

# Package ‘gmvarkit’

November 22, 2021

**Title** Estimate Gaussian or Student's t Mixture Vector Autoregressive Model

**Version** 2.0.0

## 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](https://doi.org/10.1016/j.jeconom.2016.02.012)>, Savi Virolainen (2020) <[arXiv:2007.04713](https://arxiv.org/abs/2007.04713)>, Savi Virolainen (2021) <[arXiv:2109.13648](https://arxiv.org/abs/2109.13648)>.

**Depends** R (>= 3.6.0)

**BugReports** <https://github.com/saviviro/gmvarkit/issues>

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.2

**Imports** Brodningnag (>= 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)

**Suggests** testthat, knitr, rmarkdown

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Savi Virolainen [aut, cre]

**Maintainer** Savi Virolainen <[savi.virolainen@helsinki.fi](mailto:savi.virolainen@helsinki.fi)>

**Repository** CRAN

**Date/Publication** 2021-11-22 16:00:02 UTC

**R topics documented:**

add_data . . . . .	3
alt_gmvar . . . . .	4
alt_gsmvar . . . . .	6
calc_gradient . . . . .	7
check_parameters . . . . .	9
cond_moments . . . . .	12
cond_moment_plot . . . . .	17
diagnostic_plot . . . . .	18
diag_Omegas . . . . .	20
fitGMVAR . . . . .	21
fitGSMVAR . . . . .	24
GAfit . . . . .	29
gdpdef . . . . .	36
get_boldA_eigens . . . . .	36
get_omega_eigens . . . . .	37
get_regime_autocovs . . . . .	38
get_regime_means . . . . .	39
GFEVD . . . . .	40
GIRF . . . . .	44
GMVAR . . . . .	48
gmvarkit . . . . .	52
gmvar_to_gsmvar . . . . .	52
gmvar_to_sgmvar . . . . .	53
GSMVAR . . . . .	54
gsmvar_to_sgsmvar . . . . .	59
in_paramspace . . . . .	61
in_paramspace_int . . . . .	64
iterate_more . . . . .	67
loglikelihood . . . . .	69
LR_test . . . . .	73
plot.gmvarpred . . . . .	74
plot.gsmvarpred . . . . .	75
plot.qrtest . . . . .	76
predict.gmvar . . . . .	78
predict.gsmvar . . . . .	80
print.gmvar . . . . .	82
print.gmvarsum . . . . .	83
print.gsmvarpred . . . . .	83
print.gsmvarsum . . . . .	84
print_std_errors . . . . .	85
profile_logliks . . . . .	86
quantile_residuals . . . . .	88
random_ind2 . . . . .	90
recompose_Omegas . . . . .	93
reorder_W_columns . . . . .	94
simulate.gsmvar . . . . .	96

*add\_data* 3

<code>simulateGMVAR</code>	98
<code>stmvar_to_gstmvar</code>	100
<code>swap_parametrization</code>	102
<code>swap_W_signs</code>	103
<code>uncond_moments</code>	105
<code>update_numtols</code>	106
<code>Wald_test</code>	108

**Index** 110

---

<code>add_data</code>	<i>Add data to an object of class 'gsmvar' defining a GMVAR, StMVAR, or G-StMVAR model</i>
-----------------------	--

---

**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**

<code>data</code>	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.
<code>gsmvar</code>	an object of class 'gsmvar', typically created with <code>fitGSMVAR</code> or <code>GSMVAR</code> .
<code>calc_cond_moments</code>	should conditional means and covariance matrices should be calculated? Default is TRUE if the model contains data and FALSE otherwise.
<code>calc_std_errors</code>	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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

**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\_gmvar

*DEPRECATED! USE THE FUNCTION alt\_gsmvar INSTEAD! Construct a GMVAR model based on results from an arbitrary estimation round of fitGSMVAR*

---

**Description**

DEPRECATED! USE THE FUNCTION alt\_gsmvar INSTEAD! alt\_gmvar constructs a GMVAR model based on results from an arbitrary estimation round of fitGSMVAR.

**Usage**

```
alt_gmvar(
  gmvar,
  which_round = 1,
  which_largest,
  calc_cond_moments = TRUE,
```

```

    calc_std_errors = TRUE
  )

```

### Arguments

gmvar	object of class 'gmvar'
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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

### See Also

[alt\\_gsmvar](#)

---

alt_gsmvar	<i>Construct a GMVAR, StMVAR, or G-StMVAR model based on results from an arbitrary estimation round of fitGSMVAR</i>
------------	--

---

### 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

## 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 <b>first</b> 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 but if NULL (default), then the difference h 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.

**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	<i>Check that the given parameter vector satisfies the model assumptions</i>
------------------	--

---

### 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,
  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	<b>For GMVAR and StMVAR models:</b> a positive integer specifying the number of mixture components. <b>For G-StMVAR models:</b> a size (2x1) integer vector specifying the number of <i>GMVAR type</i> components M1 in the first element and <i>StMVAR type</i> 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. <b>For unconstrained models:</b> Should be size $((M(pd^2 + d + d(d + 1)/2 + 2) - M1 - 1)x1)$ and have the form $\theta=(v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where <ul style="list-style-type: none"> <li><math>v_m = (\phi_{m,0}, \phi_m, \sigma_m)</math></li> </ul>

- $\phi_m = (\text{vec}(A_{m,1}), \dots, \text{vec}(A_{m,p}))$
- and  $\sigma_m = \text{vech}(\Omega_m)$ ,  $m=1, \dots, M$ ,
- $\nu = (\nu_{M1+1}, \dots, \nu_M)$
- $M1$  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  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

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

**For same\_means models:** Should have the form  $\theta = (\mu, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

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

**For structural models:** Should have the form  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \phi_1, \dots, \phi_M, \text{vec}(W), \lambda_2, \dots, \lambda_M, \alpha_1, \dots)$ , where

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

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

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

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

**If  $\lambda_{mi}$  are constrained:** Replace  $\lambda_2, \dots, \lambda_M$  with  $\gamma$  ( $rx1$ ) that satisfies  $(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma$  where  $C_\lambda$  is a  $(d(M-1)xr)$  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,  $\Omega_m$  denotes the error term covariance matrix of the  $m$ th mixture component, and  $\alpha_m$  is the mixing weight parameter. The  $W$  and  $\lambda_{mi}$  are structural parameters replacing the error term covariance matrices (see Virolainen, 2020). If  $M = 1$ ,  $\alpha_m$  and  $\lambda_{mi}$  are dropped. If parametrization=="mean", just replace each  $\phi_{m,0}$  with regimewise mean  $\mu_m$ .  $\text{vec}()$  is vectorization operator that stacks columns of a given matrix into a vector.  $\text{vech}()$  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  $\nu$  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  $\nu$  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 $\mu_m$ , $m=1,\dots,M$ .
constraints	a size $(Mpd^2 \times q)$ constraint matrix $C$ specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, \dots, \phi_M) = C\psi$ , where $\phi_m = (vec(A_{m,1}), \dots, vec(A_{m,p}))(pd^2 \times 1)$ , $m = 1, \dots, M$ , 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 $C = [I : \dots : I]'$ ( $Mpd^2 \times pd^2$ ) where $I = \text{diag}(p \times d^2)$ . Ignore (or set to NULL) if linear constraints should <b>not</b> 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 <code>list(1, 2:3)</code> 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. <b>This constraint is available only for mean parametrized models; that is, when parametrization="mean".</b>
structural_pars	<p>If NULL a reduced form model is considered. For structural model, should be a list containing the following elements:</p> <ul style="list-style-type: none"> <li>• <math>W</math> - a <math>(d \times d)</math> matrix with its entries imposing constraints on <math>W</math>: 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.</li> <li>• <math>C\_lambda</math> - a <math>(d(M-1) \times r)</math> constraint matrix that satisfies <math>(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma</math> where <math>\gamma</math> is the new <math>(rx1)</math> parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of <math>C\_lambda</math> must be either <b>positive</b> or <b>zero</b>. Ignore (or set to NULL) if the eigenvalues <math>\lambda_{mi}</math> should not be constrained.</li> </ul> <p>See Virolainen (2020) for the conditions required to identify the shocks and for the B-matrix as well (it is <math>W</math> times a time-varying diagonal matrix with positive diagonal entries).</p>
stat_tol	numerical tolerance for stationarity of the AR parameters: if the "bold A" matrix of any regime has eigenvalues larger than $1 - \text{stat\_tol}$ 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 $2 + \text{df\_tol}$ .

### 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. 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))
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, 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)
```

**Description**

loglikelihood 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,
  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      | <p><b>For GMVAR and StMVAR models:</b> a positive integer specifying the number of mixture components.</p> <p><b>For G-StMVAR models:</b> a size (2x1) integer vector specifying the number of <i>GMVAR type</i> components M1 in the first element and <i>StMVAR type</i> components M2 in the second element. The total number of mixture components is <math>M = M1 + M2</math>.</p>   |
| params | <p>a real valued vector specifying the parameter values.</p> <p><b>For unconstrained models:</b> Should be size <math>((M(pd^2 + d + d(d + 1)/2 + 2) - M1 - 1) \times 1)</math> and have the form <math>\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1}, \nu)</math>, where</p> <ul style="list-style-type: none"> <li>• <math>v_m = (\phi_{m,0}, \phi_m, \sigma_m)</math></li> <li>• <math>\phi_m = (vec(A_{m,1}), \dots, vec(A_{m,p}))</math></li> <li>• and <math>\sigma_m = vech(\Omega_m)</math>, <math>m = 1, \dots, M</math>,</li> <li>• <math>\nu = (v_{M1+1}, \dots, v_M)</math></li> <li>• <math>M1</math> is the number of GMVAR type regimes.</li> </ul> <p><b>For constrained models:</b> Should be size <math>((M(d + d(d + 1)/2 + 2) + q - M1 - 1) \times 1)</math> and have the form <math>\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)</math>, where</p> |

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

**For same\_means models:** Should have the form  $\theta = (\mu, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

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

**For structural models:** Should have the form  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \phi_1, \dots, \phi_M, \text{vec}(W), \lambda_2, \dots, \lambda_M, \alpha_1, \dots)$ , where

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

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

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

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

**If  $\lambda_{mi}$  are constrained:** Replace  $\lambda_2, \dots, \lambda_M$  with  $\gamma$  ( $rx1$ ) that satisfies  $(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma$  where  $C_\lambda$  is a  $(d(M-1)xr)$  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,  $\Omega_m$  denotes the error term covariance matrix of the  $m$ th mixture component, and  $\alpha_m$  is the mixing weight parameter. The  $W$  and  $\lambda_{mi}$  are structural parameters replacing the error term covariance matrices (see Virolainen, 2020). If  $M = 1$ ,  $\alpha_m$  and  $\lambda_{mi}$  are dropped. If parametrization=="mean", just replace each  $\phi_{m,0}$  with regimewise mean  $\mu_m$ .  $\text{vec}()$  is vectorization operator that stacks columns of a given matrix into a vector.  $\text{vech}()$  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  $\nu$  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  $\nu$  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  $\mu_m$ ,  $m=1, \dots, M$ .

constraints

a size  $(Mpd^2xq)$  constraint matrix  $C$  specifying general linear constraints to the autoregressive parameters. We consider constraints of form  $(\phi_1, \dots, \phi_M) = C\psi$ , where  $\phi_m = (\text{vec}(A_{m,1}), \dots, \text{vec}(A_{m,p}))(pd^2x1)$ ,  $m = 1, \dots, M$ , 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  $C = [I : \dots : I]'$  ( $Mpd^2xpd^2$ ) where  $I = \text{diag}(p*d^2)$ . 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 <code>list(1,2:3)</code> 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. <b>This constraint is available only for mean parametrized models; that is, when</b> <code>parametrization="mean"</code> .
structural_pars	<p>If NULL a reduced form model is considered. For structural model, should be a list containing the following elements:</p> <ul style="list-style-type: none"> <li>• <math>W</math> - a <math>(d \times d)</math> matrix with its entries imposing constraints on <math>W</math>: 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.</li> <li>• <math>C\_lambda</math> - a <math>(d(M-1) \times r)</math> constraint matrix that satisfies <math>(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma</math> where <math>\gamma</math> is the new <math>(r \times 1)</math> parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of <math>C\_lambda</math> must be either <b>positive</b> or <b>zero</b>. Ignore (or set to NULL) if the eigenvalues <math>\lambda_{mi}</math> should not be constrained.</li> </ul> <p>See Virolainen (2020) for the conditions required to identify the shocks and for the <math>B</math>-matrix as well (it is <math>W</math> times a time-varying diagonal matrix with positive diagonal entries).</p>
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 than $1 - \text{stat\_tol}$ 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 $2 + \text{df\_tol}$ .

## 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

**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	<i>Conditional mean or variance plot for a GMVAR, StMVAR, or G-StMVAR model</i>
------------------	---

---

### 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

### 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	<i>Quantile residual diagnostic plot for a GMVAR, StMVAR, or G-StMVAR model</i>
-----------------	---

---

**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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

### See Also

[profile\\_logliks](#), [fitGSMVAR](#), [GSMVAR](#), [quantile\\_residual\\_tests](#), [LR\\_test](#), [Wald\\_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 ( $d \times d$ ) covariance matrix ( $d > 1$ )  
Omega2            another positive definite ( $d \times d$ ) 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

```

fitGMVAR

*DEPRECATED! USE THE FUNCTION fitGSMVAR INSTEAD! Two-phase maximum likelihood estimation of a GMVAR model*

**Description**

DEPRECATED! USE THE FUNCTION fitGSMVAR INSTEAD! fitGMVAR estimates a GMVAR 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**

```

fitGMVAR(
  data,
  p,
  M,
  conditional = TRUE,
  parametrization = c("intercept", "mean"),
  constraints = NULL,
  same_means = NULL,
  structural_pars = NULL,
  ncalls = M^6,
  ncores = 2,
  maxit = 1000,
  seeds = NULL,
  print_res = 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	<p><b>For GMVAR and StMVAR models:</b> a positive integer specifying the number of mixture components.</p> <p><b>For G-StMVAR models:</b> a size (2x1) integer vector specifying the number of <i>GMVAR type</i> components M1 in the first element and <i>StMVAR type</i> components M2 in the second element. The total number of mixture components is <math>M=M1+M2</math>.</p>
conditional parametrization	<p>a logical argument specifying whether the conditional or exact log-likelihood function</p> <p>"intercept" or "mean" determining whether the model is parametrized with intercept parameters <math>\phi_{m,0}</math> or regime means <math>\mu_m</math>, <math>m=1,\dots,M</math>.</p>
constraints	<p>a size <math>(Mpd^2xq)</math> constraint matrix <math>C</math> specifying general linear constraints to the autoregressive parameters. We consider constraints of form <math>(\phi_1, \dots, \phi_M) = C\psi</math>, where <math>\phi_m = (vec(A_{m,1}), \dots, vec(A_{m,p}))(pd^2x1)</math>, <math>m = 1, \dots, M</math>, contains the coefficient matrices and <math>\psi</math> (<math>qx1</math>) contains the related parameters. For example, to restrict the AR-parameters to be the same for all regimes, set <math>C = [I : \dots : I]'</math> (<math>Mpd^2xpd^2</math>) where <math>I = \text{diag}(p \times d^2)</math>. Ignore (or set to NULL) if linear constraints should <b>not</b> be employed.</p>
same_means	<p>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 <math>M=3</math>, the argument <code>list(1, 2:3)</code> 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. <b>This constraint is available only for mean parametrized models; that is, when parametrization="mean"</b>.</p>
structural_pars	<p>If NULL a reduced form model is considered. For structural model, should be a list containing the following elements:</p> <ul style="list-style-type: none"> <li>• <math>W</math> - a <math>(dxd)</math> matrix with its entries imposing constraints on <math>W</math>: 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.</li> <li>• <math>C\_lambda</math> - a <math>(d(M-1)xr)</math> constraint matrix that satisfies <math>(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma</math> where <math>\gamma</math> is the new <math>(rx1)</math> parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of <math>C\_lambda</math> must be either <b>positive</b> or <b>zero</b>. Ignore (or set to NULL) if the eigenvalues <math>\lambda_{mi}</math> should not be constrained.</li> </ul> <p>See Virolainen (2020) for the conditions required to identify the shocks and for the B-matrix as well (it is <math>W</math> times a time-varying diagonal matrix with positive diagonal entries).</p>
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 <code>ncalls</code> 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?  
 ...              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 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 overidentifying 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`.

### 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.

### 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

### See Also

[fitGSMVAR](#)

---

fitGSMVAR

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

---



## Description

fitGSMVAR estimates a GMVAR, StMVAR, or G-StMVAR 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,
  structural_pars = NULL,
  ncalls = M^6,
  ncores = 2,
  maxit = 1000,
  seeds = NULL,
  print_res = 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	<b>For GMVAR and StMVAR models:</b> a positive integer specifying the number of mixture components. <b>For G-StMVAR models:</b> a size (2x1) integer vector specifying the number of <i>GMVAR type</i> components M1 in the first element and <i>StMVAR type</i> 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 $\mu_m$ , $m=1, \dots, M$ .
constraints	a size $(Mpd^2 \times q)$ constraint matrix $C$ specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, \dots, \phi_M) = C\psi$ ,

where  $\phi_m = (\text{vec}(A_{m,1}), \dots, \text{vec}(A_{m,p}))$  ( $pd^2 \times 1$ ),  $m = 1, \dots, M$ , 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  $C = [I : \dots : I]'$  ( $Mpd^2 \times pd^2$ ) where  $I = \text{diag}(p \times d^2)$ . 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 <code>list(1,2:3)</code> 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. <b>This constraint is available only for mean parametrized models; that is, when</b> <code>parametrization="mean"</code> .
structural_pars	If NULL a reduced form model is considered. For structural model, should be a list containing the following elements: <ul style="list-style-type: none"> <li>• <math>W</math> - a (<math>d \times d</math>) matrix with its entries imposing constraints on <math>W</math>: 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.</li> <li>• <math>C\_lambda</math> - a (<math>d(M-1) \times r</math>) constraint matrix that satisfies <math>(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma</math> where <math>\gamma</math> is the new (<math>rx1</math>) parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of <math>C\_lambda</math> must be either <b>positive</b> or <b>zero</b>. Ignore (or set to NULL) if the eigenvalues <math>\lambda_{mi}</math> should not be constrained.</li> </ul> <p>See Virolainen (2020) for the conditions required to identify the shocks and for the B-matrix as well (it is <math>W</math> times a time-varying diagonal matrix with positive diagonal entries).</p>
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 <code>ncalls</code> 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?
...	additional settings passed to the function <code>GAFit</code> employing the genetic algorithm.

## Details

If you wish to estimate a structural model without overidentifying constraints that is identified statistically, specify your  $W$  matrix in `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 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 overidentifying 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`.

**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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

**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\\_sgsvar](#), [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

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

```

## 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,
  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	<b>For GMVAR and StMVAR models:</b> a positive integer specifying the number of mixture components. <b>For G-StMVAR models:</b> a size (2x1) integer vector specifying the number of <i>GMVAR type</i> components M1 in the first element and <i>StMVAR type</i> 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 parametrization	a logical argument specifying whether the conditional or exact log-likelihood function "intercept" or "mean" determining whether the model is parametrized with intercept parameters $\phi_{m,0}$ or regime means $\mu_m$ , $m=1,\dots,M$ .
constraints	a size $(Mpd^2xq)$ constraint matrix $C$ specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, \dots, \phi_M) = C\psi$ , where $\phi_m = (vec(A_{m,1}), \dots, vec(A_{m,p}))(pd^2x1)$ , $m = 1, \dots, M$ , 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 $C = [I : \dots : I]'$ $(Mpd^2xpd^2)$ where $I = \text{diag}(p \times d^2)$ . Ignore (or set to NULL) if linear constraints should <b>not</b> 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 <code>list(1, 2:3)</code> 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. <b>This constraint is available only for mean parametrized models; that is, when parametrization="mean"</b> .
structural_pars	If NULL a reduced form model is considered. For structural model, should be a list containing the following elements: <ul style="list-style-type: none"> <li>• <math>W</math> - a <math>(d \times d)</math> matrix with its entries imposing constraints on <math>W</math>: 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.</li> <li>• <math>C\_lambda</math> - a <math>(d(M-1) \times r)</math> constraint matrix that satisfies <math>(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma</math> where <math>\gamma</math> is the new <math>(r \times 1)</math> parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of <math>C\_lambda</math> must be either <b>positive</b> or <b>zero</b>. Ignore (or set to NULL) if the eigenvalues <math>\lambda_{mi}</math> should not be constrained.</li> </ul> <p>See Virolainen (2020) for the conditions required to identify the shocks and for the B-matrix as well (it is <math>W</math> times a time-varying diagonal matrix with positive diagonal entries).</p>
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 $10 \times n\_params$ .
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  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

- $v_m = (\phi_{m,0}, \phi_m, \sigma_m)$
- $\phi_m = (\text{vec}(A_{m,1}), \dots, \text{vec}(A_{m,p}))$
- and  $\sigma_m = \text{vech}(\Omega_m)$ ,  $m=1, \dots, M$ ,
- $\nu = (\nu_{M1+1}, \dots, \nu_M)$
- $M1$  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  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

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

**For same\_means models:** Should have the form  $\theta = (\mu, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

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

**For structural models:** Should have the form  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \phi_1, \dots, \phi_M, \text{vec}(W), \lambda_2, \dots, \lambda_M, \alpha_1, \dots)$ , where

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

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

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

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

**If  $\lambda_{mi}$  are constrained:** Replace  $\lambda_2, \dots, \lambda_M$  with  $\gamma (rx1)$  that satisfies  $(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma$  where  $C_\lambda$  is a  $(d(M-1) \times r)$  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,  $\Omega_m$  denotes the error term covariance matrix of the  $m$ th mixture component, and  $\alpha_m$  is the mixing weight parameter. The  $W$  and  $\lambda_{mi}$  are structural parameters replacing the error term covariance matrices (see Virolainen, 2020). If  $M = 1$ ,  $\alpha_m$  and  $\lambda_{mi}$  are dropped. If parametrization=="mean", just replace each  $\phi_{m,0}$  with regimewise mean  $\mu_m$ .  $\text{vec}()$  is vectorization operator that stacks columns of a given matrix into a vector.  $\text{vech}()$  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  $\nu$  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  $\nu$  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  $\mu_m$  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  $\mu_m$  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)`. 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  $\Omega_1$  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  $\lambda_{mi}$  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  $m$ th regime. The expected value of the main **diagonal** elements  $ij$  of the  $m$ th ( $m > 1$ ) 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  $i$ th row of  $W$  and `ind_fun` is an indicator function that takes the value one iff the  $ij$ th 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  $(d(M - 1) \times r)$  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  $\gamma$  parameters are drawn from (the  $\gamma$  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!

<code>ar_scale</code>	a positive real number 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 $(0, 0.5)$ uniform distribution in each iteration.
<code>upper_ar_scale</code>	the upper bound for <code>ar_scale</code> 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 <code>p</code> one might want to try smaller upper bound (e.g., 0.5).
<code>ar_scale2</code>	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 <b>smaller</b> coefficients (in absolute value). <b>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</b>
<code>regime_force_scale</code>	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.
<code>red_criteria</code>	a length 2 numeric vector specifying the criteria that is used to determine whether a regime is redundant (or "wasted") or not. Any regime <code>m</code> which satisfies <code>sum(mixingWeights[,m] &gt; red_criteria[1]) &lt; red_criteria[2]*n_obs</code> will be considered "redundant". One should be careful when adjusting this argument (set <code>c(0, 0)</code> to fully disable the 'redundant regime' features from the algorithm).
<code>pre_smart_mu_prob</code>	A number in $[0, 1]$ giving a probability of a "smart mutation" occurring randomly in each iteration before the iteration given by the argument <code>smart_mu</code> .
<code>to_return</code>	should the genetic algorithm return the best fitting individual which has "positive enough" mixing weights for as many regimes as possible (" <code>alt_ind</code> ") or the individual which has the highest log-likelihood in general (" <code>best_ind</code> ") but might have more wasted regimes?
<code>minval</code>	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 <code>minval</code> and the corresponding individuals will never survive. The default is $-(10^{\lceil \log_{10}(n\_obs) \rceil} + d) - 1$ .
<code>seed</code>	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 <code>GAfit</code> from <code>fitGSMVAR</code> , use the argument <code>seeds</code> instead of passing the argument <code>seed</code> .

## 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.

## 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

## 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	<i>U.S. real GDP percent change and GDP implicit price deflator percent change.</i>
--------	---

---

**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 and the Federal Reserve Bank of Atlanta's website

---

get_boldA_eigens	<i>Calculate absolute values of the eigenvalues of the "bold A" matrices containing the AR coefficients</i>
------------------	---

---

**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  $m$ th column contains the absolute values (or modulus) of the eigenvalues of the "bold A" matrix containing the AR coefficients corresponding to regime  $m$ .

**References**

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. 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	<i>Calculate the eigenvalues of the "Omega" error term covariance matrices</i>
------------------	--

---

**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  $m$ th column contains the eigenvalues of the "Omega" error term covariance 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.
- Virolainen S. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. 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  $(d \times d \times p + 1 \times M)$  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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

### 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)
```

---

<code>get_regime_means</code>	<i>Calculate regime means <math>\mu_m</math></i>
-------------------------------	--

---

### 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  $\mu_m$  in the  $m$ :th column,  $m = 1, \dots, M$ .

## References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. 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 a structural GMVAR, StMVAR, or G-StMVAR model.*

---

## Description

GFEVD estimates generalized generalized forecast error variance decomposition for a structural GMVAR, StMVAR, or G-StMVAR model.



**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**

<code>gsmvar</code>	an object of class 'gsmvar', typically created with <code>fitGSMVAR</code> or <code>GSMVAR</code> .
<code>shock_size</code>	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.
<code>N</code>	a positive integer specifying the horizon how far ahead should the GFEVD be calculated.
<code>initval_type</code>	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 <code>init_regimes</code> . The number of initial values is set with the argument <code>R2</code> . "fixed": Estimate the GIRF for a fixed initial value only, which is specified with the argument <code>init_values</code> .
<code>R1</code>	the number of repetitions used to estimate GIRF for each initial value.
<code>R2</code>	the number of initial values to be drawn if <code>initval_type="random"</code> .
<code>init_regimes</code>	a numeric vector of length at most $M$ and elements in $1, \dots, M$ 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  $m$ th regime. Ignored if the argument `init_values` is specified.

<code>init_values</code>	a size $(pxd)$ matrix specifying the initial values, where $d$ is the number of time series in the system. The <b>last</b> 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 specified by the argument <code>init_regimes</code> .
<code>which_cumulative</code>	a numeric vector with values in $1, \dots, d$ ( $d = \text{ncol}(\text{data})$ ) specifying which the variables for which the impulse responses should be cumulative. Default is none.
<code>include_mixweights</code>	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.
<code>ncores</code>	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).
<code>seeds</code>	a numeric vector containing the random number generator seed for estimation of each GIRF. Should have the length... <ul style="list-style-type: none"> <li>• <code>...nrow(data) - p + 1</code> if <code>initval_type="data"</code>.</li> <li>• <code>...R2</code> if <code>initval_type="random"</code>.</li> <li>• <code>...1</code> if <code>initval_type="fixed."</code>.</li> </ul> Set to NULL for not initializing the seed. Exists for creating reproducible results.
<code>x</code>	object of class 'gfevd' generated by the function GFEVD.
<code>...</code>	currently not used.
<code>digits</code>	the number of decimals to print
<code>N_to_print</code>	an integer specifying the horizon how far to print the estimates. The default is that all the values are printed.

### Details

The model needs to be structural in order for this function to be applicable. A structural GMVAR, StMVAR, or G-StMVAR model can be estimated by specifying the argument `structural_pars` in the function `fitGSMVAR`.

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 (2020).

### 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.

**Methods (by generic)**

- plot: plot method
- print: print method

**References**

- Lanne M. and Nyberg H. 2016. Generalized Forecast Error Variance Decomposition for Linear and Nonlinear 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

**See Also**

[GIRF](#), [fitGSMVAR](#), [GSMVAR](#), [gsmvar\\_to\\_sgsmlvar](#), [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.

# 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.

## NOTE: Use larger R1 in empirical applications! Small R1 is used
## Below only to fasten the execution time of the examples.

# Estimating the GFEVD using all possible histories in the data as the
# initial values:
gfevd1 <- GFEVD(mod22s, N=24, R1=20, 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=20, 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	<i>Estimate generalized impulse response function for a structural GMVAR, StMVAR, or G-StMVAR model.</i>
------	--

---

### Description

GIRF estimates generalized impulse response function for a structural GMVAR, StMVAR, or G-StMVAR model.

### 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

<code>gsmvar</code>	an object of class 'gsmvar', typically created with <code>fitGSMVAR</code> or <code>GSMVAR</code> .
<code>which_shocks</code>	a numeric vector of length at most $d$ ( $=\text{ncol}(\text{data})$ ) and elements in $1, \dots, d$ specifying the structural shocks for which the GIRF should be estimated.
<code>shock_size</code>	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.
<code>N</code>	a positive integer specifying the horizon how far ahead should the generalized impulse responses be calculated.
<code>R1</code>	the number of repetitions used to estimate GIRF for each initial value.
<code>R2</code>	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 <code>init_value</code> is specified.
<code>init_regimes</code>	a numeric vector of length at most $M$ and elements in $1, \dots, M$ 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 <code>init_regimes=1:M</code> , the initial values are generated from the stationary distribution of the process and if <code>init_regimes=m</code> , the initial value are generated from the stationary distribution of the $m$ th regime. Ignored if the argument <code>init_values</code> is specified.
<code>init_values</code>	a size $(pxd)$ matrix specifying the initial values, where $d$ is the number of time series in the system. The <b>last</b> 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 specified by the argument <code>init_regimes</code> .
<code>which_cumulative</code>	a numeric vector with values in $1, \dots, d$ ( $d=\text{ncol}(\text{data})$ ) specifying which the variables for which the impulse responses should be cumulative. Default is none.
<code>scale</code>	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 <code>scale_type</code> )? Provide a length three vector where the shock of interest is given in the first element (an integer in $1, \dots, d$ ), the variable of interest is given in the second element (an integer in $1, \dots, d$ ), 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.

<code>scale_type</code>	If argument <code>scale</code> 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.
<code>scale_horizon</code>	If <code>scale_type == "peak"</code> what the maximum horizon up to which peak response is expected? Scaling won't be based on values after this.
<code>ci</code>	a numeric vector with elements in $(0, 1)$ specifying the confidence levels of the confidence intervals.
<code>include_mixweights</code>	should the generalized impulse response be calculated for the mixing weights as well? TRUE or FALSE.
<code>ncores</code>	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).
<code>plot_res</code>	TRUE if the results should be plotted, FALSE if not.
<code>seeds</code>	a length R2 vector containing the random number generator seed for estimation of each GIRF. A single number or an initial value is specified. or NULL for not initializing the seed. Exists for creating reproducible results.
<code>...</code>	arguments passed to <code>grid</code> which plots <code>grid</code> to the figure.
<code>x</code>	object of class 'girf' generated by the function GIRF.
<code>add_grid</code>	should <code>grid</code> be added to the plots?
<code>margs</code>	numeric vector of length four that adjusts the [ <code>bottom_marginal</code> , <code>left_marginal</code> , <code>top_marginal</code> , <code>right_marginal</code> ] as the relative sizes of the marginals to the figures of the responses.
<code>digits</code>	the number of decimals to print
<code>N_to_print</code>	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 needs to be structural in order for this function to be applicable. A structural GMVAR, StMVAR, or G-StMVAR model can be estimated by specifying the argument `structural_pars` in the function `fitGSMVAR`.

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 (2020) 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  $m$ th 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.

**Methods (by generic)**

- plot: plot method
- print: print method

**References**

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

**See Also**

[GFEVD](#), [fitGSMVAR](#), [GSMVAR](#), [gsmvar\\_to\\_sgsvar](#), [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.

# 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

*DEPRECATED! USE THE FUNCTION GSMVAR INSTEAD! Create a class 'gsmvar' object defining a reduced form or structural GMVAR model*

---

## Description

GSMVAR creates a class 'gsmvar' object that defines a reduced form or structural GMVAR model  
 DEPRECATED! USE THE FUNCTION GSMVAR INSTEAD!

## Usage

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

## 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	<b>For GMVAR and StMVAR models:</b> a positive integer specifying the number of mixture components.



**For G-StMVAR models:** a size  $(2 \times 1)$  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  $((M(pd^2 + d + d(d+1)/2 + 2) - M1 - 1) \times 1)$  and have the form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

- $v_m = (\phi_{m,0}, \phi_m, \sigma_m)$
- $\phi_m = (vec(A_{m,1}), \dots, vec(A_{m,p}))$
- and  $\sigma_m = vech(\Omega_m)$ ,  $m=1, \dots, M$ ,
- $\nu = (\nu_{M1+1}, \dots, \nu_M)$
- $M1$  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  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

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

**For same\_means models:** Should have the form  $\theta = (\mu, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

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

**For structural models:** Should have the form  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \phi_1, \dots, \phi_M, vec(W), \lambda_2, \dots, \lambda_M, \alpha_1, \dots)$ , where

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

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

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

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

**If  $\lambda_{mi}$  are constrained:** Replace  $\lambda_2, \dots, \lambda_M$  with  $\gamma (rx1)$  that satisfies  $(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma$  where  $C_\lambda$  is a  $(d(M-1) \times r)$  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,  $\Omega_m$  denotes the error term covariance matrix of the  $m$ th mixture component, and  $\alpha_m$  is the mixing weight parameter. The  $W$  and  $\lambda_{mi}$  are structural parameters replacing the error term covariance matrices (see Virolainen, 2020). If  $M = 1$ ,  $\alpha_m$  and  $\lambda_{mi}$  are dropped. If parametrization="mean", just replace each  $\phi_{m,0}$  with regimewise mean  $\mu_m$ .  $vec()$  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  $\nu$  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  $\nu$  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.
parametrization	"intercept" or "mean" determining whether the model is parametrized with intercept parameters $\phi_{m,0}$ or regime means $\mu_m$ , $m=1,\dots,M$ .
constraints	a size $(Mpd^2 \times q)$ constraint matrix $C$ specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, \dots, \phi_M) = C\psi$ , where $\phi_m = (vec(A_{m,1}), \dots, vec(A_{m,p}))(pd^2 \times 1)$ , $m = 1, \dots, M$ , 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 $C = [I : \dots : I]'$ ( $Mpd^2 \times pd^2$ ) where $I = \text{diag}(p \times d^2)$ . Ignore (or set to NULL) if linear constraints should <b>not</b> 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 <code>list(1,2:3)</code> 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. <b>This constraint is available only for mean parametrized models; that is, when parametrization="mean"</b> .
structural_pars	If NULL a reduced form model is considered. For structural model, should be a list containing the following elements: <ul style="list-style-type: none"> <li>• <math>W</math> - a <math>(d \times d)</math> matrix with its entries imposing constraints on <math>W</math>: 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.</li> <li>• <math>C\_lambda</math> - a <math>(d(M-1) \times r)</math> constraint matrix that satisfies <math>(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma</math> where <math>\gamma</math> is the new <math>(rx1)</math> parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of <math>C\_lambda</math> must be either <b>positive</b> or <b>zero</b>. Ignore (or set to NULL) if the eigenvalues <math>\lambda_{mi}</math> should not be constrained.</li> </ul> <p>See Virolainen (2020) for the conditions required to identify the shocks and for the <math>B</math>-matrix as well (it is <math>W</math> times a time-varying diagonal matrix with positive diagonal entries).</p>
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 than $1 - \text{stat\_tol}$ 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.

### 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 `gmvar` t.

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

### About S3 methods

Only the `print` method is available if data is not provided. If data is provided, then in addition to the ones listed above, the `predict` method is also available.

### 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

### See Also

[GSMVAR](#)

---

gmvarkit	<i>gmvarkit: Estimate Gaussian Mixture Vector Autoregressive, Student's t Mixture Vector Autoregressive, or Gaussian and Student's t Mixture Vector Autoregressive model</i>
----------	--

---

### 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.

---

gmvar_to_gsmvar	<i>Makes class 'gmvar' objects compatible with the functions using class 'gsmvar' objects</i>
-----------------	---

---

### Description

gmvar\_to\_gsmvar class 'gmvar' objects compatible with the functions using s class 'gsmvar' objects

### Usage

```
gmvar_to_gsmvar(gmvar)
```

### Arguments

gmvar 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.

---

gmvar_to_sgmvvar	<i>DEPRECATED! USE THE FUNCTION fitGSMVAR INSTEAD! Switch from two-regime reduced form GMVAR model to a structural model.</i>
------------------	---

---

### Description

DEPRECATED! USE THE FUNCTION fitGSMVAR INSTEAD! gmvar\_to\_sgsmvar constructs SGMVAR model based on a reduced form GMVAR, StMVAR, or G-StMVAR model.

### Usage

```
gmvar_to_sgmvvar(gmvar, calc_std_errors = TRUE)
```

### Arguments

gmvar	object of class 'gmvar'
calc_std_errors	should approximate standard errors be calculated?

### 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 than two regimes are not supported because the matrix decomposition does not generally exist 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.

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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.

- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

### See Also

[gsmvar\\_to\\_sgsmvar](#)

---

GSMVAR	<i>Create a class 'gsmvar' object defining a reduced form or structural GMVAR, StMVAR, or G-StMVAR model</i>
--------	--

---

### 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,
  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  $(2 \times 1)$  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.  $\text{ncol}(\text{data})$ . 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  $((M(pd^2 + d + d(d + 1)/2 + 2) - M1 - 1) \times 1)$  and have the form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

- $v_m = (\phi_{m,0}, \phi_m, \sigma_m)$
- $\phi_m = (\text{vec}(A_{m,1}), \dots, \text{vec}(A_{m,p}))$
- and  $\sigma_m = \text{vech}(\Omega_m)$ ,  $m=1, \dots, M$ ,
- $\nu = (\nu_{M1+1}, \dots, \nu_M)$
- $M1$  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  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

- $\psi$  ( $q \times 1$ ) satisfies  $(\phi_1, \dots, \phi_M) = C\psi$  where  $C$  is a  $(Mpd^2 \times q)$  constraint matrix.

**For same\_means models:** Should have the form  $\theta = (\mu, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

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

**For structural models:** Should have the form  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \phi_1, \dots, \phi_M, \text{vec}(W), \lambda_2, \dots, \lambda_M, \alpha_1, \dots)$ , where

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

**If AR parameters are constrained:** Replace  $\phi_1, \dots, \phi_M$  with  $\psi$  ( $q \times 1$ ) that satisfies  $(\phi_1, \dots, \phi_M) = C\psi$ , as above.

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

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

**If  $\lambda_{mi}$  are constrained:** Replace  $\lambda_2, \dots, \lambda_M$  with  $\gamma$  ( $rx1$ ) that satisfies  $(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma$  where  $C_\lambda$  is a  $(d(M-1) \times r)$  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,  $\Omega_m$  denotes the error term covariance matrix of the  $m$ th mixture component, and  $\alpha_m$  is the mixing weight parameter. The  $W$  and  $\lambda_{mi}$  are structural parameters replacing the error term covariance matrices (see Virolainen, 2020). If  $M = 1$ ,  $\alpha_m$  and  $\lambda_{mi}$  are dropped. If parametrization="mean", just replace each  $\phi_{m,0}$  with regimewise mean  $\mu_m$ .  $vec()$  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  $\nu$  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  $\nu$  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 $\mu_m$ , $m=1, \dots, M$ .
constraints	a size $(Mpd^2 \times xq)$ constraint matrix $C$ specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, \dots, \phi_M) = C\psi$ , where $\phi_m = (vec(A_{m,1}), \dots, vec(A_{m,p}))(pd^2 \times 1)$ , $m = 1, \dots, M$ , 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 $C = [I : \dots : I]'$ ( $Mpd^2 \times pd^2$ ) where $I = \text{diag}(p \times d^2)$ . Ignore (or set to NULL) if linear constraints should <b>not</b> 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 <code>list(1,2:3)</code> 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. <b>This constraint is available only for mean parametrized models; that is, when parametrization="mean".</b>
structural_pars	If NULL a reduced form model is considered. For structural model, should be a list containing the following elements:



- $W$  - a  $(d \times d)$  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  $(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma$  where  $\gamma$  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  $\lambda_{mi}$  should not be constrained.

See Virolainen (2020) 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).

<code>calc_cond_moments</code>	should conditional means and covariance matrices should be calculated? Default is TRUE if the model contains data and FALSE otherwise.
<code>calc_std_errors</code>	should approximate standard errors be calculated?
<code>stat_tol</code>	numerical tolerance for stationarity of the AR parameters: if the "bold A" matrix of any regime has eigenvalues larger than $1 - \text{stat\_tol}$ 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.
<code>posdef_tol</code>	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.
<code>df_tol</code>	the parameter vector is considered to be outside the parameter space if all degrees of freedom parameters are not larger than $2 + \text{df\_tol}$ .
<code>object</code>	object of class 'gsmvar' generated by <code>fitGSMVAR</code> or <code>GSMVAR</code> .
<code>...</code>	currently not used.
<code>digits</code>	number of digits to be printed.
<code>x</code>	object of class 'gsmvar' generated by <code>fitGSMVAR</code> or <code>GSMVAR</code> .
<code>type</code>	which type figure should be produced? Or both?
<code>summary_print</code>	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.

### Methods (by generic)

- `logLik`: Log-likelihood method
- `residuals`: residuals method to extract multivariate quantile residuals
- `summary`: summary method
- `plot`: plot method for class 'gsmvar'
- `print`: 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

### 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	<i>Switch from two-regime reduced form GMVAR, StMVAR, or G-StMVAR model to a structural model.</i>
-------------------	--

---

## 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)
```

## Arguments

gsmvar	an object of class 'gsmvar', typically created with fitGSMVAR or GSMVAR.
calc_std_errors	should approximate standard errors be calculated?

## 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 than two regimes are not supported because the matrix decomposition does not generally exist 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.

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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

## 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	<i>Determine whether the parameter vector lies in the parameter space</i>
---------------	---

---

## 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,
  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      | <p><b>For GMVAR and StMVAR models:</b> a positive integer specifying the number of mixture components.</p> <p><b>For G-StMVAR models:</b> a size (2x1) integer vector specifying the number of <i>GMVAR type</i> components M1 in the first element and <i>StMVAR type</i> components M2 in the second element. The total number of mixture components is <math>M=M1+M2</math>.</p>   |
| d      | the number of time series in the system.  |
| params | <p>a real valued vector specifying the parameter values.</p> <p><b>For unconstrained models:</b> Should be size <math>((M(pd^2 + d + d(d + 1)/2 + 2) - M1 - 1)x1)</math> and have the form <math>\theta=(v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1}, \nu)</math>, where</p> <ul style="list-style-type: none"> <li>• <math>v_m = (\phi_{m,0}, \phi_m, \sigma_m)</math></li> <li>• <math>\phi_m = (vec(A_{m,1}), \dots, vec(A_{m,p}))</math></li> <li>• and <math>\sigma_m = vech(\Omega_m)</math>, <math>m=1, \dots, M</math>,</li> </ul> |

- $\nu = (\nu_{M1+1}, \dots, \nu_M)$
- $M1$  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  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

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

**For same\_means models:** Should have the form  $\theta = (\mu, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

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

**For structural models:** Should have the form  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \phi_1, \dots, \phi_M, \text{vec}(W), \lambda_2, \dots, \lambda_M, \alpha_1, \dots)$ , where

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

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

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

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

**If  $\lambda_{mi}$  are constrained:** Replace  $\lambda_2, \dots, \lambda_M$  with  $\gamma$  ( $rx1$ ) that satisfies  $(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma$  where  $C_\lambda$  is a  $(d(M - 1)xr)$  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,  $\Omega_m$  denotes the error term covariance matrix of the  $m$ th mixture component, and  $\alpha_m$  is the mixing weight parameter. The  $W$  and  $\lambda_{mi}$  are structural parameters replacing the error term covariance matrices (see Virolainen, 2020). If  $M = 1$ ,  $\alpha_m$  and  $\lambda_{mi}$  are dropped. If parametrization="mean", just replace each  $\phi_{m,0}$  with regimewise mean  $\mu_m$ .  $\text{vec}()$  is vectorization operator that stacks columns of a given matrix into a vector.  $\text{vech}()$  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  $\nu$  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  $\nu$  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 $(Mpd^2xq)$ constraint matrix $C$ specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, \dots, \phi_M) = C\psi$ , where $\phi_m = (\text{vec}(A_{m,1}), \dots, \text{vec}(A_{m,p}))(pd^2x1)$ , $m = 1, \dots, M$ , 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  $C=[I:\dots:I]'$  ( $Mpd^2 \times pd^2$ ) where  $I = \text{diag}(p \times d^2)$ . 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 <code>list(1,2:3)</code> 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. <b>This constraint is available only for mean parametrized models; that is, when</b> <code>parametrization="mean"</code> .
structural_pars	<p>If NULL a reduced form model is considered. For structural model, should be a list containing the following elements:</p> <ul style="list-style-type: none"> <li>• <math>W</math> - a (<math>d \times d</math>) matrix with its entries imposing constraints on <math>W</math>: 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.</li> <li>• <math>C\_lambda</math> - a (<math>d(M-1) \times r</math>) constraint matrix that satisfies <math>(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma</math> where <math>\gamma</math> is the new (<math>r \times 1</math>) parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of <math>C\_lambda</math> must be either <b>positive</b> or <b>zero</b>. Ignore (or set to NULL) if the eigenvalues <math>\lambda_{mi}</math> should not be constrained.</li> </ul> <p>See Virolainen (2020) for the conditions required to identify the shocks and for the B-matrix as well (it is <math>W</math> times a time-varying diagonal matrix with positive diagonal entries).</p>
stat_tol	numerical tolerance for stationarity of the AR parameters: if the "bold A" matrix of any regime has eigenvalues larger than $1 - \text{stat\_tol}$ 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 $2 + \text{df\_tol}$ .

### 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. 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 (2x1) 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  $\theta=(v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where
- $v_m = (\phi_{m,0}, \phi_m, \sigma_m)$
  - $\phi_m = (vec(A_{m,1}), \dots, vec(A_{m,p}))$
  - and  $\sigma_m = vech(\Omega_m)$ ,  $m=1, \dots, M$ ,
  - $\nu = (\nu_{M1+1}, \dots, \nu_M)$
  - $M1$  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  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where
- $\psi$  ( $qx1$ ) satisfies  $(\phi_1, \dots, \phi_M) = C\psi$  where  $C$  is a  $(Mpd^2xq)$  constraint matrix.
- For same\_means models:** Should have the form  $\theta = (\mu, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where
- $\mu = (\mu_1, \dots, \mu_g)$  where  $\mu_i$  is the mean parameter for group  $i$  and  $g$  is the number of groups.
  - If AR constraints are employed,  $\psi$  is as for constrained models, and if AR constraints are not employed,  $\psi = (\phi_1, \dots, \phi_M)$ .
- For structural models:** Should have the form  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \phi_1, \dots, \phi_M, vec(W), \lambda_2, \dots, \lambda_M, \alpha_1, \dots)$ , where
- $\lambda_m = (\lambda_{m1}, \dots, \lambda_{md})$  contains the eigenvalues of the  $m$ th mixture component.
- If AR parameters are constrained:** Replace  $\phi_1, \dots, \phi_M$  with  $\psi$  ( $qx1$ ) that satisfies  $(\phi_1, \dots, \phi_M) = C\psi$ , as above.

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

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

**If  $\lambda_{mi}$  are constrained:** Replace  $\lambda_2, \dots, \lambda_M$  with  $\gamma(r x 1)$  that satisfies  $(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma$  where  $C_\lambda$  is a  $(d(M-1) \times r)$  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,  $\Omega_m$  denotes the error term covariance matrix of the  $m$ th mixture component, and  $\alpha_m$  is the mixing weight parameter. The  $W$  and  $\lambda_{mi}$  are structural parameters replacing the error term covariance matrices (see Virolainen, 2020). If  $M = 1$ ,  $\alpha_m$  and  $\lambda_{mi}$  are dropped. If parametrization=="mean", just replace each  $\phi_{m,0}$  with regimewise mean  $\mu_m$ .  $vec()$  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  $\nu$  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  $\nu$  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 $\Omega_m$ , obtained from pick_Omeegas.
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 than $1 - \text{stat\_tol}$ 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 $2 + \text{df\_tol}$ .

## 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

---

iterate_more	<i>Maximum likelihood estimation of a GMVAR, StMVAR, or G-StMVAR model with preliminary estimates</i>
--------------	---

---

**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	should approximate standard errors be calculated?

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 than $1 - \text{stat\_tol}$ 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 $2 + \text{df\_tol}$ .

### 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

### 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
```

---

loglikelihood	<i>Compute log-likelihood of a GMVAR, StMVAR, or G-StMVAR model using parameter vector</i>
---------------	--

---

**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,
  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  $(2 \times 1)$  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  $((M(pd^2 + d + d(d+1)/2 + 2) - M1 - 1) \times 1)$  and have the form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

- $v_m = (\phi_{m,0}, \phi_m, \sigma_m)$
- $\phi_m = (vec(A_{m,1}), \dots, vec(A_{m,p}))$
- and  $\sigma_m = vech(\Omega_m)$ ,  $m=1, \dots, M$ ,
- $\nu = (\nu_{M1+1}, \dots, \nu_M)$
- $M1$  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  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

- $\psi$  ( $q \times 1$ ) satisfies  $(\phi_1, \dots, \phi_M) = C\psi$  where  $C$  is a  $(Mpd^2 \times q)$  constraint matrix.

**For same\_means models:** Should have the form  $\theta = (\mu, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1}, \nu)$ , where

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

**For structural models:** Should have the form  $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \phi_1, \dots, \phi_M, vec(W), \lambda_2, \dots, \lambda_M, \alpha_1, \dots)$ , where

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

**If AR parameters are constrained:** Replace  $\phi_1, \dots, \phi_M$  with  $\psi$  ( $q \times 1$ ) that satisfies  $(\phi_1, \dots, \phi_M) = C\psi$ , as above.

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

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

**If  $\lambda_{mi}$  are constrained:** Replace  $\lambda_2, \dots, \lambda_M$  with  $\gamma$  ( $r \times 1$ ) that satisfies  $(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma$  where  $C_\lambda$  is a  $(d(M-1) \times r)$  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,  $\Omega_m$  denotes the error term covariance matrix of the  $m$ th mixture component, and  $\alpha_m$  is the mixing weight parameter. The  $W$  and  $\lambda_{mi}$  are structural parameters replacing the error term covariance

matrices (see Virolainen, 2020). If  $M = 1$ ,  $\alpha_m$  and  $\lambda_{mi}$  are dropped. If parametrization="mean", just replace each  $\phi_{m,0}$  with regimewise mean  $\mu_m$ .  $vec()$  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  $\nu$  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  $\nu$  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 $\mu_m$ , $m=1,\dots,M$ .
constraints	a size $(Mpd^2 \times xq)$ constraint matrix $C$ specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, \dots, \phi_M) = C\psi$ , where $\phi_m = (vec(A_{m,1}), \dots, vec(A_{m,p}))(pd^2 \times 1)$ , $m = 1, \dots, M$ , 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 $C = [I : \dots : I]'$ ( $Mpd^2 \times pd^2$ ) where $I = \text{diag}(p \times d^2)$ . Ignore (or set to NULL) if linear constraints should <b>not</b> 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 <code>list(1,2:3)</code> 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. <b>This constraint is available only for mean parametrized models; that is, when parametrization="mean"</b> .
structural_pars	If NULL a reduced form model is considered. For structural model, should be a list containing the following elements: <ul style="list-style-type: none"> <li>• <math>W</math> - a <math>(d \times d)</math> matrix with its entries imposing constraints on <math>W</math>: 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.</li> <li>• <math>C\_lambda</math> - a <math>(d(M-1) \times r)</math> constraint matrix that satisfies <math>(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma</math> where <math>\gamma</math> is the new <math>(rx1)</math> parameter subject to which the model is estimated (similarly to AR parameter constraints). The entries of <math>C\_lambda</math> must be either <b>positive</b> or <b>zero</b>. Ignore (or set to NULL) if the eigenvalues <math>\lambda_{mi}</math> should not be constrained.</li> </ul>

	See Virolainen (2020) 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 than $1 - \text{stat\_tol}$ 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 $2 + \text{df\_tol}$ .

### 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

### 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	<i>Perform likelihood ratio test for a GMVAR, StMVAR, or G-StMVAR model</i>
---------	---

---

## 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 <b>freely estimated</b> model.
gsmvar2	an object of class 'gsmvar' generated by fitGSMVAR or GSMVAR, containing the <b>constrained</b> 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  $\chi^2$ -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 "htest" containing the following components:

statistic	the value of the likelihood ratio statistics.
parameter	the degrees of freedom of the likelihood ratio statistic.
p.value	the p-value of the test.
alternative	a character string describing the alternative hypothesis.
method	a character string indicating the type of the test (likelihood ratio test).
data.name	a character string giving the names of the supplied models, gsmar1 and gsmar2.
gsmvar1	the supplied argument gsmvar1
gsmvar2	the supplied argument gsmvar2

## References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

## See Also

[Wald\\_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)
```

---

plot.gmvarpred

*plot method for class 'gmvarpred' objects*

---

## Description

plot.gmvarpred is plot method for gsmvarpred objects. EXISTS FOR BACKWARD COMPATIBILITY. THE CLASS 'gmvarpred' IS DEPRECATED FROM THE VERSION 2.0.0 ONWARD: WE USE THE CLASS 'gsmvarpred' NOW.

**Usage**

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

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

**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?
digits	how many digits to print?

**Details**

These methods exist so that objects created with earlier versions of the package can be used normally.

---

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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. 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 <code>fitGSMVAR</code> or <code>GSMVAR</code> .
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 than $1 - \text{stat\_tol}$ 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 $2 + \text{df\_tol}$ .

### 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 positive 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`.

### Methods (by generic)

- `plot`: Plot p-values of the autocorrelation and conditional heteroskedasticity tests.
- `print`: 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

## 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.gmvar

*DEPRECATED! USE THE FUNCTION predict.gsmvar INSTEAD!*  
*Predict method for class 'gmvar' objects*

---

## Description

predict.gsmvar is a predict method for class 'gsmvar' objects. The forecasts of the GMVAR model 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 'gmvar'
predict(
  object,
  ...,
  n_ahead,
  n_simu = 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 'gmvar'
...	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?
n_simu	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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

## See Also

[predict.gsmvar](#)

---

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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. 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.gmvar

*Deprecated S3 methods for the deprecated class 'gmvar'*


---

## Description

Deprecated S3 methods for the deprecated class 'gmvar'. From the gmvarkit version 2.0.0 onwards, class 'gsmvar' is used instead.

## Usage

```

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

## S3 method for class 'gmvar'
summary(object, ..., digits)

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

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

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

```

## Arguments

x	a class 'gmvar' object. THIS CLASS IS DEPRECATED FROM THE VERSION 2.0.0 ONWARDS.
...	See the usage from the documentation of the appropriate class 'gsmvar' S3 method.
digits	number of digits to be printed.
object	object of class 'gmvar'. THIS CLASS IS DEPRECATED FROM THE VERSION 2.0.0 ONWARDS.

**Details**

These methods exist so that models estimated with earlier versions of the package can be used normally.

---

print.gmvarsum	<i>Summary print method from objects of class 'gmvarsum'</i>
----------------	--

---

**Description**

print.gmvarsum is a print method for object 'gmvarsum'. EXISTS FOR BACKWARD COMPATIBILITY. CLASS 'gmvarsum' IS DEPRECATED FROM THE VERSION 2.0.0. ONWARDS. NOW, WE USE THE CLASS 'gsmvarsum'.

**Usage**

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

**Arguments**

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

---

print.gsmvarpred	<i>Print method for class 'gsmvarpred' objects</i>
------------------	--

---

**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_std_errors	<i>Print standard errors of a GMVAR, StMVAR, or G-StMVAR model in the same form as the model estimates are printed</i>
------------------	--

---

### 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

### 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-likelihoods 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  $((M(pd^2 + d + d(d+1)/2 + 1) - 1) \times 1)$  and have form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where:

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

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

- $\psi (qx1)$  satisfies  $(\phi_1, \dots, \phi_M) = C\psi$ . Here  $C$  is  $(Mpd^2 \times qx)$  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,  $\Omega_m$  denotes the error term covariance matrix of the  $m$ :th mixture component, and  $\alpha_m$  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 $\text{abs}(\text{scale} \times \text{estimate})$ .
nrows	how many rows should be in the plot-matrix? The default is $\max(\text{ceiling}(\log_2(\text{length}(\text{which\_pars}) - 1), 1)$ .
ncols	how many columns should be in the plot-matrix? The default is $\text{ceiling}(\text{length}(\text{which\_pars}) / \text{nrows})$ . Note that $\text{nrows} \times \text{ncols}$ should not be smaller than the length of <code>which_pars</code> .
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 than $1 - \text{stat\_tol}$ 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 $2 + \text{df\_tol}$ .

### 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

**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	<i>Calculate multivariate quantile residuals of a GMVAR, StMVAR, or G-StMVAR model</i>
--------------------	--

---

**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  $((n_{obs} - p) \times d)$  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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

**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	<i>Create somewhat random parameter vector of a GMVAR, StMVAR, or G-StMVAR model that is always stationary</i>
-------------	--

---

### 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,
  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	<p><b>For GMVAR and StMVAR models:</b> a positive integer specifying the number of mixture components.</p> <p><b>For G-StMVAR models:</b> a size (2x1) integer vector specifying the number of <i>GMVAR type</i> components M1 in the first element and <i>StMVAR type</i> components M2 in the second element. The total number of mixture components is <math>M=M1+M2</math>.</p>
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 <code>list(1,2:3)</code> 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. <b>This constraint is available only for mean parametrized models; that is, when parametrization="mean".</b>

## structural\_pars

If NULL a reduced form model is considered. For structural model, should be a list containing the following elements:

- $W$  - a  $(dx d)$  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  $(d(M - 1)xr)$  constraint matrix that satisfies  $(\lambda_2, \dots, \lambda_M) = C_\lambda \gamma$  where  $\gamma$  is the new  $(rx1)$  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  $\lambda_{mi}$  should not be constrained.

See Virolainen (2020) 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  $\mu_m$  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  $\mu_m$  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`. This argument is ignored if structural model is considered.

## ar\_scale

a positive real number 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  $(0, \theta)$  uniform distribution in each iteration.

## 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  $\Omega_1$  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  $\lambda_{mi}$  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  $m$ th regime. The expected value of the main **diagonal** elements  $ij$  of the  $m$ th ( $m > 1$ ) error term covariance matrix will be  $W\_scale[i]*(d - n\_i)^{-1}*\sum(\lambda_{m,i}*ind\_fun)$  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  $i$ th row of  $W$  and `ind_fun` is an indicator function that takes the value one iff the  $ij$ th 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  $\gamma$  parameters are drawn from (the  $\gamma$  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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.

- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

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	<b>For GMVAR and StMVAR models:</b> a positive integer specifying the number of mixture components. <b>For G-StMVAR models:</b> a size (2x1) integer vector specifying the number of <i>GMVAR type</i> components M1 in the first element and <i>StMVAR type</i> 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 $\lambda_2, \dots, \lambda_M$ where $\lambda_m = (\lambda_{m1}, \dots, \lambda_{md})$
perm	a vector of length M giving the new order of the covariance matrices (relative to the current order)

### Details

We consider the following decomposition of positive definite covariance matrices:  $\Omega_1 = WW'$ ,  $\Omega_m = W\Lambda_m W'$ ,  $m = 2, \dots, M$  where  $\Lambda_m = \text{diag}(\lambda_{m1}, \dots, \lambda_{md})$  contains the strictly positive 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 exist 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  $c(\text{vec}(\text{new\_W}), \text{new\_lambdas})$  where the lambdas parameters are in the regime-wise order (first regime 2, then 3, etc) and the "new W" and "new lambdas" 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	<i>Reorder columns of the W-matrix and lambda parameters of a structural GMVAR, StMVAR, or G-StMVAR model.</i>
-------------------	--

---

**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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. 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 structural 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 structural 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(gsmvar$model$M),
  ntimes = 1,
  drop = TRUE,
  girf_pars = NULL
)
```

## 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 $(pxd)$ matrix specifying the initial values, where $d$ is the number of time series in the system. The <b>last</b> 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 specified by the argument init_regimes.
init_regimes	a numeric vector of length at most $M$ and elements in $1, \dots, M$ 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 $m$ th 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.
girf_pars	This argument is used internally in the estimation of generalized impulse response functions (see ?GIRF). You should ignore it.

### 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

### 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))

```

---

simulateGMVAR

*DEPRECATED! USE THE FUNCTION simulate.gsmvar INSTEAD!*  
*Simulate from GMVAR process*

---

**Description**

DEPRECATED! USE THE FUNCTION simulate.gsmvar INSTEAD! simulateGMVAR simulates observations from a GMVAR

**Usage**

```
simulateGMVAR(
  gmvar,
  nsimu,
  init_values = NULL,
  ntimes = 1,
  drop = TRUE,
  seed = NULL,
  girf_pars = NULL
)
```

**Arguments**

gmvar	object of class 'gmvar'
nsimu	number of observations to be simulated.
init_values	a size ( $pxd$ ) matrix specifying the initial values, where $d$ is the number of time series in the system. The <b>last</b> 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 specified by the argument <code>init_regimes</code> .
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 <code>ntimes==1</code> , i.e., <code>\$sample</code> and <code>\$mixing_weights</code> will be matrices, and <code>\$component</code> will be vector.
seed	set seed for the random number generator?
girf_pars	This argument is used internally in the estimation of generalized impulse response functions (see ?GIRF). You should ignore it.

**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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

## See Also

[simulate.gsmvar](#)

---

stmvar_to_gstmvar	<i>Estimate a G-StMVAR model based on a StMVAR model that has large degrees of freedom parameters</i>
-------------------	---

---

## 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. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.

## See Also

[fitGSMVAR](#), [GSMVAR](#), [GIRF](#), [reorder\\_W\\_columns](#), [swap\\_W\\_signs](#), [gsmvar\\_to\\_sgsvar](#)

## 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

### 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 elements in  $1, \dots, d$  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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. 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	<i>Calculate the unconditional mean, variance, the first p autocovariances, and the first p autocorrelations of a GMVAR, StMVAR, or G-StMVAR process</i>
----------------	--

---

### 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  $(d \times d \times p + 1)$  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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

### 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	<i>Update the stationarity and positive definiteness numerical tolerances of an existing class 'gsmvar' model.</i>
----------------	--

---

### 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 than $1 - \text{stat\_tol}$ 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. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. 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
```

---

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, h = 6e-06)
```

### Arguments

gsmvar	an object of class 'gsmvar', typically created with <code>fitGSMVAR</code> or <code>GSMVAR</code> .
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.
h	difference used to approximate the derivatives.

### Details

Denoting the true parameter value by  $\theta_0$ , we test the null hypothesis  $A\theta_0 = c$ . Under the null, the test statistic is asymptotically  $\chi^2$ -distributed with  $k$  ( $=\text{nrow}(A)$ ) degrees of freedom. The parameter  $\theta_0$  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 "hctest" containing the following components:

statistic	the value of the Wald statistics.
parameter	the degrees of freedom of the Wald statistic.
p.value	the p-value of the test.
alternative	a character string describing the alternative hypothesis.
method	a character string indicating the type of the test (Wald test).
data.name	a character string giving the names of the supplied model, constraint matrix A, and vector c.
gsmvar	the supplied argument <code>gsmvar</code> .
A	the supplied argument <code>A</code> .
c	the supplied argument <code>c</code> .
h	the supplied argument <code>h</code> .

## References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Virolainen S. 2020. Structural Gaussian mixture vector autoregressive model. Unpublished working paper, available as arXiv:2007.04713.
- Virolainen S. 2021. Gaussian and Student's t mixture vector autoregressive model. Unpublished working paper, available as arXiv:2109.13648.

@keywords internal

## See Also

[LR\\_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**
  - gdpdef, 36
- \* **moment functions**
  - cond\_moments, 12
  - get\_regime\_autocovs, 38
  - get\_regime\_means, 39
  - uncond\_moments, 105
- acf, 19
- add\_data, 3, 58
- alt\_gmvar, 4
- alt\_gsmvar, 5, 6
  
- calc\_gradient, 7, 72
- calc\_hessian (calc\_gradient), 7
- check\_parameters, 9
- cond\_moment\_plot, 17, 19, 28, 74, 78, 88, 109
- cond\_moments, 12, 39, 40, 106
  
- density, 19
- diag\_Omegas, 20
- diagnostic\_plot, 17, 18, 74, 78, 88, 89, 97, 109
  
- fitGMVAR, 21
- fitGSMVAR, 4, 7, 17, 19, 24, 24, 43, 47, 58, 60, 68, 72, 74, 78, 85, 88, 89, 95, 97, 101, 102, 104, 107, 109
  
- GAFit, 29
- gdpdef, 36
- get\_boldA\_eigens, 36
- get\_foc (calc\_gradient), 7
- get\_gradient, 28
- get\_gradient (calc\_gradient), 7
- get\_hessian (calc\_gradient), 7
- get\_omega\_eigens, 37
- get\_regime\_autocovs, 16, 38, 40, 106
- get\_regime\_means, 16, 39, 39, 106
- get\_soc, 88
- get\_soc (calc\_gradient), 7
  
- GFEVD, 28, 40, 47, 81, 97
- GIRF, 28, 43, 44, 58, 60, 78, 81, 88, 95, 97, 101, 104, 107
- GMVAR, 48
- gmvar\_to\_gsmvar, 52
- gmvar\_to\_sgmvar, 53
- gmvarkit, 52
- GSMVAR, 4, 7, 17, 19, 28, 43, 47, 51, 54, 60, 68, 72, 74, 78, 85, 88, 89, 95, 97, 101, 102, 104, 107, 109
- gsmvar\_to\_sgsmlvar, 28, 43, 47, 54, 58, 59, 95, 101, 104, 107
  
- in\_paramspace, 61
- in\_paramspace\_int, 64
- iterate\_more, 4, 7, 28, 67, 102
  
- logLik.gmvar (print.gmvar), 82
- logLik.gsmvar (GSMVAR), 54
- loglikelihood, 69
- LR\_test, 17, 19, 28, 47, 73, 78, 88, 109
  
- optim, 68
  
- plot.gfevd (GFEVD), 40
- plot.girf (GIRF), 44
- plot.gmvar (print.gmvar), 82
- plot.gmvarpred, 74
- plot.gsmvar (GSMVAR), 54
- plot.gsmvarpred, 75
- plot.qrtest, 76
- predict.gmvar, 78
- predict.gsmvar, 19, 28, 47, 78, 80, 80, 89, 97
- print.gfevd (GFEVD), 40
- print.girf (GIRF), 44
- print.gmvar, 82
- print.gmvarpred (plot.gmvarpred), 74
- print.gmvarsum, 83
- print.gsmvar, 85
- print.gsmvar (GSMVAR), 54

`print.gsmvarpred`, 83  
`print.gsmvarsum`, 84  
`print.qrtest` (`plot.qrtest`), 76  
`print_std_errors`, 28, 85  
`profile_logliks`, 8, 17, 19, 28, 47, 68, 74,  
78, 85, 86, 89, 97, 109

`quantile_residual_tests`, 17, 19, 28, 47,  
74, 89, 97, 109  
`quantile_residual_tests` (`plot.qrtest`),  
76  
`quantile_residuals`, 78, 88

`random_ind2`, 90  
`redecompose_Omegas`, 93  
`reorder_W_columns`, 28, 43, 47, 58, 60, 94,  
101, 104, 107  
`residuals.gmvar` (`print.gmvar`), 82  
`residuals.gsmvar` (GSMVAR), 54

`simulate.gsmvar`, 28, 43, 47, 81, 96, 100  
`simulateGMVAR`, 98  
`stmvar_to_gstmvar`, 28, 58, 60, 95, 100, 104,  
107  
`summary.gmvar` (`print.gmvar`), 82  
`summary.gsmvar` (GSMVAR), 54  
`swap_parametrization`, 28, 58, 85, 102  
`swap_W_signs`, 28, 43, 47, 58, 60, 95, 101, 103

`uncond_moments`, 16, 39, 40, 105  
`update_numtols`, 4, 7, 28, 58, 68, 78, 102, 106

`Wald_test`, 17, 19, 28, 47, 74, 78, 88, 108