Package 'VAR.spec'

January 20, 2025

Type Package

Title Allows Specifying a Bivariate VAR (Vector Autoregression) with Desired Spectral Characteristics

Version 1.0

Date 2024-06-7

Description The spectral characteristics of a bivariate series (Marginal Spectra, Coherency- and Phase-Spectrum) determine whether there is a strong presence of short-, medium-, or long-term fluctuations (components of certain frequencies in the spectral representation of the series) in each one of them. These are induced by strong peaks of the marginal spectra of each series at the corresponding frequencies. The spectral characteristics also determine how strongly these short-, medium-, or long-term fluctuations of the two series are correlated between the two series. Information on this is provided by the Coherency spectrum at the corresponding frequencies. Finally, certain fluctuations of the two series may be lagged to each other. Information on this is provided by the Phase spectrum at the corresponding frequencies. The idea in this package is to define a VAR (Vector autoregression) model with desired spectral characteristics by specifying a number of polynomials, required to define the VAR. See Ioannidis(2007) <doi:10.1016/j.jspi.2005.12.013>. These are specified via their roots, instead of via their coefficients. This is an idea borrowed from the Time Series Library of R. Dahlhaus, where it is used for defining ARMA models for univariate time series. This way, one may e.g. specify a VAR inducing a strong presence of long-term fluctuations in series 1 and in series 2, which are weakly correlated, but lagged by a number of time units to each other, while short-term fluctuations in series 1 and in series 2, are strongly present only in one of the two series, while they are strongly correlated to each other between the two series. Simulation from such models allows studying the behavior of data-analysis tools, such as estimation of the spectra, under different circumstances, as e.g. peaks in the spectra, generating bias, induced by leakage.

License GPL-2

LazyData true

NeedsCompilation no

Author Evangelos Ioannidis [cre, aut, cph],
Panagiotis Papastamoulis [aut, cph]

Maintainer Evangelos Ioannidis <eioannid@aueb.gr>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2024-06-11 11:30:05 UTC

Contents

VAR.spec-package | *Allows Specifying a Bivariate VAR (Vector Autoregression) with Desired Spectral Characteristics*

Description

The spectral characteristics of a bivariate series (Marginal Spectra, Coherency- and Phase-Spectrum) determine whether there is a strong presence of short-, medium-, or long-term fluctuations (components of certain frequencies in the spectral representation of the series) in each one of them. These are induced by strong peaks of the marginal spectra of each series at the corresponding frequencies. The spectral characteristics also determine how strongly these short-, medium-, or long-term fluctuations of the two series are correlated between the two series. Information on this is provided by the Coherency spectrum at the corresponding frequencies. Finally, certain fluctuations of the two series may be lagged to each other. Information on this is provided by the Phase spectrum at the corresponding frequencies. The idea in this package is to define a VAR (Vector autoregression) model with desired spectral characteristics by specifying a number of polynomials, required to define the VAR. See Ioannidis(2007) <doi:10.1016/j.jspi.2005.12.013>. These are specified via their roots, instead of via their coefficients. This is an idea borrowed from the Time Series Library of R. Dahlhaus, where it is used for defining ARMA models for univariate time series. This way, one may e.g. specify a VAR inducing a strong presence of long-term fluctuations in series 1 and in series 2, which are weakly correlated, but lagged by a number of time units to each other, while short-term fluctuations in series 1 and in series 2, are strongly present only in one of the two series, while they are strongly correlated to each other between the two series. Simulation from such models allows studying the behavior of data-analysis tools, such as estimation of the spectra, under different circumstances, as e.g. peaks in the spectra, generating bias, induced by leakage.

Details

The DESCRIPTION file:

Package:	VAR.spec
Type:	Package
Title:	Allows Specifying a Bivariate VAR (Vector Autoregression) with Desired Spectral Characteristics
Version:	1.0
Date:	2024-06-7
Authors@R:	c(person("Evangelos", "Ioannidis", email = "eioannid@aueb.gr", role = c("cre", "aut", "cph")), person("Panag
Description:	The spectral characteristics of a bivariate series (Marginal Spectra, Coherency- and Phase-Spectrum) determine
License:	GPL-2
LazyData:	true
Author:	Evangelos Ioannidis [cre, aut, cph], Panagiotis Papastamoulis [aut, cph]
Maintainer:	Evangelos Ioannidis <eioannid@aueb.gr>

Index of help topics:

```
Init.var                Initializes an object of class 'var'
VAR.inv.roots.from.det.cross
                        An example 'data.frame' defining a VAR (Vector
                        autoregression) model.
VAR.inv.roots.from.eta.ksi.zeta
                        An example 'data.frame' defining a VAR model.
VAR.spec-package        Allows Specifying a Bivariate VAR (Vector
                        Autoregression) with Desired Spectral
                        Characteristics
calc.VAR.spec.from.coefs
                        Calculates the spectral matrix of a
                        multivariate VAR (Vector autoregression) model.
calculate.VAR           Attempts to define a bivariate VAR (Vector
                        autoregression) model.
plot_VAR.Phase.details
                        Plots details related to the Phase spectrum of
                        a bivariate VAR (Vector autoregression) model.
plot_VAR.spectra        Plots spectra of a bivariate VAR (Vector
                        autoregression) model.
simulate.VAR            Simulates a bivariate series from a bivariate
                        VAR (Vector autoregression) model.
```

The specification of the VAR (Vector autoregression) model is based on the following fact (see Ioannidis (2007))

<doi:10.1016/j.jspi.2005.12.013>:

For any four complex polynomials $det(z), cross(z)$ of degree $2p$ and $chi.1(z), chi.2(z)$ of degree p, satisfying

EQ(*)

$$|det(z)|^2 + |cross(z)|^2 = |chi.1(z)|^2 * |chi.2(z)|^2$$

on $|z| = 1$, there exists a bivariate VAR(p) with marginal spectra

$$f.i(w) = (1/2\pi)|chi.i(z)|^2/|det(z)|^2,$$

and cross-spectrum

$$f.1.2(w) = (1/2\pi)z^{(-p)}cross(z)/|det(z)|^2,$$

where $z = exp(-iw)$. The squared Coherency is then given by

$$|cross(z)|^2/(|det(z)|^2 + |cross(z)|^2).$$

The idea in this package is to define the necessary polynomials by specifying their roots, instead of their coefficients. This is an idea borrowed from the Time Series Library of R. Dahlhaus, where it is used for defining ARMA models for univariate time series.

Moreover, the package allows the user to specify only some of the roots of $det(z), cross(z)$ and $chi.1(z), chi.2(z)$, while it attempts to find further non-specified roots in a way such that EQ(*) is satisfied.

By specifying certain roots of $det(z), cross(z)$ and $chi.1(z), chi.2(z)$ one can induce desired features in the spectra and in the series. For example,

- if one wishes a strong presence of frequency w_0 components in series 1, there should be a root with modulus close to 1 and angle equal to w_0, the multiplicity of which for $det(z)$ is higher than its multiplicity for $chi.1(z)$, forcing the spectrum of series 1 to have a peak at w_0.
- If the frequency-w_0-components should be strongly correlated between series 1 and series 2, the multiplicity of this root for $det(z)$ should be higher than its multiplicity for $cross(z)$, forcing the Coherency at w_0 close to 1.

Required polynomials (see argument `calc.method` of function `calculate.VAR` and related Details) are first passed to function `Init.var`, via a `data.frame` or a text file, which contains the multiplicities of desired roots (rows) for each required polynomial (columns).

After calling `Init.var`, function `calculate.VAR` must be called, which attempts to find a VAR model which is compatible with the polynomials specified in its attribute `inv.roots`, by making necessary adjustments, calculates its spectra, coefficients and order, makes the necessary checks and plots it's spectra. Then, one can simulate from the specified model calling `simulate.VAR`.

Author(s)

Evangelos Ioannidis [cre, aut, cph], Panagiotis Papastamoulis [aut, cph]

Maintainer: Evangelos Ioannidis <eioannid@aueb.gr>

References

Ioannidis, E. E. (2007). Spectra of bivariate VAR(p) models. Journal of Statistical Planning and Inference 137(2), 554-566.

Ioannidis, E. E. and Chronis, G. A. (2005). Extreme spectra of VAR models and orders of near-cointegration. J. Time Ser. Anal. 26, 399-421.

See Also

Init.var, calculate.VAR, simulate.VAR,

plot_VAR.spectra, plot_VAR.Phase.details,

calc.VAR.spec.from.coefs

Examples

```
my.var <- Init.var(grid=501, order.max.init=10, inv.roots.def=NULL)
my.var$inv.roots[2,]<- c(0.98,0.017261,2,3,1,1,2, rep(0,8))
my.var$inv.roots[3,]<- c(0.92,0.897598,2,1,1,1,2, rep(0,8))
my.var$inv.roots[4,]<- c(0.98,1.795196,1,1,0,1,1, rep(0,8))
my.var <- calculate.VAR (a.var = my.var, calc.method="from.det.cross",
            plot.spectra=TRUE,suppr.spec.check.warn=TRUE)
print(my.var$validity.msg)
```

---

calc.VAR.spec.from.coefs

Calculates the spectral matrix of a multivariate VAR (Vector autore-gression) model.

Description

Calculates the spectral matrix for all grid points for any (multivariate) VAR model, represented by an object of class var, with specified coefficients (in attribute ar.list).

Usage

```
calc.VAR.spec.from.coefs(a.var)
```

Arguments

a.var   an object of class var for which the spectral matrix will be calculated. Actually only the attributes grid, order and ar.list of a.var are needed.

Details

Is being called in calculate.VAR, so there is no need to call it again after calculate.VAR has been called. Can also be used to calculate the spectral matrix of a VAR model which has been fitted to data, e.g. using function ar. See Brockwell and Davis, 1990, Example 11.8.1, Hannan, 1970, Chapter II, Section 5.iv or Brillinger, 2001, Sections 2.8 and 2.9.

Value

Returns the object of class var after having calculated and set the following attributes:

freq   A one dimensional array of length grid, containing the grid-points at which spec has been calculated.

spec   A three dimensional array (gridxkxk), where k is the number of series of the VAR model. Contains for each grid-point the spectral matrix, as calculated at the basis of ar.list. That is, spec[j,1,1] is the spectrum of series 1 at grid-point j, spec[j,1,2] is the cross-spectrum between series 1 and 2 at grid-point j, etc.

References

Brillinger, D.R. (2001). Time Series: Data Analysis and Theory, second ed. Classics in Applied Mathematics, vol. 36. SIAM, Philadelphia.

Brockwell, P. J., Davis, R. A. (1990). Time Series: Theory and Methods, second ed. Springer, New York.

Hannan, E. J. (1970). Multiple Time Series. Wiley, New York.

See Also

VAR.spec-package,Init.var, calculate.VAR, simulate.VAR,

plot_VAR.spectra, plot_VAR.Phase.details

Examples

```
my.var <- Init.var(grid=501, order.max.init=10, inv.roots.def=NULL)
my.var$inv.roots[2,]<- c(0.98,0.017261,2,3,1,1,2, rep(0,8))
my.var$inv.roots[3,]<- c(0.92,0.897598,2,1,1,1,2, rep(0,8))
my.var$inv.roots[4,]<- c(0.98,1.795196,1,1,0,1,1, rep(0,8))
my.var <- calculate.VAR(a.var=my.var,calc.method="from.det.cross",
          suppr.spec.check.warn=TRUE)
print(my.var$validity.msg)
my.sample <-simulate.VAR(a.var=my.var, sample.size=250, burn.in = 500)
est.var<-list()
est.var$ar.outp <- ar(x=my.sample, aic=FALSE, order.max=6, demean=FALSE, method="ols",
series=c("ser1","ser2"))
est.var$label <-"MY VAR(6)"
est.var$order<-dim(est.var$ar.outp$ar)[1]
est.var$ar.list$ar <- est.var$ar.outp$ar
est.var$ar.list$var.pred <- est.var$ar.outp$var.pred
est.var$grid <- 501
est.var <- calc.VAR.spec.from.coefs(est.var)
plot_VAR.spectra(a.var=est.var,both=FALSE)
```

---

calculate.VAR	*Attempts to define a bivariate VAR (Vector autoregression) model.*

Description

Attempts to find a VAR model which is compatible with the polynomials specified in its attribute inv.roots, makes necessary adjustments, calculates its spectra, coefficients and order, makes the necessary checks and plots the spectra. Finding a VAR model involves finding complex polynomials $det(z), cross(z), chi.1(z)$ and $chi.2(z)$, satisfying EQ(*):

$$|det(z)|^2 + |cross(z)|^2 = |chi.1(z)|^2|chi.2(z)|^2$$

on $z| = 1$. If `calculate.VAR` is successful in finding such polynomials, there exists a bivariate VAR(p) model with marginal spectra

$$f.i(w) = (1/2\pi) * |chi.i(z)|^2/|det(z)|^2,$$

and cross-spectrum

$$f.1.2(w) = (1/2\pi) * z^{(-p)} * cross(z)/|det(z)|^2,$$

where $z = exp(-iw)$. The squared Coherency is then given by

$$|cross(z)|^2/(|det(z)|^2 + |cross(z)|^2).$$

Usage

```
calculate.VAR(a.var, calc.method = "from.det.cross", M.fact = 1.1,
              plot.spectra = TRUE,suppr.spec.check.warn=FALSE)
```

Arguments

`a.var`	An object of class var, previously initialized by `Init.var`.
`calc.method`	One of 'from.det.cross' or 'from.eta.ksi.zeta'. See details for further explanations.
`M.fact`	Numeric. A factor >1, by which the $min(|ksi.c(z)/[eta.1(z)eta.2(z)zeta(z)]|^2)$ is multiplied, in order to ensure that $|chi.1(z) * chi.2(z)|^2 - |det(z)|^2$ is positive on $|z| = 1$. Only relevant when `calc.method` = 'from.eta.ksi.zeta'. Affects the range of values of the Coherency spectrum.
`plot.spectra`	Logical. If TRUE, the marginal spectra, Coherency and Phase spectra will be plotted.
`suppr.spec.check.warn`	
	Logical. If TRUE, warnings on whether spectra obtained directly from $cross, det, chi.1$ and $chi.2$ (in `spec.1`, `spec.2`, `Coher`, `Phase`) differ by more than `eps.for.spectra` from the spectra (in `spec`) obtained from the coefficients of the VAR model (see `Note`, below) are suppressed.

Details

In order to define a VAR model, one seeks to define $det(z), cross(z)$ and $chi.1(z), chi.2(z)$, such that EQ(*) holds. There are two ways implemented for doing so, which are specified by argument `calc.method`:

- `calc.method` = 'from.det.cross'. (See Ioannidis (2007), Proposition 2). One must first specify the roots for $det(z)$ and $cross(z)$. They fix these two polynomials, which will not be modified during the calculation. One may also specify some roots for $chi.1(z)$ and $chi.2(z)$. If these two latter are compatible with EQ(*) they will be kept. If not, they may be modified during the calculation. Further roots may be added by `compute.VAR` during the calculation to $chi.2$ to ensure that EQ(*) holds. All changes are recorded in attribute `inv.roots`. If the user wishes some of the roots of $chi.2$ to be rather added to $chi.1$, he may do so by editing attribute `inv.roots` after running the function once, by increasing the multiplicity of the root for $chi.1$ and decreasing it by the same amount for $chi.2$, and then, re-running the function. With the option `calc.method` = 'from.det.cross' one has more direct control over peaks for the

marginal spectra (via $chi.i$ and det) and the Coherency being close to 0 or 1 (via det and $cross$), as well as the Phase spectrum, i.e., the lag-/lead- structure between the two series (via $cross$).

- calc.method = 'from.eta.ksi.zeta'. (See Ioannidis (2007), Proposition 1). One must specify the roots of $eta.1, eta.2, ksi.1, ksi.2$ and $zeta$. These fix

$$chi.i(z) = M * eta.i(z)zeta(z)ksi.j(z)/ksi.c(z))$$

and

$$det(z) = M * zeta(z)ksi.1(z)ksi.2(z)/ksi.c(z),$$

where M equals M.fact*$min(|ksi.c(z)/[eta.1(z)eta.2(z)zeta(z)]|^2)$ and $ksi.c$ is the maximal common divisor of $ksi.1(z)$ and $ksi.2(z)$. Then, $cross(z)$ is automatically determined so as that EQ(*) is satisfied. The result is that marginal spectra are given by

$$f.i(w) = |eta.i(z)|^2/|ksi.i(z)|^2$$

and the squared Coherency is given by

$$1 - |ksi.c(z)|^2/|M * eta.1(z)eta.2(z)zeta(z)|^2.$$

With the option calc.method = 'from.eta.ksi.zeta' one has more direct control over peaks (via ksi) and zeroes (via eta) of the marginal spectra and, partially, over the Coherency (via $ksi.c, eta$ and $zeta$), but not of the Phase spectrum. However, for certain aspects of the Coherency, a more direct control over the relation of $cross$ and det would be required, which is not available with this option. For example, with this option, it is not possible to set the multiplicity of a root to be higher for $cross$ than for det.

Value

An object of class var, basically a list including elements:

grid	The number of equidistant gridpoints in $[0, \pi]$, used for plotting and numerical approximations.
p.max.init	An initial limitation on the order of the VAR model.
inv.roots	A data.frame containing the roots of certain polynomials, used to specify the VAR model.
niter	An integer specifying the maximal number of iterations for the Innovations algorithm.
eps.max.for.UIA	
	Numeric close to 0. Specifies the threshold for the relative increase in precision achieved at a step of the Innovations algorithm, for accepting that the algorithm has converged.
eps.for.roots	Numeric close to 0. A threshold for considering two roots as identical, if a) the inverses of their radii and b) their angles (in their polar representation) differ by less than this threshold.
eps.for.spectra	
	Numeric close to 0. A threshold for checking the validity of the calculation of the VAR model. It's spectra are calculated on the basis of the polynomials used to specify the VAR model in inv.roots.def should coincide with the spectra calculated on the basis of the VAR model's coefficients. If they differ by more than a.eps.for.spectra warning messages will be written out to the console.

validity.msg	A message containing more precise information on a) the convergence of the Innovations Algorithm and b) on the maximal difference between the spectra of the VAR model as calculated on the basis of the polynomials used to specify it and the spectra obtained on the basis of the VAR model's coefficients. The message should be printed after each call to calculate.var.
order	(Integer) The final order of the VAR model.
ar.list	A list with entries (calculated solving the Yule-Walker equations):

- order (Integer) The final order of the VAR model.
- ar A three dimensional array (orderx2x2), containing the coefficients of the VAR model.
- var.pred A 2x2 matrix containing the covariance matrix of the innovations.

cov.1, cov.2, cov.cross, cov.cross.neg	
	Three one dimensional arrays of length order+1 containing the auto- and the cross- covariances (for lags and leads respectively) of the two series for lags=0,...,order. They are being calculated as Dirichlet-approximations to the Fourier transforms of the spectra, which have been calculated from $det, cross, chi.1$ and $chi.2$.
freq	A one dimensional array of length grid. It contains the grid-points at which spectra are being calculated.
spec	A three dimensional array (gridx2x2). Contains for each grid-point the spectral matrix, as calculated **at the basis of the VAR model's coefficients**, given in attribute ar.list. That is, spec[j,1,1] is the spectrum of series 1 at grid-point j, spec[j,1,2] is the cross-spectrum between the series 1 and 2 at grid-point j, etc.
spec.1, spec.2	Two one dimensional arrays of length grid containing the **log** of the values of the spectra of series 1 and 2, respectively, **as calculated from** chi, det, i.e. spec.1$[j] = log[(1/2\pi)|chi.i(z)|^2/|det(z)|^2]$, where $z = exp(-ij\pi/(\text{grid} - 1))$.
Coher	A one dimensional array of length grid containing the values of the log of the squared Coherency spectrum between series 1 and 2, **as calculated from** $cross, det$, i.e. Coher$[j] = log(|cross(z)|^2/(|det(z)|^2 + |cross(z)|^2))$, $z = exp(-ij\pi/(\text{grid} - 1))$.
Phase	A one dimensional array of length grid containing the values of the Phase spectrum between series 1 and 2, **as calculated from** $cross$, i.e., Phase$(w) = arg[z^{(-p)} * cross(z)]$, where $z = exp(-ij\pi/(\text{grid} - 1))$.
Phase.div.freq, group.delay	
	Two one dimensional arrays of length grid containing the values of Phase(w)/w, which gives the lead/lag in units of time, and the derivative of Phase(w) with respect to w, respectively.
det, cross, chi.1, chi.2, ksi.1, ksi.2, eta.1, eta.2, zeta	
	Contain detailed information on the respective polynomials (the last five only if calc.method = 'from.eta.ksi.zeta' was used), as modified during the calculation, each one with attributes:

- const the constant of the polynomial;
- inv.roots the modulus and angle for the inverses of the roots of the polynomial (for conjugate pairs only the one with angle in $[0, \pi]$);

- `inv.roots.number`, the number of the roots of the polynomial (pairs of complex roots are counted only once);
- `order`, the degree of the polynomial;
- `coefs`, an array of length $2(\texttt{inv.roots.number} + 1)$ containing the coefficients of the polynomial;
- `fourier.coefs`, an array of length $2\texttt{pmax.init} + 1$ containing the Fourier transform of the $|polynomial(z)|^2$ on $|z| = 1$;
- `values$spec`, an array of length `grid`, containing the values of the squared modulus of the polynomial, i.e. $|polynomial(z)|^2$ on $|z| = 1$;
- `Inv.values$spec`, an array of length `grid`, containing the **log of the inverses** of the values of the squared modulus of the polynomial, i.e. $-log[|polynomial(z)|^2]$ on $|z| = 1$.

Note

`calculate.VAR` performs a check on the validity of the output and stores relevant information in attribute `validity.msg`. There are certain reasons for which the output may not be valid: First, the calculation involves numerical approximations; the relevant algorithms may have not achieved convergence to the desired precision. Second, there may be features in the spectra, which cannot be well represented by a VAR model, resulting in polynomials $cross, det, chi.1$ and $chi.2$ not satisfying EQ(*). Thus, it is imperative to check a) the validity of EQ(*). It is also important to check b) that the spectra obtained directly from $cross, det, chi.1$ and $chi.2$ (in `spec.1, spec.2, Coher, Phase`) coincide with the spectra (in `spec`) obtained from the coefficients of the VAR model (in `ar.list`). The checks performed by `calculate.VAR` concern exactly the checks under a) and b): the maximum absolute difference between the relevant quantities (in log scale, where appropriate) is passed in `validity.msg`. Moreover, warnings are written out to the console if the Innovations algorithm did not converge or if any of the quantities in 'b' exceeds `eps.for.spectra`, provided that `suppr.spec.check.warn=FALSE`. The same check is conducted visually if `plot.spectra=TRUE`: each spectrum is calculated and plotted according to both methods. The two should coincide.

The option `method='from.det.cross'` seems generally numerically more stable than the option

`method='from.eta.ksi.zeta'`.

Suggestions on how to obtain desired spectral characteristics:

Let $z_0 := r_0^{-1} exp(iw_0)$, with r_0 close to 1. For a polynomial $p(z)$ denote by $[p(z_0)]$ the multiplicity of z_0 as root of $p(z)$.

To obtain

- a peak in $f.1(w_0)$
  - with `method='from.det.cross'`: choose $[det(z_0)] > [chi.1(z_0)]$.
  - with `method='from.eta.ksi.zeta'`: choose $[ksi.1(z_0)] > [eta.1(z_0)]$.
- a trough in $f.1(w_0)$
  - with `method='from.eta.ksi.zeta'`: choose $[ksi.1(z_0)] < [eta.1(z_0)]$.
- a $Coh(w_0)$ close to 1
  - with `method='from.det.cross'`: choose $[det(z_0)] > [cross(z_0)]$.

     – with method='from.eta.ksi.zeta': choose $min([ksi.1(z_0)], [ksi.2(z_0)]) > [eta.1(z_0)] + [eta.2(z_0)] + [zeta(z_0)]$.

- a $Coh(w_0)$ close to 0

     – with method='from.det.cross': choose $[det(z_0)] < [cross(z_0)]$.

     – with method='from.eta.ksi.zeta': choose $min([ksi.1(z_0)], [ksi.2(z_0)]) < [eta.1(z_0)] + [eta.2(z_0)] + [zeta(z_0)]$.

References

Ioannidis, E. E. (2007). Spectra of bivariate VAR(p) models. Journal of Statistical Planning and Inference 137(2), 554-566.

Ioannidis, E. E. and Chronis, G. A. (2005). Extreme spectra of VAR models and orders of near-cointegration. J. Time Ser. Anal. 26, 399-421.

See Also

VAR.spec-package, Init.var, simulate.VAR,

plot_VAR.spectra, plot_VAR.Phase.details,

calc.VAR.spec.from.coefs

Examples

```
my.var <- Init.var(grid=501, order.max.init=10, inv.roots.def=NULL)
my.var$inv.roots[2,]<- c(0.98,0.017261,2,3,1,1,2, rep(0,8))
my.var$inv.roots[3,]<- c(0.92,0.897598,2,1,1,1,2, rep(0,8))
my.var$inv.roots[4,]<- c(0.98,1.795196,1,1,0,1,1, rep(0,8))
my.var <- calculate.VAR (a.var = my.var, calc.method="from.det.cross",
          plot.spectra=TRUE,suppr.spec.check.warn=TRUE)
print(my.var$validity.msg)
```

Init.var   *Initializes an object of class* var

Description

Initializes an object of class var, representing a VAR (Vector autoregression) model, sets some of its parameters and reads the specification of certain polynomials which determine the VAR model from a data.frame or from a tab-delimited text file (, which is converted to a data.frame). In this data.frame each polynomial is specified via it's (complex) roots and constant. If no such text file or data.frame is specified, an appropriate default data.frame is created, which can then be modified by the user.

Usage

```
Init.var(grid = 1001, order.max.init = 10, inv.roots.def = NULL,
              a.niter = 5000, a.eps.max.for.UIA = 1e-10,
              a.eps.for.roots = 1e-05,a.eps.for.spectra=1E-4)
```

Arguments

grid   An integer specifying the number of equidistant gridpoints in $[0, \pi]$ used for
                     plotting and numerical approximations. They are given by $j\pi/((grid-1)), j =$
                     $0, ..., grid$.

order.max.init An integer providing an initial limitation on the order of the VAR model. The
                     degrees of the polynomials specified in the `inv.roots.def` should not exceed
                     2*`order.max.init` and the lines of the `inv.roots.def` should be at least equal
                     to 6*`order.max.init`+2 (if this not the case, appropriate lines will be automat-
                     ically added).

inv.roots.def Either NULL or a `data.frame` or a tab-delimited text file (, which is converted
                     to a `data.frame`). This `data.frame` contains the roots (rows) of the polyno-
                     mials (columns) used to specify the VAR model. See `Details` for the spe-
                     cific structure of the text file. If set to NULL, a `data.frame` of the appropri-
                     ate structure will be created, which contains no roots, and corresponds, thus,
                     to a bivariate white noise series. The `data.frame` in any of the above cases
                     may be modified after the call of `init.var`, via e.g. `var.name$inv.roots <-`
                     `edit(var.name$inv.roots)`.

a.niter   An integer specifying the maximal number of iterations for the Innovations al-
                     gorithm.

a.eps.max.for.UIA

                     Numeric close to 0. Specifies the threshold for the relative increase in precision
                     achieved at a step of the Innovations algorithm, for accepting that the algorithm
                     has converged.

a.eps.for.roots

                     Numeric close to 0. A threshold for considering two roots as identical, if a) the
                     inverses of their radii and b) their angles (in their polar representation) differ by
                     less than this threshold.

a.eps.for.spectra

                     Numeric close to 0. A threshold for checking the validity of the calculation of
                     the VAR model. It's spectra are calculated on the basis of the polynomials used
                     to specify the VAR model in `inv.roots.def` should coincide with the spectra
                     calculated on the basis of the VAR model's coefficients. If they differ by more
                     than `a.eps.for.spectra` warning messages will be written out to the console.

Details

The text file [/ `data.frame`] passed in argument `inv.roots.def` should have a specific structure:

  - The first row of the file (headers) [/ names of columns of the `data.frame`] should be 'radius
    angle det cross chi.1 chi.2 chi.1.prod.2 ma.1 ma.2 eta.1 eta.2 ksi.1 ksi.2 ksi.c zeta'.

- The second row of the file [/ first row-'observation' in the data.frame] contains the constants of the polynomials, usually set initially to 1, while the first two entries, corresponding to columns radius and angle, are set to '#NA' [/ 'NA'].

- Subsequent rows, containing each one the specification of a root and the multiplicity of this root for each relevant polynomial, should have the following entries: The first entry is the inverse of its modulus (radius). For all polynomials, except for cross, this should be in $[0, 1)$. The second entry is it's argument (angle). This should be a figure in $[0, \pi]$. Complex conjugates for each non-real root will be automatically added, in order to ensure real coefficients of the polynomials. Subsequent entries in the row are non-negative integers determining the multiplicity of the given root for the polynomial corresponding to the column of the entry. Not all polynomials need to be specified (see parameter calc.method in function calculate.VAR). Non relevant polynomials should have '0' as entry for their multiplicity of the root.

- Subsequent rows containing no specified roots should have the following structure: the first two entries, corresponding to radius and angle, should equal '#NA' [/ 'NA']. The rest of the row should be filled with '0'. The total number of rows should be at least 6*order.max.init+2 for the file (, including headers) [/ or 6*order.max.init+1 for the data.frame].

Two example text files of the appropriate structure, VAR.inv.roots.from.det.cross.txt and VAR.inv.roots.from.eta.ksi.zeta.txt may be found in the extdata folder of the package. A path to them may be obtained via

fpath<-system.file("extdata","text.file",package="VAR.spec").

The corresponding example data.frames are also directly accessible to the user as

VAR.inv.roots.from.det.cross and VAR.inv.roots.from.eta.ksi.zeta.

Value

An object of class var, basically a list including elements:

grid	The integer contained in argument grid. The number of equidistant gridpoints in $[0, \pi]$, used for plotting and numerical approximations.
p.max.init	The integer contained in argument order.max.init. An initial limitation on the order of the VAR model.
inv.roots	A data.frame containing the roots of certain polynomials, required to specify the VAR model.
niter	The integer contained in argument a.niter. An integer specifying the maximal number of iterations for the Innovations algorithm.
eps.max.for.UIA	
	Numeric close to 0. Specifies the threshold for the relative increase in precision achieved at a step of the Innovations algorithm, for accepting that the algorithm has converged.
eps.for.roots	Numeric close to 0. A threshold for considering two roots as identical, if a) the inverses of their radii and b) their angles (in their polar representation) differ by less than this threshold.
eps.for.spectra	
	Numeric close to 0. A threshold for checking the validity of the calculation of the VAR model. It's spectra are calculated on the basis of the polynomials used

to specify the VAR model in `inv.roots.def` should coincide with the spectra calculated on the basis of the VAR model's coefficients. If they differ by more than `a.eps.for.spectra` warning messages will be written out to the console.

See Also

[VAR.spec-package](#), [calculate.VAR](#), [simulate.VAR](#),
[plot_VAR.spectra](#), [plot_VAR.Phase.details](#),
[calc.VAR.spec.from.coefs](#)

Examples

```
my.var <- Init.var(grid=201, order.max.init=10, inv.roots.def=NULL)
my.var$inv.roots[2,]<- c(0.98,0.017261,2,3,1,1,2, rep(0,8))
my.var$inv.roots[3,]<- c(0.92,0.897598,2,1,1,1,2, rep(0,8))
my.var$inv.roots[4,]<- c(0.98,1.795196,1,1,0,1,1, rep(0,8))
my.var$inv.roots
```

---

plot_VAR.Phase.details

Plots details related to the Phase spectrum of a bivariate VAR (Vector autoregression) model.

Description

Plots in a 2x2 layout the Phase spectrum of a VAR model for a bivariate series, represented by an object of class var, the Phase spectrum divided by frequency -which gives the lag-/lead-structure between the two series in units of time –, the derivative of the Phase spectrum with respect to frequency -also known as the group-delay–, and the Coherency spectrum of the VAR model.

Usage

```
plot_VAR.Phase.details(a.var)
```

Arguments

a.var   An object of class var obtained from `calculate.VAR` for which details of the Phase spectrum are to be plotted.

Value

No return value, called for side effects.

See Also

[VAR.spec-package](#),[Init.var](#), [calculate.VAR](#),
[simulate.VAR](#), [plot_VAR.spectra](#),
[calc.VAR.spec.from.coefs](#)

Examples

```
my.var <- Init.var(grid=501, order.max.init=10, inv.roots.def=NULL)
my.var$inv.roots[2,]<- c(0.98,0.017261,2,3,1,1,2, rep(0,8))
my.var$inv.roots[3,]<- c(0.92,0.897598,2,1,1,1,2, rep(0,8))
my.var$inv.roots[4,]<- c(0.98,1.795196,1,1,0,1,1, rep(0,8))
my.var <- calculate.VAR(a.var=my.var,calc.method="from.det.cross",
                        suppr.spec.check.warn=TRUE)
print(my.var$validity.msg)
plot_VAR.Phase.details (a.var=my.var)
```

---

plot_VAR.spectra	*Plots spectra of a bivariate VAR (Vector autoregression) model.*

Description

Plots in a 2x2 layout the marginal spectra, the squared Coherency spectrum and the Phase spectrum of a VAR model, represented by an object of class var. If both=TRUE, each one is calculated (and plotted) in two ways: once directly from $det, cross, chi.1, chi.2$, and once from the coefficients in ar.list. For each quantity the values from the two ways of calculation should coincide, if the calculation of the VAR model undertaken in calculate.VAR was correct. If both=FALSE, only the calculation based on the coefficients in attribute ar.list is plotted.

Usage

```
plot_VAR.spectra(a.var, both = TRUE)
```

Arguments

a.var   An object of class var for which spectra are to be plotted.

both   Logical. If TRUE, each quantity will be plotted calculated in two ways: once directly from $det, cross, chi.1, chi.2$ and once from the coefficients in ar.list. If FALSE, each quantity will be plotted calculated from the coefficients in ar.list only.

Details

If both=TRUE the var should have been obtained from calculate.VAR. If both=FALSE only the attributes grid, freq and spec are required. Thus, it can also be used to plot spectra of a VAR model fitted to a bivariate series, e.g. using function ar, the spectra of which have previously been calculated by a call to calc.VAR.spec.from.coefs.

Value

No return value, called for side effects.

See Also

VAR.spec-package, Init.var, calculate.VAR,

simulate.VAR, plot_VAR.Phase.details,

calc.VAR.spec.from.coefs

Examples

```
my.var <- Init.var(grid=501, order.max.init=10, inv.roots.def=NULL)
my.var$inv.roots[2,]<- c(0.98,0.017261,2,3,1,1,2, rep(0,8))
my.var$inv.roots[3,]<- c(0.92,0.897598,2,1,1,1,2, rep(0,8))
my.var$inv.roots[4,]<- c(0.98,1.795196,1,1,0,1,1, rep(0,8))
my.var <- calculate.VAR(a.var=my.var,calc.method="from.det.cross",
          suppr.spec.check.warn=TRUE)
print(my.var$validity.msg)
my.var <- calc.VAR.spec.from.coefs(a.var=my.var)
plot_VAR.spectra(a.var=my.var)
```

---

simulate.VAR	*Simulates a bivariate series from a bivariate VAR (Vector autoregression) model.*

Description

Simulates a Gaussian bivariate series from a VAR model, described by an object of class var.

Usage

```
simulate.VAR(a.var, sample.size, burn.in = 1000)
```

Arguments

a.var	An object of class var, obtained from calculate.VAR. Actually only the attributes order and ar.list of a.var are needed.
sample.size	Integer. The size of the bivariate sample to be generated.
burn.in	Integer. The number of initial observations to be discarded. The bivariate series is simulated by iterating the VAR recursion, starting with white noise (with the covariance structure of the innovations, given in ar.list$var.pred). This iteration converges to the correct distribution after a burn.in period, provided the roots of det are outside the unit circle, i.e. their inverses have a modulus less than 1. The closer roots of det are to the unit circle, the longer the burn.in period should be chosen.

Value

A numeric array of dimensions sample.sizex2, the bivariate sample series simulated from the VAR model.

See Also

VAR.spec-package, Init.var, calculate.VAR, plot_VAR.spectra, plot_VAR.Phase.details, calc.VAR.spec.from.coefs

Examples

```
my.var <- Init.var(grid=501, order.max.init=10, inv.roots.def=NULL)
my.var$inv.roots[2,]<- c(0.98,0.017261,2,3,1,1,2, rep(0,8))
my.var$inv.roots[3,]<- c(0.92,0.897598,2,1,1,1,2, rep(0,8))
my.var$inv.roots[4,]<- c(0.98,1.795196,1,1,0,1,1, rep(0,8))
my.var <- calculate.VAR(a.var=my.var,calc.method="from.det.cross",
                          suppr.spec.check.warn=TRUE)
print(my.var$validity.msg)
my.sample <-simulate.VAR(a.var=my.var, sample.size=250, burn.in = 500)
```

VAR.inv.roots.from.det.cross

An example data.frame *defining a VAR (Vector autoregression) model.*

Description

An example data.frame to be used for argument inv.roots.def of function Init.var, when subsequently argument method of function calculate.VAR is set to "from.det.cross".

Usage

```
VAR.inv.roots.from.det.cross
```

Format

A data.frame containing 3 roots (rows) and their multiplicities for the polynomials $det(z)$, $cross(z)$, $chi.1(z)$ and $chi.2(z)$, as well as their constants in the first row.

See Also

Init.var, calculate.VAR

Examples

```
my.var <- Init.var(grid=501, order.max.init=10, inv.roots.def=VAR.inv.roots.from.det.cross)
my.var <- calculate.VAR(a.var=my.var,calc.method="from.det.cross",
                          suppr.spec.check.warn=TRUE)
print(my.var$validity.msg)
```

VAR.inv.roots.from.eta.ksi.zeta

An example data.frame *defining a VAR model.*

Description

An example data.frame to be used for argument inv.roots.def of function Init.var, when subsequently argument method of function calculate.VAR is set to "from.eta.ksi.zeta".

Usage

```
VAR.inv.roots.from.eta.ksi.zeta
```

Format

A data.frame containing 3 roots (rows) and their multiplicities for the polynomials $eta.1(z), eta.2(z),$ $ksi.1(z), ksi.2(z)$ and $zeta(z)$, as well as their constants in the first row.

See Also

Init.var, calculate.VAR

Examples

```
my.var <- Init.var(grid=501, order.max.init=10,
                      inv.roots.def=VAR.inv.roots.from.eta.ksi.zeta)
my.var <- calculate.VAR(a.var=my.var,calc.method="from.eta.ksi.zeta")
```

Index