `varest`; generated by `VAR()`, or an object of class `vec2var`; generated by `vec2var()`.
- `...`: Currently not used.

Author(s)

Bernhard Pfaff

References


See Also

- `VAR`, `vec2var`

Examples

```r
data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
fitted(var.2c)
```
### Impulse response function

**Description**

Computes the impulse response coefficients of a VAR(p) (or transformed VECM to VAR(p)) or a SVAR for \(n\text{.ahead}\) steps.

**Usage**

```r
## S3 method for class 'varest'
irf(x, impulse = NULL, response = NULL, n.ahead = 10,
    ortho = TRUE, cumulative = FALSE, boot = TRUE, ci = 0.95,
    runs = 100, seed = NULL, ...)
## S3 method for class 'svarest'
irf(x, impulse = NULL, response = NULL, n.ahead = 10,
    ortho = TRUE, cumulative = FALSE, boot = TRUE, ci = 0.95,
    runs = 100, seed = NULL, ...)
## S3 method for class 'vec2var'
irf(x, impulse = NULL, response = NULL, n.ahead = 10,
    ortho = TRUE, cumulative = FALSE, boot = TRUE, ci = 0.95,
    runs = 100, seed = NULL, ...)
## S3 method for class 'svecest'
irf(x, impulse = NULL, response = NULL, n.ahead = 10,
    ortho = TRUE, cumulative = FALSE, boot = TRUE, ci = 0.95,
    runs = 100, seed = NULL, ...)
```

**Arguments**

- `x`: Object of class `varest`; generated by `VAR()`, or object of class `svarest`; generated by `SVAR()`, or object of class `vec2var`; generated by `vec2var()`, or object of class `svecest`; generated by `SVEC()`.
- `impulse`: A character vector of the impulses, default is all variables.
- `response`: A character vector of the responses, default is all variables.
- `n.ahead`: Integer specifying the steps.
- `ortho`: Logical, if `TRUE` (the default) the orthogonalised impulse response coefficients are computed (only for objects of class `varest`).
- `cumulative`: Logical, if `TRUE` the cumulated impulse response coefficients are computed. The default value is false.
- `boot`: Logical, if `TRUE` (the default) bootstrapped error bands for the impulse response coefficients are computed.
- `ci`: Numeric, the confidence interval for the bootstrapped errors bands.
- `runs`: An integer, specifying the runs for the bootstrap.
- `seed`: An integer, specifying the seed for the `rng` of the bootstrap.
- `...`: Currently not used.
Details

The impulse response coefficients of a VAR(p) for n.ahead steps are computed by utilising either the function \texttt{Phi()} or \texttt{Psi()}. If \texttt{boot = TRUE} (the default), confidence bands for a given width specified by \texttt{ci} are derived from \texttt{runs} bootstrap. Hereby, it is at the users leisure to set a seed for the random number generator.

The standard percentile interval is defined as:

\[ CI_s = [s_{\alpha/2}, s_{1-\alpha/2}] \]

with \( s_{\alpha/2} \) and \( s_{1-\alpha/2} \) are the \( \alpha/2 \) and \( 1 - \alpha/2 \) quantiles of the bootstrap distribution of \( \Psi^* \) or \( \Phi^* \) depending whether \( \text{ortho} = \text{TRUE} \). In case \( \text{cumulative} = \text{TRUE} \), the confidence bands are calculated from the cumulated impulse response coefficients.

Value

A list of class ‘\texttt{varirf}’ with the following elements is returned:

\begin{itemize}
  \item \texttt{irf} A list with matrices for each of the impulse variables containing the impulse response coefficients.
  \item \texttt{Lower} If \texttt{boot = TRUE}, a list with matrices for each of the impulse variables containing the lower bands.
  \item \texttt{Upper} If \texttt{boot = TRUE}, a list with matrices for each of the impulse variables containing the upper bands.
  \item \texttt{response} Character vector holding the names of the response variables.
  \item \texttt{impulse} Character vector holding the names of the impulse variables.
  \item \texttt{ortho} Logical, if \texttt{TRUE}, orthogonalised impulse responses have been computed.
  \item \texttt{cumulative} Logical, if \texttt{TRUE}, cumulated impulse responses have been computed.
  \item \texttt{runs} An integer, specifying the number of bootstrap runs.
  \item \texttt{ci} Numeric, defining the confidence level.
  \item \texttt{boot} Logical, if \texttt{TRUE} bootstrapped error bands have been computed.
  \item \texttt{model} Character, containing ‘\texttt{class(x)}’.
\end{itemize}

Author(s)

Bernhard Pfaff

References


See Also

\texttt{VAR}, \texttt{SVAR}, \texttt{vec2var}, \texttt{SVEC}, \texttt{Phi}, \texttt{Psi}, \texttt{plot}
Examples

```r
data(Canada)
## For VAR
var.2c <- VAR(Canada, p = 2, type = "const")
irf(var.2c, impulse = "e", response = c("prod", "rw", "U"), boot = FALSE)
## For SVAR
amat <- diag(4)
diag(amat) <- NA
svar.a <- SVAR(var.2c, estmethod = "direct", Amat = amat)
irf(svar.a, impulse = "e", response = c("prod", "rw", "U"), boot = FALSE)
```

---

### logLik

**Log-Likelihood method**

Returns the log-Likelihood of a VAR, level-VECM, SVAR or SVEC object.

#### Usage

```r
## S3 method for class 'var'est
logLik(object, ...)  
## S3 method for class 'vec2var'
logLik(object, ...)  
## S3 method for class 'svarest'
logLik(object, ...)  
## S3 method for class 'svecest'
logLik(object, ...)
```

#### Arguments

- `object` An object of class ‘varest’, generated by `VAR()`; or an object of class ‘vec2var’, generated by `vec2var()`; or an object of class ‘svarest’, generated by either `SVAR()` or an object of class ‘svecest’, generated by `SVEC()`.
- `...` Currently not used.

#### Details

The log-likelihood of a VAR or level-VECM model is defined as:

\[
\log l = -\frac{KT}{2} \log 2\pi - \frac{T}{2} \log |\Sigma_u| - \frac{1}{2} \text{tr}(U\Sigma_u^{-1}U')
\]

and for a SVAR/SVEC model the log-likelihood takes the form of:

\[
\log l = -\frac{KT}{2} \log 2\pi + \frac{T}{2} \log |A|^2 - \frac{T}{2} \log |B|^2 - \frac{T}{2} \text{tr}(A'B'^{-1}B^{-1}A\Sigma_u)
\]
normality.test

Value

An object with class attribute logLik.

Author(s)

Bernhard Pfaff

References


See Also

VAR, vec2var, SVAR, SVEC

Examples

data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
logLik(var.2c)

---

normality.test  Normality, multivariate skewness and kurtosis test

Description

This function computes univariate and multivariate Jarque-Bera tests and multivariate skewness and kurtosis tests for the residuals of a VAR(p) or of a VECM in levels.

Usage

```r
normality.test(x, multivariate.only = TRUE)
```

Arguments

- `x`  
  Object of class ‘varest’; generated by VAR(), or an object of class ‘vec2var’;
  generated by vec2var().
- `multivariate.only`  
  If TRUE (the default), only multivariate test statistics are computed.

Details

Multivariate and univariate versions of the Jarque-Bera test are applied to the residuals of a VAR. The multivariate version of this test is computed by using the residuals that are standardized by a Choleski decomposition of the variance-covariance matrix for the centered residuals. Please note, that in this case the test result is dependant upon the ordering of the variables.
Value

A list of class ‘varcheck’ with the following elements is returned:

- **resid**: A matrix of the residuals.
- **jb.uni**: A list of elements with class attribute ‘htest’ containing the univariate Jarque-Bera tests. This element is only returned if `multivariate.only = FALSE` is set.
- **jb.mul**: A list of elements with class attribute ‘htest’ containing the multivariate Jarque-Bera test, the multivariate Skewness and Kurtosis tests.

Note

This function was named `normality` in earlier versions of package `vars`; it is now deprecated. See `vars-deprecated` too.

Author(s)

Bernhard Pfaff

References


See Also

`VAR, vec2var, plot`

Examples

```r
data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
normality.test(var.2c)
```

---

**Description**

Returns the estimated coefficient matrices of the moving average representation of a stable VAR(p), of an SVAR as an array or a converted VECM to VAR.
Usage

```r
## S3 method for class 'varest'
Phi(x, nstep=10, ...)
## S3 method for class 'svarest'
Phi(x, nstep=10, ...)
## S3 method for class 'svecest'
Phi(x, nstep=10, ...)
## S3 method for class 'vec2var'
Phi(x, nstep=10, ...)
```

Arguments

- `x` An object of class `varest`, generated by `VAR()`, or an object of class `svarest`, generated by `SVAR()`, or an object of class `svecest`, generated by `SVEC()`, or an object of class `vec2var`, generated by `vec2var()`.
- `nstep` An integer specifying the number of moving error coefficient matrices to be calculated.
- `...` Currently not used.

Details

If the process $y_t$ is stationary (i.e. $I(0)$), it has a Wold moving average representation in the form of:

$$ y_t = \Phi_0 u_t + \Phi_1 u_{t-1} + \Phi_2 u_{t-2} + \ldots, $$

whith $\Phi_0 = I_k$ and the matrices $\Phi_s$ can be computed recursively according to:

$$ \Phi_s = \sum_{j=1}^{s} \Phi_{s-j} A_j \quad s = 1, 2, \ldots, $$

whereby $A_j$ are set to zero for $j > p$. The matrix elements represent the impulse responses of the components of $y_t$ with respect to the shocks $u_t$. More precisely, the $(i,j)$th element of the matrix $\Phi_s$ mirrors the expected response of $y_{i,t+s}$ to a unit change of the variable $y_{jt}$.

In case of a SVAR, the impulse response matrices are given by:

$$ \Theta_t = \Phi_t A^{-1} B. $$

Albeit the fact, that the Wold decomposition does not exist for nonstationary processes, it is however still possible to compute the $\Phi_t$ matrices likewise with integrated variables or for the level version of a VECM. However, a convergence to zero of $\Phi_t$ as $i$ tends to infinity is not ensured; hence some shocks may have a permanent effect.

Value

An array with dimension $(K \times K \times nstep + 1)$ holding the estimated coefficients of the moving average representation.

Note

The first returned array element is the starting value, i.e., $\Phi_0$. 
Author(s)
Bernhard Pfaff

References

See Also
Psi, VAR, SVAR, vec2var, SVEC

Examples
```r
data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
Phi(var.2c, nstep=4)
```

Description
Plot method for objects with class attribute `varest`, `vec2var`, `varcheck`, `varfevd`, `varirf`, `varprd`, `varstabil`.

Usage
```r
# S3 method for class 'varcheck'
plot(x, names = NULL, main.resid = NULL, main.hist = NULL, main.acf = NULL, main.pacf = NULL, main.acf2 = NULL, main.pacf2 = NULL, ylim.resid = NULL, ylim.hist = NULL, ylim.acf = NULL, ylim.pacf = NULL, lty.resid = NULL, lty.hist = NULL, lty.acf = NULL, lty.pacf = NULL, lwd.resid = NULL, lwd.hist = NULL, lwd.acf = NULL, lwd.pacf = NULL, col.resid = NULL, col.hist = NULL, col.acf = NULL, col.pacf = NULL, mar = par("mar"), oma = par("oma"), ...

# S3 method for class 'varest'
plot(x, names = NULL, main.fit = NULL, main.resid = NULL, main.acf = NULL, main.pacf = NULL, main.acf2 = NULL, main.pacf2 = NULL, ylim.fit = NULL, ylim.resid = NULL, ylim.acf = NULL, ylim.pacf = NULL, lty.fit = NULL, lty.resid = NULL, lty.acf = NULL, lty.pacf = NULL, lwd.fit = NULL, lwd.resid = NULL, lwd.acf = NULL, lwd.pacf = NULL, col.fit = NULL, col.resid = NULL, col.acf = NULL, col.pacf = NULL, ylab.fit = NULL, ylab.resid = NULL, ylab.acf = NULL, ylab.pacf = NULL, xlab.fit = NULL, xlab.resid = NULL, xlab.acf = NULL, xlab.pacf = NULL, mar = par("mar"), oma = par("oma"), adj.mtext = NA, padj.mtext = NA, col.mtext = NA, ...

# S3 method for class 'vec2var'
plot(x, ...)

# S3 method for class 'varfevd'
plot(x, plot.type = c("multiple", "single"), ...)
```
plot

names = NULL, main = NULL, col = NULL, ylim = NULL, ylab = NULL,
xlab = NULL, legend = NULL, names.arg = NULL, nc,
mar = par("mar"), oma = par("oma"), addbars = 1, ...
## S3 method for class 'varirf'
plot(x, plot.type = c("multiple", "single"), names =
NULL, main = NULL, sub = NULL, lty = NULL, lwd = NULL, col = NULL, ylim
= NULL, ylab = NULL, xlab = NULL, nc, mar.multi = c(0, 4, 0, 4),
oma.multi = c(6, 4, 6, 4), adj.mtext = NA, padj.mtext = NA, col.mtext =
NA, ...)
## S3 method for class 'varprd'
plot(x, plot.type = c("multiple", "single"),
names = NULL, main = NULL, col = NULL, ylim = NULL, lwd = NULL,
sub = NULL, lty = NULL, col = NULL, ylim = NULL, ylab = NULL, xlab = NULL, nc, mar = par("mar"),
oma = par("oma"), ...
## S3 method for class 'varstabil'
plot(x, plot.type = c("multiple", "single"), names =
NULL, main = NULL, nc, mar = par("mar"), oma = par("oma"), ...

Arguments

addbars Integer, number of empty bars in barplot to reserve space for legend. If set to zero, no legend will be returned.
adj.mtext Adjustment for mtext(), only applicable if plot.type = "multiple".
col Character vector, colors to use in plot.
col.edf Character, color of residuals’ EDF.
col.fit Character vector, colors for diagram of fit.
col.mtext Character, color for mtext(), only applicable if plot.type = "multiple".
col.resid Character vector, colors for residual plot.
lag.acf Integer, lag.max for ACF of residuals.
lag.acf2 Integer, lag.max for ACF of squared residuals.
lag.pacf Integer, lag.max for PACF of residuals.
lag.pacf2 Integer, lag.max for PACF of squared residuals.
legend Character vector of names in legend.
lty Integer/Character, the line types.
lty.fit Vector, lty for diagram of fit.
lty.resid Vector, lty for residual plot.
lwd The width of the lines.
lwd.fit Vector, lwd for diagram of fit.
lwd.resid Vector, lwd for residual plot.
main Character vector, the titles of the plot.
main.acf Character vector, main for residuals’ ACF.
main.acf2 Character vector, main for squared residuals’ ACF.
main.fit        Character vector, main for diagram of fit.
main.hist       Character vector, main for histogram of residuals.
main.pacf       Character vector, main for residuals’ PACF.
main.pacf2      Character vector, main for squared residuals’ PACF.
main.resid      Character vector, main for residual plot.
mar             Setting of margins.
mar.multi       Setting of margins, if plot.type = "multiple".
names           Character vector, the variables names to be plotted. If left NULL, all variables are plotted.
names.arg       Character vector, names for x-axis of barplot.
nc              Integer, number of columns for multiple plot.
oma             Setting of outer margins.
oma.multi       Setting of margins, if plot.type = "multiple".
padj.mtext      Adjustment for mtext(), only applicable if plot.type = "multiple".
plot.type       Character, if multiple all plots are drawn in a single device, otherwise the plots are shown consecutively.
sub             Character, sub title in plot.
x              An object of one of the above classes.
xlab           Character vector signifying the labels for the x-axis.
xlab.acf       Character, xlab for ACF and PACF of residuals and their squares in plot.varcheck.
xlab.fit       Character vector, xlab for diagram of fit.
xlab.resid     Character vector, xlab for residual plot.
ylab           Character vector signifying the labels for the y-axis.
ylab.acf       Character, ylab for ACF.
ylab.fit       Character vector, ylab for diagram of fit.
ylab.pacf      Character, ylab for PACF
ylab.resid     Character vector, ylab for residual plot.
ylim           Vector, the limits of the y-axis.
ylim.fit       Vector, ylim for diagram of fit.
ylim.hist      Vector, ylim for histogram of residuals.
ylim.resid     Vector, ylim for residual plot.
...            Passed to internal plot function.

Details
The plot-method for objects with class attribute vec2var is the same as for objects with class attribute varest. Hence, the same arguments can be utilised.

Author(s)
Bernhard Pfaff
Predict method for objects of class varest and vec2var

Forecating a VAR object of class ‘varest’ or of class ‘vec2var’ with confidence bands.
Usage

## S3 method for class 'varest'
predict(object, ..., n.ahead = 10, ci = 0.95, dumvar = NULL)
## S3 method for class 'vec2var'
predict(object, ..., n.ahead = 10, ci = 0.95, dumvar = NULL)

Arguments

- **object**: An object of class `varest`; generated by `VAR()`, or an object of class `vec2var`; generated by `vec2var()`.
- **n.ahead**: An integer specifying the number of forecast steps.
- **ci**: The forecast confidence interval.
- **dumvar**: Matrix for objects of class `vec2var` or `varest`, if the `dumvar` argument in `ca.jo()` has been used or if the `exogen` argument in `VAR()` has been used, respectively. The matrix should have the same column dimension as in the call to `ca.jo()` or to `VAR()` and row dimension equal to `n.ahead`.
- **...**: Currently not used.

Details

The `n.ahead` forecasts are computed recursively for the estimated VAR, beginning with \( h = 1, 2, \ldots, n.ahead \):

\[
y_{T+1|T} = A_1 y_T + \ldots + A_p y_{T+p} + C D_{T+1}
\]

The variance-covariance matrix of the forecast errors is a function of \( \Sigma_u \) and \( \Phi_s \).

Value

A list with class attribute `varprd` holding the following elements:

- **fcst**: A list of matrices per endogenous variable containing the forecasted values with lower and upper bounds as well as the confidence interval.
- **endog**: Matrix of the in-sample endogenous variables.
- **model**: The estimated VAR object.
- **exo.fcst**: If applicable provided values of exogenous variables, otherwise `NULL`.

Author(s)

Bernhard Pfaff

References

See Also

VAR, vec2var, plot, fanchart

Examples

data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
predict(var.2c, n.ahead = 8, ci = 0.95)

Psi  Coefficient matrices of the orthogonalised MA representation

Description

Returns the estimated orthogonalised coefficient matrices of the moving average representation of a stable VAR(p) as an array.

Usage

## S3 method for class 'varest'
Psi(x, nstep=10, ...)
## S3 method for class 'vec2var'
Psi(x, nstep=10, ...)

Arguments

x  An object of class ‘varest’, generated by VAR(), or an object of class ‘vec2var’, generated by vec2var().
nstep  An integer specifying the number of orthogonalised moving error coefficient matrices to be calculated.
...  Dots currently not used.

Details

In case that the components of the error process are instantaneously correlated with each other, that is: the off-diagonal elements of the variance-covariance matrix $\Sigma_u$ are not null, the impulses measured by the $\Phi_s$ matrices, would also reflect disturbances from the other variables. Therefore, in practice a Choleski decomposition has been propagated by considering $\Sigma_u = PP'$ and the orthogonalised shocks $\epsilon_t = P^{-1}u_t$. The moving average representation is then in the form of:

$$y_t = \Psi_0 \epsilon_t + \Psi_1 \epsilon_{t-1} + \Psi_2 \epsilon_{t-2} + \ldots,$$

whith $\Psi_0 = P$ and the matrices $\Psi_s$ are computed as $\Psi_s = \Phi_s P$ for $s = 1, 2, 3, \ldots$.

Value

An array with dimension $(K \times K \times nstep + 1)$ holding the estimated orthogonalised coefficients of the moving average representation.
Note
The first returned array element is the starting value, \( i.e., \Psi_0 \). Due to the utilisation of the Choleski decomposition, the impulse are now dependent on the ordering of the vector elements in \( \mathbf{y}_t \).

Author(s)
Bernhard Pfaff

References

See Also
Phi, VAR, SVAR, vec2var

Examples
```r
data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
Psi(var.2c, nstep=4)
```

residuals

*Residuals method for objects of class varest and vec2var*

Description
Returns the residuals of a VAR(p)-model or for a VECM in levels. For the former class the residuals-method is applied to the list element \( \text{varresult} \), which is itself a list of \( \text{lm} \)-objects.

Usage
```r
## S3 method for class 'varest'
residuals(object, ...)
## S3 method for class 'vec2var'
residuals(object, ...)
```

Arguments
- **object** An object of class ‘varest’; generated by \( \text{VAR}() \), or an object of class ‘vec2var’; generated by \( \text{vec2var}() \)
- **...** Currently not used.

Author(s)
Bernhard Pfaff
References


See Also

VAR, vec2var

Examples

```r
## Not run:
data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
resid(var.2c)
## End(Not run)
```

restrict

**Restricted VAR**

Description

Estimation of a VAR, by imposing zero restrictions manually or by significance.

Usage

`restrict(x, method = c("ser", "manual"), thresh = 2.0, resmat = NULL)`

Arguments

- `x`: An object of class ‘varest’ generated by `VAR()`.
- `method`: A character, choosing the method
- `thresh`: If method `ser`: the threshold value for the t-statistics.

Details

Given an estimated VAR object of class ‘varest’, a restricted VAR can be obtained by either choosing method `ser` or `manual`. In the former case, each equation is re-estimated separately as long as there are t-values that are in absolut value below the threshold value set by the function’s argument `thresh`. In the latter case, a restriction matrix has to be provided that consists of 0/1 values, thereby selecting the coefficients to be retained in the model.
Value

A list with class attribute ‘varest’ holding the following elements:

- varresult: list of ‘lm’ objects.
- datamat: The data matrix of the endogenous and explanatory variables.
- y: The data matrix of the endogenous variables.
- type: A character, specifying the deterministic regressors.
- p: An integer specifying the lag order.
- K: An integer specifying the dimension of the VAR.
- obs: An integer specifying the number of used observations.
- totobs: An integer specifying the total number of observations.
- restrictions: The matrix object containing the zero restrictions provided as argument resmat.
- call: The call to VAR().

Note

Currently, the restricted VAR is estimated by OLS and not by an efficient EGLS-method.

Author(s)

Bernhard Pfaff

References


See Also

VAR

Examples

data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
## Restrictions determined by thresh
restrict(var.2c, method = "ser")
## Restrictions set manually
restrict <- matrix(c(1, 1, 1, 1, 1, 0, 0, 0,
                     1, 0, 1, 0, 1, 0, 1, 1,
                     0, 0, 1, 1, 0, 1, 0, 1,
                     1, 1, 1, 0, 1, 1, 0, 1),
                   nrow=4, ncol=9, byrow=TRUE)
restrict(var.2c, method = "man", resmat = restrict)
roots

Eigenvalues of the companion coefficient matrix of a VAR(p)-process

Description

Returns a vector of the eigenvalues of the companion coefficient matrix.

Usage

roots(x, modulus = TRUE)

Arguments

x An object of class ‘varest’, generated by VAR().
modulus Logical, set to TRUE for returning the modulus.

Details

Any VAR(p)-process can be written in a first-order vector autoregressive form: the companion form. A VAR(p)-process is stable, if its reverse characteristic polynomial:

$$\det(I_K - A_1 z - \cdots - A_p z^p) \neq 0 \text{ for } |z| \leq 1,$$

has no roots in or on the complex circle. This is equivalent to the condition that all eigenvalues of the companion matrix $A$ have modulus less than 1. The function roots(), does compute the eigenvalues of the companion matrix $A$ and returns by default their moduli.

Value

A vector object with the eigen values of the companion matrix, or their moduli (default).

Author(s)

Bernhard Pfaff

References


See Also

VAR

Examples

data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
roots(var.2c)
serial.test

Test for serially correlated errors

**Description**

This function computes the multivariate Portmanteau- and Breusch-Godfrey test for serially correlated errors.

**Usage**

```r
serial.test(x, lags.pt = 16, lags.bg = 5, type = c("PT.asymptotic",
"PT.adjusted", "BG", "ES") )
```

**Arguments**

- `x`: Object of class ‘varest’; generated by `VAR()`, or an object of class ‘vec2var’; generated by `vec2var()`.
- `lags.pt`: An integer specifying the lags to be used for the Portmanteau statistic.
- `lags.bg`: An integer specifying the lags to be used for the Breusch-Godfrey statistic.
- `type`: Character, the type of test. The default is an asymptotic Portmanteau test.

**Details**

The Portmanteau statistic for testing the absence of up to the order \( h \) serially correlated disturbances in a stable \( \text{VAR}(p) \) is defined as:

\[
Q_h = T \sum_{j=1}^{h} \text{tr}(\hat{C}_j' \hat{C}_0^{-1} \hat{C}_j \hat{C}_0^{-1})
\]

where \( \hat{C}_j = \frac{1}{T} \sum_{t=i+1}^{T} \hat{u}_t \hat{u}'_{t-i} \). The test statistic is approximately distributed as \( \chi^2(K^2(h - p)) \). This test statistic is chosen by setting `type = "PT.asymptotic"`. For smaller sample sizes and/or values of \( h \) that are not sufficiently large, a corrected test statistic is computed as:

\[
Q_h^* = T^2 \sum_{j=1}^{h} \frac{1}{T-j} \text{tr}(\hat{C}_j' \hat{C}_0^{-1} \hat{C}_j \hat{C}_0^{-1})
\]

This test statistic can be accessed, if `type = "PT.adjusted"` is set.

The Breusch-Godfrey LM-statistic is based upon the following auxiliary regressions:

\[
\hat{u}_t = A_1 y_{t-1} + \ldots + A_p y_{t-p} + CD_t + B_1 \hat{u}_{t-1} + \ldots + B_h \hat{u}_{t-h} + \varepsilon_t
\]

The null hypothesis is: \( H_0 : B_1 = \ldots = B_h = 0 \) and correspondingly the alternative hypothesis is of the form \( H_1 : \exists B_i \neq 0 \) for \( i = 1, 2, \ldots, h \). The test statistic is defined as:

\[
LM_h = T(K - \text{tr}(\hat{\Sigma}_R^{-1} \hat{\Sigma}_e))
\]
where $\tilde{\Sigma}_R$ and $\tilde{\Sigma}_e$ assign the residual covariance matrix of the restricted and unrestricted model, respectively. The test statistic $LM_h$ is distributed as $\chi^2(hK^2)$. This test statistic is calculated if type = "BG" is used.

Edgerton and Shukur (1999) proposed a small sample correction, which is defined as:

$$LMF_h = \frac{1 - (1 - R^2)^{1/r}}{(1 - R^2)^{1/r}} \frac{Nr - q}{Km},$$

with $R^2 = 1 - |\tilde{\Sigma}_e|/|\tilde{\Sigma}_R|$, $r = ((K^2m^2 - 4)/(K^2 + m^2 - 5))^{1/2}$, $q = 1/2Km - 1$ and $N = T - K - m - 1/2(K - m + 1)$, whereby $n$ is the number of regressors in the original system and $m = Kh$. The modified test statistic is distributed as $F(hK^2, int(Nr - q))$. This modified statistic will be returned, if type = "ES" is provided in the call to serial().

**Value**

A list with class attribute ‘var.check’ holding the following elements:

- resid A matrix with the residuals of the VAR.
- pt.mul A list with objects of class attribute ‘htest’ containing the multivariate Portmanteau-statistic (asymptotic and adjusted).
- LMh An object with class attribute ‘htest’ containing the Breusch-Godfrey LM-statistic.
- LMFh An object with class attribute ‘htest’ containing the Edgerton-Shukur F-statistic.

**Note**

This function was named serial in earlier versions of package vars; it is now deprecated. See vars-deprecated too.

**Author(s)**

Bernhard Pfaff

**References**

See Also

VAR, vec2var, plot

Examples

data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
serial.test(var.2c, lags.pt = 16, type = "PT.adjusted")

stability

Structural stability of a VAR(p)

descriptions

Computes an empirical fluctuation process according to a specified method from the generalised fluctuation test framework. The test utilises the function efp() and its methods from package 'strucchange'.

Usage

## Default S3 method:
stability(x, type = c("OLS-CUSUM", "Rec-CUSUM", "Rec-MOSUM", "OLS-MOSUM", "RE", "ME", "Score-CUSUM", "Score-MOSUM", "fluctuation"), h = 0.15, dynamic = FALSE, rescale = TRUE, ...)
## S3 method for class 'varest'
stability(x, type = c("OLS-CUSUM", "Rec-CUSUM", "Rec-MOSUM", "OLS-MOSUM", "RE", "ME", "Score-CUSUM", "Score-MOSUM", "fluctuation"), h = 0.15, dynamic = FALSE, rescale = TRUE, ...)

Arguments

x

Object of class 'varest': generated by VAR().

type

Specifies which type of fluctuation process will be computed, the default is 'OLS-CUSUM'. For details see: efp.

h

A numeric from interval (0,1) specifying the bandwidth. Determines the size of the data window relative to sample size (for 'MOSUM' and 'ME' processes only).

dynamic

Logical. If 'TRUE' the lagged observations are included as a regressor.

rescale

Logical. If 'TRUE' the estimates will be standardized by the regressor matrix of the corresponding subsample; if 'FALSE' the whole regressor matrix will be used. (only if 'type' is either 'RE' or 'E').

...

Ellipsis, is passed to strucchange::sctest(), as default.

details

For details, please refer to documentation efp.
Value

A list with class attribute ‘varstabil’ holding the following elements:

- `stability`: A list with objects of class ‘efp’; length is equal to the dimension of the VAR.
- `names`: Character vector containing the names of the endogenous variables.
- `K`: An integer of the VAR dimension.

Author(s)

Bernhard Pfaff

References


and see the references provided in the reference section of *efp*, too.

See Also

VAR, plot, efp

Examples

data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
var.2c.stabil <- stability(var.2c, type = "OLS-CUSUM")
var.2c.stabil
## Not run:
plot(var.2c.stabil)
## End(Not run)
Usage

```r
## S3 method for class 'varest'
summary(object, equations = NULL, ...)
## S3 method for class 'varsum'
print(x, digits = max(3,getOption("digits") - 3),
signif.stars = getOption("show.signif.stars"), ...)
## S3 method for class 'svarest'
summary(object, ...)
## S3 method for class 'svrsum'
print(x, digits = max(3,getOption("digits") - 3), ...)
## S3 method for class 'svcecest'
summary(object, ...)
## S3 method for class 'svcecsum'
print(x, digits = max(3,getOption("digits") - 3), ...)
```

Arguments

- `object`: Object of class `varest`, usually, a result of a call to `VAR`, or object of class `svarest`, usually, a result of a call to `SVAR`, or object of class `svecest`, usually, a result of a call to `SVEC`.
- `equations`: Character vector of endogenous variable names for which summary results should be returned. The default is `NULL` and results are returned for all equations in the `VAR`.
- `x`: Object with class attribute `varsum`, `svrsum`.
- `digits`: the number of significant digits to use when printing.
- `signif.stars`: logical. If `TRUE`, ‘significance stars’ are printed for each coefficient.
- `...`: further arguments passed to or from other methods.

Value

Returns either a list with class attribute `varsum` which contains the following elements:

- `names`: Character vector with the names of the endogenous correlation matrix of `VAR` residuals.
- `logLik`: Numeric, value of log Likelihood.
- `obs`: Integer, sample size.
- `roots`: Vector, roots of the characteristic polynomial.
- `type`: Character vector, deterministic regressors included in `VAR`.
- `call`: Call, the initial call to `VAR`.

Or a list with class attribute `svrsum` which contains the following elements:

- `type`: Character, the type of `SVAR`-model.
- `A`: Matrix, estimated coefficients for `A` matrix.
- `B`: Matrix, estimated coefficients for `B` matrix.
Ase: Matrix, standard errors for A matrix.
Bse: Matrix, standard errors for B matrix.
LRIM: Matrix, long-run impact coefficients for BQ.
Sigma.U: Matrix, variance/covariance of reduced form residuals.
logLik: Numeric, value of log-Likelihood.
LR: htest, LR result of over-identification test.
obs: Integer, number of observations used.
opt: List, result of optim().
iter: Integer, the count of iterations.
call: Call, the call to SVAR().

Or a list with class attribute svecsum which contains the following elements:
type: Character, the type of SVEC-model.
SR: Matrix, contemporaneous impact matrix.
LR: Matrix, long-run impact matrix.
SRse: Matrix, standard errors for SR matrix.
LRse: Matrix, standard errors for LR matrix.
Sigma.U: Matrix, variance/covariance of reduced form residuals.
logLik: Numeric, value of log-Likelihood.
LRover: htest, LR result of over-identification test.
obs: Integer, number of observations used.
r: Integer, co-integration rank of VECM.
iter: Integer, the count of iterations.
call: Call, the call to SVEC().

Author(s)
Bernhard Pfaff

See Also
VAR, SVAR, SVEC

Examples
```r
data(Canada)
## summary-method for varest
var.2c <- VAR(Canada, p = 2, type = "const")
summary(var.2c)
## summary-method for svarest
amat <- diag(4)
diag(amat) <- NA
amat[2, 1] <- NA
```
amat[4, 1] <- NA

## Estimation method scoring
svar.a <- SVAR(x = var.2c, estmethod = "scoring", Amat = amat, Bmat = NULL,
max.iter = 100, maxls = 1000, conv.crit = 1.0e-8)
summary(svar.a)

## summary-method for svecest
vecm <- ca.jo(Canada[, c("prod", "e", "U", "rw")], type = "trace",
ecdet = "trend", K = 3, spec = "transitory")
SR <- matrix(NA, nrow = 4, ncol = 4)
SR[4, 2] <- 0
LR <- matrix(NA, nrow = 4, ncol = 4)
LR[1, 2:4] <- 0
LR[2:4, 4] <- 0
svec.b <- SVEC(vecm, LR = LR, SR = SR, r = 1, lrtest = FALSE, boot = FALSE)
summary(svec.b)

---

**SVAR**

*Estimation of a SVAR*

**Description**

Estimates an SVAR (either ‘A-model’, ‘B-model’ or ‘AB-model’) by using a scoring algorithm or by directly minimising the negative log-likelihood with `optim()`.

**Usage**

```r
SVAR(x, estmethod = c("scoring", "direct"), Amat = NULL, Bmat = NULL,
start = NULL, max.iter = 100, conv.crit = 0.1e-6, maxls = 1.0,
lrtest = TRUE, ...)
```

**Arguments**

- **x** Object of class `varest`; generated by `VAR()`.
- **estmethod** Character, either scoring for estimating the SVAR-model with the scoring algorithm (default), or directly minimizing the negative log-likelihood.
- **start** Vector with starting values for the parameters to be optimised.
- **lrtest** Logical, over-identification LR test, the result is set to `NULL` for just-identified system.
- **max.iter** Integer, maximum number of iteration (used if estmethod = "scoring").
- **conv.crit** Real, convergence value of algorithm (used if estmethod = "scoring").
- **maxls** Real, maximum movement of the parameters between two iterations of the scoring algorithm (used if estmethod = "scoring").
- **Amat** Matrix with dimension \((K \times K)\) for A- or AB-model.
SVAR

\texttt{Bmat} \quad \text{Matrix with dimension } (K \times K) \text{ for B- or AB-model.}

\texttt{digits} \quad \text{the number of significant digits to use when printing.}

\ldots \quad \text{further arguments passed to or from other methods.}

\textbf{Details}

Consider the following structural form of a K-dimensional vector autoregressive model:

\begin{equation}
Ay_t = A_1 y_{t-1} + \ldots + A_p y_{t-p} + C D_t + B \varepsilon_t
\end{equation}

The coefficient matrices \((A_1 | \ldots | A_p | C)\) might now differ from the ones of a VAR (see \texttt{VAR}). One can now impose restrictions on 'A' and/or 'B', resulting in an 'A-model' or 'B-model' or if the restrictions are placed on both matrices, an 'AB-model'. In case of a SVAR 'A-model', \(B = I_K\) and conversely for a SVAR 'B-model'. Please note that for either an 'A-model' or 'B-model', \(K(K-1)/2\) restrictions have to be imposed, such that the models’ coefficients are identified. For an 'AB-model' the number of restrictions amounts to: \(K^2 + K(K-1)/2\).

For an 'A-model' a \((K \times K)\) matrix has to be provided for the functional argument 'Amat' and the functional argument 'Bmat' must be set to 'NULL' (the default). Hereby, the to be estimated elements of 'Amat' have to be set as 'NA'. Conversely, for a 'B-model' a matrix object with dimension \((K \times K)\) with elements set to 'NA' at the positions of the to be estimated parameters has to be provided and the functional argument 'Amat' is 'NULL' (the default). Finally, for an 'AB-model' both arguments, 'Amat' and 'Bmat', have to be set as matrix objects containing desired restrictions and 'NA' values. The parameters are estimated by minimising the negative of the concentrated log-likelihood function:

\begin{equation}
\ln L_c(A, B) = -\frac{KT}{2} \ln(2\pi) + \frac{T}{2} \ln |A|^2 - \frac{T}{2} \ln |B|^2 - \frac{T}{2} \text{tr}(A'B'^{-1}B^{-1}A\hat{\Sigma}_u)
\end{equation}

Two alternatives are implemented for this: a scoring algorithm or direct minimization with \texttt{optim()}. If the latter is chosen, the standard errors are returned if \texttt{SVAR()} is called with 'hessian = TRUE'.

If 'start' is not set, then 0.1 is used as starting values for the unknown coefficients.

The reduced form residuals can be obtained from the above equation via the relation: \(u_t = A^{-1} B \varepsilon_t\), with variance-covariance matrix \(\Sigma_U = A^{-1} BB'A^{-1}\).

Finally, in case of an overidentified SVAR, a likelihood ratio statistic is computed according to:

\begin{equation}
LR = T(\ln \det(\hat{\Sigma}_u^r) - \ln \det(\hat{\Sigma}_u))
\end{equation}

with \(\hat{\Sigma}_u^r\) being the restricted variance-covariance matrix and \(\hat{\Sigma}_u\) being the variance covariance matrix of the reduced form residuals. The test statistic is distributed as \(\chi^2(nr - 2K^2 - \frac{1}{2}K(K + 1))\), where \(nr\) is equal to the number of restrictions.
Value

A list of class ‘svarest’ with the following elements is returned:

A        If A- or AB-model, the matrix of estimated coefficients.
Ase      The standard errors of ‘A’.
B        If A- or AB-model, the matrix of estimated coefficients.
Bse      The standard errors of ‘B’.
LRIM     For Blanchard-Quah estimation LRIM is the estimated long-run impact matrix; for all other SVAR models LRIM is NULL.
Sigma.U  The variance-covariance matrix of the reduced form residuals times 100, i.e., \( \Sigma_U = A^{-1}BB'A^{-1}' \times 100 \).
LR        Object of class ‘htest’, holding the Likelihood ratio overidentification test.
opt       List object returned by optim().
start     Vector of starting values.
type      SVAR-type, character, either ‘A-model’, ‘B-model’ or ‘AB-model’.
var       The ‘varest’ object ‘x’.
iter      Integer, the count of iterations.
call      The call to SVAR().

Author(s)

Bernhard Pfaff

References


See Also

VAR, SVEC, logLik, irf, fevd
Examples

```r
data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
amat <- diag(4)
diag(amat) <- NA
amat[2, 1] <- NA
amat[4, 1] <- NA
## Estimation method scoring
SVAR(x = var.2c, estmethod = "scoring", Amat = amat, Bmat = NULL,
      max.iter = 100, maxls = 1000, conv.crit = 1.0e-8)
## Estimation method direct
SVAR(x = var.2c, estmethod = "direct", Amat = amat, Bmat = NULL,
hessian = TRUE, method="BFGS")
```

---

SVEC  
Estimation of a SVEC

Description

Estimates an SVEC by utilising a scoring algorithm.

Usage

```r
SVEC(x, LR = NULL, SR = NULL, r = 1, start = NULL, max.iter = 100,
      conv.crit = 1e-07, maxls = 1.0, lrtest = TRUE, boot = FALSE, runs = 100)
```

S3 method for class ‘svecest’

```r
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

- `x`: Object of class ‘ca.jo’; generated by `ca.jo()` contained in `urca`.
- `LR`: Matrix of the restricted long run impact matrix.
- `SR`: Matrix of the restricted contemporaneous impact matrix.
- `r`: Integer, the cointegration rank of `x`.
- `start`: Vector of starting values for `\gamma`.
- `max.iter`: Integer, maximum number of iteration.
- `conv.crit`: Real, convergence value of algorithm.
- `maxls`: Real, maximum movement of the parameters between two iterations of the scoring algorithm.
- `lrtest`: Logical, over-identification LR test, the result is set to `NULL` for just-identified system.
- `boot`: Logical, if `TRUE`, standard errors of the parameters are computed by bootstrapping. Default is `FALSE`.
- `runs`: Integer, number of bootstrap replications.
- `digits`: the number of significant digits to use when printing.
- `...`: further arguments passed to or from other methods.
Details

Consider the following reduced form of a k-dimensional vector error correction model:

\[ A \Delta y_t = \Pi y_{t-1} + \Gamma_1 \Delta y_{t-1} + \ldots + \Gamma_p \Delta y_{t-p+1} + u_t. \]

This VECM has the following MA representation:

\[ y_t = \Xi \sum_{i=1}^l u_i + \Xi^*(L) u_t + y^*_0, \]

with \( \Xi = \beta \perp (\alpha' \perp (I - \sum_{i=1}^{p-1} \Gamma_i) \beta \perp)^{-1} \alpha' \perp \) and \( \Xi^*(L) \) signifies an infinite-order polynomial in the lag operator with coefficient matrices \( \Xi^*_j \) that tends to zero with increasing size of \( j \).

Contemporaneous restrictions on the impact matrix \( B \) must be supplied as zero entries in \( SR \) and free parameters as NA entries. Restrictions on the long run impact matrix \( \Xi B \) have to be supplied likewise. The unknown parameters are estimated by maximising the concentrated log-likelihood subject to the imposed restrictions by utilising a scoring algorithm on:

\[
\ln L_c(A, B) = -\frac{KT}{2} \ln(2\pi) + \frac{T}{2} \ln |A|^2 - \frac{T}{2} \ln |B|^2 - \frac{T}{2} \text{tr}(A'B^{-1}B^{-1}A\tilde{\Sigma}_u) \]

with \( \tilde{\Sigma}_u \) signifies the reduced form variance-covariance matrix and \( A \) is set equal to the identity matrix \( I_K \).

If ‘start’ is not set, then normal random numbers are used as starting values for the unknown coefficients. In case of an overidentified SVEC, a likelihood ratio statistic is computed according to:

\[ LR = T(\ln \det(\tilde{\Sigma}_u) - \ln \det(\tilde{\Sigma}_u)) \]

with \( \tilde{\Sigma}_u^r \) being the restricted variance-covariance matrix and \( \tilde{\Sigma}_u \) being the variance covariance matrix of the reduced form residuals. The test statistic is distributed as \( \chi^2(K \times (K+1)/2 - nr) \), where \( nr \) is equal to the number of restrictions.

Value

A list of class ‘sveccest’ with the following elements is returned:

- **SR** The estimated contemporaneous impact matrix.
- **SRse** The standard errors of the contemporaneous impact matrix, if \( \text{boot} = \text{TRUE} \).
- **LR** The estimated long run impact matrix.
- **LRse** The standard errors of the long run impact matrix, if \( \text{boot} = \text{TRUE} \).
- **Sigma.U** The variance-covariance matrix of the reduced form residuals times 100, \( \Sigma_U = A^{-1}BB'A^{-1'} \times 100 \).
- **Restrictions** Vector, containing the ranks of the restricted long run and contemporaneous impact matrices.
LRover Object of class 'htest', holding the Likelihood ratio overidentification test.

start Vector of used starting values.

type Character, type of the SVEC-model.

var The 'ca.jo' object 'x'.

LRorig The supplied long run impact matrix.

SRorig The supplied contemporaneous impact matrix.

r Integer, the supplied cointegration rank.

iter Integer, the count of iterations.

call The call to SVEC().

Author(s)

Bernhard Pfaff

References


See Also

SVAR, irf, fevd

Examples

data(Canada)
vecm <- ca.jo(Canada[, c("prod", "e", "U", "rw")], type = "trace",
            edet = "trend", K = 3, spec = "transitory")
SR <- matrix(NA, nrow = 4, ncol = 4)
SR[4, 2] <- 0
SR
LR <- matrix(NA, nrow = 4, ncol = 4)
LR[1, 2:4] <- 0
LR[2:4, 4] <- 0
LR
SVEC(vecm, LR = LR, SR = SR, r = 1, lrtest = FALSE, boot = FALSE)
VAR

Estimation of a VAR($p$)

Description

Estimation of a VAR by utilising OLS per equation.

Usage

VAR(y, p = 1, type = c("const", "trend", "both", "none"),
  season = NULL, exogen = NULL, lag.max = NULL,
  ic = c("AIC", "HQ", "SC", "FPE"))

## S3 method for class 'varest'
print(x, digits = max(3, getOption("digits") - 3), ...)

Arguments

y        Data item containing the endogenous variables
p        Integer for the lag order (default is p=1).
type     Type of deterministic regressors to include.
season   Inclusion of centered seasonal dummy variables (integer value of frequency).
exogen   Inclusion of exogenous variables.
lag.max  Integer, determines the highest lag order for lag length selection according to
          the choosen ic.
ic       Character, selects the information criteria, if lag.max is not NULL.
x        Object with class attribute 'varest'.
digits   the number of significant digits to use when printing.
...      further arguments passed to or from other methods.

Details

Estimates a VAR by OLS per equation. The model is of the following form:

$$y_t = A_1 y_{t-1} + \ldots + A_p y_{t-p} + C D_t + u_t$$

where $y_t$ is a $K \times 1$ vector of endogenous variables and $u_t$ assigns a spherical disturbance term of the same dimension. The coefficient matrices $A_1, \ldots, A_p$ are of dimension $K \times K$. In addition, either a constant and/or a trend can be included as deterministic regressors as well as centered seasonal dummy variables and/or exogenous variables (term $CD_T$, by setting the type argument to the corresponding value and/or setting season to the desired frequency (integer) and/or providing a matrix object for exogen, respectively. The default for type is const and for season and exogen the default is set to NULL.

If for lag.max an integer value is provided instead of NULL (the default), the lag length is determined by the selected information criteria in ic, the default is Akaike.
VAR

Value

A list with class attribute ‘varest’ holding the following elements:

- `varresult`: list of ‘lm’ objects.
- `datamat`: The data matrix of the endogenous and explanatory variables.
- `y`: The data matrix of the endogenous variables
- `type`: A character, specifying the deterministic regressors.
- `p`: An integer specifying the lag order.
- `K`: An integer specifying the dimension of the VAR.
- `obs`: An integer specifying the number of used observations.
- `totobs`: An integer specifying the total number of observations.
- `restrictions`: Either `NULL` or a matrix object containing the zero restrictions of the VAR(p).
- `call`: The call to VAR().

Author(s)

Bernhard Pfaff

References


See Also

- `summary`, `plot`, `coef`, `residuals`, `fitted`, `predict`, `irf`, `fevd`, `Phi`, `Psi`, `normality.test`, `arch.test`, `serial.test`, `VARselect`, `logLik`

Examples

data(Canada)
VAR(Canada, p = 2, type = "none")
VAR(Canada, p = 2, type = "const")
VAR(Canada, p = 2, type = "trend")
VAR(Canada, p = 2, type = "both")
Description

These functions are provided for compatibility with older versions of package vars only, and may be defunct as soon as the next release.

Details

'A' is a deprecated synonym for 'Acoef'.
'arch' is a deprecated synonym for 'arch.test'.
'B' is a deprecated synonym for 'Bcoef'.
'normality' is a deprecated synonym for 'normality.test'.
'serial' is a deprecated synonym for 'serial.test'.

See Also

Acoef, arch.test, Bcoef, normality.test, serial.test

VARselect

Information criteria and FPE for different VAR(p)

Description

The function returns information criteria and final prediction error for sequential increasing the lag order up to a VAR(p)-process, which are based on the same sample size.

Usage

VARselect(y, lag.max = 10, type = c("const", "trend", "both", "none"),
season = NULL, exogen = NULL)

Arguments

y
Data item containing the endogenous variables

lag.max
Integer for the highest lag order (default is lag.max = 10).

type
Type of deterministic regressors to include.

season
Inclusion of centered seasonal dummy variables (integer value of frequency).

exogen
Inclusion of exogenous variables.
Details

Estimates a VAR by OLS per equation. The model is of the following form:

\[ y_t = A_1 y_{t-1} + \ldots + A_p y_{t-p} + CD_t + u_t \]

where \( y_t \) is a \( K \times 1 \) vector of endogenous variables and \( u_t \) assigns a spherical disturbance term of the same dimension. The coefficient matrices \( A_1, \ldots, A_p \) are of dimension \( K \times K \). In addition, either a constant and/or a trend can be included as deterministic regressors as well as centered seasonal dummy variables and/or exogenous variables (term \( CD_T \), by setting the type argument to the corresponding value and/or setting season to the desired frequency (integer) and/or providing a matrix object for exogen, respectively. The default for type is const and for season and exogen the default is set to NULL.

Based on the same sample size the following information criteria and the final prediction error are computed:

\[
AIC(n) = \ln \det(\hat{\Sigma}_u(n)) + \frac{2}{T} nK^2 \\
HQ(n) = \ln \det(\hat{\Sigma}_u(n)) + \frac{2\ln(\ln(T))}{T} nK^2 \\
SC(n) = \ln \det(\hat{\Sigma}_u(n)) + \frac{\ln(T)}{T} nK^2 \\
FPE(n) = \left(\frac{T + n^*}{T - n^*}\right)^K \det(\hat{\Sigma}_u(n))
\]

with \( \hat{\Sigma}_u(n) = T^{-1} \sum_{t=1}^{T} \hat{u}_t \hat{u}_t' \) and \( n^* \) is the total number of the parameters in each equation and \( n \) assigns the lag order.

Value

A list with the following elements:

- **selection**: Vector with the optimal lag number according to each criterium.
- **criteria**: A matrix containing the values of the criteria up to lag.max.

Author(s)

Bernhard Pfaff

References


See Also

VAR

Examples

data(Canada)
VARselect(Canada, lag.max = 5, type="const")

--

vec2var

*Transform a VECM to VAR in levels*

Description

An object of formal class ‘ca.jo’ is transformed to a VAR in level presentation.

Usage

vec2var(z, r = 1)

Arguments

z An object of class ‘ca.jo’ generated by function `ca.jo()` in package ‘urca’.

r The cointegration rank (default is r=1).

Details

This function enables the user to transform a vector-error-correction model (VECM) into a level-VAR form. The rank of the matrix \( \Pi \) has to be submitted, i.e. how many cointegration relationships have been determined according to the outcome of `ca.jo()`.
Value

A list with class attribute ‘vec2var’ holding the following elements:

deterministic  The matrix of deterministic coefficients.
A  A list with matrix object(s) containing the coefficients for the lagged endogenous
variables.
p  The lag-order of the estimated VAR-process.
K  The count of endogenous variables.
y  A dataframe with the endogenous variables in levels.
obs  An integer signifying the count of used observations.
totobs  An integer signifying the total number of observations, *i.e* including observa-
tions taken as starting values..
call  The call to vec2var.
vecm  The supplied object z of formal class ca.jo.
datamat  A dataframe with the used dataset.
resid  A matrix with the residuals from the empirical VAR(p).
r  Intefer, the assigned co-integration rank from the call.

Author(s)

Bernhard Pfaff

References


See Also

ca.jo, predict, irf, fevd, Phi, Psi, normality.test, arch.test, serial.test, logLik, plot

Examples

```r
library(urca)
data(finland)
sjf <- finland
sjf.vecm <- ca.jo(sjf, ecdet = "none", type = "eigen", K = 2,
spec = "longrun", season = 4)
vec2var(sjf.vecm, r = 2)
```
Index

* A-model
  SVAR, 38
* AB-model
  SVAR, 38
* ARCH
  arch.test, 3
* Akaike
  VARselect, 46
* Autoregressive Conditional Heteroskedasticity
  arch.test, 3
* B-model
  SVAR, 38
  SVEC, 41
* Blanchard-Quah
  BQ, 6
* Breusch Godfrey
  serial.test, 32
* Causality
  causality, 9
* Empirical Fluctuation Process
  stability, 34
* Fanchart
  fanchart, 12
* Final Prediction Error
  VARselect, 46
* Forecasts of VAR
  predict, 25
* Granger Causality
  causality, 9
* Hannan-Quinn
  VARselect, 46
* Heteroskedasticity
  arch.test, 3
* Impulse Response Function
  Phi, 20
  Psi, 27
* Impulse Responses
  Phi, 20
  Psi, 27
* Information criteria
  VARselect, 46
* Instantaneous Causality
  causality, 9
* Jarque-Bera
  normality.test, 19
* Kurtosis
  normality.test, 19
* Moving Average Representation
  Phi, 20
  Psi, 27
* Normality
  normality.test, 19
* Orthogonalised Impulse Responses
  Psi, 27
* Portmanteau
  serial.test, 32
* Prediction of VAR
  predict, 25
* SVAR
  BQ, 6
  irf, 16
  logLik, 18
  SVAR, 38
* SVCECM
  logLik, 18
  SVEC, 41
* SVECM
  SVEC, 41
* Schwarz
  VARselect, 46
* Serial Correlation
  serial.test, 32
* Serially correlated errors
  serial.test, 32
* Skewness
  normality.test, 19
* Structural Stability
stability, 34

* Structural VAR
  BQ, 6
  SVAR, 38

* Structural VECM
  SVEC, 41

* Structural Vector Autoregressive
  BQ, 6
  SVAR, 38

* Structural Vector Error Correction Model
  SVEC, 41

* Structural vector autoregressive model
  irf, 16

* VAR
  arch.test, 3
  causality, 9
  fevd, 13
  irf, 16
  logLik, 18
  normality.test, 19
  Phi, 20
  predict, 25
  Psi, 27
  restrict, 29
  serial.test, 32
  stability, 34
  VAR, 44
  VARselect, 46
  vec2var, 48

* VECM
  arch.test, 3
  fevd, 13
  irf, 16
  logLik, 18
  normality.test, 19
  Phi, 20
  predict, 25
  Psi, 27
  restrict, 29
  serial.test, 32
  vec2var, 48

* Vector autoregressive model
  arch.test, 3
  causality, 9
  fevd, 13
  irf, 16
  normality.test, 19
  restrict, 29
  serial.test, 32

stability, 34
VAR, 44
VARselect, 46
vec2var, 48

* Vector autoregressive
  logLik, 18
  Phi, 20
  predict, 25
  Psi, 27

* datasets
  Canada, 8

* efp
  stability, 34

* fevd
  fevd, 13

* forecast error variance decomposition
  fevd, 13

* impulse response function
  irf, 16

* irf
  irf, 16

* logLik
  logLik, 18

* regression
  Acoef, 2
  arch.test, 3
  Bcoef, 5
  BQ, 6
  causality, 9
  coef, 11
  fanchart, 12
  fevd, 13
  fitted, 15
  irf, 16
  logLik, 18
  normality.test, 19
  Phi, 20
  plot, 22
  predict, 25
  Psi, 27
  residuals, 28
  restrict, 29
  roots, 31
  serial.test, 32
  stability, 34
  summary, 35
  SVAR, 35
  SVEC, 41
VAR, 44
vars-deprecated, 46
VARselect, 46
vec2var, 48
* restricted VAR
  restrict, 29

A (vars-deprecated), 46
A-deprecated (Acoef), 2
Acoef, 2, 5, 46
arch (vars-deprecated), 46
arch-deprecated (arch.test), 3
arch.test, 3, 25, 45, 46, 49
B (vars-deprecated), 46
B-deprecated (Bcoef), 5
Bcoef, 3, 5, 46
BQ, 6

ca.jo, 49
Canada, 8
causality, 9
coeef, 11, 45
coeeficients (coef), 11
efp, 34, 35

fanchart, 12, 25, 27
fevd, 13, 25, 40, 43, 45, 49
fitted, 15, 45

ginv, 10

irf, 16, 25, 40, 43, 45, 49

logLik, 18, 40, 45, 49

normality (vars-deprecated), 46
normality-deprecated (normality.test), 19
normality.test, 19, 25, 45, 46, 49

par, 13
Phi, 14, 17, 20, 28, 45, 49
plot, 5, 13, 14, 17, 20, 22, 27, 34, 35, 45, 49
predict, 13, 25, 25, 45, 49
print.svarest (SVAR), 38
print.svarsum (summary), 35
print.sveccest (SVEC), 41
print.svecsum (summary), 35

print.varcheck (arch.test), 3
print.varest (VAR), 44
print.varfevd (fevd), 13
print.varirf (irf), 16
print.varpred (predict), 25
print.varstabil (stability), 34
print.varsum (summary), 35
print.vec2var (vec2var), 48
Psi, 14, 17, 22, 27, 45, 49

resid, 5
resid (residuals), 28
residuals, 28, 45
restrict, 29
roots, 31

serial (vars-deprecated), 46
serial-deprecated (serial.test), 32
serial.test, 25, 32, 45, 46, 49
stability, 25, 34
summary, 35, 45
SVAR, 7, 14, 17, 19, 22, 28, 37, 38, 43
SVEC, 14, 17, 19, 22, 37, 40, 41

VAR, 3, 5, 7, 11, 13–15, 17, 19, 20, 22, 25, 27–31, 34, 35, 37, 40, 44, 48
vars-deprecated, 46
VARselect, 45, 46
vcovHC, 10
vec2var, 5, 14, 15, 17, 19, 20, 22, 25, 27–29, 34, 48