Package 'gmvarkit'

August 5, 2026

Title Estimate Gaussian and Student's t Mixture Vector Autoregressive Models

Version 2.2.2

Author Savi Virolainen [aut, cre] (ORCID: <https://orcid.org/0000-0002-5075-6821>)

Maintainer Savi Virolainen <savi.virolainen@gmail.com>

Description

Unconstrained and constrained maximum likelihood estimation of structural and reduced form Gaussian mixture vector autoregressive, Student's t mixture vector autoregressive, and Gaussian and Student's t
mixture vector autoregressive models, quantile residual tests, graphical diagnostics, simulations, forecasting, and estimation of generalized impulse response function and generalized forecast error variance decomposition.
Leena Kalliovirta, Mika Meitz, Pentti Saikkonen (2016) <doi:10.1016/j.jeconom.2016.02.012>,
Savi Virolainen (2025) <doi:10.1080/07350015.2024.2322090>,
Savi Virolainen (in press) <doi:10.1016/j.ecosta.2025.09.003>.

Depends R (>= 3.6.0)

BugReports https://github.com/saviviro/gmvarkit/issues

License GPL-3

Encoding UTF-8

LazyData true

Imports Brobdingnag (>= 1.2-4), mvnfast (>= 0.2.5), parallel (>= 3.0.0), stats (>= 3.0.0), pbapply (>= 1.4-2), graphics (>= 3.0.0), grDevices (>= 3.0.0), gsl (>= 2.1-6), methods (>= 3.0.0)

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2026-08-04 22:50:20 UTC

Config/roxygen2/version 8.0.0

Contents

gmvarkit-package . . . 3
add_data . . . 4
alt_gsmvar . . . 5
calc_gradient . . . 7
check_parameters . . . 8
cond_moments . . . 13
cond_moment_plot . . . 17
diagnostic_plot . . . 19
diag_Omegas . . . 20
estimate_sgsmvar . . . 21
euromone . . . 24
fitGSMVAR . . . 25
GAfit . . . 30
gdpdef . . . 37
get_boldA_eigens . . . 38
get_omega_eigens . . . 39
get_regime_autocovs . . . 40
get_regime_means . . . 41
GFEVD . . . 42
GIRF . . . 45
gmvar_to_gsmvar . . . 50
GSMVAR . . . 51
gsmvar_to_sgsmvar . . . 56
in_paramspace . . . 58
in_paramspace_int . . . 62
iterate_more . . . 65
linear_IRF . . . 67
loglikelihood . . . 70
LR_test . . . 74
Pearson_residuals . . . 76
plot.gsmvarpred . . . 77
plot.qrtest . . . 78
predict.gsmvar . . . 80
print.gsmvarpred . . . 82
print.gsmvarsum . . . 83
print.hypotest . . . 84
print_std_errors . . . 84
profile_logliks . . . 85
quantile_residuals . . . 88
random_ind2 . . . 89
Rao_test . . . 92
redecompose_Omegas . . . 93
reorder_W_columns . . . 95
simulate.gsmvar . . . 97
stmvar_to_gstmvar . . . 99
swap_parametrization . . . 101

swap_W_signs 103
uncond_moments 104
update_numtols 106
usamon 107
usamone 108
Wald_test 109

Index **112**

`gmvarkit-package` *gmvarkit: Estimate Gaussian and Student's t Mixture Vector Autoregressive Models*

Description

`gmvarkit` is a package for reduced form and structural Gaussian mixture vector autoregressive (GMVAR), Student's t Mixture Vector Autoregressive (StMVAR), or Gaussian and Student's t Mixture Vector Autoregressive (G-StMVAR) model analysis. It provides functions for unconstrained and constrained maximum likelihood estimation of the model parameters, quantile residuals tests, graphical diagnostics, estimation of generalized impulse response function, estimation of generalized forecast error variance decomposition, simulation from GMVAR processes, forecasting, and more.

The readme file is a good place to start and the vignette might be useful too.

Author(s)

Maintainer: Savi Virolainen <savi.virolainen@gmail.com> (ORCID)

Authors:

- Savi Virolainen <savi.virolainen@gmail.com> (ORCID)

See Also

Useful links:

- Report bugs at https://github.com/saviviro/gmvarkit/issues

add_data *Add data to an object of class 'gsmvar' defining a GMVAR, StMVAR, or G-StMVAR model*

Description

add_data adds or updates data to object of class 'gsmvar' that defines a GMVAR, StMVAR, or G-StMVAR model. Also calculates mixing weights and quantile residuals accordingly.

Usage

```
add_data(data, gsmvar, calc_cond_moments = TRUE, calc_std_errors = FALSE)
```

Arguments

data	a matrix or class 'ts' object with d>1 columns. Each column is taken to represent a univariate time series. NA values are not supported.
gsmvar	an object of class 'gsmvar', typically created with fitGSMVAR or GSMVAR.
calc_cond_moments	should conditional means and covariance matrices should be calculated? Default is TRUE if the model contains data and FALSE otherwise.
calc_std_errors	should approximate standard errors be calculated?

Value

Returns an object of class 'gsmvar' defining the specified GSMVAR, StMVAR, or G-StMVAR model with the data added to the model. If the object already contained data, the data will be updated.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

fitGSMVAR, GSMVAR, iterate_more, update_numtols

Examples

```
# GMVAR(1, 2), d=2 model:
params12 <- c(0.55, 0.112, 0.344, 0.055, -0.009, 0.718, 0.319, 0.005,
  0.03, 0.619, 0.173, 0.255, 0.017, -0.136, 0.858, 1.185, -0.012,
  0.136, 0.674)
mod12 <- GSMVAR(p=1, M=2, d=2, params=params12)
mod12

mod12_2 <- add_data(gdpdef, mod12)
mod12_2

# StMVAR(1, 2), d=2 model:
mod12t <- GSMVAR(p=1, M=2, d=2, params=c(params12, 10, 12), model="StMVAR")
mod12t
mod12t_2 <- add_data(gdpdef, mod12t)
mod12t_2

# Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
params22s <- c(0.36, 0.121, 0.484, 0.072, 0.223, 0.059, -0.151, 0.395,
  0.406, -0.005, 0.083, 0.299, 0.218, 0.02, -0.119, 0.722, 0.093, 0.032,
  0.044, 0.191, 0.057, 0.172, -0.46, 0.016, 3.518, 5.154, 0.58)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
mod22s <- GSMVAR(p=2, M=2, d=2, params=params22s, structural_pars=list(W=W_22))
mod22s

mod22s_2 <- add_data(gdpdef, mod22s)
mod22s_2
```

---

alt_gsmvar *Construct a GMVAR, StMVAR, or G-StMVAR model based on results from an arbitrary estimation round of* `fitGSMVAR`

Description

`alt_gsmvar` constructs a GMVAR, StMVAR, or G-StMVAR model based on results from an arbitrary estimation round of `fitGSMVAR`.

Usage

```
alt_gsmvar(
  gsmvar,
  which_round = 1,
  which_largest,
  calc_cond_moments = TRUE,
  calc_std_errors = TRUE
)
```

Arguments

gsmvar	an object of class 'gsmvar', typically created with fitGSMVAR or GSMVAR.
which_round	based on which estimation round should the model be constructed? An integer value in 1,...,ncalls.
which_largest	based on estimation round with which largest log-likelihood should the model be constructed? An integer value in 1,...,ncalls. For example, which_largest=2 would take the second largest log-likelihood and construct the model based on the corresponding estimates. If used, then which_round is ignored.
calc_cond_moments	should conditional means and covariance matrices should be calculated? Default is TRUE if the model contains data and FALSE otherwise.
calc_std_errors	should approximate standard errors be calculated?

Details

It's sometimes useful to examine other estimates than the one with the highest log-likelihood. This function is wrapper around GSMVAR that picks the correct estimates from an object returned by fitGSMVAR.

Value

Returns an object of class 'gsmvar' defining the specified reduced form or structural GMVAR, StMVAR, or G-StMVAR model. Can be used to work with other functions provided in gmvarkit.

Note that the first autocovariance/correlation matrix in $uncond_moments is for the lag zero, the second one for the lag one, etc.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Kalliovirta L. and Saikkonen P. 2010. Reliable Residuals for Multivariate Nonlinear Time Series Models. *Unpublished Revision of HECER Discussion Paper No. 247*.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

fitGSMVAR, GSMVAR, iterate_more, update_numtols

Examples

```
# GMVAR(1,2) model
fit12 <- fitGSMVAR(gdpdef, p=1, M=2, ncalls=2, seeds=4:5)
fit12
```

```
fit12_2 <- alt_gsmvar(fit12, which_largest=2)
fit12_2
```

---

calc_gradient *Calculate gradient or Hessian matrix*

Description

calc_gradient or calc_hessian calculates the gradient or Hessian matrix of the given function at the given point using central difference numerical approximation. get_gradient or get_hessian calculates the gradient or Hessian matrix of the log-likelihood function at the parameter estimates of a class 'gsmvar' object. get_soc returns eigenvalues of the Hessian matrix, and get_foc is the same as get_gradient but named conveniently.

Usage

```
calc_gradient(x, fn, h = 6e-06, varying_h = NULL, ...)

calc_hessian(x, fn, h = 6e-06, varying_h = NULL, ...)

get_gradient(gsmvar, custom_h = NULL)

get_hessian(gsmvar, custom_h = NULL)

get_foc(gsmvar, custom_h = NULL)

get_soc(gsmvar, custom_h = NULL)
```

Arguments

x	a numeric vector specifying the point where the gradient or Hessian should be calculated.
fn	a function that takes in argument x as the **first** argument.
h	difference used to approximate the derivatives.
varying_h	a numeric vector with the same length as x specifying the difference h for each dimension separately. If NULL (default), then the difference given as parameter h will be used for all dimensions.
...	other arguments passed to fn
gsmvar	an object of class 'gsmvar', typically created with fitGSMVAR or GSMVAR.
custom_h	same as varying_h except thaqt if NULL (default), then the difference used for differentiating overly large degrees of freedom parameters is adjusted to avoid numerical problems, and the difference 6e-6 is used for the other parameters.

Details

In particular, the functions `get_foc` and `get_soc` can be used to check whether the found estimates denote a (local) maximum point, a saddle point, or something else. Note that profile log-likelihood functions can be conveniently plotted with the function `profile_logliks`.

Value

Gradient functions return numerical approximation of the gradient and Hessian functions return numerical approximation of the Hessian. `get_soc` returns eigenvalues of the Hessian matrix.

Warning

No argument checks!

See Also

`profile_logliks`

Examples

```
# Simple function
foo <- function(x) x^2 + x
calc_gradient(x=1, fn=foo)
calc_gradient(x=-0.5, fn=foo)

# More complicated function
foo <- function(x, a, b) a*x[1]^2 - b*x[2]^2
calc_gradient(x=c(1, 2), fn=foo, a=0.3, b=0.1)


# GMVAR(1,2), d=2 model:
params12 <- c(0.55, 0.112, 0.344, 0.055, -0.009, 0.718, 0.319, 0.005,
 0.03, 0.619, 0.173, 0.255, 0.017, -0.136, 0.858, 1.185, -0.012,
 0.136, 0.674)
mod12 <- GSMVAR(gdpdef, p=1, M=2, params=params12)
get_gradient(mod12)
get_hessian(mod12)
get_soc(mod12)
```

---

check_parameters *Check that the given parameter vector satisfies the model assumptions*

Description

`check_parameters` checks whether the given parameter vector satisfies the model assumptions. Does NOT consider the identifiability condition!

Usage

```
check_parameters(
  p,
  M,
  d,
  params,
  model = c("GMVAR", "StMVAR", "G-StMVAR"),
  parametrization = c("intercept", "mean"),
  constraints = NULL,
  same_means = NULL,
  weight_constraints = NULL,
  structural_pars = NULL,
  stat_tol = 0.001,
  posdef_tol = 1e-08,
  df_tol = 1e-08
)
```

Arguments

p	a positive integer specifying the autoregressive order of the model.
M	**For GMVAR and StMVAR models:** a positive integer specifying the number of mixture components. **For G-StMVAR models:** a size (2×1) integer vector specifying the number of *GMVAR type* components M1 in the first element and *StMVAR type* components M2 in the second element. The total number of mixture components is M=M1+M2.
d	the number of time series in the system.
params	a real valued vector specifying the parameter values. **For unconstrained models:** Should be size $((M(pd^2+d+d(d+1)/2+2)-M1-1)x1)$ and have the form θ=(υ_1, ...,υ_M, $\alpha_1, ..., \alpha_{M-1}$,$\nu$), where • $\upsilon_m = (\phi_{m,0}$,$\phi_m, \sigma_m)$ • ϕ_m= $(vec(A_{m,1}), ..., vec(A_{m,p})$ • and $\sigma_m = vech(\Omega_m)$, m=1,...,M, • ν= $(\nu_{M1+1}, ..., \nu_M)$ • $M1$ is the number of GMVAR type regimes. **For constrained models:** Should be size $((M(d+d(d+1)/2+2)+q-M1-1)x1)$ and have the form θ= $(\phi_{1,0}, ..., \phi_{M,0}$,$\psi$, $\sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1}$,$\nu$), where • ψ $(qx1)$ satisfies $(\phi_1, ..., \phi_M) = C\psi$ where C is a (Mpd^2xq) constraint matrix. **For same_means models:** Should have the form θ= (μ,ψ, $\sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1}$,$\nu$), where • μ= $(\mu_1, ..., \mu_g)$ where μ_i is the mean parameter for group i and g is the number of groups. • If AR constraints are employed, ψ is as for constrained models, and if AR constraints are not employed, ψ= (ϕ_1, ...,ϕ_M).

For models with weight_constraints: Drop $\alpha_1, ..., \alpha_{M-1}$ from the parameter vector.

For structural models: Reduced form models can be directly used as recursively identified structural models. If the structural model is identified by conditional heteroskedasticity, the parameter vector should have the form $\theta = (\phi_{1,0}, ..., \phi_{M,0}, \phi_1, ..., \phi_M, vec(W), \lambda_2, ..., \lambda_M, \alpha_1, ..., \alpha_{M-1}, \nu)$, where

- $\lambda_m = (\lambda_{m1}, ..., \lambda_{md})$ contains the eigenvalues of the mth mixture component.

If AR parameters are constrained: Replace $\phi_1, ..., \phi_M$ with $\psi\ (qx1)$ that satisfies $(\phi_1, ..., \phi_M) = C\psi$, as above.

If same_means: Replace $(\phi_{1,0}, ..., \phi_{M,0})$ with $(\mu_1, ..., \mu_g)$, as above.

If W is constrained: Remove the zeros from $vec(W)$ and make sure the other entries satisfy the sign constraints.

If λ_{mi} are constrained via `C_lambda`: Replace $\lambda_2, ..., \lambda_M$ with $\gamma\ (rx1)$ that satisfies $(\lambda_2\ , ..., \lambda_M) = C_\lambda \gamma$ where C_λ is a $(d(M-1)xr)$ constraint matrix.

If λ_{mi} are constrained via `fixed_lambdas`: Drop $\lambda_2, ..., \lambda_M$ from the parameter vector.

Above, $\phi_{m,0}$ is the intercept parameter, $A_{m,i}$ denotes the ith coefficient matrix of the mth mixture component, Ω_m denotes the error term covariance matrix of the m:th mixture component, and α_m is the mixing weight parameter. The W and λ_{mi} are structural parameters replacing the error term covariance matrices (see Virolainen, 2022). If $M = 1$, α_m and λ_{mi} are dropped. If `parametrization=="mean"`, just replace each $\phi_{m,0}$ with regimewise mean μ_m. $vec()$ is vectorization operator that stacks columns of a given matrix into a vector. $vech()$ stacks columns of a given matrix from the principal diagonal downwards (including elements on the diagonal) into a vector.

In the **GMVAR model**, $M1 = M$ and ν is dropped from the parameter vector. In the **StMVAR** model, $M1 = 0$. In the **G-StMVAR** model, the first `M1` regimes are *GMVAR type* and the rest `M2` regimes are *StMVAR type*. In **StMVAR** and **G-StMVAR** models, the degrees of freedom parameters in ν should be strictly larger than two.

The notation is similar to the cited literature.

`model` is "GMVAR", "StMVAR", or "G-StMVAR" model considered? In the G-StMVAR model, the first `M1` components are GMVAR type and the rest `M2` components are StMVAR type.

`parametrization`

`"intercept"` or `"mean"` determining whether the model is parametrized with intercept parameters $\phi_{m,0}$ or regime means μ_m, m=1,...,M.

`constraints` a size $(Mpd^2 \times q)$ constraint matrix C specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, ..., \phi_M) = C\psi$, where $\phi_m = (vec(A_{m,1}), ..., vec(A_{m,p}))(pd^2 x1), m = 1, ..., M$, contains the coefficient matrices and $\psi\ (qx1)$ contains the related parameters. For example, to restrict the AR-parameters to be the same for all regimes, set C= `[I:...:I]'` $(Mpd^2 \times pd^2)$ where `I = diag(p*d^2)`. Ignore (or set to `NULL`) if linear constraints should **not** be employed.

same_means Restrict the mean parameters of some regimes to be the same? Provide a list of numeric vectors such that each numeric vector contains the regimes that should share the common mean parameters. For instance, if `M=3`, the argument `list(1, 2:3)` restricts the mean parameters of the second and third regime to be the same but the first regime has freely estimated (unconditional) mean. Ignore or set to `NULL` if mean parameters should not be restricted to be the same among any regimes. **This constraint is available only for mean parametrized models; that is, when** `parametrization="mean"`**.**

weight_constraints
a numeric vector of length $M-1$ specifying fixed parameter values for the mixing weight parameters $\alpha_m,\ m = 1, ..., M-1$. Each element should be strictly between zero and one, and the sum of all the elements should be strictly less than one.

structural_pars
If `NULL` a reduced form model is considered. Reduced models can be used directly as recursively identified structural models. For a structural model identified by conditional heteroskedasticity, should be a list containing at least the first one of the following elements:

- `W` - a (dxd) matrix with its entries imposing constraints on W: `NA` indicating that the element is unconstrained, a positive value indicating strict positive sign constraint, a negative value indicating strict negative sign constraint, and zero indicating that the element is constrained to zero.
- `C_lambda` - a $(d(M-1) \times r)$ constraint matrix that satisfies $(\lambda_2, ..., \lambda_M) = C_\lambda \gamma$ where γ is the new $(r \times 1)$ parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of `C_lambda` must be either **positive** or **zero**. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.
- `fixed_lambdas` - a length $d(M-1)$ numeric vector $(\lambda_2, ..., \lambda_M)$ with elements strictly larger than zero specifying the fixed parameter values for the parameters λ_{mi} should be constrained to. This constraint is alternative `C_lambda`. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.

See Virolainen (forthcoming) for the conditions required to identify the shocks and for the B-matrix as well (it is W times a time-varying diagonal matrix with positive diagonal entries).

stat_tol numerical tolerance for stationarity of the AR parameters: if the "bold A" matrix of any regime has eigenvalues larger that `1 - stat_tol` the model is classified as non-stationary. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.

posdef_tol numerical tolerance for positive definiteness of the error term covariance matrices: if the error term covariance matrix of any regime has eigenvalues smaller than this, the model is classified as not satisfying positive definiteness assumption. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.

df_tol the parameter vector is considered to be outside the parameter space if all degrees of freedom parameters are not larger than `2 + df_tol`.

Value

Throws an informative error if there is something wrong with the parameter vector.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. (forthcoming). A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*.
- Virolainen S. 2022. Gaussian and Student's t mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks in the Euro area. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

Examples

```
## Not run:
# These examples will cause an informative error

# GMVAR(1, 1), d=2 model:
params11 <- c(1.07, 127.71, 0.99, 0.00, -0.01, 1.00, 4.05,
  2.22, 8.87)
check_parameters(p=1, M=1, d=2, params=params11)

# GMVAR(2, 2), d=2 model:
params22 <- c(1.39, -0.77, 1.31, 0.14, 0.09, 1.29, -0.39,
 -0.07, -0.11, -0.28, 0.92, -0.03, 4.84, 1.01, 5.93, 1.25,
  0.08, -0.04, 1.27, -0.27, -0.07, 0.03, -0.31, 5.85, 10.57,
  9.84, 0.74)
check_parameters(p=2, M=2, d=2, params=params22)

# GMVAR(2, 2), d=2 model with AR-parameters restricted to be
# the same for both regimes:
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
params222c <- c(1.03, 2.36, 1.79, 3.00, 1.25, 0.06,0.04,
 1.34, -0.29, -0.08, -0.05, -0.36, 0.93, -0.15, 5.20,
 5.88, 3.56, 9.80, 1.37)
check_parameters(p=2, M=2, d=2, params=params22c, constraints=C_mat)

# Structural GMVAR(2, 2), d=2 model identified with sign-constraints
# (no error):
params22s <- c(1.03, 2.36, 1.79, 3, 1.25, 0.06, 0.04, 1.34, -0.29,
 -0.08, -0.05, -0.36, 1.2, 0.05, 0.05, 1.3, -0.3, -0.1, -0.05, -0.4,
  0.89, 0.72, -0.37, 2.16, 7.16, 1.3, 0.37)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
check_parameters(p=2, M=2, d=2, params=params22s,
 structural_pars=list(W=W_22))

## End(Not run)
```

cond_moments *Compute conditional moments of a GMVAR, StMVAR, or G-StMVAR model*

Description

cond_moments compute conditional regimewise means, conditional means, and conditional covariance matrices of a GMVAR, StMVAR, or G-StMVAR model.

Usage

```
cond_moments(
  data,
  p,
  M,
  params,
  model = c("GMVAR", "StMVAR", "G-StMVAR"),
  parametrization = c("intercept", "mean"),
  constraints = NULL,
  same_means = NULL,
  weight_constraints = NULL,
  structural_pars = NULL,
  to_return = c("regime_cmeans", "regime_ccovs", "total_cmeans", "total_ccovs",
    "arch_scalars"),
  minval = NA,
  stat_tol = 0.001,
  posdef_tol = 1e-08,
  df_tol = 1e-08
)
```

Arguments

data	a matrix or class 'ts' object with d>1 columns. Each column is taken to represent a univariate time series. NA values are not supported.
p	a positive integer specifying the autoregressive order of the model.
M	**For GMVAR and StMVAR models:** a positive integer specifying the number of mixture components. **For G-StMVAR models:** a size (2×1) integer vector specifying the number of *GMVAR type* components M1 in the first element and *StMVAR type* components M2 in the second element. The total number of mixture components is M=M1+M2.
params	a real valued vector specifying the parameter values. **For unconstrained models:** Should be size $((M(pd^2 + d + d(d+1)/2 + 2) - M1 - 1)x1)$ and have the form θ=(υ_1, ...,υ_M, α_1, ..., α_{M-1},ν), where • $\upsilon_m = (\phi_{m,0}, \phi_m, \sigma_m)$

- ϕ_m= $(vec(A_{m,1}), ..., vec(A_{m,p})$
- and $\sigma_m = vech(\Omega_m)$, m=1,...,M,
- ν= $(\nu_{M1+1}, ..., \nu_M)$
- $M1$ is the number of GMVAR type regimes.

For constrained models: Should be size $((M(d+d(d+1)/2+2)+q-M1-1)x1)$ and have the form θ= $(\phi_{1,0}, ..., \phi_{M,0}$,$\psi$, $\sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1}$,$\nu$), where

- ψ $(qx1)$ satisfies $(\phi_1, ..., \phi_M) = C\psi$ where C is a (Mpd^2xq) constraint matrix.

For same_means models: Should have the form θ= $(\mu$,ψ, $\sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1}$,$\nu$), where

- μ= $(\mu_1, ..., \mu_g)$ where μ_i is the mean parameter for group i and g is the number of groups.
- If AR constraints are employed, ψ is as for constrained models, and if AR constraints are not employed, ψ= $(\phi_1, ...,\phi_M)$.

For models with weight_constraints: Drop $\alpha_1, ..., \alpha_{M-1}$ from the parameter vector.

For structural models: Reduced form models can be directly used as recursively identified structural models. If the structural model is identified by conditional heteroskedasticity, the parameter vector should have the form θ= $(\phi_{1,0}, ..., \phi_{M,0}$,$\phi_1, ...,\phi_M, vec(W)$,$\lambda_2, ...,\lambda_M, \alpha_1, ..., \alpha_{M-1}$,$\nu$), where

- λ_m = $(\lambda_{m1}, ..., \lambda_{md})$ contains the eigenvalues of the mth mixture component.

If AR parameters are constrained: Replace $\phi_1, ..., \phi_M$ with ψ $(qx1)$ that satisfies $(\phi_1, ..., \phi_M) = C\psi$, as above.

If same_means: Replace $(\phi_{1,0}, ..., \phi_{M,0})$ with $(\mu_1, ..., \mu_g)$, as above.

If W is constrained: Remove the zeros from $vec(W)$ and make sure the other entries satisfy the sign constraints.

If λ_{mi} are constrained via `C_lambda`: Replace $\lambda_2, ..., \lambda_M$ with γ $(rx1)$ that satisfies $(\lambda_2 , ..., \lambda_M) = C_\lambda\gamma$ where C_λ is a $(d(M-1)xr)$ constraint matrix.

If λ_{mi} are constrained via `fixed_lambdas`: Drop $\lambda_2, ..., \lambda_M$ from the parameter vector.

Above, $\phi_{m,0}$ is the intercept parameter, $A_{m,i}$ denotes the ith coefficient matrix of the mth mixture component, Ω_m denotes the error term covariance matrix of the m:th mixture component, and α_m is the mixing weight parameter. The W and λ_{mi} are structural parameters replacing the error term covariance matrices (see Virolainen, 2022). If $M = 1$, α_m and λ_{mi} are dropped. If `parametrization=="mean"`, just replace each $\phi_{m,0}$ with regimewise mean μ_m. $vec()$ is vectorization operator that stacks columns of a given matrix into a vector. $vech()$ stacks columns of a given matrix from the principal diagonal downwards (including elements on the diagonal) into a vector.

In the **GMVAR model**, $M1 = M$ and ν is dropped from the parameter vector. In the **StMVAR** model, $M1 = 0$. In the **G-StMVAR** model, the first `M1` regimes are *GMVAR type* and the rest `M2` regimes are *StMVAR type*. In **StMVAR** and

G-StMVAR models, the degrees of freedom parameters in ν should be strictly larger than two.

The notation is similar to the cited literature.

`model` is "GMVAR", "StMVAR", or "G-StMVAR" model considered? In the G-StMVAR model, the first `M1` components are GMVAR type and the rest `M2` components are StMVAR type.

`parametrization`

`"intercept"` or `"mean"` determining whether the model is parametrized with intercept parameters $\phi_{m,0}$ or regime means μ_m, m=1,...,M.

`constraints` a size $(Mpd^2 \times q)$ constraint matrix C specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, ..., \phi_M) = C\psi$, where $\phi_m = (vec(A_{m,1}), ..., vec(A_{m,p})(pd^2 x 1), m = 1, ..., M$, contains the coefficient matrices and ψ $(qx1)$ contains the related parameters. For example, to restrict the AR-parameters to be the same for all regimes, set C= `[I:...:I]'` $(Mpd^2 \times pd^2)$ where `I = diag(p*d^2)`. Ignore (or set to `NULL`) if linear constraints should **not** be employed.

`same_means` Restrict the mean parameters of some regimes to be the same? Provide a list of numeric vectors such that each numeric vector contains the regimes that should share the common mean parameters. For instance, if `M=3`, the argument `list(1, 2:3)` restricts the mean parameters of the second and third regime to be the same but the first regime has freely estimated (unconditional) mean. Ignore or set to `NULL` if mean parameters should not be restricted to be the same among any regimes. **This constraint is available only for mean parametrized models; that is, when** `parametrization="mean"`**.**

`weight_constraints`

a numeric vector of length $M - 1$ specifying fixed parameter values for the mixing weight parameters α_m, $m = 1, ..., M - 1$. Each element should be strictly between zero and one, and the sum of all the elements should be strictly less than one.

`structural_pars`

If `NULL` a reduced form model is considered. Reduced models can be used directly as recursively identified structural models. For a structural model identified by conditional heteroskedasticity, should be a list containing at least the first one of the following elements:

- `W` - a (dxd) matrix with its entries imposing constraints on W: `NA` indicating that the element is unconstrained, a positive value indicating strict positive sign constraint, a negative value indicating strict negative sign constraint, and zero indicating that the element is constrained to zero.
- `C_lambda` - a $(d(M-1) \times r)$ constraint matrix that satisfies $(\lambda_2, ..., \lambda_M) = C_\lambda \gamma$ where γ is the new $(r \times 1)$ parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of `C_lambda` must be either **positive** or **zero**. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.
- `fixed_lambdas` - a length $d(M - 1)$ numeric vector $(\lambda_2, ..., \lambda_M)$ with elements strictly larger than zero specifying the fixed parameter values for the parameters λ_{mi} should be constrained to. This constraint is alternative

C_lambda. Ignore (or set to NULL) if the eigenvalues λ_{mi} should not be constrained.

See Virolainen (forthcoming) for the conditions required to identify the shocks and for the B-matrix as well (it is W times a time-varying diagonal matrix with positive diagonal entries).

to_return — should the regimewise conditional means, total conditional means, or total conditional covariance matrices be returned?

minval — the value that will be returned if the parameter vector does not lie in the parameter space (excluding the identification condition).

stat_tol — numerical tolerance for stationarity of the AR parameters: if the "bold A" matrix of any regime has eigenvalues larger that 1 - stat_tol the model is classified as non-stationary. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.

posdef_tol — numerical tolerance for positive definiteness of the error term covariance matrices: if the error term covariance matrix of any regime has eigenvalues smaller than this, the model is classified as not satisfying positive definiteness assumption. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.

df_tol — the parameter vector is considered to be outside the parameter space if all degrees of freedom parameters are not larger than 2 + df_tol.

Details

The first p values are used as the initial values, and by conditional we mean conditioning on the past. Formulas for the conditional means and covariance matrices are given in equations (3) and (4) of KMS (2016).

Value

If to_return=="regime_cmeans": an [T-p, d, M] array containing the regimewise conditional means (the first p values are used as the initial values).

If to_return=="regime_ccovs": an [d, d, T-p, M] array containing the regimewise conditional covariance matrices (the first p values are used as the initial values). The index [, , t, m] gives the time t conditional covariance matrix for the regime m.

If to_return=="total_cmeans": a [T-p, d] matrix containing the conditional means of the process (the first p values are used as the initial values).

If to_return=="total_ccov": an [d, d, T-p] array containing the conditional covariance matrices of the process (the first p values are used as the initial values).

If to_return=="arch_scalars": a [T-p, M] matrix containing the regimewise arch scalars multiplying error term covariance matrix in the conditional covariance matrix of the regime. For GMVAR type regimes, these are all ones (the first p values are used as the initial values).

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.

- Lütkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.
- McElroy T. 2017. Computation of vector ARMA autocovariances. *Statistics and Probability Letters*, **124**, 92-96.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

Other moment functions: `get_regime_autocovs()`, `get_regime_means()`, `uncond_moments()`

Examples

```
# GMVAR(2, 2), d=2 model;
params22 <- c(0.36, 0.121, 0.223, 0.059, -0.151, 0.395, 0.406, -0.005,
 0.083, 0.299, 0.215, 0.002, 0.03, 0.484, 0.072, 0.218, 0.02, -0.119,
 0.722, 0.093, 0.032, 0.044, 0.191, 1.101, -0.004, 0.105, 0.58)
cond_moments(data=gdpdef, p=2, M=2, params=params22, to_return="regime_cmeans")
cond_moments(data=gdpdef, p=2, M=2, params=params22, to_return="total_cmeans")
cond_moments(data=gdpdef, p=2, M=2, params=params22, to_return="total_ccovs")
```

---

cond_moment_plot *Conditional mean or variance plot for a GMVAR, StMVAR, or G-StMVAR model*

Description

`cond_moment_plot` plots the one-step in-sample conditional means/variances of the model along with the individual time series contained in the model (e.g. the time series the model was fitted to). Also plots the regimewise conditional means/variances multiplied with mixing weights.

Usage

```
cond_moment_plot(
  gsmvar,
  which_moment = c("mean", "variance"),
  grid = FALSE,
  ...
)
```

Arguments

`gsmvar`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.
`which_moment`	should conditional means or variances be plotted?
`grid`	add grid to the plots?
`...`	additional paramters passed to `grid(...)` plotting the grid if `grid == TRUE`.

Details

The conditional mean plot works best if the data contains positive values only. `acf` from the package `stats` and the plot method for class `'acf'` objects is employed.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lütkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.
- McElroy T. 2017. Computation of vector ARMA autocovariances. *Statistics and Probability Letters*, **124**, 92-96.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

`profile_logliks`, `fitGSMVAR`, `GSMVAR`, `quantile_residual_tests`, `LR_test`, `Wald_test`, `diagnostic_plot`

Examples

```
# GMVAR(2, 2), d=2 model;
params22 <- c(0.36, 0.121, 0.223, 0.059, -0.151, 0.395, 0.406, -0.005,
 0.083, 0.299, 0.215, 0.002, 0.03, 0.484, 0.072, 0.218, 0.02, -0.119,
 0.722, 0.093, 0.032, 0.044, 0.191, 1.101, -0.004, 0.105, 0.58)
mod22 <- GSMVAR(gdpdef, p=2, M=2, params=params22)
cond_moment_plot(mod22, which_moment="mean")
cond_moment_plot(mod22, which_moment="variance")
cond_moment_plot(mod22, which_moment="mean", grid=TRUE, lty=3)

# G-StMVAR(2, 1, 1), d=2 model:
params22gs <- c(0.697, 0.154, 0.049, 0.374, 0.476, 0.318, -0.645, -0.302,
 -0.222, 0.193, 0.042, -0.013, 0.048, 0.554, 0.033, 0.184, 0.005, -0.186,
  0.683, 0.256, 0.031, 0.026, 0.204, 0.583, -0.002, 0.048, 0.182, 4.334)
mod22gs <- GSMVAR(gdpdef, p=2, M=c(1, 1), params=params22gs, model="G-StMVAR")
cond_moment_plot(mod22gs, which_moment="mean")
cond_moment_plot(mod22gs, which_moment="variance")

#StMVAR(4, 1), d=2 model:
params41t <- c(0.512, -0.002, 0.243, 0.024, -0.088, 0.452, 0.242, 0.011,
  0.093, 0.162, -0.097, 0.033, -0.339, 0.19, 0.091, 0.006, 0.168, 0.101,
  0.516, -0.005, 0.054, 4.417)
mod41t <- GSMVAR(gdpdef, p=4, M=1, params=params41t, model="StMVAR")
cond_moment_plot(mod41t, which_moment="mean")
cond_moment_plot(mod41t, which_moment="variance")
```

`diagnostic_plot` *Quantile residual diagnostic plot for a GMVAR, StMVAR, or G-StMVAR model*

Description

`diagnostic_plot` plots a multivariate quantile residual diagnostic plot for either autocorrelation, conditional heteroskedasticity, or normality, or simply draws the quantile residual time series.

Usage

```
diagnostic_plot(
  gsmvar,
  type = c("all", "series", "ac", "ch", "norm"),
  maxlag = 12,
  wait_time = 4
)
```

Arguments

`gsmvar`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.
`type`	which type of diagnostic plot should be plotted? • `"all"` all below sequentially. • `"series"` the quantile residual time series. • `"ac"` the quantile residual autocorrelation and cross-correlation functions. • `"ch"` the squared quantile residual autocorrelation and cross-correlation functions. • `"norm"` the quantile residual histogram with theoretical standard normal density (dashed line) and standard normal QQ-plots.
`maxlag`	the maximum lag considered in types `"ac"` and `"ch"`.
`wait_time`	if `type == all` how many seconds to wait before showing next figure?

Details

Auto- and cross-correlations (types `"ac"` and `"ch"`) are calculated with the function `acf` from the package `stats` and the plot method for class `'acf'` objects is employed.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Kalliovirta L. and Saikkonen P. 2010. Reliable Residuals for Multivariate Nonlinear Time Series Models. *Unpublished Revision of HECER Discussion Paper No. 247*.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

profile_logliks, fitGSMVAR, GSMVAR, quantile_residual_tests, LR_test, Wald_test, Rao_test, cond_moment_plot, acf, density, predict.gsmvar

Examples

```
# GMVAR(1,2), d=2 model:
params12 <- c(0.55, 0.112, 0.344, 0.055, -0.009, 0.718, 0.319,
 0.005, 0.03, 0.619, 0.173, 0.255, 0.017, -0.136, 0.858, 1.185,
 -0.012, 0.136, 0.674)
mod12 <- GSMVAR(gdpdef, p=1, M=2, params=params12)
diagnostic_plot(mod12, type="series")
diagnostic_plot(mod12, type="ac")

# GMVAR(2,2), d=2 model:
params22 <-  c(0.36, 0.121, 0.223, 0.059, -0.151, 0.395, 0.406,
 -0.005, 0.083, 0.299, 0.215, 0.002, 0.03, 0.484, 0.072, 0.218,
 0.02, -0.119, 0.722, 0.093, 0.032, 0.044, 0.191, 1.101, -0.004,
  0.105, 0.58)
mod22 <- GSMVAR(gdpdef, p=2, M=2, params=params22)
diagnostic_plot(mod22, type="ch")
diagnostic_plot(mod22, type="norm")

# G-StMVAR(2, 1, 1), d=2 model:
params22gs <- c(0.697, 0.154, 0.049, 0.374, 0.476, 0.318, -0.645, -0.302,
 -0.222, 0.193, 0.042, -0.013, 0.048, 0.554, 0.033, 0.184, 0.005, -0.186,
  0.683, 0.256, 0.031, 0.026, 0.204, 0.583, -0.002, 0.048, 0.182, 4.334)
mod22gs <- GSMVAR(gdpdef, p=2, M=c(1, 1), params=params22gs, model="G-StMVAR")
diagnostic_plot(mod22gs, wait_time=0)
```

---

diag_Omegas *Simultaneously diagonalize two covariance matrices*

Description

diag_Omegas Simultaneously diagonalizes two covariance matrices using eigenvalue decomposition.

Usage

```
diag_Omegas(Omega1, Omega2)
```

Arguments

Omega1	a positive definite (dxd) covariance matrix $(d > 1)$
Omega2	another positive definite (dxd) covariance matrix

Details

See the return value and Muirhead (1982), Theorem A9.9 for details.

Value

Returns a length $d^2 + d$ vector where the first d^2 elements are $vec(W)$ with the columns of W being (specific) eigenvectors of the matrix $\Omega_2\Omega_1^{-1}$ and the rest d elements are the corresponding eigenvalues "lambdas". The result satisfies $WW' = Omega1$ and $Wdiag(lambdas)W' = Omega2$.

If `Omega2` is not supplied, returns a vectorized symmetric (and pos. def.) square root matrix of `Omega1`.

Warning

No argument checks! Does not work with dimension $d = 1$!

References

- Muirhead R.J. 1982. Aspects of Multivariate Statistical Theory, *Wiley*.

Examples

```
d <- 2
W0 <- matrix(1:(d^2), nrow=2)
lambdas0 <- 1:d
(Omg1 <- W0%*%t(W0))
(Omg2 <- W0%*%diag(lambdas0)%*%t(W0))
res <- diag_Omegas(Omg1, Omg2)
W <- matrix(res[1:(d^2)], nrow=d, byrow=FALSE)
tcrossprod(W) # == Omg1
lambdas <- res[(d^2 + 1):(d^2 + d)]
W%*%diag(lambdas)%*%t(W) # == Omg2
```

---

estimate_sgsmvar *Maximum likelihood estimation of a structural GMVAR, StMVAR, or G-StMVAR model with preliminary estimates*

Description

`estimate_gsmvar` uses a genetic algorithm and variable metric algorithm to estimate the parameters of a structural GMVAR, StMVAR, or G-StMVAR model with the method of maximum likelihood and preliminary estimates from (typically identified) reduced form or structural GSMVAR model.

Usage

```
estimate_sgsmvar(
  gsmvar,
  new_W,
  ncalls = 16,
  ncores = 2,
  maxit = 1000,
  seeds = NULL
)
```

Arguments

gsmvar	an object of class 'gsmvar', typically created with fitGSMVAR or GSMVAR.
new_W	What should be the constraints on the W-matrix (or equally B-matrix)? Provide a (dxd) matrix (where d is the number of time series in the system) expressing the constraints such that NA signifies that the element is not constrained, strictly positive real number signifies strict positive sign constraint, strictly negative real number signified strict negative sign constraints, and zero signifies a zero constraints.
ncalls	the number of estimation rounds that should be performed.
ncores	the number of CPU cores to be used in numerical differentiation. Multiple cores are not supported on Windows, though.
maxit	the maximum number of iterations in the variable metric algorithm.
seeds	a length ncalls vector containing the random number generator seed for each call to the genetic algorithm, or NULL for not initializing the seed. Exists for creating reproducible results.

Details

The purpose of estimate_gsmvar is to provide a convenient tool to estimate (typically over)identified structural GSMVAR models when preliminary estimates are available from a fitted reduced form or structural GMVAR model. Often one estimates a two-regime reduced form model and then uses the function gsmvar_to_sgsmvar to obtain the corresponding, statistically identified structural model. After obtaining the statistically identified structural model, overidentifying constraints may be placed the W-matrix (or equally B-matrix). This function makes imposing the overidentifying constraints and estimating the overidentified structural model convenient. **Reduced form models can be directly used as lower-triangular Cholesky identified SVARs without having to estimate a structural model separately.**

Note that the surface of the log-likelihood function is extremely multimodal, and this function is designed to only explore the neighbourhood of the preliminary estimates, so it finds its way reliably to the correct MLE only the preliminary estimates are close it in the first place. Use the function directly fitGSMVAR for a more thorough search of the parameter space, if necessary. This function calls fitGSMVAR by construction an initial population to the genetic algorithm from a preliminary guess of the new estimates. The smart mutations are set to begin from the first generation.

In order to the impose the constraints you wish, it might be useful to first run the model through the functioons `reorder_W_columns` and `swap_W_signs`. **Particularly check that the sign constraints are readily satisfied. If not, the estimated solution might not be correct MLE.**

`estimate_sgsmvar` can also be used to estimate models that are not identified, i.e., one regime models. If it supplied with a reduced form model, it will first apply the function `gsmvar_to_sgsmvar`, then impose the constraints and finally estimate the model.

Value

Returns an object of class `'gsmvar'` defining the estimated GMVAR, StMVAR, or G-StMVAR model.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Kalliovirta L. and Saikkonen P. 2010. Reliable Residuals for Multivariate Nonlinear Time Series Models. *Unpublished Revision of HECER Discussion Paper No. 247*.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

`fitGSMVAR`, `GSMVAR`, `optim`, `profile_logliks`, `iterate_more` `gsmvar_to_sgsmvar`

Examples

```
## These are long running examples that use parallel computing!
## Running the below examples takes 30 seconds

# GMVAR(1,2) model
fit12 <- fitGSMVAR(gdpdef, p=1, M=2, ncalls=2, seeds=1:2) # Reduced form
fit12s <- gsmvar_to_sgsmvar(fit12) # Structural
fit12s
# Constrain the lower right element of W (or B-matrix) to zero, and for
# global identification the first elements of each column to strictly positive.
(new_W <- matrix(c(1, NA, 1, 0), nrow=2))
new_fit12s <- estimate_sgsmvar(fit12s, new_W, ncalls=2, ncores=2, seeds=1:2)
new_fit12s # Overidentified model

# Cholesky VAR(1)
fit11 <- fitGSMVAR(gdpdef, p=1, M=1, ncalls=2, seeds=1:2) # Reduced form
(new_W <- matrix(c(1, NA, 0, 1), nrow=2))
new_fit11s <- estimate_sgsmvar(fit11, new_W, ncalls=2, ncores=2, seeds=1:2)
print(new_fit11s, digits=4)
# Also: gsmvar_to_sgsmvar(fit11, cholesky=TRUE) gives Cholesky VAR
```

euromone *Euro area macroeconomic data used in Virolainen (in press)*

Description

A monthly Euro area data covering the period from January 1999 to December 2021 (276 observations) and consisting four variables: cyclical component of log industrial production index, the log-difference of harmonized consumer price index, the log-difference of Brent crude oil prices (Europe), and an interest rate variable. The interest rate variable is the Euro overnight index average rate (EONIA) from January 1999 to October 2008, and after that the Wu and Xia (2016) shadow rate, which is not constrained by the zero lower bound and also quantifies unconventional monetary policy measures. The log-difference of the harmonized consumer price index is multiplied by hundred and the log-difference of oil price by ten. This data is the one that was used in Virolainen (in press).

The cyclical component of the log of industrial production index was obtained by applying the linear projection filter proposed by Hamilton (2018) using the parameter values h=24 and p=12. In order to obtain as accurate estimates as possible, we applied the filter to the full available sample from January 1991 to December 2021 before extracting our sample period from it. package lpirfs (Adämmer, 2021).

Usage

```
euromone
```

Format

A numeric matrix of class `'ts'` with 276 rows and 4 columns with one time series in each column:

First column (IPI): The cyclical component of the log of industrial production index, url is https://sdw.ecb.europa.eu/quickv

Second column (HCPI): The log-difference of harmonized consumer price index, url is https://sdw.ecb.europa.eu/quickview

Third column (OIL): The log-difference of Brent crude oil price (Europe), https://fred.stlouisfed.org/series/MCOILBRENTEU.

Third column (RATE): The EONIA from January 1999 to October 2008 and after that the Wu and Xia (2016) shadow rate, urls are https://sdw.ecb.europa.eu/quickview.do?SERIES_KEY=143.FM.M.U2.EUR.4F.MM.E and https://sites.google.com/view/jingcynthiawu/shadow-rates.

Source

The Federal Reserve Bank of St. Louis database and the Federal Reserve Bank of Atlanta's website

References

- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.
- Wu J. and Xia F. 2016. Measuring the macroeconomic impact of monetary policy at the zero lower bound. *Journal of Money, Credit and Banking*, 48(2-3): 253-291.

`fitGSMVAR` *Two-phase maximum likelihood estimation of a GMVAR, StMVAR, or G-StMVAR model*

Description

`fitGSMVAR` estimates a GMVAR, StMVAR, or G-StMVAR model model in two phases: in the first phase it uses a genetic algorithm to find starting values for a gradient based variable metric algorithm, which it then uses to finalize the estimation in the second phase. Parallel computing is utilized to perform multiple rounds of estimations in parallel.

Usage

```
fitGSMVAR(
  data,
  p,
  M,
  model = c("GMVAR", "StMVAR", "G-StMVAR"),
  conditional = TRUE,
  parametrization = c("intercept", "mean"),
  constraints = NULL,
  same_means = NULL,
  weight_constraints = NULL,
  structural_pars = NULL,
  ncalls = (M + 1)^5,
  ncores = 2,
  maxit = 1000,
  seeds = NULL,
  print_res = TRUE,
  use_parallel = TRUE,
  filter_estimates = TRUE,
  calc_std_errors = TRUE,
  ...
)
```

Arguments

`data` a matrix or class `'ts'` object with `d>1` columns. Each column is taken to represent a univariate time series. `NA` values are not supported.

`p` a positive integer specifying the autoregressive order of the model.

`M` **For GMVAR and StMVAR models:** a positive integer specifying the number of mixture components.

For G-StMVAR models: a size (2×1) integer vector specifying the number of *GMVAR type* components `M1` in the first element and *StMVAR type* components `M2` in the second element. The total number of mixture components is `M=M1+M2`.

`model` is "GMVAR", "StMVAR", or "G-StMVAR" model considered? In the G-StMVAR model, the first `M1` components are GMVAR type and the rest `M2` components are StMVAR type.

`conditional` a logical argument specifying whether the conditional or exact log-likelihood function

`parametrization`

`"intercept"` or `"mean"` determining whether the model is parametrized with intercept parameters $\phi_{m,0}$ or regime means μ_m, m=1,...,M.

`constraints` a size $(Mpd^2 \times q)$ constraint matrix C specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, ..., \phi_M) = C\psi$, where $\phi_m = (vec(A_{m,1}), ..., vec(A_{m,p})(pd^2 x 1), m = 1, ..., M$, contains the coefficient matrices and ψ $(qx1)$ contains the related parameters. For example, to restrict the AR-parameters to be the same for all regimes, set C= `[I:...:I]'` $(Mpd^2 \times pd^2)$ where `I = diag(p*d^2)`. Ignore (or set to `NULL`) if linear constraints should **not** be employed.

`same_means` Restrict the mean parameters of some regimes to be the same? Provide a list of numeric vectors such that each numeric vector contains the regimes that should share the common mean parameters. For instance, if `M=3`, the argument `list(1, 2:3)` restricts the mean parameters of the second and third regime to be the same but the first regime has freely estimated (unconditional) mean. Ignore or set to `NULL` if mean parameters should not be restricted to be the same among any regimes. **This constraint is available only for mean parametrized models; that is, when** `parametrization="mean"`**.**

`weight_constraints`

a numeric vector of length $M - 1$ specifying fixed parameter values for the mixing weight parameters α_m, $m = 1, ..., M - 1$. Each element should be strictly between zero and one, and the sum of all the elements should be strictly less than one.

`structural_pars`

If `NULL` a reduced form model is considered. Reduced models can be used directly as recursively identified structural models. For a structural model identified by conditional heteroskedasticity, should be a list containing at least the first one of the following elements:

- `W` - a (dxd) matrix with its entries imposing constraints on W: `NA` indicating that the element is unconstrained, a positive value indicating strict positive sign constraint, a negative value indicating strict negative sign constraint, and zero indicating that the element is constrained to zero.
- `C_lambda` - a $(d(M-1) \times r)$ constraint matrix that satisfies $(\lambda_2, ..., \lambda_M) = C_\lambda \gamma$ where γ is the new $(r \times 1)$ parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of `C_lambda` must be either **positive** or **zero**. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.
- `fixed_lambdas` - a length $d(M - 1)$ numeric vector $(\lambda_2, ..., \lambda_M)$ with elements strictly larger than zero specifying the fixed parameter values for the parameters λ_{mi} should be constrained to. This constraint is alternative `C_lambda`. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.

See Virolainen (forthcoming) for the conditions required to identify the shocks and for the B-matrix as well (it is W times a time-varying diagonal matrix with positive diagonal entries).

`ncalls` the number of estimation rounds that should be performed.

`ncores` the number CPU cores to be used in parallel computing.

`maxit` the maximum number of iterations in the variable metric algorithm.

`seeds` a length `ncalls` vector containing the random number generator seed for each call to the genetic algorithm, or `NULL` for not initializing the seed. Exists for creating reproducible results.

`print_res` should summaries of estimation results be printed?

`use_parallel` employ parallel computing? If `FALSE`, no progression bar etc will be generated.

`filter_estimates`
should the likely inappropriate estimates be filtered? See details.

`calc_std_errors`
calculate approximate standard errors for the estimates?

`...` additional settings passed to the function `GAfit` employing the genetic algorithm.

Details

If you wish to estimate a structural model without overidentifying constraints that is identified statistically, specify your W matrix is `structural_pars` to be such that it contains the same sign constraints in a single row (e.g. a row of ones) and leave the other elements as `NA`. In this way, the genetic algorithm works the best. The ordering and signs of the columns of the W matrix can be changed afterwards with the functions `reorder_W_columns` and `swap_W_signs`.

Because of complexity and high multimodality of the log-likelihood function, it's **not certain** that the estimation algorithms will end up in the global maximum point. It's expected that most of the estimation rounds will end up in some local maximum or saddle point instead. Therefore, a (sometimes large) number of estimation rounds is required for reliable results. Because of the nature of the model, the estimation may fail especially in the cases where the number of mixture components is chosen too large. **With two regimes and couple hundred observations in a two-dimensional time series, 50 rounds is usually enough. Several hundred estimation rounds often suffices for reliably fitting two-regimes models to 3 or 4 dimensional time series. With more than two regimes and more than couple hundred observations, thousands of estimation rounds (or more) are often required to obtain reliable results.**

The estimation process is computationally heavy and it might take considerably long time for large models with large number of observations. If the iteration limit `maxit` in the variable metric algorithm is reached, one can continue the estimation by iterating more with the function `iterate_more`. Alternatively, one may use the found estimates as starting values for the genetic algorithm and and employ another round of estimation (see `?GAfit` how to set up an initial population with the dot parameters).

If the estimation algorithm fails to create an initial population for the genetic algorithm, it usually helps to scale the individual series so that the AR coefficients (of a VAR model) will be relative small, preferably less than one. Even if one is able to create an initial population, it should be preferred to scale the series so that most of the AR coefficients will not be very large, as the

estimation algorithm works better with relatively small AR coefficients. If needed, another package can be used to fit linear VARs to the series to see which scaling of the series results in relatively small AR coefficients.

The code of the genetic algorithm is mostly based on the description by *Dorsey and Mayer (1995)* but it includes some extra features that were found useful for this particular estimation problem. For instance, the genetic algorithm uses a slightly modified version of the individually adaptive crossover and mutation rates described by *Patnaik and Srinivas (1994)* and employs (50%) fitness inheritance discussed by *Smith, Dike and Stegmann (1995)*.

The gradient based variable metric algorithm used in the second phase is implemented with function `optim` from the package `stats`.

Note that the structural models are even more difficult to estimate than the reduced form models due to the different parametrization of the covariance matrices, so larger number of estimation rounds should be considered. Also, be aware that if the lambda parameters are constrained in any other way than by restricting some of them to be identical, the parameter "lambda_scale" of the genetic algorithm (see `?GAfit`) needs to be carefully adjusted accordingly. **When estimating a structural model that imposes overidentifiying constraints to a time series with $d > 3$, it is highly recommended to create an initial population based on the estimates of a statistically identified model (when $M = 2$). This is because currently obtaining the ML estimate reliably to such a structural model seems difficult in many application.**

Finally, the function fails to calculate approximate standard errors and the parameter estimates are near the border of the parameter space, it might help to use smaller numerical tolerance for the stationarity and positive definiteness conditions. The numerical tolerance of an existing model can be changed with the function `update_numtols`.

Filtering inappropriate estimates: If `filter_estimates == TRUE`, the function will automatically filter out estimates that it deems "inappropriate". That is, estimates that are not likely solutions of interest. Specifically, solutions that incorporate a near-singular error term covariance matrix (any eigenvalue less than 0.002), mixing weights such that they are close to zero for almost all t for at least one regime, or mixing weight parameter estimate close to zero (or one). It also filters out estimates with any modulus "bold A" eigenvalues larger than 0.9985, as the solution is near the boundary of the stationarity region and likely not a local maximum. You can also set `filter_estimates=FALSE` and find the solutions of interest yourself by using the function `alt_gsmvar`.

Value

Returns an object of class `'gsmvar'` defining the estimated (reduced form or structural) GMVAR, StMVAR, or G-StMVAR model. Multivariate quantile residuals (Kalliovirta and Saikkonen 2010) are also computed and included in the returned object. In addition, the returned object contains the estimates and log-likelihood values from all the estimation rounds performed. The estimated parameter vector can be obtained at `gsmvar$params` (and corresponding approximate standard errors at `gsmvar$std_errors`). See `?GSMVAR` for the form of the parameter vector, if needed.

Remark that the first autocovariance/correlation matrix in `$uncond_moments` is for the lag zero, the second one for the lag one, etc.

S3 methods

The following S3 methods are supported for class `'gsmvar'`: `logLik`, `residuals`, `print`, `summary`, `predict`, `simulate`, and `plot`.

References

- Dorsey R. E. and Mayer W. J. 1995. Genetic algorithms for estimation problems with multiple optima, nondifferentiability, and other irregular features. *Journal of Business & Economic Statistics*, **13**, 53-66.
- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Patnaik L.M. and Srinivas M. 1994. Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms. *Transactions on Systems, Man and Cybernetics* **24**, 656-667.
- Smith R.E., Dike B.A., Stegmann S.A. 1995. Fitness inheritance in genetic algorithms. *Proceedings of the 1995 ACM Symposium on Applied Computing*, 345-350.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

`GSMVAR`, `iterate_more`, `stmvar_to_gstmvar`, `predict.gsmvar`, `profile_logliks`, `simulate.gsmvar`, `quantile_residual_tests`, `print_std_errors`, `swap_parametrization`, `get_gradient`, `GIRF`, `GFEVD`, `LR_test`, `Wald_test`, `gsmvar_to_sgsmvar`, `stmvar_to_gstmvar`, `reorder_W_columns`, `swap_W_signs`, `cond_moment_plot`, `update_numtols`

Examples

```
## These are long running examples that use parallel computing!
# Running all the below examples will take approximately 3-4 minutes.

# GMVAR(1,2) model: 10 estimation rounds with seeds set
# for reproducibility
fit12 <- fitGSMVAR(gdpdef, p=1, M=2, ncalls=10, seeds=1:10)
fit12
plot(fit12)
summary(fit12)
print_std_errors(fit12)
profile_logliks(fit12)

# The rest of the examples only use a single estimation round with a given
# seed that produces the MLE to reduce running time of the examples. When
# estimating models for empirical applications, a large number of estimation
# rounds (ncalls = a large number) should be performed to ensure reliability
# of the estimates (see the section details).

# StMVAR(2, 2) model
fit22t <- fitGSMVAR(gdpdef, p=2, M=2, model="StMVAR", ncalls=1, seeds=1)
```

```
fit22t # Overly large degrees of freedom estimate in the 2nd regime!
fit22gs <- stmvar_to_gstmvar(fit22t) # So switch it to GMVAR type!
fit22gs # This is the appropriate G-StMVAR model based on the above StMVAR model.
fit22gss <- gsmvar_to_sgsmvar(fit22gs) # Switch to structural model
fit22gss # This is the implied statistically identified structural model.

# Structural GMVAR(1,2) model identified with sign
# constraints.
W_122 <- matrix(c(1, 1, -1, 1), nrow=2)
fit12s <- fitGSMVAR(gdpdef, p=1, M=2, structural_pars=list(W=W_122),
  ncalls=1, seeds=1)
fit12s
# A statistically identified structural model can also be obtained with
# gsmvar_to_sgsmvar(fit12)


# GMVAR(2,2) model with autoregressive parameters restricted
# to be the same for both regimes
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
fit22c <- fitGSMVAR(gdpdef, p=2, M=2, constraints=C_mat, ncalls=1, seeds=1)
fit22c

# G-StMVAR(2, 1, 1) model with autoregressive parameters and unconditional means restricted
# to be the same for both regimes:
fit22gscm <- fitGSMVAR(gdpdef, p=2, M=c(1, 1), model="G-StMVAR", constraints=C_mat,
   parametrization="mean", same_means=list(1:2), ncalls=1, seeds=1)

# GMVAR(2,2) model with autoregressive parameters restricted
# to be the same for both regimes and non-diagonal elements
# the coefficient matrices constrained to zero.
tmp <- matrix(c(1, rep(0, 10), 1, rep(0, 8), 1, rep(0, 10), 1),
 nrow=2*2^2, byrow=FALSE)
C_mat2 <- rbind(tmp, tmp)
fit22c2 <- fitGSMVAR(gdpdef, p=2, M=2, constraints=C_mat2, ncalls=1,
  seeds=1)
fit22c2
```

`GAfit` *Genetic algorithm for preliminary estimation of a GMVAR, StMVAR, or G-StMVAR model*

Description

`GAfit` estimates the specified GMVAR, StMVAR, or G-StMVAR model using a genetic algorithm. It's designed to find starting values for gradient based methods.

Usage

```
GAfit(
```

```
  data,
  p,
  M,
  model = c("GMVAR", "StMVAR", "G-StMVAR"),
  conditional = TRUE,
  parametrization = c("intercept", "mean"),
  constraints = NULL,
  same_means = NULL,
  weight_constraints = NULL,
  structural_pars = NULL,
  ngen = 200,
  popsize,
  smart_mu = min(100, ceiling(0.5 * ngen)),
  initpop = NULL,
  mu_scale,
  mu_scale2,
  omega_scale,
  W_scale,
  lambda_scale,
  ar_scale = 0.2,
  upper_ar_scale = 1,
  ar_scale2 = 1,
  regime_force_scale = 1,
  red_criteria = c(0.05, 0.01),
  pre_smart_mu_prob = 0,
  to_return = c("alt_ind", "best_ind"),
  minval,
  seed = NULL
)
```

Arguments

`data`	a matrix or class `'ts'` object with `d>1` columns. Each column is taken to represent a univariate time series. `NA` values are not supported.
`p`	a positive integer specifying the autoregressive order of the model.
`M`	**For GMVAR and StMVAR models:** a positive integer specifying the number of mixture components. **For G-StMVAR models:** a size (2×1) integer vector specifying the number of *GMVAR type* components `M1` in the first element and *StMVAR type* components `M2` in the second element. The total number of mixture components is `M=M1+M2`.
`model`	is "GMVAR", "StMVAR", or "G-StMVAR" model considered? In the G-StMVAR model, the first `M1` components are GMVAR type and the rest `M2` components are StMVAR type.
`conditional`	a logical argument specifying whether the conditional or exact log-likelihood function

`parametrization`

`"intercept"` or `"mean"` determining whether the model is parametrized with intercept parameters $\phi_{m,0}$ or regime means μ_m, m=1,...,M.

`constraints` a size $(Mpd^2 \times q)$ constraint matrix C specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, ..., \phi_M) = C\psi$, where $\phi_m = (vec(A_{m,1}), ..., vec(A_{m,p})(pd^2 x 1), m = 1, ..., M$, contains the coefficient matrices and ψ $(qx1)$ contains the related parameters. For example, to restrict the AR-parameters to be the same for all regimes, set C= `[I:...:I]'` $(Mpd^2 \times pd^2)$ where `I = diag(p*d^2)`. Ignore (or set to `NULL`) if linear constraints should **not** be employed.

`same_means` Restrict the mean parameters of some regimes to be the same? Provide a list of numeric vectors such that each numeric vector contains the regimes that should share the common mean parameters. For instance, if `M=3`, the argument `list(1, 2:3)` restricts the mean parameters of the second and third regime to be the same but the first regime has freely estimated (unconditional) mean. Ignore or set to `NULL` if mean parameters should not be restricted to be the same among any regimes. **This constraint is available only for mean parametrized models; that is, when** `parametrization="mean"`**.**

`weight_constraints`

a numeric vector of length $M - 1$ specifying fixed parameter values for the mixing weight parameters α_m, $m = 1, ..., M - 1$. Each element should be strictly between zero and one, and the sum of all the elements should be strictly less than one.

`structural_pars`

If `NULL` a reduced form model is considered. Reduced models can be used directly as recursively identified structural models. For a structural model identified by conditional heteroskedasticity, should be a list containing at least the first one of the following elements:

- `W` - a (dxd) matrix with its entries imposing constraints on W: `NA` indicating that the element is unconstrained, a positive value indicating strict positive sign constraint, a negative value indicating strict negative sign constraint, and zero indicating that the element is constrained to zero.
- `C_lambda` - a $(d(M-1) \times r)$ constraint matrix that satisfies $(\lambda_2, ..., \lambda_M) = C_\lambda \gamma$ where γ is the new $(r \times 1)$ parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of `C_lambda` must be either **positive** or **zero**. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.
- `fixed_lambdas` - a length $d(M - 1)$ numeric vector $(\lambda_2, ..., \lambda_M)$ with elements strictly larger than zero specifying the fixed parameter values for the parameters λ_{mi} should be constrained to. This constraint is alternative `C_lambda`. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.

See Virolainen (forthcoming) for the conditions required to identify the shocks and for the B-matrix as well (it is W times a time-varying diagonal matrix with positive diagonal entries).

`ngen` a positive integer specifying the number of generations to be ran through in the genetic algorithm.

`popsize` a positive even integer specifying the population size in the genetic algorithm. Default is `10*n_params`.

`smart_mu` a positive integer specifying the generation after which the random mutations in the genetic algorithm are "smart". This means that mutating individuals will mostly mutate fairly close (or partially close) to the best fitting individual (which has the least regimes with time varying mixing weights practically at zero) so far.

`initpop` a list of parameter vectors from which the initial population of the genetic algorithm will be generated from. The parameter vectors should be...

For unconstrained models: Should be size $((M(pd^2+d+d(d+1)/2+2)-M1-1)\times 1)$ and have the form θ=(υ_1, ...,υ_M, $\alpha_1, ..., \alpha_{M-1}$,$\nu$), where

- $\upsilon_m = (\phi_{m,0}$,$\phi_m, \sigma_m)$
- ϕ_m= $(vec(A_{m,1}), ..., vec(A_{m,p})$
- and $\sigma_m = vech(\Omega_m)$, m=1,...,M,
- ν= $(\nu_{M1+1}, ..., \nu_M)$
- $M1$ is the number of GMVAR type regimes.

For constrained models: Should be size $((M(d+d(d+1)/2+2)+q-M1-1)\times 1)$ and have the form θ= $(\phi_{1,0}, ..., \phi_{M,0}$,$\psi$, $\sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1}$,$\nu$), where

- ψ $(qx1)$ satisfies $(\phi_1, ..., \phi_M) = C\psi$ where C is a $(Mpd^2 \times q)$ constraint matrix.

For same_means models: Should have the form θ= (μ,ψ, $\sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1}$,$\nu$), where

- μ= $(\mu_1, ..., \mu_g)$ where μ_i is the mean parameter for group i and g is the number of groups.
- If AR constraints are employed, ψ is as for constrained models, and if AR constraints are not employed, ψ= (ϕ_1, ...,ϕ_M).

For structural models: Should have the form θ= ($\phi_{1,0}$, ..., $\phi_{M,0}$,ϕ_1, ...,ϕ_M, $vec(W)$,λ_2, ...,λ_M, α_1, ..., where

- λ_m = $(\lambda_{m1}, ..., \lambda_{md})$ contains the eigenvalues of the mth mixture component.

If AR parameters are constrained: Replace $\phi_1, ..., \phi_M$ with ψ $(qx1)$ that satisfies $(\phi_1, ..., \phi_M) = C\psi$, as above.

If same_means: Replace $(\phi_{1,0}, ..., \phi_{M,0})$ with $(\mu_1, ..., \mu_g)$, as above.

If W is constrained: Remove the zeros from $vec(W)$ and make sure the other entries satisfy the sign constraints.

If λ_{mi} are constrained: Replace λ_2, ...,λ_M with γ $(rx1)$ that satisfies (λ_2, ..., λ_M) = $C_\lambda\gamma$ where C_λ is a $(d(M-1)xr)$ constraint matrix.

Above, $\phi_{m,0}$ is the intercept parameter, $A_{m,i}$ denotes the ith coefficient matrix of the mth mixture component, Ω_m denotes the error term covariance matrix of the m:th mixture component, and α_m is the mixing weight parameter. The W and λ_{mi} are structural parameters replacing the error term covariance matrices (see Virolainen, 2022). If $M = 1$, α_m and λ_{mi} are dropped. If `parametrization=="mean"`, just replace each $\phi_{m,0}$ with regimewise mean μ_m.

$vec()$ is vectorization operator that stacks columns of a given matrix into a vector. $vech()$ stacks columns of a given matrix from the principal diagonal downwards (including elements on the diagonal) into a vector.

In the **GMVAR model**, $M1 = M$ and ν is dropped from the parameter vector. In the **StMVAR** model, $M1 = 0$. In the **G-StMVAR** model, the first `M1` regimes are *GMVAR type* and the rest `M2` regimes are *StMVAR type*. In **StMVAR** and **G-StMVAR** models, the degrees of freedom parameters in ν should be strictly larger than two.

The notation is similar to the cited literature.

`mu_scale` a size $(dx1)$ vector defining **means** of the normal distributions from which each mean parameter μ_m is drawn from in random mutations. Default is `colMeans(data)`. Note that mean-parametrization is always used for optimization in `GAfit` - even when `parametrization=="intercept"`. However, input (in `initpop`) and output (return value) parameter vectors can be intercept-parametrized.

`mu_scale2` a size $(dx1)$ strictly positive vector defining **standard deviations** of the normal distributions from which each mean parameter μ_m is drawn from in random mutations. Default is `2*sd(data[,i]), i=1,..,d`.

`omega_scale` a size $(dx1)$ strictly positive vector specifying the scale and variability of the random covariance matrices in random mutations. The covariance matrices are drawn from (scaled) Wishart distribution. Expected values of the random covariance matrices are `diag(omega_scale)`. Standard deviations of the diagonal elements are `sqrt(2/d)*omega_scale[i]` and for non-diagonal elements they are `sqrt(1/d*omega_scale[i]*omega_scale[j])`. Note that for `d>4` this scale may need to be chosen carefully. Default in `GAfit` is `var(stats::ar(data[,i], order.max=10)$resid, na.rm=TRUE), i=1,...,d`. This argument is ignored if structural model is considered.

`W_scale` a size $(dx1)$ strictly positive vector partly specifying the scale and variability of the random covariance matrices in random mutations. The elements of the matrix W are drawn independently from such normal distributions that the expectation of the main **diagonal** elements of the first regime's error term covariance matrix $\Omega_1 = WW'$ is `W_scale`. The distribution of Ω_1 will be in some sense like a Wishart distribution but with the columns (elements) of W obeying the given constraints. The constraints are accounted for by setting the element to be always zero if it is subject to a zero constraint and for sign constraints the absolute value or negative the absolute value are taken, and then the variances of the elements of W are adjusted accordingly. This argument is ignored if reduced form model is considered.

`lambda_scale` a length $M - 1$ vector specifying the **standard deviation** of the mean zero normal distribution from which the eigenvalue λ_{mi} parameters are drawn from in random mutations. As the eigenvalues should always be positive, the absolute value is taken. The elements of `lambda_scale` should be strictly positive real numbers with the $m - 1$th element giving the degrees of freedom for the mth regime. The expected value of the main **diagonal** elements ij of the mth $(m > 1)$ error term covariance matrix will be `W_scale[i]*(d - n_i)^(-1)*sum(lambdas*ind_fun)` where the $(dx1)$ vector `lambdas` is drawn from the absolute value of the t-distribution, `n_i` is the number of zero constraints in the ith row of W and `ind_fun` is an indicator function that takes the

value one iff the ijth element of W is not constrained to zero. Basically, larger lambdas (or smaller degrees of freedom) imply larger variance.

If the lambda parameters are **constrained** with the $(d(M-1)xr)$ constraint matrix C_lambda, then provide a length r vector specifying the standard deviation of the (absolute value of the) mean zero normal distribution each of the γ parameters are drawn from (the γ is a $(rx1)$ vector). The expected value of the main diagonal elements of the covariance matrices then depend on the constraints.

This argument is ignored if $M == 1$ or a reduced form model is considered. Default is `rep(3, times=M-1)` if lambdas are not constrained and `rep(3, times=r)` if lambdas are constrained.

As with omega_scale and W_scale, this argument should be adjusted carefully if specified by hand. **NOTE** that if lambdas are constrained in some other way than restricting some of them to be identical, this parameter should be adjusted accordingly in order to the estimation succeed!

`ar_scale` a positive real number between zero and one, adjusting how large AR parameter values are typically proposed in construction of the initial population: larger value implies larger coefficients (in absolute value). After construction of the initial population, a new scale is drawn from `(0, upper_ar_scale)` uniform distribution in each iteration. With large `p` or `d`, `ar_scale` is restricted from above, see the details section.

`upper_ar_scale` the upper bound for `ar_scale` parameter (see above) in the random mutations. Setting this too high might lead to failure in proposing new parameters that are well enough inside the parameter space, and especially with large `p` one might want to try smaller upper bound (e.g., 0.5). With large `p` or `d`, `upper_ar_scale` is restricted from above, see the details section.

`ar_scale2` a positive real number adjusting how large AR parameter values are typically proposed in some random mutations (if AR constraints are employed, in all random mutations): larger value implies **smaller** coefficients (in absolute value). **Values larger than 1 can be used if the AR coefficients are expected to be very small. If set smaller than 1, be careful as it might lead to failure in the creation of stationary parameter candidates**

`regime_force_scale`

a non-negative real number specifying how much should natural selection favor individuals with less regimes that have almost all mixing weights (practically) at zero. Set to zero for no favoring or large number for heavy favoring. Without any favoring the genetic algorithm gets more often stuck in an area of the parameter space where some regimes are wasted, but with too much favouring the best genes might never mix into the population and the algorithm might converge poorly. Default is `1` and it gives $2x$ larger surviving probability weights for individuals with no wasted regimes compared to individuals with one wasted regime. Number `2` would give $3x$ larger probability weights etc.

`red_criteria` a length 2 numeric vector specifying the criteria that is used to determine whether a regime is redundant (or "wasted") or not. Any regime `m` which satisfies `sum(mixingWeights[,m] > red_criteria[1]) < red_criteria[2]*n_obs` will be considered "redundant". One should be careful when adjusting this argument (set `c(0, 0)` to fully disable the 'redundant regime' features from the algorithm).

`pre_smart_mu_prob`
: A number in $[0, 1]$ giving a probability of a "smart mutation" occuring randomly in each iteration before the iteration given by the argument `smart_mu`.

`to_return`
: should the genetic algorithm return the best fitting individual which has "positive enough" mixing weights for as many regimes as possible (`"alt_ind"`) or the individual which has the highest log-likelihood in general (`"best_ind"`) but might have more wasted regimes?

`minval`
: a real number defining the minimum value of the log-likelihood function that will be considered. Values smaller than this will be treated as they were `minval` and the corresponding individuals will never survive. The default is `-(10^(ceiling(log10(n_obs)) + d) - 1)`.

`seed`
: a single value, interpreted as an integer, or NULL, that sets seed for the random number generator in the beginning of the function call. If calling `GAfit` from `fitGSMVAR`, use the argument `seeds` instead of passing the argument `seed`.

Details

The core of the genetic algorithm is mostly based on the description by *Dorsey and Mayer (1995)*. It utilizes a slightly modified version of the individually adaptive crossover and mutation rates described by *Patnaik and Srinivas (1994)* and employs (50%) fitness inheritance discussed by *Smith, Dike and Stegmann (1995)*.

By "redundant" or "wasted" regimes we mean regimes that have the time varying mixing weights practically at zero for almost all t. A model including redundant regimes would have about the same log-likelihood value without the redundant regimes and there is no purpose to have redundant regimes in a model.

Some of the AR coefficients are drawn with the algorithm by Ansley and Kohn (1986). However, when using large `ar_scale` with large `p` or `d`, numerical inaccuracies caused by the imprecision of the float-point presentation may result in errors or nonstationary AR-matrices. Using smaller `ar_scale` facilitates the usage of larger `p` or `d`. Therefore, we bound `upper_ar_scale` from above by $1 - pd/150$ when `p*d>40` and by 1 otherwise.

Value

Returns the estimated parameter vector which has the form described in `initpop`.

References

- Ansley C.F., Kohn R. 1986. A note on reparameterizing a vector autoregressive moving average model to enforce stationarity. *Journal of statistical computation and simulation*, **24**:2, 99-106.
- Dorsey R. E. and Mayer W. J. 1995. Genetic algorithms for estimation problems with multiple optima, nondifferentiability, and other irregular features. *Journal of Business & Economic Statistics*, **13**, 53-66.
- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Patnaik L.M. and Srinivas M. 1994. Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms. *Transactions on Systems, Man and Cybernetics* **24**, 656-667.

- Smith R.E., Dike B.A., Stegmann S.A. 1995. Fitness inheritance in genetic algorithms. *Proceedings of the 1995 ACM Symposium on Applied Computing*, 345-350.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

Examples

```
# Preliminary estimation of a G-StMVAR(1, 1, 1) model with 50 generations.
GA_estimates <- GAfit(gdpdef, p=1, M=c(1, 1), model="G-StMVAR",
                      ngen=50, seed=1)
GA_estimates
```

`gdpdef` *U.S. real GDP percent change and GDP implicit price deflator percent change.*

Description

A dataset containing a quarterly U.S. time series with two components: the percentage change of real GDP and the percentage change of GDP implicit price deflator, covering the period from 1959Q1 - 2019Q4.

Usage

```
gdpdef
```

Format

A numeric matrix of class `'ts'` with 244 rows and 2 columns with one time series in each column:

First column (GDP): The quarterly percent change of real U.S. GDP, from 1959Q1 to 2019Q4, https://fred.stlouisfed.org/series/GDPC1.

Second column (GDPDEF): The quarterly percent change of U.S. GDP implicit price deflator, from 1959Q1 to 2019Q4, https://fred.stlouisfed.org/series/GDPDEF.

Source

The Federal Reserve Bank of St. Louis database

`get_boldA_eigens` *Calculate absolute values of the eigenvalues of the "bold A" matrices containing the AR coefficients*

Description

`get_boldA_eigens` calculates absolute values of the eigenvalues of the "bold A" matrices containing the AR coefficients for each mixture component.

Usage

```
get_boldA_eigens(gsmvar)
```

Arguments

`gsmvar`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.

Value

Returns a matrix with $d * p$ rows and M columns - one column for each regime. The mth column contains the absolute values (or modulus) of the eigenvalues of the "bold A" matrix containing the AR coefficients correspinding to regime m.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. (forthcoming). A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*.
- Virolainen S. 2022. Gaussian and Student's t mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks in the Euro area. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

Examples

```
# GMVAR(2, 2), d=2 model
params22 <- c(0.36, 0.121, 0.223, 0.059, -0.151, 0.395, 0.406, -0.005,
 0.083, 0.299, 0.215, 0.002, 0.03, 0.484, 0.072, 0.218, 0.02, -0.119,
  0.722, 0.093, 0.032, 0.044, 0.191, 1.101, -0.004, 0.105, 0.58)
mod22 <- GSMVAR(p=2, M=2, d=2, params=params22)
get_boldA_eigens(mod22)
```

`get_omega_eigens` *Calculate the eigenvalues of the "Omega" error term covariance matrices*

Description

`get_omega_eigens` calculates the eigenvalues of the "Omega" error term covariance matrices for each mixture component.

Usage

```
get_omega_eigens(gsmvar)
```

Arguments

`gsmvar`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.

Value

Returns a matrix with d rows and M columns - one column for each regime. The mth column contains the eigenvalues of the "Omega" error term covariance matrix of the mth regime.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. (forthcoming). A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*.
- Virolainen S. 2022. Gaussian and Student's t mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks in the Euro area. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

Examples

```
# GMVAR(2, 2), d=2 model
params22 <- c(0.36, 0.121, 0.223, 0.059, -0.151, 0.395, 0.406, -0.005,
 0.083, 0.299, 0.215, 0.002, 0.03, 0.484, 0.072, 0.218, 0.02, -0.119,
  0.722, 0.093, 0.032, 0.044, 0.191, 1.101, -0.004, 0.105, 0.58)
mod22 <- GSMVAR(p=2, M=2, d=2, params=params22)
get_omega_eigens(mod22)
```

`get_regime_autocovs` *Calculate regimewise autocovariance matrices*

Description

`get_regime_autocovs` calculates the first p regimewise autocovariance matrices $\Gamma_m(j)$ for the given GMVAR, StMVAR, or G-StMVAR model.

Usage

```
get_regime_autocovs(gsmvar)
```

Arguments

`gsmvar`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.

Value

Returns an $(dxdxp + 1xM)$ array containing the first p regimewise autocovariance matrices. The subset `[, , j, m]` contains the j-1:th lag autocovariance matrix of the m:th regime.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lütkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.
- McElroy T. 2017. Computation of vector ARMA autocovariances. *Statistics and Probability Letters*, **124**, 92-96.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

Other moment functions: `cond_moments()`, `get_regime_means()`, `uncond_moments()`

Examples

```
# GMVAR(1,2), d=2 model:
params12 <- c(0.55, 0.112, 0.344, 0.055, -0.009, 0.718, 0.319, 0.005,
 0.03, 0.619, 0.173, 0.255, 0.017, -0.136, 0.858, 1.185, -0.012,
 0.136, 0.674)
mod12 <- GSMVAR(gdpdef, p=1, M=2, params=params12)
get_regime_autocovs(mod12)

# Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
```

```
params22s <- c(0.36, 0.121, 0.484, 0.072, 0.223, 0.059, -0.151, 0.395,
 0.406, -0.005, 0.083, 0.299, 0.218, 0.02, -0.119, 0.722, 0.093, 0.032,
 0.044, 0.191, 0.057, 0.172, -0.46, 0.016, 3.518, 5.154, 0.58)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
mod22s <- GSMVAR(gdpdef, p=2, M=2, params=params22s, structural_pars=list(W=W_22))
mod22s
get_regime_autocovs(mod22s)
```

`get_regime_means` *Calculate regime means μ_m*

Description

`get_regime_means` calculates regime means $\mu_m = (I - \sum A_{m,i})^(- 1))$ for the given GMVAR, StMVAR, or G-StMVAR model.

Usage

```
get_regime_means(gsmvar)
```

Arguments

`gsmvar`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.

Value

Returns a (dxM) matrix containing regime mean μ_m in the m:th column, $m = 1, .., M$.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. (forthcoming). A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*.
- Virolainen S. 2022. Gaussian and Student's t mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks in the Euro area. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

See Also

`uncond_moments`, `get_regime_autocovs`, `cond_moments`

Other moment functions: `cond_moments()`, `get_regime_autocovs()`, `uncond_moments()`

Examples

```
# GMVAR(1,2), d=2 model:
params12 <- c(0.55, 0.112, 0.344, 0.055, -0.009, 0.718, 0.319, 0.005,
 0.03, 0.619, 0.173, 0.255, 0.017, -0.136, 0.858, 1.185, -0.012,
 0.136, 0.674)
mod12 <- GSMVAR(gdpdef, p=1, M=2, params=params12)
mod12
get_regime_means(mod12)

# Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
params22s <- c(0.36, 0.121, 0.484, 0.072, 0.223, 0.059, -0.151, 0.395,
 0.406, -0.005, 0.083, 0.299, 0.218, 0.02, -0.119, 0.722, 0.093, 0.032,
 0.044, 0.191, 0.057, 0.172, -0.46, 0.016, 3.518, 5.154, 0.58)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
mod22s <- GSMVAR(gdpdef, p=2, M=2, params=params22s, structural_pars=list(W=W_22))
mod22s
get_regime_means(mod22s)
```

GFEVD *Estimate generalized forecast error variance decomposition for structural (and reduced form) GMVAR, StMVAR, and G-StMVAR models.*

Description

GFEVD estimates generalized forecast error variance decomposition for structural (and reduced form) GMVAR, StMVAR, and G-StMVAR models.

Usage

```
GFEVD(
  gsmvar,
  shock_size = 1,
  N = 30,
  initval_type = c("data", "random", "fixed"),
  R1 = 250,
  R2 = 250,
  init_regimes = NULL,
  init_values = NULL,
  which_cumulative = numeric(0),
  include_mixweights = FALSE,
  ncores = 2,
  seeds = NULL
)

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

## S3 method for class 'gfevd'
print(x, ..., digits = 2, N_to_print)
```

Arguments

gsmvar	an object of class 'gsmvar', typically created with fitGSMVAR or GSMVAR.
shock_size	What shocks size should be used for all shocks? By the definition of the SGMVAR, SStMVAR, and SG-StMVAR model, the conditional covariance matrix of the structural shock is identity matrix.
N	a positive integer specifying the horizon how far ahead should the GFEVD be calculated.
initval_type	What type initial values are used for estimating the GIRFs that the GFEVD is based on? "data": Estimate the GIRF for all the possible length p histories in the data. "random": Estimate the GIRF for several random initial values generated from the stationary distribution of the process or from the stationary distribution of specific regime(s) chosen with the argument init_regimes. The number of initial values is set with the argument R2. "fixed": Estimate the GIRF for a fixed initial value only, which is specified with the argument init_values.
R1	the number of repetitions used to estimate GIRF for each initial value.
R2	the number of initial values to be drawn if initval_type="random".
init_regimes	a numeric vector of length at most M and elements in $1, ..., M$ specifying the regimes from which the initial values should be generated from. The initial values will be generated from a mixture distribution with the mixture components being the stationary distributions of the specific regimes and the (proportional) mixing weights given by the mixing weight parameters of those regimes. Note that if init_regimes=1:M, the initial values are generated from the stationary distribution of the process and if init_regimes=m, the initial value are generated from the stationary distribution of the mth regime. Ignored if the argument init_values is specified.
init_values	a size $(p \times d)$ matrix specifying the initial values, where d is the number of time series in the system. The **last** row will be used as initial values for the first lag, the second last row for second lag etc. If not specified, initial values will be drawn according to mixture distribution specifed by the argument init_regimes.
which_cumulative	a numeric vector with values in $1, ..., d$ (d=ncol(data)) specifying which the variables for which the impulse responses should be cumulative. Default is none.
include_mixweights	should the GFEVD be estimated for the mixing weights as well? Note that this is ignored if M=1 and if M=2 the GFEVD will be the same for both regime's mixing weights.
ncores	the number CPU cores to be used in parallel computing. Only single core computing is supported if an initial value is specified (and the GIRF won't thus be estimated multiple times).
seeds	a numeric vector containing the random number generator seed for estimation of each GIRF. Should have the length...

- ...nrow(data) - p + 1 if initval_type="data".
- ...R2 if initval_type="random".
- ...1 if initval_type="fixed.".

Set to NULL for not initializing the seed. Exists for creating reproducible results.

x	object of class 'gfevd' generated by the function GFEVD.
...	currently not used.
digits	the number of decimals to print
N_to_print	an integer specifying the horizon how far to print the estimates. The default is that all the values are printed.

Details

The model DOES NOT need to be structural in order for this function to be applicable. When an identified structural GMVAR, StMVAR, or G-StMVAR model is provided in the argument gsmvar, the identification imposed by the model is used. When a reduced form model is provided in the argument gsmvar, lower triangular Cholesky identification is used to identify the shocks.

The GFEVD is a forecast error variance decomposition calculated with the generalized impulse response function (GIRF). Lanne and Nyberg (2016) for details. Note, however, that the related GIRFs are calculated using the algorithm given in Virolainen (2022).

Value

Returns and object of class 'gfevd' containing the GFEVD for all the variables and if include_mixweights=TRUE also to the mixing weights. Note that the decomposition does not exist at horizon zero for mixing weights because the related GIRFs are always zero at impact.

Functions

- plot(gfevd): plot method
- print(gfevd): print method

References

- Lanne M. and Nyberg H. 2016. Generalized Forecast Error Variance Decomposition for Linear and Nonlineae Multivariate Models. *Oxford Bulletin of Economics and Statistics*, **78**, 4, 595-603.
- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

GIRF, linear_IRF, fitGSMVAR, GSMVAR, gsmvar_to_sgsmvar, reorder_W_columns, swap_W_signs, simulate.gsmvar

Examples

```
# These are long-running examples that use parallel computing.
# It takes approximately 30 seconds to run all the below examples.

## StMVAR(1, 2), d=2 model identified recursively by lower-triangular
## Cholesky decomposition (i.e., reduced form model is specified):
params12t <- c(0.55, 0.11, 0.34, 0.05, -0.01, 0.72, 0.58, 0.01, 0.06, 0.17,
  0.25, 0.34, 0.05, -0.01, 0.72, 0.50, -0.01, 0.20, 0.60, 3.00, 12.00)
mod12t <- GSMVAR(gdpdef, p=1, M=2, params=params12t, model="StMVAR")

# Estimating the GFEVD using all possible histories in the data as the
# initial values:
gfevd0 <- GFEVD(mod12t, N=24, R1=10, initval_type="data")
gfevd0
plot(gfevd0)
## NOTE: Use larger R1 is empirical applications! Small R1 is used
## here only to fasten the execution time of the examples.

## Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
params22s <- c(0.36, 0.121, 0.484, 0.072, 0.223, 0.059, -0.151, 0.395,
 0.406, -0.005, 0.083, 0.299, 0.218, 0.02, -0.119, 0.722, 0.093, 0.032,
 0.044, 0.191, 0.057, 0.172, -0.46, 0.016, 3.518, 5.154, 0.58)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
mod22s <- GSMVAR(gdpdef, p=2, M=2, params=params22s,
 structural_pars=list(W=W_22))
mod22s

# Estimating the GFEVD using all possible histories in the data as the
# initial values:
gfevd1 <- GFEVD(mod22s, N=24, R1=10, initval_type="data")
gfevd1
plot(gfevd1)

# Estimate GFEVD with the initial values generated from the stationary
# distribution of the process:
gfevd2 <- GFEVD(mod22s, N=24, R1=10, R2=100, initval_type="random")
gfevd2
plot(gfevd2)

# Estimate GFEVD with fixed hand specified initial values. We use the
# unconditional mean of the process:
myvals <- rbind(mod22s$uncond_moments$uncond_mean,
                mod22s$uncond_moments$uncond_mean)
gfevd3 <- GFEVD(mod22s, N=36, R1=50, initval_type="fixed",
 init_values=myvals, include_mixweights=TRUE)
gfevd3
plot(gfevd3)
```

GIRF *Estimate generalized impulse response function for structural (and reduced form) GMVAR, StMVAR, and G-StMVAR models.*

Description

GIRF estimates generalized impulse response function for structural (and reduced form) GMVAR, StMVAR, and G-StMVAR models.

Usage

```
GIRF(
  gsmvar,
  which_shocks,
  shock_size = 1,
  N = 30,
  R1 = 250,
  R2 = 250,
  init_regimes = 1:sum(gsmvar$model$M),
  init_values = NULL,
  which_cumulative = numeric(0),
  scale = NULL,
  scale_type = c("instant", "peak"),
  scale_horizon = N,
  ci = c(0.95, 0.8),
  include_mixweights = TRUE,
  ncores = 2,
  plot_res = TRUE,
  seeds = NULL,
  ...
)

## S3 method for class 'girf'
plot(x, add_grid = FALSE, margs, ...)

## S3 method for class 'girf'
print(x, ..., digits = 2, N_to_print)
```

Arguments

gsmvar	an object of class 'gsmvar', typically created with fitGSMVAR or GSMVAR.
which_shocks	a numeric vector of length at most d (=ncol(data)) and elements in $1, ..., d$ specifying the structural shocks for which the GIRF should be estimated.
shock_size	a non-zero scalar value specifying the common size for all scalar components of the structural shock. Note that the conditional covariance matrix of the structural shock is an identity matrix and that the (generalized) impulse responses may not be symmetric to the sign and size of the shock.

N — a positive integer specifying the horizon how far ahead should the generalized impulse responses be calculated.

R1 — the number of repetitions used to estimate GIRF for each initial value.

R2 — the number of initial values to be drawn from a stationary distribution of the process or of a specific regime? The confidence bounds will be sample quantiles of the GIRFs based on different initial values. Ignored if the argument `init_value` is specified.

init_regimes — a numeric vector of length at most M and elements in $1, ..., M$ specifying the regimes from which the initial values should be generated from. The initial values will be generated from a mixture distribution with the mixture components being the stationary distributions of the specific regimes and the (proportional) mixing weights given by the mixing weight parameters of those regimes. Note that if `init_regimes=1:M`, the initial values are generated from the stationary distribution of the process and if `init_regimes=m`, the initial value are generated from the stationary distribution of the mth regime. Ignored if the argument `init_values` is specified.

init_values — a size $(p \times d)$ matrix specifying the initial values, where d is the number of time series in the system. The **last** row will be used as initial values for the first lag, the second last row for second lag etc. If not specified, initial values will be drawn according to mixture distribution specifed by the argument `init_regimes`.

which_cumulative — a numeric vector with values in $1, ..., d$ (`d=ncol(data)`) specifying which the variables for which the impulse responses should be cumulative. Default is none.

scale — should the GIRFs to some of the shocks be scaled so that they correspond to a specific magnitude of instantaneous or peak response of some specific variable (see the argument `scale_type`)? Provide a length three vector where the shock of interest is given in the first element (an integer in $1, ..., d$), the variable of interest is given in the second element (an integer in $1, ..., d$), and the magnitude of its instantaneous or peak response in the third element (a non-zero real number). If the GIRFs of multiple shocks should be scaled, provide a matrix which has one column for each of the shocks with the columns being the length three vectors described above.

scale_type — If argument `scale` is specified, should the GIRFs be scaled to match an instantaneous response (`"instant"`) or peak response (`"peak"`). If `"peak"`, the scale is based on the largest magnitude of peak response in absolute value. Ignored if `scale` is not specified.

scale_horizon — If `scale_type == "peak"` what the maximum horizon up to which peak response is expected? Scaling won't based on values after this.

ci — a numeric vector with elements in $(0, 1)$ specifying the confidence levels of the confidence intervals.

include_mixweights — should the generalized impulse response be calculated for the mixing weights as well? `TRUE` or `FALSE`.

ncores — the number CPU cores to be used in parallel computing. Only single core computing is supported if an initial value is specified (and the GIRF won't thus be estimated multiple times).

plot_res	TRUE if the results should be plotted, FALSE if not.
seeds	a length R2 vector containing the random number generator seed for estimation of each GIRF. A single number of an initial value is specified. or NULL for not initializing the seed. Exists for creating reproducible results.
...	arguments passed to grid which plots grid to the figure.
x	object of class 'girf' generated by the function GIRF.
add_grid	should grid be added to the plots?
margs	numeric vector of length four that adjusts the [bottom_marginal, left_marginal, top_marginal, right_marginal] as the relative sizes of the marginals to the figures of the responses.
digits	the number of decimals to print
N_to_print	an integer specifying the horizon how far to print the estimates and confidence intervals. The default is that all the values are printed.

Details

The model DOES NOT need to be structural in order for this function to be applicable. When an identified structural GMVAR, StMVAR, or G-StMVAR model is provided in the argument gsmvar, the identification imposed by the model is used. When a reduced form model is provided in the argument gsmvar, lower triangular Cholesky identification is used to identify the shocks.

The confidence bounds reflect uncertainty about the initial state (but currently not about the parameter estimates) if initial values are not specified. If initial values are specified, there won't currently be confidence intervals. See the cited paper by Virolainen (2022) for details about the algorithm.

Note that if the argument scale is used, the scaled responses of the mixing weights might be more than one in absolute value.

Value

Returns a class 'girf' list with the GIRFs in the first element ($girf_res) and the used arguments the rest. The first element containing the GIRFs is a list with the mth element containing the point estimates for the GIRF in $point_est (the first element) and confidence intervals in $conf_ints (the second element). The first row is for the GIRF at impact ($n = 0$), the second for $n = 1$, the third for $n = 2$, and so on.

The element $all_girfs is a list containing results from all the individual GIRFs obtained from the MC repetitions. Each element is for one shock and results are in array of the form [horizon, variables, MC-repetitions].

Functions

- plot(girf): plot method
- print(girf): print method

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. (forthcoming). A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*.
- Virolainen S. 2022. Gaussian and Student's t mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks in the Euro area. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

See Also

GFEVD, linear_IRF, fitGSMVAR, GSMVAR, gsmvar_to_sgsmvar, reorder_W_columns, swap_W_signs, simulate.gsmvar, predict.gsmvar, profile_logliks, quantile_residual_tests, LR_test, Wald_test

Examples

```
# These are long-running examples that use parallel computing.
# It takes approximately 30 seconds to run all the below examples.

## StMVAR(1, 2), d=2 model identified recursively by lower-triangular
## Cholesky decomposition (i.e., reduced form model is specified):
params12t <- c(0.55, 0.11, 0.34, 0.05, -0.01, 0.72, 0.58, 0.01, 0.06, 0.17,
  0.25, 0.34, 0.05, -0.01, 0.72, 0.50, -0.01, 0.20, 0.60, 3.00, 12.00)
mod12t <- GSMVAR(gdpdef, p=1, M=2, params=params12t, model="StMVAR")

# Estimating the GIRFs of both structural shocks with initial values
# drawn from the stationary distribution of the process,
# 12 periods ahead, confidence levels 0.95 and 0.8:
girf0 <- GIRF(mod12t, N=12, R1=100, R2=100)
girf0
plot(girf0)
## NOTE: Small R1 and R2 is used here to shorten the estimation time.
## Larger R1 and R2 should be considered in empirical applications!

## Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
params22s <- c(0.36, 0.121, 0.484, 0.072, 0.223, 0.059, -0.151, 0.395,
 0.406, -0.005, 0.083, 0.299, 0.218, 0.02, -0.119, 0.722, 0.093, 0.032,
  0.044, 0.191, 0.057, 0.172, -0.46, 0.016, 3.518, 5.154, 0.58)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
mod22s <- GSMVAR(gdpdef, p=2, M=2, params=params22s,
 structural_pars=list(W=W_22))
mod22s
# Alternatively, use:
#fit22s <- fitGSMVAR(gdpdef, p=2, M=2, structural_pars=list(W=W_22),
#                    ncalls=20, seeds=1:20)
# To obtain an estimated version of the same model.

# Estimating the GIRFs of both structural shocks with initial values
```

```
# drawn from the stationary distribution of the process,
# 12 periods ahead, confidence levels 0.95 and 0.8:
girf1 <- GIRF(mod22s, N=12, R1=100, R2=100)
girf1
plot(girf1)

# Estimating the GIRF of the second shock only, 12 periods ahead
# and shock size 1, initial values drawn from the stationary distribution
# of the first regime, confidence level 0.9:
girf2 <- GIRF(mod22s, which_shocks=2, shock_size=1, N=12, init_regimes=1,
              ci=0.9, R1=100, R2=100)

# Estimating the GIRFs of both structural shocks, negative one standard
# error shock, N=20 periods ahead, estimation based on 200 Monte Carlo
# simulations, and fixed initial values given by the last p observations
# of the data:
girf3 <- GIRF(mod22s, shock_size=-1, N=20, R1=200,
              init_values=mod22s$data)
```

`gmvar_to_gsmvar` *Makes the old class 'gmvar' objects compatible with the functions using class 'gsmvar' objects*

Description

`gmvar_to_gsmvar` makes class 'gmvar' objects compatible with the functions using class 'gsmvar' objects

Usage

```
gmvar_to_gsmvar(gsmvar)
```

Arguments

`gsmvar`	a class 'gmvar' or 'gsmvar' object.

Details

This exists so that models estimated with earlier versions of the package can be used conveniently.

Value

If the provided object has the class 'gsmvar', the provided object is returned without modifications. If the provided object has the class 'gmvar', its element `$model` is given a new subelement called also model and this is set to be "GMVAR". Also, the class of this object is changes to 'gsmvar' and then it is returned.

GSMVAR *Create a class 'gsmvar' object defining a reduced form or structural GMVAR, StMVAR, or G-StMVAR model*

Description

GSMVAR creates a class 'gsmvar' object that defines a reduced form or structural GMVAR, StMVAR, or G-StMVAR model

Usage

```
GSMVAR(
  data,
  p,
  M,
  d,
  params,
  conditional = TRUE,
  model = c("GMVAR", "StMVAR", "G-StMVAR"),
  parametrization = c("intercept", "mean"),
  constraints = NULL,
  same_means = NULL,
  weight_constraints = NULL,
  structural_pars = NULL,
  calc_cond_moments,
  calc_std_errors = FALSE,
  stat_tol = 0.001,
  posdef_tol = 1e-08,
  df_tol = 1e-08
)

## S3 method for class 'gsmvar'
logLik(object, ...)

## S3 method for class 'gsmvar'
residuals(object, ...)

## S3 method for class 'gsmvar'
summary(object, ..., digits = 2)

## S3 method for class 'gsmvar'
plot(x, ..., type = c("both", "series", "density"))

## S3 method for class 'gsmvar'
print(x, ..., digits = 2, summary_print = FALSE)
```

Arguments

`data` a matrix or class `'ts'` object with `d>1` columns. Each column is taken to represent a single times series. `NA` values are not supported. Ignore if defining a model without data is desired.

`p` a positive integer specifying the autoregressive order of the model.

`M` **For GMVAR and StMVAR models:** a positive integer specifying the number of mixture components.

For G-StMVAR models: a size (2×1) integer vector specifying the number of *GMVAR type* components `M1` in the first element and *StMVAR type* components `M2` in the second element. The total number of mixture components is `M=M1+M2`.

`d` number of times series in the system, i.e. `ncol(data)`. This can be used to define GSMVAR models without data and can be ignored if `data` is provided.

`params` a real valued vector specifying the parameter values.

For unconstrained models: Should be size $((M(pd^2 + d + d(d+1)/2 + 2) - M1 - 1)x1)$ and have the form θ=(υ_1, ...,υ_M, $\alpha_1, ..., \alpha_{M-1}$,$\nu$), where

- $\upsilon_m = (\phi_{m,0}$,$\phi_m, \sigma_m)$
- ϕ_m= $(vec(A_{m,1}), ..., vec(A_{m,p})$
- and $\sigma_m = vech(\Omega_m)$, m=1,...,M,
- ν= $(\nu_{M1+1}, ..., \nu_M)$
- $M1$ is the number of GMVAR type regimes.

For constrained models: Should be size $((M(d+d(d+1)/2+2)+q-M1-1)x1)$ and have the form θ= $(\phi_{1,0}, ..., \phi_{M,0}$,$\psi$, $\sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1}$,$\nu$), where

- ψ $(qx1)$ satisfies $(\phi_1, ..., \phi_M) = C\psi$ where C is a (Mpd^2xq) constraint matrix.

For same_means models: Should have the form θ= (μ,ψ, $\sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1}$,$\nu$), where

- μ= $(\mu_1, ..., \mu_g)$ where μ_i is the mean parameter for group i and g is the number of groups.
- If AR constraints are employed, ψ is as for constrained models, and if AR constraints are not employed, ψ= (ϕ_1, ...,ϕ_M).

For models with weight_constraints: Drop $\alpha_1, ..., \alpha_{M-1}$ from the parameter vector.

For structural models: Reduced form models can be directly used as recursively identified structural models. If the structural model is identified by conditional heteroskedasticity, the parameter vector should have the form θ= ($\phi_{1,0}, ..., \phi_{M,0}$,$\phi_1$, ...,$\phi_M, vec(W)$,$\lambda_2$, ...,$\lambda_M, \alpha_1, ..., \alpha_{M-1}$,$\nu$), where

- $\lambda_m = (\lambda_{m1}, ..., \lambda_{md})$ contains the eigenvalues of the mth mixture component.

If AR parameters are constrained: Replace $\phi_1, ..., \phi_M$ with ψ $(qx1)$ that satisfies $(\phi_1, ..., \phi_M) = C\psi$, as above.

If same_means: Replace $(\phi_{1,0}, ..., \phi_{M,0})$ with $(\mu_1, ..., \mu_g)$, as above.

If W is constrained: Remove the zeros from $vec(W)$ and make sure the other entries satisfy the sign constraints.

If λ_{mi} are constrained via `C_lambda`: Replace $\lambda_2, ..., \lambda_M$ with γ $(rx1)$ that satisfies $(\lambda_2, ..., \lambda_M) = C_\lambda \gamma$ where C_λ is a $(d(M-1)xr)$ constraint matrix.

If λ_{mi} are constrained via `fixed_lambdas`: Drop $\lambda_2, ..., \lambda_M$ from the parameter vector.

Above, $\phi_{m,0}$ is the intercept parameter, $A_{m,i}$ denotes the ith coefficient matrix of the mth mixture component, Ω_m denotes the error term covariance matrix of the m:th mixture component, and α_m is the mixing weight parameter. The W and λ_{mi} are structural parameters replacing the error term covariance matrices (see Virolainen, 2022). If $M = 1$, α_m and λ_{mi} are dropped. If `parametrization=="mean"`, just replace each $\phi_{m,0}$ with regimewise mean μ_m. $vec()$ is vectorization operator that stacks columns of a given matrix into a vector. $vech()$ stacks columns of a given matrix from the principal diagonal downwards (including elements on the diagonal) into a vector.

In the **GMVAR model**, $M1 = M$ and ν is dropped from the parameter vector. In the **StMVAR** model, $M1 = 0$. In the **G-StMVAR** model, the first `M1` regimes are *GMVAR type* and the rest `M2` regimes are *StMVAR type*. In **StMVAR** and **G-StMVAR** models, the degrees of freedom parameters in ν should be strictly larger than two.

The notation is similar to the cited literature.

`conditional` a logical argument specifying whether the conditional or exact log-likelihood function should be used.

`model` is "GMVAR", "StMVAR", or "G-StMVAR" model considered? In the G-StMVAR model, the first `M1` components are GMVAR type and the rest `M2` components are StMVAR type.

`parametrization`

`"intercept"` or `"mean"` determining whether the model is parametrized with intercept parameters $\phi_{m,0}$ or regime means μ_m, m=1,...,M.

`constraints` a size $(Mpd^2 \times q)$ constraint matrix C specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, ..., \phi_M) = C\psi$, where $\phi_m = (vec(A_{m,1}), ..., vec(A_{m,p})(pd^2x1), m = 1, ..., M$, contains the coefficient matrices and ψ $(qx1)$ contains the related parameters. For example, to restrict the AR-parameters to be the same for all regimes, set C= `[I:...:I]'` $(Mpd^2 \times pd^2)$ where `I = diag(p*d^2)`. Ignore (or set to `NULL`) if linear constraints should **not** be employed.

`same_means` Restrict the mean parameters of some regimes to be the same? Provide a list of numeric vectors such that each numeric vector contains the regimes that should share the common mean parameters. For instance, if `M=3`, the argument `list(1, 2:3)` restricts the mean parameters of the second and third regime to be the same but the first regime has freely estimated (unconditional) mean. Ignore or set to `NULL` if mean parameters should not be restricted to be the same among any regimes. **This constraint is available only for mean parametrized models; that is, when `parametrization="mean"`.**

`weight_constraints`
: a numeric vector of length $M - 1$ specifying fixed parameter values for the mixing weight parameters $\alpha_m,\ m = 1, ..., M - 1$. Each element should be strictly between zero and one, and the sum of all the elements should be strictly less than one.

`structural_pars`
: If `NULL` a reduced form model is considered. Reduced models can be used directly as recursively identified structural models. For a structural model identified by conditional heteroskedasticity, should be a list containing at least the first one of the following elements:

  - `W` - a (dxd) matrix with its entries imposing constraints on W: `NA` indicating that the element is unconstrained, a positive value indicating strict positive sign constraint, a negative value indicating strict negative sign constraint, and zero indicating that the element is constrained to zero.
  - `C_lambda` - a $(d(M-1) \times r)$ constraint matrix that satisfies $(\lambda_2, ..., \lambda_M) = C_\lambda \gamma$ where γ is the new $(r \times 1)$ parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of `C_lambda` must be either **positive** or **zero**. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.
  - `fixed_lambdas` - a length $d(M - 1)$ numeric vector $(\lambda_2, ..., \lambda_M)$ with elements strictly larger than zero specifying the fixed parameter values for the parameters λ_{mi} should be constrained to. This constraint is alternative `C_lambda`. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.

  See Virolainen (forthcoming) for the conditions required to identify the shocks and for the B-matrix as well (it is W times a time-varying diagonal matrix with positive diagonal entries).

`calc_cond_moments`
: should conditional means and covariance matrices should be calculated? Default is `TRUE` if the model contains data and `FALSE` otherwise.

`calc_std_errors`
: should approximate standard errors be calculated?

`stat_tol`
: numerical tolerance for stationarity of the AR parameters: if the "bold A" matrix of any regime has eigenvalues larger that `1 - stat_tol` the model is classified as non-stationary. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.

`posdef_tol`
: numerical tolerance for positive definiteness of the error term covariance matrices: if the error term covariance matrix of any regime has eigenvalues smaller than this, the model is classified as not satisfying positive definiteness assumption. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.

`df_tol`
: the parameter vector is considered to be outside the parameter space if all degrees of freedom parameters are not larger than `2 + df_tol`.

`object`
: object of class `'gsmvar'` generated by `fitGSMVAR` or `GSMVAR`.

`...`
: currently not used.

`digits`
: number of digits to be printed.

x	object of class 'gsmvar' generated by fitGSMVAR or GSMVAR.
type	which type figure should be produced? Or both?
summary_print	if set to TRUE then the print will include log-likelihood and information criteria values.

Details

If data is provided, then also multivariate quantile residuals (*Kalliovirta and Saikkonen 2010*) are computed and included in the returned object.

If the function fails to calculate approximative standard errors and the parameter values are near the border of the parameter space, it might help to use smaller numerical tolerance for the stationarity and positive definiteness conditions.

The first plot displays the time series together with estimated mixing weights. The second plot displays a (Gaussian) kernel density estimates of the individual series together with the marginal stationary density implied by the model. The colored regimewise stationary densities are multiplied with the mixing weight parameter estimates.

Value

Returns an object of class 'gsmvar' defining the specified reduced form or structural GMVAR, StMVAR, or G-StMVAR model. Can be used to work with other functions provided in gmvarkit.

Note that the first autocovariance/correlation matrix in $uncond_moments is for the lag zero, the second one for the lag one, etc.

Functions

- logLik(gsmvar): Log-likelihood method
- residuals(gsmvar): residuals method to extract multivariate quantile residuals
- summary(gsmvar): summary method
- plot(gsmvar): plot method for class 'gsmvar'
- print(gsmvar): print method

About S3 methods

If data is not provided, only the print and simulate methods are available. If data is provided, then in addition to the ones listed above, predict method is also available. See ?simulate.gsmvar and ?predict.gsmvar for details about the usage.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Kalliovirta L. and Saikkonen P. 2010. Reliable Residuals for Multivariate Nonlinear Time Series Models. *Unpublished Revision of HECER Discussion Paper No. 247*.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

fitGSMVAR, add_data, swap_parametrization, GIRF, gsmvar_to_sgsmvar, stmvar_to_gstmvar, reorder_W_columns, swap_W_signs, update_numtols

Examples

```
# GMVAR(1, 2), d=2 model:
params12 <- c(0.55, 0.112, 0.344, 0.055, -0.009, 0.718, 0.319, 0.005,
  0.03, 0.619, 0.173, 0.255, 0.017, -0.136, 0.858, 1.185, -0.012,
  0.136, 0.674)
mod12 <- GSMVAR(gdpdef, p=1, M=2, params=params12)
mod12

# GMVAR(1, 2), d=2 model without data
mod12_2 <- GSMVAR(p=1, M=2, d=2, params=params12)
mod12_2

# StMVAR(1, 2), d=2 model:
mod12t <- GSMVAR(gdpdef, p=1, M=2, params=c(params12, 10, 20),
                 model="StMVAR")
mod12t

# G-StMVAR(1, 1, 1), d=2 model:
mod12gs <- GSMVAR(gdpdef, p=1, M=c(1, 1), params=c(params12, 20),
                  model="G-StMVAR")
mod12gs

# GMVAR(2, 2), d=2 model with mean-parametrization:
params22 <- c(0.869, 0.549, 0.223, 0.059, -0.151, 0.395, 0.406,
 -0.005, 0.083, 0.299, 0.215, 0.002, 0.03, 0.576, 1.168, 0.218,
 0.02, -0.119, 0.722, 0.093, 0.032, 0.044, 0.191, 1.101, -0.004,
 0.105, 0.58)
mod22 <- GSMVAR(gdpdef, p=2, M=2, params=params22, parametrization="mean")
mod22

# Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
params22s <- c(0.36, 0.121, 0.484, 0.072, 0.223, 0.059, -0.151, 0.395,
  0.406, -0.005, 0.083, 0.299, 0.218, 0.02, -0.119, 0.722, 0.093, 0.032,
  0.044, 0.191, 0.057, 0.172, -0.46, 0.016, 3.518, 5.154, 0.58)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
mod22s <- GSMVAR(gdpdef, p=2, M=2, params=params22s,
 structural_pars=list(W=W_22))
mod22s
```

`gsmvar_to_sgsmvar` *Switch from two-regime reduced form GMVAR, StMVAR, or G-StMVAR model to a structural model.*

Description

`gsmvar_to_sgsmvar` constructs SGMVAR, SStMVAR, or SG-StMVAR model based on a reduced form GMVAR, StMVAR, or G-StMVAR model.

Usage

```
gsmvar_to_sgsmvar(gsmvar, calc_std_errors = TRUE, cholesky = FALSE)
```

Arguments

`gsmvar`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.
`calc_std_errors`	should approximate standard errors be calculated?
`cholesky`	if `M == 1`, should the lower triangular Cholesky identification be employed? See details for using Cholesky identification with `M > 1`.

Details

The switch is made by simultaneously diagonalizing the two error term covariance matrices with a well known matrix decomposition (Muirhead, 1982, Theorem A9.9) and then normalizing the diagonal of the matrix W positive (which implies positive diagonal of the B-matrix). Models with more that two regimes are not supported because the matrix decomposition does not generally exists for more than two covariance matrices. If the model has only one regime (= regular SVAR model), a symmetric and pos. def. square root matrix of the error term covariance matrix is used **unless** `cholesky = TRUE` is set in the arguments, in which case Cholesky identification is employed.

In order to employ a structural model with Cholesky identification and multiple regimes (`M > 1`), use the function `GIRF` directly with a reduced form model (see `?GIRF`).

The columns of W as well as the lambda parameters can be re-ordered (without changing the implied reduced form model) afterwards with the function `reorder_W_columns`. Also all signs in any column of W can be swapped (without changing the implied reduced form model) afterwards with the function `swap_W_signs`. These two functions work with models containing any number of regimes.

Value

Returns an object of class `'gsmvar'` defining a structural GMVAR, StMVAR, or G-StMVAR model based on a two-regime reduced form GMVAR, StMVAR, or G-StMVAR model, with the main diagonal of the B-matrix normalized to be positive.

References

- Muirhead R.J. 1982. Aspects of Multivariate Statistical Theory, *Wiley*.
- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

fitGSMVAR, GSMVAR, GIRF, reorder_W_columns, swap_W_signs, stmvar_to_gstmvar

Examples

```
# Reduced form GMVAR(1,2) model
params12 <- c(0.55, 0.112, 0.344, 0.055, -0.009, 0.718, 0.319,
 0.005, 0.03, 0.619, 0.173, 0.255, 0.017, -0.136, 0.858, 1.185,
 -0.012, 0.136, 0.674)
mod12 <- GSMVAR(gdpdef, p=1, M=2, params=params12)

# Form a structural model based on the reduced form model:
mod12s <- gsmvar_to_sgsmvar(mod12)
mod12s

#' # Reduced form StMVAR(1,2) model
mod12t <- GSMVAR(gdpdef, p=1, M=2, params=c(params12, 11, 12), model="StMVAR")

# Form a structural model based on the reduced form model:
mod12ts <- gsmvar_to_sgsmvar(mod12t)
mod12ts
```

---

in_paramspace *Determine whether the parameter vector lies in the parameter space*

Description

in_paramspace checks whether the given parameter vector lies in the parameter space. Does NOT test the identification conditions!

Usage

```
in_paramspace(
  p,
  M,
  d,
  params,
  model = c("GMVAR", "StMVAR", "G-StMVAR"),
  constraints = NULL,
  same_means = NULL,
  weight_constraints = NULL,
  structural_pars = NULL,
  stat_tol = 0.001,
  posdef_tol = 1e-08,
  df_tol = 1e-08
)
```

Arguments

p	a positive integer specifying the autoregressive order of the model.
M	**For GMVAR and StMVAR models:** a positive integer specifying the number of mixture components. **For G-StMVAR models:** a size (2×1) integer vector specifying the number of *GMVAR type* components M1 in the first element and *StMVAR type* components M2 in the second element. The total number of mixture components is M=M1+M2.
d	the number of time series in the system.
params	a real valued vector specifying the parameter values. **For unconstrained models:** Should be size $((M(pd^2 + d + d(d+1)/2 + 2) - M1 - 1)x1)$ and have the form θ=(υ_1, ...,υ_M, $\alpha_1, ..., \alpha_{M-1}$,$\nu$), where • $\upsilon_m = (\phi_{m,0}, \phi_m, \sigma_m)$ • $\phi_m = (vec(A_{m,1}), ..., vec(A_{m,p})$ • and $\sigma_m = vech(\Omega_m)$, m=1,...,M, • $\nu = (\nu_{M1+1}, ..., \nu_M)$ • $M1$ is the number of GMVAR type regimes. **For constrained models:** Should be size $((M(d+d(d+1)/2+2)+q-M1-1)x1)$ and have the form $\theta = (\phi_{1,0}, ..., \phi_{M,0}, \psi, \sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1}, \nu)$, where • ψ $(qx1)$ satisfies $(\phi_1, ..., \phi_M) = C\psi$ where C is a (Mpd^2xq) constraint matrix. **For same_means models:** Should have the form $\theta = (\mu, \psi, \sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1}, \nu)$, where • $\mu = (\mu_1, ..., \mu_g)$ where μ_i is the mean parameter for group i and g is the number of groups. • If AR constraints are employed, ψ is as for constrained models, and if AR constraints are not employed, $\psi = (\phi_1, ..., \phi_M)$. **For models with weight_constraints:** Drop $\alpha_1, ..., \alpha_{M-1}$ from the parameter vector. **For structural models:** Reduced form models can be directly used as recursively identified structural models. If the structural model is identified by conditional heteroskedasticity, the parameter vector should have the form $\theta = (\phi_{1,0}, ..., \phi_{M,0}, \phi_1, ..., \phi_M, vec(W), \lambda_2, ..., \lambda_M, \alpha_1, ..., \alpha_{M-1}, \nu)$, where • $\lambda_m = (\lambda_{m1}, ..., \lambda_{md})$ contains the eigenvalues of the mth mixture component. **If AR parameters are constrained:** Replace $\phi_1, ..., \phi_M$ with ψ $(qx1)$ that satisfies $(\phi_1, ..., \phi_M) = C\psi$, as above. **If same_means:** Replace $(\phi_{1,0}, ..., \phi_{M,0})$ with $(\mu_1, ..., \mu_g)$, as above. **If W is constrained:** Remove the zeros from $vec(W)$ and make sure the other entries satisfy the sign constraints. **If λ_{mi} are constrained via C_lambda:** Replace $\lambda_2, ..., \lambda_M$ with γ $(rx1)$ that satisfies $(\lambda_2, ..., \lambda_M) = C_\lambda \gamma$ where C_λ is a $(d(M-1)xr)$ constraint matrix.

If λ_{mi} are constrained via `fixed_lambdas`: Drop $\lambda_2, ..., \lambda_M$ from the parameter vector.

Above, $\phi_{m,0}$ is the intercept parameter, $A_{m,i}$ denotes the ith coefficient matrix of the mth mixture component, Ω_m denotes the error term covariance matrix of the m:th mixture component, and α_m is the mixing weight parameter. The W and λ_{mi} are structural parameters replacing the error term covariance matrices (see Virolainen, 2022). If $M = 1$, α_m and λ_{mi} are dropped. If `parametrization=="mean"`, just replace each $\phi_{m,0}$ with regimewise mean μ_m. $vec()$ is vectorization operator that stacks columns of a given matrix into a vector. $vech()$ stacks columns of a given matrix from the principal diagonal downwards (including elements on the diagonal) into a vector.

In the **GMVAR model**, $M1 = M$ and ν is dropped from the parameter vector. In the **StMVAR** model, $M1 = 0$. In the **G-StMVAR** model, the first `M1` regimes are *GMVAR type* and the rest `M2` regimes are *StMVAR type*. In **StMVAR** and **G-StMVAR** models, the degrees of freedom parameters in ν should be strictly larger than two.

The notation is similar to the cited literature.

`model` is "GMVAR", "StMVAR", or "G-StMVAR" model considered? In the G-StMVAR model, the first `M1` components are GMVAR type and the rest `M2` components are StMVAR type.

`constraints` a size $(Mpd^2 \times q)$ constraint matrix C specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, ..., \phi_M) = C\psi$, where $\phi_m = (vec(A_{m,1}), ..., vec(A_{m,p}))(pd^2 x 1), m = 1, ..., M$, contains the coefficient matrices and ψ $(qx1)$ contains the related parameters. For example, to restrict the AR-parameters to be the same for all regimes, set C= `[I:...:I]'` $(Mpd^2 \times pd^2)$ where `I = diag(p*d^2)`. Ignore (or set to `NULL`) if linear constraints should **not** be employed.

`same_means` Restrict the mean parameters of some regimes to be the same? Provide a list of numeric vectors such that each numeric vector contains the regimes that should share the common mean parameters. For instance, if `M=3`, the argument `list(1, 2:3)` restricts the mean parameters of the second and third regime to be the same but the first regime has freely estimated (unconditional) mean. Ignore or set to `NULL` if mean parameters should not be restricted to be the same among any regimes. **This constraint is available only for mean parametrized models; that is, when `parametrization="mean"`.**

`weight_constraints`

a numeric vector of length $M - 1$ specifying fixed parameter values for the mixing weight parameters α_m, $m = 1, ..., M - 1$. Each element should be strictly between zero and one, and the sum of all the elements should be strictly less than one.

`structural_pars`

If `NULL` a reduced form model is considered. Reduced models can be used directly as recursively identified structural models. For a structural model identified by conditional heteroskedasticity, should be a list containing at least the first one of the following elements:

- `W` - a (dxd) matrix with its entries imposing constraints on W: `NA` indicating that the element is unconstrained, a positive value indicating strict positive

sign constraint, a negative value indicating strict negative sign constraint, and zero indicating that the element is constrained to zero.

- `C_lambda` - a $(d(M-1)\times r)$ constraint matrix that satisfies $(\lambda_2, ..., \lambda_M) = C_\lambda\gamma$ where γ is the new $(r \times 1)$ parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of `C_lambda` must be either **positive** or **zero**. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.
- `fixed_lambdas` - a length $d(M-1)$ numeric vector $(\lambda_2, ..., \lambda_M)$ with elements strictly larger than zero specifying the fixed parameter values for the parameters λ_{mi} should be constrained to. This constraint is alternative `C_lambda`. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.

See Virolainen (forthcoming) for the conditions required to identify the shocks and for the B-matrix as well (it is W times a time-varying diagonal matrix with positive diagonal entries).

`stat_tol` numerical tolerance for stationarity of the AR parameters: if the "bold A" matrix of any regime has eigenvalues larger that `1 - stat_tol` the model is classified as non-stationary. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.

`posdef_tol` numerical tolerance for positive definiteness of the error term covariance matrices: if the error term covariance matrix of any regime has eigenvalues smaller than this, the model is classified as not satisfying positive definiteness assumption. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.

`df_tol` the parameter vector is considered to be outside the parameter space if all degrees of freedom parameters are not larger than `2 + df_tol`.

Value

Returns `TRUE` if the given parameter vector lies in the parameter space and `FALSE` otherwise.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. (forthcoming). A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*.
- Virolainen S. 2022. Gaussian and Student's t mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks in the Euro area. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

Examples

```
# GMVAR(1,1), d=2 model:
params11 <- c(1.07, 127.71, 0.99, 0.00, -0.01, 0.99, 4.05,
  2.22, 8.87)
```

```
in_paramspace(p=1, M=1, d=2, params=params11)

# GMVAR(2,2), d=2 model:
params22 <- c(1.39, -0.77, 1.31, 0.14, 0.09, 1.29, -0.39,
 -0.07, -0.11, -0.28, 0.92, -0.03, 4.84, 1.01, 5.93, 1.25,
  0.08, -0.04, 1.27, -0.27, -0.07, 0.03, -0.31, 5.85, 3.57,
  9.84, 0.74)
in_paramspace(p=2, M=2, d=2, params=params22)

# GMVAR(2,2), d=2 model with AR-parameters restricted to be
# the same for both regimes:
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
params22c <- c(1.03, 2.36, 1.79, 3.00, 1.25, 0.06,0.04,
 1.34, -0.29, -0.08, -0.05, -0.36, 0.93, -0.15, 5.20,
 5.88, 3.56, 9.80, 0.37)
in_paramspace(p=2, M=2, d=2, params=params22c, constraints=C_mat)

# Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
params22s <- c(1.03, 2.36, 1.79, 3, 1.25, 0.06, 0.04, 1.34, -0.29,
 -0.08, -0.05, -0.36, 1.2, 0.05, 0.05, 1.3, -0.3, -0.1, -0.05, -0.4,
  0.89, 0.72, -0.37, 2.16, 7.16, 1.3, 0.37)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
in_paramspace(p=2, M=2, d=2, params=params22s,
  structural_pars=list(W=W_22))
```

`in_paramspace_int` *Determine whether the parameter vector lies in the parameter space*

Description

`in_paramspace_int` checks whether the parameter vector lies in the parameter space.

Usage

```
in_paramspace_int(
  p,
  M,
  d,
  params,
  model = c("GMVAR", "StMVAR", "G-StMVAR"),
  all_boldA,
  alphas,
  all_Omega,
  W_constraints = NULL,
  stat_tol = 0.001,
  posdef_tol = 1e-08,
  df_tol = 1e-08
)
```

Arguments

p	a positive integer specifying the autoregressive order of the model.
M	**For GMVAR and StMVAR models:** a positive integer specifying the number of mixture components. **For G-StMVAR models:** a size (2×1) integer vector specifying the number of *GMVAR type* components `M1` in the first element and *StMVAR type* components `M2` in the second element. The total number of mixture components is `M=M1+M2`.
d	the number of time series in the system.
params	a real valued vector specifying the parameter values. **For unconstrained models:** Should be size $((M(pd^2 + d + d(d+1)/2 + 2) - M1 - 1)x1)$ and have the form θ=(υ_1, ...,υ_M, $\alpha_1, ..., \alpha_{M-1}$,$\nu$), where • $\upsilon_m = (\phi_{m,0}, \phi_m, \sigma_m)$ • $\phi_m = (vec(A_{m,1}), ..., vec(A_{m,p})$ • and $\sigma_m = vech(\Omega_m)$, m=1,...,M, • $\nu = (\nu_{M1+1}, ..., \nu_M)$ • $M1$ is the number of GMVAR type regimes. **For constrained models:** Should be size $((M(d+d(d+1)/2+2)+q-M1-1)x1)$ and have the form $\theta = (\phi_{1,0}, ..., \phi_{M,0}, \psi, \sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1}, \nu)$, where • ψ $(qx1)$ satisfies $(\phi_1, ..., \phi_M) = C\psi$ where C is a (Mpd^2xq) constraint matrix. **For same_means models:** Should have the form $\theta = (\mu, \psi, \sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1}, \nu)$, where • $\mu = (\mu_1, ..., \mu_g)$ where μ_i is the mean parameter for group i and g is the number of groups. • If AR constraints are employed, ψ is as for constrained models, and if AR constraints are not employed, $\psi = (\phi_1, ..., \phi_M)$. **For models with weight_constraints:** Drop $\alpha_1, ..., \alpha_{M-1}$ from the parameter vector. **For structural models:** Reduced form models can be directly used as recursively identified structural models. If the structural model is identified by conditional heteroskedasticity, the parameter vector should have the form $\theta = (\phi_{1,0}, ..., \phi_{M,0}, \phi_1, ..., \phi_M, vec(W), \lambda_2, ..., \lambda_M, \alpha_1, ..., \alpha_{M-1}, \nu)$, where • $\lambda_m = (\lambda_{m1}, ..., \lambda_{md})$ contains the eigenvalues of the mth mixture component. **If AR parameters are constrained:** Replace $\phi_1, ..., \phi_M$ with ψ $(qx1)$ that satisfies $(\phi_1, ..., \phi_M) = C\psi$, as above. **If same_means:** Replace $(\phi_{1,0}, ..., \phi_{M,0})$ with $(\mu_1, ..., \mu_g)$, as above. **If W is constrained:** Remove the zeros from $vec(W)$ and make sure the other entries satisfy the sign constraints. **If λ_{mi} are constrained via `C_lambda`:** Replace $\lambda_2, ..., \lambda_M$ with γ $(rx1)$ that satisfies $(\lambda_2, ..., \lambda_M) = C_\lambda \gamma$ where C_λ is a $(d(M-1)xr)$ constraint matrix.

If λ_{mi} are constrained via `fixed_lambdas`: Drop $\lambda_2, ..., \lambda_M$ from the parameter vector.

Above, $\phi_{m,0}$ is the intercept parameter, $A_{m,i}$ denotes the ith coefficient matrix of the mth mixture component, Ω_m denotes the error term covariance matrix of the m:th mixture component, and α_m is the mixing weight parameter. The W and λ_{mi} are structural parameters replacing the error term covariance matrices (see Virolainen, 2022). If $M = 1$, α_m and λ_{mi} are dropped. If `parametrization=="mean"`, just replace each $\phi_{m,0}$ with regimewise mean μ_m. $vec()$ is vectorization operator that stacks columns of a given matrix into a vector. $vech()$ stacks columns of a given matrix from the principal diagonal downwards (including elements on the diagonal) into a vector.

In the **GMVAR model**, $M1 = M$ and ν is dropped from the parameter vector. In the **StMVAR** model, $M1 = 0$. In the **G-StMVAR** model, the first `M1` regimes are *GMVAR type* and the rest `M2` regimes are *StMVAR type*. In **StMVAR** and **G-StMVAR** models, the degrees of freedom parameters in ν should be strictly larger than two.

The notation is similar to the cited literature.

`model` is "GMVAR", "StMVAR", or "G-StMVAR" model considered? In the G-StMVAR model, the first `M1` components are GMVAR type and the rest `M2` components are StMVAR type.

`all_boldA` 3D array containing the $((dp) \times (dp))$ "bold A" matrices related to each mixture component VAR-process, obtained from `form_boldA`. Will be computed if not given.

`alphas` $(M \times 1)$ vector containing all mixing weight parameters, obtained from `pick_alphas`.

`all_Omega` 3D array containing all covariance matrices Ω_m, obtained from `pick_Omegas`.

`W_constraints` set `NULL` for reduced form models. For structural models, this should be the constraint matrix W from the list of structural parameters.

`stat_tol` numerical tolerance for stationarity of the AR parameters: if the "bold A" matrix of any regime has eigenvalues larger that `1 - stat_tol` the model is classified as non-stationary. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.

`posdef_tol` numerical tolerance for positive definiteness of the error term covariance matrices: if the error term covariance matrix of any regime has eigenvalues smaller than this, the model is classified as not satisfying positive definiteness assumption. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.

`df_tol` the parameter vector is considered to be outside the parameter space if all degrees of freedom parameters are not larger than `2 + df_tol`.

Details

The parameter vector in the argument `params` should be unconstrained and it is used for structural models only.

Value

Returns TRUE if the given parameter values are in the parameter space and FALSE otherwise. This function does NOT consider the identifiability condition!

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. (forthcoming). A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*.
- Virolainen S. 2022. Gaussian and Student's t mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks in the Euro area. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

---

iterate_more *Maximum likelihood estimation of a GMVAR, StMVAR, or G-StMVAR model with preliminary estimates*

Description

iterate_more uses a variable metric algorithm to finalize maximum likelihood estimation of a GMVAR, StMVAR, or G-StMVAR model (object of class 'gsmvar') which already has preliminary estimates.

Usage

```
iterate_more(
  gsmvar,
  maxit = 100,
  calc_std_errors = TRUE,
  custom_h = NULL,
  stat_tol = 0.001,
  posdef_tol = 1e-08,
  df_tol = 1e-08
)
```

Arguments

gsmvar	an object of class 'gsmvar', typically created with fitGSMVAR or GSMVAR.
maxit	the maximum number of iterations in the variable metric algorithm.
calc_std_errors	calculate approximate standard errors for the estimates?

`custom_h` A numeric vector with same the length as the parameter vector: i:th element of custom_h is the difference used in central difference approximation for partial differentials of the log-likelihood function for the i:th parameter. If `NULL` (default), then the difference used for differentiating overly large degrees of freedom parameters is adjusted to avoid numerical problems, and the difference is `6e-6` for the other parameters.

`stat_tol` numerical tolerance for stationarity of the AR parameters: if the "bold A" matrix of any regime has eigenvalues larger that `1 - stat_tol` the model is classified as non-stationary. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.

`posdef_tol` numerical tolerance for positive definiteness of the error term covariance matrices: if the error term covariance matrix of any regime has eigenvalues smaller than this, the model is classified as not satisfying positive definiteness assumption. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.

`df_tol` the parameter vector is considered to be outside the parameter space if all degrees of freedom parameters are not larger than `2 + df_tol`.

Details

The purpose of `iterate_more` is to provide a simple and convenient tool to finalize the estimation when the maximum number of iterations is reached when estimating a GMVAR, StMVAR, or G-StMVAR model with the main estimation function `fitGSMVAR`. `iterate_more` is essentially a wrapper around the function `optim` from the package `stats` and `GSMVAR` from the package `gmvarkit`.

Value

Returns an object of class `'gsmvar'` defining the estimated GMVAR, StMVAR, or G-StMVAR model.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Kalliovirta L. and Saikkonen P. 2010. Reliable Residuals for Multivariate Nonlinear Time Series Models. *Unpublished Revision of HECER Discussion Paper No. 247*.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

`fitGSMVAR`, `GSMVAR`, `optim`, `profile_logliks`, `update_numtols`

Examples

```
## These are long running examples that use parallel computing!
## Running the below examples takes approximately 2 minutes

# GMVAR(1,2) model, only 5 iterations of the variable metric
# algorithm
fit12 <- fitGSMVAR(gdpdef, p=1, M=2, ncalls=1, maxit=5, seeds=1)
fit12

# Iterate more:
fit12_2 <- iterate_more(fit12)
fit12_2
```

`linear_IRF` *Estimate linear impulse response function based on a single regime of a structural GMVAR, StMVAR, or G-StMVAR model.*

Description

`linear_IRF` estimates linear impulse response function based on a single regime of a structural GMVAR, StMVAR, or G-StMVAR model.

Usage

```
linear_IRF(
  gsmvar,
  N = 30,
  regime = 1,
  which_cumulative = numeric(0),
  scale = NULL,
  ci = NULL,
  bootstrap_reps = 100,
  ncores = 2,
  ncalls = 1,
  seeds = NULL,
  ...
)

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

## S3 method for class 'irf'
print(x, ..., digits = 2, N_to_print, shocks_to_print)
```

Arguments

gsmvar	an object of class 'gsmvar' defining a structural or reduced form GSMVAR model. For a reduced form model, the shocks are automatically identified by the lower triangular Cholesky decomposition.
N	a positive integer specifying the horizon how far ahead should the linear impulse responses be calculated.
regime	Based on which regime the linear IRF should be calculated? An integer in $1, ..., M$.
which_cumulative	a numeric vector with values in $1, ..., d$ (d=ncol(data)) specifying which the variables for which the linear impulse responses should be cumulative. Default is none.
scale	should the linear IRFs to some of the shocks be scaled so that they correspond to a specific instantaneous response of some specific variable? Provide a length three vector where the shock of interest is given in the first element (an integer in $1, ..., d$), the variable of interest is given in the second element (an integer in $1, ..., d$), and its instantaneous response in the third element (a non-zero real number). If the linear IRFs of multiple shocks should be scaled, provide a matrix which has one column for each of the shocks with the columns being the length three vectors described above.
ci	a real number in $(0, 1)$ specifying the confidence level of the confidence intervals calculated via a fixed-design wild residual bootstrap method. Available only for models that impose linear autoregressive dynamics (excluding changes in the volatility regime).
bootstrap_reps	the number of bootstrap repetitions for estimating confidence bounds.
ncores	the number of CPU cores to be used in parallel computing when bootstrapping confidence bounds.
ncalls	on how many estimation rounds should each bootstrap estimation be based on? Does not have to be very large since initial estimates used are based on the initially fitted model. Larger number of rounds gives more reliable results but is computationally more demanding.
seeds	a numeric vector of length bootstrap_reps initializing the seed for the random generator for each bootstrap replication.
...	currently not used.
x	object of class 'irf' generated by the function linear_IRF.
shocks_to_plot	IRFs of which shocks should be plotted? A numeric vector with elements in 1,...,d.
digits	the number of decimals to print
N_to_print	an integer specifying the horizon how far to print the estimates and confidence intervals. The default is that all the values are printed.
shocks_to_print	the responses to which should should be printed? A numeric vector with elements in 1,...,d. The default is that responses to all the shocks are printed.

Details

The model DOES NOT need to be structural in order for this function to be applicable. When an identified structural GMVAR, StMVAR, or G-StMVAR model is provided in the argument `gsmvar`, the identification imposed by the model is used. When a reduced form model is provided in the argument `gsmvar`, lower triangular Cholesky identification is used to identify the shocks.

If the autoregressive dynamics of the model are linear (i.e., either M == 1 or mean and AR parameters are constrained identical across the regimes), confidence bounds can be calculated based on a type of fixed-design wild residual bootstrap method. See Virolainen (forthcoming) for a related discussion. We employ the method described in Herwartz and Lütkepohl (2014); see also the relevant chapters in Kilian and Lütkepohl (2017).

Value

Returns a class `'irf'` list with the following elements:

`$point_est`**:** a 3D array `[variables, shock, horizon]` containing the point estimates of the IRFs. Note that the first slice is for the impact responses and the slice i+1 for the period i. The response of the variable 'i1' to the shock 'i2' is subsetted as `$point_est[i1, i2, ]`.

`$conf_ints`**:** bootstrapped confidence intervals for the IRFs in a `[variables, shock, horizon, bound]` 4D array. The lower bound is obtained as `$conf_ints[, , , 1]`, and similarly the upper bound as `$conf_ints[, , , 2]`. The subsetted 3D array is then the bound in a form similar to `$point_est`.

`$all_bootstrap_reps`**:** IRFs from all of the bootstrap replications in a `[variables, shock, horizon, rep]`. 4D array. The IRF from replication i1 is obtained as `$all_bootstrap_reps[, , , i1]`, and the subsetted 3D array is then the in a form similar to `$point_est`.

Other elements: contains some of the arguments the `linear_IRF` was called with.

Functions

- `plot(irf)`: plot method
- `print(irf)`: print method

References

- Herwartz H. and Lütkepohl H. 2014. Structural vector autoregressions with Markov switching: Combining conventional with statistical identification of shocks. *Journal of Econometrics*, 183, pp. 104-116.
- Kilian L. and Lütkepohl H. 2017. Structural Vectors Autoregressive Analysis. *Cambridge University Press*, Cambridge.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*, **43**:1, 44-54.

See Also

`GIRF`, `GFEVD`, `fitGSMVAR`, `GSMVAR`, `gsmvar_to_sgsmvar`, `reorder_W_columns`, `swap_W_signs`

Examples

```
# These are long running examples that take a few minutes to run

## GMVAR, p=5, M=2, d=2 model with linear AR dynamics.
# recursive identification, IRF based on the first regime:
params52cm <- c(0.788, 0.559, 0.277, 0.038, -0.061, 0.463, 0.286, 0,
              0.035, 0.161, -0.112, 0.031, -0.313, 0.183, 0.103, 0.014,
              0.002, 0.143, -0.089, -0.013, 0.182, -0.04, 1.3, 0.008,
              0.139, 0.277, -0.005, 0.032, 0.118)
mod52cm <- GSMVAR(data=gdpdef, p=5, M=2, params=params52cm,
                  constraints=rbind(diag(5*2^2), diag(5*2^2)),
                  same_means=list(1:2), parametrization="mean")
irf1 <- linear_IRF(mod52cm, regime=1, N=20, scale=cbind(c(1, 1, 1), c(2, 2, 1)))
print(irf1, digits=3)
plot(irf1)

# Identification by heteroskedasticity, bootstrapped confidence intervals and
# and scaled instantaneous effects of the shocks. Note that in actual
# empirical application, a larger number of bootstrap reps should be used.
mod52cms <- gsmvar_to_sgsmvar(mod52cm)
irf2 <- linear_IRF(mod52cms, regime=1, N=20, ci=0.68, bootstrap_reps=10,
                   ncalls=1, seeds=1:10, ncores=1)
plot(irf2)
```

`loglikelihood` *Compute log-likelihood of a GMVAR, StMVAR, or G-StMVAR model using parameter vector*

Description

`loglikelihood` computes log-likelihood of a GMVAR, StMVAR, or G-StMVAR model using parameter vector instead of an object of class 'gsmvar'. Exists for convenience if one wants to for example employ other estimation algorithms than the ones used in `fitGSMVAR`. Use `minval` to control what happens when the parameter vector is outside the parameter space.

Usage

```
loglikelihood(
  data,
  p,
  M,
  params,
  model = c("GMVAR", "StMVAR", "G-StMVAR"),
  conditional = TRUE,
  parametrization = c("intercept", "mean"),
  constraints = NULL,
  same_means = NULL,
```

```
  weight_constraints = NULL,
  structural_pars = NULL,
  minval = NA,
  stat_tol = 0.001,
  posdef_tol = 1e-08,
  df_tol = 1e-08
)
```

Arguments

data	a matrix or class 'ts' object with d>1 columns. Each column is taken to represent a univariate time series. NA values are not supported.
p	a positive integer specifying the autoregressive order of the model.
M	**For GMVAR and StMVAR models:** a positive integer specifying the number of mixture components. **For G-StMVAR models:** a size (2×1) integer vector specifying the number of *GMVAR type* components M1 in the first element and *StMVAR type* components M2 in the second element. The total number of mixture components is M=M1+M2.
params	a real valued vector specifying the parameter values. **For unconstrained models:** Should be size $((M(pd^2+d+d(d+1)/2+2)-M1-1)x1)$ and have the form θ=(υ_1, ...,υ_M, $\alpha_1, ..., \alpha_{M-1}$,$\nu$), where • $\upsilon_m = (\phi_{m,0}, \phi_m, \sigma_m)$ • $\phi_m = (vec(A_{m,1}), ..., vec(A_{m,p}))$ • and $\sigma_m = vech(\Omega_m)$, m=1,...,M, • $\nu = (\nu_{M1+1}, ..., \nu_M)$ • $M1$ is the number of GMVAR type regimes. **For constrained models:** Should be size $((M(d+d(d+1)/2+2)+q-M1-1)x1)$ and have the form $\theta = (\phi_{1,0}, ..., \phi_{M,0}, \psi, \sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1}, \nu)$, where • ψ $(qx1)$ satisfies $(\phi_1, ..., \phi_M) = C\psi$ where C is a (Mpd^2xq) constraint matrix. **For same_means models:** Should have the form $\theta = (\mu, \psi, \sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1}, \nu)$, where • $\mu = (\mu_1, ..., \mu_g)$ where μ_i is the mean parameter for group i and g is the number of groups. • If AR constraints are employed, ψ is as for constrained models, and if AR constraints are not employed, $\psi = (\phi_1, ..., \phi_M)$. **For models with weight_constraints:** Drop $\alpha_1, ..., \alpha_{M-1}$ from the parameter vector. **For structural models:** Reduced form models can be directly used as recursively identified structural models. If the structural model is identified by conditional heteroskedasticity, the parameter vector should have the form $\theta = (\phi_{1,0}, ..., \phi_{M,0}, \phi_1, ..., \phi_M, vec(W), \lambda_2, ..., \lambda_M, \alpha_1, ..., \alpha_{M-1}, \nu)$, where • $\lambda_m = (\lambda_{m1}, ..., \lambda_{md})$ contains the eigenvalues of the mth mixture component.

If AR parameters are constrained: Replace $\phi_1, ..., \phi_M$ with ψ $(qx1)$ that satisfies $(\phi_1, ..., \phi_M) = C\psi$, as above.

If same_means: Replace $(\phi_{1,0}, ..., \phi_{M,0})$ with $(\mu_1, ..., \mu_g)$, as above.

If W is constrained: Remove the zeros from $vec(W)$ and make sure the other entries satisfy the sign constraints.

If λ_{mi} are constrained via `C_lambda`: Replace $\lambda_2, ..., \lambda_M$ with γ $(rx1)$ that satisfies $(\lambda_2, ..., \lambda_M) = C_\lambda\gamma$ where C_λ is a $(d(M-1)xr)$ constraint matrix.

If λ_{mi} are constrained via `fixed_lambdas`: Drop $\lambda_2, ..., \lambda_M$ from the parameter vector.

Above, $\phi_{m,0}$ is the intercept parameter, $A_{m,i}$ denotes the ith coefficient matrix of the mth mixture component, Ω_m denotes the error term covariance matrix of the m:th mixture component, and α_m is the mixing weight parameter. The W and λ_{mi} are structural parameters replacing the error term covariance matrices (see Virolainen, 2022). If $M = 1$, α_m and λ_{mi} are dropped. If `parametrization=="mean"`, just replace each $\phi_{m,0}$ with regimewise mean μ_m. $vec()$ is vectorization operator that stacks columns of a given matrix into a vector. $vech()$ stacks columns of a given matrix from the principal diagonal downwards (including elements on the diagonal) into a vector.

In the **GMVAR model**, $M1 = M$ and ν is dropped from the parameter vector. In the **StMVAR** model, $M1 = 0$. In the **G-StMVAR** model, the first `M1` regimes are *GMVAR type* and the rest `M2` regimes are *StMVAR type*. In **StMVAR** and **G-StMVAR** models, the degrees of freedom parameters in ν should be strictly larger than two.

The notation is similar to the cited literature.

`model` is "GMVAR", "StMVAR", or "G-StMVAR" model considered? In the G-StMVAR model, the first `M1` components are GMVAR type and the rest `M2` components are StMVAR type.

`conditional` a logical argument specifying whether the conditional or exact log-likelihood function should be used.

`parametrization`

`"intercept"` or `"mean"` determining whether the model is parametrized with intercept parameters $\phi_{m,0}$ or regime means μ_m, m=1,...,M.

`constraints` a size $(Mpd^2 \times q)$ constraint matrix C specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, ..., \phi_M) = C\psi$, where $\phi_m = (vec(A_{m,1}), ..., vec(A_{m,p})(pd^2x1), m = 1, ..., M$, contains the coefficient matrices and ψ $(qx1)$ contains the related parameters. For example, to restrict the AR-parameters to be the same for all regimes, set C= `[I:...:I]'` $(Mpd^2 \times pd^2)$ where `I = diag(p*d^2)`. Ignore (or set to `NULL`) if linear constraints should **not** be employed.

`same_means` Restrict the mean parameters of some regimes to be the same? Provide a list of numeric vectors such that each numeric vector contains the regimes that should share the common mean parameters. For instance, if `M=3`, the argument `list(1, 2:3)` restricts the mean parameters of the second and third regime to be the same but the first regime has freely estimated (unconditional) mean. Ignore or set to `NULL` if mean parameters should not be restricted to be the same among any

regimes. **This constraint is available only for mean parametrized models; that is, when** `parametrization="mean"`**.**

`weight_constraints`

a numeric vector of length $M-1$ specifying fixed parameter values for the mixing weight parameters $\alpha_m,\ m = 1, ..., M-1$. Each element should be strictly between zero and one, and the sum of all the elements should be strictly less than one.

`structural_pars`

If `NULL` a reduced form model is considered. Reduced models can be used directly as recursively identified structural models. For a structural model identified by conditional heteroskedasticity, should be a list containing at least the first one of the following elements:

- `W` - a (dxd) matrix with its entries imposing constraints on W: `NA` indicating that the element is unconstrained, a positive value indicating strict positive sign constraint, a negative value indicating strict negative sign constraint, and zero indicating that the element is constrained to zero.
- `C_lambda` - a $(d(M-1)\times r)$ constraint matrix that satisfies $(\lambda_2, ..., \lambda_M) = C_\lambda\gamma$ where γ is the new $(r \times 1)$ parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of `C_lambda` must be either **positive** or **zero**. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.
- `fixed_lambdas` - a length $d(M-1)$ numeric vector $(\lambda_2, ..., \lambda_M)$ with elements strictly larger than zero specifying the fixed parameter values for the parameters λ_{mi} should be constrained to. This constraint is alternative `C_lambda`. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.

See Virolainen (forthcoming) for the conditions required to identify the shocks and for the B-matrix as well (it is W times a time-varying diagonal matrix with positive diagonal entries).

`minval` the value that will be returned if the parameter vector does not lie in the parameter space (excluding the identification condition).

`stat_tol` numerical tolerance for stationarity of the AR parameters: if the "bold A" matrix of any regime has eigenvalues larger that `1 - stat_tol` the model is classified as non-stationary. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.

`posdef_tol` numerical tolerance for positive definiteness of the error term covariance matrices: if the error term covariance matrix of any regime has eigenvalues smaller than this, the model is classified as not satisfying positive definiteness assumption. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.

`df_tol` the parameter vector is considered to be outside the parameter space if all degrees of freedom parameters are not larger than `2 + df_tol`.

Details

`loglikelihood_int` takes use of the function `dmvn` from the package `mvnfast`.

Value

Returns log-likelihood if `params` is in the parameters space and `minval` if not.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lütkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.
- McElroy T. 2017. Computation of vector ARMA autocovariances. *Statistics and Probability Letters*, **124**, 92-96.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

`fitGSMVAR`, `GSMVAR`, `calc_gradient`

Examples

```
# GMVAR(2, 2), d=2 model;
params22 <- c(0.36, 0.121, 0.223, 0.059, -0.151, 0.395, 0.406, -0.005,
 0.083, 0.299, 0.215, 0.002, 0.03, 0.484, 0.072, 0.218, 0.02, -0.119,
 0.722, 0.093, 0.032, 0.044, 0.191, 1.101, -0.004, 0.105, 0.58)
loglikelihood(data=gdpdef, p=2, M=2, params=params22)

# Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
params22s <- c(0.36, 0.121, 0.484, 0.072, 0.223, 0.059, -0.151, 0.395,
 0.406, -0.005, 0.083, 0.299, 0.218, 0.02, -0.119, 0.722, 0.093, 0.032,
 0.044, 0.191, 0.057, 0.172, -0.46, 0.016, 3.518, 5.154, 0.58)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
loglikelihood(data=gdpdef, p=2, M=2, params=params22s, structural_pars=list(W=W_22))
```

LR_test — *Perform likelihood ratio test for a GMVAR, StMVAR, or G-StMVAR model*

Description

`LR_test` performs a likelihood ratio test for a GMVAR, StMVAR, or G-StMVAR model

Usage

```
LR_test(gsmvar1, gsmvar2)
```

Arguments

gsmvar1	an object of class 'gsmvar' generated by fitGSMVAR or GSMVAR, containing the **freely estimated** model.
gsmvar2	an object of class 'gsmvar' generated by fitGSMVAR or GSMVAR, containing the **constrained** model.

Details

Performs a likelihood ratio test, testing the null hypothesis that the true parameter value lies in the constrained parameter space. Under the null, the test statistic is asymptotically χ^2-distributed with k degrees of freedom, k being the difference in the dimensions of the unconstrained and constrained parameter spaces.

Note that this function does **not** verify that the two models are actually nested.

Value

A list with class "hypotest" containing the test results and arguments used to calculate the test.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. (forthcoming). A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*.
- Virolainen S. 2022. Gaussian and Student's t mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks in the Euro area. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

See Also

Wald_test, Rao_test, fitGSMVAR, GSMVAR, diagnostic_plot, profile_logliks, quantile_residual_tests, cond_moment_plot

Examples

```
## These are long running examples that use parallel computing!
## The below examples take around 1 minute to run.

# Structural GMVAR(2, 2), d=2 model with recursive identification
W22 <- matrix(c(1, NA, 0, 1), nrow=2, byrow=FALSE)
fit22s <- fitGSMVAR(gdpdef, p=2, M=2, structural_pars=list(W=W22),
                    ncalls=1, seeds=2)

# The same model but the AR coefficients restricted to be the same
# in both regimes:
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
fit22sc <- fitGSMVAR(gdpdef, p=2, M=2, constraints=C_mat,
                     structural_pars=list(W=W22), ncalls=1, seeds=1)
```

```
# Test the AR constraints with likelihood ratio test:
LR_test(fit22s, fit22sc)
```

`Pearson_residuals` *Calculate multivariate Pearson residuals of a GMVAR, StMVAR, or G-StMVAR model*

Description

`Pearson_residuals` calculates multivariate Pearson residuals for a GMVAR, StMVAR, or G-StMVAR model.

Usage

```
Pearson_residuals(gsmvar, standardize = TRUE)
```

Arguments

`gsmvar`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.
`standardize`	Should the residuals be standardized? Use `FALSE` to obtain raw residuals.

Value

Returns $((n_obs - p)xd)$ matrix containing the residuals, j:th column corresponds to the time series in the j:th column of the data.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Kalliovirta L. and Saikkonen P. 2010. Reliable Residuals for Multivariate Nonlinear Time Series Models. *Unpublished Revision of HECER Discussion Paper No. 247*.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

`fitGSMVAR`, `GSMVAR`, `quantile_residuals`, `diagnostic_plot`

Examples

```
# GMVAR(1,2), d=2 model:
params12 <- c(0.55, 0.112, 0.344, 0.055, -0.009, 0.718, 0.319, 0.005, 0.03,
 0.619, 0.173, 0.255, 0.017, -0.136, 0.858, 1.185, -0.012, 0.136, 0.674)
mod12 <- GSMVAR(gdpdef, p=1, M=2, params=params12)
Pearson_residuals(mod12, standardize=FALSE) # Raw residuals
Pearson_residuals(mod12, standardize=TRUE) # Standardized to identity cov.matrix.

# Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
params22s <- c(0.36, 0.121, 0.484, 0.072, 0.223, 0.059, -0.151, 0.395,
 0.406, -0.005, 0.083, 0.299, 0.218, 0.02, -0.119, 0.722, 0.093, 0.032,
 0.044, 0.191, 0.057, 0.172, -0.46, 0.016, 3.518, 5.154, 0.58)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
mod22s <- GSMVAR(gdpdef, p=2, M=2, params=params22s, structural_pars=list(W=W_22))
Pearson_residuals(mod22s, standardize=FALSE) # Raw residuals
Pearson_residuals(mod22s, standardize=TRUE) # Standardized to identity cov.matrix.
```

---

plot.gsmvarpred *plot method for class 'gsmvarpred' objects*

Description

`plot.gsmvarpred` is plot method for gsmvarpred objects.

Usage

```
## S3 method for class 'gsmvarpred'
plot(x, ..., nt, mix_weights = TRUE, add_grid = TRUE)
```

Arguments

`x`	object of class `'gsmvarpred'` generated by `predict.gsmvar`.
`...`	arguments passed to `grid` which plots grid to the figure.
`nt`	a positive integer specifying the number of observations to be plotted along with the prediction (ignored if `plot_res==FALSE`). Default is `round(nrow(data)*0.15)`.
`mix_weights`	`TRUE` if forecasts for mixing weights should be plotted, `FALSE` in not.
`add_grid`	should grid be added to the plots?

Details

This method is used plot forecasts of GSMVAR processes

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. (forthcoming). A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*.
- Virolainen S. 2022. Gaussian and Student's t mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks in the Euro area. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

`plot.qrtest` *Quantile residual tests*

Description

`quantile_residual_tests` performs quantile residual tests described by *Kalliovirta and Saikkonen 2010*, testing autocorrelation, conditional heteroskedasticity, and normality.

Usage

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

## S3 method for class 'qrtest'
print(x, ..., digits = 3)

quantile_residual_tests(
  gsmvar,
  lags_ac = c(1, 3, 6, 12),
  lags_ch = lags_ac,
  nsim = 1,
  ncores = 1,
  print_res = TRUE,
  stat_tol,
  posdef_tol,
  df_tol
)
```

Arguments

`x`	object of class `'qrtest'` generated by the function `quantile_residual_tests)`.
`...`	currently not used.
`digits`	the number of decimals to print
`gsmvar`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.

`lags_ac`	a positive integer vector specifying the lags used to test autocorrelation.
`lags_ch`	a positive integer vector specifying the lags used to test conditional heteroskedasticity.
`nsim`	to how many simulations should the covariance matrix Omega used in the qr-tests be based on? If smaller than sample size, then the covariance matrix will be evaluated from the sample. Larger number of simulations might improve the tests size properties but it increases the computation time.
`ncores`	the number of CPU cores to be used in numerical differentiation. Multiple cores are not supported on Windows, though.
`print_res`	should the test results be printed while computing the tests?
`stat_tol`	numerical tolerance for stationarity of the AR parameters: if the "bold A" matrix of any regime has eigenvalues larger that `1 - stat_tol` the model is classified as non-stationary. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.
`posdef_tol`	numerical tolerance for positive definiteness of the error term covariance matrices: if the error term covariance matrix of any regime has eigenvalues smaller than this, the model is classified as not satisfying positive definiteness assumption. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.
`df_tol`	the parameter vector is considered to be outside the parameter space if all degrees of freedom parameters are not larger than `2 + df_tol`.

Details

If the function fails to calculate the tests because of numerical problems and the parameter values are near the border of the parameter space, it might help to use smaller numerical tolerance for the stationarity and positeve definiteness conditions. The numerical tolerance of an existing model can be changed with the function `update_numtols` or you can set it directly with the arguments `stat_tol` and `posdef_tol`.

Value

Returns an object of class `'qrtest'` which has its own print method. The returned object is a list containing the quantile residual test results for normality, autocorrelation, and conditional heteroskedasticity. The autocorrelation and conditional heteroskedasticity results also contain the associated (vectorized) individual statistics divided by their standard errors (see *Kalliovirta and Saikkonen 2010*, s.17-20) under the label `$ind_stats`.

Functions

- `plot(qrtest)`: Plot p-values of the autocorrelation and conditional heteroskedasticity tests.
- `print(qrtest)`: Print method for class 'qrtest'

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.

- Kalliovirta L. and Saikkonen P. 2010. Reliable Residuals for Multivariate Nonlinear Time Series Models. *Unpublished Revision of HECER Discussion Paper No. 247*.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

fitGSMVAR, GSMVAR, quantile_residuals, GIRF, diagnostic_plot, predict.gsmvar, profile_logliks, LR_test, Wald_test, cond_moment_plot, update_numtols

Examples

```
# GMVAR(3,2) model
fit32 <- fitGSMVAR(gdpdef, p=3, M=2, ncalls=1, seeds=2)
qrtests32 <- quantile_residual_tests(fit32)
qrtests32
plot(qrtests32)

# Structural GMVAR(1,2) model identified with sign
# constraints and build with hand-specified parameter values.
# Tests based on simulation procedure with nsim=1000:
params12s <- c(0.55, 0.112, 0.619, 0.173, 0.344, 0.055, -0.009, 0.718,
 0.255, 0.017, -0.136, 0.858, 0.541, 0.057, -0.162, 0.162, 3.623,
 4.726, 0.674)
W_12 <- matrix(c(1, 1, -1, 1), nrow=2)
mod12s <- GSMVAR(gdpdef, p=1, M=2, params=params12s,
                 structural_pars=list(W=W_12))
qrtests12s <- quantile_residual_tests(mod12s, nsim=1000)
qrtests12s
```

---

predict.gsmvar *Predict method for class 'gsmvar' objects*

Description

predict.gsmvar is a predict method for class 'gsmvar' objects. The forecasts of the GMVAR, StMVAR, and G-StMVAR models are computed by performing independent simulations and using the sample medians or means as point forecasts and empirical quantiles as prediction intervals. For one-step-ahead predictions using the exact conditional mean is also supported.

Usage

```
## S3 method for class 'gsmvar'
predict(
  object,
```

```
  ...,
  n_ahead,
  nsim = 2000,
  pi = c(0.95, 0.8),
  pi_type = c("two-sided", "upper", "lower", "none"),
  pred_type = c("median", "mean", "cond_mean"),
  plot_res = TRUE,
  mix_weights = TRUE,
  nt
)
```

Arguments

object	an object of class 'gsmvar', typically created with fitGSMVAR or GSMVAR.
...	additional arguments passed to grid (ignored if plot_res==FALSE) which plots grid to the figure.
n_ahead	how many steps ahead should be predicted?
nsim	to how many independent simulations should the forecast be based on?
pi	a numeric vector specifying the confidence levels of the prediction intervals.
pi_type	should the prediction intervals be "two-sided", "upper", or "lower"?
pred_type	should the prediction be based on sample "median" or "mean"? Or should it be one-step-ahead forecast based on the exact conditional mean ("cond_mean")? Prediction intervals won't be calculated if the exact conditional mean is used.
plot_res	should the results be plotted?
mix_weights	TRUE if forecasts for mixing weights should be plotted, FALSE in not.
nt	a positive integer specifying the number of observations to be plotted along with the prediction (ignored if plot_res==FALSE). Default is round(nrow(data)*0.15).

Value

Returns a class 'gsmvarpred' object containing, among the specifications,...

$pred Point forecasts

$pred_int Prediction intervals, as [, , d].

$mix_pred Point forecasts for the mixing weights

mix_pred_int Individual prediction intervals for mixing weights, as [, , m], m=1,..,M.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. (forthcoming). A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*.
- Virolainen S. 2022. Gaussian and Student's t mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks in the Euro area. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

See Also

GIRF, GFEVD, simulate.gsmvar

Examples

```
# GMVAR(2, 2), d=2 model
params22 <- c(0.36, 0.121, 0.223, 0.059, -0.151, 0.395, 0.406, -0.005,
 0.083, 0.299, 0.215, 0.002, 0.03, 0.484, 0.072, 0.218, 0.02, -0.119,
  0.722, 0.093, 0.032, 0.044, 0.191, 1.101, -0.004, 0.105, 0.58)
mod22 <- GSMVAR(gdpdef, p=2, M=2, d=2, params=params22)
p1 <- predict(mod22, n_ahead=10, pred_type="median", nsim=500)
p1
p2 <- predict(mod22, n_ahead=10, nt=20, lty=1, nsim=500)
p2
p3 <- predict(mod22, n_ahead=10, pi=c(0.99, 0.90, 0.80, 0.70),
              nt=30, lty=0, nsim=500)
p3

# StMVAR(2, 2), d=2 model
params22t <- c(0.36, 0.121, 0.223, 0.059, -0.151, 0.395, 0.406, -0.005,
 0.083, 0.299, 0.215, 0.002, 0.03, 0.484, 0.072, 0.218, 0.02, -0.119,
  0.722, 0.093, 0.032, 0.044, 0.191, 1.101, -0.004, 0.105, 0.58, 3, 4)
mod22t <- GSMVAR(gdpdef, p=2, M=2, d=2, params=params22t, model="StMVAR")
p1 <- predict(mod22t, n_ahead=12, pred_type="median", nsim=500, pi=0.9)
p1
```

---

print.gsmvarpred *Print method for class 'gsmvarpred' objects*

Description

print.gsmvarpred is a print method for object generated by predict.gsmvar.

Usage

```
## S3 method for class 'gsmvarpred'
print(x, ..., digits = 2)
```

Arguments

x	object of class 'gsmvarpred' generated by predict.gsmvar.
...	currently not used.
digits	the number of decimals to print

Examples

```
# GMVAR(2, 2), d=2 model;
params22 <- c(0.36, 0.121, 0.223, 0.059, -0.151, 0.395, 0.406, -0.005,
 0.083, 0.299, 0.215, 0.002, 0.03, 0.484, 0.072, 0.218, 0.02, -0.119,
 0.722, 0.093, 0.032, 0.044, 0.191, 1.101, -0.004, 0.105, 0.58)
mod22 <- GSMVAR(gdpdef, p=2, M=2, params=params22)
pred22 <- predict(mod22, n_ahead=3, plot_res=FALSE)
print(pred22)
print(pred22, digits=3)
```

---

print.gsmvarsum *Summary print method from objects of class 'gsmvarsum'*

Description

`print.gsmvarsum` is a print method for object `'gsmvarsum'` generated by `summary.gsmvar`.

Usage

```
## S3 method for class 'gsmvarsum'
print(x, ..., digits)
```

Arguments

`x`	object of class 'gsmvarsum' generated by `summary.gsmvar`.
`...`	currently not used.
`digits`	the number of digits to be printed.

Examples

```
# GMVAR(2, 2), d=2 model;
params22 <- c(0.36, 0.121, 0.223, 0.059, -0.151, 0.395, 0.406, -0.005,
 0.083, 0.299, 0.215, 0.002, 0.03, 0.484, 0.072, 0.218, 0.02, -0.119,
 0.722, 0.093, 0.032, 0.044, 0.191, 1.101, -0.004, 0.105, 0.58)
mod22 <- GSMVAR(gdpdef, p=2, M=2, params=params22)
sumry22 <- summary(mod22)
print(sumry22)
```

print.hypotest *Print method for the class hypotest*

Description

`print.hypotest` is the print method for the class hypotest objects.

Usage

```
## S3 method for class 'hypotest'
print(x, ..., digits = 4)
```

Arguments

`x`	object of class `'hypotest'` generated by the function `Wald_test` or `LR_test`.
`...`	currently not in use.
`digits`	how many significant digits to print?

print_std_errors *Print standard errors of a GMVAR, StMVAR, or G-StMVAR model in the same form as the model estimates are printed*

Description

`print_std_errors` prints the approximate standard errors of a GMVAR, StMVAR, or G-StMVAR model in the same form as the parameters of objects of class `'gsmvar'` are printed.

Usage

```
print_std_errors(gsmvar, digits = 3)
```

Arguments

`gsmvar`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.
`digits`	how many digits should be printed?

Details

The main purpose of `print_std_errors` is to provide a convenient tool to match the standard errors to certain parameter estimates. Note that if the model is intercept parametrized, there won't be standard errors for the unconditional means, and vice versa. Also, there is no standard error for the last mixing weight alpha_M because it is not parametrized.

Note that if linear constraints are imposed and they involve summations or multiplications, then the AR parameter standard errors are printed separately as they don't correspond one-to-one to the model parameter standard errors.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Kalliovirta L. and Saikkonen P. 2010. Reliable Residuals for Multivariate Nonlinear Time Series Models. *Unpublished Revision of HECER Discussion Paper No. 247*.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

profile_logliks, fitGSMVAR, GSMVAR, print.gsmvar, swap_parametrization

Examples

```
# GMVAR(1,2) model
fit12 <- fitGSMVAR(gdpdef, p=1, M=2, ncalls=1, seeds=1)
fit12
print_std_errors(fit12)
```

---

profile_logliks *Plot profile log-likehoods around the estimates*

Description

profile_logliks plots profile log-likelihoods around the estimates.

Usage

```
profile_logliks(
  gsmvar,
  which_pars,
  scale = 0.02,
  nrows,
  ncols,
  precision = 200,
  stat_tol = 0.001,
  posdef_tol = 1e-08,
  df_tol = 1e-08
)
```

Arguments

gsmvar	an object of class 'gsmvar', typically created with fitGSMVAR or GSMVAR.
which_pars	the profile log-likelihood function of which parameters should be plotted? An integer vector specifying the positions of the parameters in the parameter vector. The parameter vector has the form...

For unconstrained models: Should be size $((M(pd^2 + d + d(d+1)/2 + 1) - 1)x1)$ and have form θ=(υ_1, ...,υ_M, α_1, ..., α_{M-1}), where:

- $\upsilon_m = (\phi_{m,0}, \phi_m, \sigma_m)$
- ϕ_m= $(vec(A_{m,1}), ..., vec(A_{m,p})$
- and $\sigma_m = vech(\Omega_m)$, m=1,...,M.

For constrained models: Should be size $((M(d+d(d+1)/2+1)+q-1)x1)$ and have form θ= $(\phi_{1,0}, ..., \phi_{M,0}, \psi, \sigma_1, ..., \sigma_M, \alpha_1, ..., \alpha_{M-1})$, where:

- ψ $(qx1)$ satisfies $(\phi_1, ..., \phi_M) = C\psi$. Here C is (Mpd^2xq) constraint matrix.

Above, $\phi_{m,0}$ is the intercept parameter, $A_{m,i}$ denotes the i:th coefficient matrix of the m:th mixture component, Ω_m denotes the error term covariance matrix of the m:th mixture component, and α_m is the mixing weight parameter.

The default is that profile log-likelihood functions for all parameters are plotted.

scale	a numeric scalar specifying the interval plotted for each estimate: the estimate plus-minus abs(scale*estimate).
nrows	how many rows should be in the plot-matrix? The default is max(ceiling(log2(length(which_pars)) - 1), 1).
ncols	how many columns should be in the plot-matrix? The default is ceiling(length(which_pars)/nrows). Note that nrows*ncols should not be smaller than the length of which_pars.
precision	at how many points should each profile log-likelihood be evaluated at?
stat_tol	numerical tolerance for stationarity of the AR parameters: if the "bold A" matrix of any regime has eigenvalues larger that 1 - stat_tol the model is classified as non-stationary. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.
posdef_tol	numerical tolerance for positive definiteness of the error term covariance matrices: if the error term covariance matrix of any regime has eigenvalues smaller than this, the model is classified as not satisfying positive definiteness assumption. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.
df_tol	the parameter vector is considered to be outside the parameter space if all degrees of freedom parameters are not larger than 2 + df_tol.

Details

When the number of parameters is large, it might be better to plot a smaller number of profile log-likelihood functions at a time using the argument which_pars.

The red vertical line points the estimate.

Value

Only plots to a graphical device and doesn't return anything.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lütkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.
- McElroy T. 2017. Computation of vector ARMA autocovariances. *Statistics and Probability Letters*, **124**, 92-96.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

get_soc, diagnostic_plot, fitGSMVAR, GSMVAR, GIRF, LR_test, Wald_test, cond_moment_plot

Examples

```
# Running all the below examples takes approximately 2 minutes.

# GMVAR(1,2) model
fit12 <- fitGSMVAR(gdpdef, p=1, M=2, ncalls=1, seeds=1)
fit12
profile_logliks(fit12)

# Structural GMVAR(1,2) model identified with sign
# constraints: model build based on inaccurate hand-given estimates.
W_122 <- matrix(c(1, 1, -1, 1), nrow=2)
params12s <- c(0.55, 0.11, 0.62, 0.17, 0.34, 0.05, -0.01, 0.72, 0.25,
 0.02, -0.14, 0.86, 0.54, 0.06, -0.16, 0.16, 3.62, 4.73, 0.67)
mod12s <- GSMVAR(gdpdef, p=1, M=2, params=params12s,
                 structural_pars=list(W=W_122))
profile_logliks(mod12s)

#' # G-StMVAR(2, 1, 1), d=2 model:
params22gs <- c(0.697, 0.154, 0.049, 0.374, 0.476, 0.318, -0.645, -0.302,
 -0.222, 0.193, 0.042, -0.013, 0.048, 0.554, 0.033, 0.184, 0.005, -0.186,
  0.683, 0.256, 0.031, 0.026, 0.204, 0.583, -0.002, 0.048, 0.182, 4.334)
mod22gs <- GSMVAR(gdpdef, p=2, M=c(1, 1), params=params22gs, model="G-StMVAR")
profile_logliks(mod22gs, which_pars=c(1, 3, 28))
```

quantile_residuals *Calculate multivariate quantile residuals of a GMVAR, StMVAR, or G-StMVAR model*

Description

`quantile_residuals` calculates multivariate quantile residuals (proposed by *Kalliovirta and Saikkonen 2010*) for a GMVAR, StMVAR, or G-StMVAR model.

Usage

```
quantile_residuals(gsmvar)
```

Arguments

`gsmvar`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.

Value

Returns $((n_obs - p)xd)$ matrix containing the multivariate quantile residuals, j:th column corresponds to the time series in the j:th column of the data. The multivariate quantile residuals are calculated so that the first column quantile residuals are the "unconditioned ones" and the rest condition on all the previous ones in numerical order. Read the cited article by *Kalliovirta and Saikkonen 2010* for details.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Kalliovirta L. and Saikkonen P. 2010. Reliable Residuals for Multivariate Nonlinear Time Series Models. *Unpublished Revision of HECER Discussion Paper No. 247*.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

`fitGSMVAR`, `GSMVAR`, `quantile_residual_tests`, `diagnostic_plot`, `predict.gsmvar`, `profile_logliks`

Examples

```
# GMVAR(1,2), d=2 model:
params12 <- c(0.55, 0.112, 0.344, 0.055, -0.009, 0.718, 0.319, 0.005, 0.03,
 0.619, 0.173, 0.255, 0.017, -0.136, 0.858, 1.185, -0.012, 0.136, 0.674)
mod12 <- GSMVAR(gdpdef, p=1, M=2, params=params12)
quantile_residuals(mod12)
```

```
# GMVAR(2,2), d=2 model with mean-parametrization:
params22 <- c(0.869, 0.549, 0.223, 0.059, -0.151, 0.395, 0.406, -0.005,
 0.083, 0.299, 0.215, 0.002, 0.03, 0.576, 1.168, 0.218, 0.02, -0.119,
 0.722, 0.093, 0.032, 0.044, 0.191, 1.101, -0.004, 0.105, 0.58)
mod22 <- GSMVAR(gdpdef, p=2, M=2, params=params22, parametrization="mean")
quantile_residuals(mod22)

# Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
params22s <- c(0.36, 0.121, 0.484, 0.072, 0.223, 0.059, -0.151, 0.395,
 0.406, -0.005, 0.083, 0.299, 0.218, 0.02, -0.119, 0.722, 0.093, 0.032,
 0.044, 0.191, 0.057, 0.172, -0.46, 0.016, 3.518, 5.154, 0.58)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
mod22s <- GSMVAR(gdpdef, p=2, M=2, params=params22s, structural_pars=list(W=W_22))
quantile_residuals(mod22s)
```

`random_ind2` *Create somewhat random parameter vector of a GMVAR, StMVAR, or G-StMVAR model that is always stationary*

Description

`random_ind2` generates random mean-parametrized parameter vector that is always stationary.

Usage

```
random_ind2(
  p,
  M,
  d,
  model = c("GMVAR", "StMVAR", "G-StMVAR"),
  same_means = NULL,
  weight_constraints = NULL,
  structural_pars = NULL,
  mu_scale,
  mu_scale2,
  omega_scale,
  ar_scale = 1,
  W_scale,
  lambda_scale
)
```

Arguments

`p`	a positive integer specifying the autoregressive order of the model.
`M`	**For GMVAR and StMVAR models:** a positive integer specifying the number of mixture components.

For G-StMVAR models: a size (2×1) integer vector specifying the number of *GMVAR type* components `M1` in the first element and *StMVAR type* components `M2` in the second element. The total number of mixture components is `M=M1+M2`.

`d` the number of time series in the system.

`model` is "GMVAR", "StMVAR", or "G-StMVAR" model considered? In the G-StMVAR model, the first `M1` components are GMVAR type and the rest `M2` components are StMVAR type.

`same_means` Restrict the mean parameters of some regimes to be the same? Provide a list of numeric vectors such that each numeric vector contains the regimes that should share the common mean parameters. For instance, if `M=3`, the argument `list(1, 2:3)` restricts the mean parameters of the second and third regime to be the same but the first regime has freely estimated (unconditional) mean. Ignore or set to `NULL` if mean parameters should not be restricted to be the same among any regimes. **This constraint is available only for mean parametrized models; that is, when** `parametrization="mean"`**.**

`weight_constraints`
a numeric vector of length $M - 1$ specifying fixed parameter values for the mixing weight parameters $\alpha_m,\ m = 1, ..., M - 1$. Each element should be strictly between zero and one, and the sum of all the elements should be strictly less than one.

`structural_pars`
If `NULL` a reduced form model is considered. Reduced models can be used directly as recursively identified structural models. For a structural model identified by conditional heteroskedasticity, should be a list containing at least the first one of the following elements:

- `W` - a (dxd) matrix with its entries imposing constraints on W: `NA` indicating that the element is unconstrained, a positive value indicating strict positive sign constraint, a negative value indicating strict negative sign constraint, and zero indicating that the element is constrained to zero.
- `C_lambda` - a $(d(M-1) \times r)$ constraint matrix that satisfies $(\lambda_2, ..., \lambda_M) = C_\lambda \gamma$ where γ is the new $(r \times 1)$ parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of `C_lambda` must be either **positive** or **zero**. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.
- `fixed_lambdas` - a length $d(M - 1)$ numeric vector $(\lambda_2, ..., \lambda_M)$ with elements strictly larger than zero specifying the fixed parameter values for the parameters λ_{mi} should be constrained to. This constraint is alternative `C_lambda`. Ignore (or set to `NULL`) if the eigenvalues λ_{mi} should not be constrained.

See Virolainen (forthcoming) for the conditions required to identify the shocks and for the B-matrix as well (it is W times a time-varying diagonal matrix with positive diagonal entries).

`mu_scale` a size $(dx1)$ vector defining **means** of the normal distributions from which each mean parameter μ_m is drawn from in random mutations. Default is `colMeans(data)`. Note that mean-parametrization is always used for optimization in `GAfit` - even

when `parametrization=="intercept"`. However, input (in `initpop`) and output (return value) parameter vectors can be intercept-parametrized.

`mu_scale2` a size $(dx1)$ strictly positive vector defining **standard deviations** of the normal distributions from which each mean parameter μ_m is drawn from in random mutations. Default is `2*sd(data[,i]), i=1,..,d`.

`omega_scale` a size $(dx1)$ strictly positive vector specifying the scale and variability of the random covariance matrices in random mutations. The covariance matrices are drawn from (scaled) Wishart distribution. Expected values of the random covariance matrices are `diag(omega_scale)`. Standard deviations of the diagonal elements are `sqrt(2/d)*omega_scale[i]` and for non-diagonal elements they are `sqrt(1/d*omega_scale[i]*omega_scale[j])`. Note that for `d>4` this scale may need to be chosen carefully. Default in `GAfit` is `var(stats::ar(data[,i], order.max=10)$resid, na.rm=TRUE), i=1,...,d`. This argument is ignored if structural model is considered.

`ar_scale` a positive real number between zero and one, adjusting how large AR parameter values are typically proposed in construction of the initial population: larger value implies larger coefficients (in absolute value). After construction of the initial population, a new scale is drawn from `(0, upper_ar_scale)` uniform distribution in each iteration. With large `p` or `d`, `ar_scale` is restricted from above, see the details section.

`W_scale` a size $(dx1)$ strictly positive vector partly specifying the scale and variability of the random covariance matrices in random mutations. The elements of the matrix W are drawn independently from such normal distributions that the expectation of the main **diagonal** elements of the first regime's error term covariance matrix $\Omega_1 = WW'$ is `W_scale`. The distribution of Ω_1 will be in some sense like a Wishart distribution but with the columns (elements) of W obeying the given constraints. The constraints are accounted for by setting the element to be always zero if it is subject to a zero constraint and for sign constraints the absolute value or negative the absolute value are taken, and then the variances of the elements of W are adjusted accordingly. This argument is ignored if reduced form model is considered.

`lambda_scale` a length $M-1$ vector specifying the **standard deviation** of the mean zero normal distribution from which the eigenvalue λ_{mi} parameters are drawn from in random mutations. As the eigenvalues should always be positive, the absolute value is taken. The elements of `lambda_scale` should be strictly positive real numbers with the $m-1$th element giving the degrees of freedom for the mth regime. The expected value of the main **diagonal** elements ij of the mth $(m > 1)$ error term covariance matrix will be `W_scale[i]*(d - n_i)^(-1)*sum(lambdas*ind_fun)` where the $(dx1)$ vector `lambdas` is drawn from the absolute value of the t-distribution, `n_i` is the number of zero constraints in the ith row of W and `ind_fun` is an indicator function that takes the value one iff the ijth element of W is not constrained to zero. Basically, larger lambdas (or smaller degrees of freedom) imply larger variance.

If the lambda parameters are **constrained** with the $(d(M-1)xr)$ constraint matrix C_lambda, then provide a length r vector specifying the standard deviation of the (absolute value of the) mean zero normal distribution each of the γ parameters are drawn from (the γ is a $(rx1)$ vector). The expected value of the main

diagonal elements of the covariance matrices then depend on the constraints.

This argument is ignored if $M == 1$ or a reduced form model is considered. Default is `rep(3, times=M-1)` if lambdas are not constrained and `rep(3, times=r)` if lambdas are constrained.

As with omega_scale and W_scale, this argument should be adjusted carefully if specified by hand. **NOTE** that if lambdas are constrained in some other way than restricting some of them to be identical, this parameter should be adjusted accordingly in order to the estimation succeed!

Details

The coefficient matrices are generated using the algorithm proposed by Ansley and Kohn (1986) which forces stationarity. It's not clear in detail how `ar_scale` exactly affects the coefficient matrices but larger `ar_scale` seems to result in larger AR coefficients. Read the cited article by Ansley and Kohn (1986) and the source code for more information.

The covariance matrices are generated from (scaled) Wishart distribution.

Models with AR parameters constrained are not supported!

Value

Returns random mean-parametrized parameter vector that has the same form as the argument `params` in the other functions, for instance, in the function `loglikelihood`.

References

- Ansley C.F., Kohn R. 1986. A note on reparameterizing a vector autoregressive moving average model to enforce stationarity. *Journal of statistical computation and simulation*, **24**:2, 99-106.
- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

@keywords internal

`Rao_test` *Perform Rao's score test for a GSMVAR model*

Description

`Rao_test` performs Rao's score test for a GSMVAR model

Usage

```
Rao_test(gsmvar)
```

Arguments

gsmvar	an object of class 'gsmvar' generated by fitGSMVAR or GSMVAR, containing the model specified by the null hypothesis (i.e., **the constrained model**).

Details

Tests the constraints imposed in the model given in the argument GSMVAR. This implementation uses the outer product of gradients approximation in the test statistic.

Value

A list with class "hypotest" containing the test results and arguments used to calculate the test.

References

- Buse A. (1982). The Likelihood Ratio, Wald, and Lagrange Multiplier Tests: An Expository Note. *The American Statistician*, 36(3a), 153-157.

See Also

LR_test, Wald_test, fitGSMVAR, GSMVAR, diagnostic_plot, profile_logliks

Examples

```
## These are long running examples that use parallel computing!
## The below examples take around 30 seconds to run.

# Structural GMVAR(2, 2), d=2 model with recursive identification
# with the AR matrices  restricted to be the identical across the regimes:
W22 <- matrix(c(1, NA, 0, 1), nrow=2, byrow=FALSE)
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
fit22sc <- fitGSMVAR(gdpdef, p=2, M=2, constraints=C_mat,
                     structural_pars=list(W=W22), ncalls=1, seeds=1)

# Test the null:
Rao_test(fit22sc)
```

---

redecompose_Omegas *In the decomposition of the covariance matrices (Muirhead, 1982, Theorem A9.9), change the order of the covariance matrices.*

Description

redecompose_Omegas exchanges the order of the covariance matrices in the decomposition of Muirhead (1982, Theorem A9.9) and returns the new decomposition.

Usage

```
redecompose_Omegas(M, d, W, lambdas, perm = 1:sum(M))
```

Arguments

M	**For GMVAR and StMVAR models:** a positive integer specifying the number of mixture components. **For G-StMVAR models:** a size (2×1) integer vector specifying the number of *GMVAR type* components M1 in the first element and *StMVAR type* components M2 in the second element. The total number of mixture components is M=M1+M2.
d	the number of time series in the system.
W	a length d^2 vector containing the vectorized W matrix.
lambdas	a length d*(M-1) vector of the form $\lambda_2, ..., \lambda_M$ where $\lambda_m = (\lambda_{m1}, ..., \lambda_{md})$
perm	a vector of length M giving the new order of the covarince matrices (relative to the current order)

Details

We consider the following decomposition of positive definite covariannce matrices: $\Omega_1 = WW'$, $\Omega_m = W\Lambda_m W'$, $m = 2, .., M$ where $\Lambda_m = diag(\lambda_{m1}, ..., \lambda_{md})$ contains the strictly postive eigenvalues of $\Omega_m \Omega_1^{-1}$ and the column of the invertible W are the corresponding eigenvectors. Note that this decomposition does not necessarily exists for $M > 2$.

See Muirhead (1982), Theorem A9.9 for more details on the decomposition and the source code for more details on the reparametrization.

Value

Returns a $d^2 + (M-1) * dx1$ vector of the form c(vec(new_W), new_lambdas) where the lambdas parameters are in the regimewise order (first regime 2, then 3, etc) and the "new W" and "new lambdas" are constitute the new decomposition with the order of the covariance matrices given by the argument perm. Notice that if the first element of perm is one, the W matrix will be the same and the lambdas are just re-ordered.

Note that unparametrized zero elements ARE present in the returned W!

Warning

No argument checks! Does not work with dimension $d = 1$ or with only one mixture component $M = 1$.

References

- Muirhead R.J. 1982. Aspects of Multivariate Statistical Theory, *Wiley*.

Examples

```
d <- 2
M <- 2
Omega1 <- matrix(c(2, 0.5, 0.5, 2), nrow=d)
Omega2 <- matrix(c(1, -0.2, -0.2, 1), nrow=d)

# Decomposition with Omega1 as the first covariance matrix:
decomp1 <- diag_Omegas(Omega1, Omega2)
W <- matrix(decomp1[1:d^2], nrow=d, ncol=d)
lambdas <- decomp1[(d^2 + 1):length(decomp1)]
tcrossprod(W) # = Omega1
W%*%tcrossprod(diag(lambdas), W) # = Omega2

# Reorder the covariance matrices in the decomposition so that now
# the first covariance matrix is Omega2:
decomp2 <- redecompose_Omegas(M=M, d=d, W=as.vector(W), lambdas=lambdas,
                              perm=2:1)
new_W <- matrix(decomp2[1:d^2], nrow=d, ncol=d)
new_lambdas <- decomp2[(d^2 + 1):length(decomp2)]
tcrossprod(new_W) # = Omega2
new_W%*%tcrossprod(diag(new_lambdas), new_W) # = Omega1
```

`reorder_W_columns` *Reorder columns of the W-matrix and lambda parameters of a structural GMVAR, StMVAR, or G-StMVAR model.*

Description

`reorder_W_columns` reorder columns of the W-matrix and lambda parameters of a structural GMVAR, StMVAR, or G-StMVAR model.

Usage

```
reorder_W_columns(gsmvar, perm)
```

Arguments

`gsmvar`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.
`perm`	an integer vector of length d specifying the new order of the columns of W. Also lambda parameters of each regime will be reordered accordingly.

Details

The order of the columns of W can be changed without changing the implied reduced form model as long as the order of lambda parameters is also changed accordingly. Note that the constraints imposed on W (or the B-matrix) will also be modified accordingly.

This function does not support models with constraints imposed on the lambda parameters!

Also all signs in any column of W can be swapped (without changing the implied reduced form model) with the function `swap_W_signs` but this obviously also swaps the sign constraints in the corresponding columns of W.

Value

Returns an object of class `'gsmvar'` defining a structural GMVAR, StMVAR, or G-StMVAR model with the modified structural parameters and constraints.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. (forthcoming). A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*.
- Virolainen S. 2022. Gaussian and Student's t mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks in the Euro area. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

See Also

`fitGSMVAR`, `GSMVAR`, `GIRF`, `gsmvar_to_sgsmvar`, `stmvar_to_gstmvar`, `swap_W_signs`

Examples

```
# Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
params22s <- c(0.36, 0.121, 0.484, 0.072, 0.223, 0.059, -0.151, 0.395,
  0.406, -0.005, 0.083, 0.299, 0.218, 0.02, -0.119, 0.722, 0.093, 0.032,
  0.044, 0.191, 0.057, 0.172, -0.46, 0.016, 3.518, 5.154, 0.58)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
mod22s <- GSMVAR(p=2, M=2, d=2, params=params22s, structural_pars=list(W=W_22))
mod22s

# The same reduced form model, reordered W and lambda in the structual model:
mod22s_2 <- reorder_W_columns(mod22s, perm=2:1)
mod22s_2

# Structural StMVAR(2, 2), d=2 model identified with sign-constraints:
mod22ts <- GSMVAR(p=2, M=2, d=2, params=c(params22s, 10, 20), model="StMVAR",
                  structural_pars=list(W=W_22))
mod22ts

# The same reduced form model, reordered W and lambda in the structual model:
mod22ts_2 <- reorder_W_columns(mod22ts, perm=2:1)
mod22ts_2
```

`simulate.gsmvar` *Simulate method for class 'gsmvar' objects*

Description

`simulate.gsmvar` is a simulate method for class 'gsmvar' objects. It allows to simulate observations from a GMVAR, StMVAR, or G-StMVAR process.

Usage

```
## S3 method for class 'gsmvar'
simulate(
  object,
  nsim = 1,
  seed = NULL,
  ...,
  init_values = NULL,
  init_regimes = 1:sum(object$model$M),
  ntimes = 1,
  drop = TRUE
)
```

Arguments

`object`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.
`nsim`	number of observations to be simulated.
`seed`	set seed for the random number generator?
`...`	currently not in use.
`init_values`	a size $(p \times d)$ matrix specifying the initial values, where d is the number of time series in the system. The **last** row will be used as initial values for the first lag, the second last row for second lag etc. If not specified, initial values will be drawn according to mixture distribution specifed by the argument `init_regimes`.
`init_regimes`	a numeric vector of length at most M and elements in $1, ..., M$ specifying the regimes from which the initial values should be generated from. The initial values will be generated from a mixture distribution with the mixture components being the stationary distributions of the specific regimes and the (proportional) mixing weights given by the mixing weight parameters of those regimes. Note that if `init_regimes=1:M`, the initial values are generated from the stationary distribution of the process and if `init_regimes=m`, the initial value are generated from the stationary distribution of the mth regime. Ignored if the argument `init_values` is specified.
`ntimes`	how many sets of simulations should be performed?
`drop`	if `TRUE` (default) then the components of the returned list are coerced to lower dimension if `ntimes==1`, i.e., `$sample` and `$mixing_weights` will be matrices, and `$component` will be vector.

Details

The argument `ntimes` is intended for forecasting: a GMVAR, StMVAR, or G-StMVAR process can be forecasted by simulating its possible future values. One can easily perform a large number simulations and calculate the sample quantiles from the simulated values to obtain prediction intervals (see the forecasting example).

Value

If `drop==TRUE` and `ntimes==1` (default): `$sample`, `$component`, and `$mixing_weights` are matrices. Otherwise, returns a list with...

`$sample` a size (`nsim`$\times d\times$`ntimes`) array containing the samples: the dimension `[t, , ]` is the time index, the dimension `[, d, ]` indicates the marginal time series, and the dimension `[, , i]` indicates the i:th set of simulations.

`$component` a size (`nsim`$\times$`ntimes`) matrix containing the information from which mixture component each value was generated from.

`$mixing_weights` a size (`nsim`$\times M\times$`ntimes`) array containing the mixing weights corresponding to the sample: the dimension `[t, , ]` is the time index, the dimension `[, m, ]` indicates the regime, and the dimension `[, , i]` indicates the i:th set of simulations.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lütkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.
- McElroy T. 2017. Computation of vector ARMA autocovariances. *Statistics and Probability Letters*, **124**, 92-96.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

`fitGSMVAR`, `GSMVAR`, `diagnostic_plot`, `predict.gsmvar`, `profile_logliks`, `quantile_residual_tests`, `GIRF`, `GFEVD`

Examples

```
# GMVAR(1,2), d=2 process, initial values from the stationary
# distribution
params12 <- c(0.55, 0.112, 0.344, 0.055, -0.009, 0.718, 0.319, 0.005,
 0.03, 0.619, 0.173, 0.255, 0.017, -0.136, 0.858, 1.185, -0.012, 0.136,
 0.674)
mod12 <- GSMVAR(p=1, M=2, d=2, params=params12)
set.seed(1)
sim12 <- simulate(mod12, nsim=500)
plot.ts(sim12$sample)
```

```
ts.plot(sim12$mixing_weights, col=c("blue", "red"), lty=2)
plot(sim12$component, type="l")

# StMVAR(2, 2), d=2 model
params22t <- c(0.554, 0.033, 0.184, 0.005, -0.186, 0.683, 0.256, 0.031,
 0.026, 0.204, 0.583, -0.002, 0.048, 0.697, 0.154, 0.049, 0.374, 0.476,
  0.318, -0.645, -0.302, -0.222, 0.193, 0.042, -0.013, 0.048, 0.818,
   4.334, 20)
mod22t <- GSMVAR(gdpdef, p=2, M=2, params=params22t, model="StMVAR")
sim22t <- simulate(mod22t, nsim=100)
plot.ts(sim22t$mixing_weights)

## FORECASTING EXAMPLE ##
# Forecast 5-steps-ahead, 500 sets of simulations with initial
# values from the data:
# GMVAR(2,2), d=2 model
params22 <- c(0.36, 0.121, 0.223, 0.059, -0.151, 0.395, 0.406, -0.005,
 0.083, 0.299, 0.215, 0.002, 0.03, 0.484, 0.072, 0.218, 0.02, -0.119,
  0.722, 0.093, 0.032, 0.044, 0.191, 1.101, -0.004, 0.105, 0.58)
mod22 <- GSMVAR(gdpdef, p=2, M=2, params=params22)
sim22 <- simulate(mod22, nsim=5, ntimes=500)

# Point forecast + 95% prediction intervals:
apply(sim22$sample, MARGIN=1:2, FUN=quantile, probs=c(0.025, 0.5, 0.972))

# Similar forecast for the mixing weights:
apply(sim22$mixing_weights, MARGIN=1:2, FUN=quantile,
      probs=c(0.025, 0.5, 0.972))
```

`stmvar_to_gstmvar` *Estimate a G-StMVAR model based on a StMVAR model that has large degrees of freedom parameters*

Description

`stmvar_to_gstmvar` estimates a G-StMVAR model based on a StMVAR model that has large degrees of freedom parameters.

Usage

```
stmvar_to_gstmvar(
  gsmvar,
  estimate,
  calc_std_errors = estimate,
  maxdf = 100,
  maxit = 100
)
```

Arguments

gsmvar	an object of class 'gsmvar', typically created with fitGSMVAR or GSMVAR.
estimate	set TRUE if the new model should be estimated with a variable metric algorithm using the StMAR model parameter value as the initial value. By default TRUE iff the model contains data.
calc_std_errors	set TRUE if the approximate standard errors should be calculated.
maxdf	regimes with degrees of freedom parameter value larger than this will be turned into GMVAR type.
maxit	the maximum number of iterations for the variable metric algorithm. Ignored if estimate==FALSE.

Details

If a StMVAR model contains large estimates for the degrees of freedom parameters, one should consider switching to the corresponding G-StMAR model that lets the corresponding regimes to be GMVAR type. stmvar_to_gstmvar does this switch conveniently. Also G-StMVAR models are supported if some of the StMVAR type regimes have large degrees of freedom paraters.

Note that if the model imposes constraints on the autoregressive parameters, or if a structural model imposes constraints on the lambda parameters, and the ordering the regimes changes, the constraints are removed from the model. This is because of the form of the constraints that does not generally allow to switch the ordering of the regimes. If you wish to keep the constraints, you may construct the resulting G-StMVAR model parameter vector by hand, redefine your constraints accordingly, build the model with the function GSMVAR, and then estimate it with the function iterate_more. Alternatively, you can always directly estimate the constrained G-StMVAR model with the function fitGSMVAR.

Value

Returns an object of class 'gsmvar' defining a G-StMVAR model based on the provided StMVAR (or G-StMVAR) model with the regimes that had large degrees of freedom parameters changed to GMVAR type.

References

- Muirhead R.J. 1982. Aspects of Multivariate Statistical Theory, *Wiley*.
- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

fitGSMVAR, GSMVAR, GIRF, reorder_W_columns, swap_W_signs, gsmvar_to_sgsmvar

Examples

```
# StMVAR(1, 2), d=2 model:
params12t <- c(0.5453, 0.1157, 0.331, 0.0537, -0.0422, 0.7089, 0.4181, 0.0018,
  0.0413, 1.6004, 0.4843, 0.1256, -0.0311, -0.6139, 0.7221, 1.2123, -0.0357,
  0.1381, 0.8337, 7.5564, 90000)
mod12t <- GSMVAR(gdpdef, p=1, M=2, params=params12t, model="StMVAR")
mod12t

# Switch to the G-StMVAR model:
mod12gs <- stmvar_to_gstmvar(mod12t)
mod12gs
```

---

swap_parametrization *Swap the parametrization of a GMVAR, StMVAR, or G-StMVAR model*

Description

swap_parametrization swaps the parametrization of a GMVAR, StMVAR or G-StMVAR, model to "mean" if the current parametrization is "intercept", and vice versa.

Usage

```
swap_parametrization(gsmvar)
```

Arguments

gsmvar	an object of class 'gsmvar', typically created with fitGSMVAR or GSMVAR.

Details

swap_parametrization is a convenient tool if you have estimated the model in "intercept"-parametrization, but wish to work with "mean"-parametrization in the future, or vice versa. In gmvarkit, the approximate standard errors are only available for parametrized parameters.

Value

Returns an object of class 'gsmvar' defining the specified reduced form or structural GMVAR, StMVAR, or G-StMVAR model. Can be used to work with other functions provided in gmvarkit.

Note that the first autocovariance/correlation matrix in $uncond_moments is for the lag zero, the second one for the lag one, etc.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Kalliovirta L. and Saikkonen P. 2010. Reliable Residuals for Multivariate Nonlinear Time Series Models. *Unpublished Revision of HECER Discussion Paper No. 247*.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

fitGSMVAR, GSMVAR, iterate_more, update_numtols

Examples

```
# GMVAR(2, 2), d=2 model with mean-parametrization:
params22 <- c(0.869, 0.549, 0.223, 0.059, -0.151, 0.395, 0.406,
 -0.005, 0.083, 0.299, 0.215, 0.002, 0.03, 0.576, 1.168, 0.218,
 0.02, -0.119, 0.722, 0.093, 0.032, 0.044, 0.191, 1.101, -0.004,
 0.105, 0.58)
mod22 <- GSMVAR(gdpdef, p=2, M=2, params=params22, parametrization="mean")
mod22 # mean parametrization

mod22_2 <- swap_parametrization(mod22)
mod22_2 # intercept parametrization

# G-StMVAR(2, 1, 1), d=2 model with mean-parametrization:
mod22gs <- GSMVAR(gdpdef, p=2, M=c(1, 1), params=c(params22, 10), model="G-StMVAR",
                  parametrization="mean")
mod22gs # mean parametrization

mod22gs_2 <- swap_parametrization(mod22gs)
mod22gs_2 # intercept parametrization

# Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
params22s <- c(0.36, 0.121, 0.484, 0.072, 0.223, 0.059, -0.151, 0.395,
  0.406, -0.005, 0.083, 0.299, 0.218, 0.02, -0.119, 0.722, 0.093, 0.032,
  0.044, 0.191, 0.057, 0.172, -0.46, 0.016, 3.518, 5.154, 0.58)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
mod22s <- GSMVAR(p=2, M=2, d=2, params=params22s, structural_pars=list(W=W_22))
mod22s # intercept parametrization

mod22s_2 <- swap_parametrization(mod22s)
mod22s_2 # mean parametrization
```

`swap_W_signs` *Swap all signs in pointed columns a the W matrix of a structural GMVAR, StMVAR, or G-StMVAR model.*

Description

`swap_W_signs` swaps all signs in pointed columns a the W matrix of a structural GMVAR, StMVAR, or G-StMVAR model. Consequently, signs in the columns of the B-matrix are also swapped accordingly.

Usage

```
swap_W_signs(gsmvar, which_to_swap)
```

Arguments

`gsmvar`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.
`which_to_swap`	a numeric vector of length at most d and elemnts in $1, .., d$ specifying the columns of W whose sign should be swapped.

Details

All signs in any column of W can be swapped without changing the implied reduced form model. Consequently, also the signs in the columns of the B-matrix are swapped. Note that the sign constraints imposed on W (or the B-matrix) are also swapped in the corresponding columns accordingly.

Also the order of the columns of W can be changed (without changing the implied reduced form model) as long as the order of lambda parameters is also changed accordingly. This can be done with the function `reorder_W_columns`.

Value

Returns an object of class `'gsmvar'` defining a structural GMVAR, StMVAR, or G-StMVAR model with the modified structural parameters and constraints.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. (forthcoming). A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*.
- Virolainen S. 2022. Gaussian and Student's t mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks in the Euro area. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

See Also

fitGSMVAR, GSMVAR, GIRF, reorder_W_columns, gsmvar_to_sgsmvar, stmvar_to_gstmvar

Examples

```
# Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
params22s <- c(0.36, 0.121, 0.484, 0.072, 0.223, 0.059, -0.151, 0.395,
  0.406, -0.005, 0.083, 0.299, 0.218, 0.02, -0.119, 0.722, 0.093, 0.032,
  0.044, 0.191, 0.057, 0.172, -0.46, 0.016, 3.518, 5.154, 0.58)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
mod22s <- GSMVAR(p=2, M=2, d=2, params=params22s, structural_pars=list(W=W_22))
mod22s

# The same reduced form model, with signs in the second column of W swapped:
swap_W_signs(mod22s, which_to_swap=2)

# The same reduced form model, with signs in both column of W swapped:
swap_W_signs(mod22s, which_to_swap=1:2)

#' # Structural G-StMVAR(2, 1, 1), d=2 model identified with sign-constraints:
mod22gss <- GSMVAR(p=2, M=c(1, 1), d=2, params=c(params22s, 10), model="G-StMVAR",
                   structural_pars=list(W=W_22))
mod22gss

# The same reduced form model, with signs in the first column of W swapped:
swap_W_signs(mod22gss, which_to_swap=1)
```

---

uncond_moments *Calculate the unconditional mean, variance, the first p autocovariances, and the first p autocorrelations of a GMVAR, StMVAR, or G-StMVAR process*

Description

uncond_moments calculates the unconditional mean, variance, the first p autocovariances, and the first p autocorrelations of the given GMVAR, StMVAR, or G-StMVAR process.

Usage

```
uncond_moments(gsmvar)
```

Arguments

gsmvar	an object of class 'gsmvar', typically created with fitGSMVAR or GSMVAR.

Details

The unconditional moments are based on the stationary distribution of the process.

Value

Returns a list with three components:

`$uncond_mean` a length d vector containing the unconditional mean of the process.

`$autocovs` an $(dxdxp + 1)$ array containing the lag 0,1,...,p autocovariances of the process. The subset `[, , j]` contains the lag `j-1` autocovariance matrix (lag zero for the variance).

`$autocors` the autocovariance matrices scaled to autocorrelation matrices.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lütkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.
- McElroy T. 2017. Computation of vector ARMA autocovariances. *Statistics and Probability Letters*, **124**, 92-96.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Virolainen S. in press. A Gaussian and Student's mixture vector autoregressive model with an application to monetary policy shocks. Econometrics and Statistics.

See Also

Other moment functions: `cond_moments()`, `get_regime_autocovs()`, `get_regime_means()`

Examples

```
# GMVAR(1,2), d=2 model:
params12 <- c(0.55, 0.112, 0.344, 0.055, -0.009, 0.718, 0.319, 0.005,
 0.03, 0.619, 0.173, 0.255, 0.017, -0.136, 0.858, 1.185, -0.012,
 0.136, 0.674)
mod12 <- GSMVAR(gdpdef, p=1, M=2, params=params12)
uncond_moments(mod12)

# Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
params22s <- c(0.36, 0.121, 0.484, 0.072, 0.223, 0.059, -0.151, 0.395,
 0.406, -0.005, 0.083, 0.299, 0.218, 0.02, -0.119, 0.722, 0.093, 0.032,
 0.044, 0.191, 0.057, 0.172, -0.46, 0.016, 3.518, 5.154, 0.58)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
mod22s <- GSMVAR(gdpdef, p=2, M=2, params=params22s, structural_pars=list(W=W_22))
mod22s
uncond_moments(mod22s)
```

`update_numtols` *Update the stationarity and positive definiteness numerical tolerances of an existing class 'gsmvar' model.*

Description

`update_numtols` updates the stationarity and positive definiteness numerical tolerances of an existing class 'gsmvar' model.

Usage

```
update_numtols(gsmvar, stat_tol = 0.001, posdef_tol = 1e-08, df_tol = 1e-08)
```

Arguments

`gsmvar`	an object of class `'gsmvar'`, typically created with `fitGSMVAR` or `GSMVAR`.
`stat_tol`	numerical tolerance for stationarity of the AR parameters: if the "bold A" matrix of any regime has eigenvalues larger that `1 - stat_tol` the model is classified as non-stationary. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.
`posdef_tol`	numerical tolerance for positive definiteness of the error term covariance matrices: if the error term covariance matrix of any regime has eigenvalues smaller than this, the model is classified as not satisfying positive definiteness assumption. Note that if the tolerance is too small, numerical evaluation of the log-likelihood might fail and cause error.
`df_tol`	the parameter vector is considered to be outside the parameter space if all degrees of freedom parameters are not larger than `2 + df_tol`.

Details

All signs in any column of W can be swapped without changing the implied reduced form model. Consequently, also the signs in the columns of the B-matrix are swapped. Note that the sign constraints imposed on W (or the B-matrix) are also swapped in the corresponding columns accordingly.

Also the order of the columns of W can be changed (without changing the implied reduced form model) as long as the order of lambda parameters is also changed accordingly. This can be done with the function `reorder_W_columns`.

Value

Returns an object of class `'gsmvar'` defining a structural GSMVAR model with the modified structural parameters and constraints.

References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. (forthcoming). A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*.
- Virolainen S. 2022. Gaussian and Student's t mixture vector autoregressive model with application to the asymmetric effects of monetary policy shocks in the Euro area. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

See Also

fitGSMVAR, GSMVAR, GIRF, reorder_W_columns, gsmvar_to_sgsmvar, stmvar_to_gstmvar

Examples

```
# Structural GMVAR(2, 2), d=2 model identified with sign-constraints:
params22s <- c(0.36, 0.121, 0.484, 0.072, 0.223, 0.059, -0.151, 0.395,
  0.406, -0.005, 0.083, 0.299, 0.218, 0.02, -0.119, 0.722, 0.093, 0.032,
  0.044, 0.191, 0.057, 0.172, -0.46, 0.016, 3.518, 5.154, 0.58)
W_22 <- matrix(c(1, 1, -1, 1), nrow=2, byrow=FALSE)
mod22s <- GSMVAR(p=2, M=2, d=2, params=params22s, structural_pars=list(W=W_22))
mod22s

# Update numerical tolerances:
mod22s <- update_numtols(mod22s, stat_tol=1e-4, posdef_tol=1e-9, df_tol=1e-10)
mod22s # The same model
```

---

usamon *U.S. macroeconomic data used in Virolainen (2025)*

Description

A quarterly U.S. data covering the period from 1954Q3 to 2021Q4 (270 observations) and consisting four variables: the log-difference of real GDP, the log-difference of GDP implicit price deflator, the log-difference of producer price index (all commodities), and an interest rate variable. The interest rate variable is the effective federal funds rate from 1954Q3 to 2008Q2 and after that the Wu and Xia (2016) shadow rate, which is not constrained by the zero lower bound and also quantifies unconventional monetary policy measures. The log-differences of the GDP, GDP deflator, and producer price index are multiplied by hundred. This data is used in Virolainen (2025).

Usage

```
usamon
```

Format

A numeric matrix of class 'ts' with 270 rows and 4 columns with one time series in each column:

First column (GDP): The log-difference of real GDP, https://fred.stlouisfed.org/series/GDPC1.

Second column (GDPDEF): The log-difference of GDP implicit price deflator, https://fred.stlouisfed.org/series/GDPDEF.

Third column (PPI): The log-difference of producer price index (all commodities), https://fred.stlouisfed.org/series/PPIACO.

Fourth column (RATE): The Federal funds rate from 1954Q3 to 2008Q2 and after that the Wu and Xia (2016) shadow rate, https://fred.stlouisfed.org/series/FEDFUNDS, https://www.atlantafed.org/cqer/research/wu-xia-shadow-federal-funds-rate.

Source

The Federal Reserve Bank of St. Louis database and the Federal Reserve Bank of Atlanta's website

References

- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*, **43**:1, 44-54.
- Wu J. and Xia F. 2016. Measuring the macroeconomic impact of monetary policy at the zero lower bound. *Journal of Money, Credit and Banking*, 48(2-3): 253-291.

---

usamone *U.S. macroeconomic data*

Description

A quarterly U.S. data covering the period from 1954Q3 to 2021Q4 (270 observations) and consisting four variables: cyclical component of the log of real GDP, the log-difference of GDP implicit price deflator, the log-difference of producer price index (all commodities), and an interest rate variable. The interest rate variable is the effective federal funds rate from 1954Q3 to 2008Q2 and after that the Wu and Xia (2016) shadow rate, which is not constrained by the zero lower bound and also quantifies unconventional monetary policy measures. The log-differences of the GDP deflator and producer price index are multiplied by hundred.

The cyclical component of the log of real GDP was obtained by applying a one-sided Hodrick-Prescott (HP) filter with the standard smoothing parameter lambda=1600. The one-sided filter was obtained from the two-sided HP filter by applying the filter up to horizon t, taking the last observation, and repeating this procedure for the full sample t=1,...,T. In order to allow the series to start from any phase of the cycle, we applied the one-sided filter to the full available sample from 1947Q1 to 2021Q1 before extracting our sample period from it. We computed the two-sided HP filters with the R package lpirfs (Adämmer, 2021)

Usage

```
usamone
```

Format

A numeric matrix of class 'ts' with 270 rows and 4 columns with one time series in each column:

First column (GDP): The cyclical component of the log of real GDP, https://fred.stlouisfed.org/series/GDPC1.

Second column (GDPDEF): The log-difference of GDP implicit price deflator, https://fred.stlouisfed.org/series/GDPDEF.

Third column (PPI): The log-difference of producer price index (all commodities), https://fred.stlouisfed.org/series/PPIACO.

Fourth column (RATE): The Federal funds rate from 1954Q3 to 2008Q2 and after that the Wu and Xia (2016) shadow rate, https://fred.stlouisfed.org/series/FEDFUNDS, https://www.atlantafed.org/cqer/research/wu-xia-shadow-federal-funds-rate.

Source

The Federal Reserve Bank of St. Louis database and the Federal Reserve Bank of Atlanta's website

References

- Adämmer P. 2021. lprfs: Local Projections Impulse Response Functions. R package version: 0.2.0, https://CRAN.R-project.org/package=lpirfs.
- Virolainen S. 2025. A statistically identified structural vector autoregression with endogenously switching volatility regime. *Journal of Business & Economic Statistics*. **43**:1, 44-54.
- Wu J. and Xia F. 2016. Measuring the macroeconomic impact of monetary policy at the zero lower bound. *Journal of Money, Credit and Banking*, 48(2-3): 253-291.

Wald_test *Perform Wald test for a GMVAR, StMVAR, or G-StMVAR model*

Description

Wald_test performs a Wald test for a GMVAR, StMVAR, or G-StMVAR model

Usage

```
Wald_test(gsmvar, A, c, custom_h = NULL)
```

Arguments

gsmvar	an object of class 'gsmvar', typically created with fitGSMVAR or GSMVAR.
A	a size $(k \times n_params)$ matrix with full row rank specifying part of the null hypothesis where n_params is the number of parameters in the (unconstrained) model. See details for more information.
c	a length k vector specifying part of the null hypothesis. See details for more information.

`custom_h` a numeric vector with the same length as `x` specifying the difference `h` for each dimension separately. If `NULL` (default), then the difference `1e-6` used for all but overly large degrees of freedom parameters. For them, the difference is adjusted to avoid numerical problems.

Details

Denoting the true parameter value by θ_0, we test the null hypothesis $A\theta_0 = c$. Under the null, the test statistic is asymptotically χ^2-distributed with k (`=nrow(A)`) degrees of freedom. The parameter θ_0 is assumed to have the same form as in the model supplied in the argument `gsmvar` and it is presented in the documentation of the argument `params` in the function `GSMVAR` (see `?GSMVAR`).

Finally, note that this function does **not** check whether the specified constraints are feasible (e.g. whether the implied constrained model would be stationary or have positive definite error term covariance matrices).

Value

A list with class "hypotest" containing the test results and arguments used to calculate the test.

References

- Buse A. (1982). The Likelihood Ratio, Wald, and Lagrange Multiplier Tests: An Expository Note. *The American Statistician*, 36(3a), 153-157.

See Also

`LR_test`, `Rao_test`, `fitGSMVAR`, `GSMVAR`, `diagnostic_plot`, `profile_logliks`, `quantile_residual_tests`, `cond_moment_plot`

Examples

```
# Structural GMVAR(2, 2), d=2 model with recursive identification
W22 <- matrix(c(1, NA, 0, 1), nrow=2, byrow=FALSE)
fit22s <- fitGSMVAR(gdpdef, p=2, M=2, structural_pars=list(W=W22),
                    ncalls=1, seeds=2)
fit22s

# Test whether the lambda parameters (of the second regime) are identical
# (due to the zero constraint, the model is identified under the null):
# fit22s has parameter vector of length 26 with the lambda parameters
# in elements 24 and 25.
A <- matrix(c(rep(0, times=23), 1, -1, 0), nrow=1, ncol=26)
c <- 0
Wald_test(fit22s, A=A, c=c)

# Test whether the off-diagonal elements of the first regime's first
# AR coefficient matrix (A_11) are both zero:
# fit22s has parameter vector of length 26 and the off-diagonal elements
# of the 1st regime's 1st AR coefficient matrix are in the elements 6 and 7.
A <- rbind(c(rep(0, times=5), 1, rep(0, times=20)),
           c(rep(0, times=6), 1, rep(0, times=19)))
```

```
c <- c(0, 0)
Wald_test(fit22s, A=A, c=c)
```

Index

∗ **datasets**
euromone, 24
gdpdef, 37
usamon, 107
usamone, 108
∗ **moment functions**
cond_moments, 13
get_regime_autocovs, 40
get_regime_means, 41
uncond_moments, 104

acf, *20*
add_data, 4, *56*
alt_gsmvar, 5

calc_gradient, 7, *74*
calc_hessian *(*calc_gradient*)*, 7
check_parameters, 8
cond_moment_plot, 17, *20*, *29*, *75*, *80*, *87*, *110*
cond_moments, 13, *41*
cond_moments(), *40, 41*, *105*

density, *20*
diag_Omegas, 20
diagnostic_plot, *18*, 19, *75, 76*, *80*, *87, 88*, *93*, *98*, *110*

estimate_sgsmvar, 21
euromone, 24

fitGSMVAR, *4*, *6*, *18*, *20*, *23*, 25, *44*, *49*, *56*, *58*, *66*, *69*, *74–76*, *80*, *85*, *87, 88*, *93*, *96*, *98*, *100*, *102*, *104*, *107*, *110*

GAfit, 30
gdpdef, 37
get_boldA_eigens, 38
get_foc *(*calc_gradient*)*, 7
get_gradient, *29*
get_gradient *(*calc_gradient*)*, 7
get_hessian *(*calc_gradient*)*, 7
get_omega_eigens, 39
get_regime_autocovs, 40, *41*
get_regime_autocovs(), *17*, *41*, *105*
get_regime_means, 41
get_regime_means(), *17*, *40*, *105*
get_soc, *87*
get_soc *(*calc_gradient*)*, 7
GFEVD, *29*, 42, *49*, *69*, *82*, *98*
GIRF, *29*, *44*, 45, *56*, *58*, *69*, *80*, *82*, *87*, *96*, *98*, *100*, *104*, *107*
gmvar_to_gsmvar, 50
gmvarkit *(*gmvarkit-package*)*, 3
gmvarkit-package, 3
GSMVAR, *4*, *6*, *18*, *20*, *23*, *29*, *44*, *49*, 51, *58*, *66*, *69*, *74–76*, *80*, *85*, *87, 88*, *93*, *96*, *98*, *100*, *102*, *104*, *107*, *110*
gsmvar_to_sgsmvar, *23*, *29*, *44*, *49*, *56*, 56, *69*, *96*, *100*, *104*, *107*

in_paramspace, 58
in_paramspace_int, 62
iterate_more, *4*, *6*, *23*, *29*, 65, *102*

linear_IRF, *44*, *49*, 67
logLik.gsmvar *(*GSMVAR*)*, 51
loglikelihood, 70
LR_test, *18*, *20*, *29*, *49*, 74, *80*, *87*, *93*, *110*

optim, *23*, *66*

Pearson_residuals, 76
plot.gfevd *(*GFEVD*)*, 42
plot.girf *(*GIRF*)*, 46
plot.gsmvar *(*GSMVAR*)*, 51
plot.gsmvarpred, 77
plot.irf *(*linear_IRF*)*, 67
plot.qrtest, 78
predict.gsmvar, *20*, *29*, *49*, *80*, 80, *88*, *98*
print.gfevd *(*GFEVD*)*, 42
print.girf *(*GIRF*)*, 46

print.gsmvar, *85*
print.gsmvar *(*GSMVAR*)*, 51
print.gsmvarpred, 82
print.gsmvarsum, 83
print.hypotest, 84
print.irf *(*linear_IRF*)*, 67
print.qrtest *(*plot.qrtest*)*, 78
print_std_errors, *29*, 84
profile_logliks, *8*, *18*, *20*, *23*, *29*, *49*, *66*, *75*, *80*, *85*, 85, *88*, *93*, *98*, *110*

quantile_residual_tests, *18*, *20*, *29*, *49*, *75*, *88*, *98*, *110*
quantile_residual_tests *(*plot.qrtest*)*, 78
quantile_residuals, *76*, *80*, 88

random_ind2, 89
Rao_test, *20*, *75*, 92, *110*
redecompose_Omegas, 93
reorder_W_columns, *29*, *44*, *49*, *56*, *58*, *69*, 95, *100*, *104*, *107*
residuals.gsmvar *(*GSMVAR*)*, 51

simulate.gsmvar, *29*, *44*, *49*, *82*, 97
stmvar_to_gstmvar, *29*, *56*, *58*, *96*, 99, *104*, *107*
summary.gsmvar *(*GSMVAR*)*, 51
swap_parametrization, *29*, *56*, *85*, 101
swap_W_signs, *29*, *44*, *49*, *56*, *58*, *69*, *96*, *100*, 103

uncond_moments, *41*, 104
uncond_moments(), *17*, *40, 41*
update_numtols, *4*, *6*, *29*, *56*, *66*, *80*, *102*, 106
usamon, 107
usamone, 108

Wald_test, *18*, *20*, *29*, *49*, *75*, *80*, *87*, *93*, 109