Package: BrainCon (via r-universe)

August 22, 2024

Type Package

Title Inference the Partial Correlations Based on Time Series Data

Version 0.3.0

Author Yunhaonan Yang [aut, cre], Peng Wu [aut], Xin Gai [aut], Yumou Qiu [aut], Xiaohua Zhou [aut]

Maintainer Yunhaonan Yang <haonan_yy@pku.edu.cn>

Description A statistical tool to inference the multi-level partial correlations based on multi-subject time series data, especially for brain functional connectivity. It combines both individual and population level inference by using the methods of Qiu and Zhou. (2021)<DOI:10.1080/01621459.2021.1917417> and Genovese and Wasserman. (2006)<DOI:10.1198/016214506000000339>. It realizes two reliable estimation methods of partial correlation coefficients, using scaled lasso and lasso. It can be used to estimate individual- or population-level partial correlations, identify nonzero ones, and find out unequal partial correlation coefficients between two populations.

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Imports glmnet, MASS

Depends R (>= 2.10)

NeedsCompilation no

Date/Publication 2023-05-22 06:50:02 UTC

Repository https://cemrclinicencoding.r-universe.dev

RemoteUrl https://github.com/cran/BrainCon

RemoteRef HEAD

RemoteSha 4e5832a19b3802c1b61770929807aaa1e365596b

Contents

`individual.est` *Estimate individual-level partial correlation coefficients*

Description

Estimate individual-level partial correlation coefficients in time series data with $1 - \alpha$ confidence intervals. Note that these are confidence intervals for single parameters, not simultaneous confidence intervals.

Usage

```
individual.est(
  X,
  lambda = NULL,
  type = c("slasso", "lasso"),
  alpha = 0.05,
  ci = TRUE
)
```

Arguments

`X`	time series data of an individual which is a $n * p$ numeric matrix, where n is the number of periods of time and p is the number of variables.
`lambda`	a penalty parameter of order $\sqrt{\log(p)/n}$. If `NULL`, $\sqrt{2 * 2.01/n * \log(p * (\log(p))^{1.5}/n^{0.5})}$ is used in scaled lasso, and $\sqrt{2 * \log(p)/n}$ is used in lasso. Increasing the penalty parameter may lead to larger residuals in the node-wise regression, causing larger absolute values of estimates of partial correlation coefficients, which may cause more false positives in subsequent tests.
`type`	a character string representing the method of estimation. `"slasso"` means scaled lasso, and `"lasso"` means lasso. Default value is `"slasso"`.

alpha	significance level, default value is 0.05.
ci	a logical indicating whether to compute $1-\alpha$ confidence interval, default value is TRUE.

Value

An indEst class object containing two or four components.

coef a $p * p$ partial correlation coefficients matrix.

ci.lower a $p * p$ numeric matrix containing the lower bound of $1-\alpha$ confidence interval, returned if ci is TRUE.

ci.upper a $p * p$ numeric matrix containing the upper bound of $1-\alpha$ confidence interval, returned if ci is TRUE.

asym.ex a matrix measuring the asymptotic expansion of estimates, which will be used for multiple tests.

type regression type in estimation.

References

Qiu Y. and Zhou X. (2021). Inference on multi-level partial correlations based on multi-subject time series data, *Journal of the American Statistical Association*, 00, 1-15.

Sun T. and Zhang C. (2012). Scaled Sparse Linear Regression, *Biometrika*, 99, 879–898.

Liu W. (2013). Gaussian Graphical Model Estimation With False Discovery Rate Control, *The Annals of Statistics*, 41, 2948–2978.

Ren Z., Sun T., Zhang C. and Zhou H. (2015). Asymptotic Normality and Optimalities in Estimation of Large Gaussian Graphical Models, *The Annals of Statistics*, 43, 991–1026.

See Also

population.est.

Examples

```
## Quick example for the individual-level estimates
data(indsim)
# estimating partial correlation coefficients by scaled lasso
pc = individual.est(indsim)
```

`individual.test` *Identify nonzero individual-level partial correlations*

Description

Identify nonzero individual-level partial correlations in time series data by controlling the rate of the false discovery proportion (FDP) exceeding $c0$ at α, considering time dependence. Input an `indEst` class object returned by `individual.est` or `population.est`.

Usage

```
individual.test(
  indEst,
  alpha = 0.05,
  c0 = 0.1,
  targetSet = NULL,
  MBT = 3000,
  simplify = !is.null(targetSet)
)
```

Arguments

`indEst`	An `indEst` class object.
`alpha`	significance level, default value is `0.05`.
`c0`	threshold of the exceedance rate of FDP, default value is `0.1`. The choice of `c0` depends on the empirical problem. A smaller value of `c0` will reduce false positives, but it may also cost more false negatives.
`targetSet`	a two-column matrix. Each row contains two index corresponding to a pair of variables of interest. If `NULL`, any pair of two variables is considered to be of interest.
`MBT`	times of multiplier bootstrap, default value is `3000`.
`simplify`	a logical indicating whether results should be simplified if possible.

Value

If `simplify` is `FALSE`, a $p*p$ matrix with values 0 or 1 is returned. If the j-th row and k-th column of the matrix is 1, then the partial correlation coefficient between the j-th variable and the k-th variable is identified to be nonzero.

And if `simplify` is `TRUE`, a two-column matrix is returned, indicating the row index and the column index of recovered nonzero partial correlations. We only retain the results which the row index is less than the column index. Those with larger test statistics are sorted first.

References

Qiu Y. and Zhou X. (2021). Inference on multi-level partial correlations based on multi-subject time series data, *Journal of the American Statistical Association*, 00, 1-15.

See Also

`population.est` for making inferences on one individual in the population.

Examples

```
## Quick example for the individual-level inference
data(indsim)
# estimating partial correlation coefficients by scaled lasso
pc = individual.est(indsim)
# conducting hypothesis test
Res = individual.test(pc)
```

---

indsim *Simulation time series data for individual*

Description

A dataset containing values of 10 interested variables over 50 periods.

Usage

```
indsim
```

Format

An object of class `matrix` (inherits from `array`) with 50 rows and 10 columns.

Examples

```
## Generated by the following R codes
set.seed(1000)
n = 50; p = 10
Precision = diag(rep(2, p))        # generate precision matrix
for (i in 1 : (p - 1)){
  temp = ifelse(i > 2 * p / 3, 0.4, 1)
  Precision[i, i + 1] = temp
  Precision[i + 1, i] = temp
}
# R=-cov2cor(Precision) + diag(rep(2, p)) # real partial correlation matrix
Sigma = solve(Precision)           # generate covariance matrix
rho = 0.5
y = matrix(0, n, p)                # generate observed time series data
Epsilon = MASS::mvrnorm(n, rep(0, p), Sigma)
```

```
y[1, ] = Epsilon[1, ]
for (i in 2 : n){
  y[i, ] = rho * y[i - 1, ] + sqrt(1 - rho^2) * Epsilon[i, ]
}
indsim = y
```

---

popsimA *Simulation time series data for population A*

Description

A dataset containing values of 10 interested variables of 20 subjects over 50 periods.

Usage

```
popsimA
```

Format

An object of class `array` of dimension 50 x 10 x 20.

See Also

popsimB.

Examples

```
## Generated by the following R codes
set.seed(1234)
n = 50; p = 10; m1 = 20; m2 = 10
Precision1 = Precision2 = diag(rep(1, p))    # generate Precision matrix for population
for (i in 1 : (p - 1)){
  temp1 = ifelse(i > 2 * p / 3, -0.2, 0.4)
  temp2 = ifelse(i < p / 3, 0.4, -0.2)
  Precision1[i, i + 1] = Precision1[i + 1, i] = temp1
  Precision2[i, i + 1] = Precision2[i + 1, i] = temp2
}
# R1=-cov2cor(Precision1) + diag(rep(2, p))  # real partial correlation matrix
# R2=-cov2cor(Precision2) + diag(rep(2, p))
Index = matrix(0, p, p)                    # generate covariance matrix for each subject
for (i in 1 : p){
  for (j in 1 : p){
    if (i != j & abs(i - j) <= 3) Index[i, j] = 1
  }
}
SigmaAll1 = array(dim = c(p, p, m1))
SigmaAll2 = array(dim = c(p, p, m2))
for (sub in 1 : m1){
  RE = matrix(rnorm(p^2, 0, sqrt(2) * 0.05), p, p) * Index
  RE1 = (RE + t(RE)) / 2
```

```
  PrecisionInd = Precision1 + RE1
  SigmaAll1[, , sub] = solve(PrecisionInd)
}
for (sub in 1 : m2){
  RE = matrix(rnorm(p^2, 0, sqrt(2) * 0.15), p, p) * Index
  RE1 = (RE + t(RE)) / 2
  PrecisionInd = Precision2 + RE1
  SigmaAll2[, , sub] = solve(PrecisionInd)
}
rho = 0.3                                    # generate observed time series data
y1 = array(dim = c(n, p, m1))
y2 = array(dim = c(n, p, m2))
for (sub in 1 : m1){
  SigmaInd1 = SigmaAll1[, , sub]
  ytemp = matrix(0, n, p)
  Epsilon = MASS::mvrnorm(n, rep(0, p), SigmaInd1)
  ytemp[1, ] = Epsilon[1, ]
  for (i in 2 : n){
    ytemp[i, ] = rho * ytemp[i - 1, ] + sqrt(1 - rho^2) * Epsilon[i, ]
  }
  y1[, , sub] = ytemp
}
for (sub in 1 : m2){
  SigmaInd2 = SigmaAll2[, , sub]
  Xtemp = matrix(0, n, p)
  Epsilon = MASS::mvrnorm(n, rep(0, p), SigmaInd2)
  ytemp[1, ] = Epsilon[1, ]
  for (i in 2 : n){
    ytemp[i, ] = rho * ytemp[i - 1, ] + sqrt(1 - rho^2) * Epsilon[i, ]
  }
  y2[, , sub] = ytemp
}
popsimA = y1
popsimB = y2
```

---

popsimB *Simulation time series data for population B*

Description

A dataset containing values of 10 interested variables of 10 subjects over 50 periods.

Usage

```
popsimB
```

Format

An object of class `array` of dimension 50 x 10 x 10.

See Also

popsimA.

---

population.est *Estimate population-level partial correlation coefficients*

Description

Estimate population-level partial correlation coefficients in time series data. And also return coefficients for each individual. Input time series data for population as a 3-dimensional array or a list.

Usage

```
population.est(
  Z,
  lambda = NULL,
  type = c("slasso", "lasso"),
  alpha = 0.05,
  ind.ci = FALSE
)
```

Arguments

Z	If each individual shares the same number of periods of time, Z can be a $n*p*m$ dimensional array, where m is number of individuals. In general, Z should be a m-length list, and each element in the list is a $n_i * p$ matrix, where n_i stands for the number of periods of time of the i-th individual.
lambda	a scalar or a m-length vector, representing the penalty parameters of order $\sqrt{\log(p)/n_i}$ for each individual. If a scalar, the penalty parameters used in each individual are the same. If a m-length vector, the penalty parameters for each individual are specified in order. And if NULL, penalty parameters are specified by type. More details about the penalty parameters are in individual.est.
type	a character string representing the method of estimation. "slasso" means scaled lasso, and "lasso" means lasso. Default value is "slasso".
alpha	a numeric scalar, default value is 0.05. It is used when ind.ci is TRUE.
ind.ci	a logical indicating whether to compute $1 - \alpha$ confidence intervals of each subject, default value is FALSE.

Value

A popEst class object containing two components.

coef a $p * p$ partial correlation coefficients matrix.

ind.est a m-length list, containing estimates for each individuals.

type regression type in estimation.

References

Qiu Y. and Zhou X. (2021). Inference on multi-level partial correlations based on multi-subject time series data, *Journal of the American Statistical Association*, 00, 1-15.

Examples

```
## Quick example for the population-level estimates
data(popsimA)
# estimating partial correlation coefficients by scaled lasso
pc = population.est(popsimA)

## Inference on the first subject in population
Res_1 = individual.test(pc$ind.est[[1]])
```

`population.test` *The one-sample population inference*

Description

Identify the nonzero partial correlations in one-sample population, based on controlling the rate of the false discovery proportion (FDP) exceeding $c0$ at α, considering time dependence. Input a `popEst` class object returned by `population.est`.

Usage

```
population.test(
  popEst,
  alpha = 0.05,
  c0 = 0.1,
  targetSet = NULL,
  MBT = 5000,
  simplify = !is.null(targetSet)
)
```

Arguments

`popEst`	A `popEst` class object.
`alpha`	significance level, default value is `0.05`.
`c0`	threshold of the exceedance rate of FDP, default value is `0.1`. A smaller value of `c0` will reduce false positives, but it may also cost more false negatives.
`targetSet`	a two-column matrix. Each row contains two index corresponding to a pair of variables of interest. If `NULL`, any pair of two variables is considered to be of interest.
`MBT`	times of multiplier bootstrap, default value is `5000`.
`simplify`	a logical indicating whether results should be simplified if possible.

Value

If `simplify` is `FALSE`, a $p * p$ matrix with values 0 or 1 is returned, and 1 means nonzero.

And if `simplify` is `TRUE`, a two-column matrix is returned, indicating the row index and the column index of recovered nonzero partial correlations. We only retain the results which the row index is less than the column index. Those with larger test statistics are sorted first.

References

Qiu Y. and Zhou X. (2021). Inference on multi-level partial correlations based on multi-subject time series data, *Journal of the American Statistical Association*, 00, 1-15.

See Also

`individual.test`.

Examples

```
## Quick example for the one-sample population inference
data(popsimA)
# estimating partial correlation coefficients by scaled lasso
pc = population.est(popsimA)
# conducting hypothesis test
Res = population.test(pc)
# conducting hypothesis test in variables of interest
set = cbind(rep(7:9, each = 10), 1:10)
Res_like = population.test(pc, targetSet = set)
```

---

population.test.MinPv *The one-sample population inference using Genovese and Wasserman's method*

Description

Identify the nonzero partial correlations in one-sample population, based on controlling the rate of the false discovery proportion (FDP) exceeding $c0$ at α. The method is based on the minimum of the p-values. Input a `popEst` class object returned by `population.est`.

Usage

```
population.test.MinPv(
  popEst,
  alpha = 0.05,
  c0 = 0.1,
  targetSet = NULL,
  simplify = !is.null(targetSet)
)
```

Arguments

popEst	A popEst class object.
alpha	significance level, default value is 0.05.
c0	threshold of the exceedance rate of FDP, default value is 0.1.
targetSet	a two-column matrix. Each row contains two index corresponding to a pair of variables of interest. If NULL, any pair of two variables is considered to be of interest.
simplify	a logical indicating whether results should be simplified if possible.

Value

If simplify is FALSE, a $p * p$ matrix with values 0 or 1 is returned, and 1 means nonzero.

And if simplify is TRUE, a two-column matrix is returned, indicating the row index and the column index of recovered nonzero partial correlations. Those with lower p values are sorted first.

References

Genovese C. and Wasserman L. (2006). Exceedance Control of the False Discovery Proportion, *Journal of the American Statistical Association*, 101, 1408-1417.

Qiu Y. and Zhou X. (2021). Inference on multi-level partial correlations based on multi-subject time series data, *Journal of the American Statistical Association*, 00, 1-15.

See Also

population.test.

Examples

```
## Quick example for the one-sample population inference
data(popsimA)
# estimating partial correlation coefficients
pc = population.est(popsimA)
# conducting hypothesis test
Res  = population.test.MinPv(pc)
```

---

population2sample.test

Identify differences of partial correlations between two populations

---

Description

Identify differences of partial correlations between two populations in two groups of time series data by controlling the rate of the false discovery proportion (FDP) exceeding $c0$ at α, considering time dependence. Input two popEst class objects returned by population.est (the number of individuals in two groups can be different).

Usage

```
population2sample.test(
  popEst1,
  popEst2,
  alpha = 0.05,
  c0 = 0.1,
  targetSet = NULL,
  MBT = 5000,
  simplify = !is.null(targetSet)
)
```

Arguments

popEst1	A popEst class object.
popEst2	A popEst class object.
alpha	significance level, default value is 0.05.
c0	threshold of the exceedance rate of FDP, default value is 0.1. A smaller value of c0 will reduce false positives, but it may also cost more false negatives.
targetSet	a two-column matrix. Each row contains two index corresponding to a pair of variables of interest. If NULL, any pair of two variables is considered to be of interest.
MBT	times of multiplier bootstrap, default value is 5000.
simplify	a logical indicating whether results should be simplified if possible.

Value

If simplify is FALSE, a $p * p$ matrix with values 0 or 1 is returned. If the j-th row and k-th column of the matrix is 1, then the partial correlation coefficients between the j-th variable and the k-th variable in two populations are identified to be unequal.

And if simplify is TRUE, a two-column matrix is returned, indicating the row index and the column index of recovered unequal partial correlations. We only retain the results which the row index is less than the column index. Those with larger test statistics are sorted first.

References

Qiu Y. and Zhou X. (2021). Inference on multi-level partial correlations based on multi-subject time series data, *Journal of the American Statistical Association*, 00, 1-15.

Examples

```
## Quick example for the two-sample case inference
data(popsimA)
data(popsimB)
# estimating partial correlation coefficients by lasso (scaled lasso does the same)
pc1 = population.est(popsimA, type = 'l')
pc2 = population.est(popsimB, type = 'l')
# conducting hypothesis test
```

```
Res = population2sample.test(pc1, pc2)
# conducting hypothesis test and returning simplified results
Res_s = population2sample.test(pc1, pc2, simplify = TRUE)
```

`population2sample.test.MinPv` *Identify differences of partial correlations between two populations using Genovese and Wasserman's method*

Description

Identify differences of partial correlations between two populations in two groups of time series data, based on controlling the rate of the false discovery proportion (FDP) exceeding $c0$ at α. The method is based on the minimum of the p-values. Input two `popEst` class objects returned by `population.est` (the number of individuals in two groups can be different).

Usage

```
population2sample.test.MinPv(
  popEst1,
  popEst2,
  alpha = 0.05,
  c0 = 0.1,
  targetSet = NULL,
  simplify = !is.null(targetSet)
)
```

Arguments

`popEst1`	A `popEst` class object.
`popEst2`	A `popEst` class object.
`alpha`	significance level, default value is `0.05`.
`c0`	threshold of the exceedance rate of FDP, default value is `0.1`.
`targetSet`	a two-column matrix. Each row contains two index corresponding to a pair of variables of interest. If `NULL`, any pair of two variables is considered to be of interest.
`simplify`	a logical indicating whether results should be simplified if possible.

Value

If `simplify` is `FALSE`, a $p * p$ matrix with values 0 or 1 is returned, and 1 means unequal.

And if `simplify` is `TRUE`, a two-column matrix is returned, indicating the row index and the column index of recovered unequal partial correlations. Those with lower p values are sorted first.

References

Genovese C., and Wasserman L. (2006). Exceedance Control of the False Discovery Proportion, *Journal of the American Statistical Association*, 101, 1408-1417

Qiu Y. and Zhou X. (2021). Inference on multi-level partial correlations based on multi-subject time series data, *Journal of the American Statistical Association*, 00, 1-15.

Examples

```
## Quick example for the two-sample case inference
data(popsimA)
data(popsimB)
# estimating partial correlation coefficients by lasso (scaled lasso does the same)
pc1 = population.est(popsimA, type = 'l')
pc2 = population.est(popsimB, type = 'l')
# conducting hypothesis test
Res = population2sample.test.MinPv(pc1, pc2)
```

Index