Package ‘bootStateSpace’

January 18, 2025

Title Bootstrap for State Space Models

Version 0.0.0.9000

Description Provides a streamlined and user-friendly framework
for bootstrapping in state space models,
particularly when the number of subjects/units (n) exceeds one,
a scenario commonly encountered in social and behavioral sciences.
For an introduction to state space models in social and behavioral sciences,
refer to Chow, Ho, Hamaker, and Dolan (2010) <doi:10.1080/10705511003661553>.

URL `https://github.com/jeksterslab/bootStateSpace`,
`https://jeksterslab.github.io/bootStateSpace/`

BugReports `https://github.com/jeksterslab/bootStateSpace/issues`

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 3.5.0)

Imports stats, dynr, simStateSpace

Suggests knitr, rmarkdown, testthat, expm

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.bootstatespace` *Estimated Parameter Method for an Object of Class* `bootstatespace`

Description

Estimated Parameter Method for an Object of Class `bootstatespace`

Usage

```
## S3 method for class 'bootstatespace'
coef(object, ...)
```

Arguments

`object`	Object of Class `bootstatespace`.
`...`	additional arguments.

Value

Returns a vector of estimated parameters.

Author(s)

Ivan Jacob Agaloos Pesigan

`confint.bootstatespace` *Confidence Intervals Method for an Object of Class* `bootstatespace`

Description

Confidence Intervals Method for an Object of Class `bootstatespace`

Usage

```
## S3 method for class 'bootstatespace'
confint(object, parm = NULL, level = 0.95, type = "pc", ...)
```

Arguments

`object`	Object of Class `bootstatespace`.
`parm`	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	the confidence level required.
`type`	Charater string. Confidence interval type, that is, `type = "pc"` for percentile; `type = "bc"` for bias corrected.
`...`	additional arguments.

Value

Returns a matrix of confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

`extract` *Extract Generic Function*

Description

A generic function for extracting elements from objects.

Usage

```
extract(object, what)
```

Arguments

`object`	An object.
`what`	Character string.

Value

A value determined by the specific method for the object's class.

`extract.bootstatespace` *Extract Method for an Object of Class* `bootstatespace`

Description

Extract Method for an Object of Class `bootstatespace`

Usage

```
## S3 method for class 'bootstatespace'
extract(object, what = NULL)
```

Arguments

`object`	Object of Class `bootstatespace`.
`what`	Character string. What specific matrix to extract. If `what = NULL`, extract all available matrices.

Value

Returns a list. Each element of the list is a list of bootstrap estimates in matrix format.

Author(s)

Ivan Jacob Agaloos Pesigan

`PBSSMFixed` *Parametric Bootstrap for the State Space Model (Fixed Parameters)*

Description

This function simulates data from a state-space model and fits the model using the `dynr` package. The process is repeated `R` times. It assumes that the parameters remain constant across individuals and over time. At the moment, the function only supports `type = 0`.

Usage

```
PBSSMFixed(
  R,
  path,
  prefix,
  n,
  time,
  delta_t = 1,
```

```
  mu0,
  sigma0_l,
  alpha,
  beta,
  psi_l,
  nu,
  lambda,
  theta_l,
  type = 0,
  x = NULL,
  gamma = NULL,
  kappa = NULL,
  mu0_fixed = FALSE,
  sigma0_fixed = FALSE,
  alpha_level = 0.05,
  optimization_flag = TRUE,
  hessian_flag = FALSE,
  verbose = FALSE,
  weight_flag = FALSE,
  debug_flag = FALSE,
  perturb_flag = FALSE,
  xtol_rel = 1e-07,
  stopval = -9999,
  ftol_rel = -1,
  ftol_abs = -1,
  maxeval = as.integer(-1),
  maxtime = -1,
  ncores = NULL,
  seed = NULL
)
```

Arguments

`R`	Positive integer. Number of bootstrap samples.
`path`	Path to a directory to store bootstrap samples and estimates.
`prefix`	Character string. Prefix used for the file names for the bootstrap samples and estimates.
`n`	Positive integer. Number of individuals.
`time`	Positive integer. Number of time points.
`delta_t`	Numeric. Time interval. The default value is `1.0` with an option to use a numeric value for the discretized state space model parameterization of the linear stochastic differential equation model.
`mu0`	Numeric vector. Mean of initial latent variable values ($\boldsymbol{\mu}_{\boldsymbol{\eta} \mid 0}$).
`sigma0_l`	Numeric matrix. Cholesky factorization (`t(chol(sigma0))`) of the covariance matrix of initial latent variable values ($\boldsymbol{\Sigma}_{\boldsymbol{\eta} \mid 0}$).
`alpha`	Numeric vector. Vector of constant values for the dynamic model ($\boldsymbol{\alpha}$).

`beta`	Numeric matrix. Transition matrix relating the values of the latent variables at the previous to the current time point ($\boldsymbol{\beta}$).
`psi_l`	Numeric matrix. Cholesky factorization (`t(chol(psi))`) of the covariance matrix of the process noise ($\boldsymbol{\Psi}$).
`nu`	Numeric vector. Vector of intercept values for the measurement model ($\boldsymbol{\nu}$).
`lambda`	Numeric matrix. Factor loading matrix linking the latent variables to the observed variables ($\boldsymbol{\Lambda}$).
`theta_l`	Numeric matrix. Cholesky factorization (`t(chol(theta))`) of the covariance matrix of the measurement error ($\boldsymbol{\Theta}$).
`type`	Integer. State space model type. See Details for more information.
`x`	List. Each element of the list is a matrix of covariates for each individual `i` in `n`. The number of columns in each matrix should be equal to `time`.
`gamma`	Numeric matrix. Matrix linking the covariates to the latent variables at current time point ($\boldsymbol{\Gamma}$).
`kappa`	Numeric matrix. Matrix linking the covariates to the observed variables at current time point ($\boldsymbol{\kappa}$).
`mu0_fixed`	Logical. If `mu0_fixed = TRUE`, fix the initial mean vector to `mu0`. If `mu0_fixed = FALSE`, `mu0` is estimated.
`sigma0_fixed`	Logical. If `sigma0_fixed = TRUE`, fix the initial covariance matrix to `tcrossprod(sigma0_l)`. If `sigma0_fixed = FALSE`, `sigma0` is estimated.
`alpha_level`	Numeric vector. Significance level α.
`optimization_flag`	a flag (TRUE/FALSE) indicating whether optimization is to be done.
`hessian_flag`	a flag (TRUE/FALSE) indicating whether the Hessian matrix is to be calculated.
`verbose`	a flag (TRUE/FALSE) indicating whether more detailed intermediate output during the estimation process should be printed
`weight_flag`	a flag (TRUE/FALSE) indicating whether the negative log likelihood function should be weighted by the length of the time series for each individual
`debug_flag`	a flag (TRUE/FALSE) indicating whether users want additional dynr output that can be used for diagnostic purposes
`perturb_flag`	a flag (TRUE/FLASE) indicating whether to perturb the latent states during estimation. Only useful for ensemble forecasting.
`xtol_rel`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.
`stopval`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.
`ftol_rel`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.
`ftol_abs`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.
`maxeval`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.

maxtime	Stopping criteria option for parameter optimization. See dynr::dynr.model() for more details.
ncores	Positive integer. Number of cores to use. If ncores = NULL, use a single core. Consider using multiple cores when number of bootstrap samples R is a large value.
seed	Random seed.

Details

Type 0:

The measurement model is given by

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

where $\mathbf{y}_{i,t}$, $\boldsymbol{\eta}_{i,t}$, and $\boldsymbol{\varepsilon}_{i,t}$ are random variables and $\boldsymbol{\nu}$, $\boldsymbol{\Lambda}$, 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. $\boldsymbol{\nu}$ denotes a vector of intercepts, $\boldsymbol{\Lambda}$ a matrix of factor loadings, and $\boldsymbol{\Theta}$ the covariance matrix of $\boldsymbol{\varepsilon}$.

An alternative representation of the measurement error is given by

$$\boldsymbol{\varepsilon}_{i,t} = \boldsymbol{\Theta}^{\frac{1}{2}}\mathbf{z}_{i,t}, \quad \text{with} \quad \mathbf{z}_{i,t} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)$$

where $\mathbf{z}_{i,t}$ is a vector of independent standard normal random variables and $\left(\boldsymbol{\Theta}^{\frac{1}{2}}\right)\left(\boldsymbol{\Theta}^{\frac{1}{2}}\right)' = \boldsymbol{\Theta}$.

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 $\boldsymbol{\zeta}_{i,t}$.

An alternative representation of the dynamic noise is given by

$$\boldsymbol{\zeta}_{i,t} = \boldsymbol{\Psi}^{\frac{1}{2}}\mathbf{z}_{i,t}, \quad \text{with} \quad \mathbf{z}_{i,t} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)$$

where $\left(\boldsymbol{\Psi}^{\frac{1}{2}}\right)\left(\boldsymbol{\Psi}^{\frac{1}{2}}\right)' = \boldsymbol{\Psi}$.

Type 1:

The measurement model is given by

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

The dynamic structure is given by

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

where $\mathbf{x}_{i,t}$ represents a vector of covariates at time t and individual i, and $\boldsymbol{\Gamma}$ the coefficient matrix linking the covariates to the latent variables.

Type 2:
The measurement model is given by

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

where $\boldsymbol{\kappa}$ represents the coefficient matrix linking the covariates to the observed variables.
The dynamic structure is given by

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

Value

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

call Function call.

args Function arguments.

thetahatstar Sampling distribution of $\hat{\boldsymbol{\theta}}$.

vcov Sampling variance-covariance matrix of $\hat{\boldsymbol{\theta}}$.

est Vector of estimated $\hat{\boldsymbol{\theta}}$.

fun Function used ("PBSSMFixed").

method Bootstrap method used ("parametric").

Author(s)

Ivan Jacob Agaloos Pesigan

References

Chow, S.-M., Ho, M. R., Hamaker, E. L., & Dolan, C. V. (2010). Equivalence and differences between structural equation modeling and state-space modeling techniques. *Structural Equation Modeling: A Multidisciplinary Journal*, 17(2), 303–332. doi:10.1080/10705511003661553

See Also

Other Simulation of State Space Models Data Functions: `PBSSMLinSDEFixed()`, `PBSSMOUFixed()`, `PBSSMVARFixed()`

Examples

```
## Not run:
# prepare parameters
set.seed(42)
## number of individuals
n <- 5
## time points
time <- 50
## dynamic structure
p <- 3
mu0 <- rep(x = 0, times = p)
```

```
sigma0 <- 0.001 * diag(p)
sigma0_l <- t(chol(sigma0))
alpha <- rep(x = 0, times = p)
beta <- 0.50 * diag(p)
psi <- 0.001 * diag(p)
psi_l <- t(chol(psi))
## measurement model
k <- 3
nu <- rep(x = 0, times = k)
lambda <- diag(k)
theta <- 0.001 * diag(k)
theta_l <- t(chol(theta))

pb <- PBSSMFixed(
  R = 10L, # use at least 1000 in actual research
  path = getwd(),
  prefix = "ssm",
  n = n,
  time = time,
  delta_t = delta_t,
  mu0 = mu0,
  sigma0_l = sigma0_l,
  alpha = alpha,
  beta = beta,
  psi_l = psi_l,
  nu = nu,
  lambda = lambda,
  theta_l = theta_l,
  type = 0,
  ncores = 1, # consider using multiple cores
  seed = 42
)
print(pb)
summary(pb)
confint(pb)
vcov(pb)
coef(pb)
print(pb, type = "bc") # bias-corrected
summary(pb, type = "bc")
confint(pb, type = "bc")

## End(Not run)
```

`PBSSMLinSDEFixed` *Parametric Bootstrap for the Linear Stochastic Differential Equation Model using a State Space Model Parameterization (Fixed Parameters)*

Description

This function simulates data from a linear stochastic differential equation model using a state-space model parameterization and fits the model using the `dynr` package. The process is repeated `R` times. It assumes that the parameters remain constant across individuals and over time. At the moment, the function only supports `type = 0`.

Usage

```
PBSSMLinSDEFixed(
  R,
  path,
  prefix,
  n,
  time,
  delta_t = 0.1,
  mu0,
  sigma0_l,
  iota,
  phi,
  sigma_l,
  nu,
  lambda,
  theta_l,
  type = 0,
  x = NULL,
  gamma = NULL,
  kappa = NULL,
  mu0_fixed = FALSE,
  sigma0_fixed = FALSE,
  alpha_level = 0.05,
  optimization_flag = TRUE,
  hessian_flag = FALSE,
  verbose = FALSE,
  weight_flag = FALSE,
  debug_flag = FALSE,
  perturb_flag = FALSE,
  xtol_rel = 1e-07,
  stopval = -9999,
  ftol_rel = -1,
  ftol_abs = -1,
  maxeval = as.integer(-1),
  maxtime = -1,
  ncores = NULL,
  seed = NULL
)
```

Arguments

- `R` — Positive integer. Number of bootstrap samples.
- `path` — Path to a directory to store bootstrap samples and estimates.
- `prefix` — Character string. Prefix used for the file names for the bootstrap samples and estimates.
- `n` — Positive integer. Number of individuals.
- `time` — Positive integer. Number of time points.
- `delta_t` — Numeric. Time interval (Δ_t).
- `mu0` — Numeric vector. Mean of initial latent variable values ($\boldsymbol{\mu}_{\boldsymbol{\eta}|0}$).
- `sigma0_l` — Numeric matrix. Cholesky factorization (`t(chol(sigma0))`) of the covariance matrix of initial latent variable values ($\boldsymbol{\Sigma}_{\boldsymbol{\eta}|0}$).
- `iota` — Numeric vector. An unobserved term that is constant over time ($\boldsymbol{\iota}$).
- `phi` — Numeric matrix. The drift matrix which represents the rate of change of the solution in the absence of any random fluctuations ($\boldsymbol{\Phi}$).
- `sigma_l` — Numeric matrix. Cholesky factorization (`t(chol(sigma))`) of the covariance matrix of volatility or randomness in the process ($\boldsymbol{\Sigma}$).
- `nu` — Numeric vector. Vector of intercept values for the measurement model ($\boldsymbol{\nu}$).
- `lambda` — Numeric matrix. Factor loading matrix linking the latent variables to the observed variables ($\boldsymbol{\Lambda}$).
- `theta_l` — Numeric matrix. Cholesky factorization (`t(chol(theta))`) of the covariance matrix of the measurement error ($\boldsymbol{\Theta}$).
- `type` — Integer. State space model type. See Details for more information.
- `x` — List. Each element of the list is a matrix of covariates for each individual `i` in `n`. The number of columns in each matrix should be equal to `time`.
- `gamma` — Numeric matrix. Matrix linking the covariates to the latent variables at current time point ($\boldsymbol{\Gamma}$).
- `kappa` — Numeric matrix. Matrix linking the covariates to the observed variables at current time point ($\boldsymbol{\kappa}$).
- `mu0_fixed` — Logical. If `mu0_fixed = TRUE`, fix the initial mean vector to `mu0`. If `mu0_fixed = FALSE`, `mu0` is estimated.
- `sigma0_fixed` — Logical. If `sigma0_fixed = TRUE`, fix the initial covariance matrix to `tcrossprod(sigma0_l)`. If `sigma0_fixed = FALSE`, `sigma0` is estimated.
- `alpha_level` — Numeric vector. Significance level α.
- `optimization_flag` — a flag (TRUE/FALSE) indicating whether optimization is to be done.
- `hessian_flag` — a flag (TRUE/FALSE) indicating whether the Hessian matrix is to be calculated.
- `verbose` — a flag (TRUE/FALSE) indicating whether more detailed intermediate output during the estimation process should be printed
- `weight_flag` — a flag (TRUE/FALSE) indicating whether the negative log likelihood function should be weighted by the length of the time series for each individual

`debug_flag`	a flag (TRUE/FALSE) indicating whether users want additional dynr output that can be used for diagnostic purposes
`perturb_flag`	a flag (TRUE/FLASE) indicating whether to perturb the latent states during estimation. Only useful for ensemble forecasting.
`xtol_rel`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.
`stopval`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.
`ftol_rel`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.
`ftol_abs`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.
`maxeval`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.
`maxtime`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of bootstrap samples `R` is a large value.
`seed`	Random seed.

Details

Type 0:

The measurement model is given by

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

where $\mathbf{y}_{i,t}$, $\boldsymbol{\eta}_{i,t}$, and $\boldsymbol{\varepsilon}_{i,t}$ are random variables and $\boldsymbol{\nu}$, $\boldsymbol{\Lambda}$, 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. $\boldsymbol{\nu}$ denotes a vector of intercepts, $\boldsymbol{\Lambda}$ a matrix of factor loadings, and $\boldsymbol{\Theta}$ the covariance matrix of $\boldsymbol{\varepsilon}$.

An alternative representation of the measurement error is given by

$$\boldsymbol{\varepsilon}_{i,t} = \boldsymbol{\Theta}^{\frac{1}{2}}\mathbf{z}_{i,t}, \quad \text{with} \quad \mathbf{z}_{i,t} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)$$

where $\mathbf{z}_{i,t}$ is a vector of independent standard normal random variables and $\left(\boldsymbol{\Theta}^{\frac{1}{2}}\right)\left(\boldsymbol{\Theta}^{\frac{1}{2}}\right)' = \boldsymbol{\Theta}$.

The dynamic structure is given by

$$\mathrm{d}\boldsymbol{\eta}_{i,t} = \left(\boldsymbol{\iota} + \boldsymbol{\Phi}\boldsymbol{\eta}_{i,t}\right)\mathrm{d}t + \boldsymbol{\Sigma}^{\frac{1}{2}}\mathrm{d}\mathbf{W}_{i,t}$$

where $\boldsymbol{\iota}$ is a term which is unobserved and constant over time, $\boldsymbol{\Phi}$ is the drift matrix which represents the rate of change of the solution in the absence of any random fluctuations, $\boldsymbol{\Sigma}$ is the matrix of volatility or randomness in the process, and $\mathrm{d}\boldsymbol{W}$ is a Wiener process or Brownian motion, which represents random fluctuations.

Type 1:

The measurement model is given by

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

The dynamic structure is given by

$$\mathrm{d} \boldsymbol{\eta}_{i,t} = \left(\boldsymbol{\iota} + \boldsymbol{\Phi} \boldsymbol{\eta}_{i,t} \right) \mathrm{d}t + \boldsymbol{\Gamma} \mathbf{x}_{i,t} + \boldsymbol{\Sigma}^{\frac{1}{2}} \mathrm{d} \mathbf{W}_{i,t}$$

where $\mathbf{x}_{i,t}$ represents a vector of covariates at time t and individual i, and $\boldsymbol{\Gamma}$ the coefficient matrix linking the covariates to the latent variables.

Type 2:

The measurement model is given by

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

where $\boldsymbol{\kappa}$ represents the coefficient matrix linking the covariates to the observed variables.

The dynamic structure is given by

$$\mathrm{d} \boldsymbol{\eta}_{i,t} = \left(\boldsymbol{\iota} + \boldsymbol{\Phi} \boldsymbol{\eta}_{i,t} \right) \mathrm{d}t + \boldsymbol{\Gamma} \mathbf{x}_{i,t} + \boldsymbol{\Sigma}^{\frac{1}{2}} \mathrm{d} \mathbf{W}_{i,t}.$$

State Space Parameterization:

The state space parameters as a function of the linear stochastic differential equation model parameters are given by

$$\boldsymbol{\beta}_{\Delta t_{l_i}} = \exp \left(\Delta t \boldsymbol{\Phi} \right)$$

$$\boldsymbol{\alpha}_{\Delta t_{l_i}} = \boldsymbol{\Phi}^{-1} \left(\boldsymbol{\beta} - \mathbf{I}_p \right) \boldsymbol{\iota}$$

$$\mathrm{vec} \left(\boldsymbol{\Psi}_{\Delta t_{l_i}} \right) = \left[\left(\boldsymbol{\Phi} \otimes \mathbf{I}_p \right) + \left(\mathbf{I}_p \otimes \boldsymbol{\Phi} \right) \right] \left[\exp \left(\left[\left(\boldsymbol{\Phi} \otimes \mathbf{I}_p \right) + \left(\mathbf{I}_p \otimes \boldsymbol{\Phi} \right) \right] \Delta t \right) - \mathbf{I}_{p \times p} \right] \mathrm{vec} \left(\boldsymbol{\Sigma} \right)$$

where p is the number of latent variables and Δt is the time interval.

Value

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

call Function call.

args Function arguments.

thetahatstar Sampling distribution of $\hat{\boldsymbol{\theta}}$.

vcov Sampling variance-covariance matrix of $\hat{\boldsymbol{\theta}}$.

est Vector of estimated $\hat{\boldsymbol{\theta}}$.

fun Function used ("PBSSMLinSDEFixed").

method Bootstrap method used ("parametric").

Author(s)

Ivan Jacob Agaloos Pesigan

References

Chow, S.-M., Ho, M. R., Hamaker, E. L., & Dolan, C. V. (2010). Equivalence and differences between structural equation modeling and state-space modeling techniques. *Structural Equation Modeling: A Multidisciplinary Journal*, 17(2), 303–332. doi:10.1080/10705511003661553

Chow, S.-M., Losardo, D., Park, J., & Molenaar, P. C. M. (2023). Continuous-time dynamic models: Connections to structural equation models and other discrete-time models. In R. H. Hoyle (Ed.), Handbook of structural equation modeling (2nd ed.). The Guilford Press.

Harvey, A. C. (1990). Forecasting, structural time series models and the Kalman filter. Cambridge University Press. doi:10.1017/cbo9781107049994

See Also

Other Simulation of State Space Models Data Functions: `PBSSMFixed()`, `PBSSMOUFixed()`, `PBSSMVARFixed()`

Examples

```
## Not run:
# prepare parameters
## number of individuals
n <- 5
## time points
time <- 50
delta_t <- 0.10
## dynamic structure
p <- 2
mu0 <- c(-3.0, 1.5)
sigma0 <- 0.001 * diag(p)
sigma0_l <- t(chol(sigma0))
iota <- c(0.317, 0.230)
phi <- matrix(
  data = c(
    -0.10,
    0.05,
    0.05,
    -0.10
  ),
  nrow = p
)
sigma <- matrix(
  data = c(
    2.79,
    0.06,
    0.06,
    3.27
  ),
  nrow = p
```

```
)
sigma_l <- t(chol(sigma))
## measurement model
k <- 2
nu <- rep(x = 0, times = k)
lambda <- diag(k)
theta <- 0.001 * diag(k)
theta_l <- t(chol(theta))

pb <- PBSSMLinSDEFixed(
  R = 10L, # use at least 1000 in actual research
  path = getwd(),
  prefix = "lse",
  n = n,
  time = time,
  delta_t = delta_t,
  mu0 = mu0,
  sigma0_l = sigma0_l,
  iota = iota,
  phi = phi,
  sigma_l = sigma_l,
  nu = nu,
  lambda = lambda,
  theta_l = theta_l,
  type = 0,
  ncores = 1, # consider using multiple cores
  seed = 42
)
print(pb)
summary(pb)
confint(pb)
vcov(pb)
coef(pb)
print(pb, type = "bc") # bias-corrected
summary(pb, type = "bc")
confint(pb, type = "bc")

## End(Not run)
```

PBSSMOUFixed *Parametric Bootstrap for the Ornstein–Uhlenbeck Model using a State Space Model Parameterization (Fixed Parameters)*

Description

This function simulates data from a Ornstein–Uhlenbeck (OU) model using a state-space model parameterization and fits the model using the `dynr` package. The process is repeated `R` times. It assumes that the parameters remain constant across individuals and over time. At the moment, the function only supports `type = 0`.

Usage

```
PBSSMOUFixed(
  R,
  path,
  prefix,
  n,
  time,
  delta_t = 0.1,
  mu0,
  sigma0_l,
  mu,
  phi,
  sigma_l,
  nu,
  lambda,
  theta_l,
  type = 0,
  x = NULL,
  gamma = NULL,
  kappa = NULL,
  mu0_fixed = FALSE,
  sigma0_fixed = FALSE,
  alpha_level = 0.05,
  optimization_flag = TRUE,
  hessian_flag = FALSE,
  verbose = FALSE,
  weight_flag = FALSE,
  debug_flag = FALSE,
  perturb_flag = FALSE,
  xtol_rel = 1e-07,
  stopval = -9999,
  ftol_rel = -1,
  ftol_abs = -1,
  maxeval = as.integer(-1),
  maxtime = -1,
  ncores = NULL,
  seed = NULL
)
```

Arguments

`R`	Positive integer. Number of bootstrap samples.
`path`	Path to a directory to store bootstrap samples and estimates.
`prefix`	Character string. Prefix used for the file names for the bootstrap samples and estimates.
`n`	Positive integer. Number of individuals.
`time`	Positive integer. Number of time points.

delta_t	Numeric. Time interval (Δ_t).
mu0	Numeric vector. Mean of initial latent variable values ($\boldsymbol{\mu}_{\boldsymbol{\eta}\|0}$).
sigma0_l	Numeric matrix. Cholesky factorization (t(chol(sigma0))) of the covariance matrix of initial latent variable values ($\boldsymbol{\Sigma}_{\boldsymbol{\eta}\|0}$).
mu	Numeric vector. The long-term mean or equilibrium level ($\boldsymbol{\mu}$).
phi	Numeric matrix. The drift matrix which represents the rate of change of the solution in the absence of any random fluctuations ($\boldsymbol{\Phi}$). It also represents the rate of mean reversion, determining how quickly the variable returns to its mean.
sigma_l	Numeric matrix. Cholesky factorization (t(chol(sigma))) of the covariance matrix of volatility or randomness in the process ($\boldsymbol{\Sigma}$).
nu	Numeric vector. Vector of intercept values for the measurement model ($\boldsymbol{\nu}$).
lambda	Numeric matrix. Factor loading matrix linking the latent variables to the observed variables ($\boldsymbol{\Lambda}$).
theta_l	Numeric matrix. Cholesky factorization (t(chol(theta))) of the covariance matrix of the measurement error ($\boldsymbol{\Theta}$).
type	Integer. State space model type. See Details for more information.
x	List. Each element of the list is a matrix of covariates for each individual i in n. The number of columns in each matrix should be equal to time.
gamma	Numeric matrix. Matrix linking the covariates to the latent variables at current time point ($\boldsymbol{\Gamma}$).
kappa	Numeric matrix. Matrix linking the covariates to the observed variables at current time point ($\boldsymbol{\kappa}$).
mu0_fixed	Logical. If mu0_fixed = TRUE, fix the initial mean vector to mu0. If mu0_fixed = FALSE, mu0 is estimated.
sigma0_fixed	Logical. If sigma0_fixed = TRUE, fix the initial covariance matrix to tcrossprod(sigma0_l). If sigma0_fixed = FALSE, sigma0 is estimated.
alpha_level	Numeric vector. Significance level α.
optimization_flag	a flag (TRUE/FALSE) indicating whether optimization is to be done.
hessian_flag	a flag (TRUE/FALSE) indicating whether the Hessian matrix is to be calculated.
verbose	a flag (TRUE/FALSE) indicating whether more detailed intermediate output during the estimation process should be printed
weight_flag	a flag (TRUE/FALSE) indicating whether the negative log likelihood function should be weighted by the length of the time series for each individual
debug_flag	a flag (TRUE/FALSE) indicating whether users want additional dynr output that can be used for diagnostic purposes
perturb_flag	a flag (TRUE/FLASE) indicating whether to perturb the latent states during estimation. Only useful for ensemble forecasting.
xtol_rel	Stopping criteria option for parameter optimization. See dynr::dynr.model() for more details.
stopval	Stopping criteria option for parameter optimization. See dynr::dynr.model() for more details.

`ftol_rel`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.
`ftol_abs`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.
`maxeval`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.
`maxtime`	Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of bootstrap samples `R` is a large value.
`seed`	Random seed.

Details

Type 0:

The measurement model is given by

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

where $\mathbf{y}_{i,t}$, $\boldsymbol{\eta}_{i,t}$, and $\boldsymbol{\varepsilon}_{i,t}$ are random variables and $\boldsymbol{\nu}$, $\boldsymbol{\Lambda}$, 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. $\boldsymbol{\nu}$ denotes a vector of intercepts, $\boldsymbol{\Lambda}$ a matrix of factor loadings, and $\boldsymbol{\Theta}$ the covariance matrix of $\boldsymbol{\varepsilon}$.

An alternative representation of the measurement error is given by

$$\boldsymbol{\varepsilon}_{i,t} = \boldsymbol{\Theta}^{\frac{1}{2}}\mathbf{z}_{i,t}, \quad \text{with} \quad \mathbf{z}_{i,t} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)$$

where $\mathbf{z}_{i,t}$ is a vector of independent standard normal random variables and $\left(\boldsymbol{\Theta}^{\frac{1}{2}}\right)\left(\boldsymbol{\Theta}^{\frac{1}{2}}\right)' = \boldsymbol{\Theta}$.

The dynamic structure is given by

$$\mathrm{d}\boldsymbol{\eta}_{i,t} = \boldsymbol{\Phi}\left(\boldsymbol{\eta}_{i,t} - \boldsymbol{\mu}\right)\mathrm{d}t + \boldsymbol{\Sigma}^{\frac{1}{2}}\mathrm{d}\mathbf{W}_{i,t}$$

where $\boldsymbol{\mu}$ is the long-term mean or equilibrium level, $\boldsymbol{\Phi}$ is the rate of mean reversion, determining how quickly the variable returns to its mean, $\boldsymbol{\Sigma}$ is the matrix of volatility or randomness in the process, and $\mathrm{d}\boldsymbol{W}$ is a Wiener process or Brownian motion, which represents random fluctuations.

Type 1:

The measurement model is given by

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

The dynamic structure is given by

$$\mathrm{d}\boldsymbol{\eta}_{i,t} = \boldsymbol{\Phi}\left(\boldsymbol{\eta}_{i,t} - \boldsymbol{\mu}\right)\mathrm{d}t + \boldsymbol{\Gamma}\mathbf{x}_{i,t} + \boldsymbol{\Sigma}^{\frac{1}{2}}\mathrm{d}\mathbf{W}_{i,t}$$

where $\mathbf{x}_{i,t}$ represents a vector of covariates at time t and individual i, and $\boldsymbol{\Gamma}$ the coefficient matrix linking the covariates to the latent variables.

Type 2:
The measurement model is given by

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

where $\boldsymbol{\kappa}$ represents the coefficient matrix linking the covariates to the observed variables.
The dynamic structure is given by

$$\mathrm{d}\boldsymbol{\eta}_{i,t} = \boldsymbol{\Phi}\left(\boldsymbol{\eta}_{i,t} - \boldsymbol{\mu}\right)\mathrm{d}t + \boldsymbol{\Gamma}\mathbf{x}_{i,t} + \boldsymbol{\Sigma}^{\frac{1}{2}}\mathrm{d}\mathbf{W}_{i,t}.$$

The OU model as a linear stochastic differential equation model:
The OU model is a first-order linear stochastic differential equation model in the form of

$$\mathrm{d}\boldsymbol{\eta}_{i,t} = \left(\boldsymbol{\iota} + \boldsymbol{\Phi}\boldsymbol{\eta}_{i,t}\right)\mathrm{d}t + \boldsymbol{\Sigma}^{\frac{1}{2}}\mathrm{d}\mathbf{W}_{i,t}$$

where $\boldsymbol{\mu} = -\boldsymbol{\Phi}^{-1}\boldsymbol{\iota}$ and, equivalently $\boldsymbol{\iota} = -\boldsymbol{\Phi}\boldsymbol{\mu}$.

Value

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

call Function call.

args Function arguments.

thetahatstar Sampling distribution of $\hat{\boldsymbol{\theta}}$.

vcov Sampling variance-covariance matrix of $\hat{\boldsymbol{\theta}}$.

est Vector of estimated $\hat{\boldsymbol{\theta}}$.

fun Function used ("PBSSMOUFixed").

method Bootstrap method used ("parametric").

Author(s)

Ivan Jacob Agaloos Pesigan

References

Chow, S.-M., Ho, M. R., Hamaker, E. L., & Dolan, C. V. (2010). Equivalence and differences between structural equation modeling and state-space modeling techniques. *Structural Equation Modeling: A Multidisciplinary Journal*, 17(2), 303–332. doi:10.1080/10705511003661553

Chow, S.-M., Losardo, D., Park, J., & Molenaar, P. C. M. (2023). Continuous-time dynamic models: Connections to structural equation models and other discrete-time models. In R. H. Hoyle (Ed.), Handbook of structural equation modeling (2nd ed.). The Guilford Press.

Harvey, A. C. (1990). Forecasting, structural time series models and the Kalman filter. Cambridge University Press. doi:10.1017/cbo9781107049994

Oravecz, Z., Tuerlinckx, F., & Vandekerckhove, J. (2011). A hierarchical latent stochastic differential equation model for affective dynamics. Psychological Methods, 16 (4), 468–490. doi:10.1037/a0024375

Uhlenbeck, G. E., & Ornstein, L. S. (1930). On the theory of the brownian motion. Physical Review, 36 (5), 823–841. doi:10.1103/physrev.36.823

See Also

Other Simulation of State Space Models Data Functions: PBSSMFixed(), PBSSMLinSDEFixed(), PBSSMVARFixed()

Examples

```
## Not run:
# prepare parameters
## number of individuals
n <- 5
## time points
time <- 50
delta_t <- 0.10
## dynamic structure
p <- 2
mu0 <- c(-3.0, 1.5)
sigma0 <- 0.001 * diag(p)
sigma0_l <- t(chol(sigma0))
mu <- c(5.76, 5.18)
phi <- matrix(
  data = c(
    -0.10,
    0.05,
    0.05,
    -0.10
  ),
  nrow = p
)
sigma <- matrix(
  data = c(
    2.79,
    0.06,
    0.06,
    3.27
  ),
  nrow = p
)
sigma_l <- t(chol(sigma))
## measurement model
k <- 2
nu <- rep(x = 0, times = k)
lambda <- diag(k)
theta <- 0.001 * diag(k)
theta_l <- t(chol(theta))

pb <- PBSSMOUFixed(
  R = 10L, # use at least 1000 in actual research
  path = getwd(),
  prefix = "ou",
  n = n,
  time = time,
  delta_t = delta_t,
```

```
  mu0 = mu0,
  sigma0_l = sigma0_l,
  mu = mu,
  phi = phi,
  sigma_l = sigma_l,
  nu = nu,
  lambda = lambda,
  theta_l = theta_l,
  type = 0,
  ncores = 1, # consider using multiple cores
  seed = 42
)
print(pb)
summary(pb)
confint(pb)
vcov(pb)
coef(pb)
print(pb, type = "bc") # bias-corrected
summary(pb, type = "bc")
confint(pb, type = "bc")

## End(Not run)
```

PBSSMVARFixed *Parametric Bootstrap for the Vector Autoregressive Model (Fixed Parameters)*

Description

This function simulates data from a vector autoregressive model using a state-space model parameterization and fits the model using the `dynr` package. The process is repeated `R` times. It assumes that the parameters remain constant across individuals and over time. At the moment, the function only supports `type = 0`.

Usage

```
PBSSMVARFixed(
  R,
  path,
  prefix,
  n,
  time,
  mu0,
  sigma0_l,
  alpha,
  beta,
  psi_l,
  type = 0,
```

```
  x = NULL,
  gamma = NULL,
  mu0_fixed = FALSE,
  sigma0_fixed = FALSE,
  alpha_level = 0.05,
  optimization_flag = TRUE,
  hessian_flag = FALSE,
  verbose = FALSE,
  weight_flag = FALSE,
  debug_flag = FALSE,
  perturb_flag = FALSE,
  xtol_rel = 1e-07,
  stopval = -9999,
  ftol_rel = -1,
  ftol_abs = -1,
  maxeval = as.integer(-1),
  maxtime = -1,
  ncores = NULL,
  seed = NULL
)
```

Arguments

`R`	Positive integer. Number of bootstrap samples.
`path`	Path to a directory to store bootstrap samples and estimates.
`prefix`	Character string. Prefix used for the file names for the bootstrap samples and estimates.
`n`	Positive integer. Number of individuals.
`time`	Positive integer. Number of time points.
`mu0`	Numeric vector. Mean of initial latent variable values ($\boldsymbol{\mu}_{\boldsymbol{\eta}\mid 0}$).
`sigma0_l`	Numeric matrix. Cholesky factorization (`t(chol(sigma0))`) of the covariance matrix of initial latent variable values ($\boldsymbol{\Sigma}_{\boldsymbol{\eta}\mid 0}$).
`alpha`	Numeric vector. Vector of constant values for the dynamic model ($\boldsymbol{\alpha}$).
`beta`	Numeric matrix. Transition matrix relating the values of the latent variables at the previous to the current time point ($\boldsymbol{\beta}$).
`psi_l`	Numeric matrix. Cholesky factorization (`t(chol(psi))`) of the covariance matrix of the process noise ($\boldsymbol{\Psi}$).
`type`	Integer. State space model type. See Details for more information.
`x`	List. Each element of the list is a matrix of covariates for each individual `i` in `n`. The number of columns in each matrix should be equal to `time`.
`gamma`	Numeric matrix. Matrix linking the covariates to the latent variables at current time point ($\boldsymbol{\Gamma}$).
`mu0_fixed`	Logical. If `mu0_fixed = TRUE`, fix the initial mean vector to `mu0`. If `mu0_fixed = FALSE`, `mu0` is estimated.

`sigma0_fixed` Logical. If `sigma0_fixed = TRUE`, fix the initial covariance matrix to `tcrossprod(sigma0_l)`. If `sigma0_fixed = FALSE`, `sigma0` is estimated.

`alpha_level` Numeric vector. Significance level α.

`optimization_flag` a flag (TRUE/FALSE) indicating whether optimization is to be done.

`hessian_flag` a flag (TRUE/FALSE) indicating whether the Hessian matrix is to be calculated.

`verbose` a flag (TRUE/FALSE) indicating whether more detailed intermediate output during the estimation process should be printed

`weight_flag` a flag (TRUE/FALSE) indicating whether the negative log likelihood function should be weighted by the length of the time series for each individual

`debug_flag` a flag (TRUE/FALSE) indicating whether users want additional dynr output that can be used for diagnostic purposes

`perturb_flag` a flag (TRUE/FLASE) indicating whether to perturb the latent states during estimation. Only useful for ensemble forecasting.

`xtol_rel` Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.

`stopval` Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.

`ftol_rel` Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.

`ftol_abs` Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.

`maxeval` Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.

`maxtime` Stopping criteria option for parameter optimization. See `dynr::dynr.model()` for more details.

`ncores` Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of bootstrap samples `R` is a large value.

`seed` Random seed.

Details

Type 0:

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. Since the observed and latent variables are equal, we only generate data from the dynamic structure.

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 $\boldsymbol{\zeta}_{i,t}$.

An alternative representation of the dynamic noise is given by

$$\boldsymbol{\zeta}_{i,t} = \boldsymbol{\Psi}^{\frac{1}{2}} \mathbf{z}_{i,t}, \quad \text{with} \quad \mathbf{z}_{i,t} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)$$

where $\left(\boldsymbol{\Psi}^{\frac{1}{2}}\right)\left(\boldsymbol{\Psi}^{\frac{1}{2}}\right)^{\prime} = \boldsymbol{\Psi}$.

Type 1:

The measurement model is given by

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

The dynamic structure is given by

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

where $\mathbf{x}_{i,t}$ represents a vector of covariates at time t and individual i, and $\boldsymbol{\Gamma}$ the coefficient matrix linking the covariates to the latent variables.

Value

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

call Function call.

args Function arguments.

thetahatstar Sampling distribution of $\hat{\boldsymbol{\theta}}$.

vcov Sampling variance-covariance matrix of $\hat{\boldsymbol{\theta}}$.

est Vector of estimated $\hat{\boldsymbol{\theta}}$.

fun Function used ("PBSSMVARFixed").

method Bootstrap method used ("parametric").

Author(s)

Ivan Jacob Agaloos Pesigan

References

Chow, S.-M., Ho, M. R., Hamaker, E. L., & Dolan, C. V. (2010). Equivalence and differences between structural equation modeling and state-space modeling techniques. *Structural Equation Modeling: A Multidisciplinary Journal*, 17(2), 303–332. doi:10.1080/10705511003661553

See Also

Other Simulation of State Space Models Data Functions: `PBSSMFixed()`, `PBSSMLinSDEFixed()`, `PBSSMOUFixed()`

Examples

```
## Not run:
# prepare parameters
## number of individuals
n <- 5
## time points
time <- 50
## dynamic structure
p <- 3
mu0 <- rep(x = 0, times = p)
sigma0 <- 0.001 * diag(p)
sigma0_l <- t(chol(sigma0))
alpha <- rep(x = 0, times = p)
beta <- 0.50 * diag(p)
psi <- 0.001 * diag(p)
psi_l <- t(chol(psi))

boot <- PBSSMVARFixed(
  R = 10L, # use at least 1000 in actual research
  path = getwd(),
  prefix = "var",
  n = n,
  time = time,
  mu0 = mu0,
  sigma0_l = sigma0_l,
  alpha = alpha,
  beta = beta,
  psi_l = psi_l,
  type = 0,
  ncores = 1, # consider using multiple cores
  seed = 42
)
print(pb)
summary(pb)
confint(pb)
vcov(pb)
coef(pb)
print(pb, type = "bc") # bias-corrected
summary(pb, type = "bc")
confint(pb, type = "bc")

## End(Not run)
```

`print.bootstatespace` *Print Method for an Object of Class* `bootstatespace`

Description

Print Method for an Object of Class `bootstatespace`

Usage

```
## S3 method for class 'bootstatespace'
print(x, alpha = NULL, type = "pc", digits = 4, ...)
```

Arguments

`x`	Object of Class `bootstatespace`.
`alpha`	Numeric vector. Significance level α. If `alpha = NULL`, use the argument `alpha` used in `x`.
`type`	Charater string. Confidence interval type, that is, `type = "pc"` for percentile; `type = "bc"` for bias corrected.
`digits`	Digits to print.
`...`	additional arguments.

Value

Prints a matrix of estimates, standard errors, number of bootstrap replications, and confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

---

summary.bootstatespace *Summary Method for an Object of Class* `bootstatespace`

Description

Summary Method for an Object of Class `bootstatespace`

Usage

```
## S3 method for class 'bootstatespace'
summary(object, alpha = NULL, type = "pc", digits = 4, ...)
```

Arguments

`object`	Object of Class `bootstatespace`.
`alpha`	Numeric vector. Significance level α. If `alpha = NULL`, use the argument `alpha` used in `object`.
`type`	Charater string. Confidence interval type, that is, `type = "pc"` for percentile; `type = "bc"` for bias corrected.
`digits`	Digits to print.
`...`	additional arguments.

Value

Returns a matrix of estimates, standard errors, number of bootstrap replications, and confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

`vcov.bootstatespace` *Sampling Variance-Covariance Matrix Method for an Object of Class* `bootstatespace`

Description

Sampling Variance-Covariance Matrix Method for an Object of Class `bootstatespace`

Usage

```
## S3 method for class 'bootstatespace'
vcov(object, ...)
```

Arguments

`object`	Object of Class `bootstatespace`.
`...`	additional arguments.

Value

Returns the variance-covariance matrix of estimates.

Author(s)

Ivan Jacob Agaloos Pesigan

Index