Package ‘fitDTVARMx’

August 25, 2024

Title Fit the Discrete-Time Vector Autoregressive Model

Version 0.0.0.9000

Description Fit the discrete-time vector autoregressive model using the 'OpenMx' package.

URL https://github.com/jeksterslab/fitDTVARMx,
https://jeksterslab.github.io/fitDTVARMx/

BugReports https://github.com/jeksterslab/fitDTVARMx/issues

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

VignetteBuilder knitr

Depends R (>= 3.0.0), OpenMx

Imports stats

Suggests knitr, rmarkdown, testthat, simStateSpace

RoxygenNote 7.3.2

NeedsCompilation no

Author Ivan Jacob Agaloos Pesigan [aut, cre, cph]
(<https://orcid.org/0000-0003-4818-8420>)

Maintainer Ivan Jacob Agaloos Pesigan <r.jeksterslab@gmail.com>

Contents

coef.fitdtvaridmx *Parameter Estimates*

Description

Parameter Estimates

Usage

```
## S3 method for class 'fitdtvaridmx'
coef(object, alpha = FALSE, psi = FALSE, theta = FALSE, ...)
```

Arguments

`object`	Object of class `fitdtvaridmx`.
`alpha`	Logical. If `alpha = TRUE`, include estimates of the `alpha` vector, if available. If `alpha = FALSE`, exclude estimates of the `alpha` vector.
`psi`	Logical. If `psi = TRUE`, include estimates of the `psi` matrix, if available. If `psi = FALSE`, exclude estimates of the `psi` matrix.
`theta`	Logical. If `theta = TRUE`, include estimates of the `theta` matrix, if available. If `theta = FALSE`, exclude estimates of the `theta` matrix.
`...`	additional arguments.

Value

Returns a list of vectors of parameter estimates.

Author(s)

Ivan Jacob Agaloos Pesigan

coef.fitdtvarmx *Parameter Estimates*

Description

Parameter Estimates

Usage

```
## S3 method for class 'fitdtvarmx'
coef(object, alpha = FALSE, psi = FALSE, theta = FALSE, ...)
```

Arguments

`object`	Object of class `fitdtvarmx`.
`alpha`	Logical. If `alpha = TRUE`, include estimates of the `alpha` vector, if available. If `alpha = FALSE`, exclude estimates of the `alpha` vector.
`psi`	Logical. If `psi = TRUE`, include estimates of the `psi` matrix, if available. If `psi = FALSE`, exclude estimates of the `psi` matrix.
`theta`	Logical. If `theta = TRUE`, include estimates of the `theta` matrix, if available. If `theta = FALSE`, exclude estimates of the `theta` matrix.
`...`	additional arguments.

Value

Returns a vector of parameter estimates.

Author(s)

Ivan Jacob Agaloos Pesigan

DTVAR — *Fit the First-Order Discrete-Time Vector Autoregressive Model*

Description

Fit the First-Order Discrete-Time Vector Autoregressive Model

Usage

```
DTVAR(
  data,
  observed,
  id,
  byid = FALSE,
  model = 1,
  try = 1000,
  ncores = NULL,
  ...
)
```

Arguments

`data`	Data frame. A data frame object of data for potentially multiple subjects that contain a column of subject ID numbers (i.e., an ID variable), and at least one column of observed values.
`observed`	Character vector. A vector of character strings of the names of the observed variables in the data.

`id`	Character string. A character string of the name of the ID variable in the data.
`byid`	Logical. If `byid = TRUE`, fit the model by `id`.
`model`	Model number (1, 2, 3, or 4). See Details for model description.
`try`	Positive integer. Number of extra optimization tries.
`ncores`	Positive integer. Number of cores to use.
`...`	Additional optional arguments to pass to `mxTryHardctsem`.

Details

Note that the mean and covariance matrix of the initial condition are fixed to a null vector and an identity matrix, respectively. The `DTVAR()` function fits four versions of the first-order discrete-time vector autoregressive model. Use the `FitDTVARIDMx()` or `FitDTVARMx()` functions to have more control over the model specification.

Model 1:

The measurement model is given by

$$\mathbf{y}_{i,t} = \boldsymbol{\eta}_{i,t}$$

where $\mathbf{y}_{i,t}$ represents a vector of observed variables and $\boldsymbol{\eta}_{i,t}$ a vector of latent variables for individual i and time t.

The dynamic structure is given by

$$\boldsymbol{\eta}_{i,t} = \boldsymbol{\beta}\boldsymbol{\eta}_{i,t-1} + \boldsymbol{\zeta}_{i,t}, \quad \text{with} \quad \boldsymbol{\zeta}_{i,t} \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Psi}\right)$$

where $\boldsymbol{\eta}_{i,t}$, $\boldsymbol{\eta}_{i,t-1}$, and $\boldsymbol{\zeta}_{i,t}$ are random variables, and $\boldsymbol{\beta}$, and $\boldsymbol{\Psi}$ are model parameters. Here, $\boldsymbol{\eta}_{i,t}$ is a vector of latent variables at time t and individual i, $\boldsymbol{\eta}_{i,t-1}$ represents a vector of latent variables at time $t-1$ and individual i, and $\boldsymbol{\zeta}_{i,t}$ represents a vector of dynamic noise at time t and individual i. $\boldsymbol{\beta}$ denotes a matrix of autoregression and cross regression coefficients, and $\boldsymbol{\Psi}$ the covariance matrix of $\boldsymbol{\zeta}_{i,t}$. In this model, $\boldsymbol{\Psi}$ is a diagonal matrix.

Model 2:

The measurement model is given by

$$\mathbf{y}_{i,t} = \boldsymbol{\Lambda}\boldsymbol{\eta}_{i,t} + \boldsymbol{\varepsilon}_{i,t}, \quad \text{with} \quad \boldsymbol{\varepsilon}_{i,t} \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Theta}\right)$$

where $\mathbf{y}_{i,t}$, $\boldsymbol{\eta}_{i,t}$, and $\boldsymbol{\varepsilon}_{i,t}$ are random variables and $\boldsymbol{\Lambda}$, and $\boldsymbol{\Theta}$ are model parameters. $\mathbf{y}_{i,t}$ represents a vector of observed random variables, $\boldsymbol{\eta}_{i,t}$ a vector of latent random variables, and $\boldsymbol{\varepsilon}_{i,t}$ a vector of random measurement errors, at time t and individual i. $\boldsymbol{\Lambda}$ denotes a matrix of factor loadings, and $\boldsymbol{\Theta}$ the covariance matrix of $\boldsymbol{\varepsilon}$. In this model, $\boldsymbol{\Lambda}$ is an identity matrix and $\boldsymbol{\Theta}$ is a diagonal matrix.

The dynamic structure is given by

$$\boldsymbol{\eta}_{i,t} = \boldsymbol{\beta}\boldsymbol{\eta}_{i,t-1} + \boldsymbol{\zeta}_{i,t}, \quad \text{with} \quad \boldsymbol{\zeta}_{i,t} \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Psi}\right)$$

where $\boldsymbol{\eta}_{i,t}$, $\boldsymbol{\eta}_{i,t-1}$, and $\boldsymbol{\zeta}_{i,t}$ are random variables, and $\boldsymbol{\beta}$, and $\boldsymbol{\Psi}$ are model parameters. Here, $\boldsymbol{\eta}_{i,t}$ is a vector of latent variables at time t and individual i, $\boldsymbol{\eta}_{i,t-1}$ represents a vector of latent variables at time $t-1$ and individual i, and $\boldsymbol{\zeta}_{i,t}$ represents a vector of dynamic noise at time t and individual i. $\boldsymbol{\beta}$ denotes a matrix of autoregression and cross regression coefficients, and $\boldsymbol{\Psi}$ the covariance matrix of $\boldsymbol{\zeta}_{i,t}$. In this model, $\boldsymbol{\Psi}$ is a diagonal matrix.

Model 3:

The measurement model is given by

$$\mathbf{y}_{i,t} = \boldsymbol{\Lambda} \boldsymbol{\eta}_{i,t} + \boldsymbol{\varepsilon}_{i,t}, \quad \text{with} \quad \boldsymbol{\varepsilon}_{i,t} \sim \mathcal{N} \left(\mathbf{0}, \boldsymbol{\Theta} \right)$$

where $\mathbf{y}_{i,t}$, $\boldsymbol{\eta}_{i,t}$, and $\boldsymbol{\varepsilon}_{i,t}$ are random variables and $\boldsymbol{\Lambda}$, and $\boldsymbol{\Theta}$ are model parameters. $\mathbf{y}_{i,t}$ represents a vector of observed random variables, $\boldsymbol{\eta}_{i,t}$ a vector of latent random variables, and $\boldsymbol{\varepsilon}_{i,t}$ a vector of random measurement errors, at time t and individual i. $\boldsymbol{\Lambda}$ denotes a matrix of factor loadings, and $\boldsymbol{\Theta}$ the covariance matrix of $\boldsymbol{\varepsilon}$. In this model, $\boldsymbol{\Lambda}$ is an identity matrix and $\boldsymbol{\Theta}$ is a diagonal matrix.

The dynamic structure is given by

$$\boldsymbol{\eta}_{i,t} = \boldsymbol{\alpha} + \boldsymbol{\beta} \boldsymbol{\eta}_{i,t-1} + \boldsymbol{\zeta}_{i,t}, \quad \text{with} \quad \boldsymbol{\zeta}_{i,t} \sim \mathcal{N} \left(\mathbf{0}, \boldsymbol{\Psi} \right)$$

where $\boldsymbol{\eta}_{i,t}$, $\boldsymbol{\eta}_{i,t-1}$, and $\boldsymbol{\zeta}_{i,t}$ are random variables, and $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and $\boldsymbol{\Psi}$ are model parameters. Here, $\boldsymbol{\eta}_{i,t}$ is a vector of latent variables at time t and individual i, $\boldsymbol{\eta}_{i,t-1}$ represents a vector of latent variables at time $t-1$ and individual i, and $\boldsymbol{\zeta}_{i,t}$ represents a vector of dynamic noise at time t and individual i. $\boldsymbol{\alpha}$ denotes a vector of intercepts, $\boldsymbol{\beta}$ a matrix of autoregression and cross regression coefficients, and $\boldsymbol{\Psi}$ the covariance matrix of $\boldsymbol{\zeta}_{i,t}$. In this model, $\boldsymbol{\Psi}$ is a diagonal matrix.

Model 4:

Model 4 is similar to Model 3 except that $\boldsymbol{\Psi}$ is a symmetric matrix in Model 4.

Value

Returns an object of class `fitdtvaridmx` if `byid = TRUE` or `fitdtvarmx` if `byid = FALSE`. The returned object is a list with the following elements:

call Function call.

args List of function arguments.

fun Function used ("FitDTVARIDMx" if `byid = TRUE` or "FitDTVARMx" if `byid = FALSE`).

output A list of fitted OpenMx models `byid = TRUE` or a single fitted OpenMx model if `byid = FALSE`.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Hunter, M. D. (2017). State space modeling in an open source, modular, structural equation modeling environment. *Structural Equation Modeling: A Multidisciplinary Journal*, *25*(2), 307–324. doi:10.1080/10705511.2017.1369354

Neale, M. C., Hunter, M. D., Pritikin, J. N., Zahery, M., Brick, T. R., Kirkpatrick, R. M., Estabrook, R., Bates, T. C., Maes, H. H., & Boker, S. M. (2015). OpenMx 2.0: Extended structural equation and statistical modeling. *Psychometrika*, *81*(2), 535–549. doi:10.1007/s1133601494358

See Also

Other DTVAR Functions: `FitDTVARIDMx()`, `FitDTVARMx()`

Examples

```
## Not run:
# ------------------------------------------------------------------------------
# byid = FALSE
# ------------------------------------------------------------------------------
# Generate data using the simStateSpace package------------------------------
set.seed(42)
sim <- simStateSpace::SimSSMVARFixed(
  n = 5,
  time = 100,
  mu0 = rep(x = 0, times = 3),
  sigma0_l = t(chol(diag(3))),
  alpha = rep(x = 0, times = 3),
  beta = matrix(
    data = c(
      0.7, 0.5, -0.1,
      0.0, 0.6, 0.4,
      0, 0, 0.5
    ),
    nrow = 3
  ),
  psi_l = t(chol(diag(3)))
)
data <- as.data.frame(sim)

# Fit the model------------------------------------------------------------------
library(fitDTVARMx)
fit <- DTVAR(
  data = data,
  observed = c("y1", "y2", "y3"),
  id = "id",
  byid = FALSE
)
print(fit)
summary(fit)
coef(fit)
vcov(fit)

# ------------------------------------------------------------------------------
# byid = TRUE
# ------------------------------------------------------------------------------
# Generate data using the simStateSpace package------------------------------
set.seed(42)
beta_mu <- matrix(
  data = c(
    0.7, 0.5, -0.1,
    0.0, 0.6, 0.4,
    0, 0, 0.5
  ),
  nrow = 3
)
beta_sigma <- diag(3 * 3)
```

```
beta <- simStateSpace::SimBetaN(
  n = 5,
  beta = beta_mu,
  vcov_beta_vec_l = t(chol(beta_sigma))
)
sim <- simStateSpace::SimSSMVARIVary(
  n = 5,
  time = 100,
  mu0 = list(rep(x = 0, times = 3)),
  sigma0_l = list(t(chol(diag(3)))),
  alpha = list(rep(x = 0, times = 3)),
  beta = beta,
  psi_l = list(t(chol(diag(3))))
)
data <- as.data.frame(sim)

# Fit the model--------------------------------------------------------------
library(fitDTVARMx)
fit <- DTVAR(
  data = data,
  observed = c("y1", "y2", "y3"),
  id = "id",
  byid = TRUE
)
print(fit)
summary(fit)
coef(fit)
vcov(fit)

## End(Not run)
```

FitDTVARIDMx *Fit the First-Order Discrete-Time Vector Autoregressive Model by ID*

Description

Fit the First-Order Discrete-Time Vector Autoregressive Model by ID

Usage

```
FitDTVARIDMx(
  data,
  observed,
  id,
  alpha_fixed = TRUE,
  alpha_values = NULL,
  alpha_free = NULL,
  alpha_lbound = NULL,
```

```
  alpha_ubound = NULL,
  beta_values = NULL,
  beta_free = NULL,
  beta_lbound = NULL,
  beta_ubound = NULL,
  psi_diag = TRUE,
  psi_values = NULL,
  psi_free = NULL,
  psi_lbound = NULL,
  psi_ubound = NULL,
  theta_fixed = TRUE,
  theta_values = NULL,
  theta_free = NULL,
  theta_lbound = NULL,
  theta_ubound = NULL,
  mu0_fixed = TRUE,
  mu0_values = NULL,
  mu0_free = NULL,
  mu0_lbound = NULL,
  mu0_ubound = NULL,
  sigma0_fixed = TRUE,
  sigma0_diag = TRUE,
  sigma0_values = NULL,
  sigma0_free = NULL,
  sigma0_lbound = NULL,
  sigma0_ubound = NULL,
  try = 1000,
  ncores = NULL,
  ...
)
```

Arguments

`data`	Data frame. A data frame object of data for potentially multiple subjects that contain a column of subject ID numbers (i.e., an ID variable), and at least one column of observed values.
`observed`	Character vector. A vector of character strings of the names of the observed variables in the data.
`id`	Character string. A character string of the name of the ID variable in the data.
`alpha_fixed`	Logical. If `alpha_fixed = TRUE`, the dynamic model intercept vector `alpha` is fixed at zero. If `alpha_fixed = FALSE`, the dynamic model intercept vector `alpha` is estimated.
`alpha_values`	Optional starting values for `alpha`. If `alpha_fixed = TRUE`, `alpha_values` will be used as fixed values. If `alpha_fixed = FALSE`, `alpha_values` will be used as starting values.
`alpha_free`	Optional logical vector representing free parameters for `alpha`.
`alpha_lbound`	Optional lower bound for `alpha`. Ignored if `alpha_fixed = TRUE`.

alpha_ubound	Optional upper bound for alpha. Ignored if alpha_fixed = TRUE.
beta_values	Numeric matrix. Optional starting values for beta.
beta_free	Optional logical matrix representing free parameters for beta.
beta_lbound	Numeric matrix. Optional lower bound for beta.
beta_ubound	Numeric matrix. Optional upper bound for beta.
psi_diag	Logical. If psi_diag = TRUE, psi is a diagonal matrix.
psi_values	Numeric matrix. Optional starting values for psi.
psi_free	Optional logical matrix representing free parameters for psi.
psi_lbound	Numeric matrix. Optional lower bound for psi.
psi_ubound	Optional upper bound for psi.
theta_fixed	Logical. If theta_fixed = TRUE, the measurement error matrix theta is fixed to zero. If theta_fixed = FALSE, estimate the diagonal measurement error matrix theta.
theta_values	Optional starting values for theta. Ignored if theta_fixed = TRUE.
theta_free	Optional logical matrix representing free parameters for theta.
theta_lbound	Optional lower bound for theta. Ignored if theta_fixed = TRUE.
theta_ubound	Optional upper bound for theta. Ignored if theta_fixed = TRUE.
mu0_fixed	Logical. If mu0_fixed = TRUE, initial mean vector mu0 is fixed. If mu0_fixed = FALSE, initial mean vector mu0 is estimated.
mu0_values	Optional starting values for mu0. If mu0_fixed = TRUE, mu0_values will be used as fixed values. If mu0_fixed = FALSE, mu0_values will be used as starting values.
mu0_free	Optional logical vector representing free parameters for mu0.
mu0_lbound	Optional lower bound for mu0. Ignored if mu0_fixed = TRUE.
mu0_ubound	Optional upper bound for mu0. Ignored if mu0_fixed = TRUE.
sigma0_fixed	Logical. If sigma0_fixed = TRUE, initial mean vector sigma0 is fixed. If sigma0_fixed = FALSE, initial mean vector sigma0 is estimated.
sigma0_diag	Logical. If sigma0_diag = TRUE, sigma0 is a diagonal matrix.
sigma0_values	Optional starting values for sigma0. If sigma0_fixed = TRUE, sigma0_values will be used as fixed values. If sigma0_fixed = FALSE, sigma0_values will be used as starting values.
sigma0_free	Optional logical matrix representing free parameters for sigma0.
sigma0_lbound	Optional lower bound for sigma0. Ignored if sigma0_fixed = TRUE.
sigma0_ubound	Optional upper bound for sigma0. Ignored if sigma0_fixed = TRUE.
try	Positive integer. Number of extra optimization tries.
ncores	Positive integer. Number of cores to use.
...	Additional optional arguments to pass to mxTryHardctsem.

Value

Returns an object of class `fitdtvaridmx` which is a list with the following elements:

call Function call.

args List of function arguments.

fun Function used ("FitDTVARIDMx").

output A list of fitted OpenMx models.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Hunter, M. D. (2017). State space modeling in an open source, modular, structural equation modeling environment. *Structural Equation Modeling: A Multidisciplinary Journal*, *25*(2), 307–324. doi:10.1080/10705511.2017.1369354

Neale, M. C., Hunter, M. D., Pritikin, J. N., Zahery, M., Brick, T. R., Kirkpatrick, R. M., Estabrook, R., Bates, T. C., Maes, H. H., & Boker, S. M. (2015). OpenMx 2.0: Extended structural equation and statistical modeling. *Psychometrika*, *81*(2), 535–549. doi:10.1007/s1133601494358

See Also

Other DTVAR Functions: `DTVAR()`, `FitDTVARMx()`

Examples

```
## Not run:
# Generate data using the simStateSpace package------------------------------
set.seed(42)
beta_mu <- matrix(
  data = c(
    0.7, 0.5, -0.1,
    0.0, 0.6, 0.4,
    0, 0, 0.5
  ),
  nrow = 3
)
beta_sigma <- diag(3 * 3)
beta <- simStateSpace::SimBetaN(
  n = 5,
  beta = beta_mu,
  vcov_beta_vec_l = t(chol(beta_sigma))
)
sim <- simStateSpace::SimSSMVARIVary(
  n = 5,
  time = 100,
  mu0 = list(rep(x = 0, times = 3)),
  sigma0_l = list(t(chol(diag(3)))),
  alpha = list(rep(x = 0, times = 3)),
```

```
  beta = beta,
  psi_l = list(t(chol(diag(3))))
)
data <- as.data.frame(sim)

# Fit the model--------------------------------------------------------------
library(fitDTVARMx)
fit <- FitDTVARIDMx(
  data = data,
  observed = c("y1", "y2", "y3"),
  id = "id"
)
print(fit)
summary(fit)
coef(fit)
vcov(fit)

## End(Not run)
```

FitDTVARMx *Fit the First-Order Discrete-Time Vector Autoregressive Model*

Description

Fit the First-Order Discrete-Time Vector Autoregressive Model

Usage

```
FitDTVARMx(
  data,
  observed,
  id,
  alpha_fixed = TRUE,
  alpha_values = NULL,
  alpha_free = NULL,
  alpha_lbound = NULL,
  alpha_ubound = NULL,
  beta_values = NULL,
  beta_free = NULL,
  beta_lbound = NULL,
  beta_ubound = NULL,
  psi_diag = TRUE,
  psi_values = NULL,
  psi_free = NULL,
  psi_lbound = NULL,
  psi_ubound = NULL,
  theta_fixed = TRUE,
```

```
  theta_values = NULL,
  theta_free = NULL,
  theta_lbound = NULL,
  theta_ubound = NULL,
  mu0_fixed = TRUE,
  mu0_values = NULL,
  mu0_free = NULL,
  mu0_lbound = NULL,
  mu0_ubound = NULL,
  sigma0_fixed = TRUE,
  sigma0_diag = TRUE,
  sigma0_values = NULL,
  sigma0_free = NULL,
  sigma0_lbound = NULL,
  sigma0_ubound = NULL,
  try = 1000,
  ncores = NULL,
  ...
)
```

Arguments

`data`	Data frame. A data frame object of data for potentially multiple subjects that contain a column of subject ID numbers (i.e., an ID variable), and at least one column of observed values.
`observed`	Character vector. A vector of character strings of the names of the observed variables in the data.
`id`	Character string. A character string of the name of the ID variable in the data.
`alpha_fixed`	Logical. If `alpha_fixed = TRUE`, the dynamic model intercept vector `alpha` is fixed at zero. If `alpha_fixed = FALSE`, the dynamic model intercept vector `alpha` is estimated.
`alpha_values`	Optional starting values for `alpha`. If `alpha_fixed = TRUE`, `alpha_values` will be used as fixed values. If `alpha_fixed = FALSE`, `alpha_values` will be used as starting values.
`alpha_free`	Optional logical vector representing free parameters for `alpha`.
`alpha_lbound`	Optional lower bound for `alpha`. Ignored if `alpha_fixed = TRUE`.
`alpha_ubound`	Optional upper bound for `alpha`. Ignored if `alpha_fixed = TRUE`.
`beta_values`	Numeric matrix. Optional starting values for `beta`.
`beta_free`	Optional logical matrix representing free parameters for `beta`.
`beta_lbound`	Numeric matrix. Optional lower bound for `beta`.
`beta_ubound`	Numeric matrix. Optional upper bound for `beta`.
`psi_diag`	Logical. If `psi_diag = TRUE`, `psi` is a diagonal matrix.
`psi_values`	Numeric matrix. Optional starting values for `psi`.
`psi_free`	Optional logical matrix representing free parameters for `psi`.

psi_lbound	Numeric matrix. Optional lower bound for psi.
psi_ubound	Optional upper bound for psi.
theta_fixed	Logical. If theta_fixed = TRUE, the measurement error matrix theta is fixed to zero. If theta_fixed = FALSE, estimate the diagonal measurement error matrix theta.
theta_values	Optional starting values for theta. Ignored if theta_fixed = TRUE.
theta_free	Optional logical matrix representing free parameters for theta.
theta_lbound	Optional lower bound for theta. Ignored if theta_fixed = TRUE.
theta_ubound	Optional upper bound for theta. Ignored if theta_fixed = TRUE.
mu0_fixed	Logical. If mu0_fixed = TRUE, initial mean vector mu0 is fixed. If mu0_fixed = FALSE, initial mean vector mu0 is estimated.
mu0_values	Optional starting values for mu0. If mu0_fixed = TRUE, mu0_values will be used as fixed values. If mu0_fixed = FALSE, mu0_values will be used as starting values.
mu0_free	Optional logical vector representing free parameters for mu0.
mu0_lbound	Optional lower bound for mu0. Ignored if mu0_fixed = TRUE.
mu0_ubound	Optional upper bound for mu0. Ignored if mu0_fixed = TRUE.
sigma0_fixed	Logical. If sigma0_fixed = TRUE, initial mean vector sigma0 is fixed. If sigma0_fixed = FALSE, initial mean vector sigma0 is estimated.
sigma0_diag	Logical. If sigma0_diag = TRUE, sigma0 is a diagonal matrix.
sigma0_values	Optional starting values for sigma0. If sigma0_fixed = TRUE, sigma0_values will be used as fixed values. If sigma0_fixed = FALSE, sigma0_values will be used as starting values.
sigma0_free	Optional logical matrix representing free parameters for sigma0.
sigma0_lbound	Optional lower bound for sigma0. Ignored if sigma0_fixed = TRUE.
sigma0_ubound	Optional upper bound for sigma0. Ignored if sigma0_fixed = TRUE.
try	Positive integer. Number of extra optimization tries.
ncores	Positive integer. Number of cores to use.
...	Additional optional arguments to pass to mxTryHardctsem.

Value

Returns an object of class fitdtvarmx which is a list with the following elements:

call Function call.

args List of function arguments.

fun Function used ("FitDTVARMx").

output A fitted OpenMx model.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Hunter, M. D. (2017). State space modeling in an open source, modular, structural equation modeling environment. *Structural Equation Modeling: A Multidisciplinary Journal*, *25*(2), 307–324. doi:10.1080/10705511.2017.1369354

Neale, M. C., Hunter, M. D., Pritikin, J. N., Zahery, M., Brick, T. R., Kirkpatrick, R. M., Estabrook, R., Bates, T. C., Maes, H. H., & Boker, S. M. (2015). OpenMx 2.0: Extended structural equation and statistical modeling. *Psychometrika*, *81*(2), 535–549. doi:10.1007/s1133601494358

See Also

Other DTVAR Functions: `DTVAR()`, `FitDTVARIDMx()`

Examples

```
## Not run:
# Generate data using the simStateSpace package------------------------------
set.seed(42)
sim <- simStateSpace::SimSSMVARFixed(
  n = 5,
  time = 100,
  mu0 = rep(x = 0, times = 3),
  sigma0_l = t(chol(diag(3))),
  alpha = rep(x = 0, times = 3),
  beta = matrix(
    data = c(
      0.7, 0.5, -0.1,
      0.0, 0.6, 0.4,
      0, 0, 0.5
    ),
    nrow = 3
  ),
  psi_l = t(chol(diag(3)))
)
data <- as.data.frame(sim)

# Fit the model--------------------------------------------------------------
library(fitDTVARMx)
fit <- FitDTVARMx(
  data = data,
  observed = c("y1", "y2", "y3"),
  id = "id"
)
print(fit)
summary(fit)
coef(fit)
vcov(fit)

## End(Not run)
```

print.fitdtvaridmx *Print Method for Object of Class* `fitdtvaridmx`

Description

Print Method for Object of Class `fitdtvaridmx`

Usage

```
## S3 method for class 'fitdtvaridmx'
print(x, means = TRUE, ...)
```

Arguments

`x`	an object of class `fitdtvaridmx`.
`means`	Logical. If `means = TRUE`, return means. Otherwise, the function returns raw estimates.
`...`	further arguments.

Author(s)

Ivan Jacob Agaloos Pesigan

print.fitdtvarmx *Print Method for Object of Class* `fitdtvarmx`

Description

Print Method for Object of Class `fitdtvarmx`

Usage

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

Arguments

`x`	an object of class `fitdtvarmx`.
`...`	further arguments.

Author(s)

Ivan Jacob Agaloos Pesigan

`summary.fitdtvaridmx` *Summary Method for Object of Class* `fitdtvaridmx`

Description

Summary Method for Object of Class `fitdtvaridmx`

Usage

```
## S3 method for class 'fitdtvaridmx'
summary(object, means = TRUE, ...)
```

Arguments

`object`	an object of class `fitdtvaridmx`.
`means`	Logical. If `means = TRUE`, return means. Otherwise, the function returns raw estimates.
`...`	further arguments.

Author(s)

Ivan Jacob Agaloos Pesigan

`summary.fitdtvarmx` *Summary Method for Object of Class* `fitdtvarmx`

Description

Summary Method for Object of Class `fitdtvarmx`

Usage

```
## S3 method for class 'fitdtvarmx'
summary(object, ...)
```

Arguments

`object`	an object of class `fitdtvarmx`.
`...`	further arguments.

Author(s)

Ivan Jacob Agaloos Pesigan

vcov.fitdtvaridmx *Sampling Covariance Matrix of the Parameter Estimates*

Description

Sampling Covariance Matrix of the Parameter Estimates

Usage

```
## S3 method for class 'fitdtvaridmx'
vcov(object, alpha = FALSE, psi = FALSE, theta = FALSE, ...)
```

Arguments

`object`	Object of class `fitdtvaridmx`.
`alpha`	Logical. If `alpha = TRUE`, include estimates of the `alpha` vector, if available. If `alpha = FALSE`, exclude estimates of the `alpha` vector.
`psi`	Logical. If `psi = TRUE`, include estimates of the `psi` matrix, if available. If `psi = FALSE`, exclude estimates of the `psi` matrix.
`theta`	Logical. If `theta = TRUE`, include estimates of the `theta` matrix, if available. If `theta = FALSE`, exclude estimates of the `theta` matrix.
`...`	additional arguments.

Value

Returns a list of sampling variance-covariance matrices.

Author(s)

Ivan Jacob Agaloos Pesigan

vcov.fitdtvarmx *Sampling Covariance Matrix of the Parameter Estimates*

Description

Sampling Covariance Matrix of the Parameter Estimates

Usage

```
## S3 method for class 'fitdtvarmx'
vcov(object, alpha = FALSE, psi = FALSE, theta = FALSE, ...)
```

Arguments

`object`	Object of class `fitdtvarmx`.
`alpha`	Logical. If `alpha = TRUE`, include estimates of the `alpha` vector, if available. If `alpha = FALSE`, exclude estimates of the `alpha` vector.
`psi`	Logical. If `psi = TRUE`, include estimates of the `psi` matrix, if available. If `psi = FALSE`, exclude estimates of the `psi` matrix.
`theta`	Logical. If `theta = TRUE`, include estimates of the `theta` matrix, if available. If `theta = FALSE`, exclude estimates of the `theta` matrix.
`...`	additional arguments.

Value

Returns a list of sampling variance-covariance matrices.

Author(s)

Ivan Jacob Agaloos Pesigan

Index