Package 'JSparO'

January 20, 2025

Title Joint Sparse Optimization via Proximal Gradient Method for Cell
Fate Conversion

Version 1.5.0

Author Xinlin Hu [aut, cre],
Yaohua Hu [cph, aut]

Description Implementation of joint sparse optimization (JSparO) to infer the gene regulatory network for cell fate conversion. The proximal gradient method is implemented to solve different low-order regularization models for JSparO.

Maintainer Xinlin Hu <thompson-xinlin.hu@connect.polyu.hk>

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 4.0.0)

RoxygenNote 7.1.1

Imports pracma

NeedsCompilation no

Repository CRAN

Date/Publication 2022-08-18 08:40:08 UTC

Contents

---

demo_JSparO   *demo_JSparO - The demo of JSparO package*

Description

This is the main function of JSparO aimed to solve the low-order regularization models with $l_{p,q}$ norm.

Usage

```
demo_JSparO(A, B, X, s, p, q, maxIter = 200)
```

Arguments

A	Gene expression data of transcriptome factors (i.e. feature matrix in machine learning). The dimension of A is m * n.
B	Gene expression data of target genes (i.e. observation matrix in machine learning). The dimension of B is m * t.
X	Gene expression data of Chromatin immunoprecipitation or other matrix (i.e. initial iterative point in machine learning). The dimension of X is n * t.
s	joint sparsity level
p	value for $l_{p,q}$ norm (i.e., p = 1 or 2)
q	value for $l_{p,q}$ norm (i.e., 0 <= q <= 1)
maxIter	maximum iteration

Details

The demo_JSparO function is used to solve joint sparse optimization problem via different algorithms. Based on $l_{p,q}$ norm, functions with different p and q are implemented to solve the problem:

$$\min \|AX - B\|_F^2 + \lambda\|X\|_{p,q}$$

to obtain s-joint sparse solution.

Value

The solution of proximal gradient method with $l_{p,q}$ regularizer.

Author(s)

Xinlin Hu thompson-xinlin.hu@connect.polyu.hk

Yaohua Hu mayhhu@szu.edu.cn

Examples

```
m <- 256; n <- 1024; t <- 5; maxIter0 <- 50
A0 <- matrix(rnorm(m * n), nrow = m, ncol = n)
B0 <- matrix(rnorm(m * t), nrow = m, ncol = t)
X0 <- matrix(0, nrow = n, ncol = t)
res_JSparO <- demo_JSparO(A0, B0, X0, s = 10, p = 2, q = 'half', maxIter = maxIter0)
```

L1HalfThr	*L1HalfThr - Iterative Half Thresholding Algorithm based on* $l\_1, 1/2$ *norm*

Description

The function aims to solve $l_{1,1/2}$ regularized least squares.

Usage

```
L1HalfThr(A, B, X, s, maxIter = 200)
```

Arguments

A	Gene expression data of transcriptome factors (i.e. feature matrix in machine learning). The dimension of A is m * n.
B	Gene expression data of target genes (i.e. observation matrix in machine learning). The dimension of B is m * t.
X	Gene expression data of Chromatin immunoprecipitation or other matrix (i.e. initial iterative point in machine learning). The dimension of X is n * t.
s	joint sparsity level
maxIter	maximum iteration

Details

The L1HalfThr function aims to solve the problem:

$$\min \|AX - B\|_F^2 + \lambda \|X\|_{1,1/2}$$

to obtain s-joint sparse solution.

Value

The solution of proximal gradient method with $l_{1,1/2}$ regularizer.

Author(s)

Xinlin Hu thompson-xinlin.hu@connect.polyu.hk

Yaohua Hu mayhhu@szu.edu.cn

Examples

```
m <- 256; n <- 1024; t <- 5; maxIter0 <- 50
A0 <- matrix(rnorm(m * n), nrow = m, ncol = n)
B0 <- matrix(rnorm(m * t), nrow = m, ncol = t)
X0 <- matrix(0, nrow = n, ncol = t)
NoA <- norm(A0, '2'); A0 <- A0/NoA; B0 <- B0/NoA
res_L1half <- L1HalfThr(A0, B0, X0, s = 10, maxIter = maxIter0)
```

L1HardThr	*L1HardThr - Iterative Hard Thresholding Algorithm based on* $l\_1, 0$ *norm*

Description

The function aims to solve $l_{1,0}$ regularized least squares.

Usage

```
L1HardThr(A, B, X, s, maxIter = 200)
```

Arguments

A	Gene expression data of transcriptome factors (i.e. feature matrix in machine learning). The dimension of A is m * n.
B	Gene expression data of target genes (i.e. observation matrix in machine learning). The dimension of B is m * t.
X	Gene expression data of Chromatin immunoprecipitation or other matrix (i.e. initial iterative point in machine learning). The dimension of X is n * t.
s	joint sparsity level
maxIter	maximum iteration

Details

The L1HardThr function aims to solve the problem:

$$\min \|AX - B\|_F^2 + \lambda \|X\|_{1,0}$$

to obtain s-joint sparse solution.

Value

The solution of proximal gradient method with $l_{1,0}$ regularizer.

Author(s)

Xinlin Hu thompson-xinlin.hu@connect.polyu.hk

Yaohua Hu mayhhu@szu.edu.cn

Examples

```
m <- 256; n <- 1024; t <- 5; maxIter0 <- 50
A0 <- matrix(rnorm(m * n), nrow = m, ncol = n)
B0 <- matrix(rnorm(m * t), nrow = m, ncol = t)
X0 <- matrix(0, nrow = n, ncol = t)
NoA <- norm(A0, '2'); A0 <- A0/NoA; B0 <- B0/NoA
res_L10 <- L1HardThr(A0, B0, X0, s = 10, maxIter = maxIter0)
```

L1normFun   *L1normFun*

Description

The function aims to compute the l_1 norm.

Usage

```
L1normFun(x)
```

Arguments

x   vector

Details

The L1normFun aims to compute the l_1 norm: $\sum_i^n |x_i|$

Value

The l_1 norm of vector x

Author(s)

Xinlin Hu thompson-xinlin.hu@connect.polyu.hk

Yaohua Hu mayhhu@szu.edu.cn

L1SoftThr   *L1SoftThr - Iterative Soft Thresholding Algorithm based on* $l\_1, 1$
                                norm

Description

The function aims to solve $l_{1,1}$ regularized least squares.

Usage

```
L1SoftThr(A, B, X, s, maxIter = 200)
```

Arguments

A	Gene expression data of transcriptome factors (i.e. feature matrix in machine learning). The dimension of A is m * n.
B	Gene expression data of target genes (i.e. observation matrix in machine learning). The dimension of B is m * t.
X	Gene expression data of Chromatin immunoprecipitation or other matrix (i.e. initial iterative point in machine learning). The dimension of X is n * t.
s	joint sparsity level
maxIter	maximum iteration

Details

The L1SoftThr function aims to solve the problem:

$$\min \|AX - B\|_F^2 + \lambda \|X\|_{1,1}$$

to obtain s-joint sparse solution.

Value

The solution of proximal gradient method with $l_{1,1}$ regularizer.

Author(s)

Xinlin Hu thompson-xinlin.hu@connect.polyu.hk
Yaohua Hu mayhhu@szu.edu.cn

Examples

```
m <- 256; n <- 1024; t <- 5; maxIter0 <- 50
A0 <- matrix(rnorm(m * n), nrow = m, ncol = n)
B0 <- matrix(rnorm(m * t), nrow = m, ncol = t)
X0 <- matrix(0, nrow = n, ncol = t)
NoA <- norm(A0, '2'); A0 <- A0/NoA; B0 <- B0/NoA
res_L11 <- L1SoftThr(A0, B0, X0, s = 10, maxIter = maxIter0)
```

L1twothirdsThr	*L1twothirdsThr - Iterative Thresholding Algorithm based on $l\_1, 2/3$ norm*

Description

The function aims to solve $l_{1,2/3}$ regularized least squares.

Usage

```
L1twothirdsThr(A, B, X, s, maxIter = 200)
```

Arguments

A	Gene expression data of transcriptome factors (i.e. feature matrix in machine learning). The dimension of A is m * n.
B	Gene expression data of target genes (i.e. observation matrix in machine learning). The dimension of B is m * t.
X	Gene expression data of Chromatin immunoprecipitation or other matrix (i.e. initial iterative point in machine learning). The dimension of X is n * t.
s	joint sparsity level
maxIter	maximum iteration

Details

The L1twothirdsThr function aims to solve the problem:

$$\min \|AX - B\|_F^2 + \lambda \|X\|_{1,2/3}$$

to obtain s-joint sparse solution.

Value

The solution of proximal gradient method with $l_{1,2/3}$ regularizer.

Author(s)

Xinlin Hu thompson-xinlin.hu@connect.polyu.hk

Yaohua Hu mayhhu@szu.edu.cn

Examples

```
m <- 256; n <- 1024; t <- 5; maxIter0 <- 50
A0 <- matrix(rnorm(m * n), nrow = m, ncol = n)
B0 <- matrix(rnorm(m * t), nrow = m, ncol = t)
X0 <- matrix(0, nrow = n, ncol = t)
NoA <- norm(A0, '2'); A0 <- A0/NoA; B0 <- B0/NoA
res_L1twothirds <- L1twothirdsThr(A0, B0, X0, s = 10, maxIter = maxIter0)
```

L2HalfThr   *L2HalfThr - Iterative Half Thresholding Algorithm based on $l\_2, 1/2$*
                                   norm

Description

The function aims to solve $l_{2,1/2}$ regularized least squares.

Usage

```
L2HalfThr(A, B, X, s, maxIter = 200)
```

Arguments

A	Gene expression data of transcriptome factors (i.e. feature matrix in machine learning). The dimension of A is m * n.
B	Gene expression data of target genes (i.e. observation matrix in machine learning). The dimension of B is m * t.
X	Gene expression data of Chromatin immunoprecipitation or other matrix (i.e. initial iterative point in machine learning). The dimension of X is n * t.
s	joint sparsity level
maxIter	maximum iteration

Details

The L2HalfThr function aims to solve the problem:

$$\min \|AX - B\|_F^2 + \lambda \|X\|_{2,1/2}$$

to obtain s-joint sparse solution.

Value

The solution of proximal gradient method with $l_{2,1/2}$ regularizer.

Author(s)

Xinlin Hu thompson-xinlin.hu@connect.polyu.hk

Yaohua Hu mayhhu@szu.edu.cn

Examples

```
m <- 256; n <- 1024; t <- 5; maxIter0 <- 50
A0 <- matrix(rnorm(m * n), nrow = m, ncol = n)
B0 <- matrix(rnorm(m * t), nrow = m, ncol = t)
X0 <- matrix(0, nrow = n, ncol = t)
NoA <- norm(A0, '2'); A0 <- A0/NoA; B0 <- B0/NoA
res_L2half <- L2HalfThr(A0, B0, X0, s = 10, maxIter = maxIter0)
```

L2HardThr	*L2HardThr - Iterative Hard Thresholding Algorithm based on $l\_2, 0$ norm*

Description

The function aims to solve $l_{2,0}$ regularized least squares.

Usage

```
L2HardThr(A, B, X, s, maxIter = 200)
```

Arguments

A	Gene expression data of transcriptome factors (i.e. feature matrix in machine learning). The dimension of A is m * n.
B	Gene expression data of target genes (i.e. observation matrix in machine learning). The dimension of B is m * t.
X	Gene expression data of Chromatin immunoprecipitation or other matrix (i.e. initial iterative point in machine learning). The dimension of X is n * t.
s	joint sparsity level
maxIter	maximum iteration

Details

The L2HardThr function aims to solve the problem:

$$\min \|AX - B\|_F^2 + \lambda \|X\|_{2,0}$$

to obtain s-joint sparse solution.

Value

The solution of proximal gradient method with $l_{2,0}$ regularizer.

Author(s)

Xinlin Hu thompson-xinlin.hu@connect.polyu.hk
Yaohua Hu mayhhu@szu.edu.cn

Examples

```
m <- 256; n <- 1024; t <- 5; maxIter0 <- 50
A0 <- matrix(rnorm(m * n), nrow = m, ncol = n)
B0 <- matrix(rnorm(m * t), nrow = m, ncol = t)
X0 <- matrix(0, nrow = n, ncol = t)
NoA <- norm(A0, '2'); A0 <- A0/NoA; B0 <- B0/NoA
res_L20 <- L2HardThr(A0, B0, X0, s = 10, maxIter = maxIter0)
```

L2NewtonThr	*L2NewtonThr - Iterative Thresholding Algorithm based on l_2, q norm with Newton method*

Description

The function aims to solve $l_{2,q}$ regularized least squares, where the proximal optimization subproblems will be solved by Newton method.

Usage

```
L2NewtonThr(A, B, X, s, q, maxIter = 200, innMaxIter = 30, innEps = 1e-06)
```

Arguments

A	Gene expression data of transcriptome factors (i.e. feature matrix in machine learning). The dimension of A is m * n.
B	Gene expression data of target genes (i.e. observation matrix in machine learning). The dimension of B is m * t.
X	Gene expression data of Chromatin immunoprecipitation or other matrix (i.e. initial iterative point in machine learning). The dimension of X is n * t.
s	joint sparsity level
q	value for $l_{2,q}$ norm (i.e., 0 < q < 1)
maxIter	maximum iteration
innMaxIter	maximum iteration in Newton step
innEps	criterion to stop inner iteration

Details

The L2NewtonThr function aims to solve the problem:

$$\min \|AX - B\|_F^2 + \lambda\|X\|_{2,q}$$

to obtain s-joint sparse solution.

Value

The solution of proximal gradient method with $l_{2,q}$ regularizer.

Author(s)

Xinlin Hu thompson-xinlin.hu@connect.polyu.hk

Yaohua Hu mayhhu@szu.edu.cn

Examples

```
m <- 256; n <- 1024; t <- 5; maxIter0 <- 50
A0 <- matrix(rnorm(m * n), nrow = m, ncol = n)
B0 <- matrix(rnorm(m * t), nrow = m, ncol = t)
X0 <- matrix(0, nrow = n, ncol = t)
NoA <- norm(A0, '2'); A0 <- A0/NoA; B0 <- B0/NoA
res_L2q <- L2NewtonThr(A0, B0, X0, s = 10, q = 0.2, maxIter = maxIter0)
```

L2SoftThr	*L2SoftThr - Iterative Soft Thresholding Algorithm based on* $l\_2, 1$ *norm*

Description

The function aims to solve $l_{2,1}$ regularized least squares.

Usage

```
L2SoftThr(A, B, X, s, maxIter = 200)
```

Arguments

A	Gene expression data of transcriptome factors (i.e. feature matrix in machine learning). The dimension of A is m * n.
B	Gene expression data of target genes (i.e. observation matrix in machine learning). The dimension of B is m * t.
X	Gene expression data of Chromatin immunoprecipitation or other matrix (i.e. initial iterative point in machine learning). The dimension of X is n * t.
s	joint sparsity level
maxIter	maximum iteration

Details

The L2SoftThr function aims to solve the problem:

$$\min \|AX - B\|_F^2 + \lambda \|X\|_{2,1}$$

to obtain s-joint sparse solution.

Value

The solution of proximal gradient method with $l_{2,1}$ regularizer.

Author(s)

Xinlin Hu thompson-xinlin.hu@connect.polyu.hk

Yaohua Hu mayhhu@szu.edu.cn

Examples

```
m <- 256; n <- 1024; t <- 5; maxIter0 <- 50
A0 <- matrix(rnorm(m * n), nrow = m, ncol = n)
B0 <- matrix(rnorm(m * t), nrow = m, ncol = t)
X0 <- matrix(0, nrow = n, ncol = t)
NoA <- norm(A0, '2'); A0 <- A0/NoA; B0 <- B0/NoA
res_L21 <- L2SoftThr(A0, B0, X0, s = 10, maxIter = maxIter0)
```

L2twothirdsThr	*L2twothirdsThr - Iterative Thresholding Algorithm based on $l\_2, 2/3$ norm*

Description

The function aims to solve $l_{2,2/3}$ regularized least squares.

Usage

```
L2twothirdsThr(A, B, X, s, maxIter = 200)
```

Arguments

A	Gene expression data of transcriptome factors (i.e. feature matrix in machine learning). The dimension of A is m * n.
B	Gene expression data of target genes (i.e. observation matrix in machine learning). The dimension of B is m * t.
X	Gene expression data of Chromatin immunoprecipitation or other matrix (i.e. initial iterative point in machine learning). The dimension of X is n * t.
s	joint sparsity level
maxIter	maximum iteration

Details

The L2twothirdsThr function aims to solve the problem:

$$\min \|AX - B\|_F^2 + \lambda \|X\|_{2,2/3}$$

to obtain s-joint sparse solution.

Value

The solution of proximal gradient method with $l_{2,2/3}$ regularizer.

Author(s)

Xinlin Hu thompson-xinlin.hu@connect.polyu.hk
Yaohua Hu mayhhu@szu.edu.cn

Examples

```
m <- 256; n <- 1024; t <- 5; maxIter0 <- 50
A0 <- matrix(rnorm(m * n), nrow = m, ncol = n)
B0 <- matrix(rnorm(m * t), nrow = m, ncol = t)
X0 <- matrix(0, nrow = n, ncol = t)
NoA <- norm(A0, '2'); A0 <- A0/NoA; B0 <- B0/NoA
res_L2twothirds <- L2twothirdsThr(A0, B0, X0, s = 10, maxIter = maxIter0)
```

Index