Package: pvars (via r-universe)

November 21, 2026

Type Package

Title VAR Modeling for Heterogeneous Panels

Version 1.1.1

Maintainer Lennart Empting <lennart.empting@vwl.uni-due.de>

Description Implements (1) panel cointegration rank tests, (2) estimators for panel vector autoregressive (VAR) models, and (3) identification methods for panel structural vector autoregressive (SVAR) models as described in the accompanying vignette. The implemented functions allow to account for cross-sectional dependence and for structural breaks in the deterministic terms of the VAR processes. Among the large set of functions, particularly noteworthy are those that implement (1) the correlation-augmented inverse normal test on the cointegration rank by Arsova and Oersal (2021, <doi:10.1016/j.ecosta.2020.05.002>), (2) the two-step estimator for pooled cointegrating vectors by Breitung (2005, <doi:10.1081/ETC-200067895>), and (3) the pooled identification based on independent component analysis by Herwartz and Wang (2024, <doi:10.1002/jae.3044>).

License MIT + file LICENSE

URL https://github.com/Lenni89/pvars

BugReports https://github.com/Lenni89/pvars/issues

Depends R (>= 3.5.0), svars (>= 1.3.4)

Imports clue, copula, DEoptim, expm, ggplot2, MASS, pbapply, reshape2, scales, stats, steadyICA, utils, vars

Suggests ggfortify, ggpubr, knitr, plm, RColorBrewer, testthat (>= 2.1.0), tikzDevice, urca

Encoding UTF-8

LazyData TRUE

VignetteBuilder knitr

NeedsCompilation no

RoxygenNote 7.3.3

Author Lennart Empting [aut, cre, cph] (ORCID:
<https://orcid.org/0009-0004-5068-4639>)

Config/pak/sysreqs libgsl0-dev libicu-dev

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

Date/Publication 2026-11-21 02:02:00 UTC

RemoteUrl https://github.com/cranhaven/cranhaven.r-universe.dev

RemoteRef package/pvars

RemoteSha 68c63a0dbe9c614a002f80e5995351628d5893c4

RemoteSubdir pvars

Contents

as.pplot . . . 3
as.pvarx . . . 5
as.t_D . . . 7
as.varx . . . 8
coint . . . 10
ERPT . . . 12
EU_w . . . 13
EURO . . . 14
fevd.id . . . 15
ICAP . . . 16
id.grt . . . 17
id.iv . . . 19
irf.pvarx . . . 21
irf.varx . . . 22
MDEM . . . 24
MERM . . . 25
PCAP . . . 25
PCIT . . . 28
pcoint . . . 29
pid.chol . . . 32
pid.cvm . . . 33
pid.dc . . . 36
pid.grt . . . 39
pid.iv . . . 41
PP . . . 43
pvars . . . 45
pvarx . . . 46
rboot.normality . . . 49
sboot.mb . . . 50
sboot.mg . . . 53
sboot.pmb . . . 55
speci.factors . . . 58

speci.VAR 60
VECM 63

Index **66**

`as.pplot` *Coerce into a* '`pplot`' *object*

Description

Coerce into a '`pplot`' object. This function allows (1) to overlay and colorize multiple plots of IRF and confidence bands in a single '`ggplot`' object and (2) to overwrite shock and variable names in verbatim LaTeX math mode suitable for the export via **tikzDevice**.

Usage

```
as.pplot(
  ...,
  names_k = NULL,
  names_s = NULL,
  names_g = NULL,
  color_g = NULL,
  shape_g = NULL,
  n.rows = NULL,
  scales = "free_y",
  Latex = FALSE
)
```

Arguments

`...`	A single ggplot or list(s) of ggplots to be transformed.
`names_k`	Vector. Names of the variables $k = 1, \ldots, K$. If `NULL` (the default), the names of the first plot are reused. For LaTeX exports, use e.g. `paste0("y_{ ", 1:dim_K, " }")`.
`names_s`	Vector. Names of the shocks $s = 1, \ldots, S$. If `NULL` (the default), the names of the first plot are reused. For LaTeX exports, use e.g. `paste0("\\epsilon_{ ", 1:dim_S, " }")`.
`names_g`	Vector. Names of the layered plots $g = 1, \ldots, G$. If `NULL` (the default), the names of the plots given in `...` are reused.
`color_g`	Vector. Colors of the layered plots $g = 1, \ldots, G$.
`shape_g`	Vector. Shapes of the layered plots $g = 1, \ldots, G$, see **ggplot2**'s `geom_point` for shapes. If `NULL` (the default), no points will be set on the IRF-lines.
`n.rows`	Integer. Number of rows in `facet_wrap`. If `NULL` (the default), the dimensions of the facet plots given in `...` are reused.
`scales`	Character. Should scales be fixed (`"fixed"`), free (`"free"`), or free in one dimension (`"free_x"`, `"free_y"`, the default)? See `facet_wrap`.
`Latex`	Logical. If `TRUE`, the arrows of the facet labels are written in LaTeX math mode.

Details

as.pplot is used as an intermediary in the 'pplot' functions to achieve compatibility with different classes. Equivalently to as.pvarx for panels of N VAR objects, as.pplot summarizes panels of G plot objects that have been returned from the 'plot' method for class 'svarirf' or 'sboot'. If the user wishes to extend the compatibility of the 'pplot' functions with further classes, she may simply specify accordant as.pplot-methods instead of altering the original 'pplot' functions.

Value

A list of class 'pplot'. Objects of this class contain the elements:

F.plot	'ggplot' object for the merged plot.
L.plot	List of 'ggplot' objects containing all G plots.
args_pplot	List of characters and integers indicating the specifications used for creating F.plot.

See Also

PP, irf.pvarx, pid.dc, and id.iv for further examples of edited plots, in particular for subset and reordered facet plots with reshaped facet dimensions.

Examples

```
### gallery of merged IRF plots ###

library("ggplot2")
data("PCAP")
names_k = c("g", "k", "l", "y")  # variable names
names_i = levels(PCAP$id_i)      # country names
L.data  = sapply(names_i, FUN=function(i)
  ts(PCAP[PCAP$id_i==i, names_k], start=1960, end=2019, frequency=1),
  simplify=FALSE)
L.vars = lapply(L.data, FUN=function(x) vars::VAR(x, p=2, type="both"))
L.chol = lapply(L.vars, FUN=function(x) svars::id.chol(x))

# overlay all IRF to get an overview on the stability #
L.irf = lapply(L.chol, FUN=function(x) plot(irf(x, n.ahead=30)))
summary(as.pvarx(L.vars))
as.pplot(L.irf)

# overlay IRF of selected countries and quantiles of all countries #
F.mg  = plot(sboot.mg(L.chol, n.ahead=30), lowerq=0.05, upperq=0.95)
R.irf = as.pplot(MG=F.mg, L.irf[c("DEU", "FRA", "ITA", "JPN")])
plot(R.irf)  # emphasize MG-IRF in next step
R.irf = as.pplot(R.irf, color_g="black", shape_g=c(20, rep(NA, 4)))
R.irf$F.plot + guides(fill="none") + labs(color="Country", shape="Country")

# compare two mean-groups and their quantiles #
idx_nord = c(5, 6, 10, 17, 20)  # Nordic countries
R.irf = as.pplot(color_g=c("black", "blue"),
  Other  = plot(sboot.mg(L.chol[-idx_nord])),
```

```
  Nordic = plot(sboot.mg(L.chol[ idx_nord])))
plot(R.irf)

# compare different shock ordering for MG-IRF #
R.pid1 = pid.chol(L.vars)
R.pid2 = pid.chol(L.vars, order_k=4:1)
R.pid3 = pid.chol(L.vars, order_k=c(1,4,2,3))

R.pal = RColorBrewer::brewer.pal(n=8, name="Spectral")[c(8, 1, 4)]
R.irf = as.pplot(color_g=R.pal, shape_g=c(2, 3, 20),
  GKLY = plot(irf(R.pid1, n.ahead=25)),
  YLKG = plot(irf(R.pid2, n.ahead=25)),
  GYKL = plot(irf(R.pid3, n.ahead=25)))
R.mg = as.pplot(color_g=R.pal, shape_g=c(2, 3, 20),
  GKLY = plot(sboot.mg(R.pid1, n.ahead=25), lowerq=0.05, upperq=0.95),
  YLKG = plot(sboot.mg(R.pid2, n.ahead=25), lowerq=0.05, upperq=0.95),
  GYKL = plot(sboot.mg(R.pid3, n.ahead=25), lowerq=0.05, upperq=0.95))

# colorize and export a single sub-plot to Latex #
library("tikzDevice")
textwidth = 15.5/2.54  # LaTeX textwidth from "cm" into "inch"
file_fig  = file.path(tempdir(), "Fig_irf.tex")

R.irf = as.pplot(
  DEU = plot(irf(L.chol[["DEU"]], n.ahead=50), selection=list(4, 1)),
  FRA = plot(irf(L.chol[["FRA"]], n.ahead=50), selection=list(4, 1)),
  color_g = c("black", "blue"),
  names_g = c("Germany", "France"),
  names_k = "y",
  names_s = "\\epsilon_{ g }",
  Latex   = TRUE)

tikz(file=file_fig, width=1.2*textwidth, height=0.8*textwidth)
  R.irf$F.plot + labs(color="Country") + theme_minimal()
dev.off()
```

`as.pvarx` *Coerce into a* 'pvarx' *object*

Description

Coerce into a 'pvarx' object. On top of the parent class 'pvarx', the child class 'pid' is imposed if the input object to be transformed contains a panel SVAR model.

Usage

```
as.pvarx(x, w = NULL, ...)
```

Arguments

x	A panel VAR object to be transformed.
w	Numeric, logical, or character vector. N numeric elements weighting the individual coefficients, or names or N logical elements selecting a subset from the individuals $i = 1, \ldots, N$ for the MG estimation. If NULL (the default), all N individuals are included without weights.
...	Additional arguments to be passed to or from methods.

Details

as.pvarx is used as an intermediary in the **pvars** functions to achieve compatibility with different classes of panel VAR objects. If the user wishes to extend this compatibility with further classes, she may simply specify accordant as.pvarx-methods instead of altering the original **pvars** function.

Value

A list of class 'pvarx'. Objects of this class contain the elements:

A	Matrix. The lined-up coefficient matrices $A_j, j = 1, \ldots, p$ for the lagged variables in the panel VAR.
B	Matrix. The $(K \times S)$ structural impact matrix of the panel SVAR model or an identity matrix I_K as a placeholder for the unidentified VAR model.
beta	Matrix. The $((K + n_{d1}) \times r)$ cointegrating matrix of the VAR model if transformed from a rank-restricted VECM.
L.varx	List of varx objects for the individual estimation results.
args_pvarx	List of characters and integers indicating the estimator and specifications that have been used.
args_pid	List of characters and integers indicating the identification methods and specifications that have been used. This element is specific to the child-class 'pid' for panel SVAR models, that inherit from parent-class 'pvarx' for any panel VAR model.

Examples

```
data("PCAP")
names_k = c("g", "k", "l", "y")  # variable names
names_i = levels(PCAP$id_i)      # country names
L.data  = sapply(names_i, FUN=function(i)
  ts(PCAP[PCAP$id_i==i, names_k], start=1960, end=2019, frequency=1),
  simplify=FALSE)

L.vars = lapply(L.data, FUN=function(x) vars::VAR(x, p=2, type="both"))
as.pvarx(L.vars)
```

as.t_D *Deterministic regressors in* **pvars**

Description

Deterministic regressors can be specified via the arguments of the ***conventional*** 'type', ***customized*** 'D', and ***period-specific*** 't_D'. While 'type' is a single character and 'D' a data matrix of dimension $(n_\bullet \times (p+T))$, the specifications for τ in the list 't_D' are more complex and therefore preventively checked by as.t_D.

Usage

```
as.t_D(x, ...)
```

Arguments

x	A list of vectors for τ to be checked. Since 'x' is just checked, Section "Value" explains function-input and -output likewise.
...	Additional arguments to be passed to or from methods.

Value

A list of class 't_D' specifying τ. Objects of this class can exclusively contain the elements:

t_break	Vector of integers. The activating periods for trend breaks $d = [\ldots, 0, 0, 1, 2, 3, \ldots]$.
t_shift	Vector of integers. The activating periods for shifts in the constant $d = [\ldots, 0, 0, 1, 1, 1, \ldots]$.
t_impulse	Vector of integers. The activating periods for single impulses $d = [\ldots, 0, 0, 1, 0, 0, \ldots]$.
t_blip	Vector of integers. The activating period for blips $d = [\ldots, 0, 0, 1, -1, 0, \ldots]$.
n.season	Integer. The number of seasons.

Reference Time Interval

The complete time series (i.e. including the presample) constitutes the reference time interval. Accordingly, 'D' contains $p+T$ observations, and 't_D' contains the positions of activating periods τ in $1, \ldots, (p+T)$. In a balanced panel $p_i + T_i = T^*$, the same date implies the same τ in $1, \ldots, T^*$, as shown in the example for pcoint.CAIN. However, in an unbalanced panel, the same date can imply different τ across i in accordance with the individual time interval $1, \ldots, (p_i + T_i)$. Note that across the time series in 'L.data', it is the last observation in each data matrix that refers to the same date.

Conventional Type

An overview is given here and a detailed explanation in the package vignette.

- **type (VAR)** is specified in VAR models just as in **vars**' VAR, namely by a 'const', a linear 'trend', 'both', or 'none' of those.

- **type_SL** is used in the 'additive' SL procedure for testing the cointegration rank only, which removes the mean ('SL_mean') or mean and linear trend ('SL_trend') by GLS-detrending.
- **type (VECM)** is used in the 'innovative' Johansen procedure for testing the cointegration rank and estimating the VECM. In accordance with Juselius (2007, Ch.6.3), the available model specifications are: 'Case1' for none, 'Case2' for a constant in the cointegration relation, 'Case3' for an unrestricted constant, 'Case4' for a linear trend in the cointegration relation and an unrestricted constant, or 'Case5' for an unrestricted constant and linear trend.

References

Juselius, K. (2007): *The Cointegrated VAR Model: Methodology and Applications*, Advanced Texts in Econometrics, Oxford University Press, USA, 2nd ed.

Examples

```
t_D = list(t_impulse=c(10, 20, 35), t_shift=10)
as.t_D(t_D)
```

---

as.varx *Coerce into a 'varx' object*

Description

Coerce into a 'varx' object. On top of the parent class 'varx', the child class 'id' is imposed if the input object to be transformed contains an SVAR model.

Usage

```
as.varx(x, ...)
```

Arguments

x	A VAR object to be transformed.
...	Additional arguments to be passed to or from methods.

Details

as.varx is used as an intermediary in the **pvars** functions to achieve compatibility with different classes of VAR objects. If the user wishes to extend this compatibility with further classes, she may simply specify accordant as.varx-methods instead of altering the original **pvars** function. Classes already covered by **pvars** are those of the **vars** ecosystem, in particular the classes

- 'varest' for reduced-form VAR estimates from VAR,
- 'vec2var' for reduced-form VECM estimates from vec2var,
- 'svarest' for structural VAR estimates from BQ,
- 'svecest' for structural VECM estimates from SVEC, and

- 'svars' for structural VAR estimates from **svars**' id.chol, id.cvm, or id.dc.

By transformation to 'varx', these VAR estimates can thus be subjected to **pvars**' bootstrap procedure sboot.mb and S3 methods such as summary and toLatex.

Value

A list of class 'varx'. Objects of this class contain the elements:

A	Matrix. The lined-up VAR coefficient matrices $A_j, j = 1, \ldots, p$ for the lagged variables.
B	Matrix. The $(K \times S)$ structural impact matrix of the SVAR model or an identity matrix I_K as a placeholder for the unidentified VAR model.
beta	Matrix. The $((K + n_{d1}) \times r)$ cointegrating matrix of the VAR model if transformed from a rank-restricted VECM.
SIGMA	Matrix. The $(K \times K)$ residual covariance matrix estimated by least-squares.

The following integers indicate the size of dimensions:

dim_K	Integer. The number of endogenous variables K in the full-system.
dim_S	Integer. The number of identified shocks S in the SVAR model.
dim_T	Integer. The number of time periods T without presample.
dim_p	Integer. The lag-order p of the VAR model in levels.
dim_r	Integer. The cointegration rank r of the VAR model if transformed from a rank-restricted VECM.

Some further elements required for the bootstrap functions are:

y	Matrix. The $(K \times (p + T))$ endogenous variables.
D, D1, D2	Matrices. The $(n_\bullet \times (p + T))$ deterministic variables, fixed over bootstrap iterations, (un)restricted to the cointegration relations of the VAR model if transformed from a rank-restricted VECM.
resid	Matrix. The $(K \times T)$ residual matrix.
args_varx	List of characters and integers indicating the estimator and specifications that have been used.
args_id	List of characters and integers indicating the identification methods and specifications that have been used. This element is specific to the child-class 'id' for SVAR models, that inherit from parent-class 'varx' for any VAR model.

Examples

```
data("PCIT")
names_k = c("APITR", "ACITR", "PITB", "CITB", "GOV", "RGDP", "DEBT")

# estimate reduced-form VAR and coerce into 'varx' object #
R.vars = vars::VAR(PCIT[ , names_k], p=4, type="const")
as.varx(R.vars)
```

```
# identify structural VAR and coerce into 'id' object #
R.svar = svars::id.chol(R.vars, order_k=names_k)
as.varx(R.svar)
```

`coint` *Test procedures for the cointegration rank*

Description

Performs test procedures for the rank of cointegration in a single VAR model. The p-values are approximated by gamma distributions, whose moments are automatically adjusted to potential period-specific deterministic regressors and weakly exogenous regressors in the partial VECM.

Usage

```
coint.JO(
  y,
  dim_p,
  x = NULL,
  dim_q = dim_p,
  type = c("Case1", "Case2", "Case3", "Case4", "Case5"),
  t_D1 = NULL,
  t_D2 = NULL
)

coint.SL(y, dim_p, type_SL = c("SL_mean", "SL_trend"), t_D = NULL)
```

Arguments

`y`	Matrix. A $(K \times (p+T))$ data matrix of the K endogenous time series variables.
`dim_p`	Integer. Lag-order p for the endogenous variables `y`.
`x`	Matrix. A $(L \times (q+T))$ data matrix of the L weakly exogenous time series variables.
`dim_q`	Integer. Lag-order q for the weakly exogenous variables `x`. The literature uses `dim_p` (the default).
`type`	Character. The conventional case of the deterministic term in the Johansen procedure.
`t_D1`	List of vectors. The activating break periods τ for the period-specific deterministic regressors in $d_{1,t}$, which are restricted to the cointegration relations. The accompanying lagged regressors are automatically included in $d_{2,t}$. The p-values are calculated for up to two breaks resp. three sub-samples.
`t_D2`	List of vectors. The activating break periods τ for the period-specific deterministic regressors in $d_{2,t}$, which are unrestricted.
`type_SL`	Character. The conventional case of the deterministic term in the Saikkonen-Luetkepohl (SL) procedure.

`t_D` List of vectors. The activation periods τ for the period-specific deterministic regressors in d_t of the SL-procedure. The accompanying lagged regressors are automatically included in d_t. The p-values are calculated for up to two breaks resp. three sub-samples.

Value

A list of class '`coint`', which contains elements of length K for each $r_{H0} = 0, \ldots, K-1$:

`r_H0`	Rank under each null hypothesis.
`stats_TR`	Trace (TR) test statistics.
`stats_ME`	Maximum eigenvalue (ME) test statistics.
`pvals_TR`	p-values of the TR test.
`pvals_ME`	p-values of the ME test. `NA` if moments of the gamma distribution are not available for the chosen data generating process.
`lambda`	Eigenvalues, the squared canonical correlation coeffcients (saved only for the Johansen procedure).
`args_coint`	List of characters and integers indicating the cointegration test and specifications that have been used.

Functions

- `coint.JO()`: Johansen procedure.
- `coint.SL()`: (Trenkler)-Saikkonen-Luetkepohl procedure.

References

Johansen, S. (1988): "Statistical Analysis of Cointegration Vectors", *Journal of Economic Dynamics and Control*, 12, pp. 231-254.

Doornik, J. (1998): "Approximations to the Asymptotic Distributions of Cointegration Tests", *Journal of Economic Surveys*, 12, pp. 573-93.

Johansen, S., Mosconi, R., and Nielsen, B. (2000): "Cointegration Analysis in the Presence of Structural Breaks in the Deterministic Trend", *Econometrics Journal*, 3, pp. 216-249.

Kurita, T., Nielsen, B. (2019): "Partial Cointegrated Vector Autoregressive Models with Structural Breaks in Deterministic Terms", *Econometrics*, 7, pp. 1-35.

Saikkonen, P., and Luetkepohl, H. (2000): "Trend Adjustment Prior to Testing for the Cointegrating Rank of a Vector Autoregressive Process", *Journal of Time Series Analysis*, 21, pp. 435-456.

Trenkler, C. (2008): "Determining p-Values for Systems Cointegration Tests with a Prior Adjustment for Deterministic Terms", *Computational Statistics*, 23, pp. 19-39.

Trenkler, C., Saikkonen, P., and Luetkepohl, H. (2008): "Testing for the Cointegrating Rank of a VAR Process with Level Shift and Trend Break", *Journal of Time Series Analysis*, 29, pp. 331-358.

Examples

```
### reproduce basic example in "urca" ###
library("urca")
data(denmark)
sjd = denmark[ , c("LRM", "LRY", "IBO", "IDE")]

# rank test and estimation of the full VECM as in "urca" #
R.JOrank = coint.JO(y=sjd,      dim_p=2, type="Case2", t_D2=list(n.season=4))
R.JOvecm = VECM(y=sjd, dim_r=1, dim_p=2, type="Case2", t_D2=list(n.season=4))

# ... and of the partial VECM, i.e. after imposing weak exogeneity #
R.KNrank = coint.JO(y=sjd[ , c("LRM"), drop=FALSE],   dim_p=2,
                    x=sjd[ , c("LRY", "IBO", "IDE")], dim_q=2,
                    type="Case2", t_D1=list(t_shift=36), t_D2=list(n.season=4))
R.KNvecm = VECM(y=sjd[ , c("LRM"), drop=FALSE],   dim_p=2,
                x=sjd[ , c("LRY", "IBO", "IDE")], dim_q=2, dim_r=1,
                type="Case2", t_D1=list(t_shift=36), t_D2=list(n.season=4))

### reproduce Oersal,Arsova 2016:22, Tab.7.5 "France" ###
data("ERPT")
names_k = c("lpm5", "lfp5", "llcusd")  # variable names for "Chemicals and related products"
names_i = levels(ERPT$id_i)[c(1,6,2,5,4,3,7)]  # ordered country names
L.data  = sapply(names_i, FUN=function(i)
   ts(ERPT[ERPT$id_i==i, names_k], start=c(1995, 1), frequency=12),
   simplify=FALSE)
R.TSLrank = coint.SL(y=L.data$France, dim_p=3, type_SL="SL_trend", t_D=list(t_break=89))
```

ERPT *Data set on the* Exchange Rate Pass-Through

Description

The data set ERPT consists of monthly observations for the logarithm of import prices lm^*, foreign prices lf^*, and the exchange rate against the US dollar $llcusd$. It covers the period January 1995 to March 2005 ($T = 123$) for $N = 7$ countries. The asterisk denotes the industry of the variables and can take values from 0 to 8:

- 0: *Food and live animals chiefly for food*
- 1: *Beverages and tobacco*
- 2: *Crude materials (inedible, except fuels)*
- 3: *Mineral fuels, lubricants and related materials*
- 4: *Animal and vegetable oils, fats and waxes*
- 5: *Chemicals and related products*
- 6: *Manufactured goods classified chiefly by materials*
- 7: *Machines, transport equipment*
- 8: *Manufactured goods*

Usage

```
data("ERPT")
```

Format

A long-format data panel of class 'data.frame', where the columns id_i and id_t indicate the country and month respectively.

Source

The prepared *Eurostat* data set is directly obtainable from the *ZBW Journal Data Archive*: doi:10.15456/jae.2022321.0717881037. This is open data under the CC BY 4.0 license.

References

Banerjee, A., and Carrion-i-Silvestre, J. L. (2015): "Cointegration in Panel Data with Structural Breaks and Cross-Section Dependence", *Journal of Applied Econometrics*, 30 (1), pp. 1-23.

See Also

Other data sets: EURO, EU_w, ICAP, MDEM, MERM, PCAP, PCIT

EU_w *Weights for the* Euro Monetary Policy Transmission

Description

The data set EU_w is a vector of 14 elements. These are weights for $N = 14$ member countries of the Euro area, constructed as the average share of their respective real GDP over the sample period in Herwartz, Wang (2024).

Usage

```
data("EU_w")
```

Format

A numeric vector containing 14 named elements.

Source

The prepared *Eurostat* data set is directly obtainable from the *ZBW Journal Data Archive*: doi:10.15456/jae.2024044.1425287131. This is open data under the CC BY 4.0 license in accordance with the deposit license of the *ZBW Journal Data Archive*.

References

Herwartz, H., and Wang, S. (2024): "Statistical Identification in Panel Structural Vector Autoregressive Models based on Independence Criteria", *Journal of Applied Econometrics*, 39 (4), pp. 620-639.

See Also

Other data sets: ERPT, EURO, ICAP, MDEM, MERM, PCAP, PCIT

EURO *Data set on the* Euro Monetary Policy Transmission

Description

The data set EURO is a list of 15 'data.frame' objects, each consisting of quarterly observations for

- the first-difference of log real GDP on national dl_GDP or aggregated EA-level EA_dl_GDP,
- the annualized inflation of the (log) GDP deflator on national $dl_deflator$ or aggregated EA-level EA_pi,
- the EA-wide short-term interest rate IR,
- the EA-wide option-adjusted bond spreads BBB,
- the first-difference of log real GDP in the remaining countries dl_GDP_EA,
- the weighted inflation in the remaining countries $dl_deflator_EA$,
- the inflation of a world commodity price index WCP,
- the US effective federal funds rate US_FFR,
- the trade volume in percentage of GDP $trade$, and
- the government spending in percentage of GDP ge.

The data covers the period Q1 2001 to Q1 2020 ($T = 77$) for the aggregate of the Euro area (EA, first element in list) and $N = 14$ of its member countries (subsequent 14 elements in list).

Usage

```
data("EURO")
```

Format

A list-format data panel of class 'list' containing 15 'data.frame' objects with named time series.

Source

The prepared *Eurostat* data set is directly obtainable from the *ZBW Journal Data Archive*: doi:10.15456/jae.2024044.1425287131. This is open data under the CC BY 4.0 license in accordance with the deposit license of the *ZBW Journal Data Archive*.

References

Herwartz, H., and Wang, S. (2024): "Statistical Identification in Panel Structural Vector Autoregressive Models based on Independence Criteria", *Journal of Applied Econometrics*, 39 (4), pp. 620-639.

See Also

Other data sets: ERPT, EU_w, ICAP, MDEM, MERM, PCAP, PCIT

`fevd.id` *Forecast Error Variance Decomposition*

Description

Calculates the forecast error variance decomposition. Respects SVAR models of cases $S \neq K$, i.e. partially identified or excess shocks, too.

Usage

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

Arguments

x	SVAR object of class 'id' or any other that will be coerced to ('id', 'varx').
n.ahead	Integer specifying the steps ahead, i.e. the horizon of the FEVD.
...	Currently not used.

Value

A list of class 'svarfevd' holding the forecast error variance decomposition of each variables as a 'data.frame'.

References

Luetkepohl, H. (2005): *New Introduction to Multiple Time Series Analysis*, Springer, 2nd ed.

Jentsch, Lunsford (2022): "Asymptotically Valid Bootstrap Inference for Proxy SVARs", *Journal of Business and Economic Statistics*, 40, pp. 1876-1891.

Examples

```
data("PCIT")
names_k = c("APITR", "ACITR", "PITB", "CITB", "GOV", "RGDP", "DEBT")
names_l = c("m_PI", "m_CI")  # proxy names
names_s = paste0("epsilon[ ", c("PI", "CI"), " ]")  # shock names
dim_p   = 4  # lag-order

# estimate and identify proxy SVAR #
R.vars = vars::VAR(PCIT[ , names_k], p=dim_p, type="const")
R.idBL = id.iv(R.vars, iv=PCIT[-(1:dim_p), names_l], S2="MR", cov_u="OMEGA")
colnames(R.idBL$B) = names_s  # labeling

# calculate and plot FEVD under partial identification #
plot(fevd(R.idBL, n.ahead=20))
```

ICAP *Data set on* Infrastructure Capital Stocks

Description

The data set ICAP consists of annual observations for

- the real GDP Y in US dollars,
- the physical capital stocks K in US dollars,
- the physical capital stocks $KBC30$ via "backcasting" in US dollars,
- the number of workers $LWDI$ providing the total labor force, and
- the average years of secondary education $secondary$ of the population.

It further reports physical measures of infrastructure given by

- the electricity generation capacity EGC in megawatts,
- the number of main phone lines $mlines$,
- the kilometers of total roads $troads$,
- the number of cell phones lines $cells$,
- the kilometers of paved roads $proads$, and
- the kilometers of rails $rails$.

It covers the period 1960 to 2000 ($T = 41$) for $N = 97$ countries. The monetary values are given in US-Dollars at 2000 prices, i.e. constant PPP.

Usage

```
data("ICAP")
```

Format

A long-format data panel of class 'data.frame', where the columns id_i and id_t indicate the country and year respectively. Column COUNTRY contains the complete country names.

Source

The prepared data set is directly obtainable from the *ZBW Journal Data Archive*: doi:10.15456/jae.2022321.0717368489. This is open data under the CC BY 4.0 license.

References

Calderon, C., Moral-Benito, E., and Serven, L. (2015): "Is Infrastructure Capital Productive? A Dynamic Heterogeneous Approach", *Journal of Applied Econometrics*, 30 (2), pp. 177-198.

See Also

Other data sets: ERPT, EURO, EU_w, MDEM, MERM, PCAP, PCIT

---

id.grt *Identification of SVEC models by imposing long- and short-run restrictions*

Description

Identifies an SVEC model by utilizing a scoring algorithm to impose long- and short-run restrictions. See the details of SVEC in **vars**.

Usage

```
id.grt(
  x,
  LR = NULL,
  SR = NULL,
  start = NULL,
  max.iter = 100,
  conv.crit = 1e-07,
  maxls = 1
)
```

Arguments

x	VAR object of class 'varx' estimated under rank-restriction or any other that will be coerced to 'varx'.
LR	Matrix. The restricted long-run impact matrix.
SR	Matrix. The restricted contemporaneous impact matrix.
start	Vector. The starting values for γ, set by rnorm if NULL (the default).

max.iter	Integer. The maximum number of iterations.
conv.crit	Real number. Convergence value of algorithm.
maxls	Real number. Maximum movement of the parameters between two iterations of the scoring algorithm.

Value

List of class 'id'.

References

Amisano, G. and Giannini, C. (1997): *Topics in Structural VAR Econometrics*, Springer, 2nd ed.

Breitung, J., Brueggemann R., and Luetkepohl, H. (2004): "Structural Vector Autoregressive Modeling and Impulse Responses", in *Applied Time Series Econometrics*, ed. by H. Luetkepohl and M. Kraetzig, Cambridge University Press, Cambridge.

Johansen, S. (1996): *Likelihood-Based Inference in Cointegrated Vector Autoregressive Models*, Advanced Texts in Econometrics, Oxford University Press, USA.

Luetkepohl, H. (2005): *New Introduction to Multiple Time Series Analysis*, Springer, 2nd ed.

Pfaff, B. (2008): "VAR, SVAR and SVEC Models: Implementation within R Package **vars**", *Journal of Statistical Software*, 27, pp. 1-32.

See Also

...the original SVEC by Pfaff (2008) in **vars**. Note that id.grt is just a graftage, but allows for the additional model specifications in VECM and for the bootstrap procedures in sboot.mb, both provided by the **pvars** package.

Other identification functions: id.iv()

Examples

```
### reproduce basic example in "vars" ###
library(vars)
data("Canada")
names_k = c("prod", "e", "U", "rw")  # variable names
names_s = NULL  # optional shock names

# colnames of the restriction matrices are passed as shock names #
SR = matrix(NA, nrow=4, ncol=4, dimnames=list(names_k, names_s))
SR[4, 2] = 0
LR = matrix(NA, nrow=4, ncol=4, dimnames=list(names_k, names_s))
LR[1, 2:4] = 0
LR[2:4, 4] = 0

# estimate and identify SVECM #
R.vecm = VECM(y=Canada[ , names_k], dim_p=3, dim_r=1, type="Case4")
R.grt  = id.grt(R.vecm, LR=LR, SR=SR)
```

`id.iv` *Identification of SVAR models by means of proxy variables*

Description

Given an estimated VAR model, this function uses proxy variables to partially identify the structural impact matrix B of the corresponding SVAR model

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

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

In general, identification procedures determine B up to column ordering, scale, and sign. For a unique solution, `id.iv` follows the literature on proxy SVAR. The S columns in $B = [B_1 : B_2]$ of the identified shocks $\epsilon_{ts}, s = 1, \ldots, S$, are ordered first, and the variance $\sigma^2_{\epsilon,s} = 1$ is normalized to unity (see e.g. Lunsford 2015:6, Eq. 9). Further, the sign is fixed to a positive correlation between proxy and shock series. A normalization of the impulsed shock that may fix the size of the impact response in the IRF can be imposed subsequently via '`normf`' in `irf.varx` and `sboot.mb`.

Usage

```
id.iv(x, iv, S2 = c("MR", "JL", "NQ"), cov_u = "OMEGA", R0 = NULL)
```

Arguments

`x`	VAR object of class '`varx`' or any other that will be coerced to '`varx`'.
`iv`	Matrix. A $(L \times T)$ data matrix of the L proxy time series m_t.
`S2`	Character. Identification within multiple proxies m_t via '`MR`' for lower-triangular $[I_S : -B_{11}B_{12}^{-1}]B_1$ by Mertens, Ravn (2013), via '`JL`' for chol$(\Sigma_{mu}\Sigma_u^{-1}\Sigma_{um})$ by Jentsch, Lunsford (2021), or via '`NQ`' for the nearest orthogonal matrix from `svd` decomposition by Empting et al. (2025). In case of $S = L = 1$ proxy, all three choices provide identical results on B_1.
`cov_u`	Character. Selection of the estimated residual covariance matrix $\hat{\Sigma}_u$ to be used in the identification procedure. Either `'OMEGA'` (the default) for $\hat{U}\hat{U}'/T_i$ as used in Mertens, Ravn (2013) and Jentsch, Lunsford (2021) or `'SIGMA'` for $\hat{U}\hat{U}'/(T - n_z)$, which corrects for the number of regressors n_z. Both character options refer to the name of the respective estimate in the '`varx`' object.
`R0`	Matrix. A $(L \times S)$ selection matrix for '`NQ`' that governs the attribution of the L proxies to their specific S structural shock series. If `NULL` (the default), `R0` $= I_S$ will be used such that the $S = L$ columns of B_1 are one-by-one estimated from the $S = L$ proxy series '`iv`' available.

Value

List of class '`id`'.

References

Mertens, K., and Ravn, M. O. (2013): "The Dynamic Effects of Personal and Corporate Income Tax Changes in the United States", *American Economic Review*, 103, pp. 1212-1247.

Lunsford, K. G. (2015): "Identifying Structural VARs with a Proxy Variable and a Test for a Weak Proxy", Working Paper, No 15-28, Federal Reserve Bank of Cleveland.

Jentsch, C., and Lunsford, K. G. (2019): "The Dynamic Effects of Personal and Corporate Income Tax Changes in the United States: Comment", *American Economic Review*, 109, pp. 2655-2678.

Jentsch, C., and Lunsford, K. G. (2021): "Asymptotically Valid Bootstrap Inference for Proxy SVARs", *Journal of Business and Economic Statistics*, 40, pp. 1876-1891.

Empting, L. F. T., Maxand, S., Oeztuerk, S., and Wagner, K. (2025): "Inference in Panel SVARs with Cross-Sectional Dependence of Unknown Form".

See Also

. . . the individual identification approaches by Lange et al. (2021) in **svars**.

Other identification functions: `id.grt()`

Examples

```
### reproduce Jentsch,Lunsford 2019:2668, Ch.III ###
data("PCIT")
names_k = c("APITR", "ACITR", "PITB", "CITB", "GOV", "RGDP", "DEBT")
names_l = c("m_PI", "m_CI")  # proxy names
names_s = paste0("epsilon[ ", c("PI", "CI"), " ]")  # shock names
dim_p   = 4  # lag-order

# estimate and identify under ordering "BLUE" of Fig.1 and 2 #
R.vars = vars::VAR(PCIT[ , names_k], p=dim_p, type="const")
R.idBL = id.iv(R.vars, iv=PCIT[-(1:dim_p), names_l], S2="MR", cov_u="OMEGA")
colnames(R.idBL$B) = names_s  # labeling

# estimate and identify under ordering "RED" of Fig.1 and 2 #
R.vars = vars::VAR(PCIT[ , names_k[c(2:1, 3:7)]], p=dim_p, type="const")
R.idRD = id.iv(R.vars, iv=PCIT[-(1:dim_p),names_l[2:1]], S2="MR", cov_u="OMEGA")
colnames(R.idRD$B) = names_s[2:1]  # labeling


# select minimal or full example #
is_min = TRUE
n.boot = ifelse(is_min, 5, 10000)

# bootstrap both under 1%-response normalization #
set.seed(2389)
R.norm = function(B) B / matrix(-diag(B), nrow(B), ncol(B), byrow=TRUE)
R.sbBL = sboot.mb(R.idBL, b.length=19, n.boot=n.boot, normf=R.norm)
R.sbRD = sboot.mb(R.idRD, b.length=19, n.boot=n.boot, normf=R.norm)

# plot IRF of Fig.1 and 2 with 68%-confidence levels #
library("ggplot2")
```

```
L.idx = list(BLUE1=c(1, 11, 5, 7, 3,  9)+0.1,
             RED1 =c(4, 12, 6, 8, 2, 10)+0.1,
             RED2 =c(1, 11, 7, 5, 3,  9)+0.1,
             BLUE2=c(4, 12, 8, 6, 2, 10)+0.1)
# Indexes to subset and reorder sub-plots in plot.sboot(), where
# the 14 IRF-subplots in the 2D-panel are numbered as a 1D-sequence
# vectorized by row. '+0.1' makes sub-setting robust against
# truncation errors from as.integer(). In a given figure, the plots
# RED and BLUE display the same selection of IRF-subplots.

R.fig1 = as.pplot(
 BLUE=plot(R.sbBL, lowerq=0.16, upperq=0.84, selection=list(1, L.idx[[1]])),
 RED =plot(R.sbRD, lowerq=0.16, upperq=0.84, selection=list(1, L.idx[[2]])),
 names_g=c("APITR first", "ACITR first"), color_g=c("blue", "red"), n.rows=3)

R.fig2 = as.pplot(
 RED =plot(R.sbRD, lowerq=0.16, upperq=0.84, selection=list(1, L.idx[[3]])),
 BLUE=plot(R.sbBL, lowerq=0.16, upperq=0.84, selection=list(1, L.idx[[4]])),
 names_g=c("ACITR first", "APITR first"), color_g=c("red", "blue"), n.rows=3)

R.fig1$F.plot + labs(x="Quarters", color="Ordering", fill="Ordering")
R.fig2$F.plot + labs(x="Quarters", color="Ordering", fill="Ordering")
```

`irf.pvarx` *Impulse Response Functions for panel SVAR models*

Description

Calculates impulse response functions for panel VAR objects.

Usage

```
## S3 method for class 'pvarx'
irf(x, ..., n.ahead = 20, normf = NULL, w = NULL, MG_IRF = TRUE)
```

Arguments

`x`	Panel VAR object of class 'pid' or 'pvarx' or a list of VAR objects that will be coerced to 'varx'.
`...`	Currently not used.
`n.ahead`	Integer. Number of periods to consider after the initial impulse, i.e. the horizon of the IRF.
`normf`	Function. A given function that normalizes the $K \times S$ input-matrix into an output matrix of same dimension. See the example in `id.iv` for the normalization of Jentsch and Lunsford (2021) that fixes the size of the impact response.

w	Numeric, logical, or character vector. N numeric elements weighting the individual coefficients, or names or N logical elements selecting a subset from the individuals $i = 1, \ldots, N$ for the MG estimation. If NULL (the default), all N individuals are included without weights.
MG_IRF	Logical. If TRUE (the default), the mean-group of individual IRF is calculated in accordance with Gambacorta et al. (2014). If FALSE, the IRF is calculated for the mean-group of individual VAR estimates.

Value

A list of class 'svarirf' holding the impulse response functions as a 'data.frame'.

References

Luetkepohl, H. (2005): *New Introduction to Multiple Time Series Analysis*, Springer, 2nd ed.

Gambacorta L., Hofmann B., and Peersman G. (2014): "The Effectiveness of Unconventional Monetary Policy at the Zero Lower Bound: A Cross-Country Analysis", *Journal of Money, Credit and Banking*, 46, pp. 615-642.

Jentsch, C., and Lunsford, K. G. (2021): "Asymptotically Valid Bootstrap Inference for Proxy SVARs", *Journal of Business and Economic Statistics*, 40, pp. 1876-1891.

Examples

```
data("PCAP")
names_k = c("g", "k", "l", "y")  # variable names
names_i = levels(PCAP$id_i)      # country names
L.data  = sapply(names_i, FUN=function(i)
  ts(PCAP[PCAP$id_i==i, names_k], start=1960, end=2019, frequency=1),
  simplify=FALSE)

# estimate and identify panel SVAR #
L.vars = lapply(L.data, FUN=function(x) vars::VAR(x, p=2, type="both"))
R.pid  = pid.chol(L.vars, order_k=names_k)

# calculate and plot MG-IRF #
library("ggplot2")
R.irf = irf(R.pid, n.ahead=60)
F.irf = plot(R.irf, selection=list(2:4, 1:2))
as.pplot(F.irf=F.irf, color_g="black", n.rows=3)$F.plot + guides(color="none")
```

---

irf.varx *Impulse Response Functions*

Description

Calculates impulse response functions.

Usage

```
## S3 method for class 'varx'
irf(x, ..., n.ahead = 20, normf = NULL)
```

Arguments

x	VAR object of class 'varx', 'id', or any other that will be coerced to 'varx'.
...	Currently not used.
n.ahead	Integer. Number of periods to consider after the initial impulse, i.e. the horizon of the IRF.
normf	Function. A given function that normalizes the $K \times S$ input-matrix into an output matrix of same dimension. See the example in id.iv for the normalization of Jentsch and Lunsford (2021) that fixes the size of the impact response.

Value

A list of class 'svarirf' holding the impulse response functions as a 'data.frame'.

References

Luetkepohl, H. (2005): *New Introduction to Multiple Time Series Analysis*, Springer, 2nd ed.

Jentsch, C., and Lunsford, K. G. (2021): "Asymptotically Valid Bootstrap Inference for Proxy SVARs", *Journal of Business and Economic Statistics*, 40, pp. 1876-1891.

Examples

```
data("PCIT")
names_k = c("APITR", "ACITR", "PITB", "CITB", "GOV", "RGDP", "DEBT")
names_l = c("m_PI", "m_CI")  # proxy names
names_s = paste0("epsilon[ ", c("PI", "CI"), " ]")  # shock names
dim_p   = 4  # lag-order

# estimate and identify proxy SVAR #
R.vars = vars::VAR(PCIT[ , names_k], p=dim_p, type="const")
R.idBL = id.iv(R.vars, iv=PCIT[-(1:dim_p), names_l], S2="MR", cov_u="OMEGA")
colnames(R.idBL$B) = names_s  # labeling

# calculate and plot normalized IRF #
R.norm = function(B) B / matrix(-diag(B), nrow(B), ncol(B), byrow=TRUE)
plot(irf(R.idBL, normf=R.norm))
```

MDEM *Data set for the* Monetary Demand Model

Description

The data set MDEM consists of annual observations for the nominal short-term interest rate R and the logarithm of the real money aggregate $m1$ and real GDP gdp. It covers the period 1957 to 1996 $(T = 40)$ for $N = 19$ countries.

Usage

```
data("MDEM")
```

Format

A long-format data panel of class 'data.frame', where the columns id_i and id_t indicate the country and year respectively.

Source

The prepared data is sourced from OECD and IMF's *International Financial Statistics* of the year 1998, see the open terms of use. Employed by Carrion-i-Silvestre and Surdeanu (2011:24, Ch.6.1), it has been originally compiled and described in the unpublished appendix of Mark and Sul (2003). See the related working paper of Mark and Sul (1999, Appendix B).

References

Carrion-i-Silvestre, J. L., and Surdeanu L. (2011): "Panel Cointegration Rank Testing with Cross-Section Dependence", *Studies in Nonlinear Dynamics & Econometrics*, 15 (4), pp. 1-43.

Mark, N. C., and Sul, D. (1999): "A Computationally Simple Cointegration Vector Estimator for Panel Data", Working Paper, Department of Economics, Ohio State University.

Mark, N. C., and Sul, D. (2003): "Cointegration Vector Estimation by Panel DOLS and Long-Run Money Demand," *Oxford Bulletin of Economics and Statistics*, 65, pp. 655-680.

See Also

Other data sets: ERPT, EURO, EU_w, ICAP, MERM, PCAP, PCIT

MERM *Data set for the* Monetary Exchange Rate Model

Description

The data set MERM consists of monthly observations for the log-ratios of prices p, money supply m, and industrial production y as well as the natural logarithm of nominal exchange rates against the dollar s. It covers the period January 1995 to December 2007 ($T = 156$) for $N = 19$ countries.

Usage

```
data("MERM")
```

Format

A long-format data panel of class 'data.frame', where the columns id_i and id_t indicate the country and month respectively.

Source

The prepared data set is directly obtainable from the journal website: doi:10.1016/j.ecosta.2016.10.001. Supplementary Raw Research Data. This is open data under the CC BY 4.0 license.

References

Oersal, D. D. K., and Arsova, A. (2017): "Meta-Analytic Panel Cointegrating Rank Tests for Dependent Panels", *Econometrics and Statistics*, 2, pp. 61-72.

See Also

Other data sets: ERPT, EURO, EU_w, ICAP, MDEM, PCAP, PCIT

PCAP *Data set on* Public Capital Stocks

Description

The data set PCAP consists of annual observations for

- the governmental capital stocks G and their logarithm g,
- the private capital stocks K and their logarithm k,
- the total hours worked L and their logarithm l, and
- the real GDP Y and its logarithm y.

It is constructed from the annual observations for

- the governmental investments IG,
- the private non-residential investments and capital stocks IB and B,
- the private housing investments and capital stocks IH and H, and
- the persons employed ET and hours worked per person HRS.

It covers the period 1960 to 2019 $(T = 60)$ for $N = 23$ OECD countries. All monetary values are given in US-Dollars at 2005 prices, i.e. constant PPP.

Usage

```
data("PCAP")
```

Format

A long-format data panel of class 'data.frame', where the columns id_i and id_t indicate the country and year respectively.

Source

Own compilation based on data from *PWT*, *Eurostat*, and OECD's *Economic Outlook*. Capital stocks are derived by the *Perpetual Inventory Method* as described by Kamps (2006). This is open data under the CC BY 4.0 license.

References

Empting, L. F. T., and Herwartz, H. (2025): "Revisiting the 'Productivity of Public Capital': VAR Evidence on the Heterogeneous Dynamics in a New Panel of 23 OECD Countries".

Feenstra, R. C., Inklaar, R., and Timmer, M. P. (2015): "The Next Generation of the Penn World Table", *American Economic Review*, 105, pp. 3150-3182.

Kamps, C. (2006): "New Estimates of Government Net Capital Stocks for 22 OECD Countries, 1960-2001", *IMF Staff Papers*, 53, pp. 120-150.

See Also

Other data sets: ERPT, EURO, EU_w, ICAP, MDEM, MERM, PCIT

Examples

```
### Latex figure: public capital stocks ###
data(PCAP)
names_i = c("DEU", "FRA", "GRC", "ITA", "NLD")  # select 5 exemplary countries
idx_i  = PCAP$id_i %in% names_i
idx_pl = c(1, 3, 5, 9, 10)
pallet = c(RColorBrewer::brewer.pal(n=11, name="Spectral")[idx_pl], "#000000")
names(pallet) = c(names_i, "\\O23")
breaks = factor(c(names_i, "\\O23"), levels=c(names_i, "\\O23"))
events = data.frame(
  label   = paste0("\\quad \\textbf{ ",
             c("Oil Crisis", "Oil Crisis II", "Early 1990s", "Early 2000s",
               "Great Recession", "European Debt Crisis", "COVID-19"), " }"),
```

```
  t_start = c(1973, 1979, 1990, 2000, 2007, 2010, 2020),
  t_end   = c(1975, 1982, 1993, 2003, 2010, 2013, 2022))

# plot events #
library("ggplot2")
F.base <- ggplot() +
  geom_hline(yintercept = 0, color="grey") +
  geom_rect(data=events, aes(xmin=t_start, xmax=t_end, ymin=-Inf, ymax=+Inf),
            fill="black", alpha=0.2) +
  geom_text(data=events, aes(x=(t_start+t_end)/2, y=-Inf, label=label),
            hjust=0, angle=90, colour='white') +
  scale_x_continuous(breaks = seq(1960, 2020, 10), limits = c(1960, 2022)) +
  theme_bw(base_size=10)

# add levels #
F.G <- F.base +
  geom_line(data=PCAP[idx_i, ], aes(x=id_t, y=G/1e+12, colour=id_i, group=id_i),
            linewidth=2) +
  stat_summary(data=PCAP[idx_i, ], aes(x=id_t, y=G/1e+12, color="\\O23"),
               fun=mean, geom="line", linewidth=0) +
  geom_text(data=events, aes(x=(t_start+t_end)/2, y=-Inf, label=label),
            hjust=0, angle=90, colour='white', alpha=0.6) +
  scale_colour_manual(values=pallet, breaks=breaks) +
  labs(x=NULL, y="Trillion US-\\$ at 2005 prices", colour="Country", title=NULL) +
  guides(colour=guide_legend(override.aes = list(linewidth=2))) +
  theme(legend.position="inside", legend.position.inside=c(0.01, 0.99),
        legend.justification = c(0, 1),
        legend.title=element_text(size=8), legend.text=element_text(size=6),
        legend.key.width = unit(0.35, "cm"), legend.key.height = unit(0.35, "cm"))

# add growth rates #
PCAP$gG = ave(PCAP$G, PCAP$id_i, FUN=function(x)
  c(diff(x), NA)/x*100)  # beginning-of-the-year observations!
F.gG <- F.base +
  geom_line(data=PCAP[idx_i, ], aes(x=id_t, y=gG, colour=id_i, group=id_i),
            linewidth=2) +
  stat_summary(data=PCAP, aes(x=id_t, y=gG, color="\\O23"),
               fun=mean, geom="line", linewidth=2) +
  geom_text(data=events, aes(x=(t_start+t_end)/2, y=-Inf, label=label),
            hjust=0, angle=90, colour='white', alpha=0.6) +
  scale_colour_manual(values=pallet, breaks=breaks, guide="none") +
  labs(x="Year", y="Growth in \\%", colour="Country", title=NULL)

# export to Latex #

library(tikzDevice)
textwidth = 15.5/2.54  # LaTeX textwidth from "cm" into "inch"
file_fig  = file.path(tempdir(), "Fig_G.tex")

tikz(file=file_fig, width=1.2*textwidth, height=0.8*textwidth)
 # gridExtra::grid.arrange(grobs=list(F.G, F.gG), layout_matrix=cbind(1:2))
 ggpubr::ggarrange(F.G, F.gG, ncol=1, nrow=2, align="v")
dev.off()
```

PCIT *Data set on* Personal and Corporate Income Tax

Description

The data set PCIT consists of quarterly observations for

- the average personal income tax rates $APITR$,
- the average corporate income tax rates $ACITR$,
- the logarithm of the personal income tax base $PITB$,
- the logarithm of the corporate income tax base $CITB$,
- the logarithm of government spending GOV,
- the logarithm of GDP divided by population $RGDP$, and
- the logarithm of government debt held by the public divided by the GDP deflator and population $DEBT$.

Moreover, the proxies for shocks to personal m_PI and corporate m_CI income taxes are prepended, where non-zero observations from the related narratively identified shock series T_PI resp. T_CI have been demeaned. The data set covers the period Q1 1950 to Q4 2006 ($T = 228$) for the US.

Usage

```
data("PCIT")
```

Format

A time series data set of class 'data.frame', where the column id_t indicates the quarter of the year.

Source

The prepared data set is directly obtainable from *openICPSR*: doi:10.3886/E116190V1. Supplementary Research Data. This is open data under the CC BY 4.0 license.

References

Mertens, K., and Ravn, M. O. (2013): "The Dynamic Effects of Personal and Corporate Income Tax Changes in the United States", *American Economic Review*, 103, pp. 1212-1247.

Jentsch, C., and Lunsford, K. G. (2019): "The Dynamic Effects of Personal and Corporate Income Tax Changes in the United States: Comment", *American Economic Review*, 109, pp. 2655-2678.

Mertens, K., and Ravn, M. O. (2019): "The Dynamic Effects of Personal and Corporate Income Tax Changes in the United States: Reply", *American Economic Review*, 109, pp. 2679-2691.

See Also

Other data sets: ERPT, EURO, EU_w, ICAP, MDEM, MERM, PCAP

---

pcoint *Panel cointegration rank tests*

Description

Performs test procedures for the rank of cointegration in a panel of VAR models. First, the chosen individual procedure is applied over all N individual entities for $r_{H0} = 0, \ldots, K-1$. Then, the $K \times N$ individual statistics and p-values are combined to panel test results on each r_{H0} using all combination approaches available for the chosen procedure.

Usage

```
pcoint.JO(
  L.data,
  lags,
  type = c("Case1", "Case2", "Case3", "Case4"),
  t_D1 = NULL,
  t_D2 = NULL,
  n.factors = FALSE
)

pcoint.BR(
  L.data,
  lags,
  type = c("Case1", "Case2", "Case3", "Case4"),
  t_D1 = NULL,
  t_D2 = NULL,
  n.iterations = FALSE
)

pcoint.SL(L.data, lags, type = "SL_trend", t_D = NULL, n.factors = FALSE)

pcoint.CAIN(L.data, lags, type = "SL_trend", t_D = NULL)
```

Arguments

L.data	List of 'data.frame' objects for each individual. The variables must have the same succession $k = 1, \ldots, K$ in each individual 'data.frame'.
lags	Integer or vector of integers. Lag-order of the VAR models in levels, which is either a common p for all individuals or individual-specific p_i for each individual. In the vector, p_i must have the same succession $i = 1, \ldots, N$ as argument L.data.
type	Character. The conventional case of the deterministic term.

t_D1	List of vectors. The activating break periods τ for the period-specific deterministic regressors in $d_{1,it}$, which are restricted to the cointegration relations. The accompanying lagged regressors are automatically included in $d_{2,it}$. The p-values are calculated for up to two breaks resp. three sub-samples.
t_D2	List of vectors. The activating break periods τ for the period-specific deterministic regressors in $d_{2,it}$, which are unrestricted.
n.factors	Integer. Number of common factors to be subtracted for the PANIC by Arsova and Oersal (2017, 2018). Deactivated if FALSE (the default).
n.iterations	Integer. The (maximum) number of iterations for the pooled estimation of the cointegrating vectors.
t_D	List of vectors. The activating break periods τ for the period-specific deterministic regressors in d_{it} of the SL-procedure. The accompanying lagged regressors are automatically included in d_{it}. The p-values are calculated for up to two breaks resp. three sub-samples.

Value

A list of class 'pcoint' with elements:

panel	List for the panel test results, which contains one matrix for the test statistics and one for the p-values.
individual	List for the individual test results, which contains one matrix for the test statistics and one for the p-values.
CSD	List of measures for cross-sectional dependency. NULL if a first generation test has been used.
args_pcoint	List of characters and integers indicating the panel cointegration test and specifications that have been used.
beta_H0	List of matrices, which comprise the pooled cointegrating vectors for each rank r_{H0}. NULL if any other test than BR has been used.

Functions

- pcoint.JO(): based on the Johansen procedure.
- pcoint.BR(): based on the pooled two-step estimation of the cointegrating vectors.
- pcoint.SL(): based on the Saikkonen-Luetkepohl procedure.
- pcoint.CAIN(): accounting for correlated probits between the individual SL-procedures.

References

Larsson, R., Lyhagen, J., and Lothgren, M. (2001): "Likelihood-based Cointegration Tests in Heterogeneous Panels", *Econometrics Journal*, 4, pp. 109-142.

Choi, I. (2001): "Unit Root Tests for Panel Data", *Journal of International Money and Finance*, 20, pp. 249-272.

Arsova, A., and Oersal, D. D. K. (2018): "Likelihood-based Panel Cointegration Test in the Presence of a Linear Time Trend and Cross-Sectional Dependence", *Econometric Reviews*, 37, pp. 1033-1050.

Breitung, J. (2005): "A Parametric Approach to the Estimation of Cointegration Vectors in Panel Data", *Econometric Reviews*, 24, pp. 151-173.

Oersal, D. D. K., and Droge, B. (2014): "Panel Cointegration Testing in the Presence of a Time Trend", *Computational Statistics & Data Analysis*, 76, pp. 377-390.

Oersal, D. D. K., and Arsova, A. (2017): "Meta-Analytic Cointegrating Rank Tests for Dependent Panels", *Econometrics and Statistics*, 2, pp. 61-72.

Arsova, A., and Oersal, D. D. K. (2018): "Likelihood-based Panel Cointegration Test in the Presence of a Linear Time Trend and Cross-Sectional Dependence", *Econometric Reviews*, 37, pp. 1033-1050.

Hartung, J. (1999): "A Note on Combining Dependent Tests of Significance", *Biometrical Journal*, 41, pp. 849-855.

Arsova, A., and Oersal, D. D. K. (2021): "A Panel Cointegrating Rank Test with Structural Breaks and Cross-Sectional Dependence", *Econometrics and Statistics*, 17, pp. 107-129.

Examples

```
### reproduce Oersal,Arsova 2017:67, Ch.5 ###
data("MERM")
names_k = colnames(MERM)[-(1:2)] # variable names
names_i = levels(MERM$id_i)      # country names
L.data  = sapply(names_i, FUN=function(i)
   ts(MERM[MERM$id_i==i, names_k], start=c(1995, 1), frequency=12),
   simplify=FALSE)

# Oersal,Arsova 2017:67, Tab.5 #
R.lags = c(2, 2, 2, 2, 1, 2, 2, 4, 2, 3, 2, 2, 2, 2, 2, 1, 1, 2, 2)
names(R.lags) = names_i  # individual lags by AIC (lag_max=4)
n.factors = 8  # number of common factors by Onatski's (2010) criterion
R.pcsl = pcoint.SL(L.data, lags=R.lags, n.factors=n.factors, type="SL_trend")
R.pcjo = pcoint.JO(L.data, lags=R.lags, n.factors=n.factors, type="Case4")

# Oersal,Arsova 2017:67, Tab.6 #
R.Ftsl = coint.SL(y=R.pcsl$CSD$Ft, dim_p=2, type_SL="SL_trend")  # lag-order by AIC
R.Ftjo = coint.JO(y=R.pcsl$CSD$Ft, dim_p=2, type="Case4")

### reproduce Oersal,Arsova 2016:13, Ch.6 ###
data("ERPT")
names_k = c("lpm5", "lfp5", "llcusd") # variable names for "Chemicals and related products"
names_i = levels(ERPT$id_i)[c(1,6,2,5,4,3,7)]  # ordered country names
L.data  = sapply(names_i, FUN=function(i)
   ts(ERPT[ERPT$id_i==i, names_k], start=c(1995, 1), frequency=12),
   simplify=FALSE)

# Oersal,Arsova 2016:21, Tab.6 (only for individual results) #
R.lags = c(3, 3, 3, 4, 3, 3, 3); names(R.lags)=names_i  # lags of VAR model by MAIC
R.cain = pcoint.CAIN(L.data, lags=R.lags, type="SL_trend")
R.pcsl = pcoint.SL(L.data,   lags=R.lags, type="SL_trend")

# Oersal,Arsova 2016:22, Tab.7/8 #
R.lags = c(3, 3, 3, 4, 4, 3, 4); names(R.lags)=names_i  # lags of VAR model by MAIC
```

```
R.t_D  = list(t_break=89)  # a level shift and trend break in 2002_May for all countries
R.cain = pcoint.CAIN(L.data, lags=R.lags, t_D=R.t_D, type="SL_trend")
```

`pid.chol` *Recursive identification of panel SVAR models via Cholesky decomposition*

Description

Given an estimated panel of VAR models, this function uses the Cholesky decomposition to identify the structural impact matrix B_i of the corresponding SVAR model

$$y_{it} = c_{it} + A_{i1} y_{i,t-1} + ... + A_{i,p_i} y_{i,t-p_i} + u_{it}$$

$$= c_{it} + A_{i1} y_{i,t-1} + ... + A_{i,p_i} y_{i,t-p_i} + B_i \epsilon_{it}.$$

Matrix B_i corresponds to the decomposition of the least squares covariance matrix $\Sigma_{u,i} = B_i B_i'$.

Usage

```
pid.chol(x, order_k = NULL)
```

Arguments

`x`	An object of class '`pvarx`' or a list of VAR objects that will be coerced to '`varx`'. Estimated panel of VAR objects.
`order_k`	Vector. Vector of characters or integers specifying the assumed structure of the recursive causality. Change the causal ordering in the instantaneous effects without permuting variables and re-estimating the VAR model.

Value

List of class '`pid`' with elements:

`A`	Matrix. The lined-up coefficient matrices $A_j, j = 1, \ldots, p$ for the lagged variables in the panel VAR.
`B`	Matrix. Mean group of the estimated structural impact matrices B_i, i.e. the unique decomposition of the covariance matrices of reduced-form errors.
`L.varx`	List of '`varx`' objects for the individual estimation results to which the structural impact matrices B_i have been added.
`args_pid`	List of characters and integers indicating the identification methods and specifications that have been used.
`args_pvarx`	List of characters and integers indicating the estimator and specifications that have been used.

References

Luetkepohl, H. (2005): *New Introduction to Multiple Time Series Analysis*, Springer, 2nd ed.

Sims, C. A. (2008): "Macroeconomics and Reality", *Econometrica*, 48, pp. 1-48.

See Also

Other panel identification functions: `pid.cvm()`, `pid.dc()`, `pid.grt()`, `pid.iv()`

Examples

```
data("PCAP")
names_k = c("g", "k", "l", "y")  # variable names
names_i = levels(PCAP$id_i)      # country names
L.data  = sapply(names_i, FUN=function(i)
  ts(PCAP[PCAP$id_i==i, names_k], start=1960, end=2019, frequency=1),
  simplify=FALSE)

# estimate and identify panel SVAR #
L.vars = lapply(L.data, FUN=function(x) vars::VAR(x, p=2, type="both"))
R.pid  = pid.chol(L.vars, order_k=names_k)
```

`pid.cvm` *Independence-based identification of panel SVAR models via Cramer-von Mises (CVM) distance*

Description

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

$$y_{it} = c_{it} + A_{i1} y_{i,t-1} + ... + A_{i,p_i} y_{i,t-p_i} + u_{it}$$

$$= c_{it} + A_{i1} y_{i,t-1} + ... + A_{i,p_i} y_{i,t-p_i} + B_i \epsilon_{it}.$$

Matrix B_i corresponds to the unique decomposition of the least squares covariance matrix $\Sigma_{u,i} = B_i B_i'$ if the vector of structural shocks ϵ_{it} 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

```
pid.cvm(
  x,
  combine = c("group", "pool", "indiv"),
  n.factors = NULL,
  dd = NULL,
```

```
  itermax = 500,
  steptol = 100,
  iter2 = 75
)
```

Arguments

x	An object of class 'pvarx' or a list of VAR objects that will be coerced to 'varx'. Estimated panel of VAR objects.
combine	Character. The combination of the individual reduced-form residuals via 'group' for the group ICA by Calhoun et al. (2001) using common structural shocks, via 'pool' for the pooled shocks by Herwartz and Wang (2024) using a common rotation matrix, or via 'indiv' for individual-specific $B_i \forall i$ using strictly separated identification runs.
n.factors	Integer. Number of common structural shocks across all individuals if the group ICA is selected.
dd	Object of class 'indepTestDist' generated by 'indepTest' from package 'copula'. Simulated independent sample(s) of the same size as the data. If the sample sizes $T_i - p_i$ differ across 'i', the strictly separated identification requires a list of N individual 'indepTestDist'-objects with respective sample sizes. If NULL (the default), a suitable object will be calculated during the call of pid.cvm.
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

List of class 'pid' with elements:

A	Matrix. The lined-up coefficient matrices $A_j, j = 1, \ldots, p$ for the lagged variables in the panel VAR.
B	Matrix. Mean group of the estimated structural impact matrices B_i, i.e. the unique decomposition of the covariance matrices of reduced-form errors.
L.varx	List of 'varx' objects for the individual estimation results to which the structural impact matrices B_i have been added.
eps0	Matrix. The combined whitened residuals ϵ_0 to which the ICA procedure has been applied subsequently. These are still the unrotated baseline shocks! NULL if 'indiv' identifications have been used.
ICA	List of objects resulting from the underlying ICA procedure. NULL if 'indiv' identifications have been used.
rotation_angles	Numeric vector. The rotation angles suggested by the combined identification procedure. NULL if 'indiv' identifications have been used.
args_pid	List of characters and integers indicating the identification methods and specifications that have been used.
args_pvarx	List of characters and integers indicating the estimator and specifications that have been used.

References

Comon, P. (1994): "Independent Component Analysis: A new Concept?", *Signal Processing*, 36, pp. 287-314.

Genest, C., and Remillard, B. (2004): "Tests of Independence and Randomness based on the Empirical Copula Process", Test, 13, pp. 335-370.

Herwartz, H., and Wang, S. (2024): "Statistical Identification in Panel Structural Vector Autoregressive Models based on Independence Criteria", *Journal of Applied Econometrics*, 39 (4), pp. 620-639.

Herwartz, H. (2018): "Hodges Lehmann detection of structural shocks - An Analysis of macroeconomic Dynamics in the Euro Area", *Oxford Bulletin of Economics and Statistics*, 80, pp. 736-754.

Herwartz, H., and Ploedt, M. (2016): "The Macroeconomic Effects of Oil Price Shocks: Evidence from a Statistical Identification Approach", *Journal of International Money and Finance*, 61, pp. 30-44.

See Also

. . . the individual `id.cvm` by Lange et al. (2021) in **svars**. Note that `pid.cvm` relies on a modification of their procedure and thus performs ICA on the pre-whitened shocks 'eps0' directly.

Other panel identification functions: `pid.chol()`, `pid.dc()`, `pid.grt()`, `pid.iv()`

Examples

```
# select minimal or full example #
is_min = TRUE
idx_i  = ifelse(is_min, 1, 1:14)

# load and prepare data #
data("EURO")
data("EU_w")
names_i = names(EURO[idx_i+1])  # country names (#1 is EA-wide aggregated data)
idx_k   = 1:4   # endogenous variables in individual data matrices
idx_t   = 1:76  # periods from 2001Q1 to 2019Q4
trend2  = idx_t^2

# individual VARX models with common lag-order p=2 #
L.data = lapply(EURO[idx_i+1], FUN=function(x) x[idx_t, idx_k])
L.exog = lapply(EURO[idx_i+1], FUN=function(x) cbind(trend2, x[idx_t, 5:10]))
L.vars = sapply(names_i, FUN=function(i)
  vars::VAR(L.data[[i]], p=2, type="both", exogen=L.exog[[i]]),
  simplify=FALSE)

# identify under common orthogonal matrix (with pooled sample size (T-p)*N) #
S.pind = copula::indepTestSim(n=(76-2)*length(names_i), p=length(idx_k), N=100)
R.pcvm = pid.cvm(L.vars, dd=S.pind, combine="pool")
R.irf  = irf(R.pcvm, n.ahead=50, w=EU_w)
plot(R.irf, selection=list(1:2, 3:4))

# identify individually (with same sample size T-p for all 'i') #
S.pind = copula::indepTestSim(n=(76-2), p=length(idx_k), N=100)
```

```
R.pcvm = pid.cvm(L.vars, dd=S.pind, combine="indiv")
R.irf  = irf(R.pcvm, n.ahead=50, w=EU_w)
plot(R.irf, selection=list(1:2, 3:4))
```

`pid.dc` *Independence-based identification of panel SVAR models using distance covariance (DC) statistic*

Description

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

$$y_{it} = c_{it} + A_{i1}y_{i,t-1} + ... + A_{i,p_i}y_{i,t-p_i} + u_{it}$$
$$= c_{it} + A_{i1}y_{i,t-1} + ... + A_{i,p_i}y_{i,t-p_i} + B_i\epsilon_{it}.$$

Matrix B_i corresponds to the unique decomposition of the least squares covariance matrix $\Sigma_{u,i} = B_iB_i'$ if the vector of structural shocks ϵ_{it} 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_i.

Usage

```
pid.dc(
  x,
  combine = c("group", "pool", "indiv"),
  n.factors = NULL,
  n.iterations = 100,
  PIT = FALSE
)
```

Arguments

`x`	An object of class 'pvarx' or a list of VAR objects that will be coerced to 'varx'. Estimated panel of VAR objects.
`combine`	Character. The combination of the individual reduced-form residuals via 'group' for the group ICA by Calhoun et al. (2001) using common structural shocks, via 'pool' for the pooled shocks by Herwartz and Wang (2024) using a common rotation matrix, or via 'indiv' for individual-specific $B_i \forall i$ using strictly separated identification runs.
`n.factors`	Integer. Number of common structural shocks across all individuals if the group ICA is selected.
`n.iterations`	Integer. The maximum number of iterations in the 'steadyICA' algorithm. The default in 'steadyICA' is 100.
`PIT`	Logical. If PIT='TRUE', the distribution and density of the independent components are estimated using Gaussian kernel density estimates.

Value

List of class 'pid' with elements:

A	Matrix. The lined-up coefficient matrices $A_j, j = 1, \ldots, p$ for the lagged variables in the panel VAR.
B	Matrix. Mean group of the estimated structural impact matrices B_i, i.e. the unique decomposition of the covariance matrices of reduced-form errors.
L.varx	List of 'varx' objects for the individual estimation results to which the structural impact matrices B_i have been added.
eps0	Matrix. The combined whitened residuals ϵ_0 to which the ICA procedure has been applied subsequently. These are still the unrotated baseline shocks! NULL if 'indiv' identifications have been used.
ICA	List of objects resulting from the underlying ICA procedure. NULL if 'indiv' identifications have been used.
rotation_angles	Numeric vector. The rotation angles suggested by the combined identification procedure. NULL if 'indiv' identifications have been used.
args_pid	List of characters and integers indicating the identification methods and specifications that have been used.
args_pvarx	List of characters and integers indicating the estimator and specifications that have been used.

Notes on the Reproduction in "Examples"

The reproduction of Herwartz and Wang (HW, 2024:630) serves as an exemplary application and unit-test of the implementation by **pvars**. While **vars**' VAR employs equation-wise lm with the QR-decomposition of the regressor matrix X, HW2024 and accordingly the reproduction by pvarx.VAR both calculate $X'(XX')^{-1}$ for the multivariate least-squares estimation of A_i. Moreover, both use steadyICA for identification such that the reproduction result for the pooled rotation matrix Q is close to HW2024, the mean absolute difference between both 4x4 matrices is less than 0.0032. Note that the single EA-Model is estimated and identified the same way, which can be extracted as a separate 'varx' object from the trivial panel object by $L.varx[[1]] and even bootstrapped by sboot.mb.

Some differences remain such that the example does not exactly reproduce the results in HW2024. To account for the n exogenous and deterministic regressors in slot $D, pvarx.VAR calculates $\Sigma_{u,i}$ with the degrees of freedom $T - Kp_i - n$ instead of HW2024's $T - Kp_i - 1$. Moreover, the confidence bands for the IRF are based on **pvars**' panel moving-block- instead of HW2024's wild bootstrap. The responses of real GDP and of inflation are not scaled by 0.01, unlike in HW2024. Note that both bootstrap procedures keep D fixed over their iterations.

References

Calhoun, V. D., Adali, T., Pearlson, G. D., and Pekar, J. J. (2001): "A Method for Making Group Inferences from Functional MRI Data using Independent Component Analysis", *Human Brain Mapping*, 16, pp. 673-690.

Comon, P. (1994): "Independent Component Analysis: A new Concept?", *Signal Processing*, 36, pp. 287-314.

Herwartz, H., and Wang, S. (2024): "Statistical Identification in Panel Structural Vector Autoregressive Models based on Independence Criteria", *Journal of Applied Econometrics*, 39 (4), pp. 620-639.

Matteson, D. S., and Tsay, R. S. (2017): "Independent Component Analysis via Distance Covariance", *Journal of the American Statistical Association*, 112, pp. 623-637.

Risk, B., Matteson, D. S., Ruppert, D., Eloyan, A., and Caffo, B. S. (2014): "An Evaluation of Independent Component Analyses with an Application to Resting-State fMRI", *Biometrics*, 70, pp. 224-236.

Szekely, G. J., Rizzo, M. L., and Bakirov, N. K. (2007): "Measuring and Testing Dependence by Correlation of Distances", *Annals of Statistics*, 35, pp. 2769-2794.

See Also

Other panel identification functions: `pid.chol()`, `pid.cvm()`, `pid.grt()`, `pid.iv()`

Examples

```
### replicate Herwartz,Wang 2024:630, Ch.4 ###

# select minimal or full example #
is_min = TRUE
n.boot = ifelse(is_min, 5, 1000)
idx_i  = ifelse(is_min, 1, 1:14)

# load and prepare data #
data("EURO")
names_i = names(EURO[idx_i+1])  # country names (#1 is EA-wide aggregated data)
names_s = paste0("epsilon[ ", c(1:2, "m", "f"), " ]")  # shock names
idx_k   = 1:4   # endogenous variables in individual data matrices
idx_t   = 1:(nrow(EURO[[1]])-1)  # periods from 2001Q1 to 2019Q4
trend2  = idx_t^2

# panel SVARX model, Ch.4.1 #
L.data = lapply(EURO[idx_i+1], FUN=function(x) x[idx_t, idx_k])
L.exog = lapply(EURO[idx_i+1], FUN=function(x) cbind(trend2, x[idx_t, 5:10]))
R.lags = c(1,2,1,2,2,2,2,2,1,2,2,2,2,1)[idx_i]; names(R.lags) = names_i
R.pvar = pvarx.VAR(L.data, lags=R.lags, type="both", D=L.exog)
R.pid  = pid.dc(R.pvar, combine="pool")
print(R.pid)  # suggests e3 and e4 to be MP and financial shocks, respectively.
colnames(R.pid$B) = names_s  # accordant labeling

# EA-wide SVARX model, Ch.4.2 #
R.data = EURO[[1]][idx_t, idx_k]
R.exog = cbind(trend2, EURO[[1]][idx_t, 5:6])
R.varx = pvarx.VAR(list(EA=R.data), lags=2, type="both", D=list(EA=R.exog))
R.id   = pid.dc(R.varx, combine="indiv")$L.varx$EA
colnames(R.id$B) = names_s  # labeling

# comparison of IRF without confidence bands, Ch.4.3.1 #
data("EU_w")  # GDP weights with the same ordering names_i as L.varx in R.pid
R.norm = function(B) B / matrix(diag(B), nrow(B), ncol(B), byrow=TRUE) * 25
```

```
R.irf  = as.pplot(
  EA=plot(irf(R.id,  normf=R.norm), selection=list(idx_k, 3:4)),
  MG=plot(irf(R.pid, normf=R.norm, w=EU_w), selection=list(idx_k, 3:4)),
  color_g=c("#3B4992FF", "#008B45FF"), shape_g=16:17, n.rows=length(idx_k))
plot(R.irf)

# comparison of IRF with confidence bands, Ch.4.3.1 #
R.boot_EA = sboot.mb(R.id, b.length=8, n.boot=n.boot, n.cores=2, normf=R.norm)
R.boot_MG = sboot.pmb(R.pid, b.dim=c(8, FALSE), n.boot=n.boot, n.cores=2,
                      normf=R.norm, w=EU_w)
R.irf = as.pplot(
  EA=plot(R.boot_EA, lowerq=0.16, upperq=0.84, selection=list(idx_k, 3:4)),
  MG=plot(R.boot_MG, lowerq=0.16, upperq=0.84, selection=list(idx_k, 3:4)),
  color_g=c("#3B4992FF", "#008B45FF"), shape_g=16:17, n.rows=length(idx_k))
plot(R.irf)
```

`pid.grt` *Identification of panel SVEC models by imposing long- and short-run restrictions*

Description

Identifies a panel of SVEC models by utilizing a scoring algorithm to impose long- and short-run restrictions. See the details of `SVEC` in **vars**.

Usage

```
pid.grt(
  x,
  LR = NULL,
  SR = NULL,
  start = NULL,
  max.iter = 100,
  conv.crit = 1e-07,
  maxls = 1
)
```

Arguments

`x`	An object of class 'pvarx' or a list of VECM objects that will be coerced to 'varx'. Panel of VAR objects estimated under rank-restriction.
`LR`	Matrix. The restricted long-run impact matrix.
`SR`	Matrix. The restricted contemporaneous impact matrix.
`start`	Vector. The starting values for γ, set by `rnorm` if `NULL` (the default).
`max.iter`	Integer. The maximum number of iterations.

conv.crit	Real number. Convergence value of algorithm.
maxls	Real number. Maximum movement of the parameters between two iterations of the scoring algorithm.

Value

List of class 'pid' with elements:

A	Matrix. The lined-up coefficient matrices $A_j, j = 1, \ldots, p$ or the lagged variables in the panel VAR.
B	Matrix. Mean group of the estimated structural impact matrices B_i, i.e. the unique decomposition of the covariance matrices of reduced-form errors.
L.varx	List of 'varx' objects for the individual estimation results to which the structural impact matrices B_i have been added.
args_pid	List of characters and integers indicating the identification methods and specifications that have been used.
args_pvarx	List of characters and integers indicating the estimator and specifications that have been used.

References

Amisano, G. and Giannini, C. (1997): *Topics in Structural VAR Econometrics*, Springer, 2nd ed.

Breitung, J., Brueggemann R., and Luetkepohl, H. (2004): "Structural Vector Autoregressive Modeling and Impulse Responses", in *Applied Time Series Econometrics*, ed. by H. Luetkepohl and M. Kraetzig, Cambridge University Press, Cambridge.

Johansen, S. (1996): *Likelihood-Based Inference in Cointegrated Vector Autoregressive Models*, Advanced Texts in Econometrics, Oxford University Press, USA.

Luetkepohl, H. (2005): *New Introduction to Multiple Time Series Analysis*, Springer, 2nd ed.

Pfaff, B. (2008): "VAR, SVAR and SVEC Models: Implementation within R Package **vars**", *Journal of Statistical Software*, 27, pp. 1-32.

See Also

... the original SVEC by Pfaff (2008) in **vars**. Note that pid.grt relies on this underlying procedure, but allows for the additional model specifications in pvarx.VEC and for the bootstrap procedures in sboot.pmb, both provided by the **pvars** package.

Other panel identification functions: pid.chol(), pid.cvm(), pid.dc(), pid.iv()

Examples

```
data("PCAP")
names_k = c("g", "k", "l", "y")  # variable names
names_i = levels(PCAP$id_i)      # country names
names_s = NULL                   # optional shock names
L.data  = sapply(names_i, FUN=function(i)
  ts(PCAP[PCAP$id_i==i, names_k], start=1960, end=2019, frequency=1),
  simplify=FALSE)
```

```
# colnames of the restriction matrices are passed as shock names #
SR = matrix(NA, nrow=4, ncol=4, dimnames=list(names_k, names_s))
SR[1, 2] = 0
SR[3, 4] = 0
LR = matrix(NA, nrow=4, ncol=4, dimnames=list(names_k, names_s))
LR[ , 3:4] = 0

# estimate and identify panel SVECM #
R.pvec = pvarx.VEC(L.data, lags=2, dim_r=2, type="Case4")
R.pid  = pid.grt(R.pvec, LR=LR, SR=SR)
```

`pid.iv` *Identification of panel SVAR models by means of proxy variables*

Description

Given an estimated panel of VAR models, this function uses proxy variables to partially identify the structural impact matrix B_i of the corresponding SVAR model

$$y_{it} = c_{it} + A_{i1}y_{i,t-1} + ... + A_{i,p_i}y_{i,t-p_i} + u_{it}$$

$$= c_{it} + A_{i1}y_{i,t-1} + ... + A_{i,p_i}y_{i,t-p_i} + B_i\epsilon_{it}.$$

In general, identification procedures determine B_i up to column ordering, scale, and sign. For a unique solution, `pid.iv` follows the literature on proxy SVAR. The S columns in $B_i = [B_{i,1} : B_{i,2}]$ of the identified shocks $\epsilon_{its}, s = 1, \ldots, S$, are ordered first, and the variance $\sigma^2_{\epsilon,is} = 1$ is normalized to unity (see e.g. Lunsford 2015:6, Eq. 9). Further, the sign is fixed to a positive correlation between proxy and shock series. A normalization of the impulsed shock that may fix the size of the impact response in the IRF can be imposed subsequently via '`normf`' in `irf.pvarx` and `sboot.pmb`.

Usage

```
pid.iv(
  x,
  iv,
  S2 = c("MR", "JL", "NQ"),
  cov_u = "OMEGA",
  R0 = NULL,
  combine = c("pool", "indiv")
)
```

Arguments

`x`	An object of class '`pvarx`' or a list of VAR objects that will be coerced to '`varx`'. Estimated panel of VAR objects.
`iv`	List. A single '`data.frame`' of the L common proxy time series m_t or a list of N '`data.frame`' objects of the L individual proxy time series m_{it}. The proxies must have the same succession $l = 1, \ldots, L$ in each individual '`data.frame`'.

S2 — Character. Identification within multiple proxies m_{it} via 'MR' for lower-triangular $[I_S : -B_{i,11}B_{i,12}^{-1}]B_{i,1}$ by Mertens, Ravn (2013), via 'JL' for chol$(\Sigma_{mu,i}\Sigma_{u,i}^{-1}\Sigma_{um,i})$ by Jentsch, Lunsford (2021), or via 'NQ' for the nearest orthogonal matrix from svd decomposition by Empting et al. (2025). In case of $S = L = 1$ proxy, all three choices provide identical results on $B_{i,1}$. In case of combine='pool', the argument is automatically fixed to 'NQ'.

cov_u — Character. Selection of the estimated residual covariance matrices $\hat{\Sigma}_{u,i}$ to be used in the identification procedure. Either 'OMEGA' (the default) for $\hat{U}_i\hat{U}_i'/T_i$ as used in Mertens, Ravn (2013) and Jentsch, Lunsford (2021) or 'SIGMA' for $\hat{U}_i\hat{U}_i'/(T_i - n_{zi})$, which corrects for the number of regressors n_{zi}. Both character options refer to the name of the respective estimate in the 'varx' objects.

R0 — Matrix. A $(L \times S)$ selection matrix for 'NQ' that governs the attribution of the L proxies to their specific S structural shock series. If NULL (the default), R0 $= I_S$ will be used such that the $S = L$ columns of $B_{i,1}$ are one-by-one estimated from the $S = L$ proxy series 'iv' available.

combine — Character. The combination of the individual reduced-form residuals via 'pool' for the pooled shocks by Empting et al. (2025) using a common orthogonal matrix or via 'indiv' for individual-specific $B_i \forall i$ using strictly separated identification runs.

Value

List of class 'pid' with elements:

A — Matrix. The lined-up coefficient matrices $A_j, j = 1, \ldots, p$ for the lagged variables in the panel VAR.

B — Matrix. Mean group of the estimated structural impact matrices B_i, i.e. the unique decomposition of the covariance matrices of reduced-form errors.

L.varx — List of 'varx' objects for the individual estimation results to which the structural impact matrices B_i have been added.

eps0 — Matrix. The combined whitened residuals ϵ_0 to which the pooled identification procedure has been applied subsequently. These are still the unrotated baseline shocks! NULL if 'indiv' identifications have been used.

Q — Matrix. The orthogonal matrix suggested by the pooled identification procedure. NULL if 'indiv' identifications have been used.

args_pid — List of characters and integers indicating the identification methods and specifications that have been used.

args_pvarx — List of characters and integers indicating the estimator and specifications that have been used.

References

Mertens, K., and Ravn, M. O. (2013): "The Dynamic Effects of Personal and Corporate Income Tax Changes in the United States", *American Economic Review*, 103, pp. 1212-1247.

Jentsch, C., and Lunsford, K. G. (2019): "The Dynamic Effects of Personal and Corporate Income Tax Changes in the United States: Comment", *American Economic Review*, 109, pp. 2655-2678.

Jentsch, C., and Lunsford, K. G. (2021): "Asymptotically Valid Bootstrap Inference for Proxy SVARs", *Journal of Business and Economic Statistics*, 40, pp. 1876-1891.

Empting, L. F. T., Maxand, S., Oeztuerk, S., and Wagner, K. (2025): "Inference in Panel SVARs with Cross-Sectional Dependence of Unknown Form".

See Also

Other panel identification functions: `pid.chol()`, `pid.cvm()`, `pid.dc()`, `pid.grt()`

Examples

```
data("PCIT")
names_k = c("APITR", "ACITR", "PITB", "CITB", "GOV", "RGDP", "DEBT")
names_l = c("m_PI", "m_CI")  # proxy names
names_s = paste0("epsilon[ ", c("PI", "CI"), " ]")  # shock names
dim_p   = 4  # lag-order

# estimate and identify panel SVAR #
L.vars = list(USA = vars::VAR(PCIT[ , names_k], p=dim_p, type="const"))
L.iv   = list(USA = PCIT[-(1:dim_p), names_l])
R.pid  = pid.iv(L.vars, iv=L.iv, S2="NQ", cov_u="SIGMA", combine="pool")
colnames(R.pid$B)[1:2] = names_s  # labeling
```

PP *Persistence Profiles*

Description

Calculates persistence profiles for each of the r long-run relationships.

Usage

```
PP.system(x, n.ahead = 20)

PP.variable(x, n.ahead = 20, shock = NULL)
```

Arguments

x	Rank-restricted VAR object of class 'varx' or any other that can be coerced to 'varx', e.g. 'vec2var'. If the object is also of child class 'id', `PP.variable` calculates the persistence profiles which are initiated by the provided structural shocks.
n.ahead	Integer. Number of periods to consider after the initial impulse, i.e. the horizon of the PP.
shock	Matrix. Each column vector specifies a set of simultaneous shocks, which initiate r persistence profiles. If NULL (the default), a separate unit impulse is set for each shock.

Value

A list of class 'svarirf' holding the persistence profiles as a 'data.frame'.

Functions

- PP.system(): PP due to a system-wide shock
- PP.variable(): PP due to a structural or variable-specific shock

References

Lee, K., C., Pesaran, M. H. (1993): "Persistence Profiles and Business Cycle Fluctuations in a Disaggregated Model of UK Output Growth", *Richerche Economiche*, 47, pp. 293-322.

Pesaran, M. H., and Shin, Y. (1996): "Cointegration and Speed of Convergence to Equilibrium", *Journal of Econometrics*, 71, pp. 117-143.

Examples

```
data("PCAP")
names_k = c("g", "k", "l", "y") # variable names
names_i = levels(PCAP$id_i)     # country names
L.data  = sapply(names_i, FUN=function(i)
  ts(PCAP[PCAP$id_i==i, names_k], start=1960, end=2019, frequency=1),
  simplify=FALSE)

# estimate VAR for DNK under rank-restriction r=2 #
dim_r  = 2  # cointegrataion rank
R.t_D1 = list(t_break=c(23, 49))  # trend breaks
R.vecm = VECM(y=L.data$DNK, dim_p=2, dim_r=dim_r, type="Case4", t_D1=R.t_D1)

# define shocks #
shock1 = diag(4)   # 4 separate shocks
shock2 = cbind(c(1, 0,  0, 0),  # positive shock on "g"
               c(0, 0, -1, 0),  # negative shock on "l"
               c(0, 0,  1, 1))  # simultaneous shocks

# calculate persistence profiles #
R.ppv1 = PP.variable(R.vecm, n.ahead=50, shock=shock1)
R.ppv2 = PP.variable(R.vecm, n.ahead=50, shock=shock2)
R.ppsy = PP.system(R.vecm, n.ahead=50)

# edit plots #
library("ggplot2")
as.pplot(ppv1=plot(R.ppv1), n.rows=4)$F.plot + guides(color="none")
as.pplot(ppv2=plot(R.ppv2), n.rows=3, color_g="black") # reshape facet array
plot(R.ppsy, selection=list(1, c(1,4)))  # dismiss cross-term PP
```

pvars *pvars: VAR Modeling for Heterogeneous Panels*

Description

This package implements (1) panel cointegration rank tests, (2) estimators for panel vector autoregressive (VAR) models, and (3) identification methods for panel structural vector autoregressive (SVAR) models as described in the accompanying vignette. The implemented functions allow to account for cross-sectional dependence and for structural breaks in the deterministic terms of the VAR processes.

Details

(1) The panel functions to determine the cointegration rank are:

- `pcoint.JO` panel Johansen procedures,
- `pcoint.BR` panel test with pooled two-step estimation,
- `pcoint.SL` panel Saikkonen-Luetkepohl procedures,
- `pcoint.CAIN` correlation-augmented inverse normal test.

(2) The panel functions to estimate the VAR models are:

- `pvarx.VAR` mean-group of a panel of VAR models,
- `pvarx.VEC` pooled cointegrating vectors in a panel VECM.

(3) The panel functions to retrieve structural impact matrices are:

- `pid.chol` identification of panel SVAR models using Cholesky decomposition to impose recursive causality,
- `pid.grt` identification of panel SVEC models by imposing long- and short-run restrictions,
- `pid.iv` identification of panel SVAR models by means of proxy variables,
- `pid.dc` independence-based identification of panel SVAR models using distance covariance (DC) statistic,
- `pid.cvm` independence-based identification of panel SVAR models using Cramer-von Mises (CVM) distance.

Supporting tools, such as the specification functions (`speci.VAR`, `speci.factors`) and the panel block bootstrap procedure (`sboot.pmb`), complement the panel VAR functions and complete this coherent approach to ***VAR modeling for heterogeneous panels*** within the **vars** ecosystem. The provided data sets further allow for the exact replication of the implemented literature.

Author(s)

Lennart Empting <lennart.empting@vwl.uni-due.de> (ORCID: 0009-0004-5068-4639)

See Also

Useful links:

- https://github.com/Lenni89/pvars
- Report bugs at https://github.com/Lenni89/pvars/issues

---

pvarx *Estimation of VAR models for heterogeneous panels*

Description

Performs the (pooled) mean-group estimation of a panel VAR model. First, VAR models are estimated for all N individual entities. Then, their (pooled) mean-group estimate is calculated for each coefficient.

Usage

```
pvarx.VAR(
  L.data,
  lags,
  type = c("const", "trend", "both", "none"),
  t_D = NULL,
  D = NULL,
  n.factors = FALSE,
  n.iterations = FALSE
)

pvarx.VEC(
  L.data,
  lags,
  dim_r,
  type = c("Case1", "Case2", "Case3", "Case4", "Case5"),
  t_D1 = NULL,
  t_D2 = NULL,
  D1 = NULL,
  D2 = NULL,
  idx_pool = NULL,
  n.factors = FALSE,
  n.iterations = FALSE
)
```

Arguments

L.data	List of 'data.frame' objects for each individual. The variables must have the same succession $k = 1, \ldots, K$ in each individual 'data.frame'.

`lags` Integer or vector of integers. Lag-order of the VAR models in levels, which is either a common p for all individuals or individual-specific p_i for each individual. In the vector, p_i must have the same succession $i = 1, \ldots, N$ as argument '`L.data`'.

`type` Character. The conventional case of the deterministic term.

`t_D` List of vectors. The activating break periods τ for the period-specific deterministic regressors in d_{it} of the VAR model in levels.

`D` List. A single '`data.frame`' of common deterministic regressors or a list of N '`data.frame`' objects of the individual deterministic regressors added to d_{it}. These customized regressors correspond to '`exogen`' in **vars**' `VAR`, which is fixed over bootstrap iterations.

`n.factors` Integer. Number of common factors to be used for SUR. Deactivated if `FALSE` (the default).

`n.iterations` Integer. The (maximum) number of iterations for the estimation of SUR resp. the pooled cointegrating vectors.

`dim_r` Integer. Common cointegration-rank r of the VECM.

`t_D1` List of vectors. The activating break periods τ for the period-specific deterministic regressors in $d_{1,it}$, which are restricted to the cointegration relations. Just as in `pcoint`, the accompanying lagged regressors are automatically included in $d_{2,it}$.

`t_D2` List of vectors. The activating break periods τ for the period-specific deterministic regressors in $d_{2,it}$, which are unrestricted.

`D1` List. A single '`data.frame`' of common deterministic regressors regressors or a list of N '`data.frame`' objects of individual deterministic regressors added to $d_{1,it}$, which are restricted to the cointegration relations. Unlike '`t_D1`', these customized regressors require potential accompanying lagged regressors to be manually included in $d_{2,it}$

`D2` List. A single '`data.frame`' of common deterministic regressors or a list of N '`data.frame`' objects of individual deterministic regressors added to $d_{2,it}$, which are unrestricted. These customized regressors correspond to '`dumvar`' in **urca**'s `ca.jo`, which is fixed over bootstrap iterations.

`idx_pool` Vector. Position $k = 1, ..., K + n_1$ of each variable to be pooled using the two-step estimator by Breitung (2005). The integer vector specifies throughout heterogeneous coefficients up to the uniform upper block I_r estimated with the individual estimator by Ahn and Reinsel (1990) if exclusively in the interval $[0, ..., r]$. Deactivated if `NULL` (the default).

Value

A list of class '`pvarx`' with the elements:

`A` Matrix. The lined-up coefficient matrices $A_j, j = 1, \ldots, p$ for the lagged variables in the panel VAR estimated by mean-group.

`B` Matrix. Placeholder for the structural impact matrix.

`beta` Matrix. The $((K + n_{d1}) \times r)$ cointegrating matrix of the VAR model if transformed from a rank-restricted VECM.

L.varx	List of 'varx' objects for the individual estimation results.
L.data	List of 'data.frame' objects for each individual.
CSD	List of measures for cross-sectional dependency. NULL if the individual VAR models have been estimated under independence.
args_pvarx	List of characters and integers indicating the estimator and specifications that have been used.

Functions

- pvarx.VAR(): Mean Group (MG) of VAR models in levels.
- pvarx.VEC(): (Pooled) Mean Group (PMG) of VECM.

References

Canova, F., and Ciccarelli, M. (2013): "Panel Vector Autoregressive Models: A Survey", *Advances in Econometrics*, 32, pp. 205-246.

Hsiao, C. (2014): *Analysis of Panel Data*, Econometric Society Monographs, Cambridge University Press, 3rd ed.

Luetkepohl, H. (2005): *New Introduction to Multiple Time Series Analysis*, Springer, 2nd ed.

Pesaran, M. H., and Smith R. J. (1995): "Estimating Long-Run Relationships from Dynamic Heterogeneous Panels", *Journal of Econometrics*, 68, pp. 79-113.

Rebucci, A. (2010): "Estimating VARs with Long Stationary Heterogeneous Panels: A Comparison of the Fixed Effect and the Mean Group Estimators", *Economic Modelling*, 27, pp. 1183-1198.

Ahn, S. K., and Reinsel (1990): "Estimation for Partially Nonstationary Multivariate Autoregressive Models", *Journal of the American Statistical Association*, 85, pp. 813-823.

Breitung, J. (2005): "A Parametric Approach to the Estimation of Cointegration Vectors in Panel Data", *Econometric Reviews*, 24, pp. 151-173.

Johansen, S. (1996): *Likelihood-based Inference in Cointegrated Vector Autoregressive Models*, Advanced Texts in Econometrics, Oxford University Press, USA.

Pesaran, M. H., Shin, Y, and Smith R. J. (1999): "Pooled Mean Group Estimation of Dynamic Heterogeneous Panels", *Journal of the American Statistical Association*, 94, pp. 621-634.

Examples

```
data("PCAP")
names_k = c("g", "k", "l", "y")  # variable names
names_i = levels(PCAP$id_i)      # country names
L.data  = sapply(names_i, FUN=function(i)
  ts(PCAP[PCAP$id_i==i, names_k], start=1960, end=2019, frequency=1),
  simplify=FALSE)
R.lags = c(2, 4, 2, 3, 2, 4, 4, 2, 2, 3, 3, 3, 2, 4, 4, 2, 2, 2, 4, 2, 2, 2, 4)
names(R.lags) = names_i

### MG of VAR by OLS ###
R.t_D  = list(t_shift=10)  # common level shift for all countries
R.pvar = pvarx.VAR(L.data, lags=R.lags, type="both", t_D=R.t_D)
```

```
R.pirf = irf(R.pvar, n.ahead=50)  # MG of individual forecast-error IRF
plot(R.pirf)

### Pooled MG of rank-restricted VAR ###
R.pvec = pvarx.VEC(L.data, lags=R.lags, dim_r=2, idx_pool=1:4, type="Case4")
R.pirf = irf(R.pvec, n.ahead=50)  # MG of individual forecast-error IRF
plot(R.pirf)
```

`rboot.normality` *Bootstrap for JB normality test*

Description

Bootstraps the distribution of the Jarque-Bera test for individual VAR and VECM as described by Kilian, Demiroglu (2000).

Usage

```
rboot.normality(x, n.boot = 1000, n.cores = 1, fix_beta = FALSE)
```

Arguments

`x`	VAR object of class '`varx`' or any other that will be coerced to '`varx`'.
`n.boot`	Integer. Number of bootstrap iterations.
`n.cores`	Integer. Number of allocated processor cores.
`fix_beta`	Logical. If `TRUE`, the cointegrating vectors β are fixed over all bootstrap iterations. Kilian and Demiroglu (2000:43) suggest this for VECM with known β. Ignored in case of rank-unrestricted VAR.

Value

A list of class '`rboot`' with elements:

`sim`	Array of dimension $(3 \times (1 + K) \times$ `n.boot`$)$ containing the `n.boot` iteration results for *(i)* the JB, skewness and kurtosis test and for *(ii)* the multivariate and each univariate test for the K residual time series.
`stats`	Matrix of dimension $(3 \times (1 + K))$ containing the empirical test statistics.
`pvals`	Matrix of dimension $(3 \times (1 + K))$ containing the p-values.
`varx`	Input VAR object of class '`varx`'.
`args_rboot`	List of characters and integers indicating the test and specifications that have been used.

References

Jarque, C. M. and Bera, A. K. (1987): "A Test for Normality of Observations and Regression Residuals", *International Statistical Review*, 55, pp. 163-172.

Kilian, L. and Demiroglu, U. (2000): "Residual-Based Tests for Normality in Autoregressions: Asymptotic Theory and Simulation Evidence", *Journal of Business and Economic Statistics*, 32, pp. 40-50.

See Also

...the `normality.test` by Pfaff (2008) in **vars**, which relies on theoretical distributions. Just as this asymptotic version, the bootstrapped version is computed by using the residuals that are standardized by a Cholesky decomposition of the residual covariance matrix. Therefore, the results of the multivariate test depend on the ordering of the variables in the VAR model.

Examples

```
# select minimal or full example #
is_min = TRUE
n.boot = ifelse(is_min, 50, 5000)

# prepare the data, estimate and test the VAR model #
set.seed(23211)
library("vars")
data("Canada")
exogen = cbind(qtrend=(1:nrow(Canada))^2)  # quadratic trend
R.vars = VAR(Canada, p=2, type="both", exogen=exogen)
R.norm = rboot.normality(x=R.vars, n.boot=n.boot, n.cores=1)

# density plot #
library("ggplot2")
R.data = data.frame(t(R.norm$sim[ , "MULTI", ]))
R.args = list(df=2*R.vars$K)
F.density = ggplot() +
  stat_density(data=R.data, aes(x=JB, color="bootstrap"), geom="line") +
  stat_function(fun=dchisq, args=R.args, n=500, aes(color="theoretical")) +
  labs(x="JB statistic", y="Density", color="Distribution", title=NULL) +
  theme_bw()
plot(F.density)
```

`sboot.mb` *Bootstrap with residual moving blocks for individual SVAR models*

Description

Calculates confidence bands for impulse response functions via recursive-design bootstrap.

Usage

```
sboot.mb(
  x,
  b.length = 1,
  n.ahead = 20,
  n.boot = 500,
  n.cores = 1,
  fix_beta = TRUE,
  deltas = cumprod((100:0)/100),
  normf = NULL
)
```

Arguments

x	VAR object of class 'id' or 'varx' or any other that can be coerced to 'varx', e.g. 'svars'. If a bias term x$PSI_bc is available for coefficient matrix A (such as in 'sboot2'), the bias-corrected second-step of the bootstrap-after-bootstrap procedure by Kilian (1998) is performed.
b.length	Integer. Length $b_{(t)}$ of each residual time series block, which is often set to $T/10$.
n.ahead	Integer. Number of periods to consider after the initial impulse, i.e. the horizon of the IRF.
n.boot	Integer. Number of bootstrap iterations.
n.cores	Integer. Number of allocated processor cores.
fix_beta	Logical. If TRUE (the default), the cointegrating vectors β are fixed over all bootstrap iterations. Ignored in case of rank-unrestricted VAR. Use this for VECM with known β, too. Note that β is fixed in vars:::.bootsvec, but not in vars:::.bootirfsvec or vars:::.bootirfvec2var.
deltas	Vector. Numeric weights δ_j that are successively multiplied to the bias estimate $\hat{\Psi}$ for a stationary correction. The default weights deltas = cumprod((100:0)/100) correspond to the iterative correction procedure of Step 1b in Kilian (1998). Choosing deltas = NULL deactivates the bootstrap-after-bootstrap procedure.
normf	Function. A given function that normalizes the $K \times S$ input-matrix into an output matrix of same dimension. See the example in id.iv for the normalization of Jentsch and Lunsford (2021) that fixes the size of the impact response in the IRF.

Value

A list of class 'sboot2' with elements:

true	Point estimate of impulse response functions.
bootstrap	List of length 'n.boot' holding bootstrap impulse response functions.
A	List for the VAR coefficients containing the matrix of point estimates 'par' and the array of bootstrap results 'sim'.
B	List for the structural impact matrix containing the matrix of point estimates 'par' and the array of bootstrap results 'sim'.

PSI_bc	Matrix of the estimated bias term $\hat{\Psi}$ for the VAR coefficients $\hat{A}$ according to Kilian (1998).
varx	Input VAR object of class 'varx' that has been subjected to the first-step bias-correction.
nboot	Number of correct bootstrap iterations.
b_length	Length of each block.
design	Character indicating that the recursive design bootstrap has been performed.
method	Used bootstrap method.
stars	Matrix of ($T \times$n.boot) integers containing the T resampling draws from each bootstrap iteration.

References

Breitung, J., Brueggemann R., and Luetkepohl, H. (2004): "Structural Vector Autoregressive Modeling and Impulse Responses", in *Applied Time Series Econometrics*, ed. by H. Luetkepohl and M. Kraetzig, Cambridge University Press, Cambridge.

Brueggemann R., Jentsch, C., and Trenkler, C. (2016): "Inference in VARs with Conditional Heteroskedasticity of Unknown Form", *Journal of Econometrics*, 191, pp. 69-85.

Jentsch, C., and Lunsford, K. G. (2021): "Asymptotically Valid Bootstrap Inference for Proxy SVARs", *Journal of Business and Economic Statistics*, 40, pp. 1876-1891.

Kilian, L. (1998): "Small-Sample Confidence Intervals for Impulse Response Functions", *Review of Economics and Statistics*, 80, pp. 218-230.

Luetkepohl, H. (2005): *New Introduction to Multiple Time Series Analysis*, Springer, 2nd ed.

See Also

mb.boot, irf, and the panel counterpart sboot.pmb.

Examples

```
# select minimal or full example #
is_min = TRUE
n.boot = ifelse(is_min, 5, 500)

# use 'b.length=1' to conduct basic "vars" bootstraps #
set.seed(23211)
data("Canada")
R.vars = vars::VAR(Canada, p=2, type="const")
R.svar = svars::id.chol(R.vars)
R.boot = sboot.mb(R.svar, b.length=1, n.boot=n.boot, n.ahead=30, n.cores=1)
summary(R.boot, idx_par="A", level=0.9)  # VAR coefficients with 90%-confidence intervals
plot(R.boot, lowerq = c(0.05, 0.1, 0.16), upperq = c(0.95, 0.9, 0.84))

# second step of bootstrap-after-bootstrap #
R.bab = sboot.mb(R.boot, b.length=1, n.boot=n.boot, n.ahead=30, n.cores=1)
summary(R.bab, idx_par="A", level=0.9)  # VAR coefficients with 90%-confidence intervals
plot(R.bab, lowerq = c(0.05, 0.1, 0.16), upperq = c(0.95, 0.9, 0.84))
```

```
# conduct bootstraps for Blanchard-Quah type SVAR from "vars" #
set.seed(23211)
data("Canada")
R.vars = vars::VAR(Canada, p=2, type="const")
R.svar = vars::BQ(R.vars)
R.boot = sboot.mb(R.svar, b.length=1, n.boot=n.boot, n.ahead=30, n.cores=1)
summary(R.boot, idx_par="B", level=0.9)  # impact matrix with 90%-confidence intervals
plot(R.boot, lowerq = c(0.05, 0.1), upperq = c(0.95, 0.9), cumulative=2:3)
# impulse responses of the second and third variable are accumulated

# set 'args_id' to CvM defaults of "svars" bootstraps #
set.seed(23211)
data("USA")
R.vars = vars::VAR(USA, lag.max=10, ic="AIC")
R.cob  = copula::indepTestSim(R.vars$obs, R.vars$K, verbose=FALSE)
R.svar = svars::id.cvm(R.vars, dd=R.cob)

R.varx = as.varx(R.svar, dd=R.cob, itermax=300, steptol=200, iter2=50)
R.boot = sboot.mb(R.varx, b.length=15, n.boot=n.boot, n.ahead=30, n.cores=1)
plot(R.boot, lowerq = c(0.05, 0.1, 0.16), upperq = c(0.95, 0.9, 0.84))
```

`sboot.mg` *Mean group inference for panel SVAR models*

Description

Calculates confidence bands for impulse response functions via mean group inference. The function does not perform bootstraps, but coerces the panel VAR object to class 'sboot' and, therewith, gives a distributional overview on the parameter heterogeneity.

Usage

```
sboot.mg(x, n.ahead = 20, normf = NULL, idx_i = NULL)
```

Arguments

`x`	Panel VAR object of class 'pid' or 'pvarx' or a list of VAR objects that will be coerced to 'varx'.
`n.ahead`	Integer. Number of periods to consider after the initial impulse, i.e. the horizon of the IRF.
`normf`	Function. A given function that normalizes the $K \times S$ input-matrix into an output matrix of same dimension. See the example in `id.iv` for the normalization of Jentsch and Lunsford (2021) that fixes the size of the impact response in the IRF.
`idx_i`	Logical or character vector. Names or N logical elements selecting a subset from the individuals $i = 1, \ldots, N$ for the MG estimation. If `NULL` (the default), all N individuals are included.

Details

MG inference presumes the individual estimates to be the empirical variation around a common parameter. In case of heterogeneous lag-orders p_i, specifically the 'summary' of VAR coefficient matrices fills $\hat{A}_{ij} = 0_{K \times K}$ for $p_i < j \leq max(p_1, \ldots, p_N)$ in accordance with the finite order VAR(p_i).

Value

A list of class 'sboot2' with elements:

true	Mean group estimate of impulse response functions.
bootstrap	List of length N holding the individual impulse response functions.
A	List for the VAR coefficients containing the matrix of mean group estimates 'par' and the array of individual results 'sim'.
B	List for the structural impact matrix containing the matrix of mean group estimates 'par' and the array of individual results 'sim'.
pvarx	Input panel VAR object of class 'pvarx'.
nboot	Integer '0' indicating that no bootstrap iteration has been performed.
method	Method used for inference.

References

Pesaran, M. H., and Smith R. J. (1995): "Estimating Long-Run Relationships from Dynamic Heterogeneous Panels", *Journal of Econometrics*, 68, pp. 79-113.

See Also

For an actual panel bootstrap procedure see sboot.pmb.

Examples

```
data("PCAP")
names_k = c("g", "k", "l", "y")  # variable names
names_i = levels(PCAP$id_i)      # country names
L.data  = sapply(names_i, FUN=function(i)
  ts(PCAP[PCAP$id_i==i, names_k], start=1960, end=2019, frequency=1),
  simplify=FALSE)
R.lags = c(2, 4, 2, 3, 2, 4, 4, 2, 2, 3, 3, 3, 2, 4, 4, 2, 2, 2, 4, 2, 2, 2, 4)
names(R.lags) = names_i
idx_nord = c("DNK", "FIN", "ISL", "SWE")

R.pvec = pvarx.VEC(L.data, lags=R.lags, dim_r=2, type="Case4")
R.pid  = pid.chol(R.pvec)
R.boot = sboot.mg(R.pid, idx_i=idx_nord)
plot(R.boot, lowerq=c(0, 0.25), upperq=c(1, 0.75))
summary(as.pvarx(R.pid$L.varx[idx_nord]))

# suppress imprecise results of restricted cointegrating coefficients #
dim_r = R.pvec$args_pvarx$dim_r
```

```
R.boot$beta$sim[ , 1:dim_r, ] = diag(dim_r)  # for normalized beta
summary(R.boot, idx_par="beta", level=0.95)
```

---

sboot.pmb *Bootstrap with residual panel blocks for panel SVAR models*

Description

Calculates confidence bands for impulse response functions via recursive-design bootstrap.

Usage

```
sboot.pmb(
  x,
  b.dim = c(1, 1),
  n.ahead = 20,
  n.boot = 500,
  n.cores = 1,
  fix_beta = TRUE,
  deltas = cumprod((100:0)/100),
  normf = NULL,
  w = NULL,
  MG_IRF = TRUE
)
```

Arguments

x	Panel VAR object of class 'pid' or 'pvarx' or a list of VAR objects that will be coerced to 'varx'. If a list x$L.PSI_bc of N bias terms are available for the N coefficient matrices A_i (such as in sboot2), the bias-corrected second-step of the bootstrap-after-bootstrap procedure by Empting et al. (2025) is performed.
b.dim	Vector of two integers. The dimensions $(b_{(t)}, b_{(i)})$ of each residual panel block for temporal and cross-sectional resampling. The default c(1, 1) specifies an $i.i.d.$ resampling in both dimensions, c(1, FALSE) a temporal resampling, and c(FALSE, 1) a cross-sectional resampling. Integers > 1 assemble blocks of consecutive residuals.
n.ahead	Integer. Number of periods to consider after the initial impulse, i.e. the horizon of the IRF.
n.boot	Integer. Number of bootstrap iterations.
n.cores	Integer. Number of allocated processor cores.
fix_beta	Logical. If TRUE (the default), the cointegrating vectors β are fixed over all bootstrap iterations. Ignored in case of rank-unrestricted VAR. Use this for VECM with known β, too. Note that β is fixed in vars:::.bootsvec, but not in vars:::.bootirfsvec or vars:::.bootirfvec2var.

`deltas` Vector. Numeric weights δ_j that are successively multiplied to each bias estimate $\hat{\Psi}_i$ for a stationary correction. The default weights `deltas = cumprod((100:0)/100)` correspond to the iterative correction procedure of Step 1b in Kilian (1998). Choosing `deltas = NULL` deactivates the bootstrap-after-bootstrap procedure.

`normf` Function. A given function that normalizes the $K \times S$ input-matrix into an output matrix of same dimension. See the example in `id.iv` for the normalization of Jentsch and Lunsford (2021) that fixes the size of the impact response in the IRF.

`w` Numeric, logical, or character vector. N numeric elements weighting the individual coefficients, or names or N logical elements selecting a subset from the individuals $i = 1, \ldots, N$ for the MG estimation. If `NULL` (the default), all N individuals are included without weights.

`MG_IRF` Logical. If `TRUE` (the default), the mean-group of individual IRF is calculated in accordance with Gambacorta et al. (2014). If `FALSE`, the IRF is calculated for the mean-group of individual VAR estimates.

Details

In case of heterogeneous lag-orders p_i or sample sizes T_i, the initial periods are fixed in accordance with the usage of presamples. Only the $(K \times T_{min} \times N)$ array of the $T_{min} = min(T_1, \ldots, T_N)$ last residuals is resampled.

Value

A list of class '`sboot2`' with elements:

`true` Mean group estimate of impulse response functions.

`bootstrap` List of length `nboot` holding bootstrap impulse response functions.

`A` List for the VAR coefficients containing the matrix of point estimates '`par`' and the array of bootstrap results '`sim`'.

`B` List for the structural impact matrix containing the matrix of point estimates '`par`' and the array of bootstrap results '`sim`'.

`L.PSI_bc` List of the N estimated bias terms $\hat{\Psi}_i$ for the individual VAR coefficients $\hat{A}_i$ according to Kilian (1998).

`pvarx` Input panel VAR object of class '`pvarx`' that has been subjected to the first-step bias-correction.

`b.dim` Dimensions of each block.

`nboot` Number of correct bootstrap iterations.

`design` Character indicating that the recursive design bootstrap has been performed.

`method` Used bootstrap method.

`stars_t` Matrix of ($T \times$`n.boot`) integers containing the T temporal resampling draws from each bootstrap iteration.

`stars_i` Matrix of ($N \times$`n.boot`) integers containing the N cross-sectional resampling draws from each bootstrap iteration.

References

Brueggemann R., Jentsch, C., and Trenkler, C. (2016): "Inference in VARs with Conditional Heteroskedasticity of Unknown Form", *Journal of Econometrics*, 191, pp. 69-85.

Empting, L. F. T., Maxand, S., Oeztuerk, S., and Wagner, K. (2025): "Inference in Panel SVARs with Cross-Sectional Dependence of Unknown Form".

Kapetanios, G. (2008): "A Bootstrap Procedure for Panel Data Sets with many Cross-sectional Units", *The Econometrics Journal*, 11, pp.377-395.

Kilian, L. (1998): "Small-Sample Confidence Intervals for Impulse Response Functions", *Review of Economics and Statistics*, 80, pp. 218-230.

Gambacorta L., Hofmann B., and Peersman G. (2014): "The Effectiveness of Unconventional Monetary Policy at the Zero Lower Bound: A Cross-Country Analysis", *Journal of Money, Credit and Banking*, 46, pp. 615-642.

See Also

For the the individual counterpart see `sboot.mb`.

Examples

```
# select minimal or full example #
is_min = TRUE
n.boot = ifelse(is_min, 5, 500)

# prepare data panel #
data("PCAP")
names_k = c("g", "k", "l", "y")  # variable names
names_i = levels(PCAP$id_i)      # country names
L.data  = sapply(names_i, FUN=function(i)
  ts(PCAP[PCAP$id_i==i, names_k], start=1960, end=2019, frequency=1),
  simplify=FALSE)
R.lags = c(2, 4, 2, 3, 2, 4, 4, 2, 2, 3, 3, 3, 2, 4, 4, 2, 2, 2, 4, 2, 2, 2, 4)
names(R.lags) = names_i

# estimate, identify, and bootstrap #
R.pvar = pvarx.VAR(L.data, lags=R.lags, type="both")
R.pid  = pid.chol(R.pvar)
R.boot = sboot.pmb(R.pid, n.boot=n.boot)
summary(R.boot, idx_par="A", level=0.95)  # VAR coefficients with 95%-confidence intervals
plot(R.boot, lowerq = c(0.05, 0.1, 0.16), upperq = c(0.95, 0.9, 0.84))

# second step of bootstrap-after-bootstrap #
R.bab = sboot.pmb(R.boot, n.boot=n.boot)
summary(R.bab, idx_par="A", level=0.95)  # VAR coefficients with 95%-confidence intervals
plot(R.bab, lowerq = c(0.05, 0.1, 0.16), upperq = c(0.95, 0.9, 0.84))
```

speci.factors *Criteria on the number of common factors*

Description

Determines the number of factors in an approximate factor model for a data panel, where both dimensions $(T \times KN)$ are large, and calculates the factor time series and corresponding list of N idiosyncratic components. See Corona et al. (2017) for an overview and further details.

Usage

```
speci.factors(
  L.data,
  k_max = 20,
  n.iterations = 4,
  differenced = FALSE,
  centered = FALSE,
  scaled = FALSE,
  n.factors = NULL
)
```

Arguments

L.data	List of N data.frame objects each collecting the K_i time series along the rows $t = 1, \ldots, T$. The $\sum_{i=1}^{N} K_i = NK$ time series are immediately combined into the $T \times KN$ data panel X.
k_max	Integer. The maximum number of factors to consider.
n.iterations	Integer. Number of iterations for the Onatski criterion.
differenced	Logical. If TRUE, each time series of the panel X is first-differenced prior to any further transformation. Thereby, all criteria are calculated as outlined by Corona et al. (2017).
centered	Logical. If TRUE, each time series of the panel X is centered.
scaled	Logical. If TRUE, each time series of the panel X is scaled. Thereby, the PCA is applied via the correlation matrix instead of the covariance matrix of X.
n.factors	Integer. A presumed number of factors under which the idiosyncratic component L.idio is calculated. Deactivated if NULL (the default).

Details

If differenced is TRUE, the approximate factor model is estimated as proposed by Bai, Ng (2004). If all data transformations are selected, the estimation results are identical to the objects in $CSD for PANIC analyses in 'pcoint' objects.

Value

A list of class 'speci', which contains the elements:

eigenvals	Data frame. The eigenvalues of the PCA, which have been used to calculate the criteria, and their respective share on the total variance in the data panel.
Ahn	Matrix. The eigenvalue ratio $ER(k)$ and growth rate $GR(k)$ by Ahn, Horenstein (2013) for $k = 0, \ldots,$k_max factors.
Onatski	Matrix. The calibrated threshold δ and suggested number of factors $\hat{r}(\delta)$ by Onatski (2010) for each iteration.
Bai	Array. The values of the criteria $PC(k)$, $IC(k)$, and $IPC(k)$ with penalty weights $p1$, $p2$, and $p3$ for $k = 0, \ldots,$k_max factors.
selection	List of the optimal number of common factors: (1) A matrix of k^* which minimizes each information criterion with each penalty weight. (2) A vector of k^* which maximizes ER and GR respectively. ED denotes the result by Onatski's (2010) "edge distribution" after convergence.
Ft	Matrix. The common factors of dimension ($T \times$ n.factors) estimated by PCA.
LAMBDA	Matrix. The loadings of dimension ($KN \times$ n.factors) estimated by OLS.
L.idio	List of N data.frame objects each collecting the K_i idiosyncratic series $\hat{e}_{it}$ along the rows $t = 1, \ldots, T$. The series $\hat{e}_{it}$ are given in levels and may contain a deterministic component with (1) the initial $\hat{e}_{i1}$ being non-zero and (2) re-accumulated means of the the first-differenced series.
args_speci	List of characters and integers indicating the specifications that have been used.

References

Ahn, S., and Horenstein, A. (2013): "Eigenvalue Ratio Test for the Number of Factors", *Econometrica*, 81, pp. 1203-1227.

Bai, J. (2004): "Estimating Cross-Section Common Stochastic Trends in Nonstationary Panel Data", *Journal of Econometrics*, 122, pp. 137-183.

Bai, J., and Ng, S. (2002): "Determining the Number of Factors in Approximate Factor Models", *Econometrica*, 70, pp. 191-221.

Bai, J., and Ng, S. (2004): "A PANIC Attack on Unit Roots and Cointegration", *Econometrica*, 72, pp. 1127-117.

Corona, F., Poncela, P., and Ruiz, E. (2017): "Determining the Number of Factors after Stationary Univariate Transformations", *Empirical Economics*, 53, pp. 351-372.

Onatski, A. (2010): "Determining the Number of Factors from Empirical Distribution of Eigenvalues", *Review of Econometrics and Statistics*, 92, pp. 1004-1016.

See Also

Other specification functions: speci.VAR()

Examples

```
### reproduce Oersal,Arsova 2017:67, Ch.5 ###
data("MERM")
names_k = colnames(MERM)[-(1:2)] # variable names
names_i = levels(MERM$id_i)      # country names
L.data  = sapply(names_i, FUN=function(i)
   ts(MERM[MERM$id_i==i, names_k], start=c(1995, 1), frequency=12),
   simplify=FALSE)

R.fac1 = speci.factors(L.data, k_max=20, n.iterations=4)
R.fac0 = speci.factors(L.data, k_max=20, n.iterations=4,
   differenced=TRUE, centered=TRUE, scaled=TRUE, n.factors=8)

# scree plot #
library("ggplot2")
pal = c("#999999", RColorBrewer::brewer.pal(n=8, name="Spectral"))
lvl = levels(R.fac0$eigenvals$scree)
F.scree = ggplot(R.fac0$eigenvals[1:20, ]) +
  geom_col(aes(x=n, y=share, fill=scree), color="black", width=0.75) +
  scale_fill_manual(values=pal, breaks=lvl, guide="none") +
  labs(x="Component number", y="Share on total variance", title=NULL) +
  theme_bw()
plot(F.scree)

# factor plot (comp. Oersal,Arsova 2017:71, Fig.4) #
library("ggfortify")
Ft = ts(R.fac0$Ft, start=c(1995, 1), frequency=12)
F.factors = autoplot(Ft, facets=FALSE, size=1.5) +
  scale_color_brewer(palette="Spectral") +
  labs(x="Year", y=NULL, color="Factor", title=NULL) +
  theme_bw()
plot(F.factors)
```

`speci.VAR` *Criteria on the lag-order and break period(s)*

Description

Determines the lag-order p and break period(s) τ jointly via information criteria on the OLS-estimated VAR model for a given number of breaks. These m breaks are common to all K equations of the system and partial, as pertaining the deterministic term only.

Usage

```
speci.VAR(
  x,
  lag_set = 1:10,
  dim_m = FALSE,
```

```
  trim = 0.15,
  type_break = "const",
  add_dummy = FALSE,
  n.cores = 1
)
```

Arguments

x	VAR object of class 'varx' or any other that will be coerced to 'varx'. Specifically for **vars**' VAR, use p = min(lag_set) or simply p=1 such that the customized $D from the coerced 'varx' object contains no NA in the effective sample.
lag_set	Vector. Set of candidates for the lag-order p. If only a single integer is provided, the criteria just reflect the variation of $\det(\hat{U}_\tau \hat{U}_\tau')$ uniformly and determine the break period(s) τ unanimously as $\hat{\tau} = \arg\min \det(\hat{U}_\tau \hat{U}_\tau')$ under the given p.
dim_m	Integer. Number of breaks in the deterministic term to consider. If FALSE (the default), the criteria determine only the lag-order p just like **vars**' VARselect.
trim	Numeric. Either a numeric value $h \in (p_{max}/T, 1/m)$ that defines the minimal fraction relative to the total sample size T or an integer that defines the minimal number of observations in each sub-sample. For example, $h = 0.15$ (the default) specifies the window $[0.15 \cdot T, 0.85 \cdot T]$ that is often used as the set of candidates for $m = 1$ single period τ_1.
type_break	Character. Whether the m common breaks pertain the 'const' (the default), the linear 'trend', or 'both'. Adds the period-specific deterministic term activated during τ.
add_dummy	Logical. If TRUE (not the default), accompanying impulse dummies activated in $\tau + (0, \ldots, p-1)$ are added to each break.
n.cores	Integer. Number of allocated processor cores. Note that parallel processing is exclusively activated for the combined determination of lag-order p and break period(s) τ only.

Details

The literature on structural breaks in time series deals mostly with the determination of the number m and position τ of breaks (e.g. Bai, Perron 1998 and 2003), but leaves the lag-order p aside. For example, under a given p, Luetkepohl et al. (2004) use a full-rank VAR in levels to determine $m = 1$ common break period τ_1 and subsequently perform cointegration analysis with coint.SL (which actually provides p-values for up to $m = 2$). Note yet that the lag-order of a VECM is usually determined via information criteria of a full-rank VAR in levels alike.

speci.VAR combines Bai, Perron (2003) and Approach 3 of Yang (2002) into a global minimization of information criteria on the pair (p, τ). Specifically, Yang (2002:378, Ch.2.2) estimates all candidate VAR models by OLS and then determines their optimal lag-order p^* and $m = 1$ break period τ^* jointly via the global minimum of the information criteria. Bai and Perron (2003, Ch.3) determine $\tau^* = (\tau_1^*, \ldots, \tau_m^*)$ of multiple breaks via the minimum sum of squared residuals from a single-equation model $(K = 1)$. They use dynamic programming to reduce the number of least-squares operations. Although adapting their streamlined set of admissible combinations for τ, speci.VAR yet resorts to (parallelized brute-force) OLS estimation of all candidate VAR models and therewith circumvents issues of correct initialization and iterative updating for the models with partial breaks.

Value

A list of class 'speci', which contains the elements:

df	A 'data.frame' of $(1 + m) + 4$ columns for all admissible combinations of candidate (p, τ) and their values of $AIC(p, \tau)$, $HQC(p, \tau)$, $SIC(p, \tau)$, and $FPE(p, \tau)$.
selection	A $(1 + m) \times 4$ matrix of the specification pairs (p^*, τ^*) suggested by the global minimum of the AIC (Akaike 1969), HQC (Hannan, Quinn 1979), SIC (Schwarz 1978), and FPE respectively.
args_speci	List of characters and integers indicating the specifications that have been used.

References

Bai, J., and Perron, P. (1998): "Estimating and Testing Linear Models with Multiple Structural Changes", *Econometrica*, 66, pp. 47-78.

Bai, J., and Perron, P. (2003): "Computation and Analysis of Multiple Structural Change Models", *Journal of Applied Econometrics*, 18, pp. 1-22.

Luetkepohl, H., Saikkonen, P., and Trenkler, C. (2004): "Testing for the Cointegrating Rank of a VAR Process with Level Shift at Unknown Time", *Econometrica*, 72, pp. 647-662.

Yang, M. (2002): "Lag Length and Mean Break in Stationary VAR Models", *Econometrics Journal*, 5, pp. 374-386.

See Also

Other specification functions: speci.factors()

Examples

```
### extend basic example in "urca" ###
library("urca")
library("vars")
data("denmark")
sjd = denmark[, c("LRM", "LRY", "IBO", "IDE")]

# use the single lag-order p=2 to determine only the break period #
R.vars  = VAR(sjd, type="both", p=1, season=4)
R.speci = speci.VAR(R.vars, lag_set=2, dim_m=1, trim=3, add_dummy=FALSE)

library("ggfortify")
autoplot(ts(R.speci$df[3:5], start=1+R.speci$args_speci$trim),
 main="For a single 'p', all IC just reflect the variation of det(UU').")
print(R.speci)

# perform cointegration test procedure with detrending #
R.t_D   = list(t_shift=8, n.season=4)
R.coint = coint.SL(sjd, dim_p=2, type_SL="SL_trend", t_D=R.t_D)
summary(R.coint)

# m=1: line plot #
```

```
library("ggplot2")
R.speci1 = speci.VAR(R.vars, lag_set=1:5, dim_m=1, trim=6)
R.values = c("#BDD7E7", "#6BAED6", "#3182BD", "#08519C", "#08306B")
F.line   = ggplot(R.speci1$df) +
  geom_line( aes(x=tau_1, y=HQC, color=as.factor(p), group=as.factor(p))) +
  geom_point(aes(x=tau_1, y=HQC, color=as.factor(p), group=as.factor(p))) +
  geom_point(x=R.speci1$selection["tau_1", "HQC"],
             y=min(R.speci1$df$HQC), color="red") +
  scale_x_continuous(limits=c(1, nrow(sjd))) +
  scale_color_manual(values=R.values) +
  labs(x=expression(tau), y="HQ Criterion", color="Lag order", title=NULL) +
  theme_bw()
plot(F.line)

# m=2: discrete heat map #
R.speci2 = speci.VAR(R.vars, lag_set=2, dim_m=2, trim=3)
dim_T    = nrow(sjd)  # total sample size
F.heat   = ggplot(R.speci2$df) +
  geom_point(aes(x=tau_1, y=tau_2, color=AIC), size=3) +
  geom_abline(intercept=0, slope=-1, color="grey") +
  scale_x_continuous(limits=c(1, dim_T), expand=c(0, 0)) +
  scale_y_reverse(limits=c(dim_T, 1), expand=c(0, 0)) +
  scale_color_continuous(type="viridis") +
  labs(x=expression(tau[1]), y=expression(tau[2]), color="AIC", title=NULL) +
  theme_bw()
plot(F.heat)
```

VECM *Estimation of a Vector Error Correction Model*

Description

Estimates a VECM under a given cointegration-rank restriction or cointegrating vectors.

Usage

```
VECM(
  y,
  dim_p,
  x = NULL,
  dim_q = dim_p,
  dim_r = NULL,
  beta = NULL,
  type = c("Case1", "Case2", "Case3", "Case4", "Case5"),
  t_D1 = list(),
  t_D2 = list(),
  D1 = NULL,
  D2 = NULL
)
```

Arguments

y	Matrix. A $(K \times (p+T))$ data matrix of the K endogenous time series variables.
dim_p	Integer. Lag-order p for the endogenous variables y.
x	Matrix. A $(L \times (p+T))$ data matrix of the L weakly exogenous time series variables.
dim_q	Integer. Lag-order q for the distributed lag of the weakly exogenous variables x.
dim_r	Integer. Cointegration-rank r of the VECM.
beta	Matrix. A $((K+L+n_{d1}) \times r)$ cointegrating matrix to be imposed – or estimated by the reduced-rank regression if NULL (the default).
type	Character. The conventional case of the deterministic term in the Johansen procedure.
t_D1	List of vectors. The activating break periods τ for the period-specific deterministic regressors in $d_{1,t}$, which are restricted to the cointegration relations. Just as in coint, the accompanying lagged regressors are automatically included in $d_{2,t}$.
t_D2	List of vectors. The activating break periods τ for the period-specific deterministic regressors in $d_{2,t}$, which are unrestricted.
D1	Matrix. A $(n_\bullet \times (p+T))$ data matrix of customized deterministic regressors added to $d_{1,t}$, which are restricted to the cointegration relations. Unlike 't_D1', these customized regressors require potential accompanying regressors to be manually included in $d_{2,it}$.
D2	Matrix. A $(n_\bullet \times (p+T))$ data matrix of customized deterministic regressors added to $d_{2,t}$, which are unrestricted. These additional regressors correspond to 'dumvar' in **urca**'s ca.jo, which is fixed over bootstrap iterations.

Value

A list of class 'varx'.

References

Johansen, S. (1996): *Likelihood-Based Inference in Cointegrated Vector Autoregressive Models*, Advanced Texts in Econometrics, Oxford University Press, USA.

Luetkepohl, H. (2005): *New Introduction to Multiple Time Series Analysis*, Springer, 2nd ed.

Examples

```
### extend basic example in "vars" ###
library(vars)
data(Canada)
names_k = c("e", "U", "rw")  # names of endogenous variables
names_l = c("prod")  # names of exogenous variables
names_s = NULL  # optional shock names
x = Canada[ , names_l, drop=FALSE]
y = Canada[ , names_k, drop=FALSE]
```

```
# colnames of the restriction matrices are passed as shock names #
SR = matrix(NA, nrow=4, ncol=4, dimnames=list(NULL, names_s))
SR[4, 2] = 0
LR = matrix(NA, nrow=4, ncol=4, dimnames=list(NULL, names_s))
LR[1, 2:4] = 0
LR[2:4, 4] = 0

# estimate, identify, and plot the IRF #
R.vecm = VECM(y=y, dim_p=3, x=x, dim_q=3, dim_r=1, type="Case4")
R.grt  = id.grt(R.vecm, LR=LR, SR=SR)
R.irf  = irf(R.grt, n.ahead=50)
plot(R.irf)
```

Index

∗ **cointegration rank tests**
  coint, 10
∗ **data sets**
  ERPT, 12
  EU_w, 13
  EURO, 14
  ICAP, 16
  MDEM, 24
  MERM, 25
  PCAP, 25
  PCIT, 28
∗ **datasets**
  ERPT, 12
  EU_w, 13
  EURO, 14
  ICAP, 16
  MDEM, 24
  MERM, 25
  PCAP, 25
  PCIT, 28
∗ **identification functions**
  id.grt, 17
  id.iv, 19
∗ **panel cointegration rank tests**
  pcoint, 29
∗ **panel estimation functions**
  pvarx, 46
∗ **panel identification functions**
  pid.chol, 32
  pid.cvm, 33
  pid.dc, 36
  pid.grt, 39
  pid.iv, 41
∗ **specification functions**
  speci.factors, 58
  speci.VAR, 60

as.pplot, 3, *4*
as.pvarx, *4*, 5, *6*
as.t_D, 7, *7*
as.varx, 8, *8*

BQ, *8*

ca.jo, *47*, *64*
coerced, *15*, *17*, *19*, *21*, *23*, *32*, *34*, *36*, *39*, *41*, *43*, *49*, *51*, *53*, *55*, *61*
coint, 10, *64*
coint.SL, *61*

deterministic regressors, *10, 11*, *30*, *47*, *64*
deterministic term, *10*, *29*, *47*, *60, 61*, *64*

ERPT, 12, *14, 15*, *17*, *24–26*, *29*
EU_w, 13, *13*, *15*, *17*, *24–26*, *29*
EURO, *13*, 14, *14*, *17*, *24–26*, *29*

facet_wrap, *3*
fevd.id, 15

geom_point, *3*

ICAP, *13–15*, 16, *24–26*, *29*
id, *18, 19*
id.chol, *9*
id.cvm, *9*, *35*
id.dc, *9*
id.grt, 17, *18*, *20*
id.iv, *4*, *18*, 19, *21*, *23*, *51*, *53*, *56*
irf, *52*
irf.pvarx, *4*, 21, *41*
irf.varx, *19*, 22

lm, *37*

mb.boot, *52*
MDEM, *13–15*, *17*, 24, *25, 26*, *29*
MERM, *13–15*, *17*, *24*, 25, *26*, *29*

normality.test, *50*

panel moving-block-, *37*
PCAP, *13–15*, *17*, *24*, 25, *25*, *29*
PCIT, *13–15*, *17*, *24–26*, 28
pcoint, 29, *47*
pcoint.BR, *45*
pcoint.CAIN, *7*, *45*
pcoint.JO, *45*
pcoint.SL, *45*
pid.chol, 32, *35*, *38*, *40*, *43*, *45*
pid.cvm, 33, *33–35*, *38*, *40*, *43*, *45*
pid.dc, *4*, *33*, *35*, 36, *40*, *43*, *45*
pid.grt, *33*, *35*, *38*, 39, *40*, *43*, *45*
pid.iv, *33*, *35*, *38*, *40*, 41, *45*
PP, *4*, 43
PP.variable, *43*
pvars, 45
pvars-package *(*pvars*)*, 45
pvarx, 46, *47*
pvarx.VAR, *37*, *45*
pvarx.VEC, *40*, *45*

rboot.normality, 49
rnorm, *17*, *39*

sboot.mb, *9*, *18, 19*, *37*, 50, *57*
sboot.mg, 53
sboot.pmb, *40, 41*, *45*, *52*, *54*, 55
speci.factors, *45*, 58, *62*
speci.VAR, *45*, *59*, 60, *61*
steadyICA, *37*
summary, *9*
svd, *19*, *42*
SVEC, *8*, *17, 18*, *39, 40*

toLatex, *9*

VAR, *7, 8*, *37*, *47*, *61*
VARselect, *61*
varx, *64*
vec2var, *8*, *43*
VECM, *18*, 63