R code for "Fast and Separable Estimation in High-dimensional Tensor Gaussian Graphical Models"

Keqian Min, Qing Mai and Xin Zhang   Florida State University

Main Functions

The R code files `Separate.fit.R` and `cv.Separate.R` implement the proposed method in "Fast and Separable Estimation in High-dimensional Tensor Gaussian Graphical Models".

- The `Separate.fit()` function estimates the precision matrices in the tensor graphical model using the proposed parallel scheme.
  - Arguments
    - `x`: Data set of size $p_1 \times \cdots \times p_M \times n$.
    - `val`: (Optional) Validation set. If supplied, `lambda.list` should be provided.
    - `est.mode`: Index set of precision matrices to be estimated. If not specified, all precision matrices will be estimated. Default is c$(1, \ldots, M)$.
    - `lambda.vec`: The sequence of regularization parameters for each mode in `est.mode`.
    - `lambda.list`: A list of regularization parameters that provides a lambda sequence for each mode in `est.mode`. If a validation set is supplied, the optimal tuning parameters will be chosen from `lambda.list` based on the log-likelihood calculated using validation set.
    - `Omegatilde.list`: (Optional) A list of M matrices. User-specified value for $\widetilde{\mathbf{\Omega}}_m$.
    - `scale.vec`: A sequence of constants to scale each $\widehat{\mathbf{\Omega}}_m$ to avoid infinite value for their log-determinants when calculating the log-likelihood. Default is 1 for all modes.
    - `normalize`: Indicates whether $\widetilde{\mathbf{\Omega}}_m$ and $\widehat{\mathbf{\Omega}}_m$ should be normalized to have unit Frobenius norm. Default is TRUE.
    - `thres`: Threshold for convergence. Default value is 1e-4.
    - `maxit`: Maximum number of iterations for fitting glasso. Default value is 10,000.
    - `njobs`: Number of nodes used in parallel computing.
  - Value
    - `Omegahat`: Output precision matrices for each mode in `est.mode`.
    - `lambda`: The actual values used for regularization parameters.
    - `loglik`: For an estimated precision matrix $\widehat{\mathbf{\Omega}}_m$, the program only calculates $\text{loglik}[m] = \log(\det(\widehat{\mathbf{\Omega}}_m * \text{scale.vec}[m])) - \text{trace}(\widetilde{\mathbf{S}}_m\widehat{\mathbf{\Omega}}_m)$ for $\widetilde{\mathbf{S}}_m$ calculated using the validation set. If validation set is not supplied, no output is returned.
- The `cv.Separate()` function estimates the precision matrices in the tensor graphical model by choosing tuning parameters using cross-validation.
  - Arguments
    - `x`: Data set of size $p_1 \times \cdots \times p_M \times n$.
    - `est.mode`: Index set of precision matrices to be estimated. If not specified, all precision matrices will be estimated. Default is c$(1, \ldots, M)$.
    - `lambda.list`: A list of regularization parameters that provides a lambda sequence for each mode in `est.mode` for cross validation.
    - `Omegatilde.list`: (Optional) A list of M matrices. User-specified value for $\widetilde{\mathbf{\Omega}}_m$.
    - `scale.vec`: A sequence of constants to scale each $\widehat{\mathbf{\Omega}}_m$ to avoid infinite value for their log-determinants when calculating the log-likelihood. Default is 1 for all modes.
    - `normalize`: Indicates whether $\widetilde{\mathbf{\Omega}}_m$ and $\widehat{\mathbf{\Omega}}_m$ should be normalized to have unit Frobenius norm. Default is TRUE.

        - `nfolds`: Number of folds. Default is 5.
        - `foldid`: (Optional) A vector of values between 1 and nfolds identifying what fold each observation is in. If supplied, `nfolds` can be missing.
        - `thres`: Threshold for convergence. Default value is 1e-4.
        - `maxit`: Maximum number of iterations for fitting glasso. Default value is 10,000.
        - `njobs`: Number of nodes used in parallel computing.
    - Value
        - `Omegahat`: Output precision matrices for each mode in `est.mode`.
        - `lambda`: The best regularization parameters with the largest cross-validated log-likelihoods.
        - `loglik`: The mean cross-validated log-likelihoods for each mode in `est.mode`.
        - `loglik.se`: Standard error of `loglik`.

Note that in `Separate.fit()` function, `val` and `lambda.list` should be provided together for tuning purposes. Otherwise, `lambda.vec` will be directly used to fit the model. We recommend to provide a validation set or use `cv.Separate()` function to tune parameters. The estimators obtained during the tuning process will not be returned since the results are redundant. Only the best estimations will be output.

Required packages:

```r
library(Tlasso)
library(tensr)
library(glasso)
library(expm)
library(rTensor)
library(doParallel)
```

Please make sure the packages are installed and then one can use

```r
source("Separate.fit.R")
source("cv.Separate.R")
```

Simulation Example

The R code files `example_Separate.R`, `example_Oracle.R`, `example_Sequential.R` and `example_Cyclic.R` reproduce the results for Model 7 in Table 1 in the supplementary material. In this example, we generate $n = 20$ i.i.d tensor observations, each with dimension $(30, 36, 30)$.

- `example_Separate.R` obtains the "Separate" estimators from our proposed method.
- `example_Oracle.R` obtains the "Oracle" estimators.
- `example_Sequential.R` obtains the "Sequential" estimators using the `Tlasso` package.
- `example_Cyclic.R` obtains the "Cyclic" estimators using the `Tlasso` package.
- `model.R` contains the model settings for Models 1-14.
- The `Model7()` function generates training set and validation set from Model 7.
- The `simulation.summary()` function computes the estimation errors, true positive rate and true negative rate for each mode and their averages across all modes after obtaining the estimators in a simulation study. It is similar to the `est.analysis()` function in the `Tlasso` package, but it has been tailored for our study.
    - Arguments
        - `Omega.hat.list`: List of estimation of precision matrices.
        - `Omega.true.list`: List of true precision matrices.
        - `offdiag`: Indicates if excludes diagonal when computing performance measures.
    - Value
        - `error.f`: estimation error in Frobenius norm of each mode.
        - `error.max`: estimation error in Maximum norm of each mode.
        - `tpr`: True positive rate of each mode.
        - `tnr`: True negative rate of each mode.
        - `av.error.f`: Averaged Frobenius norm error.

    * `av.error.max`: Averaged Maximum norm error.
    * `av.tpr`: Average true positive rate.
    * `av.tnr`: Average true negative rate.

In the following, we demonstrate how to reproduce one replicate with our proposed method for Model 7 in the simulation study. We start from loading some packages and source files.

```
rm(list = ls())
library(Tlasso)
library(tensr)
library(glasso)
library(expm)
library(rTensor)
library(doParallel)
source("Separate.fit.R")
source("cv.Separate.R")
source("simulation.summary.R")
source("Model7.R")
```

Model Setting

Next, we set up Model 7. We use function `ChainOmega()` from the `Tlasso` package to generate sparse precision matrices with unit Frobenius norm.

```
n <- 20 # sample size
dimen <- c(30, 36, 30) # dimension of tensor
nvars <- prod(dimen) # number of variables
K <- 3 # order of tensor

# set-up of precision matrices
Omega <- array(list(), length(dimen)) # a list of precision matrices
for (i in 1:length(dimen)) {
  Omega[[i]] <- ChainOmega(dimen[i], sd = i * 100, norm.type = 2)
}
```

Data Generation

We generate a training set and a validation set with the same sample size. We will use the validation set to choose tuning parameters in the next step.

```
# Generate training set and validation set
data <- Model7(n, seed = 123456)
x <- data$x # training set
vax <- data$vax # validation set
```

Parameter Tuning and Model Fitting

When a list of tuning parameters, `lambda.list`, is supplied, the function `Separate.fit()` will choose the best lambda for each mode that maximizes the log-likelihood calculated using the validation set and return the estimated precision matrices.

```
# proper candidates of tuning parameters
lamseq <- seq(0.015, 0.1, length.out = 10)
lambda.list <- list() # a list containing candidates of tuning parameters for each mode
for (i in 1:K) {
  lambda.list[[i]] <- lamseq
}
```

```r
# Model fitting
fit <- Separate.fit(x, vax, lambda.list = lambda.list)
```

Performance Evaluation

```r
# Simulation summary of estimation errors, TPR and TNR
out <- simulation.summary(fit$Omegahat, Omega, offdiag = FALSE)
out$av.error.f # averaged estimation error in Frobenius norm
out$av.error.max # averaged estimation error in Maximum norm
out$av.tpr # averaged true positive rate
out$av.tnr # averaged true negative rate
out$error.f # estimation error in Frobenius norm for each mode
out$error.max # estimation error in Maximum norm for each mode
out$tpr # true positive rate for each mode
out$tnr # true negative rate for each mode
```

Real Data

- `EEG_dataset.RData` contains the EEG dataset. It contains `X.mat` with size $256 \times 64 \times 122$ and the label `y.vec` with size 1×122. The label $y = 1$ indicates the individual is from the alcoholic group and $y = 0$ indicates the individual is from the nonalcoholic group.
- `EEG_Separate.R` implements the proposed method on the alcoholic and nonalcoholic groups in EEG dataset.
- `EEG_Cyclic.R` implements the cyclic method using `Tlasso` package on the alcoholic and nonalcoholic groups in EEG dataset.