Package ‘DIRMR’

November 26, 2024

Type Package

Version 0.5.0

Date 2024-11-22

Title Distributed Imputation for Random Effects Models with Missing Responses

Description By adding over-relaxation factor to PXEM (Parameter Expanded Expectation Maximization) method, the MOPXEM (Monotonically Overrelaxed Parameter Expanded Expectation Maximization) method is obtained. Compare it with the existing EM (Expectation-Maximization)-like methods. Then, distribute and process five methods and compare them, achieving good performance in convergence speed and result quality.The philosophy of the package is described in Guo G. (2022) <doi:10.1007/s00180-022-01270-z>.

License GPL-2

Imports MASS,lava,mvtnorm

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation no

Author Guangbao Guo [aut, cre] (<https://orcid.org/0000-0002-4115-6218>), Yaping Li [aut]

Maintainer Guangbao Guo <ggb11111111@163.com>

Repository CRAN

Depends R (>= 3.5.0)

Date/Publication 2024-11-26 14:10:02 UTC

Contents

data 2
DECME 3
DEM 4
df1 5
DMCEM 6
DMOPXEM 7

DPXEM 8
ECME 9
EM 10
MCEM 11
MOPXEM 12
PXEM 13

Index **15**

`data` *Hox pupil popularity data*

Description

data data set

Usage

```
data("data")
```

Format

A data frame with 2000 observations on the following 7 variables.

`PUPIL` a numeric vector

`SCHOOL` a numeric vector

`POPULAR` a numeric vector

`SEX` a numeric vector

`TEXP` a numeric vector

`CONST` a numeric vector

`TEACHPOP` a numeric vector

Details

The original, complete dataset was generated by Joop Hox as an example of well-behaved multilevel data set.

Source

The Heart failure data set comes from the R package"mice".

References

Hox, J. J. (2002) Multilevel analysis. Techniques and applications. Mahwah, NJ: Lawrence Erlbaum.

Examples

```
data(data)
## maybe str(data) ; plot(data) ...
```

DECME *DECME*

Description

The DECME algorithm can significantly improve the speed of processing large-scale data sets. It can reduce the algorithm's memory requirements, enabling the algorithm to handle larger data sets.

Usage

```
DECME(data,df1,M,maxiter)
```

Arguments

`data`	The real data sets with missing data used in the method
`df1`	The real data sets used in the method
`M`	The number of Blocks
`maxiter`	The maximum number of iterations

Value

`Y011`	The response variable value after projection for each block
`Yhat`	The estimated response variable value after projection for each block
`Ymean`	The mean of response variable value after projection for each block
`Yhatmean`	The mean of response variable value after projection for each block

Author(s)

Guangbao Guo,Yu Li

Examples

```
set.seed(99)
library(MASS)
library(mvtnorm)
n=50;p=6;q=5;M=2;omega=0.15;ratio=0.1;maxiter=15;nob=round(n-(n*ratio))
dd.start=1;sigma2_e.start=1
X0=matrix(runif(n*p,0,2),ncol=p)
beta=matrix(rnorm(p*1,0,3),nrow=p)
Z0=matrix(runif(n*q,2,3),ncol=q)
e=matrix(rnorm(n*1,0,sigma2_e.start),n,1)
b=matrix(rnorm(q*1,0,1),q,1)
Y0=X0
```

```
df1=data.frame(Y=Y0,X=X0,Z=Z0)
misra=function(data,ratio){
  nob=round(n-(n*ratio))
  data[sample(n,n-nob),1]=NA
  return(data)}
data=misra(data=df1,ratio=0.1)
DECME(data,df1,M=2,maxiter=15)
```

DEM *DEM*

Description

The DEM method is mainly applied to statistical analysis of large-scale datasets, where the dataset is distributed across different computing nodes to process data in parallel and update model parameters.

Usage

```
DEM(data,df1,M,maxiter)
```

Arguments

`data`	The real data sets with missing data used in the method
`df1`	The real data sets used in the method
`M`	The number of Blocks
`maxiter`	The maximum number of iterations

Value

`Y011`	The response variable value after projection for each block
`Yhat`	The estimated response variable value after projection for each block
`Ymean`	The mean of response variable value after projection for each block
`Yhatmean`	The mean of response variable value after projection for each block

Author(s)

Guangbao Guo,Yu Li

Examples

```
set.seed(99)
library(MASS)
library(mvtnorm)
n=50;p=6;q=5;M=2;omega=0.15;ratio=0.1;maxiter=15;nob=round(n-(n*ratio))
dd.start=1;sigma2_e.start=1
X0=matrix(runif(n*p,0,2),ncol=p)
```

```
beta=matrix(rnorm(p*1,0,3),nrow=p)
Z0=matrix(runif(n*q,2,3),ncol=q)
e=matrix(rnorm(n*1,0,sigma2_e.start),n,1)
b=matrix(rnorm(q*1,0,1),q,1)
Y0=X0
df1=data.frame(Y=Y0,X=X0,Z=Z0)
misra=function(data,ratio){
  nob=round(n-(n*ratio))
  data[sample(n,n-nob),1]=NA
  return(data)}
data=misra(data=df1,ratio=0.1)
DEM(data,df1,M=2,maxiter=15)
```

`df1` *Hox pupil popularity data with missing popularity scores*

Description

df1 data set

Usage

```
data("df1")
```

Format

A data frame with 2000 observations on the following 7 variables.

`PUPIL` a numeric vector

`SCHOOL` a numeric vector

`POPULAR` a numeric vector

`SEX` a numeric vector

`TEXP` a numeric vector

`CONST` a numeric vector

`TEACHPOP` a numeric vector

Details

The original, complete dataset was generated by Joop Hox as an example of well-behaved multilevel data set. The distributed data contains missing data in pupil popularity.

Source

The Heart failure data set comes from the R package"mice".

References

Hox, J. J. (2002) Multilevel analysis. Techniques and applications. Mahwah, NJ: Lawrence Erlbaum.

Examples

```
data(df1)
## maybe str(df1) ; plot(df1) ...
```

`DMCEM` *DMCEM*

Description

The DMCEM method uses the sample mean to approximate the integral in step E, rather than performing a single Monte Carlo sampling on the entire dataset. This enables DMCEM to significantly reduce the computation time and memory consumption when processing large datasets, while updating model parameters by calculating the sample mean of each subset.

Usage

```
DMCEM(data,df1,M,maxiter)
```

Arguments

`data`	The real data sets with missing data used in the method
`df1`	The real data sets used in the method
`M`	The number of Blocks
`maxiter`	The maximum number of iterations

Value

`Y011`	The response variable value after projection for each block
`Yhat`	The estimated response variable value after projection for each block
`Ymean`	The mean of response variable value after projection for each block
`Yhatmean`	The mean of response variable value after projection for each block

Author(s)

Guangbao Guo,Yu Li

Examples

```
set.seed(99)
library(MASS)
library(mvtnorm)
n=50;p=6;q=5;M=2;omega=0.15;ratio=0.1;maxiter=15;nob=round(n-(n*ratio))
dd.start=1;sigma2_e.start=1
X0=matrix(runif(n*p,0,2),ncol=p)
beta=matrix(rnorm(p*1,0,3),nrow=p)
Z0=matrix(runif(n*q,2,3),ncol=q)
e=matrix(rnorm(n*1,0,sigma2_e.start),n,1)
b=matrix(rnorm(q*1,0,1),q,1)
Y0=X0
df1=data.frame(Y=Y0,X=X0,Z=Z0)
misra=function(data,ratio){
  nob=round(n-(n*ratio))
  data[sample(n,n-nob),1]=NA
  return(data)}
data=misra(data=df1,ratio=0.1)
DMCEM(data,df1,M=2,maxiter=15)
```

DMOPXEM *DMOPXEM*

Description

In DMOPXEM method, data is allocated to different computing nodes for parallel processing. Each node independently executes the EM algorithm and updates the local model parameters. Then, each node passes the local model parameters to other nodes for the merging and updating of global model parameters.

Usage

```
DMOPXEM(data,df1,M,omega,maxiter)
```

Arguments

`data`	The real data sets with missing data used in the method
`df1`	The real data sets used in the method
`M`	The number of Blocks
`omega`	A variable of this method
`maxiter`	The maximum number of iterations

Value

`Y011`	The response variable value after projection for each block
`Yhat`	The estimated response variable value after projection for each block
`Ymean`	The mean of response variable value after projection for each block
`Yhatmean`	The mean of response variable value after projection for each block

Author(s)

Guangbao Guo,Yu Li

Examples

```
set.seed(99)
library(MASS)
library(mvtnorm)
n=50;p=6;q=5;M=2;omega=0.15;ratio=0.1;maxiter=15;nob=round(n-(n*ratio))
dd.start=1;sigma2_e.start=1
X0=matrix(runif(n*p,0,2),ncol=p)
beta=matrix(rnorm(p*1,0,3),nrow=p)
Z0=matrix(runif(n*q,2,3),ncol=q)
e=matrix(rnorm(n*1,0,sigma2_e.start),n,1)
b=matrix(rnorm(q*1,0,1),q,1)
Y0=X0
df1=data.frame(Y=Y0,X=X0,Z=Z0)
misra=function(data,ratio){
  nob=round(n-(n*ratio))
  data[sample(n,n-nob),1]=NA
  return(data)}
data=misra(data=df1,ratio=0.1)
DMOPXEM(data,df1,M=2,omega=0.15,maxiter=15)
```

DPXEM *DPXEM*

Description

The DPXEM method is mainly used for clustering analysis of large-scale datasets. It distributes the dataset across different computing nodes, processes the data in parallel, and updates model parameters. Through parallel processing, the DPXEM algorithm can significantly improve the speed of processing large-scale datasets.

Usage

```
DPXEM(data,df1,M,maxiter)
```

Arguments

`data`	The real data sets with missing data used in the method
`df1`	The real data sets used in the method
`M`	The number of Blocks
`maxiter`	The maximum number of iterations

Value

Y011	The response variable value after projection for each block
Yhat	The estimated response variable value after projection for each block
Ymean	The mean of response variable value after projection for each block
Yhatmean	The mean of response variable value after projection for each block

Author(s)

Guangbao Guo,Yu Li

Examples

```
set.seed(99)
library(MASS)
library(mvtnorm)
n=50;p=6;q=5;M=2;omega=0.15;ratio=0.1;maxiter=15;nob=round(n-(n*ratio))
dd.start=1;sigma2_e.start=1
X0=matrix(runif(n*p,0,2),ncol=p)
beta=matrix(rnorm(p*1,0,3),nrow=p)
Z0=matrix(runif(n*q,2,3),ncol=q)
e=matrix(rnorm(n*1,0,sigma2_e.start),n,1)
b=matrix(rnorm(q*1,0,1),q,1)
Y0=X0
df1=data.frame(Y=Y0,X=X0,Z=Z0)
misra=function(data,ratio){
  nob=round(n-(n*ratio))
  data[sample(n,n-nob),1]=NA
  return(data)}
data=misra(data=df1,ratio=0.1)
DPXEM(data,df1,M=2,maxiter=15)
```

ECME *ECME*

Description

The ECME method calculates the conditional expectation of each hidden variable based on known data and current parameter estimates. Then, based on the known data, the conditional expectation of the hidden variables, and the current parameter estimates, the likelihood function is maximized to update the parameter estimates.

Usage

```
ECME(data,df1,maxiter)
```

Arguments

data	The real data sets with missing data used in the method
df1	The real data sets used in the method
maxiter	The maximum number of iterations

Value

Y01	The response variable value after projection
Yhat	The estimated response variable value after projection

Author(s)

Guangbao Guo,Yu Li

Examples

```
set.seed(99)
library(MASS)
library(mvtnorm)
n=50;p=6;q=5;M=2;omega=0.15;ratio=0.1;maxiter=15;nob=round(n-(n*ratio))
dd.start=1;sigma2_e.start=1
X0=matrix(runif(n*p,0,2),ncol=p)
beta=matrix(rnorm(p*1,0,3),nrow=p)
Z0=matrix(runif(n*q,2,3),ncol=q)
e=matrix(rnorm(n*1,0,sigma2_e.start),n,1)
b=matrix(rnorm(q*1,0,1),q,1)
Y0=X0
df1=data.frame(Y=Y0,X=X0,Z=Z0)
misra=function(data,ratio){
  nob=round(n-(n*ratio))
  data[sample(n,n-nob),1]=NA
  return(data)}
data=misra(data=df1,ratio=0.1)
ECME(data,df1,maxiter=15)
```

EM *EM*

Description

The EM method is an iterative algorithm used for maximum likelihood estimation or maximum posterior probability estimation of parameters in probabilistic models with hidden variables. It is essentially a method for estimating parameters, based on existing sample data, to estimate parameter values that are consistent with the model.

Usage

```
EM(data,df1,maxiter)
```

Arguments

`data`	The real data sets with missing data used in the method
`df1`	The real data sets used in the method
`maxiter`	The maximum number of iterations

Value

`Y01`	The response variable value after projection
`Yhat`	The estimated response variable value after projection

Author(s)

Guangbao Guo,Yu Li

Examples

```
set.seed(99)
library(MASS)
library(mvtnorm)
n=50;p=6;q=5;M=2;omega=0.15;ratio=0.1;maxiter=15;nob=round(n-(n*ratio))
dd.start=1;sigma2_e.start=1
X0=matrix(runif(n*p,0,2),ncol=p)
beta=matrix(rnorm(p*1,0,3),nrow=p)
Z0=matrix(runif(n*q,2,3),ncol=q)
e=matrix(rnorm(n*1,0,sigma2_e.start),n,1)
b=matrix(rnorm(q*1,0,1),q,1)
Y0=X0
df1=data.frame(Y=Y0,X=X0,Z=Z0)
misra=function(data,ratio){
  nob=round(n-(n*ratio))
  data[sample(n,n-nob),1]=NA
  return(data)}
data=misra(data=df1,ratio=0.1)
EM(data,df1,maxiter=15)
```

MCEM   *MCEM*

Description

The MCEM method is an algorithm that utilizes the Monte Carlo method to solve the difficult E-step integral in the EM algorithm. It avoids complex numerical integration calculations by converting the integral in the E-step into a numerical integral.

Usage

```
MCEM(data, df1, maxiter)
```

Arguments

`data`	The real data sets with missing data used in the method
`df1`	The real data sets used in the method
`maxiter`	The maximum number of iterations

Value

`Y01`	The response variable value after projection
`Yhat`	The estimated response variable value after projection

Author(s)

Guangbao Guo,Yu Li

Examples

```
set.seed(99)
library(MASS)
library(mvtnorm)
n=50;p=6;q=5;M=2;omega=0.15;ratio=0.1;maxiter=15;nob=round(n-(n*ratio))
dd.start=1;sigma2_e.start=1
X0=matrix(runif(n*p,0,2),ncol=p)
beta=matrix(rnorm(p*1,0,3),nrow=p)
Z0=matrix(runif(n*q,2,3),ncol=q)
e=matrix(rnorm(n*1,0,sigma2_e.start),n,1)
b=matrix(rnorm(q*1,0,1),q,1)
Y0=X0
df1=data.frame(Y=Y0,X=X0,Z=Z0)
misra=function(data,ratio){
  nob=round(n-(n*ratio))
  data[sample(n,n-nob),1]=NA
  return(data)}
data=misra(data=df1,ratio=0.1)
MCEM(data,df1,maxiter=15)
```

`MOPXEM` *MOPXEM*

Description

The MOPXEM method is an improved EM algorithm that combines the monotonic super-relaxation strategy with the PXEM strategy. The main idea of the MOPXEM method is to accelerate the EM algorithm using the ULS strategy, while simultaneously expanding and optimizing the model parameters using the PX-EM strategy.

Usage

```
MOPXEM(data,df1,omega,maxiter)
```

Arguments

`data`	The real data sets with missing data used in the method
`df1`	The real data sets used in the method
`omega`	A variable of this method
`maxiter`	The maximum number of iterations

Value

`Y01`	The response variable value after projection
`Yhat`	The estimated response variable value after projection

Author(s)

Guangbao Guo,Yu Li

Examples

```
set.seed(99)
library(MASS)
library(mvtnorm)
n=50;p=6;q=5;M=2;omega=0.15;ratio=0.1;maxiter=15;nob=round(n-(n*ratio))
dd.start=1;sigma2_e.start=1
X0=matrix(runif(n*p,0,2),ncol=p)
beta=matrix(rnorm(p*1,0,3),nrow=p)
Z0=matrix(runif(n*q,2,3),ncol=q)
e=matrix(rnorm(n*1,0,sigma2_e.start),n,1)
b=matrix(rnorm(q*1,0,1),q,1)
Y0=X0
df1=data.frame(Y=Y0,X=X0,Z=Z0)
misra=function(data,ratio){
  nob=round(n-(n*ratio))
  data[sample(n,n-nob),1]=NA
  return(data)}
data=misra(data=df1,ratio=0.1)
MOPXEM(data,df1,omega=0.15,maxiter=15)
```

`PXEM` *PXEM*

Description

The PXEM method is an algorithm that accelerates the convergence rate of the EM algorithm. By introducing additional parameters, improving the model, and expanding it, it has better parameter estimation results compared to the EM method.

Usage

```
PXEM(data,df1,maxiter)
```

Arguments

`data`	The real data sets with missing data used in the method
`df1`	The real data sets used in the method
`maxiter`	The maximum number of iterations

Value

`Y01`	The response variable value after projection
`Yhat`	The estimated response variable value after projection

Author(s)

Guangbao Guo,Yu Li

Examples

```
set.seed(99)
library(MASS)
library(mvtnorm)
n=50;p=6;q=5;M=2;omega=0.15;ratio=0.1;maxiter=15;nob=round(n-(n*ratio))
dd.start=1;sigma2_e.start=1
X0=matrix(runif(n*p,0,2),ncol=p)
beta=matrix(rnorm(p*1,0,3),nrow=p)
Z0=matrix(runif(n*q,2,3),ncol=q)
e=matrix(rnorm(n*1,0,sigma2_e.start),n,1)
b=matrix(rnorm(q*1,0,1),q,1)
Y0=X0
df1=data.frame(Y=Y0,X=X0,Z=Z0)
misra=function(data,ratio){
  nob=round(n-(n*ratio))
  data[sample(n,n-nob),1]=NA
  return(data)}
data=misra(data=df1,ratio=0.1)
PXEM(data,df1,maxiter=15)
```

Index

∗ **datasets**
  data, 2
  df1, 5

data, 2
DECME, 3
DEM, 4
df1, 5
DMCEM, 6
DMOPXEM, 7
DPXEM, 8

ECME, 9
EM, 10

MCEM, 11
MOPXEM, 12

PXEM, 13