Package ‘cTMed’

October 1, 2025

Title Continuous Time Mediation

Version 1.0.7

Description Calculates effect sizes, standard errors and confidence intervals in continuous-time mediation models (Deboeck and Preacher, 2015 <doi:10.1080/10705511.2014.973960>; Ryan and Hamaker, 2021 <doi:10.1007/s11336-021-09767-0>).

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

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

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 3.5.0)

LinkingTo Rcpp, RcppArmadillo (>= 15.0.2-2)

Imports Rcpp, numDeriv, parallel, simStateSpace

Suggests knitr, rmarkdown, testthat, expm, dynr, bootStateSpace

SystemRequirements GNU GSL (>= 2.0)

RoxygenNote 7.3.3.9000

NeedsCompilation yes

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

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

Contents

`BootBeta` *Bootstrap Sampling Distribution for the Elements of the Matrix of Lagged Coefficients Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a bootstrap method sampling distribution for the elements of the matrix of lagged coefficients $\boldsymbol{\beta}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$.

Usage

```
BootBeta(phi, phi_hat, delta_t, ncores = NULL, tol = 0.01)
```

Arguments

`phi`	List of numeric matrices. Each element of the list is a bootstrap estimate of the drift matrix ($\boldsymbol{\Phi}$).
`phi_hat`	Numeric matrix. The estimated drift matrix ($\hat{\boldsymbol{\Phi}}$) from the original data set. `phi_hat` should have row and column names pertaining to the variables in the system.
`delta_t`	Numeric. Time interval (Δt).
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of replications `R` is a large value.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `Total()`.

Value

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

call Function call.

args Function arguments.

fun Function used ("BootBeta").

output A list with length of `length(delta_t)`.

Each element in the `output` list has the following elements:

est Estimated elements of the matrix of lagged coefficients.

thetahatstar A matrix of bootstrap elements of the matrix of lagged coefficients.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
## Not run:
library(bootStateSpace)
# prepare parameters
## number of individuals
n <- 50
## time points
time <- 100
delta_t <- 0.10
## dynamic structure
p <- 3
mu0 <- rep(x = 0, times = p)
sigma0 <- matrix(
  data = c(
    1.0,
    0.2,
    0.2,
    0.2,
    1.0,
    0.2,
    0.2,
    0.2,
```

```
    1.0
  ),
  nrow = p
)
sigma0_l <- t(chol(sigma0))
mu <- rep(x = 0, times = p)
phi <- matrix(
  data = c(
    -0.357,
    0.771,
    -0.450,
    0.0,
    -0.511,
    0.729,
    0,
    0,
    -0.693
  ),
  nrow = p
)
sigma <- matrix(
  data = c(
    0.24455556,
    0.02201587,
    -0.05004762,
    0.02201587,
    0.07067800,
    0.01539456,
    -0.05004762,
    0.01539456,
    0.07553061
  ),
  nrow = p
)
sigma_l <- t(chol(sigma))
## measurement model
k <- 3
nu <- rep(x = 0, times = k)
lambda <- diag(k)
theta <- 0.2 * diag(k)
theta_l <- t(chol(theta))

boot <- 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,
  ncores = NULL, # consider using multiple cores
  seed = 42
)
phi_hat <- phi
colnames(phi_hat) <- rownames(phi_hat) <- c("x", "m", "y")
phi <- extract(object = boot, what = "phi")

# Specific time interval ----------------------------------------------------
BootBeta(
  phi = phi,
  phi_hat = phi_hat,
  delta_t = 1
)

# Range of time intervals ---------------------------------------------------
boot <- BootBeta(
  phi = phi,
  phi_hat = phi_hat,
  delta_t = 1:5
)
plot(boot)
plot(boot, type = "bc") # bias-corrected

# Methods -------------------------------------------------------------------
# BootBeta has a number of methods including
# print, summary, confint, and plot
print(boot)
summary(boot)
confint(boot, level = 0.95)
print(boot, type = "bc") # bias-corrected
summary(boot, type = "bc")
confint(boot, level = 0.95, type = "bc")

## End(Not run)
```

`BootBetaStd` *Bootstrap Sampling Distribution for the Elements of the Standardized Matrix of Lagged Coefficients Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a bootstrap method sampling distribution for the elements of the standardized matrix of lagged coefficients $\boldsymbol{\beta}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$.

Usage

```
BootBetaStd(phi, sigma, phi_hat, sigma_hat, delta_t, ncores = NULL, tol = 0.01)
```

Arguments

`phi`	List of numeric matrices. Each element of the list is a bootstrap estimate of the drift matrix ($\boldsymbol{\Phi}$).
`sigma`	List of numeric matrices. Each element of the list is a bootstrap estimate of the process noise covariance matrix ($\boldsymbol{\Sigma}$).
`phi_hat`	Numeric matrix. The estimated drift matrix ($\hat{\boldsymbol{\Phi}}$) from the original data set. `phi_hat` should have row and column names pertaining to the variables in the system.
`sigma_hat`	Numeric matrix. The estimated process noise covariance matrix ($\hat{\boldsymbol{\Sigma}}$) from the original data set.
`delta_t`	Numeric. Time interval (Δt).
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of replications `R` is a large value.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `TotalStd()`.

Value

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

call Function call.

args Function arguments.

fun Function used ("BootBetaStd").

output A list with length of `length(delta_t)`.

Each element in the `output` list has the following elements:

est Estimated elements of the standardized matrix of lagged coefficients.

thetahatstar A matrix of bootstrap elements of the standardized matrix of lagged coefficients.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), Direct(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
## Not run:
library(bootStateSpace)
# prepare parameters
## number of individuals
n <- 50
## time points
time <- 100
delta_t <- 0.10
## dynamic structure
p <- 3
mu0 <- rep(x = 0, times = p)
sigma0 <- matrix(
  data = c(
    1.0,
    0.2,
    0.2,
    0.2,
    1.0,
    0.2,
    0.2,
    0.2,
    1.0
  ),
  nrow = p
)
sigma0_l <- t(chol(sigma0))
mu <- rep(x = 0, times = p)
phi <- matrix(
  data = c(
    -0.357,
```

```
    0.771,
    -0.450,
    0.0,
    -0.511,
    0.729,
    0,
    0,
    -0.693
  ),
  nrow = p
)
sigma <- matrix(
  data = c(
    0.24455556,
    0.02201587,
    -0.05004762,
    0.02201587,
    0.07067800,
    0.01539456,
    -0.05004762,
    0.01539456,
    0.07553061
  ),
  nrow = p
)
sigma_l <- t(chol(sigma))
## measurement model
k <- 3
nu <- rep(x = 0, times = k)
lambda <- diag(k)
theta <- 0.2 * diag(k)
theta_l <- t(chol(theta))

boot <- 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,
  ncores = NULL, # consider using multiple cores
  seed = 42
)
phi_hat <- phi
colnames(phi_hat) <- rownames(phi_hat) <- c("x", "m", "y")
```

```
sigma_hat <- sigma
phi <- extract(object = boot, what = "phi")
sigma <- extract(object = boot, what = "sigma")

# Specific time interval ----------------------------------------------------
BootBetaStd(
  phi = phi,
  sigma = sigma,
  phi_hat = phi_hat,
  sigma_hat = sigma_hat,
  delta_t = 1
)

# Range of time intervals ---------------------------------------------------
boot <- BootBetaStd(
  phi = phi,
  sigma = sigma,
  phi_hat = phi_hat,
  sigma_hat = sigma_hat,
  delta_t = 1:5
)
plot(boot)
plot(boot, type = "bc") # bias-corrected

# Methods -------------------------------------------------------------------
# BootBetaStd has a number of methods including
# print, summary, confint, and plot
print(boot)
summary(boot)
confint(boot, level = 0.95)
print(boot, type = "bc") # bias-corrected
summary(boot, type = "bc")
confint(boot, level = 0.95, type = "bc")

## End(Not run)
```

`BootIndirectCentral` *Bootstrap Sampling Distribution for the Indirect Effect Centrality Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a bootstrap method sampling distribution for the indirect effect centrality over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$.

Usage

```
BootIndirectCentral(phi, phi_hat, delta_t, ncores = NULL, tol = 0.01)
```

Arguments

phi
: List of numeric matrices. Each element of the list is a bootstrap estimate of the drift matrix ($\boldsymbol{\Phi}$).

phi_hat
: Numeric matrix. The estimated drift matrix ($\hat{\boldsymbol{\Phi}}$) from the original data set. phi_hat should have row and column names pertaining to the variables in the system.

delta_t
: Numeric. Time interval (Δt).

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

tol
: Numeric. Smallest possible time interval to allow.

Details

See IndirectCentral() more details.

Value

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

call Function call.

args Function arguments.

fun Function used ("BootIndirectCentral").

output A list with length of length(delta_t).

Each element in the output list has the following elements:

est A vector of indirect effect centrality.

thetahatstar A matrix of bootstrap indirect effect centrality.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), Direct(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
## Not run:
library(bootStateSpace)
# prepare parameters
## number of individuals
n <- 50
## time points
time <- 100
delta_t <- 0.10
## dynamic structure
p <- 3
mu0 <- rep(x = 0, times = p)
sigma0 <- matrix(
  data = c(
    1.0,
    0.2,
    0.2,
    0.2,
    1.0,
    0.2,
    0.2,
    0.2,
    1.0
  ),
  nrow = p
)
sigma0_l <- t(chol(sigma0))
mu <- rep(x = 0, times = p)
phi <- matrix(
  data = c(
    -0.357,
    0.771,
    -0.450,
    0.0,
    -0.511,
    0.729,
    0,
    0,
    -0.693
  ),
  nrow = p
)
sigma <- matrix(
```

```
  data = c(
    0.24455556,
    0.02201587,
    -0.05004762,
    0.02201587,
    0.07067800,
    0.01539456,
    -0.05004762,
    0.01539456,
    0.07553061
  ),
  nrow = p
)
sigma_l <- t(chol(sigma))
## measurement model
k <- 3
nu <- rep(x = 0, times = k)
lambda <- diag(k)
theta <- 0.2 * diag(k)
theta_l <- t(chol(theta))

boot <- 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,
  ncores = NULL, # consider using multiple cores
  seed = 42
)
phi_hat <- phi
colnames(phi_hat) <- rownames(phi_hat) <- c("x", "m", "y")
phi <- extract(object = boot, what = "phi")

# Specific time interval ----------------------------------------------------
BootIndirectCentral(
  phi = phi,
  phi_hat = phi_hat,
  delta_t = 1
)

# Range of time intervals ---------------------------------------------------
boot <- BootIndirectCentral(
  phi = phi,
```

```
  phi_hat = phi_hat,
  delta_t = 1:5
)
plot(boot)
plot(boot, type = "bc") # bias-corrected

# Methods -----------------------------------------------------------------
# BootIndirectCentral has a number of methods including
# print, summary, confint, and plot
print(boot)
summary(boot)
confint(boot, level = 0.95)
print(boot, type = "bc") # bias-corrected
summary(boot, type = "bc")
confint(boot, level = 0.95, type = "bc")

## End(Not run)
```

BootMed — *Bootstrap Sampling Distribution of Total, Direct, and Indirect Effects of X on Y Through M Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a bootstrap method sampling distribution of the total, direct and indirect effects of the independent variable X on the dependent variable Y through mediator variables $\mathbf{m}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$.

Usage

```
BootMed(phi, phi_hat, delta_t, from, to, med, ncores = NULL, tol = 0.01)
```

Arguments

`phi`	List of numeric matrices. Each element of the list is a bootstrap estimate of the drift matrix ($\boldsymbol{\Phi}$).
`phi_hat`	Numeric matrix. The estimated drift matrix ($\hat{\boldsymbol{\Phi}}$) from the original data set. `phi_hat` should have row and column names pertaining to the variables in the system.
`delta_t`	Numeric. Time interval (Δt).
`from`	Character string. Name of the independent variable X in `phi`.
`to`	Character string. Name of the dependent variable Y in `phi`.
`med`	Character vector. Name/s of the mediator variable/s in `phi`.
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of replications `R` is a large value.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See Total(), Direct(), and Indirect() for more details.

Value

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

call Function call.

args Function arguments.

fun Function used ("BootMed").

output A list with length of length(delta_t).

Each element in the output list has the following elements:

est A vector of total, direct, and indirect effects.

thetahatstar A matrix of bootstrap total, direct, and indirect effects.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), Direct(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
## Not run:
library(bootStateSpace)
# prepare parameters
## number of individuals
n <- 50
## time points
time <- 100
```

```
delta_t <- 0.10
## dynamic structure
p <- 3
mu0 <- rep(x = 0, times = p)
sigma0 <- matrix(
  data = c(
    1.0,
    0.2,
    0.2,
    0.2,
    1.0,
    0.2,
    0.2,
    0.2,
    1.0
  ),
  nrow = p
)
sigma0_l <- t(chol(sigma0))
mu <- rep(x = 0, times = p)
phi <- matrix(
  data = c(
    -0.357,
    0.771,
    -0.450,
    0.0,
    -0.511,
    0.729,
    0,
    0,
    -0.693
  ),
  nrow = p
)
sigma <- matrix(
  data = c(
    0.24455556,
    0.02201587,
    -0.05004762,
    0.02201587,
    0.07067800,
    0.01539456,
    -0.05004762,
    0.01539456,
    0.07553061
  ),
  nrow = p
)
sigma_l <- t(chol(sigma))
## measurement model
k <- 3
nu <- rep(x = 0, times = k)
lambda <- diag(k)
```

```
theta <- 0.2 * diag(k)
theta_l <- t(chol(theta))

boot <- 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,
  ncores = NULL, # consider using multiple cores
  seed = 42
)
phi_hat <- phi
colnames(phi_hat) <- rownames(phi_hat) <- c("x", "m", "y")
phi <- extract(object = boot, what = "phi")

# Specific time interval ----------------------------------------------------
BootMed(
  phi = phi,
  phi_hat = phi_hat,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m"
)

# Range of time intervals ---------------------------------------------------
boot <- BootMed(
  phi = phi,
  phi_hat = phi_hat,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)
plot(boot)
plot(boot, type = "bc") # bias-corrected

# Methods -------------------------------------------------------------------
# BootMed has a number of methods including
# print, summary, confint, and plot
print(boot)
summary(boot)
confint(boot, level = 0.95)
```

```
print(boot, type = "bc") # bias-corrected
summary(boot, type = "bc")
confint(boot, level = 0.95, type = "bc")

## End(Not run)
```

BootMedStd *Bootstrap Sampling Distribution of Standardized Total, Direct, and Indirect Effects of X on Y Through M Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a bootstrap method sampling distribution of the standardized total, direct and indirect effects of the independent variable X on the dependent variable Y through mediator variables $\mathbf{m}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$.

Usage

```
BootMedStd(
  phi,
  sigma,
  phi_hat,
  sigma_hat,
  delta_t,
  from,
  to,
  med,
  ncores = NULL,
  tol = 0.01
)
```

Arguments

phi	List of numeric matrices. Each element of the list is a bootstrap estimate of the drift matrix ($\boldsymbol{\Phi}$).
sigma	List of numeric matrices. Each element of the list is a bootstrap estimate of the process noise covariance matrix ($\boldsymbol{\Sigma}$).
phi_hat	Numeric matrix. The estimated drift matrix ($\hat{\boldsymbol{\Phi}}$) from the original data set. phi_hat should have row and column names pertaining to the variables in the system.
sigma_hat	Numeric matrix. The estimated process noise covariance matrix ($\hat{\boldsymbol{\Sigma}}$) from the original data set.
delta_t	Numeric. Time interval (Δt).

`from`	Character string. Name of the independent variable X in `phi`.
`to`	Character string. Name of the dependent variable Y in `phi`.
`med`	Character vector. Name/s of the mediator variable/s in `phi`.
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of replications `R` is a large value.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `TotalStd()`, `DirectStd()`, and `IndirectStd()` for more details.

Value

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

call Function call.

args Function arguments.

fun Function used ("BootMedStd").

output A list with length of `length(delta_t)`.

Each element in the `output` list has the following elements:

est A vector of standardized total, direct, and indirect effects.

thetahatstar A matrix of bootstrap standardized total, direct, and indirect effects.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
## Not run:
library(bootStateSpace)
# prepare parameters
## number of individuals
n <- 50
## time points
time <- 100
delta_t <- 0.10
## dynamic structure
p <- 3
mu0 <- rep(x = 0, times = p)
sigma0 <- matrix(
  data = c(
    1.0,
    0.2,
    0.2,
    0.2,
    1.0,
    0.2,
    0.2,
    0.2,
    1.0
  ),
  nrow = p
)
sigma0_l <- t(chol(sigma0))
mu <- rep(x = 0, times = p)
phi <- matrix(
  data = c(
    -0.357,
    0.771,
    -0.450,
    0.0,
    -0.511,
    0.729,
    0,
    0,
    -0.693
  ),
  nrow = p
)
sigma <- matrix(
  data = c(
    0.24455556,
    0.02201587,
    -0.05004762,
    0.02201587,
    0.07067800,
    0.01539456,
    -0.05004762,
    0.01539456,
```

```
    0.07553061
  ),
  nrow = p
)
sigma_l <- t(chol(sigma))
## measurement model
k <- 3
nu <- rep(x = 0, times = k)
lambda <- diag(k)
theta <- 0.2 * diag(k)
theta_l <- t(chol(theta))

boot <- 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,
  ncores = NULL, # consider using multiple cores
  seed = 42
)
phi_hat <- phi
colnames(phi_hat) <- rownames(phi_hat) <- c("x", "m", "y")
sigma_hat <- sigma
phi <- extract(object = boot, what = "phi")
sigma <- extract(object = boot, what = "sigma")

# Specific time interval ----------------------------------------------------
BootMedStd(
  phi = phi,
  sigma = sigma,
  phi_hat = phi_hat,
  sigma_hat = sigma_hat,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m"
)

# Range of time intervals ---------------------------------------------------
boot <- BootMedStd(
  phi = phi,
  sigma = sigma,
  phi_hat = phi_hat,
```

```
  sigma_hat = sigma_hat,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)
plot(boot)
plot(boot, type = "bc") # bias-corrected

# Methods -----------------------------------------------------------------
# BootMedStd has a number of methods including
# print, summary, confint, and plot
print(boot)
summary(boot)
confint(boot, level = 0.95)
print(boot, type = "bc") # bias-corrected
summary(boot, type = "bc")
confint(boot, level = 0.95, type = "bc")

## End(Not run)
```

`BootTotalCentral` *Bootstrap Sampling Distribution for the Total Effect Centrality Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a bootstrap method sampling distribution for the total effect centrality over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$.

Usage

```
BootTotalCentral(phi, phi_hat, delta_t, ncores = NULL, tol = 0.01)
```

Arguments

`phi`	List of numeric matrices. Each element of the list is a bootstrap estimate of the drift matrix ($\boldsymbol{\Phi}$).
`phi_hat`	Numeric matrix. The estimated drift matrix ($\hat{\boldsymbol{\Phi}}$) from the original data set. `phi_hat` should have row and column names pertaining to the variables in the system.
`delta_t`	Numeric. Time interval (Δt).
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of replications `R` is a large value.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `TotalCentral()` more details.

Value

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

call Function call.

args Function arguments.

fun Function used ("BootTotalCentral").

output A list with length of `length(delta_t)`.

Each element in the `output` list has the following elements:

est A vector of total effect centrality.

thetahatstar A matrix of bootstrap total effect centrality.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
## Not run:
library(bootStateSpace)
# prepare parameters
## number of individuals
n <- 50
## time points
time <- 100
```

```
delta_t <- 0.10
## dynamic structure
p <- 3
mu0 <- rep(x = 0, times = p)
sigma0 <- matrix(
  data = c(
    1.0,
    0.2,
    0.2,
    0.2,
    1.0,
    0.2,
    0.2,
    0.2,
    1.0
  ),
  nrow = p
)
sigma0_l <- t(chol(sigma0))
mu <- rep(x = 0, times = p)
phi <- matrix(
  data = c(
    -0.357,
    0.771,
    -0.450,
    0.0,
    -0.511,
    0.729,
    0,
    0,
    -0.693
  ),
  nrow = p
)
sigma <- matrix(
  data = c(
    0.24455556,
    0.02201587,
    -0.05004762,
    0.02201587,
    0.07067800,
    0.01539456,
    -0.05004762,
    0.01539456,
    0.07553061
  ),
  nrow = p
)
sigma_l <- t(chol(sigma))
## measurement model
k <- 3
nu <- rep(x = 0, times = k)
lambda <- diag(k)
```

```
theta <- 0.2 * diag(k)
theta_l <- t(chol(theta))

boot <- 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,
  ncores = NULL, # consider using multiple cores
  seed = 42
)
phi_hat <- phi
colnames(phi_hat) <- rownames(phi_hat) <- c("x", "m", "y")
phi <- extract(object = boot, what = "phi")

# Specific time interval ----------------------------------------------------
BootTotalCentral(
  phi = phi,
  phi_hat = phi_hat,
  delta_t = 1
)

# Range of time intervals ---------------------------------------------------
boot <- BootTotalCentral(
  phi = phi,
  phi_hat = phi_hat,
  delta_t = 1:5
)
plot(boot)
plot(boot, type = "bc") # bias-corrected

# Methods -------------------------------------------------------------------
# BootTotalCentral has a number of methods including
# print, summary, confint, and plot
print(boot)
summary(boot)
confint(boot, level = 0.95)
print(boot, type = "bc") # bias-corrected
summary(boot, type = "bc")
confint(boot, level = 0.95, type = "bc")

## End(Not run)
```

confint.ctmedboot *Bootstrap Method Confidence Intervals*

Description

Bootstrap Method Confidence Intervals

Usage

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

Arguments

object	Object of class ctmedboot.
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 data frame of confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
## Not run:
library(bootStateSpace)
# prepare parameters
## number of individuals
n <- 50
## time points
time <- 100
delta_t <- 0.10
## dynamic structure
p <- 3
mu0 <- rep(x = 0, times = p)
sigma0 <- matrix(
  data = c(
    1.0,
    0.2,
```

```
    0.2,
    0.2,
    1.0,
    0.2,
    0.2,
    0.2,
    1.0
  ),
  nrow = p
)
sigma0_l <- t(chol(sigma0))
mu <- rep(x = 0, times = p)
phi <- matrix(
  data = c(
    -0.357,
    0.771,
    -0.450,
    0.0,
    -0.511,
    0.729,
    0,
    0,
    -0.693
  ),
  nrow = p
)
sigma <- matrix(
  data = c(
    0.24455556,
    0.02201587,
    -0.05004762,
    0.02201587,
    0.07067800,
    0.01539456,
    -0.05004762,
    0.01539456,
    0.07553061
  ),
  nrow = p
)
sigma_l <- t(chol(sigma))
## measurement model
k <- 3
nu <- rep(x = 0, times = k)
lambda <- diag(k)
theta <- 0.2 * diag(k)
theta_l <- t(chol(theta))

boot <- PBSSMOUFixed(
  R = 1000L,
  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,
    ncores = parallel::detectCores() - 1,
    seed = 42
  )
  phi_hat <- phi
  colnames(phi_hat) <- rownames(phi_hat) <- c("x", "m", "y")
  phi <- extract(object = boot, what = "phi")

  # Specific time interval ----------------------------------------------------
  boot <- BootMed(
    phi = phi,
    phi_hat = phi_hat,
    delta_t = 1,
    from = "x",
    to = "y",
    med = "m"
  )
  confint(boot)
  confint(boot, type = "bc") # bias-corrected

  # Range of time intervals ---------------------------------------------------
  boot <- BootMed(
    phi = phi,
    phi_hat = phi_hat,
    delta_t = 1:5,
    from = "x",
    to = "y",
    med = "m"
  )
  confint(boot)
  confint(boot, type = "bc") # bias-corrected

  ## End(Not run)
```

`confint.ctmeddelta` *Delta Method Confidence Intervals*

Description

Delta Method Confidence Intervals

Usage

```
## S3 method for class 'ctmeddelta'
confint(object, parm = NULL, level = 0.95, ...)
```

Arguments

`object`	Object of class `ctmeddelta`.
`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.
`...`	additional arguments.

Value

Returns a data frame of confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
```

```
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

# Specific time interval ----------------------------------------------------
delta <- DeltaMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m"
)
confint(delta, level = 0.95)

# Range of time intervals ---------------------------------------------------
delta <- DeltaMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)
confint(delta, level = 0.95)
```

confint.ctmedmc *Monte Carlo Method Confidence Intervals*

Description

Monte Carlo Method Confidence Intervals

Usage

```
## S3 method for class 'ctmedmc'
confint(object, parm = NULL, level = 0.95, ...)
```

Arguments

`object`	Object of class `ctmedmc`.
`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.
`...`	additional arguments.

Value

Returns a data frame of confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
set.seed(42)
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
```

```
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

# Specific time interval ----------------------------------------------------
mc <- MCMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m",
  R = 100L # use a large value for R in actual research
)
confint(mc, level = 0.95)

# Range of time intervals ---------------------------------------------------
mc <- MCMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m",
  R = 100L # use a large value for R in actual research
)
confint(mc, level = 0.95)
```

`DeltaBeta` *Delta Method Sampling Variance-Covariance Matrix for the Elements of the Matrix of Lagged Coefficients Over a Specific Time Interval or a Range of Time Intervals*

Description

This function computes the delta method sampling variance-covariance matrix for the elements of the matrix of lagged coefficients $\boldsymbol{\beta}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model's drift matrix $\boldsymbol{\Phi}$.

Usage

```
DeltaBeta(phi, vcov_phi_vec, delta_t, ncores = NULL, tol = 0.01)
```

Arguments

`phi`	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). `phi` should have row and column names pertaining to the variables in the system.
`vcov_phi_vec`	Numeric matrix. The sampling variance-covariance matrix of $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$.
`delta_t`	Vector of positive numbers. Time interval (Δt).
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when the length of `delta_t` is long.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `Total()`.

Delta Method:

Let $\boldsymbol{\theta}$ be $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$, that is, the elements of the $\boldsymbol{\Phi}$ matrix in vector form sorted column-wise. Let $\hat{\boldsymbol{\theta}}$ be $\mathrm{vec}\left(\hat{\boldsymbol{\Phi}}\right)$. By the multivariate central limit theory, the function $\mathbf{g}$ using $\hat{\boldsymbol{\theta}}$ as input can be expressed as:

$$\sqrt{n}\left(\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) - \mathbf{g}\left(\boldsymbol{\theta}\right)\right) \xrightarrow[]{\mathrm{D}} \mathcal{N}\left(0, \mathbf{J}\boldsymbol{\Gamma}\mathbf{J}^{\prime}\right)$$

where $\mathbf{J}$ is the matrix of first-order derivatives of the function $\mathbf{g}$ with respect to the elements of $\boldsymbol{\theta}$ and $\boldsymbol{\Gamma}$ is the asymptotic variance-covariance matrix of $\hat{\boldsymbol{\theta}}$.

From the former, we can derive the distribution of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ as follows:

$$\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) \approx \mathcal{N}\left(\mathbf{g}\left(\boldsymbol{\theta}\right), n^{-1}\mathbf{J}\boldsymbol{\Gamma}\mathbf{J}^{\prime}\right)$$

The uncertainty associated with the estimator $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ is, therefore, given by $n^{-1}\mathbf{J}\boldsymbol{\Gamma}\mathbf{J}^{\prime}$. When $\boldsymbol{\Gamma}$ is unknown, by substitution, we can use the estimated sampling variance-covariance matrix of $\hat{\boldsymbol{\theta}}$, that is, $\hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)$ for $n^{-1}\boldsymbol{\Gamma}$. Therefore, the sampling variance-covariance matrix of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ is given by

$$\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) \approx \mathcal{N}\left(\mathbf{g}\left(\boldsymbol{\theta}\right), \mathbf{J}\hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)\mathbf{J}^{\prime}\right).$$

Value

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

call Function call.

args Function arguments.

fun Function used ("DeltaBeta").

output A list the length of which is equal to the length of `delta_t`.

Each element in the `output` list has the following elements:

delta_t Time interval.

jacobian Jacobian matrix.

est Estimated elements of the matrix of lagged coefficients.

vcov Sampling variance-covariance matrix of estimated elements of the matrix of lagged coefficients.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
```

```
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

# Specific time interval ----------------------------------------------------
DeltaBeta(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1
)

# Range of time intervals ---------------------------------------------------
delta <- DeltaBeta(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5
)
plot(delta)

# Methods --------------------------------------------------------------------
# DeltaBeta has a number of methods including
# print, summary, confint, and plot
print(delta)
summary(delta)
confint(delta, level = 0.95)
plot(delta)
```

`DeltaBetaStd` *Delta Method Sampling Variance-Covariance Matrix for the Elements of the Standardized Matrix of Lagged Coefficients Over a Specific Time Interval or a Range of Time Intervals*

Description

This function computes the delta method sampling variance-covariance matrix for the elements of the standardized matrix of lagged coefficients $\boldsymbol{\beta}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model's drift matrix $\boldsymbol{\Phi}$ and process noise covariance matrix $\boldsymbol{\Sigma}$.

Usage

```
DeltaBetaStd(phi, sigma, vcov_theta, delta_t, ncores = NULL, tol = 0.01)
```

Arguments

`phi`	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). `phi` should have row and column names pertaining to the variables in the system.
`sigma`	Numeric matrix. The process noise covariance matrix ($\boldsymbol{\Sigma}$).
`vcov_theta`	Numeric matrix. The sampling variance-covariance matrix of $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$ and $\mathrm{vech}\left(\boldsymbol{\Sigma}\right)$
`delta_t`	Numeric. Time interval (Δt).
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of replications `R` is a large value.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `TotalStd()`.

Delta Method:

Let $\boldsymbol{\theta}$ be a vector that combines $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$, that is, the elements of the $\boldsymbol{\Phi}$ matrix in vector form sorted column-wise and $\mathrm{vech}\left(\boldsymbol{\Sigma}\right)$, that is, the unique elements of the $\boldsymbol{\Sigma}$ matrix in vector form sorted column-wise. Let $\hat{\boldsymbol{\theta}}$ be a vector that combines $\mathrm{vec}\left(\hat{\boldsymbol{\Phi}}\right)$ and $\mathrm{vech}\left(\hat{\boldsymbol{\Sigma}}\right)$. By the multivariate central limit theory, the function $\mathbf{g}$ using $\hat{\boldsymbol{\theta}}$ as input can be expressed as:

$$\sqrt{n}\left(\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) - \mathbf{g}\left(\boldsymbol{\theta}\right)\right) \xrightarrow{\mathrm{D}} \mathcal{N}\left(0, \mathbf{J}\boldsymbol{\Gamma}\mathbf{J}'\right)$$

where $\mathbf{J}$ is the matrix of first-order derivatives of the function $\mathbf{g}$ with respect to the elements of $\boldsymbol{\theta}$ and $\boldsymbol{\Gamma}$ is the asymptotic variance-covariance matrix of $\hat{\boldsymbol{\theta}}$.

From the former, we can derive the distribution of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ as follows:

$$\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) \approx \mathcal{N}\left(\mathbf{g}\left(\boldsymbol{\theta}\right), n^{-1}\mathbf{J}\boldsymbol{\Gamma}\mathbf{J}'\right)$$

The uncertainty associated with the estimator $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ is, therefore, given by $n^{-1}\mathbf{J}\boldsymbol{\Gamma}\mathbf{J}'$. When $\boldsymbol{\Gamma}$ is unknown, by substitution, we can use the estimated sampling variance-covariance matrix of $\hat{\boldsymbol{\theta}}$, that is, $\hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)$ for $n^{-1}\boldsymbol{\Gamma}$. Therefore, the sampling variance-covariance matrix of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ is given by

$$\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) \approx \mathcal{N}\left(\mathbf{g}\left(\boldsymbol{\theta}\right), \mathbf{J}\hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)\mathbf{J}'\right).$$

Value

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

call Function call.

args Function arguments.

fun Function used ("DeltaBetaStd").

output A list the length of which is equal to the length of `delta_t`.

Each element in the `output` list has the following elements:

delta_t Time interval.

jacobian Jacobian matrix.

est Estimated elements of the standardized matrix of lagged coefficients.

vcov Sampling variance-covariance matrix of estimated elements of the standardized matrix of lagged coefficients.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
```

```
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
sigma <- matrix(
  data = c(
    0.24455556, 0.02201587, -0.05004762,
    0.02201587, 0.07067800, 0.01539456,
    -0.05004762, 0.01539456, 0.07553061
  ),
  nrow = 3
)
vcov_theta <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151, -0.00600, -0.00033,
    0.00110, 0.00324, 0.00020, -0.00061, -0.00115,
    0.00011, 0.00015, 0.00001, -0.00002, -0.00001,
    0.00040, 0.00374, 0.00016, -0.00022, -0.00273,
    -0.00016, 0.00009, 0.00150, 0.00012, -0.00010,
    -0.00026, 0.00002, 0.00012, 0.00004, -0.00001,
    -0.00151, 0.00016, 0.00389, 0.00103, -0.00007,
    -0.00283, -0.00050, 0.00000, 0.00156, 0.00021,
    -0.00005, -0.00031, 0.00001, 0.00007, 0.00006,
    -0.00600, -0.00022, 0.00103, 0.00644, 0.00031,
    -0.00119, -0.00374, -0.00021, 0.00070, 0.00064,
    -0.00015, -0.00005, 0.00000, 0.00003, -0.00001,
    -0.00033, -0.00273, -0.00007, 0.00031, 0.00287,
    0.00013, -0.00014, -0.00170, -0.00012, 0.00006,
    0.00014, -0.00001, -0.00015, 0.00000, 0.00001,
    0.00110, -0.00016, -0.00283, -0.00119, 0.00013,
    0.00297, 0.00063, -0.00004, -0.00177, -0.00013,
    0.00005, 0.00017, -0.00002, -0.00008, 0.00001,
    0.00324, 0.00009, -0.00050, -0.00374, -0.00014,
    0.00063, 0.00495, 0.00024, -0.00093, -0.00020,
    0.00006, -0.00010, 0.00000, -0.00001, 0.00004,
    0.00020, 0.00150, 0.00000, -0.00021, -0.00170,
    -0.00004, 0.00024, 0.00214, 0.00012, -0.00002,
    -0.00004, 0.00000, 0.00006, -0.00005, -0.00001,
    -0.00061, 0.00012, 0.00156, 0.00070, -0.00012,
    -0.00177, -0.00093, 0.00012, 0.00223, 0.00004,
    -0.00002, -0.00003, 0.00001, 0.00003, -0.00013,
    -0.00115, -0.00010, 0.00021, 0.00064, 0.00006,
    -0.00013, -0.00020, -0.00002, 0.00004, 0.00057,
    0.00001, -0.00009, 0.00000, 0.00000, 0.00001,
    0.00011, -0.00026, -0.00005, -0.00015, 0.00014,
    0.00005, 0.00006, -0.00004, -0.00002, 0.00001,
    0.00012, 0.00001, 0.00000, -0.00002, 0.00000,
    0.00015, 0.00002, -0.00031, -0.00005, -0.00001,
    0.00017, -0.00010, 0.00000, -0.00003, -0.00009,
    0.00001, 0.00014, 0.00000, 0.00000, -0.00005,
    0.00001, 0.00012, 0.00001, 0.00000, -0.00015,
    -0.00002, 0.00000, 0.00006, 0.00001, 0.00000,
    0.00000, 0.00000, 0.00010, 0.00001, 0.00000,
    -0.00002, 0.00004, 0.00007, 0.00003, 0.00000,
    -0.00008, -0.00001, -0.00005, 0.00003, 0.00000,
```

```
    -0.00002, 0.00000, 0.00001, 0.00005, 0.00001,
    -0.00001, -0.00001, 0.00006, -0.00001, 0.00001,
    0.00001, 0.00004, -0.00001, -0.00013, 0.00001,
    0.00000, -0.00005, 0.00000, 0.00001, 0.00012
  ),
  nrow = 15
)

# Specific time interval ----------------------------------------------------
DeltaBetaStd(
  phi = phi,
  sigma = sigma,
  vcov_theta = vcov_theta,
  delta_t = 1
)

# Range of time intervals ---------------------------------------------------
delta <- DeltaBetaStd(
  phi = phi,
  sigma = sigma,
  vcov_theta = vcov_theta,
  delta_t = 1:5
)
plot(delta)

# Methods -------------------------------------------------------------------
# DeltaBetaStd has a number of methods including
# print, summary, confint, and plot
print(delta)
summary(delta)
confint(delta, level = 0.95)
plot(delta)
```

`DeltaIndirectCentral` *Delta Method Sampling Variance-Covariance Matrix for the Indirect Effect Centrality Over a Specific Time Interval or a Range of Time Intervals*

Description

This function computes the delta method sampling variance-covariance matrix for the indirect effect centrality over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model's drift matrix $\boldsymbol{\Phi}$.

Usage

```
DeltaIndirectCentral(phi, vcov_phi_vec, delta_t, ncores = NULL, tol = 0.01)
```

Arguments

phi	Numeric matrix. The drift matrix ($\mathbf{\Phi}$). phi should have row and column names pertaining to the variables in the system.
vcov_phi_vec	Numeric matrix. The sampling variance-covariance matrix of $\mathrm{vec}\left(\mathbf{\Phi}\right)$.
delta_t	Vector of positive numbers. Time interval (Δt).
ncores	Positive integer. Number of cores to use. If ncores = NULL, use a single core. Consider using multiple cores when the length of delta_t is long.
tol	Numeric. Smallest possible time interval to allow.

Details

See IndirectCentral() more details.

Delta Method:

Let $\boldsymbol{\theta}$ be $\mathrm{vec}\left(\mathbf{\Phi}\right)$, that is, the elements of the $\mathbf{\Phi}$ matrix in vector form sorted column-wise. Let $\hat{\boldsymbol{\theta}}$ be $\mathrm{vec}\left(\hat{\mathbf{\Phi}}\right)$. By the multivariate central limit theory, the function $\mathbf{g}$ using $\hat{\boldsymbol{\theta}}$ as input can be expressed as:

$$\sqrt{n}\left(\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) - \mathbf{g}\left(\boldsymbol{\theta}\right)\right) \xrightarrow[]{\text{D}} \mathcal{N}\left(0, \mathbf{J}\mathbf{\Gamma}\mathbf{J}^{\prime}\right)$$

where $\mathbf{J}$ is the matrix of first-order derivatives of the function $\mathbf{g}$ with respect to the elements of $\boldsymbol{\theta}$ and $\mathbf{\Gamma}$ is the asymptotic variance-covariance matrix of $\hat{\boldsymbol{\theta}}$.

From the former, we can derive the distribution of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ as follows:

$$\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) \approx \mathcal{N}\left(\mathbf{g}\left(\boldsymbol{\theta}\right), n^{-1}\mathbf{J}\mathbf{\Gamma}\mathbf{J}^{\prime}\right)$$

The uncertainty associated with the estimator $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ is, therefore, given by $n^{-1}\mathbf{J}\mathbf{\Gamma}\mathbf{J}^{\prime}$. When $\mathbf{\Gamma}$ is unknown, by substitution, we can use the estimated sampling variance-covariance matrix of $\hat{\boldsymbol{\theta}}$, that is, $\hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)$ for $n^{-1}\mathbf{\Gamma}$. Therefore, the sampling variance-covariance matrix of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ is given by

$$\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) \approx \mathcal{N}\left(\mathbf{g}\left(\boldsymbol{\theta}\right), \mathbf{J}\hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)\mathbf{J}^{\prime}\right).$$

Value

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

call Function call.

args Function arguments.

fun Function used ("DeltaIndirectCentral").

output A list the length of which is equal to the length of delta_t.

Each element in the output list has the following elements:

delta_t Time interval.

jacobian Jacobian matrix.

est Estimated indirect effect centrality.

vcov Sampling variance-covariance matrix of estimated indirect effect centrality.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.002704274, -0.001475275, 0.000949122,
    -0.001619422, 0.000885122, -0.000569404,
    0.00085493, -0.000465824, 0.000297815,
    -0.001475275, 0.004428442, -0.002642303,
    0.000980573, -0.00271817, 0.001618805,
    -0.000586921, 0.001478421, -0.000871547,
    0.000949122, -0.002642303, 0.006402668,
    -0.000697798, 0.001813471, -0.004043138,
    0.000463086, -0.001120949, 0.002271711,
    -0.001619422, 0.000980573, -0.000697798,
```

```
    0.002079286, -0.001152501, 0.000753,
    -0.001528701, 0.000820587, -0.000517524,
    0.000885122, -0.00271817, 0.001813471,
    -0.001152501, 0.00342605, -0.002075005,
    0.000899165, -0.002532849, 0.001475579,
    -0.000569404, 0.001618805, -0.004043138,
    0.000753, -0.002075005, 0.004984032,
    -0.000622255, 0.001634917, -0.003705661,
    0.00085493, -0.000586921, 0.000463086,
    -0.001528701, 0.000899165, -0.000622255,
    0.002060076, -0.001096684, 0.000686386,
    -0.000465824, 0.001478421, -0.001120949,
    0.000820587, -0.002532849, 0.001634917,
    -0.001096684, 0.003328692, -0.001926088,
    0.000297815, -0.000871547, 0.002271711,
    -0.000517524, 0.001475579, -0.003705661,
    0.000686386, -0.001926088, 0.004726235
  ),
  nrow = 9
)

# Specific time interval ----------------------------------------------------
DeltaIndirectCentral(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1
)

# Range of time intervals ---------------------------------------------------
delta <- DeltaIndirectCentral(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5
)
plot(delta)

# Methods -------------------------------------------------------------------
# DeltaIndirectCentral has a number of methods including
# print, summary, confint, and plot
print(delta)
summary(delta)
confint(delta, level = 0.95)
plot(delta)
```

DeltaMed — *Delta Method Sampling Variance-Covariance Matrix for the Total, Direct, and Indirect Effects of X on Y Through M Over a Specific Time Interval or a Range of Time Intervals*

Description

This function computes the delta method sampling variance-covariance matrix for the total, direct, and indirect effects of the independent variable X on the dependent variable Y through mediator variables $\mathbf{m}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model's drift matrix $\boldsymbol{\Phi}$.

Usage

```
DeltaMed(phi, vcov_phi_vec, delta_t, from, to, med, ncores = NULL, tol = 0.01)
```

Arguments

`phi`	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). `phi` should have row and column names pertaining to the variables in the system.
`vcov_phi_vec`	Numeric matrix. The sampling variance-covariance matrix of $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$.
`delta_t`	Vector of positive numbers. Time interval (Δt).
`from`	Character string. Name of the independent variable X in `phi`.
`to`	Character string. Name of the dependent variable Y in `phi`.
`med`	Character vector. Name/s of the mediator variable/s in `phi`.
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when the length of `delta_t` is long.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `Total()`, `Direct()`, and `Indirect()` for more details.

Delta Method:

Let $\boldsymbol{\theta}$ be $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$, that is, the elements of the $\boldsymbol{\Phi}$ matrix in vector form sorted column-wise. Let $\hat{\boldsymbol{\theta}}$ be $\mathrm{vec}\left(\hat{\boldsymbol{\Phi}}\right)$. By the multivariate central limit theory, the function $\mathbf{g}$ using $\hat{\boldsymbol{\theta}}$ as input can be expressed as:

$$\sqrt{n}\left(\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) - \mathbf{g}\left(\boldsymbol{\theta}\right)\right) \xrightarrow{\mathrm{D}} \mathcal{N}\left(0, \mathbf{J}\boldsymbol{\Gamma}\mathbf{J}'\right)$$

where $\mathbf{J}$ is the matrix of first-order derivatives of the function $\mathbf{g}$ with respect to the elements of $\boldsymbol{\theta}$ and $\boldsymbol{\Gamma}$ is the asymptotic variance-covariance matrix of $\hat{\boldsymbol{\theta}}$.

From the former, we can derive the distribution of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ as follows:

$$\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) \approx \mathcal{N}\left(\mathbf{g}\left(\boldsymbol{\theta}\right), n^{-1}\mathbf{J}\boldsymbol{\Gamma}\mathbf{J}'\right)$$

The uncertainty associated with the estimator $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ is, therefore, given by $n^{-1}\mathbf{J}\boldsymbol{\Gamma}\mathbf{J}'$. When $\boldsymbol{\Gamma}$ is unknown, by substitution, we can use the estimated sampling variance-covariance matrix of $\hat{\boldsymbol{\theta}}$, that is, $\hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)$ for $n^{-1}\boldsymbol{\Gamma}$. Therefore, the sampling variance-covariance matrix of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ is given by

$$\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) \approx \mathcal{N}\left(\mathbf{g}\left(\boldsymbol{\theta}\right), \mathbf{J}\hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)\mathbf{J}'\right).$$

Value

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

call Function call.

args Function arguments.

fun Function used ("DeltaMed").

output A list the length of which is equal to the length of `delta_t`.

Each element in the `output` list has the following elements:

delta_t Time interval.

jacobian Jacobian matrix.

est Estimated total, direct, and indirect effects.

vcov Sampling variance-covariance matrix of the estimated total, direct, and indirect effects.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
```

```
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

# Specific time interval ----------------------------------------------------
DeltaMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m"
)

# Range of time intervals ---------------------------------------------------
delta <- DeltaMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)
```

```
plot(delta)

# Methods -----------------------------------------------------------------
# DeltaMed has a number of methods including
# print, summary, confint, and plot
print(delta)
summary(delta)
confint(delta, level = 0.95)
plot(delta)
```

DeltaMedStd *Delta Method Sampling Variance-Covariance Matrix for the Standardized Total, Direct, and Indirect Effects of X on Y Through M Over a Specific Time Interval or a Range of Time Intervals*

Description

This function computes the delta method sampling variance-covariance matrix for the standardized total, direct, and indirect effects of the independent variable X on the dependent variable Y through mediator variables $\mathbf{m}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model's drift matrix $\boldsymbol{\Phi}$ and process noise covariance matrix $\boldsymbol{\Sigma}$.

Usage

```
DeltaMedStd(
  phi,
  sigma,
  vcov_theta,
  delta_t,
  from,
  to,
  med,
  ncores = NULL,
  tol = 0.01
)
```

Arguments

phi	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). phi should have row and column names pertaining to the variables in the system.
sigma	Numeric matrix. The process noise covariance matrix ($\boldsymbol{\Sigma}$).
vcov_theta	Numeric matrix. The sampling variance-covariance matrix of $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$ and $\mathrm{vech}\left(\boldsymbol{\Sigma}\right)$
delta_t	Numeric. Time interval (Δt).
from	Character string. Name of the independent variable X in phi.

`to`	Character string. Name of the dependent variable Y in `phi`.
`med`	Character vector. Name/s of the mediator variable/s in `phi`.
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of replications `R` is a large value.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `TotalStd()`, `DirectStd()`, and `IndirectStd()` for more details.

Delta Method:

Let $\boldsymbol{\theta}$ be a vector that combines $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$, that is, the elements of the $\boldsymbol{\Phi}$ matrix in vector form sorted column-wise and $\mathrm{vech}\left(\boldsymbol{\Sigma}\right)$, that is, the unique elements of the $\boldsymbol{\Sigma}$ matrix in vector form sorted column-wise. Let $\hat{\boldsymbol{\theta}}$ be a vector that combines $\mathrm{vec}\left(\hat{\boldsymbol{\Phi}}\right)$ and $\mathrm{vech}\left(\hat{\boldsymbol{\Sigma}}\right)$. By the multivariate central limit theory, the function $\mathbf{g}$ using $\hat{\boldsymbol{\theta}}$ as input can be expressed as:

$$\sqrt{n}\left(\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) - \mathbf{g}\left(\boldsymbol{\theta}\right)\right) \xrightarrow{\mathrm{D}} \mathcal{N}\left(0, \mathbf{J}\boldsymbol{\Gamma}\mathbf{J}^{\prime}\right)$$

where $\mathbf{J}$ is the matrix of first-order derivatives of the function $\mathbf{g}$ with respect to the elements of $\boldsymbol{\theta}$ and $\boldsymbol{\Gamma}$ is the asymptotic variance-covariance matrix of $\hat{\boldsymbol{\theta}}$.

From the former, we can derive the distribution of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ as follows:

$$\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) \approx \mathcal{N}\left(\mathbf{g}\left(\boldsymbol{\theta}\right), n^{-1}\mathbf{J}\boldsymbol{\Gamma}\mathbf{J}^{\prime}\right)$$

The uncertainty associated with the estimator $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ is, therefore, given by $n^{-1}\mathbf{J}\boldsymbol{\Gamma}\mathbf{J}^{\prime}$. When $\boldsymbol{\Gamma}$ is unknown, by substitution, we can use the estimated sampling variance-covariance matrix of $\hat{\boldsymbol{\theta}}$, that is, $\hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)$ for $n^{-1}\boldsymbol{\Gamma}$. Therefore, the sampling variance-covariance matrix of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ is given by

$$\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) \approx \mathcal{N}\left(\mathbf{g}\left(\boldsymbol{\theta}\right), \mathbf{J}\hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)\mathbf{J}^{\prime}\right).$$

Value

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

call Function call.

args Function arguments.

fun Function used ("DeltaMedStd").

output A list the length of which is equal to the length of `delta_t`.

Each element in the `output` list has the following elements:

delta_t Time interval.

jacobian Jacobian matrix.

est Estimated standardized total, direct, and indirect effects.

vcov Sampling variance-covariance matrix of the estimated standardized total, direct, and indirect effects.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaTotalCentral(), Direct(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
sigma <- matrix(
  data = c(
    0.24455556, 0.02201587, -0.05004762,
    0.02201587, 0.07067800, 0.01539456,
    -0.05004762, 0.01539456, 0.07553061
  ),
  nrow = 3
)
vcov_theta <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151, -0.00600, -0.00033,
    0.00110, 0.00324, 0.00020, -0.00061, -0.00115,
    0.00011, 0.00015, 0.00001, -0.00002, -0.00001,
    0.00040, 0.00374, 0.00016, -0.00022, -0.00273,
    -0.00016, 0.00009, 0.00150, 0.00012, -0.00010,
    -0.00026, 0.00002, 0.00012, 0.00004, -0.00001,
    -0.00151, 0.00016, 0.00389, 0.00103, -0.00007,
```

```
    -0.00283, -0.00050, 0.00000, 0.00156, 0.00021,
    -0.00005, -0.00031, 0.00001, 0.00007, 0.00006,
    -0.00600, -0.00022, 0.00103, 0.00644, 0.00031,
    -0.00119, -0.00374, -0.00021, 0.00070, 0.00064,
    -0.00015, -0.00005, 0.00000, 0.00003, -0.00001,
    -0.00033, -0.00273, -0.00007, 0.00031, 0.00287,
    0.00013, -0.00014, -0.00170, -0.00012, 0.00006,
    0.00014, -0.00001, -0.00015, 0.00000, 0.00001,
    0.00110, -0.00016, -0.00283, -0.00119, 0.00013,
    0.00297, 0.00063, -0.00004, -0.00177, -0.00013,
    0.00005, 0.00017, -0.00002, -0.00008, 0.00001,
    0.00324, 0.00009, -0.00050, -0.00374, -0.00014,
    0.00063, 0.00495, 0.00024, -0.00093, -0.00020,
    0.00006, -0.00010, 0.00000, -0.00001, 0.00004,
    0.00020, 0.00150, 0.00000, -0.00021, -0.00170,
    -0.00004, 0.00024, 0.00214, 0.00012, -0.00002,
    -0.00004, 0.00000, 0.00006, -0.00005, -0.00001,
    -0.00061, 0.00012, 0.00156, 0.00070, -0.00012,
    -0.00177, -0.00093, 0.00012, 0.00223, 0.00004,
    -0.00002, -0.00003, 0.00001, 0.00003, -0.00013,
    -0.00115, -0.00010, 0.00021, 0.00064, 0.00006,
    -0.00013, -0.00020, -0.00002, 0.00004, 0.00057,
    0.00001, -0.00009, 0.00000, 0.00000, 0.00001,
    0.00011, -0.00026, -0.00005, -0.00015, 0.00014,
    0.00005, 0.00006, -0.00004, -0.00002, 0.00001,
    0.00012, 0.00001, 0.00000, -0.00002, 0.00000,
    0.00015, 0.00002, -0.00031, -0.00005, -0.00001,
    0.00017, -0.00010, 0.00000, -0.00003, -0.00009,
    0.00001, 0.00014, 0.00000, 0.00000, -0.00005,
    0.00001, 0.00012, 0.00001, 0.00000, -0.00015,
    -0.00002, 0.00000, 0.00006, 0.00001, 0.00000,
    0.00000, 0.00000, 0.00010, 0.00001, 0.00000,
    -0.00002, 0.00004, 0.00007, 0.00003, 0.00000,
    -0.00008, -0.00001, -0.00005, 0.00003, 0.00000,
    -0.00002, 0.00000, 0.00001, 0.00005, 0.00001,
    -0.00001, -0.00001, 0.00006, -0.00001, 0.00001,
    0.00001, 0.00004, -0.00001, -0.00013, 0.00001,
    0.00000, -0.00005, 0.00000, 0.00001, 0.00012
  ),
  nrow = 15
)

# Specific time interval ----------------------------------------------------
DeltaMedStd(
  phi = phi,
  sigma = sigma,
  vcov_theta = vcov_theta,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m"
)
```

```
# Range of time intervals ---------------------------------------------------
delta <- DeltaMedStd(
  phi = phi,
  sigma = sigma,
  vcov_theta = vcov_theta,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)
plot(delta)

# Methods -------------------------------------------------------------------
# DeltaMedStd has a number of methods including
# print, summary, confint, and plot
print(delta)
summary(delta)
confint(delta, level = 0.95)
plot(delta)
```

`DeltaTotalCentral` *Delta Method Sampling Variance-Covariance Matrix for the Total Effect Centrality Over a Specific Time Interval or a Range of Time Intervals*

Description

This function computes the delta method sampling variance-covariance matrix for the total effect centrality over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model's drift matrix $\boldsymbol{\Phi}$.

Usage

```
DeltaTotalCentral(phi, vcov_phi_vec, delta_t, ncores = NULL, tol = 0.01)
```

Arguments

`phi`	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). `phi` should have row and column names pertaining to the variables in the system.
`vcov_phi_vec`	Numeric matrix. The sampling variance-covariance matrix of $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$.
`delta_t`	Vector of positive numbers. Time interval (Δt).
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when the length of `delta_t` is long.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `TotalCentral()` more details.

Delta Method:

Let $\boldsymbol{\theta}$ be $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$, that is, the elements of the $\boldsymbol{\Phi}$ matrix in vector form sorted column-wise. Let $\hat{\boldsymbol{\theta}}$ be $\mathrm{vec}\left(\hat{\boldsymbol{\Phi}}\right)$. By the multivariate central limit theory, the function $\mathbf{g}$ using $\hat{\boldsymbol{\theta}}$ as input can be expressed as:

$$\sqrt{n}\left(\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) - \mathbf{g}\left(\boldsymbol{\theta}\right)\right) \xrightarrow[]{\mathrm{D}} \mathcal{N}\left(0, \mathbf{J}\boldsymbol{\Gamma}\mathbf{J}^{\prime}\right)$$

where $\mathbf{J}$ is the matrix of first-order derivatives of the function $\mathbf{g}$ with respect to the elements of $\boldsymbol{\theta}$ and $\boldsymbol{\Gamma}$ is the asymptotic variance-covariance matrix of $\hat{\boldsymbol{\theta}}$.

From the former, we can derive the distribution of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ as follows:

$$\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) \approx \mathcal{N}\left(\mathbf{g}\left(\boldsymbol{\theta}\right), n^{-1}\mathbf{J}\boldsymbol{\Gamma}\mathbf{J}^{\prime}\right)$$

The uncertainty associated with the estimator $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ is, therefore, given by $n^{-1}\mathbf{J}\boldsymbol{\Gamma}\mathbf{J}^{\prime}$. When $\boldsymbol{\Gamma}$ is unknown, by substitution, we can use the estimated sampling variance-covariance matrix of $\hat{\boldsymbol{\theta}}$, that is, $\hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)$ for $n^{-1}\boldsymbol{\Gamma}$. Therefore, the sampling variance-covariance matrix of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ is given by

$$\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right) \approx \mathcal{N}\left(\mathbf{g}\left(\boldsymbol{\theta}\right), \mathbf{J}\hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)\mathbf{J}^{\prime}\right).$$

Value

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

call Function call.

args Function arguments.

fun Function used ("DeltaTotalCentral").

output A list the length of which is equal to the length of `delta_t`.

Each element in the `output` list has the following elements:

delta_t Time interval.

jacobian Jacobian matrix.

est Estimated total effect centrality.

vcov Sampling variance-covariance matrix of estimated total effect centrality.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), Direct(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
```

```
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

# Specific time interval ----------------------------------------------------
DeltaTotalCentral(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1
)

# Range of time intervals ---------------------------------------------------
delta <- DeltaTotalCentral(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5
)
plot(delta)

# Methods -------------------------------------------------------------------
# DeltaTotalCentral has a number of methods including
# print, summary, confint, and plot
print(delta)
summary(delta)
confint(delta, level = 0.95)
plot(delta)
```

Direct *Direct Effect of X on Y Over a Specific Time Interval*

Description

This function computes the direct effect of the independent variable X on the dependent variable Y through mediator variables $\mathbf{m}$ over a specific time interval Δt using the first-order stochastic differential equation model's drift matrix $\boldsymbol{\Phi}$.

Usage

```
Direct(phi, delta_t, from, to, med)
```

Arguments

phi	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). phi should have row and column names pertaining to the variables in the system.
delta_t	Numeric. Time interval (Δt).
from	Character string. Name of the independent variable X in phi.
to	Character string. Name of the dependent variable Y in phi.
med	Character vector. Name/s of the mediator variable/s in phi.

Details

The direct effect of the independent variable X on the dependent variable Y relative to some mediator variables $\mathbf{m}$ is given by

$$\mathrm{Direct}_{\Delta t_{i,j}} = \exp\left(\Delta t \mathbf{D} \boldsymbol{\Phi} \mathbf{D}\right)_{i,j}$$

where $\boldsymbol{\Phi}$ denotes the drift matrix, $\mathbf{D}$ a diagonal matrix where the diagonal elements corresponding to mediator variables $\mathbf{m}$ are set to zero and the rest to one, i the row index of Y in $\boldsymbol{\Phi}$, j the column index of X in $\boldsymbol{\Phi}$, and Δt the time interval.

Linear Stochastic Differential Equation Model:

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

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.

Value

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

call Function call.

args Function arguments.

fun Function used ("Direct").

output The direct effect.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
delta_t <- 1
Direct(
  phi = phi,
  delta_t = delta_t,
  from = "x",
  to = "y",
  med = "m"
)
phi <- matrix(
  data = c(
    -6, 5.5, 0, 0,
    1.25, -2.5, 5.9, -7.3,
    0, 0, -6, 2.5,
    5, 0, 0, -6
  ),
  nrow = 4
)
```

```
colnames(phi) <- rownames(phi) <- paste0("y", 1:4)
Direct(
  phi = phi,
  delta_t = delta_t,
  from = "y2",
  to = "y4",
  med = c("y1", "y3")
)
```

DirectStd *Standardized Direct Effect of X on Y Over a Specific Time Interval*

Description

This function computes the standardized direct effect of the independent variable X on the dependent variable Y through mediator variables $\mathbf{m}$ over a specific time interval Δt using the first-order stochastic differential equation model's drift matrix $\boldsymbol{\Phi}$ and process noise covariance matrix $\boldsymbol{\Sigma}$.

Usage

```
DirectStd(phi, sigma, delta_t, from, to, med)
```

Arguments

phi	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). phi should have row and column names pertaining to the variables in the system.
sigma	Numeric matrix. The process noise covariance matrix ($\boldsymbol{\Sigma}$).
delta_t	Numeric. Time interval (Δt).
from	Character string. Name of the independent variable X in phi.
to	Character string. Name of the dependent variable Y in phi.
med	Character vector. Name/s of the mediator variable/s in phi.

Details

The standardized direct effect of the independent variable X on the dependent variable Y relative to some mediator variables $\mathbf{m}$ is given by

$$\mathrm{Direct}^{*}_{\Delta t_{i,j}} = \mathrm{Direct}_{\Delta t_{i,j}} \left(\frac{\sigma_{x_j}}{\sigma_{y_i}} \right)$$

where $\boldsymbol{\Phi}$ denotes the drift matrix, σ_{x_j} and σ_{y_i} are the steady-state model-implied standard deviations of the state independent and dependent variables, respectively, and Δt the time interval.

Value

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

call Function call.

args Function arguments.

fun Function used ("DirectStd").

output The standardized direct effect.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
sigma <- matrix(
  data = c(
    0.24455556, 0.02201587, -0.05004762,
    0.02201587, 0.07067800, 0.01539456,
    -0.05004762, 0.01539456, 0.07553061
  ),
  nrow = 3
```

```
)
delta_t <- 1
DirectStd(
  phi = phi,
  sigma = sigma,
  delta_t = delta_t,
  from = "x",
  to = "y",
  med = "m"
)
```

Indirect *Indirect Effect of X on Y Through M Over a Specific Time Interval*

Description

This function computes the indirect effect of the independent variable X on the dependent variable Y through mediator variables $\mathbf{m}$ over a specific time interval Δt using the first-order stochastic differential equation model's drift matrix $\boldsymbol{\Phi}$.

Usage

```
Indirect(phi, delta_t, from, to, med)
```

Arguments

phi	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). phi should have row and column names pertaining to the variables in the system.
delta_t	Numeric. Time interval (Δt).
from	Character string. Name of the independent variable X in phi.
to	Character string. Name of the dependent variable Y in phi.
med	Character vector. Name/s of the mediator variable/s in phi.

Details

The indirect effect of the independent variable X on the dependent variable Y relative to some mediator variables $\mathbf{m}$ over a specific time interval Δt is given by

$$\mathrm{Indirect}_{\Delta t_{i,j}} = \exp\left(\Delta t \boldsymbol{\Phi}\right)_{i,j} - \exp\left(\Delta t \mathbf{D}_{\mathbf{m}} \boldsymbol{\Phi} \mathbf{D}_{\mathbf{m}}\right)_{i,j}$$

where $\boldsymbol{\Phi}$ denotes the drift matrix, $\mathbf{D}_{\mathbf{m}}$ a matrix where the off diagonal elements are zeros and the diagonal elements are zero for the index/indices of mediator variables $\mathbf{m}$ and one otherwise, i the row index of Y in $\boldsymbol{\Phi}$, j the column index of X in $\boldsymbol{\Phi}$, and Δt the time interval.

Linear Stochastic Differential Equation Model:
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 $\left(\boldsymbol{\Theta}^{\frac{1}{2}}\right)\left(\boldsymbol{\Theta}^{\frac{1}{2}}\right)^{\prime} = \boldsymbol{\Theta}$.
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.

Value

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

call Function call.

args Function arguments.

fun Function used ("Indirect").

output The indirect effect.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), Direct(), DirectStd(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
delta_t <- 1
Indirect(
  phi = phi,
  delta_t = delta_t,
  from = "x",
  to = "y",
  med = "m"
)
phi <- matrix(
  data = c(
    -6, 5.5, 0, 0,
    1.25, -2.5, 5.9, -7.3,
    0, 0, -6, 2.5,
    5, 0, 0, -6
  ),
  nrow = 4
)
colnames(phi) <- rownames(phi) <- paste0("y", 1:4)
Indirect(
  phi = phi,
  delta_t = delta_t,
  from = "y2",
  to = "y4",
  med = c("y1", "y3")
)
```

IndirectCentral   *Indirect Effect Centrality*

Description

Indirect Effect Centrality

Usage

```
IndirectCentral(phi, delta_t, tol = 0.01)
```

Arguments

phi	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). phi should have row and column names pertaining to the variables in the system.
delta_t	Vector of positive numbers. Time interval (Δt).
tol	Numeric. Smallest possible time interval to allow.

Details

Indirect effect centrality is the sum of all possible indirect effects between different pairs of variables in which a specific variable serves as the only mediator.

Value

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

call Function call.

args Function arguments.

fun Function used ("IndirectCentral").

output A matrix of indirect effect centrality.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), Direct(), DirectStd(), Indirect(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")

# Specific time interval ----------------------------------------------------
IndirectCentral(
  phi = phi,
  delta_t = 1
)

# Range of time intervals ---------------------------------------------------
indirect_central <- IndirectCentral(
  phi = phi,
  delta_t = 1:30
)
plot(indirect_central)

# Methods -------------------------------------------------------------------
# IndirectCentral has a number of methods including
# print, summary, and plot
indirect_central <- IndirectCentral(
  phi = phi,
  delta_t = 1:5
)
print(indirect_central)
summary(indirect_central)
plot(indirect_central)
```

IndirectStd — *Standardized Indirect Effect of X on Y Through M Over a Specific Time Interval*

Description

This function computes the standardized indirect effect of the independent variable X on the dependent variable Y through mediator variables $\mathbf{m}$ over a specific time interval Δt using the first-order stochastic differential equation model's drift matrix $\boldsymbol{\Phi}$ and process noise covariance matrix $\boldsymbol{\Sigma}$.

Usage

```
IndirectStd(phi, sigma, delta_t, from, to, med)
```

Arguments

`phi`	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). `phi` should have row and column names pertaining to the variables in the system.
`sigma`	Numeric matrix. The process noise covariance matrix ($\boldsymbol{\Sigma}$).
`delta_t`	Numeric. Time interval (Δt).
`from`	Character string. Name of the independent variable X in `phi`.
`to`	Character string. Name of the dependent variable Y in `phi`.
`med`	Character vector. Name/s of the mediator variable/s in `phi`.

Details

The standardized indirect effect of the independent variable X on the dependent variable Y relative to some mediator variables $\mathbf{m}$ over a specific time interval Δt is given by

$$\mathrm{Indirect}^{*}_{\Delta t_{i,j}} = \mathrm{Total}^{*}_{\Delta t_{i,j}} - \mathrm{Direct}^{*}_{\Delta t_{i,j}}$$

where $\mathrm{Total}^{*}_{\Delta t}$ and $\mathrm{Direct}^{*}_{\Delta t}$ are standardized total and direct effects for time interval Δt.

Value

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

call Function call.

args Function arguments.

fun Function used ("IndirectStd").

output The standardized indirect effect.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
sigma <- matrix(
  data = c(
    0.24455556, 0.02201587, -0.05004762,
    0.02201587, 0.07067800, 0.01539456,
    -0.05004762, 0.01539456, 0.07553061
  ),
  nrow = 3
)
delta_t <- 1
IndirectStd(
  phi = phi,
  sigma = sigma,
  delta_t = delta_t,
  from = "x",
  to = "y",
  med = "m"
)
```

MCBeta	*Monte Carlo Sampling Distribution for the Elements of the Matrix of Lagged Coefficients Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a Monte Carlo method sampling distribution for the elements of the matrix of lagged coefficients $\boldsymbol{\beta}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$.

Usage

```
MCBeta(
  phi,
  vcov_phi_vec,
  delta_t,
  R,
  test_phi = TRUE,
  ncores = NULL,
  seed = NULL,
  tol = 0.01
)
```

Arguments

`phi`	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). `phi` should have row and column names pertaining to the variables in the system.
`vcov_phi_vec`	Numeric matrix. The sampling variance-covariance matrix of $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$.
`delta_t`	Numeric. Time interval (Δt).
`R`	Positive integer. Number of replications.
`test_phi`	Logical. If `test_phi = TRUE`, the function tests the stability of the generated drift matrix $\boldsymbol{\Phi}$. If the test returns `FALSE`, the function generates a new drift matrix $\boldsymbol{\Phi}$ and runs the test recursively until the test returns `TRUE`.
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of replications `R` is a large value.
`seed`	Random seed.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `Total()`.

Monte Carlo Method:

Let $\boldsymbol{\theta}$ be $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$, that is, the elements of the $\boldsymbol{\Phi}$ matrix in vector form sorted column-wise. Let $\hat{\boldsymbol{\theta}}$ be $\mathrm{vec}\left(\hat{\boldsymbol{\Phi}}\right)$. Based on the asymptotic properties of maximum likelihood estimators, we can assume that estimators are normally distributed around the population parameters.

$$\hat{\boldsymbol{\theta}} \sim \mathcal{N}\left(\boldsymbol{\theta}, \mathbb{V}\left(\hat{\boldsymbol{\theta}}\right)\right)$$

Using this distributional assumption, a sampling distribution of $\hat{\boldsymbol{\theta}}$ which we refer to as $\hat{\boldsymbol{\theta}}^{*}$ can be generated by replacing the population parameters with sample estimates, that is,

$$\hat{\boldsymbol{\theta}}^{*} \sim \mathcal{N}\left(\hat{\boldsymbol{\theta}}, \hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)\right).$$

Let $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ be a parameter that is a function of the estimated parameters. A sampling distribution of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$, which we refer to as $\mathbf{g}\left(\hat{\boldsymbol{\theta}}^{*}\right)$, can be generated by using the simulated estimates to calculate $\mathbf{g}$. The standard deviations of the simulated estimates are the standard errors. Percentiles corresponding to $100\left(1 - \alpha\right)\%$ are the confidence intervals.

Value

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

call Function call.

args Function arguments.

fun Function used ("MCBeta").

output A list the length of which is equal to the length of `delta_t`.

Each element in the `output` list has the following elements:

est Estimated elements of the matrix of lagged coefficients.

thetahatstar A matrix of Monte Carlo elements of the matrix of lagged coefficients.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), Direct(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
set.seed(42)
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)
```

```
# Specific time interval ----------------------------------------------------
MCBeta(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1,
  R = 100L # use a large value for R in actual research
)

# Range of time intervals ---------------------------------------------------
mc <- MCBeta(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5,
  R = 100L # use a large value for R in actual research
)
plot(mc)

# Methods -------------------------------------------------------------------
# MCBeta has a number of methods including
# print, summary, confint, and plot
print(mc)
summary(mc)
confint(mc, level = 0.95)
plot(mc)
```

MCBetaStd — *Monte Carlo Sampling Distribution for the Elements of the Standardized Matrix of Lagged Coefficients Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a Monte Carlo method sampling distribution for the elements of the standardized matrix of lagged coefficients $\boldsymbol{\beta}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$ and process noise covariance matrix $\boldsymbol{\Sigma}$.

Usage

```
MCBetaStd(
  phi,
  sigma,
  vcov_theta,
  delta_t,
  R,
  test_phi = TRUE,
  ncores = NULL,
```

```
    seed = NULL,
    tol = 0.01
)
```

Arguments

phi	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). phi should have row and column names pertaining to the variables in the system.
sigma	Numeric matrix. The process noise covariance matrix ($\boldsymbol{\Sigma}$).
vcov_theta	Numeric matrix. The sampling variance-covariance matrix of $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$ and $\mathrm{vech}\left(\boldsymbol{\Sigma}\right)$
delta_t	Numeric. Time interval (Δt).
R	Positive integer. Number of replications.
test_phi	Logical. If test_phi = TRUE, the function tests the stability of the generated drift matrix $\boldsymbol{\Phi}$. If the test returns FALSE, the function generates a new drift matrix $\boldsymbol{\Phi}$ and runs the test recursively until the test returns TRUE.
ncores	Positive integer. Number of cores to use. If ncores = NULL, use a single core. Consider using multiple cores when number of replications R is a large value.
seed	Random seed.
tol	Numeric. Smallest possible time interval to allow.

Details

See TotalStd().

Monte Carlo Method:

Let $\boldsymbol{\theta}$ be a vector that combines $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$, that is, the elements of the $\boldsymbol{\Phi}$ matrix in vector form sorted column-wise and $\mathrm{vech}\left(\boldsymbol{\Sigma}\right)$, that is, the unique elements of the $\boldsymbol{\Sigma}$ matrix in vector form sorted column-wise. Let $\hat{\boldsymbol{\theta}}$ be a vector that combines $\mathrm{vec}\left(\hat{\boldsymbol{\Phi}}\right)$ and $\mathrm{vech}\left(\hat{\boldsymbol{\Sigma}}\right)$. Based on the asymptotic properties of maximum likelihood estimators, we can assume that estimators are normally distributed around the population parameters.

$$\hat{\boldsymbol{\theta}} \sim \mathcal{N}\left(\boldsymbol{\theta}, \mathbb{V}\left(\hat{\boldsymbol{\theta}}\right)\right)$$

Using this distributional assumption, a sampling distribution of $\hat{\boldsymbol{\theta}}$ which we refer to as $\hat{\boldsymbol{\theta}}^{*}$ can be generated by replacing the population parameters with sample estimates, that is,

$$\hat{\boldsymbol{\theta}}^{*} \sim \mathcal{N}\left(\hat{\boldsymbol{\theta}}, \hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)\right).$$

Let $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ be a parameter that is a function of the estimated parameters. A sampling distribution of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$, which we refer to as $\mathbf{g}\left(\hat{\boldsymbol{\theta}}^{*}\right)$, can be generated by using the simulated estimates to calculate $\mathbf{g}$. The standard deviations of the simulated estimates are the standard errors. Percentiles corresponding to $100\left(1 - \alpha\right)\%$ are the confidence intervals.

Value

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

call Function call.

args Function arguments.

fun Function used ("MCBetaStd").

output A list the length of which is equal to the length of `delta_t`.

Each element in the `output` list has the following elements:

est Estimated elements of the standardized matrix of lagged coefficients.

thetahatstar A matrix of Monte Carlo elements of the standardized matrix of lagged coefficients.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
sigma <- matrix(
  data = c(
```

```
      0.24455556, 0.02201587, -0.05004762,
      0.02201587, 0.07067800, 0.01539456,
      -0.05004762, 0.01539456, 0.07553061
    ),
    nrow = 3
  )
  vcov_theta <- matrix(
    data = c(
      0.00843, 0.00040, -0.00151, -0.00600, -0.00033,
      0.00110, 0.00324, 0.00020, -0.00061, -0.00115,
      0.00011, 0.00015, 0.00001, -0.00002, -0.00001,
      0.00040, 0.00374, 0.00016, -0.00022, -0.00273,
      -0.00016, 0.00009, 0.00150, 0.00012, -0.00010,
      -0.00026, 0.00002, 0.00012, 0.00004, -0.00001,
      -0.00151, 0.00016, 0.00389, 0.00103, -0.00007,
      -0.00283, -0.00050, 0.00000, 0.00156, 0.00021,
      -0.00005, -0.00031, 0.00001, 0.00007, 0.00006,
      -0.00600, -0.00022, 0.00103, 0.00644, 0.00031,
      -0.00119, -0.00374, -0.00021, 0.00070, 0.00064,
      -0.00015, -0.00005, 0.00000, 0.00003, -0.00001,
      -0.00033, -0.00273, -0.00007, 0.00031, 0.00287,
      0.00013, -0.00014, -0.00170, -0.00012, 0.00006,
      0.00014, -0.00001, -0.00015, 0.00000, 0.00001,
      0.00110, -0.00016, -0.00283, -0.00119, 0.00013,
      0.00297, 0.00063, -0.00004, -0.00177, -0.00013,
      0.00005, 0.00017, -0.00002, -0.00008, 0.00001,
      0.00324, 0.00009, -0.00050, -0.00374, -0.00014,
      0.00063, 0.00495, 0.00024, -0.00093, -0.00020,
      0.00006, -0.00010, 0.00000, -0.00001, 0.00004,
      0.00020, 0.00150, 0.00000, -0.00021, -0.00170,
      -0.00004, 0.00024, 0.00214, 0.00012, -0.00002,
      -0.00004, 0.00000, 0.00006, -0.00005, -0.00001,
      -0.00061, 0.00012, 0.00156, 0.00070, -0.00012,
      -0.00177, -0.00093, 0.00012, 0.00223, 0.00004,
      -0.00002, -0.00003, 0.00001, 0.00003, -0.00013,
      -0.00115, -0.00010, 0.00021, 0.00064, 0.00006,
      -0.00013, -0.00020, -0.00002, 0.00004, 0.00057,
      0.00001, -0.00009, 0.00000, 0.00000, 0.00001,
      0.00011, -0.00026, -0.00005, -0.00015, 0.00014,
      0.00005, 0.00006, -0.00004, -0.00002, 0.00001,
      0.00012, 0.00001, 0.00000, -0.00002, 0.00000,
      0.00015, 0.00002, -0.00031, -0.00005, -0.00001,
      0.00017, -0.00010, 0.00000, -0.00003, -0.00009,
      0.00001, 0.00014, 0.00000, 0.00000, -0.00005,
      0.00001, 0.00012, 0.00001, 0.00000, -0.00015,
      -0.00002, 0.00000, 0.00006, 0.00001, 0.00000,
      0.00000, 0.00000, 0.00010, 0.00001, 0.00000,
      -0.00002, 0.00004, 0.00007, 0.00003, 0.00000,
      -0.00008, -0.00001, -0.00005, 0.00003, 0.00000,
      -0.00002, 0.00000, 0.00001, 0.00005, 0.00001,
      -0.00001, -0.00001, 0.00006, -0.00001, 0.00001,
      0.00001, 0.00004, -0.00001, -0.00013, 0.00001,
      0.00000, -0.00005, 0.00000, 0.00001, 0.00012
```

```
  ),
  nrow = 15
)

# Specific time interval ----------------------------------------------------
MCBetaStd(
  phi = phi,
  sigma = sigma,
  vcov_theta = vcov_theta,
  delta_t = 1,
  R = 100L # use a large value for R in actual research
)

# Range of time intervals ---------------------------------------------------
mc <- MCBetaStd(
  phi = phi,
  sigma = sigma,
  vcov_theta = vcov_theta,
  delta_t = 1:5,
  R = 100L # use a large value for R in actual research
)
plot(mc)

# Methods -------------------------------------------------------------------
# MCBetaStd has a number of methods including
# print, summary, confint, and plot
print(mc)
summary(mc)
confint(mc, level = 0.95)
plot(mc)
```

`MCIndirectCentral` *Monte Carlo Sampling Distribution of Indirect Effect Centrality Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a Monte Carlo method sampling distribution of the indirect effect centrality at a particular time interval Δt using the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$.

Usage

```
MCIndirectCentral(
  phi,
  vcov_phi_vec,
  delta_t,
  R,
```

```
    test_phi = TRUE,
    ncores = NULL,
    seed = NULL,
    tol = 0.01
)
```

Arguments

`phi`	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). `phi` should have row and column names pertaining to the variables in the system.
`vcov_phi_vec`	Numeric matrix. The sampling variance-covariance matrix of $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$.
`delta_t`	Numeric. Time interval (Δt).
`R`	Positive integer. Number of replications.
`test_phi`	Logical. If `test_phi = TRUE`, the function tests the stability of the generated drift matrix $\boldsymbol{\Phi}$. If the test returns `FALSE`, the function generates a new drift matrix $\boldsymbol{\Phi}$ and runs the test recursively until the test returns `TRUE`.
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of replications `R` is a large value.
`seed`	Random seed.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `IndirectCentral()` for more details.

Monte Carlo Method:

Let $\boldsymbol{\theta}$ be $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$, that is, the elements of the $\boldsymbol{\Phi}$ matrix in vector form sorted column-wise. Let $\hat{\boldsymbol{\theta}}$ be $\mathrm{vec}\left(\hat{\boldsymbol{\Phi}}\right)$. Based on the asymptotic properties of maximum likelihood estimators, we can assume that estimators are normally distributed around the population parameters.

$$\hat{\boldsymbol{\theta}} \sim \mathcal{N}\left(\boldsymbol{\theta}, \mathbb{V}\left(\hat{\boldsymbol{\theta}}\right)\right)$$

Using this distributional assumption, a sampling distribution of $\hat{\boldsymbol{\theta}}$ which we refer to as $\hat{\boldsymbol{\theta}}^{*}$ can be generated by replacing the population parameters with sample estimates, that is,

$$\hat{\boldsymbol{\theta}}^{*} \sim \mathcal{N}\left(\hat{\boldsymbol{\theta}}, \hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)\right).$$

Let $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ be a parameter that is a function of the estimated parameters. A sampling distribution of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$, which we refer to as $\mathbf{g}\left(\hat{\boldsymbol{\theta}}^{*}\right)$, can be generated by using the simulated estimates to calculate $\mathbf{g}$. The standard deviations of the simulated estimates are the standard errors. Percentiles corresponding to $100\left(1 - \alpha\right)\%$ are the confidence intervals.

Value

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

call Function call.

args Function arguments.

fun Function used ("MCIndirectCentral").

output A list the length of which is equal to the length of `delta_t`.

Each element in the `output` list has the following elements:

est A vector of indirect effect centrality.

thetahatstar A matrix of Monte Carlo indirect effect centrality.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
set.seed(42)
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
```

```
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

# Specific time interval ----------------------------------------------------
MCIndirectCentral(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1,
  R = 100L # use a large value for R in actual research
)

# Range of time intervals ---------------------------------------------------
mc <- MCIndirectCentral(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5,
  R = 100L # use a large value for R in actual research
)
plot(mc)

# Methods -------------------------------------------------------------------
# MCIndirectCentral has a number of methods including
# print, summary, confint, and plot
print(mc)
```

```
summary(mc)
confint(mc, level = 0.95)
plot(mc)
```

MCMed *Monte Carlo Sampling Distribution of Total, Direct, and Indirect Effects of X on Y Through M Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a Monte Carlo method sampling distribution of the total, direct and indirect effects of the independent variable X on the dependent variable Y through mediator variables $\mathbf{m}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$.

Usage

```
MCMed(
  phi,
  vcov_phi_vec,
  delta_t,
  from,
  to,
  med,
  R,
  test_phi = TRUE,
  ncores = NULL,
  seed = NULL,
  tol = 0.01
)
```

Arguments

phi	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). phi should have row and column names pertaining to the variables in the system.
vcov_phi_vec	Numeric matrix. The sampling variance-covariance matrix of $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$.
delta_t	Numeric. Time interval (Δt).
from	Character string. Name of the independent variable X in phi.
to	Character string. Name of the dependent variable Y in phi.
med	Character vector. Name/s of the mediator variable/s in phi.
R	Positive integer. Number of replications.
test_phi	Logical. If test_phi = TRUE, the function tests the stability of the generated drift matrix $\boldsymbol{\Phi}$. If the test returns FALSE, the function generates a new drift matrix $\boldsymbol{\Phi}$ and runs the test recursively until the test returns TRUE.

ncores	Positive integer. Number of cores to use. If ncores = NULL, use a single core. Consider using multiple cores when number of replications R is a large value.
seed	Random seed.
tol	Numeric. Smallest possible time interval to allow.

Details

See Total(), Direct(), and Indirect() for more details.

Monte Carlo Method:

Let $\boldsymbol{\theta}$ be $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$, that is, the elements of the $\boldsymbol{\Phi}$ matrix in vector form sorted column-wise. Let $\hat{\boldsymbol{\theta}}$ be $\mathrm{vec}\left(\hat{\boldsymbol{\Phi}}\right)$. Based on the asymptotic properties of maximum likelihood estimators, we can assume that estimators are normally distributed around the population parameters.

$$\hat{\boldsymbol{\theta}} \sim \mathcal{N}\left(\boldsymbol{\theta}, \mathbb{V}\left(\hat{\boldsymbol{\theta}}\right)\right)$$

Using this distributional assumption, a sampling distribution of $\hat{\boldsymbol{\theta}}$ which we refer to as $\hat{\boldsymbol{\theta}}^{*}$ can be generated by replacing the population parameters with sample estimates, that is,

$$\hat{\boldsymbol{\theta}}^{*} \sim \mathcal{N}\left(\hat{\boldsymbol{\theta}}, \hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)\right).$$

Let $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ be a parameter that is a function of the estimated parameters. A sampling distribution of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$, which we refer to as $\mathbf{g}\left(\hat{\boldsymbol{\theta}}^{*}\right)$, can be generated by using the simulated estimates to calculate $\mathbf{g}$. The standard deviations of the simulated estimates are the standard errors. Percentiles corresponding to $100\left(1-\alpha\right)\%$ are the confidence intervals.

Value

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

call Function call.

args Function arguments.

fun Function used ("MCMed").

output A list with length of length(delta_t).

Each element in the output list has the following elements:

est A vector of total, direct, and indirect effects.

thetahatstar A matrix of Monte Carlo total, direct, and indirect effects.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), Direct(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
set.seed(42)
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
```

```
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

# Specific time interval ----------------------------------------------------
MCMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m",
  R = 100L # use a large value for R in actual research
)

# Range of time intervals ---------------------------------------------------
mc <- MCMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m",
  R = 100L # use a large value for R in actual research
)
plot(mc)

# Methods -------------------------------------------------------------------
# MCMed has a number of methods including
# print, summary, confint, and plot
print(mc)
summary(mc)
confint(mc, level = 0.95)
```

`MCMedStd` *Monte Carlo Sampling Distribution of Standardized Total, Direct, and Indirect Effects of X on Y Through M Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a Monte Carlo method sampling distribution of the standardized total, direct and indirect effects of the independent variable X on the dependent variable Y through mediator variables $\mathbf{m}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$ and process noise covariance matrix $\boldsymbol{\Sigma}$.

Usage

```
MCMedStd(
  phi,
  sigma,
  vcov_theta,
  delta_t,
  from,
  to,
  med,
  R,
  test_phi = TRUE,
  ncores = NULL,
  seed = NULL,
  tol = 0.01
)
```

Arguments

phi	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). phi should have row and column names pertaining to the variables in the system.
sigma	Numeric matrix. The process noise covariance matrix ($\boldsymbol{\Sigma}$).
vcov_theta	Numeric matrix. The sampling variance-covariance matrix of $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$ and $\mathrm{vech}\left(\boldsymbol{\Sigma}\right)$
delta_t	Numeric. Time interval (Δt).
from	Character string. Name of the independent variable X in phi.
to	Character string. Name of the dependent variable Y in phi.
med	Character vector. Name/s of the mediator variable/s in phi.
R	Positive integer. Number of replications.
test_phi	Logical. If test_phi = TRUE, the function tests the stability of the generated drift matrix $\boldsymbol{\Phi}$. If the test returns FALSE, the function generates a new drift matrix $\boldsymbol{\Phi}$ and runs the test recursively until the test returns TRUE.
ncores	Positive integer. Number of cores to use. If ncores = NULL, use a single core. Consider using multiple cores when number of replications R is a large value.
seed	Random seed.
tol	Numeric. Smallest possible time interval to allow.

Details

See `TotalStd()`, `DirectStd()`, and `IndirectStd()` for more details.

Monte Carlo Method:
Let $\boldsymbol{\theta}$ be a vector that combines $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$, that is, the elements of the $\boldsymbol{\Phi}$ matrix in vector form sorted column-wise and $\mathrm{vech}\left(\boldsymbol{\Sigma}\right)$, that is, the unique elements of the $\boldsymbol{\Sigma}$ matrix in vector form sorted column-wise. Let $\hat{\boldsymbol{\theta}}$ be a vector that combines $\mathrm{vec}\left(\hat{\boldsymbol{\Phi}}\right)$ and $\mathrm{vech}\left(\hat{\boldsymbol{\Sigma}}\right)$. Based on the asymptotic properties of maximum likelihood estimators, we can assume that estimators are normally distributed around the population parameters.

$$\hat{\boldsymbol{\theta}} \sim \mathcal{N}\left(\boldsymbol{\theta}, \mathbb{V}\left(\hat{\boldsymbol{\theta}}\right)\right)$$

Using this distributional assumption, a sampling distribution of $\hat{\boldsymbol{\theta}}$ which we refer to as $\hat{\boldsymbol{\theta}}^{\ast}$ can be generated by replacing the population parameters with sample estimates, that is,

$$\hat{\boldsymbol{\theta}}^{\ast} \sim \mathcal{N}\left(\hat{\boldsymbol{\theta}}, \hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)\right).$$

Let $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ be a parameter that is a function of the estimated parameters. A sampling distribution of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$, which we refer to as $\mathbf{g}\left(\hat{\boldsymbol{\theta}}^{\ast}\right)$, can be generated by using the simulated estimates to calculate $\mathbf{g}$. The standard deviations of the simulated estimates are the standard errors. Percentiles corresponding to $100\left(1 - \alpha\right)\%$ are the confidence intervals.

Value

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

call Function call.

args Function arguments.

fun Function used ("MCMedStd").

output A list with length of `length(delta_t)`.

Each element in the `output` list has the following elements:

est A vector of standardized total, direct, and indirect effects.

thetahatstar A matrix of Monte Carlo standardized total, direct, and indirect effects.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), Direct(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
sigma <- matrix(
  data = c(
    0.24455556, 0.02201587, -0.05004762,
    0.02201587, 0.07067800, 0.01539456,
    -0.05004762, 0.01539456, 0.07553061
  ),
  nrow = 3
)
vcov_theta <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151, -0.00600, -0.00033,
    0.00110, 0.00324, 0.00020, -0.00061, -0.00115,
    0.00011, 0.00015, 0.00001, -0.00002, -0.00001,
    0.00040, 0.00374, 0.00016, -0.00022, -0.00273,
    -0.00016, 0.00009, 0.00150, 0.00012, -0.00010,
    -0.00026, 0.00002, 0.00012, 0.00004, -0.00001,
    -0.00151, 0.00016, 0.00389, 0.00103, -0.00007,
    -0.00283, -0.00050, 0.00000, 0.00156, 0.00021,
    -0.00005, -0.00031, 0.00001, 0.00007, 0.00006,
    -0.00600, -0.00022, 0.00103, 0.00644, 0.00031,
    -0.00119, -0.00374, -0.00021, 0.00070, 0.00064,
    -0.00015, -0.00005, 0.00000, 0.00003, -0.00001,
    -0.00033, -0.00273, -0.00007, 0.00031, 0.00287,
    0.00013, -0.00014, -0.00170, -0.00012, 0.00006,
    0.00014, -0.00001, -0.00015, 0.00000, 0.00001,
    0.00110, -0.00016, -0.00283, -0.00119, 0.00013,
    0.00297, 0.00063, -0.00004, -0.00177, -0.00013,
    0.00005, 0.00017, -0.00002, -0.00008, 0.00001,
    0.00324, 0.00009, -0.00050, -0.00374, -0.00014,
    0.00063, 0.00495, 0.00024, -0.00093, -0.00020,
    0.00006, -0.00010, 0.00000, -0.00001, 0.00004,
    0.00020, 0.00150, 0.00000, -0.00021, -0.00170,
    -0.00004, 0.00024, 0.00214, 0.00012, -0.00002,
```

```
    -0.00004, 0.00000, 0.00006, -0.00005, -0.00001,
    -0.00061, 0.00012, 0.00156, 0.00070, -0.00012,
    -0.00177, -0.00093, 0.00012, 0.00223, 0.00004,
    -0.00002, -0.00003, 0.00001, 0.00003, -0.00013,
    -0.00115, -0.00010, 0.00021, 0.00064, 0.00006,
    -0.00013, -0.00020, -0.00002, 0.00004, 0.00057,
    0.00001, -0.00009, 0.00000, 0.00000, 0.00001,
    0.00011, -0.00026, -0.00005, -0.00015, 0.00014,
    0.00005, 0.00006, -0.00004, -0.00002, 0.00001,
    0.00012, 0.00001, 0.00000, -0.00002, 0.00000,
    0.00015, 0.00002, -0.00031, -0.00005, -0.00001,
    0.00017, -0.00010, 0.00000, -0.00003, -0.00009,
    0.00001, 0.00014, 0.00000, 0.00000, -0.00005,
    0.00001, 0.00012, 0.00001, 0.00000, -0.00015,
    -0.00002, 0.00000, 0.00006, 0.00001, 0.00000,
    0.00000, 0.00000, 0.00010, 0.00001, 0.00000,
    -0.00002, 0.00004, 0.00007, 0.00003, 0.00000,
    -0.00008, -0.00001, -0.00005, 0.00003, 0.00000,
    -0.00002, 0.00000, 0.00001, 0.00005, 0.00001,
    -0.00001, -0.00001, 0.00006, -0.00001, 0.00001,
    0.00001, 0.00004, -0.00001, -0.00013, 0.00001,
    0.00000, -0.00005, 0.00000, 0.00001, 0.00012
  ),
  nrow = 15
)

# Specific time interval ----------------------------------------------------
MCMedStd(
  phi = phi,
  sigma = sigma,
  vcov_theta = vcov_theta,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m",
  R = 100L # use a large value for R in actual research
)

# Range of time intervals ---------------------------------------------------
mc <- MCMedStd(
  phi = phi,
  sigma = sigma,
  vcov_theta = vcov_theta,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m",
  R = 100L # use a large value for R in actual research
)
plot(mc)

# Methods -------------------------------------------------------------------
# MCMedStd has a number of methods including
```

```
# print, summary, confint, and plot
print(mc)
summary(mc)
confint(mc, level = 0.95)
```

MCPhi *Generate Random Drift Matrices Using the Monte Carlo Method*

Description

This function generates random drift matrices $\boldsymbol{\Phi}$ using the Monte Carlo method.

Usage

```
MCPhi(phi, vcov_phi_vec, R, test_phi = TRUE, ncores = NULL, seed = NULL)
```

Arguments

`phi`	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). `phi` should have row and column names pertaining to the variables in the system.
`vcov_phi_vec`	Numeric matrix. The sampling variance-covariance matrix of $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$.
`R`	Positive integer. Number of replications.
`test_phi`	Logical. If `test_phi = TRUE`, the function tests the stability of the generated drift matrix $\boldsymbol{\Phi}$. If the test returns `FALSE`, the function generates a new drift matrix $\boldsymbol{\Phi}$ and runs the test recursively until the test returns `TRUE`.
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of replications `R` is a large value.
`seed`	Random seed.

Details

Monte Carlo Method:

Let $\boldsymbol{\theta}$ be $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$, that is, the elements of the $\boldsymbol{\Phi}$ matrix in vector form sorted column-wise. Let $\hat{\boldsymbol{\theta}}$ be $\mathrm{vec}\left(\hat{\boldsymbol{\Phi}}\right)$. Based on the asymptotic properties of maximum likelihood estimators, we can assume that estimators are normally distributed around the population parameters.

$$\hat{\boldsymbol{\theta}} \sim \mathcal{N}\left(\boldsymbol{\theta}, \mathbb{V}\left(\hat{\boldsymbol{\theta}}\right)\right)$$

Using this distributional assumption, a sampling distribution of $\hat{\boldsymbol{\theta}}$ which we refer to as $\hat{\boldsymbol{\theta}}^{*}$ can be generated by replacing the population parameters with sample estimates, that is,

$$\hat{\boldsymbol{\theta}}^{*} \sim \mathcal{N}\left(\hat{\boldsymbol{\theta}}, \hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)\right).$$

Value

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

call Function call.

args Function arguments.

fun Function used ("MCPhi").

output A list simulated drift matrices.

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
set.seed(42)
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
MCPhi(
  phi = phi,
  vcov_phi_vec = 0.1 * diag(9),
  R = 100L # use a large value for R in actual research
)
phi <- matrix(
  data = c(
    -6, 5.5, 0, 0,
    1.25, -2.5, 5.9, -7.3,
    0, 0, -6, 2.5,
    5, 0, 0, -6
  ),
  nrow = 4
)
colnames(phi) <- rownames(phi) <- paste0("y", 1:4)
MCPhi(
  phi = phi,
```

```
  vcov_phi_vec = 0.1 * diag(16),
  R = 100L, # use a large value for R in actual research
  test_phi = FALSE
)
```

MCPhiSigma *Generate Random Drift Matrices and Process Noise Covariance Matrices Using the Monte Carlo Method*

Description

This function generates random drift matrices $\boldsymbol{\Phi}$ and process noise covariabces matrices $\boldsymbol{\Sigma}$ using the Monte Carlo method.

Usage

```
MCPhiSigma(
  phi,
  sigma,
  vcov_theta,
  R,
  test_phi = TRUE,
  ncores = NULL,
  seed = NULL
)
```

Arguments

phi	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). phi should have row and column names pertaining to the variables in the system.
sigma	Numeric matrix. The process noise covariance matrix ($\boldsymbol{\Sigma}$).
vcov_theta	Numeric matrix. The sampling variance-covariance matrix of $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$ and $\mathrm{vech}\left(\boldsymbol{\Sigma}\right)$
R	Positive integer. Number of replications.
test_phi	Logical. If test_phi = TRUE, the function tests the stability of the generated drift matrix $\boldsymbol{\Phi}$. If the test returns FALSE, the function generates a new drift matrix $\boldsymbol{\Phi}$ and runs the test recursively until the test returns TRUE.
ncores	Positive integer. Number of cores to use. If ncores = NULL, use a single core. Consider using multiple cores when number of replications R is a large value.
seed	Random seed.

Details

Monte Carlo Method:

Let $\boldsymbol{\theta}$ be a vector that combines $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$, that is, the elements of the $\boldsymbol{\Phi}$ matrix in vector form sorted column-wise and $\mathrm{vech}\left(\boldsymbol{\Sigma}\right)$, that is, the unique elements of the $\boldsymbol{\Sigma}$ matrix in vector form sorted column-wise. Let $\hat{\boldsymbol{\theta}}$ be a vector that combines $\mathrm{vec}\left(\hat{\boldsymbol{\Phi}}\right)$ and $\mathrm{vech}\left(\hat{\boldsymbol{\Sigma}}\right)$. Based on the asymptotic properties of maximum likelihood estimators, we can assume that estimators are normally distributed around the population parameters.

$$\hat{\boldsymbol{\theta}} \sim \mathcal{N}\left(\boldsymbol{\theta}, \mathbb{V}\left(\hat{\boldsymbol{\theta}}\right)\right)$$

Using this distributional assumption, a sampling distribution of $\hat{\boldsymbol{\theta}}$ which we refer to as $\hat{\boldsymbol{\theta}}^{*}$ can be generated by replacing the population parameters with sample estimates, that is,

$$\hat{\boldsymbol{\theta}}^{*} \sim \mathcal{N}\left(\hat{\boldsymbol{\theta}}, \hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)\right).$$

Value

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

call Function call.

args Function arguments.

fun Function used ("MCPhiSigma").

output A list simulated drift matrices.

Author(s)

Ivan Jacob Agaloos Pesigan

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
set.seed(42)
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
```

```
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
sigma <- matrix(
  data = c(
    0.24455556, 0.02201587, -0.05004762,
    0.02201587, 0.07067800, 0.01539456,
    -0.05004762, 0.01539456, 0.07553061
  ),
  nrow = 3
)
MCPhiSigma(
  phi = phi,
  sigma = sigma,
  vcov_theta = 0.1 * diag(15),
  R = 100L # use a large value for R in actual research
)
```

MCTotalCentral *Monte Carlo Sampling Distribution of Total Effect Centrality Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a Monte Carlo method sampling distribution of the total effect centrality at a particular time interval Δt using the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$.

Usage

```
MCTotalCentral(
  phi,
  vcov_phi_vec,
  delta_t,
  R,
  test_phi = TRUE,
  ncores = NULL,
  seed = NULL,
  tol = 0.01
)
```

Arguments

phi	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). phi should have row and column names pertaining to the variables in the system.
vcov_phi_vec	Numeric matrix. The sampling variance-covariance matrix of $\mathrm{vec}(\boldsymbol{\Phi})$.
delta_t	Numeric. Time interval (Δt).
R	Positive integer. Number of replications.

`test_phi`	Logical. If `test_phi = TRUE`, the function tests the stability of the generated drift matrix $\boldsymbol{\Phi}$. If the test returns `FALSE`, the function generates a new drift matrix $\boldsymbol{\Phi}$ and runs the test recursively until the test returns `TRUE`.
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of replications `R` is a large value.
`seed`	Random seed.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `TotalCentral()` for more details.

Monte Carlo Method:

Let $\boldsymbol{\theta}$ be $\mathrm{vec}\left(\boldsymbol{\Phi}\right)$, that is, the elements of the $\boldsymbol{\Phi}$ matrix in vector form sorted column-wise. Let $\hat{\boldsymbol{\theta}}$ be $\mathrm{vec}\left(\hat{\boldsymbol{\Phi}}\right)$. Based on the asymptotic properties of maximum likelihood estimators, we can assume that estimators are normally distributed around the population parameters.

$$\hat{\boldsymbol{\theta}} \sim \mathcal{N}\left(\boldsymbol{\theta}, \mathbb{V}\left(\hat{\boldsymbol{\theta}}\right)\right)$$

Using this distributional assumption, a sampling distribution of $\hat{\boldsymbol{\theta}}$ which we refer to as $\hat{\boldsymbol{\theta}}^{*}$ can be generated by replacing the population parameters with sample estimates, that is,

$$\hat{\boldsymbol{\theta}}^{*} \sim \mathcal{N}\left(\hat{\boldsymbol{\theta}}, \hat{\mathbb{V}}\left(\hat{\boldsymbol{\theta}}\right)\right).$$

Let $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$ be a parameter that is a function of the estimated parameters. A sampling distribution of $\mathbf{g}\left(\hat{\boldsymbol{\theta}}\right)$, which we refer to as $\mathbf{g}\left(\hat{\boldsymbol{\theta}}^{*}\right)$, can be generated by using the simulated estimates to calculate $\mathbf{g}$. The standard deviations of the simulated estimates are the standard errors. Percentiles corresponding to $100\left(1 - \alpha\right)\%$ are the confidence intervals.

Value

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

call Function call.

args Function arguments.

fun Function used ("MCTotalCentral").

output A list the length of which is equal to the length of `delta_t`.

Each element in the `output` list has the following elements:

est A vector of total effect centrality.

thetahatstar A matrix of Monte Carlo total effect centrality.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), Direct(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
set.seed(42)
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
```

```
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

# Specific time interval ----------------------------------------------------
MCTotalCentral(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1,
  R = 100L # use a large value for R in actual research
)

# Range of time intervals ---------------------------------------------------
mc <- MCTotalCentral(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5,
  R = 100L # use a large value for R in actual research
)
plot(mc)

# Methods -------------------------------------------------------------------
# MCTotalCentral has a number of methods including
# print, summary, confint, and plot
print(mc)
summary(mc)
confint(mc, level = 0.95)
plot(mc)
```

Med	*Total, Direct, and Indirect Effects of X on Y Through M Over a Specific Time Interval or a Range of Time Intervals*

Description

This function computes the total, direct, and indirect effects of the independent variable X on the dependent variable Y through mediator variables $\mathbf{m}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model's drift matrix $\boldsymbol{\Phi}$.

Usage

```
Med(phi, delta_t, from, to, med, tol = 0.01)
```

Arguments

phi	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). phi should have row and column names pertaining to the variables in the system.
delta_t	Vector of positive numbers. Time interval (Δt).
from	Character string. Name of the independent variable X in phi.
to	Character string. Name of the dependent variable Y in phi.
med	Character vector. Name/s of the mediator variable/s in phi.
tol	Numeric. Smallest possible time interval to allow.

Details

See Total(), Direct(), and Indirect() for more details.

Linear Stochastic Differential Equation Model:
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 $\left(\boldsymbol{\Theta}^{\frac{1}{2}}\right)\left(\boldsymbol{\Theta}^{\frac{1}{2}}\right)^{\prime} = \boldsymbol{\Theta}$.
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.

Value

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

call Function call.

args Function arguments.

fun Function used ("Med").

output A matrix of total, direct, and indirect effects.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), Direct(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")

# Specific time interval ----------------------------------------------------
Med(
  phi = phi,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m"
)

# Range of time intervals ---------------------------------------------------
med <- Med(
  phi = phi,
  delta_t = 1:30,
  from = "x",
  to = "y",
  med = "m"
```

```
)
plot(med)

# Methods -------------------------------------------------------------------
# Med has a number of methods including
# print, summary, and plot
med <- Med(
  phi = phi,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)
print(med)
summary(med)
plot(med)
```

MedStd *Standardized Total, Direct, and Indirect Effects of X on Y Through M Over a Specific Time Interval or a Range of Time Intervals*

Description

This function computes the standardized total, direct, and indirect effects of the independent variable X on the dependent variable Y through mediator variables $\mathbf{m}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model's drift matrix $\boldsymbol{\Phi}$ and process noise covariance matrix $\boldsymbol{\Sigma}$.

Usage

```
MedStd(phi, sigma, delta_t, from, to, med, tol = 0.01)
```

Arguments

`phi`	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). `phi` should have row and column names pertaining to the variables in the system.
`sigma`	Numeric matrix. The process noise covariance matrix ($\boldsymbol{\Sigma}$).
`delta_t`	Numeric. Time interval (Δt).
`from`	Character string. Name of the independent variable X in `phi`.
`to`	Character string. Name of the dependent variable Y in `phi`.
`med`	Character vector. Name/s of the mediator variable/s in `phi`.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `TotalStd()`, `DirectStd()`, and `IndirectStd()` for more details.

Value

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

call Function call.

args Function arguments.

fun Function used ("MedStd").

output A standardized matrix of total, direct, and indirect effects.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
sigma <- matrix(
  data = c(
    0.24455556, 0.02201587, -0.05004762,
    0.02201587, 0.07067800, 0.01539456,
    -0.05004762, 0.01539456, 0.07553061
  ),
  nrow = 3
```

```
)

# Specific time interval ----------------------------------------------------
MedStd(
  phi = phi,
  sigma = sigma,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m"
)

# Range of time intervals ---------------------------------------------------
med <- MedStd(
  phi = phi,
  sigma = sigma,
  delta_t = 1:30,
  from = "x",
  to = "y",
  med = "m"
)
plot(med)

# Methods -------------------------------------------------------------------
# MedStd has a number of methods including
# print, summary, and plot
med <- MedStd(
  phi = phi,
  sigma = sigma,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)
print(med)
summary(med)
plot(med)
```

---

plot.ctmedboot *Plot Method for an Object of Class* ctmedboot

---

Description

Plot Method for an Object of Class ctmedboot

Usage

```
## S3 method for class 'ctmedboot'
plot(x, alpha = 0.05, col = NULL, type = "pc", ...)
```

Arguments

x	Object of class ctmedboot.
alpha	Numeric. Significance level
col	Character vector. Optional argument. Character vector of colors.
type	Charater string. Confidence interval type, that is, type = "pc" for percentile; type = "bc" for bias corrected.
...	Additional arguments.

Value

Displays plots of point estimates and confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
## Not run:
library(bootStateSpace)
# prepare parameters
## number of individuals
n <- 50
## time points
time <- 100
delta_t <- 0.10
## dynamic structure
p <- 3
mu0 <- rep(x = 0, times = p)
sigma0 <- matrix(
  data = c(
    1.0,
    0.2,
    0.2,
    0.2,
    1.0,
    0.2,
    0.2,
    0.2,
    1.0
  ),
  nrow = p
)
sigma0_l <- t(chol(sigma0))
mu <- rep(x = 0, times = p)
phi <- matrix(
  data = c(
    -0.357,
    0.771,
    -0.450,
```

```
    0.0,
    -0.511,
    0.729,
    0,
    0,
    -0.693
  ),
  nrow = p
)
sigma <- matrix(
  data = c(
    0.24455556,
    0.02201587,
    -0.05004762,
    0.02201587,
    0.07067800,
    0.01539456,
    -0.05004762,
    0.01539456,
    0.07553061
  ),
  nrow = p
)
sigma_l <- t(chol(sigma))
## measurement model
k <- 3
nu <- rep(x = 0, times = k)
lambda <- diag(k)
theta <- 0.2 * diag(k)
theta_l <- t(chol(theta))

boot <- PBSSMOUFixed(
  R = 1000L,
  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,
  ncores = parallel::detectCores() - 1,
  seed = 42
)
phi_hat <- phi
colnames(phi_hat) <- rownames(phi_hat) <- c("x", "m", "y")
phi <- extract(object = boot, what = "phi")
```

```
# Range of time intervals ---------------------------------------------------
boot <- BootMed(
  phi = phi,
  phi_hat = phi_hat,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)
confint(boot)
confint(boot, type = "bc") # bias-corrected

## End(Not run)
```

`plot.ctmeddelta` *Plot Method for an Object of Class* `ctmeddelta`

Description

Plot Method for an Object of Class `ctmeddelta`

Usage

```
## S3 method for class 'ctmeddelta'
plot(x, alpha = 0.05, col = NULL, ...)
```

Arguments

`x`	Object of class `ctmeddelta`.
`alpha`	Numeric. Significance level
`col`	Character vector. Optional argument. Character vector of colors.
`...`	Additional arguments.

Value

Displays plots of point estimates and confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

# Range of time intervals ---------------------------------------------------
delta <- DeltaMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)
```

```
plot(delta)
```

---

plot.ctmedmc *Plot Method for an Object of Class* ctmedmc

Description

Plot Method for an Object of Class `ctmedmc`

Usage

```
## S3 method for class 'ctmedmc'
plot(x, alpha = 0.05, col = NULL, ...)
```

Arguments

x	Object of class `ctmedmc`.
alpha	Numeric. Significance level
col	Character vector. Optional argument. Character vector of colors.
...	Additional arguments.

Value

Displays plots of point estimates and confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
set.seed(42)
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
```

```
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

# Range of time intervals ---------------------------------------------------
mc <- MCMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m",
  R = 100L # use a large value for R in actual research
)
plot(mc)
```

`plot.ctmedmed` *Plot Method for an Object of Class* `ctmedmed`

Description

Plot Method for an Object of Class `ctmedmed`

Usage

```
## S3 method for class 'ctmedmed'
plot(x, col = NULL, legend_pos = "topright", ...)
```

Arguments

`x`	Object of class `ctmedmed`.
`col`	Character vector. Optional argument. Character vector of colors.
`legend_pos`	Character vector. Optional argument. Legend position.
`...`	Additional arguments.

Value

Displays plots of point estimates and confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")

# Range of time intervals ---------------------------------------------------
med <- Med(
  phi = phi,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)
plot(med)
```

`plot.ctmedtraj` *Plot Method for an Object of Class* `ctmedtraj`

Description

Plot Method for an Object of Class `ctmedtraj`

Usage

```
## S3 method for class 'ctmedtraj'
plot(x, legend_pos = "topright", total = TRUE, ...)
```

Arguments

x	Object of class ctmedtraj.
legend_pos	Character vector. Optional argument. Legend position.
total	Logical. If total = TRUE, include the total effect trajectory. If total = FALSE, exclude the total effect trajectory.
...	Additional arguments.

Value

Displays trajectory plots of the effects.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")

traj <- Trajectory(
  mu0 = c(3, 3, -3),
  time = 150,
  phi = phi,
  med = "m"
)

plot(traj)
```

PosteriorBeta *Posterior Sampling Distribution for the Elements of the Matrix of Lagged Coefficients Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a posterior sampling distribution for the elements of the matrix of lagged coefficients $\boldsymbol{\beta}$ over a specific time interval Δt or a range of time intervals using the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$.

Usage

```
PosteriorBeta(phi, delta_t, ncores = NULL, tol = 0.01)
```

Arguments

`phi`	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). `phi` should have row and column names pertaining to the variables in the system.
`delta_t`	Numeric. Time interval (Δt).
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of replications `R` is a large value.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `Total()`.

Value

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

call Function call.

args Function arguments.

fun Function used ("PosteriorBeta").

output A list the length of which is equal to the length of `delta_t`.

Each element in the `output` list has the following elements:

est A vector of total, direct, and indirect effects.

thetahatstar A matrix of Monte Carlo total, direct, and indirect effects.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), Direct(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)
```

```
phi <- MCPhi(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  R = 1000L
)$output

# Specific time interval ----------------------------------------------------
PosteriorBeta(
  phi = phi,
  delta_t = 1
)

# Range of time intervals ---------------------------------------------------
posterior <- PosteriorBeta(
  phi = phi,
  delta_t = 1:5
)
plot(posterior)

# Methods -------------------------------------------------------------------
# PosteriorBeta has a number of methods including
# print, summary, confint, and plot
print(posterior)
summary(posterior)
confint(posterior, level = 0.95)
plot(posterior)
```

PosteriorIndirectCentral *Posterior Distribution of the Indirect Effect Centrality Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a posterior distribution of the indirect effect centrality over a specific time interval Δt or a range of time intervals using the posterior distribution of the first-order stochastic differential equation model drift matrix $\mathbf{\Phi}$.

Usage

```
PosteriorIndirectCentral(phi, delta_t, ncores = NULL, tol = 0.01)
```

Arguments

phi	List of numeric matrices. Each element of the list is a sample from the posterior distribution of the drift matrix ($\mathbf{\Phi}$). Each matrix should have row and column names pertaining to the variables in the system.
delta_t	Numeric. Time interval (Δt).

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

tol Numeric. Smallest possible time interval to allow.

Details

See `TotalCentral()` for more details.

Value

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

call Function call.

args Function arguments.

fun Function used ("PosteriorIndirectCentral").

output A list the length of which is equal to the length of `delta_t`.

Each element in the `output` list has the following elements:

est Mean of the posterior distribution of the total, direct, and indirect effects.

thetahatstar Posterior distribution of the total, direct, and indirect effects.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

phi <- MCPhi(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  R = 1000L
)$output

# Specific time interval ----------------------------------------------------
PosteriorIndirectCentral(
  phi = phi,
```

```
  delta_t = 1
)

# Range of time intervals ---------------------------------------------------
posterior <- PosteriorIndirectCentral(
  phi = phi,
  delta_t = 1:5
)

# Methods -------------------------------------------------------------------
# PosteriorIndirectCentral has a number of methods including
# print, summary, confint, and plot
print(posterior)
summary(posterior)
confint(posterior, level = 0.95)
plot(posterior)
```

PosteriorMed — *Posterior Distribution of Total, Direct, and Indirect Effects of X on Y Through M Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a posterior distribution of the total, direct and indirect effects of the independent variable X on the dependent variable Y through mediator variables $\mathbf{m}$ over a specific time interval Δt or a range of time intervals using the posterior distribution of the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$.

Usage

```
PosteriorMed(phi, delta_t, from, to, med, ncores = NULL, tol = 0.01)
```

Arguments

phi	List of numeric matrices. Each element of the list is a sample from the posterior distribution of the drift matrix ($\boldsymbol{\Phi}$). Each matrix should have row and column names pertaining to the variables in the system.
delta_t	Numeric. Time interval (Δt).
from	Character string. Name of the independent variable X in `phi`.
to	Character string. Name of the dependent variable Y in `phi`.
med	Character vector. Name/s of the mediator variable/s in `phi`.
ncores	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of replications `R` is a large value.
tol	Numeric. Smallest possible time interval to allow.

Details

See Total(), Direct(), and Indirect() for more details.

Value

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

call Function call.

args Function arguments.

fun Function used ("PosteriorMed").

output A list the length of which is equal to the length of delta_t.

Each element in the output list has the following elements:

est Mean of the posterior distribution of the total, direct, and indirect effects.

thetahatstar Posterior distribution of the total, direct, and indirect effects.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), Direct(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorTotalCentral(), Total(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
```

```
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

phi <- MCPhi(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  R = 1000L
)$output

# Specific time interval ----------------------------------------------------
PosteriorMed(
  phi = phi,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m"
)

# Range of time intervals ---------------------------------------------------
posterior <- PosteriorMed(
  phi = phi,
```

```
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)

# Methods -----------------------------------------------------------------
# PosteriorMed has a number of methods including
# print, summary, confint, and plot
print(posterior)
summary(posterior)
confint(posterior, level = 0.95)
plot(posterior)
```

`PosteriorTotalCentral` *Posterior Distribution of the Total Effect Centrality Over a Specific Time Interval or a Range of Time Intervals*

Description

This function generates a posterior distribution of the total effect centrality over a specific time interval Δt or a range of time intervals using the posterior distribution of the first-order stochastic differential equation model drift matrix $\boldsymbol{\Phi}$.

Usage

```
PosteriorTotalCentral(phi, delta_t, ncores = NULL, tol = 0.01)
```

Arguments

`phi`	List of numeric matrices. Each element of the list is a sample from the posterior distribution of the drift matrix ($\boldsymbol{\Phi}$). Each matrix should have row and column names pertaining to the variables in the system.
`delta_t`	Numeric. Time interval (Δt).
`ncores`	Positive integer. Number of cores to use. If `ncores = NULL`, use a single core. Consider using multiple cores when number of replications `R` is a large value.
`tol`	Numeric. Smallest possible time interval to allow.

Details

See `TotalCentral()` for more details.

Value

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

call Function call.

args Function arguments.

fun Function used ("PosteriorTotalCentral").

output A list the length of which is equal to the length of `delta_t`.

Each element in the `output` list has the following elements:

est Mean of the posterior distribution of the total, direct, and indirect effects.

thetahatstar Posterior distribution of the total, direct, and indirect effects.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `Total()`, `TotalCentral()`, `TotalStd()`, `Trajectory()`

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
```

```
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

phi <- MCPhi(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  R = 1000L
)$output

# Specific time interval ----------------------------------------------------
PosteriorTotalCentral(
  phi = phi,
  delta_t = 1
)

# Range of time intervals ---------------------------------------------------
posterior <- PosteriorTotalCentral(
  phi = phi,
  delta_t = 1:5
)

# Methods -------------------------------------------------------------------
# PosteriorTotalCentral has a number of methods including
# print, summary, confint, and plot
print(posterior)
```

```
summary(posterior)
confint(posterior, level = 0.95)
plot(posterior)
```

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

Description

Print Method for Object of Class `ctmedboot`

Usage

```
## S3 method for class 'ctmedboot'
print(x, alpha = 0.05, digits = 4, type = "pc", ...)
```

Arguments

`x`	an object of class `ctmedboot`.
`alpha`	Numeric vector. Significance level α.
`digits`	Integer indicating the number of decimal places to display.
`type`	Charater string. Confidence interval type, that is, `type = "pc"` for percentile; `type = "bc"` for bias corrected.
`...`	further arguments.

Value

Prints a list of matrices of time intervals, estimates, standard errors, number of bootstrap replications, and confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
## Not run:
library(bootStateSpace)
# prepare parameters
## number of individuals
n <- 50
## time points
time <- 100
delta_t <- 0.10
## dynamic structure
p <- 3
mu0 <- rep(x = 0, times = p)
```

```
sigma0 <- matrix(
  data = c(
    1.0,
    0.2,
    0.2,
    0.2,
    1.0,
    0.2,
    0.2,
    0.2,
    1.0
  ),
  nrow = p
)
sigma0_l <- t(chol(sigma0))
mu <- rep(x = 0, times = p)
phi <- matrix(
  data = c(
    -0.357,
    0.771,
    -0.450,
    0.0,
    -0.511,
    0.729,
    0,
    0,
    -0.693
  ),
  nrow = p
)
sigma <- matrix(
  data = c(
    0.24455556,
    0.02201587,
    -0.05004762,
    0.02201587,
    0.07067800,
    0.01539456,
    -0.05004762,
    0.01539456,
    0.07553061
  ),
  nrow = p
)
sigma_l <- t(chol(sigma))
## measurement model
k <- 3
nu <- rep(x = 0, times = k)
lambda <- diag(k)
theta <- 0.2 * diag(k)
theta_l <- t(chol(theta))

boot <- PBSSMOUFixed(
```

```
  R = 1000L,
  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,
  ncores = parallel::detectCores() - 1,
  seed = 42
)
phi_hat <- phi
colnames(phi_hat) <- rownames(phi_hat) <- c("x", "m", "y")
phi <- extract(object = boot, what = "phi")

# Specific time interval ----------------------------------------------------
boot <- BootMed(
  phi = phi,
  phi_hat = phi_hat,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m"
)
print(boot)
print(boot, type = "bc") # bias-corrected

# Range of time intervals ---------------------------------------------------
boot <- BootMed(
  phi = phi,
  phi_hat = phi_hat,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)
print(boot)
print(boot, type = "bc") # bias-corrected

## End(Not run)
```

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

Description

Print Method for Object of Class `ctmeddelta`

Usage

```
## S3 method for class 'ctmeddelta'
print(x, alpha = 0.05, digits = 4, ...)
```

Arguments

`x`	an object of class `ctmeddelta`.
`alpha`	Numeric vector. Significance level α.
`digits`	Integer indicating the number of decimal places to display.
`...`	further arguments.

Value

Prints a list of matrices of time intervals, estimates, standard errors, test statistics, p-values, and confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
```

```
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

# Specific time interval ----------------------------------------------------
delta <- DeltaMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m"
)
print(delta)

# Range of time intervals ---------------------------------------------------
delta <- DeltaMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)
print(delta)
```

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

Description

Print Method for Object of Class `ctmedeffect`

Usage

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

Arguments

x	an object of class ctmedeffect.
digits	Integer indicating the number of decimal places to display.
...	further arguments.

Value

Prints the effects.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
delta_t <- 1

# Time Interval of One -----------------------------------------------------

## Total Effect ------------------------------------------------------------
total_dt <- Total(
  phi = phi,
  delta_t = delta_t
)
print(total_dt)

## Direct Effect -----------------------------------------------------------
direct_dt <- Direct(
  phi = phi,
  delta_t = delta_t,
  from = "x",
  to = "y",
  med = "m"
)
print(direct_dt)

## Indirect Effect ---------------------------------------------------------
```

```
indirect_dt <- Indirect(
  phi = phi,
  delta_t = delta_t,
  from = "x",
  to = "y",
  med = "m"
)
print(indirect_dt)
```

---

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

Description

Print Method for Object of Class `ctmedmc`

Usage

```
## S3 method for class 'ctmedmc'
print(x, alpha = 0.05, digits = 4, ...)
```

Arguments

`x`	an object of class `ctmedmc`.
`alpha`	Numeric vector. Significance level α.
`digits`	Integer indicating the number of decimal places to display.
`...`	further arguments.

Value

Prints a list of matrices of time intervals, estimates, standard errors, number of Monte Carlo replications, and confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
set.seed(42)
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
```

```
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

# Specific time interval ----------------------------------------------------
mc <- MCMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m",
  R = 100L # use a large value for R in actual research
)
print(mc)

# Range of time intervals ---------------------------------------------------
mc <- MCMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5,
  from = "x",
```

```
  to = "y",
  med = "m",
  R = 100L # use a large value for R in actual research
)
print(mc)
```

---

print.ctmedmcphi *Print Method for Object of Class* ctmedmcphi

---

Description

Print Method for Object of Class ctmedmcphi

Usage

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

Arguments

x	an object of class ctmedmcphi.
digits	Integer indicating the number of decimal places to display.
...	further arguments.

Value

Prints a list of drift matrices.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
set.seed(42)
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
mc <- MCPhi(
  phi = phi,
  vcov_phi_vec = 0.1 * diag(9),
  R = 100L # use a large value for R in actual research
```

```
)
print(mc)
```

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

Description

Print Method for Object of Class `ctmedmed`

Usage

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

Arguments

`x`	an object of class `ctmedmed`.
`digits`	Integer indicating the number of decimal places to display.
`...`	further arguments.

Value

Prints a matrix of effects.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")

# Specific time interval ----------------------------------------------------
med <- Med(
  phi = phi,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m"
```

```
)
print(med)

# Range of time intervals ----------------------------------------------------
med <- Med(
  phi = phi,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)
print(med)
```

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

Description

Print Method for Object of Class `ctmedtraj`

Usage

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

Arguments

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

Value

Prints a data frame of simulated data.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
```

```
colnames(phi) <- rownames(phi) <- c("x", "m", "y")

traj <- Trajectory(
  mu0 = c(3, 3, -3),
  time = 150,
  phi = phi,
  med = "m"
)

print(traj)
```

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

Description

Summary Method for an Object of Class `ctmedboot`

Usage

```
## S3 method for class 'ctmedboot'
summary(object, alpha = 0.05, type = "pc", ...)
```

Arguments

`object`	Object of class `ctmedboot`.
`alpha`	Numeric vector. Significance level α.
`type`	Charater string. Confidence interval type, that is, `type = "pc"` for percentile; `type = "bc"` for bias corrected.
`...`	additional arguments.

Value

Returns a data frame of effects, time intervals, estimates, standard errors, number of bootstrap replications, and confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
## Not run:
library(bootStateSpace)
# prepare parameters
## number of individuals
n <- 50
## time points
time <- 100
delta_t <- 0.10
## dynamic structure
p <- 3
mu0 <- rep(x = 0, times = p)
sigma0 <- matrix(
  data = c(
    1.0,
    0.2,
    0.2,
    0.2,
    1.0,
    0.2,
    0.2,
    0.2,
    1.0
  ),
  nrow = p
)
sigma0_l <- t(chol(sigma0))
mu <- rep(x = 0, times = p)
phi <- matrix(
  data = c(
    -0.357,
    0.771,
    -0.450,
    0.0,
    -0.511,
    0.729,
    0,
    0,
    -0.693
  ),
  nrow = p
)
sigma <- matrix(
  data = c(
    0.24455556,
    0.02201587,
    -0.05004762,
    0.02201587,
    0.07067800,
    0.01539456,
    -0.05004762,
    0.01539456,
```

```
    0.07553061
  ),
  nrow = p
)
sigma_l <- t(chol(sigma))
## measurement model
k <- 3
nu <- rep(x = 0, times = k)
lambda <- diag(k)
theta <- 0.2 * diag(k)
theta_l <- t(chol(theta))

boot <- PBSSMOUFixed(
  R = 1000L,
  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,
  ncores = parallel::detectCores() - 1,
  seed = 42
)
phi_hat <- phi
colnames(phi_hat) <- rownames(phi_hat) <- c("x", "m", "y")
phi <- extract(object = boot, what = "phi")

# Specific time interval ----------------------------------------------------
boot <- BootMed(
  phi = phi,
  phi_hat = phi_hat,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m"
)
summary(boot)
summary(boot, type = "bc") # bias-corrected

# Range of time intervals ---------------------------------------------------
boot <- BootMed(
  phi = phi,
  phi_hat = phi_hat,
  delta_t = 1:5,
  from = "x",
  to = "y",
```

```
  med = "m"
)
summary(boot)
summary(boot, type = "bc") # bias-corrected

## End(Not run)
```

---

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

Description

Summary Method for an Object of Class `ctmeddelta`

Usage

```
## S3 method for class 'ctmeddelta'
summary(object, alpha = 0.05, ...)
```

Arguments

`object`	Object of class `ctmeddelta`.
`alpha`	Numeric vector. Significance level α.
`...`	additional arguments.

Value

Returns a data frame of effects, time intervals, estimates, standard errors, test statistics, p-values, and confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
```

```
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

# Specific time interval ----------------------------------------------------
delta <- DeltaMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m"
)
summary(delta)

# Range of time intervals ---------------------------------------------------
delta <- DeltaMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
)
summary(delta)
```

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

Description

Summary Method for an Object of Class `ctmedmc`

Usage

```
## S3 method for class 'ctmedmc'
summary(object, alpha = 0.05, ...)
```

Arguments

`object`	Object of class `ctmedmc`.
`alpha`	Numeric vector. Significance level α.
`...`	additional arguments.

Value

Returns a data frame of effects, time intervals, estimates, standard errors, number of Monte Carlo replications, and confidence intervals.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
set.seed(42)
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
vcov_phi_vec <- matrix(
  data = c(
    0.00843, 0.00040, -0.00151,
    -0.00600, -0.00033, 0.00110,
    0.00324, 0.00020, -0.00061,
    0.00040, 0.00374, 0.00016,
    -0.00022, -0.00273, -0.00016,
    0.00009, 0.00150, 0.00012,
    -0.00151, 0.00016, 0.00389,
    0.00103, -0.00007, -0.00283,
```

```
    -0.00050, 0.00000, 0.00156,
    -0.00600, -0.00022, 0.00103,
    0.00644, 0.00031, -0.00119,
    -0.00374, -0.00021, 0.00070,
    -0.00033, -0.00273, -0.00007,
    0.00031, 0.00287, 0.00013,
    -0.00014, -0.00170, -0.00012,
    0.00110, -0.00016, -0.00283,
    -0.00119, 0.00013, 0.00297,
    0.00063, -0.00004, -0.00177,
    0.00324, 0.00009, -0.00050,
    -0.00374, -0.00014, 0.00063,
    0.00495, 0.00024, -0.00093,
    0.00020, 0.00150, 0.00000,
    -0.00021, -0.00170, -0.00004,
    0.00024, 0.00214, 0.00012,
    -0.00061, 0.00012, 0.00156,
    0.00070, -0.00012, -0.00177,
    -0.00093, 0.00012, 0.00223
  ),
  nrow = 9
)

# Specific time interval ----------------------------------------------------
mc <- MCMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m",
  R = 100L # use a large value for R in actual research
)
summary(mc)

# Range of time intervals ---------------------------------------------------
mc <- MCMed(
  phi = phi,
  vcov_phi_vec = vcov_phi_vec,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m",
  R = 100L # use a large value for R in actual research
)
summary(mc)
```

---

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

Description

Summary Method for an Object of Class `ctmedmed`

Usage

```
## S3 method for class 'ctmedmed'
summary(object, digits = 4, ...)
```

Arguments

`object`	an object of class `ctmedmed`.
`digits`	Integer indicating the number of decimal places to display.
`...`	further arguments.

Value

Returns a matrix of effects.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")

# Specific time interval ----------------------------------------------------
med <- Med(
  phi = phi,
  delta_t = 1,
  from = "x",
  to = "y",
  med = "m"
)
summary(med)

# Range of time intervals ---------------------------------------------------
med <- Med(
  phi = phi,
  delta_t = 1:5,
  from = "x",
  to = "y",
  med = "m"
```

```
)
summary(med)
```

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

Description

Summary Method for Object of Class `ctmedposteriorphi`

Usage

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

Arguments

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

Value

Returns a list of the posterior means (in matrix form) and covariance matrix.

Author(s)

Ivan Jacob Agaloos Pesigan

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

Description

Summary Method for an Object of Class `ctmedtraj`

Usage

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

Arguments

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

Value

Returns a data frame of simulated data.

Author(s)

Ivan Jacob Agaloos Pesigan

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")

traj <- Trajectory(
  mu0 = c(3, 3, -3),
  time = 150,
  phi = phi,
  med = "m"
)

summary(traj)
```

`Total` *Total Effect Matrix Over a Specific Time Interval*

Description

This function computes the total effects matrix over a specific time interval Δt using the first-order stochastic differential equation model's drift matrix $\mathbf{\Phi}$.

Usage

```
Total(phi, delta_t)
```

Arguments

`phi`	Numeric matrix. The drift matrix ($\mathbf{\Phi}$). `phi` should have row and column names pertaining to the variables in the system.
`delta_t`	Numeric. Time interval (Δt).

Details

The total effect matrix over a specific time interval Δt is given by

$$\mathrm{Total}_{\Delta t} = \exp\left(\Delta t \boldsymbol{\Phi}\right)$$

where $\boldsymbol{\Phi}$ denotes the drift matrix, and Δt the time interval.

Linear Stochastic Differential Equation Model:
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.

Value

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

call Function call.

args Function arguments.

fun Function used ("Total").

output The matrix of total effects.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), Direct(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), TotalCentral(), TotalStd(), Trajectory()

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
delta_t <- 1
Total(
  phi = phi,
  delta_t = delta_t
)
phi <- matrix(
  data = c(
    -6, 5.5, 0, 0,
    1.25, -2.5, 5.9, -7.3,
    0, 0, -6, 2.5,
    5, 0, 0, -6
  ),
  nrow = 4
)
colnames(phi) <- rownames(phi) <- paste0("y", 1:4)
Total(
  phi = phi,
  delta_t = delta_t
)
```

TotalCentral *Total Effect Centrality*

Description

Total Effect Centrality

Usage

```
TotalCentral(phi, delta_t, tol = 0.01)
```

Arguments

phi — Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). phi should have row and column names pertaining to the variables in the system.

delta_t — Vector of positive numbers. Time interval (Δt).

tol — Numeric. Smallest possible time interval to allow.

Details

The total effect centrality of a variable is the sum of the total effects of a variable on all other variables at a particular time interval.

Value

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

call Function call.

args Function arguments.

fun Function used ("TotalCentral").

output A matrix of total effect centrality.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: BootBeta(), BootBetaStd(), BootIndirectCentral(), BootMed(), BootMedStd(), BootTotalCentral(), DeltaBeta(), DeltaBetaStd(), DeltaIndirectCentral(), DeltaMed(), DeltaMedStd(), DeltaTotalCentral(), Direct(), DirectStd(), Indirect(), IndirectCentral(), IndirectStd(), MCBeta(), MCBetaStd(), MCIndirectCentral(), MCMed(), MCMedStd(), MCPhi(), MCPhiSigma(), MCTotalCentral(), Med(), MedStd(), PosteriorBeta(), PosteriorIndirectCentral(), PosteriorMed(), PosteriorTotalCentral(), Total(), TotalStd(), Trajectory()

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")

# Specific time interval ----------------------------------------------------
TotalCentral(
  phi = phi,
  delta_t = 1
)

# Range of time intervals ---------------------------------------------------
total_central <- TotalCentral(
  phi = phi,
  delta_t = 1:30
)
plot(total_central)

# Methods -------------------------------------------------------------------
# TotalCentral has a number of methods including
# print, summary, and plot
total_central <- TotalCentral(
  phi = phi,
  delta_t = 1:5
)
print(total_central)
summary(total_central)
plot(total_central)
```

TotalStd *Standardized Total Effect Matrix Over a Specific Time Interval*

Description

This function computes the standardized total effects matrix over a specific time interval Δt using the first-order stochastic differential equation model's drift matrix $\boldsymbol{\Phi}$ and process noise covariance matrix $\boldsymbol{\Sigma}$.

Usage

```
TotalStd(phi, sigma, delta_t)
```

Arguments

`phi`	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). `phi` should have row and column names pertaining to the variables in the system.
`sigma`	Numeric matrix. The process noise covariance matrix ($\boldsymbol{\Sigma}$).
`delta_t`	Numeric. Time interval (Δt).

Details

The standardized total effect matrix over a specific time interval Δt is given by

$$\mathrm{Total}^{*}_{\Delta t_{i,j}} = \mathrm{Total}_{\Delta t_{i,j}} \left(\frac{\sigma_{x_j}}{\sigma_{y_i}} \right)$$

where $\boldsymbol{\Phi}$ denotes the drift matrix, σ_{x_j} and σ_{y_i} are the steady-state model-implied standard deviations of the state independent and dependent variables, respectively, and Δt the time interval.

Value

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

call Function call.

args Function arguments.

fun Function used ("TotalStd").

output The standardized matrix of total effects.

Author(s)

Ivan Jacob Agaloos Pesigan

References

Bollen, K. A. (1987). Total, direct, and indirect effects in structural equation models. Sociological Methodology, 17, 37. doi:10.2307/271028

Deboeck, P. R., & Preacher, K. J. (2015). No need to be discrete: A method for continuous time mediation analysis. Structural Equation Modeling: A Multidisciplinary Journal, 23 (1), 61–75. doi:10.1080/10705511.2014.973960

Ryan, O., & Hamaker, E. L. (2021). Time to intervene: A continuous-time approach to network analysis and centrality. Psychometrika, 87 (1), 214–252. doi:10.1007/s11336021097670

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `Trajectory()`

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")
sigma <- matrix(
  data = c(
    0.24455556, 0.02201587, -0.05004762,
    0.02201587, 0.07067800, 0.01539456,
    -0.05004762, 0.01539456, 0.07553061
  ),
  nrow = 3
)
delta_t <- 1
TotalStd(
  phi = phi,
  sigma = sigma,
  delta_t = delta_t
)
```

`Trajectory` *Simulate Trajectories of Variables*

Description

This function simulates trajectories of variables without measurement error or process noise. `Total` corresponds to the total effect and `Direct` corresponds to the portion of the total effect where the indirect effect is removed.

Usage

```
Trajectory(mu0, time, phi, med)
```

Arguments

`mu0`	Numeric vector. Initial values of the variables.
`time`	Positive integer. Number of time points.
`phi`	Numeric matrix. The drift matrix ($\boldsymbol{\Phi}$). `phi` should have row and column names pertaining to the variables in the system.
`med`	Character vector. Name/s of the mediator variable/s in `phi`.

Value

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

call Function call.

args Function arguments.

fun Function used ("Trajectory").

output A data frame of simulated data.

See Also

Other Continuous Time Mediation Functions: `BootBeta()`, `BootBetaStd()`, `BootIndirectCentral()`, `BootMed()`, `BootMedStd()`, `BootTotalCentral()`, `DeltaBeta()`, `DeltaBetaStd()`, `DeltaIndirectCentral()`, `DeltaMed()`, `DeltaMedStd()`, `DeltaTotalCentral()`, `Direct()`, `DirectStd()`, `Indirect()`, `IndirectCentral()`, `IndirectStd()`, `MCBeta()`, `MCBetaStd()`, `MCIndirectCentral()`, `MCMed()`, `MCMedStd()`, `MCPhi()`, `MCPhiSigma()`, `MCTotalCentral()`, `Med()`, `MedStd()`, `PosteriorBeta()`, `PosteriorIndirectCentral()`, `PosteriorMed()`, `PosteriorTotalCentral()`, `Total()`, `TotalCentral()`, `TotalStd()`

Examples

```
phi <- matrix(
  data = c(
    -0.357, 0.771, -0.450,
    0.0, -0.511, 0.729,
    0, 0, -0.693
  ),
  nrow = 3
)
colnames(phi) <- rownames(phi) <- c("x", "m", "y")

traj <- Trajectory(
  mu0 = c(3, 3, -3),
  time = 150,
  phi = phi,
  med = "m"
)
plot(traj)

# Methods -----------------------------------------------------------------
# Trajectory has a number of methods including
# print, summary, and plot

traj <- Trajectory(
  mu0 = c(3, 3, -3),
  time = 25,
  phi = phi,
  med = "m"
)
print(traj)
summary(traj)
plot(traj)
```

Index