Package ‘svars’

September 10, 2019

**Type** Package

**Title** Data-Driven Identification of SVAR Models

**Version** 1.3.2

**Date** 2019-09-10

**Author** Alexander Lange [aut, cre], Bernhard Dalheimer [aut], Helmut Herwartz [aut], Simone Maxand [aut], Hannes Riebl [ctb]

**Maintainer** Alexander Lange <alexander.lange@uni-goettingen.de>


**Depends** R (>= 2.10), vars (>= 1.5.3)

**Imports** expm, reshape2, ggplot2, copula, clue, pbapply, steadyICA, tsDyn, DEoptim, zoo, strucchange, Rcpp

**LinkingTo** Rcpp, RcppArmadillo

**NeedsCompilation** yes

**License** MIT + file LICENSE

**LazyData** TRUE

**RoxygenNote** 6.1.1

**Suggests** testthat (>= 2.1.0)
Description

Bootstrap intervals based on bias-adjusted estimators

Usage

ba.boot(x, nc = 1)

Arguments

x  SVAR object of class "sboot"
nc  Integer. Number of processor cores
Value

A list of class "sboot" with elements

- `true` Point estimate of impulse response functions
- `bootstrap` List of length "nboot" holding bootstrap impulse response functions
- `SE` Bootstrapped standard errors of estimated covariance decomposition (only if "x" has method "Cramer von-Mises", or "Distance covariances")
- `nboot` Number of bootstrap iterations
- `b_length` Length of each block
- `point_estimate` Point estimate of covariance decomposition
- `boot_mean` Mean of bootstrapped covariance decompositions
- `signrest` Evaluated sign pattern
- `sign_complete` Frequency of appearance of the complete sign pattern in all bootstrapped covariance decompositions
- `sign_part` Frequency of bootstrapped covariance decompositions which conform the complete predetermined sign pattern. If `signrest=NULL`, the frequency of bootstrapped covariance decompositions that hold the same sign pattern as the point estimate is provided.
- `sign_part` Frequency of single shocks in all bootstrapped covariance decompositions which accord to a specific predetermined sign pattern
- `cov_bs` Covariance matrix of bootstrapped parameter in impact relations matrix
- `method` Used bootstrap method

References


See Also

- `mb.boot`, `wild.boot`

Examples

```r
# data contains quarterly observations from 1965Q1 to 2008Q3
# x = output gap
# pi = inflation
# i = interest rates
set.seed(23211)
v1 <- vars::VAR(USA, lag.max = 10, ic = "AIC")
x1 <- id.dc(v1)
summary(x1)

# Bootstrap
bb <- mb.boot(x1, b.length = 15, nboot = 300, n.ahead = 30, nc = 1, signrest = NULL)
```
summary(bb)
plot(bb, lowerq = 0.16, upperq = 0.84)

# Bias-adjusted bootstrap
bb2 <- ba.boot(bb, nc = 1)
plot(bb2, lowerq = 0.16, upperq = 0.84)

chow.test

Chow Test for Structural Break

Description
The Chow test for structural change is implemented as sample-split and break-point test (see Luetkepohl and Kraetzig, 2004, p. 135). An estimated VAR model and the presupposed structural break need to be provided.

Usage
chow.test(x, SB, nboot = 500, start = NULL, end = NULL,
          frequency = NULL, format = NULL, dateVector = NULL)

Arguments
x
An object of class 'vars', 'vec2var', 'nlVar'. Estimated VAR object. Or an object of class 'chowpretest' from stability()

SB
Integer, vector or date character. The structural break is specified either by an integer (number of observations in the pre-break period), a vector of ts() frequencies if a ts object is used in the VAR or a date character. If a date character is provided, either a date vector containing the whole time line in the corresponding format or common time parameters need to be provided

nboot
Integer. Number of bootstrap iterations to calculate quantiles and p-values

start
Character. Start of the time series (only if dateVector is empty)

end
Character. End of the time series (only if dateVector is empty)

frequency
Character. Frequency of the time series (only if dateVector is empty)

format
Character. Date format (only if dateVector is empty)

dateVector
Vector. Vector of time periods containing SB in corresponding format

Value
A list of class "chow" with elements

lambda_bp Test statistic of the Chow test with break point
testcrit_bp Critical value of the test statistic lambda_bp
p.value_bp p-value of the test statistic lambda_bp
chow.test

lambda_sp  Test statistic of the Chow test with sample split
testcrit_sp  Critical value of the test statistic lambda_sp
p.value_sp  p-value of the test statistic lambda_sp
SB  Structural break tested
SBcharacter  Structural break tested as character
p  Number of lags used

References


See Also

stability

Examples

# Testing for structural break in USA data
#' # data contains quartlery observations from 1965Q1 to 2008Q2
#' # assumed structural break in 1979Q3
#' # x = output gap
#' # pi = inflation
#' # i = interest rates
set.seed(23211)
v1 <- vars::VAR(USA, lag.max = 10, ic = "AIC")
z1 <- chow.test(v1, SB = 59)
summary(z1)

#Using stability() to find potential break point and sample split
x1 <- stability(v1, type = "mv-chow-test")
plot(x1)
z1.1 <- chow.test(x1)
summary(z1.1)

#Or using sample split as benchmark
x1$break_point <- FALSE
z1.1 <- chow.test(x1)
summary(z1.1)

#Structural brake via Dates
#given that time series vector with dates is available
dateVector <- seq(as.Date("1965/1/1"), as.Date("2008/7/1"), "quarter")
z2 <- chow.test(v1, SB = "1979-07-01", format = "%Y-%m-%d", dateVector = dateVector)
summary(z2)

# alternatively pass sequence arguments directly
z3 <- chow.test(v1, SB = "1979-07-01", format = "%Y-%m-%d",
  start = "1965-01-01", end = "2008-07-01",
  dateVector = dateVector)
fevd

Forecast error variance decomposition for SVAR Models

Description
Calculation of forecast error variance decomposition for an identified SVAR object 'svars' derived by function id.st(), id.cvm(), id.cv(), id.dc() or id.ngml().

Usage
```r
## S3 method for class 'svars'
fevd(x, n.ahead = 10, ...)
```

Arguments
- `x` : SVAR object of class "svars".
- `n.ahead` : Integer specifying the steps.
- `...` : Currently not used.

Value
A list with class attribute "svarfevd" holding the forecast error variance decompositions as data frames.

References

See Also
- `id.cvm`, `id.garch`, `id.dc`, `id.ngml`, `id.cv` or `id.st`

Examples
```r
v1 <- vars::VAR(USA, lag.max = 10, ic = "AIC")
x1 <- id.dc(v1)
x2 <- fevd(x1, n.ahead = 30)
plot(x2)
```
Historical decomposition for SVAR Models

Description

Calculation of historical decomposition for an identified SVAR object 'svars' derived by function id.st(), id.cvm(), id.cv(), id.dc() or id.ngml().

Usage

hd(x, series = 1, transition = 0)

Arguments

x: SVAR object of class "svars"
series: Integer indicating the series that should be decomposed.
transition: Numeric. Value from [0, 1] indicating how many initial values should be discarded, i.e., 0.1 means that the first 10 per cent observations of the sample are considered as transient.

Value

A list with class attribute "hd" holding the historical decomposition as data frame.

References


See Also

id.cvm, id.dc, id.ngml, id.cv, id.garch or id.st

Examples

v1 <- vars::VAR(USA, lag.max = 10, ic = "AIC")
x1 <- id.dc(v1)
x2 <- hd(x1, series = 2)
plot(x2)
id.cv

Changes in volatility identification of SVAR models

Description

Given an estimated VAR model, this function applies changes in volatility to identify the structural impact matrix $B$ of the corresponding SVAR model

$$y_t = c_t + A_1 y_{t-1} + \ldots + A_p y_{t-p} + u_t = c_t + A_1 y_{t-1} + \ldots + A_p y_{t-p} + B \epsilon_t.$$ 

Matrix $B$ corresponds to the decomposition of the pre-break covariance matrix $\Sigma_1 = BB'$. The post-break covariance corresponds to $\Sigma_2 = BAB'$ where $\Lambda$ is the estimated unconditional heteroskedasticity matrix.

Usage

```
id.cv(x, SB, start = NULL, end = NULL, frequency = NULL, format = NULL, dateVector = NULL, max.iter = 50, crit = 0.001, restriction_matrix = NULL)
```

Arguments

- **x**: An object of class 'vars', 'vec2var', 'nlVar'. Estimated VAR object
- **SB**: Integer, vector or date character. The structural break is specified either by an integer (number of observations in the pre-break period), a vector of ts() frequencies if a ts object is used in the VAR or a date character. If a date character is provided, either a date vector containing the whole time line in the corresponding format (see examples) or common time parameters need to be provided
- **start**: Character. Start of the time series (only if dateVector is empty)
- **end**: Character. End of the time series (only if dateVector is empty)
- **frequency**: Character. Frequency of the time series (only if dateVector is empty)
- **format**: Character. Date format (only if dateVector is empty)
- **dateVector**: Vector. Vector of time periods containing SB in corresponding format
- **max.iter**: Integer. Number of maximum GLS iterations
- **crit**: Numeric. Critical value for the precision of the GLS estimation
- **restriction_matrix**: Matrix. A matrix containing presupposed entries for matrix $B$, NA if no restriction is imposed (entries to be estimated). Alternatively, a $K^2 \times K^2$ matrix can be passed, where ones on the diagonal designate unrestricted and zeros restricted coefficients. (as suggested in Luetkepohl, 2017, section 5.2.1).
Value

A list of class "svars" with elements

- **Lambda**: Estimated unconditional heteroscedasticity matrix \( \Lambda \)
- **Lambda_SE**: Matrix of standard errors of Lambda
- **B**: Estimated structural impact matrix \( B \), i.e. unique decomposition of the covariance matrix of reduced form residuals
- **B_SE**: Standard errors of matrix \( B \)
- **n**: Number of observations
- **Fish**: Observed Fisher information matrix
- **Lik**: Function value of likelihood
- **wald_statistic**: Results of pairwise Wald tests
- **iteration**: Number of GLS estimations
- **method**: Method applied for identification
- **SB**: Structural break (number of observations)
- **A_hat**: Estimated VAR parameter via GLS
- **type**: Type of the VAR model, e.g. 'const'
- **SBcharacter**: Structural break (date; if provided in function arguments)
- **restrictions**: Number of specified restrictions
- **restriction_matrix**: Specified restriction matrix
- **y**: Data matrix
- **p**: Number of lags
- **K**: Dimension of the VAR

References


See Also

For alternative identification approaches see `id.st, id.garch, id.cvm, id.dc` or `id.ngml`

Examples

```r
# data contains quarterly observations from 1965Q1 to 2008Q2
# assumed structural break in 1979Q3
# x = output gap
# pi = inflation
# i = interest rates
set.seed(23211)
```
id.cvm

Independence-based identification of SVAR models based on Cramervon Mises distance

Description

Given an estimated VAR model, this function applies independence-based identification for the structural impact matrix B of the corresponding SVAR model

\[ y_t = c_t + A_1 y_{t-1} + \ldots + A_p y_{t-p} + u_t = c_t + A_{11} y_{t-1} + \ldots + A_{1p} y_{t-p} + B \epsilon_t. \]
Matrix $B$ corresponds to the unique decomposition of the least squares covariance matrix $\Sigma_u = BB'$ if the vector of structural shocks $\epsilon_t$ contains at most one Gaussian shock (Comon, 1994). A nonparametric dependence measure, the Cramer-von Mises distance (Genest and Remillard, 2004), determines least dependent structural shocks. The minimum is obtained by a two step optimization algorithm similar to the technique described in Herwartz and Ploedt (2016).

**Usage**

```r
id.cvm(x, dd = NULL, itermax = 500, steptol = 100, iter2 = 75)
```

**Arguments**

- **x**: An object of class 'vars', 'vec2var', 'nlVar'. Estimated VAR object.
- **dd**: Object of class 'indepTestDist' (generated by 'indepTest' from package 'copula'). A simulated independent sample of the same size as the data. If not supplied, it will be calculated by the function.
- **itermax**: Integer. Maximum number of iterations for DEoptim.
- **steptol**: Numeric. Tolerance for steps without improvement for DEoptim.
- **iter2**: Integer. Number of iterations for the second optimization.

**Value**

A list of class "svars" with elements:

- **B**: Estimated structural impact matrix $B$, i.e. unique decomposition of the covariance matrix of reduced form errors.
- **A_hat**: Estimated VAR parameter.
- **method**: Method applied for identification.
- **n**: Number of observations.
- **type**: Type of the VAR model, e.g. 'const'.
- **y**: Data matrix.
- **p**: Number of lags.
- **K**: Dimension of the VAR.
- **rotation_angles**: Rotation angles, which lead to maximum independence.
- **inc**: Indicator. 1 = second optimization increased the estimation precision. 0 = second optimization did not increase the estimation precision.
- **test.stats**: Computed test statistics of independence test.
- **iter1**: Number of iterations of first optimization.
- **test1**: Minimum test statistic from first optimization.
- **test2**: Minimum test statistic from second optimization.
References


See Also

For alternative identification approaches see id.st, id.garch, id.cv, id.dc or id.ngml

Examples

```r
# data contains quarterly observations from 1965Q1 to 2008Q3
# x = output gap
# pi = inflation
# i = interest rates
set.seed(23211)
v1 <- vars::VAR(USA, lag.max = 10, ic = "AIC")
cob <- copula::indepTestSim(v1$obs, v1$K, verbose=FALSE)
x1 <- id.cvm(v1, dd = cob)
summary(x1)

# switching columns according to sign pattern
x1$B <- x1$B[,c(3,2,1)]
x1$B[,3] <- x1$B[,3]*(-1)

# impulse response analysis
i1 <- irf(x1, n.ahead = 30)
plot(i1, scales = 'free_y')
```

id.dc

Independence-based identification of SVAR models based on distance covariances

Description

Given an estimated VAR model, this function applies independence-based identification for the structural impact matrix $B$ of the corresponding SVAR model

$$y_t = c_t + A_1 y_{t-1} + ... + A_p y_{t-p} + u_t = c_t + A_1 y_{t-1} + ... + A_p y_{t-p} + B \epsilon_t.$$
Matrix B corresponds to the unique decomposition of the least squares covariance matrix \( \Sigma_u = B B' \) if the vector of structural shocks \( \epsilon_t \) contains at most one Gaussian shock (Comon, 1994). A nonparametric dependence measure, the distance covariance (Szekely et al., 2007), determines least dependent structural shocks. The algorithm described in Matteson and Tsay (2013) is applied to calculate the matrix B.

**Usage**

```r
id.dc(x, PIT = FALSE)
```

**Arguments**

- `x`: An object of class 'vars', 'vec2var', 'nlVar'. Estimated VAR object
- `PIT`: Logical. If PIT='TRUE', the distribution and density of the independent components are estimated using gaussian kernel density estimates

**Value**

A list of class "svars" with elements

- `B`: Estimated structural impact matrix B, i.e. unique decomposition of the covariance matrix of reduced form errors
- `A_hat`: Estimated VAR parameter
- `method`: Method applied for identification
- `n`: Number of observations
- `type`: Type of the VAR model, e.g. 'const'
- `y`: Data matrix
- `p`: Number of lags
- `K`: Dimension of the VAR
- `PIT`: Logical, if PIT is used

**References**

Matteson, D. S. & Tsay, R. S., 2013. Independent Component Analysis via Distance Covariance, pre-print

**See Also**

For alternative identification approaches see `id.st`, `id.garch`, `id.cvm`, `id.cv` or `id.ngml`
Examples

```r
# data contains quarterly observations from 1965Q1 to 2008Q3
# x = output gap
# pi = inflation
# i = interest rates
set.seed(23211)
v1 <- vars::VAR(USA, lag.max = 10, ic = "AIC")
x1 <- id.dc(v1)
summary(x1)

# switching columns according to sign pattern
x1$B <- x1$B[,c(3,2,1)]
x1$B[,3] <- x1$B[,3]*(-1)

# impulse response analysis
i1 <- irf(x1, n.ahead = 30)
plot(i1, scales = 'free_y')
```

**id.garch**

**GARCH identification of SVAR models**

**Description**

Given an estimated VAR model, this function uses GARCH-type variances to identify the structural impact matrix \( B \) of the corresponding SVAR model

\[
y_t = c_t + A_1 y_{t-1} + \ldots + A_p y_{t-p} + u_t = c_t + A_1 y_{t-1} + \ldots + A_p y_{t-p} + B \epsilon_t.
\]

Matrix \( B \) corresponds to the decomposition of the least squares covariance matrix \( \Sigma_u = B \Lambda_t B' \), where \( \Lambda_t \) is the estimated conditional heteroskedasticity matrix.

**Usage**

`id.garch(x, max.iter = 5, crit = 0.001, restriction_matrix = NULL)`

**Arguments**

- **x**: An object of class 'vars', 'vec2var', 'nlVar'. Estimated VAR object
- **max.iter**: Integer. Number of maximum likelihood optimizations
- **crit**: Numeric. Critical value for the precision of the iterative procedure
- **restriction_matrix**: Matrix. A matrix containing presupposed entries for matrix \( B \), NA if no restriction is imposed (entries to be estimated). Alternatively, a \( K^2 \times K^2 \) matrix can be passed, where ones on the diagonal designate unrestricted and zeros restricted coefficients. (as suggested in Luetkepohl, 2017, section 5.2.1).
Value

A list of class "svars" with elements

- **B**: Estimated structural impact matrix B, i.e. unique decomposition of the covariance matrix of reduced form residuals
- **B_SE**: Standard errors of matrix B
- **GARCH_parameter**: Estimated GARCH parameters of univariate GARCH models
- **GARCH_SE**: Standard errors of GARCH parameters
- **n**: Number of observations
- **Fish**: Observed Fisher information matrix
- **Lik**: Function value of likelihood
- **iteration**: Number of likelihood optimizations
- **method**: Method applied for identification
- **A_hat**: Estimated VAR parameter via GLS
- **type**: Type of the VAR model, e.g. 'const'
- **restrictions**: Number of specified restrictions
- **restriction_matrix**: Specified restriction matrix
- **y**: Data matrix
- **p**: Number of lags
- **K**: Dimension of the VAR

References


See Also

For alternative identification approaches see `id.st`, `id.cvm`, `id.cv`, `id.dc` or `id.ngml`

Examples

```r
# data contains quarterly observations from 1965Q1 to 2008Q2
# assumed structural break in 1979Q3
# x = output gap
# pi = inflation
# i = interest rates
set.seed(23211)
v1 <- vars::VAR(USA, lag.max = 10, ic = "AIC")
x1 <- id.garch(v1)
```
summary(x1)

# Impulse response analysis
i1 <- irf(x1, n.ahead = 30)
plot(i1, scales = 'free_y')

# Restrictions
# Assuming that the interest rate does’t influence the output gap on impact
restMat <- matrix(rep(NA, 9), ncol = 3)
restMat[1, 3] <- 0
x2 <- id.garch(v1, restriction_matrix = restMat)
summary(x2)

id.ngml
Non-Gaussian maximum likelihood identification of SVAR models

Description
Given an estimated VAR model, this function applies identification by means of a non-Gaussian
likelihood for the structural impact matrix B of the corresponding SVAR model

\[ y_t = c_t + A_1 y_{t-1} + \ldots + A_p y_{t-p} + u_t = c_t + A_1 y_{t-1} + \ldots + A_p y_{t-p} + B \epsilon_t. \]

Matrix B corresponds to the unique decomposition of the least squares covariance matrix \( \Sigma_u = BB' \) if the vector of structural shocks \( \epsilon_t \) contains at most one Gaussian shock (Comon, 94). A
likelihood function of independent t-distributed structural shocks \( \epsilon_t = B^{-1} u_t \) is maximized with
respect to the entries of B and the degrees of freedom of the t-distribution (Lanne et al., 2017).

Usage
id.ngml(x, stage3 = FALSE, restriction_matrix = NULL)

Arguments
x
An object of class ‘vars’, ‘vec2var’, ‘nlVar’. Estimated VAR object

stage3
Logical. If stage3="TRUE", the VAR parameters are estimated via non-gaussian
maximum likelihood (computationally demanding)

restriction_matrix
Matrix. A matrix containing presupposed entries for matrix B, NA if no restric-
tion is imposed (entries to be estimated). Alternatively, a \( K^2 \times K^2 \) matrix can
be passed, where ones on the diagonal designate unrestricted and zeros restricted
coefficients. (as suggested in Luetkepohl, 2017, section 5.2.1).
Value

A list of class "svars" with elements

- **B**: Estimated structural impact matrix B, i.e. unique decomposition of the covariance matrix of reduced form errors
- **sigma**: Estimated scale of the standardized matrix B_stand, i.e. $B = B_{stand} \cdot diag(\sigma_1, ..., \sigma_K)$
- **sigma_SE**: Standard errors of the scale
- **df**: Estimated degrees of freedom
- **df_SE**: Standard errors of the degrees of freedom
- **Fish**: Observed Fisher information matrix
- **A_hat**: Estimated VAR parameter via ML
- **B_stand**: Estimated standardized structural impact matrix
- **B_stand_SE**: Standard errors of standardized matrix B_stand
- **Lik**: Function value of likelihood
- **method**: Method applied for identification
- **n**: Number of observations
- **type**: Type of the VAR model, e.g. 'const'
- **y**: Data matrix
- **p**: Number of lags
- **K**: Dimension of the VAR
- **restrictions**: Number of specified restrictions
- **restriction_matrix**: Specified restriction matrix
- **stage3**: Logical, whether Stage 3 is performed

References


See Also

For alternative identification approaches see `id.st, id.garch, id.cvm, id.dc` or `id.cv`

Examples

```r
# data contains quarterly observations from 1965Q1 to 2008Q3
# x = output gap
# pi = inflation
# i = interest rates
set.seed(23211)
v1 <- vars::VAR(USA, lag.max = 10, ic = "AIC")
```
x1 <- id.ngml(v1)
summary(x1)

# switching columns according to sign pattern
x1$B <- x1$B[,c(3,2,1)]
x1$B[,3] <- x1$B[,3]*(-1)

# impulse response analysis
i1 <- irf(x1, n.ahead = 30)
plot(i1, scales = 'free_y')

---

**id.st**

**Identification of SVAR models by means of a smooth transition of volatility**

**Description**

Given an estimated VAR model, this function uses a smooth transition in the covariance to identify the structural impact matrix $B$ of the corresponding SVAR model

$$y_t = c_t + A_1 y_{t-1} + ... + A_p y_{t-p} + u_t = c_t + A_1 y_{t-1} + ... + A_p y_{t-p} + B \epsilon_t.$$  

Matrix $B$ corresponds to the decomposition of the pre-break covariance matrix $\Sigma_1 = BB'$. The post-break covariance corresponds to $\Sigma_2 = B \Lambda B'$ where $\Lambda$ is the estimated heteroskedasticity matrix.

**Usage**

```r
id.st(x, c_lower = 0.3, c_upper = 0.7, c_step = 5, c_fix = NULL, 
transition_variable = NULL, gamma_lower = -3, gamma_upper = 2, 
gamma_step = 0.5, gamma_fix = NULL, nc = 1, max.iter = 5, 
crit = 0.001, restriction_matrix = NULL, lr_test = FALSE)
```

**Arguments**

- **x**: An object of class 'vars', 'vec2var', 'nlVar'. Estimated VAR object
- **c_lower**: Numeric. Starting point for the algorithm to start searching for the volatility shift. Default is 0.3*(Total number of observations)
- **c_upper**: Numeric. Ending point for the algorithm to stop searching for the volatility shift. Default is 0.7*(Total number of observations). Note that in case of a stochastic transition variable, the input requires an absolute value
- **c_step**: Integer. Step width of c. Default is 5. Note that in case of a stochastic transition variable, the input requires an absolute value
- **c_fix**: Numeric. If the transition point is known, it can be passed as an argument where transition point = Number of observations - c_fix
transition_variable
A numeric vector that represents the transition variable. By default (NULL),
the time is used as transition variable. Note that c_lower, c_upper, c_step and/or
c_fix have to be adjusted to the specified transition variable

gamma_lower
Numeric. Lower bound for gamma. Small values indicate a flat transition func-
tion. Default is -3

gamma_upper
Numeric. Upper bound for gamma. Large values indicate a steep transition
function. Default is 2

gamma_step
Numeric. Step width of gamma. Default is 0.5

gamma_fix
Numeric. A fixed value for gamma, alternative to gamma found by the function

nc
Integer. Number of processor cores Note that the smooth transition model is
computationally extremely demanding.

max.iter
Integer. Number of maximum GLS iterations

crit
Numeric. Critical value for the precision of the GLS estimation

restriction_matrix
Matrix. A matrix containing presupposed entries for matrix B, NA if no restric-
tion is imposed (entries to be estimated). Alternatively, a K^2*K^2 matrix can
be passed, where ones on the diagonal designate unrestricted and zeros restricted
coefficients. (as suggested in Luetkepohl, 2017, section 5.2.1).

lr_test
Logical. Indicates whether the restricted model should be tested against the
unrestricted model via a likelihood ratio test

Value
A list of class "svars" with elements

Lambda
Estimated heteroscedasticity matrix \( \Lambda \)

Lambda_SE
Matrix of standard errors of Lambda

B
Estimated structural impact matrix B, i.e. unique decomposition of the covari-
ance matrix of reduced form residuals

B_SE
Standard errors of matrix B

n
Number of observations

Fish
Observed Fisher information matrix

Lik
Function value of likelihood

wald_statistic
Results of pairwise Wald tests

iteration
Number of GLS estimations

method
Method applied for identification

est_c
Structural break (number of observations)

est_g
Transition coefficient

transition_variable
Vector of transition variable

comb
Number of all grid combinations of gamma and c
transition_function

  Vector of transition function

A_hat

  Estimated VAR parameter via GLS

type

  Type of the VAR model e.g., 'const'

y

  Data matrix

p

  Number of lags

K

  Dimension of the VAR

restrictions

  Number of specified restrictions

restriction_matrix

  Specified restriction matrix

lr_test

  Logical, whether a likelihood ratio test is performed

lRatioTest

  Results of likelihood ratio test

References


See Also

For alternative identification approaches see id.cv, id.garch, id.cvm, id.dc, or id.ngml

Examples

# data contains quartlery observations from 1965Q1 to 2008Q2
# x = output gap
# pi = inflation
# i = interest rates
set.seed(23211)
v1 <- vars::VAR(USA, lag.max = 10, ic = "AIC")
x1 <- id.st(v1, c_fix = 80, gamma_fix = 0)
summary(x1)
plot(x1)

# switching columns according to sign patter
x1$B <- x1$B[,c(3,2,1)]
x1$B[,3] <- x1$B[,3]*(-1)

# Impulse response analysis
i1 <- irf(x1, n.ahead = 30)
plot(i1, scales = 'free_y')

# Example with same data set as in Luetkepohl and Nestunajev 2017
v1 <- vars::VAR(LN, p = 3, type = 'const')
x1 <- id.st(v1, c_fix = 167, gamma_fix = -2.77)
summary(x1)
plot(x1)
Impulse Response Functions for SVAR Models

Description
Calculation of impulse response functions for an identified SVAR object 'svars' derived by function id.cvm(), id.cv(), id.dc(), id.ngml() or id.st().

Usage
## S3 method for class 'svars'
irf(x, ..., n.ahead = 20)

Arguments
x
SVAR object of class "svars".

... Currently not used.

n.ahead Integer specifying the steps.

Value
A list with class attribute "svarirf" holding the impulse response functions as data frame.

References

See Also
id.cvm, id.dc, id.ngml, id.cv or id.st

Examples
v1 <- vars::VAR(USA, lag.max = 10, ic = "AIC")
x1 <- id.ngml(v1)
x2 <- irf(x1, n.ahead = 20)
plot(x2)
Chi-square test for joint hypotheses

Description

Based on an existing bootstrap object, the test statistic allows to test joint hypotheses for selected entries of the structural matrix B. The test statistic reads as

\[(Rvec(\hat{B}) - r)'R(\hat{Cov}[vec(B^*)])^{-1}R'(Rvec(\hat{b} - r)) \sim \chi^2_J,\]

where \(\hat{Cov}[vec(B^*)]\) is the estimated covariance of vectorized bootstrap estimates of structural parameters. The composite null hypothesis is \(H_0 : Rvec(B) = r\).

Usage

js.test(x, R, r = NULL)

Arguments

x Object of class 'sboot'
R A J*K^2 selection matrix, where J is the number of hypotheses and K the number of time series.
r A J*1 vector of restrictions

Value

A list of class "jstest" with elements

test_statistic Test statistic
p_value P-value
R Selection matrix
r Vector of restrictions

References


See Also

mb.boot, wild.boot
Examples

```r
# data contains quarterly observations from 1965Q1 to 2008Q3
# x = output gap
# pi = inflation
# i = interest rates
v1 <- vars::VAR(USA, lag.max = 10, ic = "AIC")
x1 <- id.de(v1)

# Bootstrapping of SVAR
bb <- wild.boot(x1, nboot = 1000, n.ahead = 30)

# Testing the hypothesis of a lower triangular matrix as
# relation between structural and reduced form errors
R <- rbind(c(0,0,0,1,0,0,0,0,0),
          c(0,0,0,0,0,0,1,0,0),
          c(0,0,0,0,0,0,0,1,0))
c.test <- js.test(bb, R)
summary(c.test)
```

---

Description

A five dimensional time series model which is commonly used to analyze the interaction between monetary policy and the stock market. Monthly observations from 1970M1 to 2007M6:

- **q**: Linearly detrended log of an industrial production index
- **pi**: Annual change in the log of consumer prices (CPI index) (x100)
- **c**: Annual change in the log of the World Bank (non energy) commodity price index (x100)
- **s**: Log of the real S&P500 stock price index deflated by the consumer price index to measure the real stock prices; the series is first differenced to represent monthly returns
- **r**: Interest rate on Federal funds

All series, with exception of the commodity price index (c), are taken from the FRED database and transformed as in Luetkepohl & Netsunajev (2017). The commodity price index comes from the World Bank. A more detailed description of the data and a corresponding VAR model implementation can be found in Luetkepohl & Netsunajev (2017).

Usage
mb.boot

Moving block bootstrap for IRFs of identified SVARs

Description

Calculating confidence bands for impulse response via moving block bootstrap

Usage

mb.boot(x, design = "recursive", b.length = 15, n.ahead = 20, nboot = 500, nc = 1, dd = NULL, signrest = NULL, itermax = 300, steptol = 200, iter2 = 50)

Arguments

x : SVAR object of class "svars"
design : character. If design="fixed", a fixed design bootstrap is performed. If design="recursive", a recursive design bootstrap is performed.
b.length : Integer. Length of each block
n.ahead : Integer specifying the steps
nboot : Integer. Number of bootstrap iterations
nc : Integer. Number of processor cores
dd : Object of class 'indepTestDist'. A simulated independent sample of the same size as the data. If not supplied, it will be calculated by the function
signrest : A list with vectors containing 1 and -1, e.g. c(1,-1,1), indicating a sign pattern of specific shocks to be tested with the help of the bootstrap samples.
itermax : Integer. Maximum number of iterations for DEoptim
steptol : Numeric. Tolerance for steps without improvement for DEoptim
iter2 : Integer. Number of iterations for the second optimization
Value

A list of class "sboot" with elements

- **true**: Point estimate of impulse response functions
- **bootstrap**: List of length "nboot" holding bootstrap impulse response functions
- **SE**: Bootstrapped standard errors of estimated covariance decomposition (only if "x" has method "Cramer von-Mises", or "Distance covariances")
- **nboot**: Number of bootstrap iterations
- **design**: character. Whether a fixed design or recursive design bootstrap is performed
- **b_length**: Length of each block
- **point_estimate**: Point estimate of covariance decomposition
- **boot_mean**: Mean of bootstrapped covariance decompositions
- **signrest**: Evaluated sign pattern
- **sign_complete**: Frequency of appearance of the complete sign pattern in all bootstrapped covariance decompositions
- **sign_part**: Frequency of bootstrapped covariance decompositions which conform the complete predetermined sign pattern. If signrest=NULL, the frequency of bootstrapped covariance decompositions that hold the same sign pattern as the point estimate is provided.
- **sign_part**: Frequency of single shocks in all bootstrapped covariance decompositions which accord to a specific predetermined sign pattern
- **cov_bs**: Covariance matrix of bootstrapped parameter in impact relations matrix
- **method**: Used bootstrap method

References


See Also

id.cvm, id.dc, id.ngml, id.garch, id.cv or id.st

Examples

```r
# data contains quarterly observations from 1965Q1 to 2008Q3
# x = output gap
# pi = inflation
# i = interest rates
set.seed(23211)
v1 <- vars::VAR(USA, lag.max = 10, ic = "AIC")
x1 <- id.dc(v1)
summary(x1)
```
# impulse response analysis with confidence bands
# Checking how often theory based impact relations appear
signrest <- list(demand = c(1,1,1), supply = c(-1,1,1), money = c(-1,-1,1))
bb <- mb.boot(x1, b.length = 15, nboot = 500, n.ahead = 30, nc = 1, signrest = signrest)
summary(bb)
plot(bb, lowerq = 0.16, upperq = 0.84)

---

stability

Structural stability of a VAR(p)

Description

Computes an empirical fluctuation process according to a specified method from the generalized fluctuation test framework. The test utilises the function efp() and its methods from package ‘strucchange’. Additionally, the function provides the option to compute a multivariate chow test.

Usage

```r
## S3 method for class 'varest'
stability(x, type = c("OLS-CUSUM", "Rec-CUSUM", "Rec-MOSUM", "OLS-MOSUM", "RE", "ME", "Score-CUSUM", "Score-MOSUM", "fluctuation", "mv-chow-test"), 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` and `chow.test`.
- `h`: A numeric from interval (0,1) specifying the bandwidth. Determines the size of the data window relative to sample size (for 'MOSUM', 'ME' and 'mv-chow-test' only).
- `dynamic`: Logical. If ‘TRUE’ the lagged observations are included as a regressor (not if 'type' is 'mv-chow-test').
- `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` and `chow.test`. 
**stability**

**Value**

A list with either class attribute ‘varstabil’ or ‘chowpretest’ holding the following elements in case of class ‘varstabil’:

- **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.

In case of class ‘chowpretest’ the list consists of the following elements:

- **teststat_bp**: A vector containing the calculated break point test statistics for all considered break points.
- **teststat_sp**: A vector containing the calculated sample split test statistics for all considered sample splits.
- **from**: An integer specifying the first observation as possible break date.
- **to**: An integer specifying the last observation as possible break date.
- **var**: A list with objects of class ‘varest’.
- **break_point**: Logical, if the break point test should be the benchmark for later analysis.

**Author(s)**

Bernhard Pfaff, Alexander Lange, Bernhard Dalheimer, Simone Maxand, Helmut Herwartz

**References**


and see the references provided in the reference section of `efp` and `chow.test`, too.

**See Also**

`VAR, plot, efp, chow.test`

**Examples**

data(Canada)
var.2c <- VAR(Canada, p = 2, type = "const")
var.2c.stabil <- stability(var.2c, type = "OLS-CUSUM")

plot(var.2c.stabil)


data(USA)
v1 <- VAR(USA, p = 6)
x1 <- stability(v1, type = "mv-chow-test")
plot(x1)
Description

This package implements data-driven identification methods for structural vector autoregressive (SVAR) models. Based on an existing VAR model object, the structural impact matrix $B$ may be obtained via different forms of heteroskedasticity or independent components.

Details

The main functions to retrieve structural impact matrices are:

- `id.cv` Identification via changes in volatility,
- `id.cvm` Independence-based identification of SVAR models based on Cramer-von Mises distance,
- `id.dc` Independence-based identification of SVAR models based on distance covariances,
- `id.garch` Identification through patterns of conditional heteroskedasticity,
- `id.ngml` Identification via Non-Gaussian maximum likelihood,
- `id.st` Identification by means of smooth transition in covariance.

All of these functions require an estimated var object. Currently the classes 'vars' and 'vec2var' from the vars package, 'nlVar', which includes both VAR and VECM, from the tsDyn package as well as the list from MTS package are supported. Besides these core functions, additional tools to calculate confidence bands for impulse response functions using bootstrap techniques as well as the Chow-Test for structural changes are implemented. The USA dataset is used to showcase the functionalities in examples throughout the package.
Description

The time series of output gap (x), inflation (pi) and interest rate (i) are taken from the FRED database and transformed as in Herwartz & Ploedt (2016). The trivariate time series model is commonly used to analyze monetary policy shocks.
Quarterly observations from 1965Q1 to 2008Q3:

- x: Percentage log-deviation of real GDP wrt the estimate of potential output by the Congressional Budget Office
- pi: Annualized quarter-on-quarter growth of the GDP deflator
- i: Interest rate on Federal funds

A more detailed description of the data and a corresponding VAR model implementation can be found in Herwartz & Ploedt (2016).

Usage

USA

Format

A data.frame containing 174 observations on 3 variables.

Source

Data originally from FRED database of the Federal Reserve Bank of St. Louis.

Description

Calculating confidence bands for impulse response functions via wild bootstrap techniques (Goncalves and Kilian, 2004).
Usage

wild.boot(x, design = "fixed", distr = "rademacher", n.ahead = 20,
nboot = 500, nc = 1, dd = NULL, signrest = NULL, itermax = 300,
steptol = 200, iter2 = 50, rademacher = "deprecated")

Arguments

x
SVAR object of class "svars"

design
character. If design="fixed", a fixed design bootstrap is performed. If de-
sign="recursive", a recursive design bootstrap is performed.

distr
character. If distr="rademacher", the Rademacher distribution is used to gen-
erate the bootstrap samples. If distr="mammen", the Mammen distribution is
used. If distr = "gaussian", the gaussian distribution is used.

n.ahead
Integer specifying the steps

nboot
Integer. Number of bootstrap iterations

nc
Integer. Number of processor cores

dd
Object of class 'indepTestDist'. A simulated independent sample of the same
size as the data. roxIf not supplied, it will be calculated by the function

signrest
A list with vectors containing 1 and -1, e.g. c(1,-1,1), indicating a sign pattern
of specific shocks to be tested with the help of the bootstrap samples.

itermax
Integer. Maximum number of iterations for DEoptim

steptol
Integer. Tolerance for steps without improvement for DEoptim

iter2
Integer. Number of iterations for the second optimization

rademacher
deprecated, use "design" instead.

Value

A list of class "sboot" with elements

true
Point estimate of impulse response functions

bootstrap
List of length "nboot" holding bootstrap impulse response functions

SE
Bootstrapped standard errors of estimated covariance decomposition (only if "x"
has method "Cramer von-Mises", or "Distance covariances")

nboot
Number of bootstrap iterations

distr
Character, whether the Gaussian, Rademacher or Mammen distribution is used
in the bootstrap

design
character. Whether a fixed design or recursive design bootstrap is performed

point_estimate
Point estimate of covariance decomposition

boot_mean
Mean of bootstrapped covariance decompositions

signrest
Evaluated sign pattern

sign_complete
Frequency of appearance of the complete sign pattern in all bootstrapped covari-
ance decompositions
wild.boot

sign_part
Frequency of bootstrapped covariance decompositions which conform the complete predetermined sign pattern. If signrest=NULL, the frequency of bootstrapped covariance decompositions that hold the same sign pattern as the point estimate is provided.

sign_part
Frequency of single shocks in all bootstrapped covariance decompositions which accord to a specific predetermined sign pattern

cov_bs
Covariance matrix of bootstrapped parameter in impact relations matrix

method
Used bootstrap method

References

See Also
id.cvm, id.dc, id.garch, id.ngml, id.cv or id.st

Examples

# data contains quarterly observations from 1965Q1 to 2008Q3
# x = output gap
# pi = inflation
# i = interest rates
set.seed(23211)
v1 <- vars::VAR(USA, lag.max = 10, ic = "AIC")
x1 <- id.dc(v1)
summary(x1)

# impulse response analysis with confidence bands
# Checking how often theory based impact relations appear
signrest <- list(demand = c(1,1,1), supply = c(-1,1,1), money = c(-1,-1,1))
bb <- wild.boot(x1, nboot = 500, n.ahead = 30, nc = 1, signrest = signrest)
summary(bb)
plot(bb, lowerq = 0.16, upperq = 0.84)
Index

*Topic **Empirical**
  stability, 26

*Topic **Fluctuation**
  stability, 26

*Topic **Structural**
  stability, 26

*Topic **VAR**
  stability, 26

*Topic **Vector**
  stability, 26

*Topic **autoregressive**
  stability, 26

*Topic **regression**
  stability, 26

ba.boot, 2

chow.test, 4, 26, 27

efp, 26, 27

fevd, 6

hd, 7

id.cv, 6, 7, 8, 12, 13, 15, 17, 20, 21, 25, 28, 31

id.cvm, 6, 7, 9, 10, 13, 15, 17, 20, 21, 25, 28, 31

id.dc, 6, 7, 9, 12, 15, 17, 20, 21, 25, 28, 31

id.garch, 6, 7, 9, 12, 13, 14, 17, 20, 25, 28, 31

id.ngml, 6, 7, 9, 12, 13, 15, 16, 20, 21, 25, 28, 31

id.st, 6, 7, 9, 12, 13, 15, 17, 18, 21, 25, 28, 31

irf, 21

js.test, 22

LN, 23

mb.boot, 3, 22, 24

plot, 27