Package 'CDNL'

November 27, 2024

Type Package

Title Cooperative Differential Network Learning

Version 0.1.0

Author Dingzi Guo [aut, cre]

Maintainer Dingzi Guo <guodingzi@mail.sdu.edu.cn>

Description Differential network modeling, or network comparison, has become an essential tool for identifying differences in brain connectivity between healthy and affected individuals. We propose a Cooperative Differential Network Learning (CDNL) method for the spatial by temporal matrix-valued fMRI data from multiple research centers, aimed at achieving better joint estimation of multiple differential networks.

License GPL-3

Encoding UTF-8

Imports SGL, Matrix, foreach, doParallel, DensParcorr, mnormt, expm

Depends R (>= 3.5.0)

R topics documented:

CDNL *Cooperative Differential Network Learning*

Description

Using logistic regression in combination with Cooperative penalty, Hub penalty and Lasso penalty to jointly estimate differential networks across different sites.

Usage

```
CDNL(data1,
    data2,
    index,
    boot_strap = 4, threshold = 2, cores = 4,
    alpha_min = 0, alpha_max = 1,
    lam_min = 0.001, lam_max = 2,
    nalpha = 4, nlambda = 4, fold = 3)
```

Arguments

data1	Site 1 data. A list containing two elements. The first element is a list of length n11, representing the case group dataset, each of which is p-by-q spatial-temporal matrix data. The second element is a list of length n12, representing the control group dataset, each of which is p-by-q spatial-temporal matrix data. n11 denotes the number of samples in case group, n12 denotes the number of samples in control group. p denotes the spatial dimension, q denotes the temporal dimension.
data2	Site 2 data. A list containing two elements. The first element is a list of length n21, representing the case group dataset, each of which is p-by-q spatial-temporal matrix data. The second element is a list of length n22, representing the control group dataset, each of which is p-by-q spatial-temporal matrix data. n21 denotes the number of samples in case group, n22 denotes the number of samples in control group. p denotes the spatial dimension, q denotes the temporal dimension.
index	A 3*p*(p-1)-vector indicating group membership of each network edge.
boot_strap	The number of training iterations using bootstrapped samples (default: 4).
threshold	Threshold for selecting differential edges from the differential edge weights matrix (default: 2).
cores	Number of cores used in parallel (default: 4).
alpha_min	Minimum value of the tuning parameter alpha for L2 penalty in sparse group lasso (default: 0).
alpha_max	Maximum value of the tuning parameter alpha for L2 penalty in sparse group lasso (default: 1).
lam_min	Minimum value of the tuning parameter lambda for L1 penalty in sparse group lasso (default: 0.001).
lam_max	Maximum value of the tuning parameter lambda for L1 penalty in sparse group lasso (default: 2).
nalpha	Number of alternative alpha when performing Cross-Validation (default: 4).
nlambda	Number of alternative lambda performing Cross-Validation (default: 4).
fold	Number of folds - default is 3. Although fold can be as large as the sample size (leave-one-out CV), it is not recommended for large dataset.

Value

An R list containing the following terms:

weight.matrix.1
: Estimated differential edge weights matrix in Site 1.

`weight.matrix.2`
: Estimated differential edge weights matrix in Site 2.

`diff.matrix.1` Estimated differential network in Site 1.

`diff.matrix.2` Estimated differential network in Site 2.

`error.rate1` Classification error rate for testing samples in Site 1.

`error.rate2` Classification error rate for testing samples in Site 2.

Examples

```
set.seed(0918)
p <- 20
q <- 15
N11 = 10
N12 = 10
N21 = 10
N22 = 10
m=2

index_mat <- matrix(NA, (m*p), (p-1))
for (i in 1:(m*p)) {
  index_mat[i,] <- rep(i,(p-1))
}
index_1 <- as.vector(t(index_mat)[,1:p])
index_2 <- as.vector(t(index_mat)[,(p+1):(m*p)])
index_inte <- c((m*p+1):(m*p+p*(p-1)/2))
index <- c(index_1, index_inte, index_2, index_inte)

generate_list_of_matrices <- function(N, p, q) {
  lapply(1:N, function(i) matrix(rnorm(p * q), nrow = p, ncol = q))
}

X11_w <- generate_list_of_matrices(N11, p, q)
X12_w <- generate_list_of_matrices(N12, p, q)
X21_w <- generate_list_of_matrices(N21, p, q)
X22_w <- generate_list_of_matrices(N22, p, q)

data1 = data2 = list()
data1[[1]]=X11_w
data1[[2]]=X12_w
data2[[1]]=X21_w
data2[[2]]=X22_w

result = CDNL(data1,data2,index,boot_strap=4,threshold=2,cores=4,
              alpha_min=0,alpha_max=1,lam_min=0.001,lam_max=2,nalpha=4,nlambda=4,fold=3)
result
```

`DensPcorr` *Dens-based approach for precision matrix estimation.*

Description

Dens-based approach for precision matrix estimation.

Usage

```
DensPcorr(
  data,
  select = FALSE,
  dens.level = 0.5,
  plateau.thresh = 0.01,
  Parcorr.est = NULL,
  lambda = NULL
)
```

Arguments

data	Input data matrix of size n (observations) times p (variables).
select	Whether to conduct the Dens-based selection. If FALSE, output will only contain the estimated partial correlation list and precision matrix list corresponding to the default tuning parameter series ranging from 1e-8 to 0.6. If TRUE, the ouput will include the previous results and the selected partial correlation matrix and percision matrix corresponding to the specified density level. Default is FALSE.
dens.level	Specify the density level in Dens-based tuning parameter selection method (0<dens.level<1). This option is valid only when select=TRUE.
plateau.thresh	The criterion to select the plateau.
Parcorr.est	Previous output from DensPcorr function.
lambda	The tuning parameters for estimating the precision matrix ranging from 0 to 1.

Details

This function implements the statistical method proposed in Wang et al. (2016). See Rpackage "DensParcorr".

Value

An R list containing the following terms:

`selected.precision`
: Selected Precision matrix corresponding to dens.level.

`selected.lambda`
: Selected tuning parameter corresponding to dens.level.

---

predictSGL *Outputs predicted response values for new input observations*

Description

Outputs predicted response values for new input observations.

Usage

```
predictSGL(x, newX)
```

Arguments

x	fitted "SGL" object
newX	Covariate matrix for new observations whose responses we wish to predict

Details

This function implements the statistical method proposed in Simon et al. (2011). See Rpackage "SGL".

References

Simon, N., Friedman, J., Hastie T., and Tibshirani, R. (2011) *A Sparse-Group Lasso*, http://faculty.washington.edu/nrsimon/SGLpaper.pdf

Index