

# Package ‘gmvarkit’

March 12, 2020

**Title** Estimate Gaussian Mixture Vector Autoregressive Model

**Version** 1.1.3

**Description** Unconstrained and constrained maximum likelihood estimation of Gaussian Mixture Vector Autoregressive (GMVAR) model, quantile residual tests, graphical diagnostics, simulations, and forecasting.  
Leena Kalliovirta, Mika Meitz, Pentti Saikkonen (2016) <doi:10.1016/j.jeconom.2016.02.012>.

**Depends** R (>= 3.4.0)

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.0.2

**Imports** Brodningnag (>= 1.2-5), mvnfast (>= 0.2.5), parallel (>= 3.4.0), stats (>= 3.4.0), pbapply (>= 1.3-4), graphics (>= 3.4.0), grDevices (>= 3.4.0)

**Suggests** testthat, knitr, rmarkdown

**VignetteBuilder** knitr

**NeedsCompilation** no

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

**Maintainer** Savi Virolainen <savi.virolainen@helsinki.fi>

**Repository** CRAN

**Date/Publication** 2020-03-12 17:00:03 UTC

## R topics documented:

add_data . . . . .	3
all_pos_ints . . . . .	4
alt_gmvar . . . . .	5
calc_gradient . . . . .	6
change_parametrization . . . . .	8
change_regime . . . . .	10
check_constraints . . . . .	11

check_data	12
check_gmvar	12
check_null_data	13
check_parameters	13
check_pMd	15
cond_moments	15
diagnostic_plot	17
dlogmultinorm	19
eurusd	19
fitGMVAR	20
format_valuef	24
form_boldA	24
GAsit	25
get_boldA_eigens	28
get_IC	29
get_minval	30
get_omega_eigens	30
get_regime_autocovs	31
get_regime_autocovs_int	32
get_regime_means	33
get_regime_means_int	34
get_test_Omega	36
GMVAR	38
gmvarkit	41
in_paramspace	41
in_paramspace_int	43
is_stationary	44
iterate_more	45
loglikelihood	46
loglikelihood_int	49
n_params	51
pick_allA	52
pick_all_phi0_A	53
pick_alphas	54
pick_Am	55
pick_Ami	56
pick_Omegas	57
pick_phi0	58
pick_regime	59
plot.gmvarpred	60
plot.qrtest	61
predict.gmvar	63
print.gmvarpred	65
print.gmvarsum	65
print_std_errors	66
profile_logliks	68
quantile_residuals	70
quantile_residuals_int	71

random_coefmats . . . . .	73
random_coefmats2 . . . . .	74
random_covmat . . . . .	74
random_ind . . . . .	75
random_ind2 . . . . .	76
reform_constrained_pars . . . . .	77
reform_data . . . . .	79
regime_distance . . . . .	79
simulateGMVAR . . . . .	80
smart_covmat . . . . .	82
smart_ind . . . . .	83
sort_components . . . . .	85
standard_errors . . . . .	86
swap_parametrization . . . . .	88
uncond_moments . . . . .	89
uncond_moments_int . . . . .	90
unvec . . . . .	92
unvech . . . . .	93
vec . . . . .	93
vech . . . . .	94

## **Index** **95**

---

add_data	<i>Add data to an object of class 'gmvar' defining a GMVAR model</i>
----------	--

---

### **Description**

add\_data adds or updates data to object of class 'gmvar' that defines a GMVAR model. Also calculates mixing weights and quantile residuals accordingly.

### **Usage**

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

### **Arguments**

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

**Value**

Returns an object of class 'gmvar' defining the specified GMVAR 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.

**See Also**

[fitGMVAR](#), [GMVAR](#), [iterate\\_more](#)

**Examples**

```
# These examples use the data 'eurusd' which comes with the
# package, but in a scaled form.
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

# GMVAR(1,2), d=2 model:
params122 <- c(0.623, -0.129, 0.959, 0.089, -0.006, 1.006, 1.746,
  0.804, 5.804, 3.245, 7.913, 0.952, -0.037, -0.019, 0.943, 6.926,
  3.982, 12.135, 0.789)
mod122 <- GMVAR(p=1, M=2, d=2, params=params122)
mod122

mod122_2 <- add_data(data, mod122)
mod122_2

# GMVAR(2,2), d=2 model with AR-parameters restricted to be
# the same for both regimes:
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
params222c <- c(1.031, 2.356, 1.786, 3.000, 1.250, 0.060, 0.036,
  1.335, -0.290, -0.083, -0.047, -0.356, 0.934, -0.152, 5.201, 5.883,
  3.560, 9.799, 0.368)
mod222c <- GMVAR(p=2, M=2, d=2, params=params222c, constraints=C_mat)
mod222c

mod222c_2 <- add_data(data, mod222c)
mod222c_2
```

---

all\_pos\_ints

*Check whether all arguments are positive integers*

---

**Description**

all\_pos\_ints checks whether all the elements in a vector are positive integers.

**Usage**

```
all_pos_ints(x)
```

**Arguments**

x a vector containing the elements to be tested.

**Value**

Returns TRUE or FALSE accordingly.

---

alt_gmvar	<i>Construct a GMVAR model based on results from an arbitrary estimation round of fitGMVAR.</i>
-----------	---

---

**Description**

alt\_gmvar constructs a GMVAR model based on results from an arbitrary estimation round of fitGMVAR.

**Usage**

```
alt_gmvar(
  gmvar,
  which_round = 1,
  calc_cond_moments = TRUE,
  calc_std_errors = TRUE
)
```

**Arguments**

gmvar an object of class 'gmvar' created with fitGMVAR or GMVAR.

which\_round based on which estimation round should the model be constructed? An integer value in 1,...,ncalls.

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 GMVAR that picks the correct estimates from an object returned by fitGMVAR.

**Value**

Returns an object of class 'gmvar' defining the specified GMVAR model. Can be used to work with other functions provided in gmvarKit.

Remark 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*.

**See Also**

[fitGMVAR](#), [GMVAR](#), [iterate\\_more](#)

**Examples**

```
# These are long running examples and use parallel computing
data(eurusd, package="gmvarKit")
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

fit12 <- fitGMVAR(data, 1, 2, ncalls=2, seeds=7:8)
fit12
fit12_2 <- alt_gmvar(fit12, which_round=1)
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 'gmvar' 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, ...)
```

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

```
get_gradient(gmvar, h = 6e-06)
```

```
get_hessian(gmvar, h = 6e-06)
```

```
get_soc(gmvar, h = 6e-06)
```

```
get_foc(gmvar, h = 6e-06)
```

**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.
...	other arguments passed to fn
gmvar	an object of class 'gmvar' created with fitGMVAR or GMVAR.

**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)

# These examples below use the data 'eurusd' which comes
# with the package, but in a scaled form.
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

# GMVAR(1,2), d=2 model:
params122 <- c(0.623, -0.129, 0.959, 0.089, -0.006, 1.006, 1.746,
  0.804, 5.804, 3.245, 7.913, 0.952, -0.037, -0.019, 0.943, 6.926,
  3.982, 12.135, 0.789)
mod122 <- GMVAR(data, p=1, M=2, params=params122)
get_gradient(mod122)
get_hessian(mod122)
get_soc(mod122)

```

---

change\_parametrization

*Change parametrization of a parameter vector*

---

## Description

change\_parametrization changes the parametrization of the given parameter vector to change\_to.

## Usage

```

change_parametrization(
  p,
  M,
  d,
  params,
  constraints = NULL,
  change_to = c("intercept", "mean")
)

```

## Arguments

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

**M** a positive integer specifying the number of mixture components.

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

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

**For unconstrained models:** Should be size  $((M(pd^2 + d + d(d + 1)/2 + 1) - 1)x1)$  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. 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. The notation is in line with the cited article by *Kalliovirta, Meitz and Saikkonen (2016)* introducing the GMVAR model.

constraints	a size $(Mpd^2 \times qx)$ 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 constrained 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.
change_to	either "intercept" or "mean" specifying to which parametrization it should be switched to. If set to "intercept", it's assumed that params is mean-parametrized, and if set to "mean" it's assumed that params is intercept-parametrized.

### Value

Returns parameter vector described in params, but with parametrization changed from intercept to mean (when change\_to=="mean") or from mean to intercept (when change\_to=="intercept").

### Warning

No argument checks!

### References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.

---

change_regime	Change regime parameters $v_m = (\phi_m, 0, \phi_m, \sigma_m)$ of the given parameter vector
---------------	--

---

### Description

change\_regime changes the regime parameters (excluding mixing weights parameter) of the pointed regime to the new given parameters.

### Usage

```
change_regime(p, M, d, params, m, regime_pars)
```

### Arguments

**p** a positive integer specifying the autoregressive order of the model.  
**M** a positive integer specifying the number of mixture components.  
**d** number of time series in the system, i.e. the dimension.  
**params** a real valued vector specifying the parameter values.

**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:

- $\psi (q \times 1)$  satisfies  $(\phi_1, \dots, \phi_M) = C\psi$ . Here  $C$  is  $(Mpd^2 \times q)$  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. 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. The notation is in line with the cited article by *Kalliovirta, Meitz and Saikkonen (2016)* introducing the GMVAR model.

**m** which component?  
**regime\_pars** a size  $((pd^2 + d + d(d+1)/2) \times 1)$  vector  $v_m = (\phi_{m,0}, \phi_m, \sigma_m)$ .

### Value

Returns parameter vector with  $m$ :th regime changed to regime\_pars.

**Warning**

No argument checks!

**References**

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

---

check_constraints	<i>Check the constraint matrix has the correct form</i>
-------------------	---

---

**Description**

check\_constraints checks that the constraints are correctly set.

**Usage**

```
check_constraints(p, M, d, constraints = NULL)
```

**Arguments**

p	a positive integer specifying the autoregressive order of the model.
M	a positive integer specifying the number of mixture components.
d	the number of time series in the system.
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 constrained 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.

**Details**

If `is.null(constraints)`, then this function doesn't do anything.

**Value**

Checks the constraint matrix **C** and throws an error if something is wrong.

---

check_data	<i>Check the data is in the correct form</i>
------------	--

---

**Description**

check\_data checks the data.

**Usage**

```
check_data(data, p)
```

**Arguments**

data	a matrix or class 'ts' object with $d > 1$ columns. Each column is taken to represent a single time series. NA values are not supported.
p	a positive integer specifying the autoregressive order of the model.

**Value**

Checks the data and tries to correct it. Throws an error if something is wrong and returns the corrected data otherwise.

---

check_gmvar	<i>Checks whether the given object has class attribute "gmvar"</i>
-------------	--

---

**Description**

check\_gmvar checks that the object has class attribute "gmvar".

**Usage**

```
check_gmvar(object)
```

**Arguments**

object	S3 object to be tested
--------	------------------------

**Value**

Throws an error if the object doesn't have the class attribute "gmvar".

---

check_null_data	<i>Checks whether the given object contains data</i>
-----------------	--

---

**Description**

check\_null\_data checks that the gmvar object has data.

**Usage**

```
check_null_data(gmvar)
```

**Arguments**

gmvar            an object of class 'gmvar' created with fitGMVAR or GMVAR.

**Value**

Throws an error if is.null(gmvar\$data).

---

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, constraints = NULL)
```

**Arguments**

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

M                a positive integer specifying the number of mixture components.

d                the number of time series in the system.

params           a real valued vector specifying the parameter values.

**For unconstrained models:** Should be size  $((M(pd^2 + d + d(d + 1)/2 + 1) - 1)x1)$  and have form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where:

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

**For constrained models:** Should be size  $((M(d + d(d + 1)/2 + 1) + q - 1)x1)$  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  $(Mpd^2xq)$  constraint matrix.

Above,  $\phi_{m,0}$  is the intercept parameter,  $A_{m,i}$  denotes the  $i$ :th coefficient matrix of the  $m$ :th mixture component,  $\Omega_m$  denotes the error term covariance matrix of the  $m$ :th mixture component, and  $\alpha_m$  is the mixing weight parameter. 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. The notation is in line with the cited article by Kalliovirta, Meitz and Saikkonen (2016) introducing the GMVAR model.

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 constrained 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.

### 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.

### Examples

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

# GMVAR(1,1), d=2 model:
params112 <- 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:
params222 <- 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=params222)

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

```

5.88, 3.56, 9.80, 1.37)
check_parameters(p=2, M=2, d=2, params=params222c, constraints=C_mat)

## End(Not run)

```

---

check_pMd	<i>Check that p, M, and d are correctly set</i>
-----------	---

---

### Description

check\_pMd checks the arguments p, M, and d.

### Usage

```
check_pMd(p, M, d)
```

### Arguments

p	a positive integer specifying the autoregressive order of the model.
M	a positive integer specifying the number of mixture components.
d	the number of time series in the system.

### Value

Throws an error if something is wrong.

---

cond_moments	<i>Compute conditional moments of a GMVAR model</i>
--------------	---

---

### Description

loglikelihood compute conditional regimewise means, conditional means, and conditional covariance matrices of a GMVAR model.

### Usage

```

cond_moments(
  data,
  p,
  M,
  params,
  parametrization = c("intercept", "mean"),
  constraints = NULL,
  to_return = c("regime_cmeans", "total_cmeans", "total_ccovs")
)

```

**Arguments**

data	a matrix or class 'ts' object with $d > 1$ columns. Each column is taken to represent a single time series. NA values are not supported.
p	a positive integer specifying the autoregressive order of the model.
M	a positive integer specifying the number of mixture components.
params	a real valued vector specifying the parameter values.

**For unconstrained models:** Should be size  $((M(pd^2 + d + d(d+1)/2 + 1) - 1)x1)$  and have form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where:

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

**For constrained models:** Should be size  $((M(d + d(d+1)/2 + 1) + q - 1)x1)$  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  $(Mpd^2 x q)$  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. 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. The notation is in line with the cited article by *Kalliovirta, Meitz and Saikkonen (2016)* introducing the GMVAR model.

parametrization	"mean" or "intercept" determining whether the model is parametrized with regime means $\mu_m$ or intercept parameters $\phi_{m,0}$ , $m=1, \dots, M$ .
constraints	a size $(Mpd^2 x q)$ constraint matrix $C$ specifying general linear constraints to the autoregressive parameters. We consider constraints of form $(\phi_1, \dots, \phi_M) = C\psi$ , where $\phi_m = (vec(A_{m,1}), \dots, vec(A_{m,p}))(pd^2 x 1)$ , $m = 1, \dots, M$ contains the coefficient matrices and $\psi$ ( $qx1$ ) contains the constrained parameters. For example, to restrict the AR-parameters to be the same for all regimes, set $C = [I : \dots : I]'$ ( $Mpd^2 x pd^2$ ) where $I = \text{diag}(p \times d^2)$ . Ignore (or set to NULL) if linear constraints should <b>not</b> be employed.
to_return	should the regimewise conditional means, total conditional means, or total conditional covariance matrices be returned?

**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=="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).

**References**

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.

**See Also**

Other moment functions: [get\\_regime\\_autocovs\(\)](#), [get\\_regime\\_means\(\)](#), [uncond\\_moments\(\)](#)

**Examples**

```
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
params222 <- c(-11.904, 154.684, 1.314, 0.145, 0.094, 1.292, -0.389,
-0.070, -0.109, -0.281, 0.920, -0.025, 4.839, 11.633, 124.983, 1.248,
0.077, -0.040, 1.266, -0.272, -0.074, 0.034, -0.313, 5.855, 3.570,
9.838, 0.740)
cond_moments(data=data, p=2, M=2, params=params222, parametrization="mean",
to_return="regime_cmeans")
cond_moments(data=data, p=2, M=2, params=params222, parametrization="mean",
to_return="total_cmeans")
cond_moments(data=data, p=2, M=2, params=params222, parametrization="mean",
to_return="total_ccovs")
```

---

diagnostic\_plot

*Quantile residual diagnostic plot for a GMVAR model*


---

**Description**

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

**Usage**

```
diagnostic_plot(gmvar, type = c("series", "ac", "ch", "norm"), maxlag = 10)
```

**Arguments**

gmvar	an object of class 'gmvar' created with fitGMVAR or GMVAR.
type	which type of diagnostic plot should be plotted? <ul style="list-style-type: none"> <li>• "series" the quantile residual time series.</li> <li>• "ac" the quantile residual autocorrelation and cross-correlation functions.</li> <li>• "ch" the squared quantile residual autocorrelation and cross-correlation functions.</li> <li>• "norm" the quantile residual histogram with theoretical standard normal density (dashed line) and standard normal QQ-plots.</li> </ul>
maxlag	the maximum lag considered in types "ac" and "ch".

**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*.

**See Also**

[profile\\_logliks](#), [fitGMVAR](#), [GMVAR](#), [quantile\\_residual\\_tests](#), [acf](#), [density](#), [predict.gmvar](#)

**Examples**

```
# These examples use the data 'eurusd' which comes with the
# package, but in a scaled form.
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

# GMVAR(1,2), d=2 model:
params122 <- c(0.623, -0.129, 0.959, 0.089, -0.006, 1.006, 1.746,
  0.804, 5.804, 3.245, 7.913, 0.952, -0.037, -0.019, 0.943, 6.926,
  3.982, 12.135, 0.789)
mod122 <- GMVAR(data, p=1, M=2, params=params122)
diagnostic_plot(mod122, type="series")
diagnostic_plot(mod122, type="ac")

# GMVAR(2,2), d=2 model:
params222 <- c(1.386, -0.765, 1.314, 0.145, 0.094, 1.292, -0.389,
  -0.070, -0.109, -0.281, 0.920, -0.025, 4.839, 1.005, 5.928, 1.248,
  0.077, -0.040, 1.266, -0.272, -0.074, 0.034, -0.313, 5.855, 3.570,
  9.838, 0.740)
mod222 <- GMVAR(data, p=2, M=2, params=params222)
diagnostic_plot(mod222, type="ch")
```

```

diagnostic_plot(mod222, type="norm")

# GMVAR(2,2), d=2 model with AR-parameters restricted to be
# the same for both regimes:
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
params222c <- c(1.031, 2.356, 1.786, 3.000, 1.250, 0.060, 0.036,
  1.335, -0.290, -0.083, -0.047, -0.356, 0.934, -0.152, 5.201, 5.883,
  3.560, 9.799, 0.368)
mod222c <- GMVAR(data, p=2, M=2, params=params222c, constraints=C_mat)
diagnostic_plot(mod222c)
diagnostic_plot(mod222c, type="ac", maxlag=12)

```

---

dlogmultinorm	<i>Calculate logarithms of multiple multivariate normal densities with varying mean and constant covariance matrix</i>
---------------	--

---

### Description

dlogmultinorm calculates logarithms of multiple multivariate normal densities with varying mean and constant covariance matrix.

### Usage

```
dlogmultinorm(y, mu, Omega)
```

### Arguments

y	dimension $(Txk)$ matrix where each row is a k-dimensional random vector
mu	dimension $(Txk)$ matrix where each row is the mean of the k-dimensional random vector in corresponding row of y.
Omega	the $(kxk)$ covariance matrix Omega.

### Value

Returns a size  $(Tx1)$  vector containing the multinormal densities in logarithm.

---

eurusd	<i>Euro area and U.S. long-term government bond yields and Euro-U.S. dollar exchange rate.</i>
--------	--

---

### Description

A dataset containing time series of the difference between the monthly Euro area and U.S. long-term government bond yields and monthly average Euro - U.S. dollar exchange rate. The data covers the time period January 1989 - December 2009 with monthly frequency. This is the same data (in non-scaled form) that is used by Kalliovirta et. al. (2016).

**Usage**

eurusd

**Format**

A numeric matrix of class 'ts' with 252 rows and 2 columns with one time series in each column:

**First column:** The difference between the monthly Euro area and U.S. long-term government bond yields (10 year maturity,  $i_{\text{euro}} - i_{\text{us}}$ ), from January 1989 to December 2009. calculated by the ECB and the Federal Reserve Board; prior to 2001, the Euro area data refer to the "EU11" countries, and afterwards with changing composition eventually to the "EU17" by the end of the data period.

**Second column:** Monthly average Euro - U.S. dollar exchange rate, from January 1989 to December 2009. Based on the ECU - USD exchange rate prior to 1999.

**Source**

OECD Statistics

**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*.

---

fitGMVAR

*Two-phase maximum likelihood estimation of a GMVAR model*

---

**Description**

fitGMVAR estimates a GMVAR 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,
  ncalls = round(10 + 9 * log(M)),
  ncores = min(2, ncalls, parallel::detectCores()),
```

```

    maxit = 300,
    seeds = NULL,
    print_res = TRUE,
    ...
)

```

### Arguments

data	a matrix or class 'ts' object with $d > 1$ columns. Each column is taken to represent a single time series. NA values are not supported.
p	a positive integer specifying the autoregressive order of the model.
M	a positive integer specifying the number of mixture components.
conditional	a logical argument specifying whether the conditional or exact log-likelihood function should be used.
parametrization	"mean" or "intercept" determining whether the model is parametrized with regime means $\mu_m$ or intercept parameters $\phi_{m,0}$ , $m=1,\dots,M$ .
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 constrained 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.
ncalls	the number of estimation rounds that should be performed.
ncores	the number CPU cores to be used in parallel computing.
maxit	the maximum number of iterations in the variable metric algorithm.
seeds	a length ncalls vector containing the random number generator seed for each call to the genetic algorithm, or NULL for not initializing the seed. Exists for creating reproducible results.
print_res	should summaries of estimation results be printed?
...	additional settings passed to the function GAFit employing the genetic algorithm.

### Details

Because of complexity and 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 point instead. Therefore a 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.

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.

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`.

## Value

Returns an object of class 'gmvar' defining the estimated GMVAR 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 `gmvar$params` (and corresponding approximate standard errors at `gmvar$std_errors`) and it is...

**Unconstrained models:** a size  $((M(pd^2 + d + d(d+1)/2 + 1) - 1)x1)$  vector that has form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where:

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

**Constrained models:** a size  $((M(d + d(d+1)/2 + 1) + q - 1)x1)$  vector that has 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  $(Mpd^2 x q)$  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. If `parameterization=="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. The notation is in line with the cited article by *Kalliovirta, Meitz and Saikkonen (2016)* which introduces the GMVAR model.

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 'gmvar': `logLik`, `residuals`, `print`, `summary`, `predict` 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.

- Kalliovirta L. and Saikkonen P. 2010. Reliable Residuals for Multivariate Nonlinear Time Series Models. *Unpublished Revision of HECER Discussion Paper No. 247*.
- 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.

### See Also

[GMVAR](#), [iterate\\_more](#), [predict.gmvar](#), [profile\\_logliks](#), [simulateGMVAR](#), [quantile\\_residual\\_tests](#), [print\\_std\\_errors](#), [swap\\_parametrization](#), [get\\_gradient](#)

### Examples

```
## These are long running examples that use parallel computing!

# These examples use the data 'eurusd' which comes with the
# package, but in a scaled form (similar to Kalliovirta et al. 2016).
data(eurusd, package="gmvarkit")
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(dat) <- colnames(eurusd)

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

# GMVAR(2,2) model with mean parametrization
fit22 <- fitGMVAR(data, p=2, M=2, parametrization="mean",
                  ncalls=16, seeds=1:16)
fit22

# 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 <- fitGMVAR(data, p=2, M=2, constraints=C_mat)
fit22c

# 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. Estimation
# with only 10 estimation rounds.
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 <- fitGMVAR(data, p=2, M=2, constraints=C_mat2)
fit22c2
```

---

format_valuef	<i>Function factory for value formatting</i>
---------------	--

---

**Description**

format\_valuef is a function factory for formatting values with certain number of digits.

**Usage**

```
format_valuef(digits)
```

**Arguments**

digits	number of digits to use
--------	-------------------------

**Value**

Returns a function that takes an atomic vector as argument and returns it formatted to character with digits decimals.

---

form_boldA	<i>Form the <math>((dp)x(dp))</math> "bold A" matrices related to the VAR processes</i>
------------	---

---

**Description**

form\_boldA creates the "bold A" coefficient matrices related to VAR processes.

**Usage**

```
form_boldA(p, M, d, all_A)
```

**Arguments**

p	a positive integer specifying the autoregressive order of the model.
M	a positive integer specifying the number of mixture components.
d	the number of time series in the system.
all_A	4D array containing all coefficient matrices $A_{m,i}$ , obtained from pick_allA.

**Value**

Returns 3D array containing the  $((dp)x(dp))$  "bold A" matrices related to each component VAR-process. The matrix  $A_m$  can be obtained by choosing  $[, , m]$ .

**Warning**

No argument checks!



## References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. *New Introduction to Multiple Time Series Analysis*, Springer.

---

GAfit

*Genetic algorithm for preliminary estimation of a GMVAR model*


---

## Description

GAfit estimates the specified GMVAR model using a genetic algorithm. It's designed to find starting values for gradient based methods.

## Usage

```
GAfit(
  data,
  p,
  M,
  conditional = TRUE,
  parametrization = c("intercept", "mean"),
  constraints = NULL,
  ngen = 200,
  popsize,
  smart_mu = min(100, ceiling(0.5 * ngen)),
  initpop = NULL,
  mu_scale,
  mu_scale2,
  omega_scale,
  ar_scale = 1,
  regime_force_scale = 1,
  red_criteria = c(0.05, 0.01),
  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 single time series. NA values are not supported.
p	a positive integer specifying the autoregressive order of the model.
M	a positive integer specifying the number of mixture components.
conditional	a logical argument specifying whether the conditional or exact log-likelihood function should be used.

parametrization	"mean" or "intercept" determining whether the model is parametrized with regime means $\mu_m$ or intercept parameters $\phi_{m,0}$ , $m=1,\dots,M$ .
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 constrained 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 <b>not</b> be employed.
ngen	a positive integer specifying the number of generations to be ran through in the genetic algorithm.
popsize	a positive even integer specifying the population size in the genetic algorithm. Default is $10*n\_params$ .
smart_mu	a positive integer specifying the generation after which the random mutations in the genetic algorithm are "smart". This means that mutating individuals will mostly mutate fairly close (or partially close) to the best fitting individual (which has the least regimes with time varying mixing weights practically at zero) so far.
initpop	a list of parameter vectors from which the initial population of the genetic algorithm will be generated from. The parameter vectors should be...  <b>For unconstrained models:</b> Should be size $((M(pd^2 + d + d(d+1)/2 + 1) - 1)x1)$ and have form $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where: <ul style="list-style-type: none"> <li>• <math>v_m = (\phi_{m,0}, \phi_m, \sigma_m)</math></li> <li>• <math>\phi_m = (\text{vec}(A_{m,1}), \dots, \text{vec}(A_{m,p}))</math></li> <li>• and <math>\sigma_m = \text{vech}(\Omega_m)</math>, <math>m=1,\dots,M</math>.</li> </ul> <b>For constrained models:</b> Should be size $((M(d + d(d+1)/2 + 1) + q - 1)x1)$ and have form $\theta = (\phi_{1,0}, \dots, \phi_{M,0}, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1})$ , where: <ul style="list-style-type: none"> <li>• <math>\psi (qx1)</math> satisfies <math>(\phi_1, \dots, \phi_M) = C\psi</math>. Here <math>C</math> is <math>(Mpd^2xq)</math> constraint matrix.</li> </ul> <p>Above, <math>\phi_{m,0}</math> is the intercept parameter, <math>A_{m,i}</math> denotes the <math>i</math>:th coefficient matrix of the <math>m</math>:th mixture component, <math>\Omega_m</math> denotes the error term covariance matrix of the <math>m</math>:th mixture component, and <math>\alpha_m</math> is the mixing weight parameter. If parametrization=="mean", just replace each <math>\phi_{m,0}</math> with the regimewise mean <math>\mu_m</math>. <math>\text{vec}()</math> is vectorization operator that stacks columns of a given matrix into a vector. <math>\text{vech}()</math> stacks columns of a given matrix from the principal diagonal downwards (including elements on the diagonal) into a vector. The notation is in line with the cited article by <i>Kalliovirta, Meitz and Saikkonen (2016)</i> which introduces the GMVAR model.</p>
mu_scale	a size $(dx1)$ vector defining <b>means</b> of the normal distributions from which each mean parameter $\mu_m$ is drawn from in random mutations. Default is <code>colMeans(data)</code> . 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 ( $d \times 1$ ) strictly positive vector defining <b>standard deviations</b> of the normal distributions from which each mean parameter $\mu_m$ is drawn from in random mutations. Default is $2 * \text{sd}(\text{data}[, i]), i=1, \dots, d$ .
omega_scale	a size ( $d \times 1$ ) 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 $\text{diag}(\text{omega\_scale})$ . Standard deviations of the diagonal elements are $\sqrt{2/d} * \text{omega\_scale}[i]$ and for non-diagonal elements they are $\sqrt{1/d * \text{omega\_scale}[i] * \text{omega\_scale}[j]}$ . Note that for $d > 4$ this scale may need to be chosen carefully. Default in GAfit is $\text{var}(\text{stats} : : \text{ar}(\text{data}[, i], \text{order} . \text{max} = 10) \$ \text{resid}$
ar_scale	a positive real number adjusting how large AR parameter values are typically generated in some random mutations. See the function <code>random_coefmats2</code> for details. This is ignored when estimating constrained models.
regime_force_scale	a non-negative real number specifying how much should natural selection favour individuals with less regimes that have almost all mixing weights (practically) at zero. Set to zero for no favouring or large number for heavy favouring. Without any favouring the genetic algorithm gets more often stuck in an area of the parameter space where some regimes are wasted, but with too much favouring the best genes might never mix into the population and the algorithm might converge poorly. Default is 1 and it gives $2x$ larger surviving probability weights for individuals with no wasted regimes compared to individuals with one wasted regime. Number 2 would give $3x$ larger probability weights etc.
red_criteria	a length 2 numeric vector specifying the criteria that is used to determine whether a regime is redundant (or "wasted") or not. Any regime $m$ which satisfies $\text{sum}(\text{mixingWeights}[, m] > \text{red\_criteria}[1]) < \text{red\_criteria}[2] * n\_obs$ will be considered "redundant". One should be careful when adjusting this argument (set $c(0, 0)$ to fully disable the 'redundant regime' features from the algorithm).
to_return	should the genetic algorithm return the best fitting individual which has "positive enough" mixing weights for as many regimes as possible ("alt_ind") or the individual which has the highest log-likelihood in general ("best_ind") but might have more wasted regimes?
minval	a real number defining the minimum value of the log-likelihood function that will be considered. Values smaller than this will be treated as they were minval and the corresponding individuals will never survive. The default is $-(10^{(\text{ceiling}(\log_{10}(n\_obs)) + d) - 1})$ .
seed	a single value, interpreted as an integer, or NULL, that sets seed for the random number generator in the beginning of the function call. If calling GAfit from <code>fitGMVAR</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 `ini_tpop`.

### 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.

@export

---

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(gmvar)
```

### Arguments

gmvar            an object of class 'gmvar' created with fitGMVAR or GMVAR.

### Value

Returns a list with  $M$  elements - one for each regime. Each element contains the absolute values (or modulus) of the eigenvalues of the "bold A" matrix containing the AR coefficients.

## References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. *New Introduction to Multiple Time Series Analysis*, Springer.

## Examples

```
params222 <- c(-11.904, 154.684, 1.314, 0.145, 0.094, 1.292, -0.389,
-0.070, -0.109, -0.281, 0.920, -0.025, 4.839, 11.633, 124.983, 1.248,
0.077, -0.040, 1.266, -0.272, -0.074, 0.034, -0.313, 5.855, 3.570,
9.838, 0.740)
mod222 <- GMVAR(d=2, p=2, M=2, params=params222, parametrization="mean")
get_boldA_eigens(mod222)
```

---

get_IC	<i>Calculate AIC, HQIC, and BIC</i>
--------	-------------------------------------

---

## Description

get\_IC calculates the information criteria values AIC, HQIC, and BIC.

## Usage

```
get_IC(loglik, npars, obs)
```

## Arguments

loglik	log-likelihood value
npars	number of (freely estimated) parameters in the model
obs	numbers of observations with starting values excluded for conditional models.

## Details

Note that for conditional models with different autoregressive order p the information criteria values are **NOT** comparable.

## Value

Returns a data frame containing the information criteria values.

---

get_minval	<i>Returns the default smallest allowed log-likelihood for given data.</i>
------------	--

---

**Description**

get\_minval returns the default smallest allowed log-likelihood for given data.

**Usage**

```
get_minval(data)
```

**Arguments**

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

**Details**

This function exists to avoid duplication inside the package.

**Value**

Returns  $-(10^{(\text{ceiling}(\log_{10}(\text{nrow}(\text{data})) + \text{ncol}(\text{data})))} - 1)$

**See Also**

[fitGMVAR](#), [GAFit](#)

---

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(gmvar)
```

**Arguments**

gmvar	an object of class 'gmvar' created with fitGMVAR or GMVAR.
-------	--

**Value**

Returns a list with  $M$  elements - one for each regime. Each element contains the eigenvalues of the "Omega" error term covariance matrix.

**References**

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. *New Introduction to Multiple Time Series Analysis*, Springer.

**Examples**

```
params222 <- c(-11.904, 154.684, 1.314, 0.145, 0.094, 1.292, -0.389,
-0.070, -0.109, -0.281, 0.920, -0.025, 4.839, 11.633, 124.983, 1.248,
0.077, -0.040, 1.266, -0.272, -0.074, 0.034, -0.313, 5.855, 3.570,
9.838, 0.740)
mod222 <- GMVAR(d=2, p=2, M=2, params=params222, parametrization="mean")
get_omega_eigens(mod222)
```

---

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 model.

**Usage**

```
get_regime_autocovs(gmvar)
```

**Arguments**

gmvar                    an object of class 'gmvar' created with fitGMVAR or GMVAR.

**Value**

Returns an  $(dx \times xp + 1 \times M)$  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.
- Lutkepohl H. 2005. *New Introduction to Multiple Time Series Analysis*, Springer.

**See Also**

Other moment functions: [cond\\_moments\(\)](#), [get\\_regime\\_means\(\)](#), [uncond\\_moments\(\)](#)

**Examples**

```
# GMVAR(1,2), d=2 model:
params122 <- c(0.623, -0.129, 0.959, 0.089, -0.006, 1.006, 1.746,
  0.804, 5.804, 3.245, 7.913, 0.952, -0.037, -0.019, 0.943, 6.926,
  3.982, 12.135, 0.789)
mod122 <- GMVAR(p=1, M=2, d=2, params=params122)
get_regime_autocovs(mod122)

# GMVAR(2,2), d=2 model with AR-parameters restricted to be
# the same for both regimes:
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
params222c <- c(1.031, 2.356, 1.786, 3.000, 1.250, 0.060, 0.036,
  1.335, -0.290, -0.083, -0.047, -0.356, 0.934, -0.152, 5.201, 5.883,
  3.560, 9.799, 0.368)
mod222c <- GMVAR(p=2, M=2, d=2, params=params222c, constraints=C_mat)
get_regime_autocovs(mod222c)
```

---

```
get_regime_autocovs_int
```

*Calculate regimewise autocovariance matrices*

---

**Description**

`get_regime_autocovs_int` calculates the regimewise autocovariance matrices  $\Gamma_m(j)$   $j = 0, 1, \dots, p$  for the given GMVAR model.

**Usage**

```
get_regime_autocovs_int(p, M, d, params, constraints = NULL)
```

**Arguments**

`p` a positive integer specifying the autoregressive order of the model.  
`M` a positive integer specifying the number of mixture components.  
`d` the number of time series in the system.  
`params` a real valued vector specifying the parameter values.

**For unconstrained models:** Should be size  $((M(pd^2 + d + d(d + 1)/2 + 1) - 1) \times 1)$  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  $((M(d + d(d + 1)/2 + 1) + q - 1)x1)$  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  $(Mpd^2xq)$  constraint matrix.

Above,  $\phi_{m,0}$  is the intercept parameter,  $A_{m,i}$  denotes the  $i$ :th coefficient matrix of the  $m$ :th mixture component,  $\Omega_m$  denotes the error term covariance matrix of the  $m$ :th mixture component, and  $\alpha_m$  is the mixing weight parameter. 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. The notation is in line with the cited article by Kalliovirta, Meitz and Saikkonen (2016) introducing the GMVAR model.

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 constrained 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 **not** be employed.

## Value

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

## References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.

---

get\_regime\_means      Calculate regime means  $\mu_m$

---

## Description

get\_regime\_means calculates regime means  $\mu_m = (I - \sum A_{m,i})^{-1}$  for the given GMVAR model.

## Usage

```
get_regime_means(gmvar)
```

## Arguments

gmvar      an object of class 'gmvar' created with fitGMVAR or GMVAR.

**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.
- Lutkepohl H. 2005. *New Introduction to Multiple Time Series Analysis*, Springer.

**See Also**

[uncond\\_moments](#), [get\\_regime\\_autocovs](#), [cond\\_moments](#)

Other moment functions: [cond\\_moments\(\)](#), [get\\_regime\\_autocovs\(\)](#), [uncond\\_moments\(\)](#)

**Examples**

```
# These examples use the data 'eurusd' which comes with the
# package, but in a scaled form.
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

# GMVAR(1,2), d=2 model:
params122 <- c(0.623, -0.129, 0.959, 0.089, -0.006, 1.006, 1.746,
  0.804, 5.804, 3.245, 7.913, 0.952, -0.037, -0.019, 0.943, 6.926,
  3.982, 12.135, 0.789)
mod122 <- GMVAR(data, p=1, M=2, params=params122)
mod122
get_regime_means(mod122)

# GMVAR(2,2), d=2 model with mean-parametrization:
params222 <- c(-11.904, 154.684, 1.314, 0.145, 0.094, 1.292, -0.389,
  -0.070, -0.109, -0.281, 0.920, -0.025, 4.839, 11.633, 124.983, 1.248,
  0.077, -0.040, 1.266, -0.272, -0.074, 0.034, -0.313, 5.855, 3.570,
  9.838, 0.740)
mod222 <- GMVAR(data, p=2, M=2, params=params222, parametrization="mean")
mod222
get_regime_means(mod222)
```

---

get\_regime\_means\_int    *Calculate regime means  $\mu_m$*

---

**Description**

get\_regime\_means calculates regime means  $\mu_m = (I - \sum A)^{-1}$  from the given parameter vector.

**Usage**

```

get_regime_means_int(
  p,
  M,
  d,
  params,
  parametrization = c("intercept", "mean"),
  constraints = NULL
)

```

**Arguments**

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

**M** a positive integer specifying the number of mixture components.

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

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

**For unconstrained models:** Should be size  $((M(pd^2 + d + d(d + 1)/2 + 1) - 1) \times 1)$  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. 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. The notation is in line with the cited article by *Kalliovirta, Meitz and Saikkonen (2016)* introducing the GMVAR model.

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

**constraints** a size  $(Mpd^2 \times qx)$  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 constrained 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.

**Value**

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

**Warning**

No argument checks!

**References**

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. *New Introduction to Multiple Time Series Analysis*, Springer.

---

get\_test\_Omega

*Compute covariance matrix Omega used in quantile residual tests*

---

**Description**

get\_test\_Omega computes the covariance matrix Omega used in the quantile residuals tests described by *Kalliovirta and Saikkonen 2010*.

**Usage**

```
get_test_Omega(
  data,
  p,
  M,
  params,
  conditional,
  parametrization,
  constraints,
  g,
  dim_g
)
```

**Arguments**

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

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

**M** a positive integer specifying the number of mixture components.

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

**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:

- $\psi (q \times 1)$  satisfies  $(\phi_1, \dots, \phi_M) = C\psi$ . Here  $C$  is  $(Mpd^2 \times q)$  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. 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. The notation is in line with the cited article by *Kalliovirta, Meitz and Saikkonen (2016)* introducing the GMVAR model.

conditional	a logical argument specifying whether the conditional or exact log-likelihood function should be used.
parametrization	"mean" or "intercept" determining whether the model is parametrized with regime means $\mu_m$ or intercept parameters $\phi_{m,0}$ , $m=1, \dots, M$ .
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 $\psi (q \times 1)$ contains the constrained 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.
g	function g specifying the transformation.
dim_g	output dimension of the transformation g.

## Value

Returns the covariance matrix Omega described by *Kalliovirta and Saikkonen 2010*.

## 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*.

GMVAR

*Create a class 'gmvar' object defining a GMVAR model***Description**

GMVAR creates a class 'gmvar' object that defines a GMVAR model

**Usage**

```
GMVAR(
  data,
  p,
  M,
  d,
  params,
  conditional = TRUE,
  parametrization = c("intercept", "mean"),
  constraints = NULL,
  calc_cond_moments,
  calc_std_errors = FALSE
)

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

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

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

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

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

**Arguments**

<code>data</code>	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.
<code>p</code>	a positive integer specifying the autoregressive order of the model.
<code>M</code>	a positive integer specifying the number of mixture components.
<code>d</code>	number of times series in the system, i.e. <code>ncol(data)</code> . This can be used to define GMVAR models without data and can be ignored if data is provided.

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 + 1) - 1)x1)</math> and have form <math>\theta=(v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})</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> <p><b>For constrained models:</b> Should be size <math>((M(d + d(d+1)/2 + 1) + q - 1)x1)</math> and have form <math>\theta= (\phi_{1,0}, \dots, \phi_{M,0}, \psi, \sigma_1, \dots, \sigma_M, \alpha_1, \dots, \alpha_{M-1})</math>, where:</p> <ul style="list-style-type: none"> <li>• <math>\psi (qx1)</math> satisfies <math>(\phi_1, \dots, \phi_M) = C\psi</math>. Here <math>C</math> is <math>(Mpd^2xq)</math> constraint matrix.</li> </ul> <p>Above, <math>\phi_{m,0}</math> is the intercept parameter, <math>A_{m,i}</math> denotes the <math>i</math>:th coefficient matrix of the <math>m</math>:th mixture component, <math>\Omega_m</math> denotes the error term covariance matrix of the <math>m</math>:th mixture component, and <math>\alpha_m</math> is the mixing weight parameter. If parametrization=="mean", just replace each <math>\phi_{m,0}</math> with regimewise mean <math>\mu_m</math>. <math>vec()</math> is vectorization operator that stacks columns of a given matrix into a vector. <math>vech()</math> stacks columns of a given matrix from the principal diagonal downwards (including elements on the diagonal) into a vector. The notation is in line with the cited article by <i>Kalliovirta, Meitz and Saikkonen (2016)</i> introducing the GMVAR model.</p>
conditional	a logical argument specifying whether the conditional or exact log-likelihood function should be used.
parametrization	"mean" or "intercept" determining whether the model is parametrized with regime means $\mu_m$ or intercept parameters $\phi_{m,0}$ , $m=1, \dots, M$ .
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 constrained 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 <b>not</b> be employed.
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?
object	object of class 'gmvar' generated by fitGMVAR or GMVAR.
...	currently not used.
digits	number of digits to be printed.
x	object of class 'gmvar' generated by fitGMVAR or GMVAR.
summary_print	if set to TRUE then the print will include log-likelihood and information criteria values.

## Details

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

**Value**

Returns an object of class 'gmvar' defining the specified GMVAR model. Can be used to work with other functions provided in `gmvar` package.

Remark 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 'gmvar'
- `print`: print method

**S3 methods**

Only the `print` method is available if data is not provided. If data is provided, then `summary`, `predict`, and `plot` methods are 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*.

**See Also**

[fitGMVAR](#), [add\\_data](#), [swap\\_parametrization](#)

**Examples**

```
# These examples use the data 'eurusd' which comes with the
# package, but in a scaled form.
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

# GMVAR(1,2), d=2 model:
params122 <- c(0.623, -0.129, 0.959, 0.089, -0.006, 1.006, 1.746,
  0.804, 5.804, 3.245, 7.913, 0.952, -0.037, -0.019, 0.943, 6.926,
  3.982, 12.135, 0.789)
mod122 <- GMVAR(data, p=1, M=2, params=params122)
mod122

# GMVAR(1,2), d=2 model without data
mod122_2 <- GMVAR(p=1, M=2, d=2, params=params122)
mod122_2

# GMVAR(2,2), d=2 model with mean-parametrization:
```



```

params222 <- c(-11.904, 154.684, 1.314, 0.145, 0.094, 1.292, -0.389,
  -0.070, -0.109, -0.281, 0.920, -0.025, 4.839, 11.633, 124.983, 1.248,
  0.077, -0.040, 1.266, -0.272, -0.074, 0.034, -0.313, 5.855, 3.570,
  9.838, 0.740)
mod222 <- GMVAR(data, p=2, M=2, params=params222, parametrization="mean")
mod222

# GMVAR(2,2), d=2 model with AR-parameters restricted to be
# the same for both regimes:
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
params222c <- c(1.031, 2.356, 1.786, 3.000, 1.250, 0.060, 0.036,
  1.335, -0.290, -0.083, -0.047, -0.356, 0.934, -0.152, 5.201, 5.883,
  3.560, 9.799, 0.368)
mod222c <- GMVAR(data, p=2, M=2, params=params222c, constraints=C_mat)
mod222c

# GMVAR(2,2), d=2 model with AR-parameters restricted to be
# the same for both regimes and the non-diagonal elements of
# 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)
params222c2 <- c(0.355, 3.193, -0.114, 2.829, 1.263, 1.338, -0.292,
  -0.362, 5.597, 3.456, 9.622, 0.982, -0.327, 5.236, 0.650)
mod222c2 <- GMVAR(data, p=2, M=2, params=params222c2,
  constraints=C_mat2)
mod222c2

```

---

gmvarkit

*gmvarkit: Estimate Gaussian Mixture Vector Autoregressive (GM-  
VAR) model*


---

## Description

gmvarkit is a package for estimating Gaussian Mixture Vector Autoregressive (GMVAR) model. It provides functions for unconstrained and constraints maximum likelihood estimation of the model parameters, quantile residuals tests, graphical diagnostics, simulation from GMVAR processes, and forecasting.

Most of the functions documented are not exported but intended internal use only. The readme file is a good place to start, and the vignette might be useful too.

---

in\_paramspace

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


---

## Description

in\_paramspace checks whether the given parameter vector lies in the parameter space. Does NOT consider the identifiability condition!

**Usage**

```
in_paramspace(p, M, d, params, constraints = NULL)
```

**Arguments**

**p** a positive integer specifying the autoregressive order of the model.  
**M** a positive integer specifying the number of mixture components.  
**d** the number of time series in the system.  
**params** a real valued vector specifying the parameter values.

**For unconstrained models:** Should be size  $((M(pd^2 + d + d(d+1)/2 + 1) - 1) \times 1)$  and have form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where:

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

**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. 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. The notation is in line with the cited article by *Kalliovirta, Meitz and Saikkonen (2016)* introducing the GMVAR model.

**constraints** a size  $(Mpd^2 \times qx)$  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 constrained 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.

**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.

**Examples**

```
# GMVAR(1,1), d=2 model:
params112 <- 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=params112)

# GMVAR(2,2), d=2 model:
params222 <- 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=params222)

# GMVAR(2,2), d=2 model with AR-parameters restricted to be
# the same for both regimes:
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
params222c <- c(1.03, 2.36, 1.79, 3.00, 1.25, 0.06, 0.04,
  1.34, -0.29, -0.08, -0.05, -0.36, 0.93, -0.15, 5.20,
  5.88, 3.56, 9.80, 0.37)
in_paramspace(p=2, M=2, d=2, params=params222c, constraints=C_mat)
```

---

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

---

**Description**

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

**Usage**

```
in_paramspace_int(p, M, d, all_boldA, alphas, all_Omega)
```

**Arguments**

<code>p</code>	a positive integer specifying the autoregressive order of the model.
<code>M</code>	a positive integer specifying the number of mixture components.
<code>d</code>	the number of time series in the system.
<code>all_boldA</code>	3D array containing the $((dp)x(dp))$ "bold A" matrices related to each mixture component VAR-process, obtained from <code>form_boldA</code> . Will be computed if not given.
<code>alphas</code>	(Mx1) vector containing all mixing weight parameters, obtained from <code>pick_alphas</code> .
<code>all_Omega</code>	3D array containing all covariance matrices $\Omega_m$ , obtained from <code>pick_Omegas</code> .

**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.

---

is_stationary	<i>Check the stationary condition of a given GMVAR model</i>
---------------	--

---

## Description

is\_stationary checks the stationarity condition of a GMVAR model.

## Usage

```
is_stationary(p, M, d, params, all_boldA = NULL, tolerance = 0.001)
```

## Arguments

- |           |   |
|-----------|---|
| p         | a positive integer specifying the autoregressive order of the model.  |
| M         | a positive integer specifying the number of mixture components.   |
| d         | the number of time series in the system.  |
| params    | <p>a real valued vector specifying the parameter values. Should be size <math>((M(pd^2 + d + d(d+1)/2 + 1) - 1) \times 1)</math> and have form <math>\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})</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> <p>Above, <math>\phi_{m,0}</math> is the intercept parameter, <math>A_{m,i}</math> denotes the <math>i</math>:th coefficient matrix of the <math>m</math>:th mixture component, <math>\Omega_m</math> denotes the error term covariance matrix of the <math>m</math>:th mixture component, and <math>\alpha_m</math> is the mixing weight parameter.</p> <p>If parametrization=="mean", just replace each <math>\phi_{m,0}</math> with regimewise mean <math>\mu_m</math>. <math>vec()</math> is vectorization operator that stacks columns of a given matrix into a vector. <math>vech()</math> stacks columns of a given matrix from the principal diagonal downwards (including elements on the diagonal) into a vector. The notation is in line with the cited article by KMS (2016) introducing the GMVAR model.</p> |
| 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.  |
| tolerance | Returns FALSE if modulus of any eigenvalue is larger or equal to $1 - \text{tolerance}$ .   |

## Value

Returns TRUE if the model is stationary and FALSE if not. Based on the argument tolerance, is\_stationary may return FALSE when the parameter vector is in the stationarity region, but very close to the boundary (this is used to ensure numerical stability in estimation of the model parameters).

**Warning**

No argument checks!

**References**

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.

---

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

---

**Description**

iterate\_more uses a variable metric algorithm to finalize maximum likelihood estimation of a GMVAR model (object of class 'gmvar') which already has preliminary estimates.

**Usage**

```
iterate_more(gmvar, maxit = 100, calc_std_errors = TRUE)
```

**Arguments**

gmvar	an object of class 'gmvar' created with fitGMVAR or GMVAR.
maxit	the maximum number of iterations in the variable metric algorithm.
calc_std_errors	should approximate standard errors be calculated?

**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 model with the main estimation function fitGMVAR. iterate\_more is essentially a wrapper around the function optim from the package stats and GMVAR from the package gmvarkit.

**Value**

Returns an object of class 'gmvar' defining the estimated GMVAR 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*.

**See Also**

[fitGMVAR](#), [GMVAR](#), [optim](#), [profile\\_logliks](#)

**Examples**

```
## These are long running examples that use parallel computing!

# These examples use the data 'eurusd' which comes with the
# package, but in a scaled form.
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

# GMVAR(1,2) model, only 5 iterations of the variable metric
# algorithm
fit12 <- fitGMVAR(data, p=1, M=2, maxit=5)
fit12

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

# GMVAR(2,2) model with autoregressive parameters restricted
# to be the same for all regimes, only 10 iterations of the
# variable metric algorithm
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
fit22c <- fitGMVAR(data, p=2, M=2, constraints=C_mat, maxit=10)
fit22c

# Iterate more:
fit22c_2 <- iterate_more(fit22c)
fit22c_2

# GMVAR(3,2) model, only 10 iterations of the variable metric
# algorithm
fit32 <- fitGMVAR(data, p=3, M=2, maxit=10)
fit32

# Iterate more:
fit32_2 <- iterate_more(fit32)
fit32_2
```

**Description**

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

**Usage**

```
loglikelihood(
  data,
  p,
  M,
  params,
  conditional = TRUE,
  parametrization = c("intercept", "mean"),
  constraints = NULL,
  minval = NA
)
```

**Arguments**

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

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

**M** a positive integer specifying the number of mixture components.

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

**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:

- $\psi (q \times 1)$  satisfies  $(\phi_1, \dots, \phi_M) = C\psi$ . Here  $C$  is  $(Mpd^2 \times q)$  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. 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. The notation is in line with the cited article by *Kalliovirta, Meitz and Saikkonen (2016)* introducing the GMVAR model.

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

parametrization	"mean" or "intercept" determining whether the model is parametrized with regime means $\mu_m$ or intercept parameters $\phi_{m,0}$ , $m=1,\dots,M$ .
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 constrained 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.
minval	the value that will be returned if the parameter vector does not lie in the parameter space (excluding the identification condition).

### Details

loglikelihood\_int takes use of the function dmvn from the package mvnfast to cut down computation time. Values extremely close to zero are handled with the package Brobdingnag.

### 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.
- Lutkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.

### See Also

[fitGMVAR](#), [GMVAR](#), [calc\\_gradient](#)

### Examples

```
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
params222 <- c(-11.904, 154.684, 1.314, 0.145, 0.094, 1.292, -0.389,
-0.070, -0.109, -0.281, 0.920, -0.025, 4.839, 11.633, 124.983, 1.248,
0.077, -0.040, 1.266, -0.272, -0.074, 0.034, -0.313, 5.855, 3.570,
9.838, 0.740)
loglikelihood(data=data, p=2, M=2, params=params222, parametrization="mean")
```



---

loglikelihood_int	<i>Compute log-likelihood of a Gaussian Mixture Vector Autoregressive model</i>
-------------------	---

---

## Description

loglikelihood\_int computes log-likelihood of a GMVAR model.

## Usage

```
loglikelihood_int(
  data,
  p,
  M,
  params,
  conditional = TRUE,
  parametrization = c("intercept", "mean"),
  constraints = NULL,
  to_return = c("loglik", "mw", "mw_tplus1", "loglik_and_mw", "terms", "regime_cmeans",
    "total_cmeans", "total_ccovs"),
  check_params = TRUE,
  minval = NULL
)
```

## Arguments

data	a matrix or class 'ts' object with $d > 1$ columns. Each column is taken to represent a single time series. NA values are not supported.
p	a positive integer specifying the autoregressive order of the model.
M	a positive integer specifying the number of mixture components.
params	a real valued vector specifying the parameter values.

**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:

- $\psi (q \times 1)$  satisfies  $(\phi_1, \dots, \phi_M) = C\psi$ . Here  $C$  is  $(Mpd^2 \times q)$  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. 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. <code>vech()</code> stacks columns of a given matrix from the principal diagonal downwards (including elements on the diagonal) into a vector. The notation is in line with the cited article by <i>Kalliovirta, Meitz and Saikkonen (2016)</i> introducing the GMVAR model.
conditional	a logical argument specifying whether the conditional or exact log-likelihood function should be used.
parametrization	"mean" or "intercept" determining whether the model is parametrized with regime means $\mu_m$ or intercept parameters $\phi_{m,0}$ , $m=1,\dots,M$ .
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 constrained 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.
to_return	should the returned object be the log-likelihood value, mixing weights, mixing weights including value for $alpha_{m,T+1}$ , a list containing log-likelihood value and mixing weights, or the terms $l_t : t = 1, \dots, T$ in the log-likelihood function (see <i>KMS 2016, eq.(9)</i> )? Or should the regimewise conditional means, total conditional means, or total conditional covariance matrices be returned? Default is the log-likelihood value ("loglik").
check_params	should it be checked that the parameter vector satisfies the model assumptions? Can be skipped to save computation time if it does for sure.
minval	the value that will be returned if the parameter vector does not lie in the parameter space (excluding the identification condition).

### Details

`loglikelihood_int` takes use of the function `dmvn` from the package `mvnfast` to cut down computation time. Values extremely close to zero are handled with the package `Broddingnag`.

### Value

**By default:** log-likelihood value of the specified GMVAR model,

**If `to_return=="mw"`:** a size  $((n_{\text{obs}}-p) \times M)$  matrix containing the mixing weights: for  $m$ :th component in  $m$ :th column.

**If `to_return=="mw_tplus1"`:** a size  $((n_{\text{obs}}-p+1) \times M)$  matrix containing the mixing weights: for  $m$ :th component in  $m$ :th column. The last row is for  $\alpha_{m,T+1}$ .

**If `to_return=="terms"`:** a size  $((n_{\text{obs}}-p) \times 1)$  numeric vector containing the terms  $l_t$ .

**if `to_return=="loglik_and_mw"`:** a list of two elements. The first element contains the log-likelihood value and the second element contains the mixing weights.

**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=="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).

## References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.

---

n_params	Calculate the number of parameters in GMVAR model parameter vector
----------	--

---

## Description

n\_params calculates the number of parameters in the model.

## Usage

```
n_params(p, M, d, constraints = NULL)
```

## Arguments

p	a positive integer specifying the autoregressive order of the model.
M	a positive integer specifying the number of mixture components.
d	the number of time series in the system.
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 constrained 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 $I = diag(p \times d^2)$ . Ignore (or set to NULL) if linear constraints should <b>not</b> be employed.

## Value

Returns the number of parameters in parameter vector of the specified GMVAR model.

## Warning

No argument checks!

## References

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

---

pick\_allA *Pick coefficient all matrices*

---

### Description

pick\_allA picks all coefficient matrices  $A_{m,i}$  ( $i = 1, \dots, p, m = 1, \dots, M$ ) from the given parameter vector so that they are arranged in a 4D array with the fourth dimension indicating each component and third dimension indicating each lag.

### Usage

```
pick_allA(p, M, d, params)
```

### Arguments

**p** a positive integer specifying the autoregressive order of the model.  
**M** a positive integer specifying the number of mixture components.  
**d** the number of time series in the system.  
**params** a real valued vector specifying the parameter values. Should be size  $((M(pd^2 + d + d(d+1)/2 + 1) - 1)x1)$  and have form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where:

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

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.

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. The notation is in line with the cited article by KMS (2016) introducing the GMVAR model.

### Value

Returns a 4D array containing the coefficient matrices of the all components. Coefficient matrix  $A_{m,i}$  can be obtained by choosing  $[, , i, m]$ .

### Warning

No argument checks!

### References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.

---

pick\_all\_phi0\_A      *Pick all  $\phi_{m,0}$  or  $\mu_m$  and  $A_m$  parameter values*

---

### Description

pick\_all\_phi0\_A picks the intercept or mean parameters and vectorized coefficient matrices from the given parameter vector.

### Usage

pick\_all\_phi0\_A(p, M, d, params)

### Arguments

p                    a positive integer specifying the autoregressive order of the model.  
M                    a positive integer specifying the number of mixture components.  
d                    the number of time series in the system.  
params              a real valued vector specifying the parameter values. Should be size  $((M(pd^2 + d + d(d+1)/2 + 1) - 1)x1)$  and have form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where:

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

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.

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. The notation is in line with the cited article by KMS (2016) introducing the GMVAR model.

### Value

Returns a  $((pd^2 + d)xM)$  matrix containing  $(\phi_{m,0}, vec(A_m))$  in the  $m$ :th column, or  $(\mu_m, vec(A_m))$  if the parameter vector is mean-parametrized,  $m=1, \dots, M$ .

### Warning

No argument checks!

### References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.

---

pick_alphas	<i>Pick mixing weight parameters <math>\alpha_m, m = 1, \dots, M</math></i>
-------------	---

---

### Description

pick\_alphas picks the mixing weight parameters from the given parameter vector.

### Usage

```
pick_alphas(p, M, d, params)
```

### Arguments

- |        |  |
|--------|--|
| p      | a positive integer specifying the autoregressive order of the model.   |
| M      | a positive integer specifying the number of mixture components.  |
| d      | the number of time series in the system.   |
| params | a real valued vector specifying the parameter values. 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: <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), m=1, \dots, M</math>.</li> </ul> |

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.

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. The notation is in line with the cited article by KMS (2016) introducing the GMVAR model.

### Value

Returns a length M vector containing the mixing weight parameters  $alpha_m, m = 1, \dots, M$ , including non-parametrized  $alpha_M$ .

### Warning

No argument checks!

### References

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

---

pick\_Am *Pick coefficient matrices*

---

### Description

pick\_Am picks the coefficient matrices  $A_{m,i}$  ( $i = 1, \dots, p$ ) from the given parameter vector so that they are arranged in a 3D array with the third dimension indicating each lag.

### Usage

```
pick_Am(p, M, d, params, m)
```

### Arguments

**p** a positive integer specifying the autoregressive order of the model.  
**M** a positive integer specifying the number of mixture components.  
**d** number of time series in the system, i.e. the dimension.  
**params** a real valued vector specifying the parameter values.

**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. 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. The notation is in line with the cited article by *Kalliovirta, Meitz and Saikkonen (2016)* introducing the GMVAR model.

**m** which component?

### Value

Returns a 3D array containing the coefficient matrices of the given component. A coefficient matrix  $A_{m,i}$  can be obtained by choosing  $[, , i]$ .

### Warning

No argument checks!

## References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.

---

pick_Ami	<i>Pick coefficient matrix</i>
----------	--------------------------------

---

## Description

pick\_Ami picks the coefficient matrix  $A_{m,i}$  from the given parameter vector.

## Usage

```
pick_Ami(p, M, d, params, m, i, unvec = TRUE)
```

## Arguments

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

M a positive integer specifying the number of mixture components.

d number of time series in the system, i.e. the dimension.

params a real valued vector specifying the parameter values.

**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:

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

**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. 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. The notation is in line with the cited article by Kalliovirta, Meitz and Saikkonen (2016) introducing the GMVAR model.

m which component?

i which lag in 1, ..., p?

unvec if FALSE then vectorized version of  $A_{m,i}$  will be returned instead of matrix. Default if TRUE.



**Value**

Returns the  $i$ :th lag coefficient matrix of  $m$ :th component,  $A_{m,i}$ .

**Warning**

No argument checks!

**References**

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.

---

pick\_Omegas

*Pick covariance matrices*

---

**Description**

pick\_Omegas picks the covariance matrices  $\Omega_m (m = 1, \dots, M)$  from the given parameter vector so that they are arranged in a 3D array with the third dimension indicating each component.

**Usage**

pick\_Omegas(p, M, d, params)

**Arguments**

- |        |   |
|--------|---|
| p      | a positive integer specifying the autoregressive order of the model.  |
| M      | a positive integer specifying the number of mixture components.   |
| d      | the number of time series in the system.  |
| params | a real valued vector specifying the parameter values. 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: <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> |

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.

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. The notation is in line with the cited article by KMS (2016) introducing the GMVAR model.

**Value**

Returns a 3D array containing the covariance matrices of the given model. Coefficient matrix  $\Omega_m$  can be obtained by choosing  $[ , , m]$ .

**Warning**

No argument checks!

**References**

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

---

pick_phi0	Pick $\phi_{m,0}$ or $\mu_m$ , $m=1,\dots,M$ vectors
-----------	--

---

**Description**

pick\_phi0 picks the intercept or mean parameters from the given parameter vector.

**Usage**

```
pick_phi0(p, M, d, params)
```

**Arguments**

- |        |   |
|--------|---|
| p      | a positive integer specifying the autoregressive order of the model.  |
| M      | a positive integer specifying the number of mixture components.   |
| d      | the number of time series in the system.  |
| params | a real valued vector specifying the parameter values. 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: <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> |

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.

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. The notation is in line with the cited article by KMS (2016) introducing the GMVAR model.

**Value**

Returns a  $(dxM)$  matrix containing  $\phi_{m,0}$  in the  $m$ :th column or  $\mu_m$  if the parameter vector is mean-parametrized,  $m = 1, \dots, M$ .

**Warning**

No argument checks!

**References**

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.

---

pick_regime	<i>Pick regime parameters <math>v_m = (\phi_{m,0}, \phi_m, \sigma_m)</math></i>
-------------	---

---

**Description**

pick\_regime picks the regime-parameters from the given parameter vector.

**Usage**

```
pick_regime(p, M, d, params, m)
```

**Arguments**

**p** a positive integer specifying the autoregressive order of the model.  
**M** a positive integer specifying the number of mixture components.  
**d** number of time series in the system, i.e. the dimension.  
**params** a real valued vector specifying the parameter values.

**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. 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. The notation is in line with the cited article by *Kalliovirta, Meitz and Saikkonen (2016)* introducing the GMVAR model.

m which component?

### Value

Returns length  $pd^2+d+d(d+1)/2$  vector containing  $v_m = (\phi_{m,0}, \phi_m, \sigma_m)$ , where  $\phi_m = (vec(A_{m,1}), \dots, vec(A_{m,p}))$  and  $\sigma_m = vech(\Omega_m)$ .

### Warning

No argument checks!

### References

- Kalliovirta L., Meitz M. and Saikkonen P. 2016. Gaussian mixture vector autoregression. *Journal of Econometrics*, **192**, 485-498.
- Lutkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.

---

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

---

### Description

plot.gmvarpred is plot method for gmvarpred objects.

### Usage

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

### Arguments

x	object of class 'gmvarpred' generated by predict.gmvar.
...	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 GMVAR processes

**References**

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

---

plot.qrtest	<i>Quantile residual tests</i>
-------------	--------------------------------

---

**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(
  gmvar,
  lags_ac = c(1:2, 4, 8),
  lags_ch = lags_ac,
  nsimu = 1,
  print_res = TRUE
)
```

**Arguments**

x	object of class 'qrtest' generated by the function quantile_residual_tests).
...	currently not used.
digits	number of decimals to print
gmvar	an object of class 'gmvar' created with fitGMVAR or GMVAR.
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.
nsimu	to how many simulations should the covariance matrix Omega used in the qrtests 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 increase the computation time.
print_res	should the test results be printed while computing the tests?

**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*.

**See Also**

[fitGMVAR](#), [GMVAR](#), [quantile\\_residuals](#), [diagnostic\\_plot](#), [predict.gmvar](#), [profile\\_logliks](#),

**Examples**

```
## These are long running examples that use parallel computing!

# These examples use the data 'eurusd' which comes with the
# package, but in a scaled form.
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

# GMVAR(1,2) model with default settings
fit12 <- fitGMVAR(data, p=1, M=2)
qrtests12 <- quantile_residual_tests(fit12, nsimu=1)
qrtests12
plot(qrtests12)

# GMVAR(2,2) model with mean parametrization
fit22 <- fitGMVAR(data, p=2, M=2, parametrization="mean",
  ncalls=1, seeds=20)
qrtests22 <- quantile_residual_tests(fit22, nsimu=1)
qrtests22

# GMVAR(2,2) model with autoregressive parameters restricted
# to be the same for all regimes
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
fit22c <- fitGMVAR(data, p=2, M=2, constraints=C_mat, ncalls=12)
qrtests22c <- quantile_residual_tests(fit22c, lags_ac=c(1, 4),
```

```

                                nsimu=10000, print_res=TRUE)
qrtests22c

```

---

predict.gmvar

*Predict method for class 'gmvar' objects*

---

## Description

Forecast GMVAR process defined as a class 'gmvar' object. The forecasts 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', generated by function fitGMVAR or 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 'gmvarpred' 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.

**Examples**

```
## These are long running examples that use parallel computing!

# These examples use the data 'eurusd' which comes with the
# package, but in a scaled form.
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

# GMVAR(2,2) model
fit22 <- fitGMVAR(data, p=2, M=2, ncalls=12, seeds=42:53)
p1 <- predict(fit22, n_ahead=20, pred_type="median")
p1
p2 <- predict(fit22, n_ahead=10, nt=20, lty=1)
p2
p3 <- predict(fit22, n_ahead=10, pi=c(0.99, 0.90, 0.80, 0.70),
             nt=30, lty=0)
p3

# GMVAR(1,2) model
fit12 <- fitGMVAR(data, p=1, M=2)
p1 <- predict(fit12, n_ahead=1, pred_type="cond_mean",
             plot_res=FALSE)
p1
p2 <- predict(fit12, n_ahead=10, pred_type="mean")
p2
p3 <- predict(fit12, n_ahead=10, pi_type="upper")
p3
```



---

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

---

### Description

print.gmvarpred is a print method for object generated by predict.gmvar.

### Usage

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

### Arguments

x	object of class 'gmvarpred' generated by predict.gmvar.
...	currently not used.
digits	number of decimals to print

### Examples

```
# This example uses the data 'eurusd' which comes with the  
# package, but in a scaled form.  
data <- cbind(10*eurusd[,1], 100*eurusd[,2])  
colnames(data) <- colnames(eurusd)  
  
# GMVAR(2,2), d=2 model  
params222 <- c(-11.904, 154.684, 1.314, 0.145, 0.094, 1.292, -0.389,  
-0.070, -0.109, -0.281, 0.920, -0.025, 4.839, 11.633, 124.983, 1.248,  
0.077, -0.040, 1.266, -0.272, -0.074, 0.034, -0.313, 5.855, 3.570,  
9.838, 0.740)  
mod222 <- GMVAR(data, p=2, M=2, params=params222, parametrization="mean")  
pred222 <- predict(mod222, n_ahead=3, plot_res=FALSE)  
print(pred222)  
print(pred222, digits=3)
```

---

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

---

### Description

print.gmvarsum is a print method for object 'gmvarsum' generated by summary.gmvar.

### Usage

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

**Arguments**

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

**Examples**

```
# This example uses the data 'eurusd' which comes with the
# package, but in a scaled form.
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

# GMVAR(2,2), d=2 model
params222 <- c(-11.904, 154.684, 1.314, 0.145, 0.094, 1.292, -0.389,
-0.070, -0.109, -0.281, 0.920, -0.025, 4.839, 11.633, 124.983, 1.248,
 0.077, -0.040, 1.266, -0.272, -0.074, 0.034, -0.313, 5.855, 3.570,
 9.838, 0.740)
mod222 <- GMVAR(data, p=2, M=2, params=params222, parametrization="mean")
sumry222 <- summary(mod222)
print(sumry222)
```

---

print\_std\_errors            *Print standard errors of GMVAR model in the same form as the model estimates are printed*

---

**Description**

print\_std\_errors prints the approximate standard errors of GMVAR model in the same form as the parameters of objects of class 'gmvar' are printed.

**Usage**

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

**Arguments**

gmvar                an object of class 'gmvar' created with fitGMVAR or GMVAR.  
 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 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*.

## See Also

[profile\\_logliks](#), [fitGMVAR](#), [GMVAR](#), [print.gmvar](#), [swap\\_parametrization](#)

## Examples

```
## These are long running examples that use parallel computing!

# These examples use the data 'eurusd' which comes with the
# package, but in a scaled form.
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

# GMVAR(1,2) model with default settings
fit12 <- fitGMVAR(data, p=1, M=2)
fit12
print_std_errors(fit12)

# GMVAR(2,2) model with mean parametrization
fit22 <- fitGMVAR(data, p=2, M=2, parametrization="mean")
fit22
print_std_errors(fit22)

# GMVAR(2,2) model with autoregressive parameters restricted
# to be the same for all regimes
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
fit22c <- fitGMVAR(data, p=2, M=2, constraints=C_mat)
fit22c
print_std_errors(fit22c)

# GMVAR(2,2) model with autoregressive parameters restricted
# to be the same for all 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 <- fitGMVAR(data, p=2, M=2, constraints=C_mat2, ncalls=10)
fit22c2
print_std_errors(fit22c2)
```

---

profile\_logliks      *Plot profile log-likelihoods around the estimates*

---

### Description

profile\_logliks plots profile log-likelihoods around the estimates.

### Usage

```
profile_logliks(
  gmvar,
  which_pars,
  scale = 0.02,
  nrows,
  ncols,
  precision = 200
)
```

### Arguments

**gmvar**            an object of class 'gmvar' created with fitGMVAR or GMVAR.  
**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:

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

**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:

- $\psi$  ( $q \times 1$ ) satisfies  $(\phi_1, \dots, \phi_M) = C\psi$ . Here  $C$  is  $(Mpd^2 \times q)$  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 `which_pars`.

**precision**        at how many points should each profile log-likelihood be evaluated at?

## 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.
- Lutkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.

## See Also

[get\\_soc](#), [diagnostic\\_plot](#), [fitGMVAR](#), [GMVAR](#)

## Examples

```
# These examples use the data 'eurusd' which comes with the
# package, but in a scaled form (similar to Kalliovirta et al. 2016).
data(eurusd, package="gmvarkit")
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

# GMVAR(1,2) model: 10 estimation rounds with seeds set
# for reproducibility
fit12 <- fitGMVAR(data, p=1, M=2, ncalls=10, seeds=1:10)
fit12
profile_logliks(fit12)
# GMVAR(2,2) model with mean parametrization
fit22 <- fitGMVAR(data, p=2, M=2, parametrization="mean",
                 ncalls=16, seeds=11:26)
profile_logliks(fit22)

# 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 <- fitGMVAR(data, p=2, M=2, constraints=C_mat)
profile_logliks(fit22c)

# 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 <- fitGMVAR(data, p=2, M=2, constraints=C_mat2)
```

```
profile_logliks(fit22c2)
```

---

```
quantile_residuals      Calculate multivariate quantile residuals of GMVAR model
```

---

### Description

`quantile_residuals` calculates multivariate quantile residuals (described by *Kalliovirta and Saikkonen 2010*) for GMVAR model.

### Usage

```
quantile_residuals(gmvar)
```

### Arguments

`gmvar` an object of class 'gmvar' created with `fitGMVAR` or `GMVAR`.

### 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*.

### See Also

[fitGMVAR](#), [GMVAR](#), [quantile\\_residual\\_tests](#), [diagnostic\\_plot](#), [predict.gmvar](#), [profile\\_logliks](#)

### Examples

```
# These examples use the data 'eurusd' which comes with the
# package, but in a scaled form.
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

# GMVAR(1,2), d=2 model:
params122 <- c(0.623, -0.129, 0.959, 0.089, -0.006, 1.006, 1.746,
0.804, 5.804, 3.245, 7.913, 0.952, -0.037, -0.019, 0.943, 6.926,
3.982, 12.135, 0.789)
```

```

mod122 <- GMVAR(data, p=1, M=2, params=params122)
quantile_residuals(mod122)

# GMVAR(2,2), d=2 model with mean-parametrization:
params222 <- c(-11.904, 154.684, 1.314, 0.145, 0.094, 1.292, -0.389,
-0.070, -0.109, -0.281, 0.920, -0.025, 4.839, 11.633, 124.983, 1.248,
0.077, -0.040, 1.266, -0.272, -0.074, 0.034, -0.313, 5.855, 3.570,
9.838, 0.740)
mod222 <- GMVAR(data, p=2, M=2, params=params222, parametrization="mean")
quantile_residuals(mod222)

# GMVAR(2,2), d=2 model with AR-parameters restricted to be
# the same for both regimes:
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
params222c <- c(1.031, 2.356, 1.786, 3.000, 1.250, 0.060, 0.036,
1.335, -0.290, -0.083, -0.047, -0.356, 0.934, -0.152, 5.201, 5.883,
3.560, 9.799, 0.368)
mod222c <- GMVAR(data, p=2, M=2, params=params222c, constraints=C_mat)
quantile_residuals(mod222c)

# GMVAR(2,2), d=2 model with AR-parameters restricted to be
# the same for both regimes and the non-diagonal elements of
# 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)
params222c2 <- c(0.355, 3.193, -0.114, 2.829, 1.263, 1.338, -0.292,
-0.362, 5.597, 3.456, 9.622, 0.982, -0.327, 5.236, 0.650)
mod222c2 <- GMVAR(data, p=2, M=2, params=params222c2,
constraints=C_mat2)
quantile_residuals(mod222c2)

```

---

quantile\_residuals\_int

*Calculate multivariate quantile residuals of GMVAR model*

---

## Description

quantile\_residuals\_int is a wrapper for quantile\_residuals to compute quantile residuals using parameter values instead of class gmvar object.

## Usage

```

quantile_residuals_int(
  data,
  p,
  M,
  params,
  conditional,

```

```

    parametrization,
    constraints
)

```

### Arguments

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

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

**M** a positive integer specifying the number of mixture components.

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

**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. If `parametrization=="mean"`, just replace each  $\phi_{m,0}$  with regime-wise 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. The notation is in line with the cited article by *Kalliovirta, Meitz and Saikkonen (2016)* introducing the GMVAR model.

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

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

**constraints** a size  $(Mpd^2 \times qx)$  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 constrained 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.

### 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.

### Warning

No argument checks!

### 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*.

---

random_coefmats	Create random VAR-model ( $d \times d$ ) coefficient matrices $A$ .
-----------------	---

---

### Description

random\_coefmats generates random VAR model coefficient matrices.

### Usage

```
random_coefmats(d, how_many, scale)
```

### Arguments

d	the number of time series in the system.
how_many	how many ( $d \times d$ ) coefficient matrices $A$ should be drawn?
scale	non-diagonal elements will be drawn from mean zero normal distribution with $sd=0.3/scale$ and diagonal elements from one with $sd=0.6/scale$ . Larger scale will hence more likely result stationary coefficient matrices, but will explore smaller area of the parameter space. Can be for example $1 + \log(2 * \text{mean}(c((p-0.2)^(1.25), d)))$

### Value

Returns  $((\text{how\_many} * d^2) \times 1)$  vector containing vectorized coefficient matrices  $(\text{vec}(A_1), \dots, \text{vec}(A_{\text{how\_many}}))$ . Note that if  $\text{how\_many} = p$ , then the returned vector equals  $\phi_m$ .

---

random_coefmats2	Create random stationary VAR model ( $d \times d$ ) coefficient matrices $A$ .
------------------	--

---

### Description

random\_coefmats2 generates random VAR model coefficient matrices.

### Usage

```
random_coefmats2(p, d, ar_scale = 1)
```

### Arguments

p	a positive integer specifying the autoregressive order of the model.
d	the number of time series in the system.
ar_scale	a positive real number. Larger values will typically result larger AR coefficients.

### 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 affects the coefficient matrices. Read the cited article by Ansley and Kohn (1986) and the source code for more information.

### Value

Returns  $((pd^2) \times 1)$  vector containing stationary vectorized coefficient matrices  $(vec(A_1), \dots, vec(A_p))$ .

### 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.

---

random_covmat	Create random VAR model error term covariance matrix
---------------	--

---

### Description

random\_covmat generates random VAR model ( $d \times d$ ) error term covariance matrix  $\Omega$  from (scaled) Wishart distribution.

### Usage

```
random_covmat(d, omega_scale)
```

**Arguments**

d	the number of time series in the system.
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 $\text{diag}(\text{omega\_scale})$ . Standard deviations of the diagonal elements are $\sqrt{2/d} * \text{omega\_scale}[i]$ and for non-diagonal elements they are $\sqrt{1/d * \text{omega\_scale}[i] * \text{omega\_scale}[j]}$ . Note that for $d > 4$ this scale may need to be chosen carefully. Default in <code>GAfit</code> is <code>var(stats::ar(data[,i],order.max=10)\$resid</code>

**Value**

Returns  $(d(d+1)/2x1)$  vector containing vech-vectorized covariance matrix  $\Omega$ .

---

random_ind	<i>Create random mean-parametrized parameter vector of a GMVAR model that may not be stationary</i>
------------	---

---

**Description**

random\_ind generates random mean-parametrized parameter vector that may not be stationary.

**Usage**

```
random_ind(p, M, d, constraints = NULL, mu_scale, mu_scale2, omega_scale)
```

**Arguments**

p	a positive integer specifying the autoregressive order of the model.
M	a positive integer specifying the number of mixture components.
d	the number of time series in the system.
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 constrained 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 <b>not</b> be employed.
mu_scale	a size $(dx1)$ vector defining <b>means</b> of the normal distributions from which each mean parameter $\mu_m$ is drawn from in random mutations. Default is <code>colMeans(data)</code> . Note that mean-parametrization is always used for optimization in <code>GAfit</code> - even when <code>parametrization="intercept"</code> . However, input (in <code>initpop</code> ) and output (return value) parameter vectors can be intercept-parametrized.
mu_scale2	a size $(dx1)$ strictly positive vector defining <b>standard deviations</b> of the normal distributions from which each mean parameter $\mu_m$ is drawn from in random mutations. Default is <code>2*sd(data[,i]), i=1, ..., d</code> .

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  $\text{diag}(\text{omega\_scale})$ . Standard deviations of the diagonal elements are  $\sqrt{2/d} * \text{omega\_scale}[i]$  and for non-diagonal elements they are  $\sqrt{1/d * \text{omega\_scale}[i] * \text{omega\_scale}[j]}$ . 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`

### Value

Returns random mean-parametrized parameter vector that has form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where:

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

### References

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

---

random_ind2	<i>Create somewhat random parameter vector of a GMVAR 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, mu_scale, mu_scale2, omega_scale, ar_scale = 1)
```

### Arguments

p	a positive integer specifying the autoregressive order of the model.
M	a positive integer specifying the number of mixture components.
d	the number of time series in the system.
mu_scale	a size ( $dx1$ ) vector defining <b>means</b> of the normal distributions from which each mean parameter $\mu_m$ is drawn from in random mutations. Default is <code>colMeans(data)</code> . Note that mean-parametrization is always used for optimization in <code>GAFit</code> - even when <code>parametrization="intercept"</code> . However, input (in <code>initpop</code> ) and output (return value) parameter vectors can be intercept-parametrized.
mu_scale2	a size ( $dx1$ ) strictly positive vector defining <b>standard deviations</b> of the normal distributions from which each mean parameter $\mu_m$ is drawn from in random mutations. Default is <code>2*sd(data[,i]), i=1, ..., d</code> .

omega_scale	a size ( $d \times 1$ ) 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 $\text{diag}(\text{omega\_scale})$ . Standard deviations of the diagonal elements are $\sqrt{2/d} * \text{omega\_scale}[i]$ and for non-diagonal elements they are $\sqrt{1/d * \text{omega\_scale}[i] * \text{omega\_scale}[j]}$ . Note that for $d > 4$ this scale may need to be chosen carefully. Default in <code>GAFit</code> is <code>var(stats::ar(data[,i], order.max=10)\$resid</code>
ar_scale	a positive real number adjusting how large AR parameter values are typically generated in some random mutations. See the function <code>random_coefmats2</code> for details. This is ignored when estimating constrained models.

### 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` 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.

Constrained models are not supported!

### Value

Returns random mean-parametrized parameter vector that has form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where:

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

### 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.

---

reform\_constrained\_pars

*Reform constrained parameter vector into the "standard" form*

---

### Description

`reform_constrained_pars` reforms constrained parameter vector into the form that corresponds to unconstrained parameter vectors.

**Usage**

```
reform_constrained_pars(p, M, d, params, constraints = NULL, change_na = FALSE)
```

**Arguments**

**p** a positive integer specifying the autoregressive order of the model.  
**M** a positive integer specifying the number of mixture components.  
**d** the number of time series in the system.  
**params** a real valued vector specifying the parameter values.

**For unconstrained models:** Should be size  $((M(pd^2 + d + d(d+1)/2 + 1) - 1)x1)$  and have form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where:

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

**For constrained models:** Should be size  $((M(d + d(d+1)/2 + 1) + q - 1)x1)$  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  $(Mpd^2xq)$  constraint matrix.

Above,  $\phi_{m,0}$  is the intercept parameter,  $A_{m,i}$  denotes the  $i$ :th coefficient matrix of the  $m$ :th mixture component,  $\Omega_m$  denotes the error term covariance matrix of the  $m$ :th mixture component, and  $\alpha_m$  is the mixing weight parameter. If `parameterization=="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. The notation is in line with the cited article by *Kalliovirta, Meitz and Saikkonen (2016)* introducing the GMVAR model.

**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 constrained 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.

**change\_na** change NA parameter values of constrained models to -9.999?

**Value**

Returns "regular model" parameter vector corresponding to the constraints.

**Warning**

No argument checks!

**References**

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

---

reform_data	<i>Reform data</i>
-------------	--------------------

---

**Description**

reform\_data reforms the data into a form that is easier to use when calculating log-likelihood values etc.

**Usage**

```
reform_data(data, p)
```

**Arguments**

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

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

**Value**

Returns the data reformed into a  $((n_{obs} - p + 1) \times (dp))$  matrix. The  $i$ :th row of the matrix contains the vector  $(y'_{i-1}, \dots, y'_{i-p})$   $((dp) \times 1)$ , where  $y_i = (y_{1i}, \dots, y_{di})$   $(dx1)$ .

**Warning**

No argument checks!

---

regime_distance	<i>Calculate "distance" between two (scaled) regimes <math>v_m = (\phi_m, 0, \phi_m, \sigma_m)</math></i>
-----------------	---

---

**Description**

regime\_distance calculates "distance" between two scaled regimes. This is used in the genetic algorithm.

**Usage**

```
regime_distance(regime_pars1, regime_pars2)
```

**Arguments**

regime\_pars1 a length  $pd^2 + d + d(d+1)/2$  vector  $v_m = (\phi_{m,0}, \phi_m, \sigma_m)$ .  
 regime\_pars2 a length  $pd^2 + d + d(d+1)/2$  vector  $v_m = (\phi_{m,0}, \phi_m, \sigma_m)$ .

**Value**

Returns "distance" between regime\_pars1 and regime\_pars2. Values are scaled before calculating the "distance". Read the source code for more details.

**Warning**

No argument checks!

**References**

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

---

simulateGMVAR	<i>Simulate from GMVAR process</i>
---------------	------------------------------------

---

**Description**

simulateGMVAR simulates observations from a GMVAR process.

**Usage**

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

**Arguments**

gmvar an object of class 'gmvar' created with fitGMVAR or GMVAR.  
 nsimu number of observations to be simulated.  
 init\_values a size  $(pxd)$  matrix specifying the initial values to be used in the simulation, where d is the number of time series in the system. The **last** row will be used as initial values for the first lag, the second last row for second lag etc. If not specified, initial values will be drawn from the stationary distribution.  
 ntimes how many sets of simulations should be performed?  
 drop if TRUE (default) then the components of the returned list are coerced to lower dimension if ntimes==1, i.e., \$sample and \$mixing\_weights will be matrices, and \$component will be vector.

**Details**

The argument ntimes is intended for forecasting: a GMVAR 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 (`nsimuxdxntimes`) 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 (`nsimuxntimes`) matrix containing the information from which mixture component each value was generated from.

`$mixing_weights` a size (`nsimuxMxntimes`) 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.

**Examples**

```
# These examples use the data 'eurusd' which comes with the
# package, but in a scaled form.
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

# GMVAR(1,2), d=2 process, initial values from the stationary
# distribution
params122 <- c(0.623, -0.129, 0.959, 0.089, -0.006, 1.006, 1.746,
  0.804, 5.804, 3.245, 7.913, 0.952, -0.037, -0.019, 0.943, 6.926,
  3.982, 12.135, 0.789)
mod122 <- GMVAR(p=1, M=2, d=2, params=params122)
set.seed(1)
sim122 <- simulateGMVAR(mod122, nsimu=500)
plot.ts(sim122$sample)
ts.plot(sim122$mixing_weights, col=c("blue", "red"), lty=2)
plot(sim122$component, type="l")

## FORECASTING EXAMPLE
# Forecast 5-steps-ahead, 10000 sets of simulations with initial
# values from the data:
# GMVAR(2,2), d=2 model with mean-parametrization:
params222 <- c(-11.904, 154.684, 1.314, 0.145, 0.094, 1.292, -0.389,
  -0.070, -0.109, -0.281, 0.920, -0.025, 4.839, 11.633, 124.983, 1.248,
  0.077, -0.040, 1.266, -0.272, -0.074, 0.034, -0.313, 5.855, 3.570,
  9.838, 0.740)
mod222 <- GMVAR(data, p=2, M=2, params=params222, parametrization="mean")
sim222 <- simulateGMVAR(mod222, nsimu=5, ntimes=10000)

# Point forecast + 95% prediction intervals:
apply(sim222$sample, 1:2, quantile, probs=c(0.025, 0.5, 0.972))
```

```
# Similar forecast for the mixing weights:
apply(sim222$mixing_weights, 1:2, quantile, probs=c(0.025, 0.5, 0.972))

# GMVAR(2,2), d=2 model with AR parameters restricted to be
# the same for both regimes, custom initial values:
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
params222c <- c(1.031, 2.356, 1.786, 3.000, 1.250, 0.060, 0.036,
  1.335, -0.290, -0.083, -0.047, -0.356, 0.934, -0.152, 5.201, 5.883,
  3.560, 9.799, 0.368)
mod222c <- GMVAR(data, p=2, M=2, params=params222c, constraints=C_mat)
sim222c <- simulateGMVAR(mod222c, nsimu=100,
  init_values=matrix(c(30, 30, 80, 80), nrow=2))
plot.ts(sim222c$sample)
ts.plot(sim222c$mixing_weights, col=c("blue", "red"), lty=2)
plot(sim222c$component, type="l")
```

---

smart_covmat	<i>Create random VAR-model (<math>d \times d</math>) error term covariance matrix <math>\Omega</math> fairly close to a given <b>positive definite</b> covariance matrix using (scaled) Wishart distribution</i>
--------------	--

---

## Description

random\_covmat generates random VAR model ( $d \times d$ ) error term covariance matrix  $\Omega$  from (scaled) Wishart distribution that is fairly close to the given matrix.

## Usage

```
smart_covmat(d, Omega, accuracy)
```

## Arguments

- |          |   |
|----------|---|
| d        | the number of time series in the system.  |
| Omega    | a symmetric positive definite ( $d \times d$ ) covariance matrix specifying expected value of the matrix to be generated.   |
| accuracy | a positive real number adjusting how close to the given covariance matrix the returned individual should be. Standard deviation of each diagonal element is <ul style="list-style-type: none"> <li>• <math>\omega_{i,i}/\text{accuracy}</math> when <math>\text{accuracy} &gt; d/2</math></li> <li>• and <math>\sqrt{2/d} * \omega_{i,i}</math> when <math>\text{accuracy} \leq d/2</math>.</li> </ul> Wishart distribution is used, but for more details read the source code. |

## Value

Returns  $(d(d+1)/2 \times 1)$  vector containing vech-vectorized covariance matrix  $\Omega$ .

---

smart_ind	<i>Create random parameter vector of a GMVAR model fairly close to a given parameter vector</i>
-----------	---

---

### Description

smart\_ind creates random mean-parametrized parameter vector of a GMVAR model fairly close to a given parameter vector. The result may not be stationary.

### Usage

```
smart_ind(
  p,
  M,
  d,
  params,
  constraints = NULL,
  accuracy = 1,
  which_random = numeric(0),
  mu_scale,
  mu_scale2,
  omega_scale,
  ar_scale = 1
)
```

### Arguments

- |        |   |
|--------|---|
| p      | a positive integer specifying the autoregressive order of the model.  |
| M      | a positive integer specifying the number of mixture components.   |
| d      | the number of time series in the system.  |
| params | a real valued vector specifying the parameter values. 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: <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> |

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.

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. The notation is in line with the cited article by KMS (2016) introducing the GMVAR model.

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 constrained 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.
accuracy	a positive real number adjusting how close to the given parameter vector the returned individual should be. Larger number means larger accuracy. Read the source code for details.
which_random	a vector with length between 1 and M specifying the mixture components that should be random instead of close to the given parameter vector. If constraints are employed, then this does not consider AR coefficients. Default is NULL.
mu_scale	a size $(dx1)$ vector defining <b>means</b> of the normal distributions from which each mean parameter $\mu_m$ is drawn from in random mutations. Default is <code>colMeans(data)</code> . Note that mean-parametrization is always used for optimization in <code>GAFit</code> - even when <code>parametrization=="intercept"</code> . However, input (in <code>initpop</code> ) and output (return value) parameter vectors can be intercept-parametrized.
mu_scale2	a size $(dx1)$ strictly positive vector defining <b>standard deviations</b> of the normal distributions from which each mean parameter $\mu_m$ is drawn from in random mutations. Default is <code>2*sd(data[,i]), i=1, . . . , d</code> .
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 <code>diag(omega_scale)</code> . Standard deviations of the diagonal elements are <code>sqrt(2/d)*omega_scale[i]</code> and for non-diagonal elements they are <code>sqrt(1/d*omega_scale[i]*omega_scale[j])</code> . Note that for $d > 4$ this scale may need to be chosen carefully. Default in <code>GAFit</code> is <code>var(stats::ar(data[,i], order.max=10)\$resid</code>
ar_scale	a positive real number adjusting how large AR parameter values are typically generated in some random mutations. See the function <code>random_coefmats2</code> for details. This is ignored when estimating constrained models.

### Value

Returns random mean-parametrized parameter vector that has form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where:

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

### Warning

No argument checks!

## References

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

---

sort_components	<i>Sort components in parameter vector according to mixing weights into a decreasing order</i>
-----------------	--

---

## Description

sort\_components sorts mixture components in the parameter vector according to mixing weights into a decreasing order.

## Usage

```
sort_components(p, M, d, params)
```

## Arguments

- |        |   |
|--------|---|
| p      | a positive integer specifying the autoregressive order of the model.  |
| M      | a positive integer specifying the number of mixture components.   |
| d      | the number of time series in the system.  |
| params | a real valued vector specifying the parameter values. 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: <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> |

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.

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. The notation is in line with the cited article by KMS (2016) introducing the GMVAR model.

## Details

Constrained parameter vectors are not supported!

**Value**

Returns sorted parameter vector with  $\alpha_1 > \dots > \alpha_m$ , that has form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where:

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

Above,  $\phi_{m,0}$  is the intercept parameter,  $A_{m,i}$  denotes the  $i$ :th coefficient matrix of the  $m$ :th component,  $\Omega_m$  denotes the error term covariance matrix of the  $m$ :th component, and  $\alpha_m$  is the mixing weight parameter.  $\text{vec}()$  is vectorization operator that stack columns of the given matrix into a vector.  $\text{vech}()$  stacks columns of the given matrix from the principal diagonal downwards (including elements on the diagonal) to form a vector. The notation is in line with the cited article by KMS (2016) introducing the GMVAR model.

**Warning**

No argument checks!

**References**

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

---

standard\_errors

*Calculate standard errors for estimates of GMVAR model*

---

**Description**

standard\_errors numerically calculates approximate standard errors for the GMVAR model using square roots of the diagonal of inverse of observed information matrix.

**Usage**

```
standard_errors(
  data,
  p,
  M,
  params,
  conditional = TRUE,
  parametrization = c("intercept", "mean"),
  constraints = NULL,
  minval
)
```

**Arguments**

- data a matrix or class 'ts' object with  $d > 1$  columns. Each column is taken to represent a single time series. NA values are not supported.
- p a positive integer specifying the autoregressive order of the model.
- M a positive integer specifying the number of mixture components.
- params a real valued vector specifying the parameter values.

**For unconstrained models:** Should be size  $((M(pd^2 + d + d(d + 1)/2 + 1) - 1)x1)$  and have form  $\theta = (v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where:

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

**For constrained models:** Should be size  $((M(d + d(d + 1)/2 + 1) + q - 1)x1)$  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  $(Mpd^2 xq)$  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. 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. The notation is in line with the cited article by *Kalliovirta, Meitz and Saikkonen (2016)* introducing the GMVAR model.

- conditional a logical argument specifying whether the conditional or exact log-likelihood function should be used.
- parametrization "mean" or "intercept" determining whether the model is parametrized with regime means  $\mu_m$  or intercept parameters  $\phi_{m,0}$ ,  $m=1, \dots, M$ .
- constraints a size  $(Mpd^2 xq)$  constraint matrix  $C$  specifying general linear constraints to the autoregressive parameters. We consider constraints of form  $(\phi_1, \dots, \phi_M) = C\psi$ , where  $\phi_m = (vec(A_{m,1}), \dots, vec(A_{m,p}))(pd^2 x1)$ ,  $m = 1, \dots, M$  contains the coefficient matrices and  $\psi (qx1)$  contains the constrained parameters. For example, to restrict the AR-parameters to be the same for all regimes, set  $C = [I: \dots: I]'$   $(Mpd^2 xpd^2)$  where  $I = \text{diag}(p \cdot d^2)$ . Ignore (or set to NULL) if linear constraints should **not** be employed.
- minval the value that will be returned if the parameter vector does not lie in the parameter space (excluding the identification condition).

**Value**

a vector containing the approximate standard errors of the estimates

## References

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

---

swap\_parametrization    *Swap the parametrization of a GMVAR model*

---

## Description

swap\_parametrization swaps the parametrization of a GMVAR model to "mean" if the current parametrization is "intercept", and vice versa.

## Usage

```
swap_parametrization(gmvar)
```

## Arguments

gmvar                    an object of class 'gmvar' created with fitGMVAR or GMVAR.

## 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 'gmvar' defining the specified GMVAR model. Can be used to work with other functions provided in gmvarKit.

Remark 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*.

## See Also

[fitGMVAR](#), [GMVAR](#), [iterate\\_more](#)



**Examples**

```

# These examples use the data 'eurusd' which comes with the
# package, but in a scaled form.
data <- cbind(10*eurusd[,1], 100*eurusd[,2])
colnames(data) <- colnames(eurusd)

# GMVAR(1,2), d=2 model:
params122 <- c(0.623, -0.129, 0.959, 0.089, -0.006, 1.006, 1.746,
  0.804, 5.804, 3.245, 7.913, 0.952, -0.037, -0.019, 0.943, 6.926,
  3.982, 12.135, 0.789)
mod122 <- GMVAR(data, p=1, M=2, params=params122)
mod122 # intercept-parametrization

mod122_2 <- swap_parametrization(mod122)
mod122_2 # mean-parametrization

# GMVAR(2,2), d=2 model:
params222 <- c(-11.904, 154.684, 1.314, 0.145, 0.094, 1.292, -0.389,
  -0.070, -0.109, -0.281, 0.920, -0.025, 4.839, 11.633, 124.983, 1.248,
  0.077, -0.040, 1.266, -0.272, -0.074, 0.034, -0.313, 5.855, 3.570,
  9.838, 0.740)
mod222 <- GMVAR(data, p=2, M=2, params=params222, parametrization="mean")
mod222 # mean-parametrization

mod222_2 <- swap_parametrization(mod222)
mod222_2 # intercept-parametrization

```

---

uncond_moments	<i>Calculate the unconditional mean, variance, the first p autocovariances, and the first p autocorrelations of a GMVAR process</i>
----------------	---

---

**Description**

uncond\_moments calculates the unconditional mean, variance, the first p autocovariances, and the first p autocorrelations of the given GMVAR process.

**Usage**

```
uncond_moments(gmvar)
```

**Arguments**

gmvar            an object of class 'gmvar' created with fitGMVAR or GMVAR.

**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  $0, 1, \dots, p$  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.
- Lutkepohl H. 2005. *New Introduction to Multiple Time Series Analysis*, Springer.

**See Also**

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

**Examples**

```
# GMVAR(1,2), d=2 model:
params122 <- c(0.623, -0.129, 0.959, 0.089, -0.006, 1.006, 1.746,
  0.804, 5.804, 3.245, 7.913, 0.952, -0.037, -0.019, 0.943, 6.926,
  3.982, 12.135, 0.789)
mod122 <- GMVAR(p=1, M=2, d=2, params=params122)
uncond_moments(mod122)

# GMVAR(2,2), d=2 model with AR-parameters restricted to be
# the same for both regimes:
C_mat <- rbind(diag(2*2^2), diag(2*2^2))
params222c <- c(1.031, 2.356, 1.786, 3.000, 1.250, 0.060, 0.036,
  1.335, -0.290, -0.083, -0.047, -0.356, 0.934, -0.152, 5.201, 5.883,
  3.560, 9.799, 0.368)
mod222c <- GMVAR(p=2, M=2, d=2, params=params222c, constraints=C_mat)
uncond_moments(mod222c)
```

---

<code>uncond_moments_int</code>	<i>Calculate the unconditional mean, variance, the first <math>p</math> autocovariances, and the first <math>p</math> autocorrelations of a GMVAR process</i>
---------------------------------	---

---

**Description**

`uncond_moments_int` calculates the unconditional mean, variance, the first  $p$  autocovariances, and the first  $p$  autocorrelations of the specified GMVAR process.

**Usage**

```
uncond_moments_int(
  p,
  M,
  d,
  params,
  parametrization = c("intercept", "mean"),
  constraints = NULL
)
```

**Arguments**

- p a positive integer specifying the autoregressive order of the model.
- M a positive integer specifying the number of mixture components.
- d the number of time series in the system.
- params a real valued vector specifying the parameter values.

**For unconstrained models:** Should be size  $((M(pd^2 + d + d(d + 1)/2 + 1) - 1)x1)$  and have form  $\theta=(v_1, \dots, v_M, \alpha_1, \dots, \alpha_{M-1})$ , where:

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

**For constrained models:** Should be size  $((M(d + d(d + 1)/2 + 1) + q - 1)x1)$  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  $(Mpd^2qx)$  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. 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. The notation is in line with the cited article by *Kalliovirta, Meitz and Saikkonen (2016)* introducing the GMVAR model.

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

constraints a size  $(Mpd^2qx)$  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 constrained 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.

**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  $0, 1, \dots, p$  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.
- Lutkepohl H. 2005. New Introduction to Multiple Time Series Analysis, *Springer*.

---

 unvec

*Reverse vectorization operator*


---

**Description**

unvec forms a square matrix from a vector of stacked columns, stacked by vec.

**Usage**

unvec(d, a)

**Arguments**

d the number of rows in the square matrix to be formed.

a a size  $(d^2 \times 1)$  vector to be unvectorized into a  $(d \times d)$  matrix.

**Value**

a matrix of size  $(d \times d)$ .

**Warning**

No argument checks!

---

unvech	<i>Reverse operator of the parsimonious vectorization operator vech</i>
--------	---

---

**Description**

unvech creates a symmetric matrix from the given vector by copying the lower triangular part to be the upper triangular part as well.

**Usage**

```
unvech(d, a)
```

**Arguments**

d                    number of rows the square matrix to be formed.  
a                    a size  $(d(d+1)/2 \times 1)$  vector to be unvectorized into a symmetric  $(d \times d)$  matrix.

**Value**

a symmetric matrix of size  $(d \times d)$ .

**Warning**

No argument checks!

---

vec	<i>Vectorization operator</i>
-----	-------------------------------

---

**Description**

vec stacks columns of the given matrix to form a vector.

**Usage**

```
vec(A)
```

**Arguments**

A                    a size  $(d \times d)$  square matrix to be vectorized.

**Value**

a vector of size  $(d^2 \times 1)$ .

**Warning**

No argument checks!

---

vech

*Parsimonious vectorization operator for symmetric matrices*

---

**Description**

vech stacks columns of the given matrix from the principal diagonal downwards (including elements on the diagonal) to form a vector.

**Usage**

vech(A)

**Arguments**

A a size  $(d \times d)$  symmetric matrix to be vectorized parsimoniously.

**Value**

a vector of size  $(d(d + 1)/2 \times 1)$ .

**Warning**

No argument checks!

# Index

## \*Topic **datasets**

eurusd, 19

acf, 18

add\_data, 3, 40

all\_pos\_ints, 4

alt\_gmvar, 5

calc\_gradient, 6, 48

calc\_hessian (calc\_gradient), 6

change\_parametrization, 8

change\_regime, 10

check\_constraints, 11

check\_data, 12

check\_gmvar, 12

check\_null\_data, 13

check\_parameters, 13

check\_pMd, 15

cond\_moments, 15, 32, 34, 90

density, 18

diagnostic\_plot, 17, 62, 69, 70

dlogmultinorm, 19

eurusd, 19

fitGMVAR, 4, 6, 18, 20, 30, 40, 46, 48, 62, 67, 69, 70, 88

form\_boldA, 24

format\_valuef, 24

GAFit, 25, 30

get\_boldA\_eigens, 28

get\_foc (calc\_gradient), 6

get\_gradient, 23

get\_gradient (calc\_gradient), 6

get\_hessian (calc\_gradient), 6

get\_IC, 29

get\_minval, 30

get\_omega\_eigens, 30

get\_regime\_autocovs, 17, 31, 34, 90

get\_regime\_autocovs\_int, 32

get\_regime\_means, 17, 32, 33, 90

get\_regime\_means\_int, 34

get\_soc, 69

get\_soc (calc\_gradient), 6

get\_test\_Omega, 36

GMVAR, 4, 6, 18, 23, 38, 46, 48, 62, 67, 69, 70, 88

gmvarkit, 41

in\_paramspace, 41

in\_paramspace\_int, 43

is\_stationary, 44

iterate\_more, 4, 6, 23, 45, 88

logLik.gmvar (GMVAR), 38

loglikelihood, 46

loglikelihood\_int, 49

n\_params, 51

optim, 46

pick\_all\_phi0\_A, 53

pick\_allA, 52

pick\_alphas, 54

pick\_Am, 55

pick\_Ami, 56

pick\_Omegas, 57

pick\_phi0, 58

pick\_regime, 59

plot.gmvar (GMVAR), 38

plot.gmvarpred, 60

plot.qrtest, 61

predict.gmvar, 18, 23, 62, 63, 70

print.gmvar, 67

print.gmvar (GMVAR), 38

print.gmvarpred, 65

print.gmvarsum, 65

print.qrtest (plot.qrtest), 61

print\_std\_errors, 23, 66

profile\_logliks, [7](#), [18](#), [23](#), [46](#), [62](#), [67](#), [68](#), [70](#)

quantile\_residual\_tests, [18](#), [23](#), [70](#)

quantile\_residual\_tests (plot.qrtest),  
[61](#)

quantile\_residuals, [62](#), [70](#)

quantile\_residuals\_int, [71](#)

random\_coefmats, [73](#)

random\_coefmats2, [74](#)

random\_covmat, [74](#)

random\_ind, [75](#)

random\_ind2, [76](#)

reform\_constrained\_pars, [77](#)

reform\_data, [79](#)

regime\_distance, [79](#)

residuals.gmvar (GMVAR), [38](#)

simulateGMVAR, [23](#), [80](#)

smart\_covmat, [82](#)

smart\_ind, [83](#)

sort\_components, [85](#)

standard\_errors, [86](#)

summary.gmvar (GMVAR), [38](#)

swap\_parametrization, [23](#), [40](#), [67](#), [88](#)

uncond\_moments, [17](#), [32](#), [34](#), [89](#)

uncond\_moments\_int, [90](#)

unvec, [92](#)

unvech, [93](#)

vec, [93](#)

vech, [94](#)