Package 'IDLFM'

May 11, 2025

Title Individual Dynamic Latent Factor Model

Version 1.0.0

Maintainer Siyang Liu <liusiyang.lucia@gmail.com>

Description A personalized dynamic latent factor model (Zhang et al. (2024) <doi:10.1093/biomet/asae015>) for irregular multi-resolution time series data, to interpolate unsampled measurements from low-resolution time series.

Depends R (>= 4.4.0)

Imports stats, methods, splines, SparseArray

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Siyang Liu [aut, cre],
Jiuchen Zhang [aut],
Annie Qu [aut]

Repository CRAN

Date/Publication 2025-05-11 09:50:02 UTC

Contents

generate_data *Generate data for simulation*

Description

This function generates simulated data in multiple time series with heterogeneity and non-stationarity. It includes 3 settings in Setion 5.3.

Usage

```
generate_data(n_patients, n_var, time, idx_x, idx_y, rank, k, N)
```

Arguments

`n_patients`	the number of patients
`n_var`	the number of X variables
`time`	maximum time
`idx_x`	indices for the x data, a sparse matrix
`idx_y`	indices for the y data, a sparse matrix
`rank`	rank for the random matrices
`k`	spline smoothness
`N`	number of knots in the splineS

Value

A list is returned, containing output_x and output_y as sparse matrices of x_data and y_data, spline knots, individualized dynamic latent factor, shared latent factor for X and Y.

References

Zhang, J., F. Xue, Q. Xu, J. Lee, and A. Qu. "Individualized dynamic latent factor model for multi-resolutional data with application to mobile health." Biometrika (2024): asae015.

Examples

```
library(splines)
#if (!require("BiocManager", quietly = TRUE))
#install.packages("BiocManager")
#BiocManager::install("SparseArray")
library(SparseArray)

I <- 3
J <- 5
time <- 1000
R <- 3
k <- 3
```

```
N <- 300
idx_x <- randomSparseArray(c(I, J, time), density=0.8)
idx_y_train <- randomSparseArray(c(I, 1, time), density=0.2)
idx_y_test <- randomSparseArray(c(I, 1, time), density=0.2)
generate_data(I, J, time, idx_x, idx_y_train, R, k, N)
```

IDLFM *Individualized Dynamic Latent Factor Model*

Description

This function implements the individualized dynamic latent factor model.

Usage

```
IDLFM(
  X,
  Y,
  n_patients,
  n_var,
  time,
  idx_x,
  idx_y,
  rank,
  k,
  N,
  lambda1 = 1,
  lambda2 = 1,
  Niter = 100,
  alpha = 0.001,
  ebs = 1e-04,
  l = 1,
  verbose
)
```

Arguments

X	a sparse matrix for predictor variables
Y	a sparse matrix for response variables
n_patients	the number of patients
n_var	the number of X variables
time	maximum time
idx_x	indices for the X data, a sparse matrix
idx_y	indices for the Y data, a sparse matrix
rank	rank for the random matrices

k	spline smoothness
N	number of knots in the spline
lambda1	regularization parameter for fused lasso, with the default value 1
lambda2	regularization parameter for total variation, with the default value 1
Niter	number of iterations for the Adam optimizer, with the default value 100
alpha	learning rate for the Adam optimizer, with the default value 0.001
ebs	convergence threshold, with the default value 0.0001
l	regularization parameter, with the default value 1
verbose	to control the console output

Value

A list is returned, containing the model weights, factor matrix, spline knots, predicted X and Y.

References

Zhang, J., F. Xue, Q. Xu, J. Lee, and A. Qu. "Individualized dynamic latent factor model for multi-resolutional data with application to mobile health." Biometrika (2024): asae015.

Examples

```
library(splines)
#if (!require("BiocManager", quietly = TRUE))
#install.packages("BiocManager")
#BiocManager::install("SparseArray")
library(SparseArray)

I <- 3
J <- 5
time <- 1000
R <- 3
k <- 3
N <- 300
idx_x <- randomSparseArray(c(I, J, time), density=0.8)
idx_y_train <- randomSparseArray(c(I, 1, time), density=0.2)
idx_y_test <- randomSparseArray(c(I, 1, time), density=0.2)
data <- generate_data(I, J, time, idx_x, idx_y_train, R, k, N)
output_x <- data[[1]]
output_y <- data[[2]]
knots <- data[[3]]
weights <- data[[4]]
Fx <- data[[5]]
Fy <- data[[6]]
IDLFM(X = output_x, Y = output_y, n_patients = I, n_var = J, time = time,
idx_x = idx_x, idx_y = idx_y_train, rank = R, k = k, N = N, verbose = FALSE)
```

Index